Actes des 18es Journées d'Intelligence Artificielle Fondamentale et des 19es Journées Francophones sur la Planification, la Décision et l'Apprentissage pour la conduite de systèmes

Jean-Guy Mailly, François Schwarzentruber, Anaelle Wilczynski

AfIA

Association française
pour l'Intelligence Artificielle

JIAF-JFPDA

Journées d'Intelligence Artificielle Fondamentale
et Journées Francophones sur
la Planification, la Décision et l'Apprentissage
pour la conduite de systèmes

PFIA 2024

Table des matières

Éditorial

Journées d'Intelligence Artificielle Fondamentale et Journées Francophones sur la Planification, la Décision et l'Apprentissage pour la conduite de systèmes

Les Journées d'Intelligence Artificielle Fondamentale (JIAF) constituent un rendez-vous annuel de la communauté francophone travaillant sur l'Intelligence Artificielle Fondamentale et la Planification. Depuis l'édition 2023, JIAF intègre les Journées Francophones sur la Planification, la Décision et l'Apprentissage pour la conduite de systèmes (JFPDA). La conférence JIAF est soutenue par le Collège Représentation et Raisonnement de l'AFIA.

Les thématiques de recherche de JIAF sont relatives aux méthodes et outils fondamentaux de l'Intelligence Artificielle : modèles de représentation des informations, méthodes de raisonnements sur ces informations, méthodes de codage des informations et d'algorithmes de traitement efficaces, modélisation formelle de l'interaction, analyse de la prise de décision multi-agents, dans l'incertain ou séquentielle. Les journées sont composées d'exposés de synthèse, permettant à la communauté de découvrir des thématiques connexes au travers d'exposés de spécialistes, et de communications sélectionnées par le comité de programme, qui était composé de 30 personnes et animé par Jean-Guy Mailly (IRIT, Université de Toulouse, UT Capitole), François Schwarzentruber (IRISA, ENS Rennes, Université de Rennes) et Anaëlle Wilczynski (MICS, CentraleSupélec, Université Paris-Saclay).

Lors des 18èmes Journées d'Intelligence Artificielle Fondamentale (JIAF 2024), nous avons reçu 19 soumissions, dont 18 ont été acceptées pour présentation lors de la conférence et inclusion dans les actes. Les sujets traités balayent un large spectre de domaines de l'intelligence artificielle (raisonnement par analogie, argumentation, choix social, explicabilité, jeux, logique, planification). Nous avons également le plaisir d'accueillir trois exposés invités. Simon Parsons (University of Lincoln) présentera ses travaux sur l'utilisation de l'IA pour rendre l'agriculture plus durable. Tristan Cazenave (LAMSADE, Université Paris-Dauphine) discutera de comment utiliser l'IA pour améliorer l'IA. Enfin, Laurent Perron (Google Research, invité commun avec le GDR RADIA) parlera du solveur CP-SAT dans la suite logicielle OR-Tools.

Jean-Guy Mailly, François Schwarzentruber, Anaëlle Wilczynski

Comité de programme

Présidence

— Jean-Guy Mailly (IRIT, Université de Toulouse, UT Capitole) ;
— François Schwarzentruber (IRISA, ENS Rennes, Université de Rennes) ;
— Anaëlle Wilczynski (MICS, CentraleSupélec, Université Paris-Saclay).

Membres

— Francesco Belardinelli (Imperial College London) ;
— Aurélie Beynier (LIP6, Sorbonne Université) ;
— Elise Bonzon (LIPADE, Université Paris Cité) ;
— Olivier Buffet (INRIA / LORIA) ;
— Martin Cooper (IRIT, Université de Toulouse, UT3) ;
— Célia da Costa Pereira (Université Côte d'Azur) ;
— Sylvie Coste-Marquis (CRIL, Université d'Artois) ;
— Tiago de Lima (Université d'Artois, CRIL CNRS) ;
— Jilles Dibangoye (INSA Lyon, CITI lab, INRIA) ;
— Sylvie Doutre (IRIT, Université de Toulouse, UT Capitole) ;
— Florence Dupin de Saint-Cyr (IRIT, Université de Toulouse, UT3) ;
— Alain Dutech (Loria - Inria) ;
— Hugo Gilbert (Lamsade - Université Paris Dauphine) ;
— Andreas Herzig (IRIT, Université de Toulouse, CNRS) ;
— Sébastien Konieczny (CRIL - CNRS) ;
— Raida Ktari (Aix-Marseille Université) ;
— Jérôme Lang (CNRS, LAMSADE, Université Paris-Dauphine) ;
— Daniel Le Berre (CRIL, Université d'Artois) ;
— Jean Lieber (LORIA - INRIA Lorraine) ;
— Pierre Marquis (CRIL, U. Artois & CNRS - Institut Universitaire de France) ;
— Amedeo Napoli (LORIA Nancy, CNRS - Inria - Université de Lorraine) ;
— Arianna Novaro (Centre d'Economie de la Sorbonne (CES), Université Paris 1 Panthéon-Sorbonne) ;
— Damien Pellier (Laboratoire d'Informatique de Grenoble) ;
— Laurent Perrussel (IRIT, Université de Toulouse, UT Capitole) ;
— Sophie Pinchinat (IRISA Rennes) ;
— Julien Rossit (Université Paris Cité, LIPADE) ;
— Stephanie Roussel (ONERA) ;
— Régis Sabbadin (INRA-UBIA) ;
— Vincent Thomas (LORIA) ;
— Bruno Zanuttini (GREYC, Université de Normandie).

Session 1 : Logique propositionnelle

Actes JIAF 2024

Effacement des croyances en logique propositionnelle

Nadia Creignou[1] **Raïda Ktari**[1,2] **Odile Papini**[1]

[1] Aix Marseille Univ, CNRS, LIS, Marseille, France
[2] Université de Sfax, ISIMS, CES-Lab LR11ES49, Sfax, Tunisie

{nadia.creignou, raida.ktari, odile.papini}@univ-amu.fr

Résumé

L'un des thèmes importants de la représentation des connaissances et du raisonnement en intelligence artificielle est le changement de croyances. Dans le cadre logique l'approche AGM s'est imposée comme un standard et différentes opérations de changement de croyances ont été envisagées. Alors que la révision, la contraction, et la mise à jour ont donné lieu à de nombreux travaux, l'effacement a suscité jusqu'à présent moins d'intérêt. L'effacement est à la contraction ce que la mise à jour est à la révision. L'article porte sur l'étude de l'effacement dans le cadre de la logique propositionnelle. Il prolonge l'approche de Katsuno et Mendelzon par des postulats supplémentaires capturant la minimalité du changement et propose deux théorèmes de représentation pour les opérateurs d'effacement, l'un en termes de préordres totaux sur les interprétations, l'autre en termes de préordres partiels sur les interprétations. Enfin, il complète les travaux de Caridroit, Konieczny et Marquis pour la contraction en proposant un nouveau théorème de représentation pour les opérateurs de contraction en termes de préordres partiels sur les interprétations.

Abstract

Belief change is an important topic of knowledge representation and reasoning in artificial intelligence. Within the logical framework, the AGM approach has become a standard and various belief change operations have been considered. While revision, contraction and updating have given rise to a great deal of work, erasure has so far attracted less interest. Erasure is to contraction what update is to revision.This article deals with the study of erasure within the framework of propositional logic. It extends Katsuno and Mendelzon's approach with additional postulates capturing the minimality of change and proposes two representation theorems for erasure operators, one in terms of total preorders on interpretations, the other in terms of partial preorders on interpretations. Finally, it completes the work of Caridroit, Konieczny and Marquis for contraction by proposing a new representation theorem for contraction operators in terms of partial preorders on interpretations.

1 Introduction

Le changement de croyances est une thématique importante dans le domaine de la représentation des connaissances et du raisonnement en intelligence artificielle. Les approches logiques ont donné lieux à de nombreux travaux depuis l'approche AGM qui s'est imposée comme un standard [1, 10]. Différentes opérations de changement de croyances ont été étudiées, révision [1, 10, 6, 19], mise à jour [9, 20, 14, 11], contraction [2, 10, 4] [1].

Afin de caractériser différentes approches sémantiques de la révision dans un même cadre, Katsuno et Mendelzon [13] ont restreint l'approche AGM à la logique propositionnelle. Ils ont reformulés les postulats AGM, et proposé deux théorèmes de représentation, l'un en termes de préordres totaux sur les interprétations et l'autre en termes de préordres partiels sur les interprétations. Ce cadre sémantique a permis de clairement distinguer les opérations de révision et de mise à jour [14]. D'un point de vue sémantique, les opérateurs de révision reposent sur une minimisation globale, ils sélectionnent les modèles de la nouvelle information les plus "proches" des modéles de l'ensemble de croyances intial, alors que les opérateurs de mise à jour reposent sur une minimisation locale, ils sélectionnent, pour chaque modèle de l'ensemble de croyances intial, les modèles de la nouvelle information les plus "proches" de celui-ci.

Plus récemment, s'inspirant de Katsuno et Mendelzon, Caridroit, Konieczny et Marquis ont reformulé les postulats AGM de la contraction dans le cadre propositionnel et ont proposé un théorème de représentation pour les opérateurs de contraction en termes de préordres totaux sur les interprétations [4].

Dans [14], Katsuno et Mendelzon définissent une nouvelle opération de changement de croyances, qu'ils

1. Pour plus de détails sur les différentes approches proposées pour ces opérations consulter par exemple [17] ou [18].

nomment *effacement*, qui est à la contraction ce que la mise à jour est à la révision. Effacer une information repose sur une minimisation locale, c'est à dire que les opérateurs d'effacement sélectionnent les modèles de l'ensemble de croyances initiales ainsi que les modèles qui falsifient l'information à effacer, les plus "proches" de chaque modèle de l'ensemble de croyances initiales. Katsuno et Mendelzon proposent une définition sémantique de l'effacement ainsi que des postulats basiques que cette opération doit satisfaire.

L'effacement a été étudié comme beaucoup d'opérations de changement de croyances dans le cadre de fragments propositionnels [5], cependant les études réalisées sur l'effacement restaient, jusqu'à présent, incomplètes car elles ne proposaient qu'un ensemble de postulats basiques pour cette opération insuffisant pour conduire à un théorème de représentation.

L'objet de cet article est de compléter l'étude de l'effacement en proposant :

— des postulats supplémentaires qui capturent le changement minimal de l'opération d'effacement,
— un théorème de représentation pour les opérateurs d'effacement en termes de préordres totaux,
— un théorème de représentation pour les opérateurs d'effacement en termes de préordres partiels.

Par ailleurs, l'article complète également les travaux de Caridroit, Konieczny et Marquis sur la contraction [4] en présentant un théorème de représentation pour les opérateurs de contraction en termes de préordres partiels qui, à ce jour, manquait.

Ainsi, ce travail met une touche finale au panorama de l'étude sémantique des opérateurs de changement de croyances pour la contraction et l'effacement.

L'article se décline comme suit. Après quelques préliminaires en Section 2 qui rappellent quelques notions utiles à la suite de l'article, la Section 3 présente un rapide état de l'art sur les opérations de changement de croyances, révision, contraction et mise à jour. La Section 4 présente la contribution de l'article sur l'effacement (postulats supplémentaires, théorèmes de représentation) ainsi que le théorème de représentation pour les opérateurs de contraction en termes de préordres partiels avant de conclure en Section 5.

2 Préliminaires

Logique propositionnelle. Nous considérons $\mathcal{L}$ le langage de la logique propositionnelle défini sur un ensemble infini dénombrable de variables propositionnelles (ou atomes), noté $\mathcal{V}$, muni des connecteurs logiques usuels $\vee$, $\wedge$, $\neg$ et des constantes $\top$, $\bot$. Les lettres minuscules romaines $a, b, c, \ldots$ désignent les atomes, les lettres grecques $\alpha, \beta, \varphi, \ldots$ désignent les formules bien formées et les lettres majuscules désignent les ensembles d'atomes ou les ensembles de formules bien formées. Pour toute formule φ de $\mathcal{L}$, on note par $\mathrm{Var}(\varphi)$ l'ensemble des atomes apparaissant dans la formule φ.

Une interprétation est une fonction $w : \mathcal{V} \to \{0, 1\}$ qui associe à chaque atome une valeur de vérité, 0 pour faux ou 1 pour vrai. ($w(\bot) = 0$ et $w(\top) = 1$). Soit $\mathcal{U} \subseteq \mathcal{V}$ un ensemble fini d'atomes, une *interprétation* sur $\mathcal{U}$ est représentée par un ensemble $w \subseteq \mathcal{U}$ d'atomes évalués à vrai ou par son vecteur caractéristique correspondant de longueur $|\mathcal{U}|$. Une interprétation qui satisfait une formule φ est appelée *modèle* de φ et nous notons $\mathrm{Mod}(\varphi)$ l'ensemble des modèles de φ. De plus, $\varphi \models \psi$ si $\mathrm{Mod}(\varphi) \subseteq \mathrm{Mod}(\psi)$ et $\psi \equiv \varphi$ (ψ et φ sont équivalentes) si $\mathrm{Mod}(\psi) = \mathrm{Mod}(\varphi)$.

Une formule ψ est dite *complète* si pour toute formule $\mu \in \mathcal{L}$ soit $\psi \models \mu$ soit $\psi \models \neg\mu$. D'une manière équivalente, une formule satisfaisable est complète si et seulement si elle admet exactement un modèle.

Soit $\mathcal{I}$ un ensemble, un préordre est une relation binaire de $\mathcal{I} \times \mathcal{I}$, noté $\leq$, qui est réflexive et transitive. Le préordre strict associé à $\leq$ est défini par $w < w'$ si $w \leq w'$ et $w' \not\leq w$. Une relation d'équivalence induite par $\leq$, notée $\approx$, est définie par $w \approx w'$ si $w \leq w'$ et $w' \leq w$. Un préordre est dit total si pour tout w, w' de $\mathcal{I}$, soit $w \leq w'$ ou $w' \leq w$. L'ensemble des éléments minimaux de $\mathcal{I}$ selon le préordre $\leq$, noté $Min_{\leq}(\mathcal{I})$ est défini par $Min_{\leq}(\mathcal{I}) = \{w \in \mathcal{I} \mid \text{il n'existe pas } w' \in \mathcal{I} \text{ tel que } w' < w\}$.

Nous rappelons deux distances entre un modèle w et une formule. L'une basée sur la cardinalité, notée $|\Delta|_w^{min}(\mu)$, comme étant le nombre minimal de variables propositionnelles pour lesquelles w et les modèles de μ diffèrent. Formellement, $|\Delta|_w^{min}(\mu) = min\{|w\Delta w'| : w' \in \mathrm{Mod}(\mu)\}$. L'autre basée sur l'inclusion ensembliste, notée $\Delta_w^{min}(\mu)$, comme étant les ensembles minimaux, au sens de l'inclusion, de variables propositionnelles pour lesquelles w et les modèles de μ diffèrent. Formellement, $\Delta_w^{min}(\mu) = min_{\subseteq}(\{w\Delta w' : w' \in \mathrm{Mod}(\mu)\})$.

3 Etat de l'art

3.1 Révision des croyances

La révision de croyances est une opération qui permet d'accepter une nouvelle information, en préservant la cohérence tout en modifiant le moins possible les croyances initiales. Plus formellement, une opération de révision, est une fonction, notée $\circ$, de $\mathcal{L} \times \mathcal{L}$ vers $\mathcal{L}$, qui à partir d'une formule ψ (les croyances initiales d'un agent), et d'une formule μ (la nouvelle information), fournit une nouvelle formule, $\psi \circ \mu$ (les croyances de l'agent révisées).

Alchourròn, Gärdenfors et Makinson [1] ont étudié la révision lorsque les croyances d'un agent sont représentées par une théorie (ensemble de formules déductivement clos) et ont proposé un ensemble de postulats, appelés *postulats AGM*, que devrait satisfaire tout opérateur de révision "rationnel". Dans le cadre propositionnel, Katsuno et Men-

delzon [12] ont reformulé les postulats AGM lorsqu'une théorie est représenté par les modèles d'une formule. Dans ce cadre, la révision de ψ par μ revient à rechercher les modèles de μ les plus proches de ceux de ψ. Ces postulats connus sous le nom de postulats KM sont les suivants :

Soit $\psi, \psi_1, \psi_2, \mu, \mu_1, \mu_2 \in \mathcal{L}$.

(R1) $\psi \circ \mu \models \mu$.
(R2) Si $(\psi \wedge \mu)$ est satisfaisable, alors $\psi \circ \mu \equiv \psi \wedge \mu$.
(R3) Si μ est satisfaisable, alors $\psi \circ \mu$ est satisfaisable.
(R4) Si $\psi_1 \equiv \psi_2$ et $\mu_1 \equiv \mu_2$, alors $\psi_1 \circ \mu_1 \equiv \psi_2 \circ \mu_2$.
(R5) $(\psi \circ \mu_1) \wedge \mu_2 \models \psi \circ (\mu_1 \wedge \mu_2)$.
(R6) Si $(\psi \circ \mu_1) \wedge \mu_2$ est satisfaisable alors $\psi \circ (\mu_1 \wedge \mu_2) \models (\psi \circ \mu_1) \wedge \mu_2$.
(R7) Si $(\psi \circ \mu_1) \models \mu_2$ et $(\psi \circ \mu_2) \models \mu_1$, alors $\psi \circ \mu_1 \equiv \psi \circ \mu_2$.
(R8) $(\psi \circ \mu_1) \wedge (\psi \circ \mu_2) \models \psi \circ (\mu_1 \vee \mu_2)$.

Une description détaillée de ces postulats est dans [13].

Katsuno et Mendelzon [12, 13] ont montré que l'opération de révision $\circ$, selon l'ensemble de postulats KM qu'elle satisfait, peut se traduire par un préordre total sur les interprétations ou par un préordre partiel sur les interprétations. Plus formellement, une *affectation fidèle* est une fonction qui associe à chaque formule ψ un préordre $\leq_\psi$ sur les interprétations tel que :

1. Si $w, w' \in \text{Mod}(\psi)$ alors $w \not<_\psi w'$
2. Si $w \in \text{Mod}(\psi)$ et $w' \notin \text{Mod}(\psi)$ alors $w <_\psi w'$
3. Si $\psi \equiv \psi'$, alors $\leq_\psi = \leq_{\psi'}$

Ils ont fourni le théorème de représentation suivant :

Theorem 1 [13]

— *Un opérateur de révision $\circ$ satisfait tous les postulats* (R1)–(R6) *si et seulement si il existe une affectation fidèle qui associe à chaque formule ψ un préordre total $\leq_\psi$ tel que* $\text{Mod}(\psi \circ \mu) = \mathit{Min}_{\leq_\psi}(\text{Mod}(\mu))$.

— *Un opérateur de révision $\circ$ satisfait les postulats* $(R1)$–$(R5)$, $(R7)$ *and* $(R8)$ *si et seulement si il existe une affectation fidèle qui associe à chaque formule ψ un préordre partiel $\leq_\psi$ tel que* $\text{Mod}(\psi \circ \mu) = \mathit{Min}_{\leq_\psi}(\text{Mod}(\mu))$.

Il existe de nombreux opérateurs de révision dans la littérature, nous nous limitons ici à présenter l'opérateur de révision de Dalal, noté $(\circ_D)$, [6], et l'opérateur de révision de Satoh, noté $(\circ_S)$, [19]. La proximité pour l'opérateur Dalal se mesure en terme de cardinalité de la différence symétrique entre modèles,

$$\text{Mod}(\psi \circ_D \mu) = \{m \in \text{Mod}(\mu) : \exists m' \in \text{Mod}(\psi) \text{ tel que } |m \Delta m'| = |\Delta|_m^{min}(\mu)\}.$$

Tandis que la proximité pour l'opérateur Satoh se mesure en terme d'inclusion ensembliste de la différence symétrique entre modèles.

$$\text{Mod}(\psi \circ_S \mu) = \{m \in \text{Mod}(\mu) : \exists m' \in \text{Mod}(\psi) \text{ tel que } m \Delta m' \in \Delta_m^{min}(\mu)\}.$$

Notons que l'opérateur de révision de Dalal satisfait (R1)-(R6) [8, 13] tandis que l'opérateur de révision de Satoh satisfait (R1)-(R5), (R7) et (R8) [13].

3.2 Contraction des croyances

La contraction de croyances est une opération qui permet de retirer une croyance de l'ensemble des croyances intitiales en respectant le principe de changement minimal tout en préservant la cohérence. Formellelement, une opération de contraction est une fonction, notée $-$, de $\mathcal{L} \times \mathcal{L}$ vers $\mathcal{L}$ qui à partir d'une formule ψ (les croyances initiales d'un agent) et d'une formule μ (la croyance à retirer), fournit $\psi - \mu$ (les croyances de l'agent contractées).

Alchourrón, Gärdenfors et Makinson [1] ont étudié la contraction lorsque les croyances d'un agent sont représentées par une théorie et ont proposé des postulats que tout opérateur de contraction devrait satisfaire. Katsuno et Mendelzon [14] ont proposé une reformulation de certains postulats AGM pour la contraction lorsqu'une théorie est représenté par les modèles d'une formule. Plus récemment, Caridroit, Konieczny et Marquis [4] ont revisité les postulats de la contraction, en proposant, en particulier, des postulats supplémentaires caractérisant le principe de changement minimal.

Soit $\psi, \psi_1, \psi_2, \mu, \mu_1, \mu_2 \in \mathcal{L}$.

(C1) $\psi \models \psi - \mu$.
(C2) Si $\psi \not\models \mu$, alors $\psi - \mu \models \psi$.
(C3) Si $\psi - \mu \models \mu$, alors $\models \mu$.
(C4) Si $\psi_1 \equiv \psi_2$ et $\mu_1 \equiv \mu_2$, alors $\psi_1 - \mu_1 \equiv \psi_2 - \mu_2$.
(C5) Si $\psi \models \mu$, alors $(\psi - \mu) \wedge \mu \models \psi$.
(C6) $\psi - (\mu_1 \wedge \mu_2) \models (\psi - \mu_1) \vee (\psi - \mu_2)$.
(C7) Si $\psi - (\mu_1 \wedge \mu_2) \not\models \mu_1$, alors $\psi - \mu_1 \models \psi - (\mu_1 \wedge \mu_2)$.

Une description détaillée de ces postulats est dans [14] et [4]. [2]

Caridroit, Konieczny et Marquis [4] ont montré qu'une opération de contraction – satisfaisant l'ensemble des postulats qu'ils ont proposés peut se traduire par un préordre total sur les interprétations.

2. Dans un souci de cohérence nous avons adopté la numérotation des postulats initialement proposée par Katsuno et Mendelzon, et qui diffère de celle proposée par Caridroit, Konieczny et Marquis, les postulats (C4) et (C5) étant inversés.

Theorem 2 [3] *Un opérateur de contraction − satisfait les postulats* (C1)–(C7) *si et seulement si il existe une affectation fidèle qui associe à chaque formule ψ un préordre total $\leq_\psi$ tel que* $\text{Mod}(\psi - \mu) = \text{Mod}(\psi) \cup \textit{Min}_{\leq_\psi}(\text{Mod}(\neg\mu))$.

Les opérations de révision et de contraction sont extrêmement liées comme le reflètent les identités de Levi et Harper :

$$\psi - \mu \equiv \psi \vee (\psi \circ \neg\mu) \text{ (identité de Harper)}$$

$$\psi \circ \mu \equiv (\psi - \neg\mu) \wedge \mu) \text{ (identité de Lévi)}$$

Ces identités ont permis à Caridroit, Konieczny et Marquis de montrer les correspondances entre les postulats de la révision et ceux de la contraction [4].

Theorem 3 [4]

- *Si l'opérateur de contraction − satisfait* (C1)–(C5) *alors son analogue pour la révision $\circ$ défini par l'identité de Lévi satisfait* (R1)–(R4). *De plus si* (C6) *est satisfait, alors* (R5) *est satisfait pour la révision. Enfin, si le postulat* (C7) *est satisfait, alors le postulat* (R6) *l'est aussi.*
- *Si l'opérateur de révision $\circ$ satisfait* (R1)–(R4) *alors son analogue pour la contraction − défini par l'identité de Harper satisfait* (C1)–(C5). *De plus si* (R5) *est satisfait, alors* (C6) *est satisfait pour la contraction. Enfin, si* (R6) *est satisfait, alors* (C7) *l'est aussi.*

Cela conduit naturellement à définir des opérateurs de contraction à partir d'opérateurs de révision. L'opérateur de contraction $-_D$, analogue à l'opérateur de révision de Dalal $\circ_D$, et l'opérateur de contraction $-_S$, analogue à l'opérateur de révision de Satoh $\circ_S$, sont respectivement définis comme suit :

$$\text{Mod}(\psi -_D \mu) = \text{Mod}(\psi) \cup \text{Mod}(\psi \circ_D \neg\mu).$$

$$\text{Mod}(\psi -_S \mu) = \text{Mod}(\psi) \cup \text{Mod}(\psi \circ_S \neg\mu).$$

Notons que l'opérateur $-_D$ satisfait (C1)–(C7) tandis que l'opérateur $-_S$ satisfait (C1)–(C6), mais viole (C7) [4].

3.3 Mise à jour des croyances

Keller et Winslett [15], puis Katsuno et Mendelzon [14] ont mis en évidence les differences entre révision et mise à jour. La mise à jour de croyances est une opération qui incorpore une nouvelle information reflétant un changement de l'environnement, en préservant la cohérence tout en modifiant le moins possible les croyances initiales. Formellement, une opération de la mise à jour est une fonction, notée $\diamond$, de $\mathcal{L} \times \mathcal{L}$ vers $\mathcal{L}$ qui à partir d'une formule ψ (les croyances initiales d'un agent) et d'une formule μ (la nouvelle information reflétant le changement de l'environnement), fournit $\psi \diamond \mu$ (les croyances de l'agent mises à jour). Les postulats KM [14] de la mise à jour sont les suivants :

Soit $\psi, \psi_1, \psi_2, \mu, \mu_1, \mu_2 \in \mathcal{L}$.

(U1) $\psi \diamond \mu \models \mu$.
(U2) Si $\psi \models \mu$, alors $\psi \diamond \mu \equiv \psi$.
(U3) Si ψ et μ sont satisfaisables, alors $\psi \diamond \mu$ l'est aussi.
(U4) Si $\psi_1 \equiv \psi_2$ et $\mu_1 \equiv \mu_2$, alors $\psi_1 \diamond \mu_1 \equiv \psi_2 \diamond \mu_2$.
(U5) $(\psi \diamond \mu) \wedge \phi \models \psi \diamond (\mu \wedge \phi)$.
(U6) Si $(\psi \diamond \mu_1) \models \mu_2$ et $(\psi \diamond \mu_2) \models \mu_1$, alors $\psi \diamond \mu_1 \equiv \psi \diamond \mu_2$.
(U7) Si ψ est complète, alors $(\psi \diamond \mu_1) \wedge (\psi \diamond \mu_2) \models \psi \diamond (\mu_1 \vee \mu_2)$.
(U8) $(\psi_1 \vee \psi_2) \diamond \mu \equiv (\psi_1 \diamond \mu) \vee (\psi_2 \diamond \mu)$.
(U9) Si ψ est complète et $(\psi \diamond \mu) \wedge \phi$ est satisfaisable, alors $\psi \diamond (\mu \wedge \phi) \models (\psi \diamond \mu) \wedge \phi$.

Une description détaillée de ces postulats est dans [14].

Katsuno et Mendelzon [14] ont montré que l'opération de mise à jour $\diamond$, selon l'ensemble de postulats KM qu'elle satisfait, peut se traduire par un préordre total sur les interprétations ou par un préordre partiel sur les interprétations [14]. Plus formellement, une *affectation fidèle ponctuelle* est une fonction qui associe à chaque interprétation w un préordre $\leq_w$ sur les interprétations, tel que pour toute interpretation w', si $w' \neq w$ alors $w <_w w'$. Ils ont fourni le théorème de représentation suivant :

Theorem 4 *[14]*

- *Un opérateur de mise à jour $\diamond$ satisfait les postulats* (U1)–(U5) *et* (U8)–(U9) *si et seulement si il existe une affectation fidèle ponctuelle qui associe à chaque interpretation w un préordre total $\leq_w$ tel que* $\text{Mod}(\psi \diamond \mu) = \bigcup_{w \in \text{Mod}(\psi)} \textit{min}(\text{Mod}(\mu), \leq_w)$.
- *Un opérateur de mise à jour $\diamond$ satisfait les postulats* (U1)–(U8) *si et seulement si il existe une affectation fidèle ponctuelle qui associe à chaque interpretation w un préordre partiel $\leq_w$ tel que* $\text{Mod}(\psi \diamond \mu) = \bigcup_{w \in \text{Mod}(\psi)} \textit{min}(\text{Mod}(\mu), \leq_w)$.

Ce théorème de représentation permet de mettre en évidence la différence entre révision et mise-à-jour. La mise à jour repose sur une minimisation ponctuelle, modèle par modèle de ψ alors que la révision repose sur une minimisation globale. Il est à noter que lorsque ψ est une formule complète révision et mise à jour coïncident.

Nous nous limitons à présenteur deux opérateurs de mise à jour, l'opérateur de Forbus [9], noté $\diamond_F$, et l'opérateur de de Winslett est aussi appelé *PMA (Possible Models Approach)*[20], noté $\diamond_W$, définis respectivement comme suit :

$$\text{Mod}(\psi \diamond_F \mu) = \bigcup_{w \in \text{Mod}(\psi)} \{w' \in \text{Mod}(\mu) : |w\Delta w'| = |\Delta|_w^{min}(\mu)\}.$$

$$\text{Mod}(\psi \diamond_W \mu) = \bigcup_{m \in \text{Mod}(\psi)} \{m' \in \text{Mod}(\mu) : m\Delta m' \in \Delta_m^{min}(\mu)\}.$$

Notons que l'opérateur $\diamond_F$ satisfait (U1)–(U9) [14, 11] tandis que l'opérateur $\diamond_W$ satisfait (U1)–(U8) [14], mais viole (U9)[16].

4 Effacement des croyances

L'effacement, introduit par Katsuno and Mendelzon [14], est à la contraction ce que la mise à jour est à la révision. Intuitivement, effacer une croyance signifie que l'environnement de l'agent a changé de telle sorte que cette croyance peut ne plus être valide. D'un point de vue logique lorsque les croyances d'un agent sont représentées par une formule ψ, effacer une croyance μ de ψ signifie ajouter localement les modèles de $\neg\mu$ aux modèles de ψ. Les opérateurs d'effacement que nous étudions sont des fonctions, notées $\triangleleft$ de $\mathcal{L}\times\mathcal{L}$ vers $\mathcal{L}$ qui à partir d'une formule ψ qui représente les croyances initiales d'un agent et d'une formule μ à effacer, renvoie une formule $\psi \triangleleft \mu$.

4.1 Postulats de base pour l'effacement

Des postulats que tout opérateur d'effacement devrait satisfaire ont été proposés par Katsuno et Mendelzon [14] dans le même esprit que ceux proposés pour la contraction et la mise à jour.

Soit $\psi, \psi_1, \psi_2, \mu, \mu_1, \mu_2 \in \mathcal{L}$.

(E1) $\psi \models \psi \triangleleft \mu$.
(E2) Si $\psi \not\models \mu$, alors $\psi \triangleleft \mu \equiv \psi$.
(E3) Si ψ est satisfaisable et $\not\models \mu$, alors $\psi \triangleleft \mu \not\models \mu$.
(E4) Si $\psi_1 \equiv \psi_2$ et $\mu_1 \equiv \mu_2$, alors $\psi_1 \triangleleft \mu_1 \equiv \psi_2 \triangleleft \mu_2$.
(E5) $(\psi \triangleleft \mu) \wedge \mu \models \psi$.
(E8) $(\psi_1 \vee \psi_2) \triangleleft \mu \equiv (\psi_1 \triangleleft \mu) \vee (\psi_2 \triangleleft \mu)$.

De façon similaire aux identités de Levi et Harper liant contraction et révision Katsuno and Mendelzon [14] ont proposé des identités reliant effacement et mise à jour :

(Id_1) $\psi \triangleleft \mu \equiv \psi \vee (\psi \diamond \neg\mu)$
(Id_2) $\psi \diamond \mu \equiv (\psi \triangleleft \neg\mu) \wedge \mu$

De plus, ils ont proposé le résultat suivant concernant la satisfaction des postulats.

Proposition 1 [14]

1. *Si un opérateur de mise à jour $\diamond$ satisfait (*U1)-(U4) et (U8), alors l'opérateur d'effacement $\triangleleft$ defini par l'identité (Id_1) satisfait (E1)-(E5) et (E8).
2. Si un opérateur d'effacement $\triangleleft$ satisfait (E1)-(E4) et (E8), alors l'opérateur de mise à jour $\diamond$ defini par l'identité (Id_2) satisfait (U1)-(U4) et (U8).
3. Si un opérateur de mise à jour $\diamond$ satisfait (U1)-(U4) et (U8), alors il est possible de définir un opérateur d'effacement grâce à (Id_1). L'opérateur de mise à jour obtenu à partir de cet opérateur d'effacement via (Id_2) est égal à l'opérateur de mise à jour initial $\diamond$.
4. Si un opérateur d'effacement $\triangleleft$ satisfait (E1)-(E5) et (E8), alors il est possible de définir un opérateur de mise à jour grâce à (Id_2). L'opérateur d'effacement obtenu à partir de cet opérateur de mise à jour via (Id_1) est égal à l'opérateur d'effacement initial $\triangleleft$.

L'identité (Id_1) nous a permis de définir des opérateurs d'effacement à partir des opérateurs de mise à jour connus [5]. Un opérateur d'effacement, noté $\triangleleft_F$, est défini par

$$\mathrm{Mod}(\psi \triangleleft_F \mu) = \mathrm{Mod}(\psi) \cup \mathrm{Mod}(\psi \diamond_F \neg\mu)$$

où $\diamond_F$ est l'opérateur de mise à jour de Forbus. Un opérateur d'effacement, noté $\triangleleft_W$, est défini par

$$\mathrm{Mod}(\psi \triangleleft_W \mu) = \mathrm{Mod}(\psi) \cup \mathrm{Mod}(\psi \diamond_W \neg\mu)$$

où $\diamond_W$ est l'opérateur de mise à jour de Winslett. Selon la Proposition 1, les opérateurs d'effacement $\triangleleft_F$ et $\triangleleft_W$ satisfont (E1) – (E5) et (E8).

L'exemple suivant illustre les opérateurs $\triangleleft_F$ et $\triangleleft_W$.

Example 1 *Soit ψ, μ deux formules propositionnelles telles que $Mod(\psi) = \{abcd, a\}$ et $Mod(\mu) = \{a, c, d, ab, ac, ad, bc, cd, abc, abd, bcd, abcd, \emptyset\}$. Nous avons $Mod(\neg\mu) = \{acd, bd, b\}$ et d'après la Table 1, $Mod(\psi \diamond_F \neg\mu) = \{acd, b\}$ et $Mod(\psi \diamond_W \neg\mu) = \{acd, bd, b\}$. Les résultats de l'effacement avec les opérateurs $\triangleleft_F$ et $\triangleleft_W$ sont respectivement $Mod(\psi \triangleleft_F \mu) = \{abcd, a, acd, b\}$ et $Mod(\psi \triangleleft_W \mu) = \{abcd, a, acd, bd, b\}$.*

	$Mod(\psi)$		
	Δ	$abcd$	a
$Mod(\neg\mu)$	acd	$\mathbf{b^*}$	$\mathbf{cd^*}$
	bd	ac^*	abd
	b	acd	$\mathbf{ab^*}$

Table 1 – Différences symétriques ; par colonne, les minimaux selon la cardinalité sont notés en gras et les minimaux selon l'inclusion sont notés avec une asterisque.

4.2 De nouveaux postulats capturant la minimalité du changement

Nous pouvons ajouter deux postulats supplémentaires pour capturer la minimalité du changement. Ils sont équivalents à (C6) et (C7), à la seule différence qu'en raison de la règle de disjonction (E8), le postulat (E7) peut être restreint aux formules complètes.

(E6) $\psi \triangleleft (\mu_1 \wedge \mu_2) \models (\psi \triangleleft \mu_1) \vee (\psi \triangleleft \mu_2)$.
(E7) Si ψ est complète et $\psi \triangleleft (\mu_1 \wedge \mu_2) \not\models \mu_1$, alors $\psi \triangleleft \mu_1 \models \psi \triangleleft (\mu_1 \wedge \mu_2)$.

On peut observer que l'opérateur d'effacement de Forbus $\triangleleft_F$ satisfait (E6) et (E7), tandis que l'opérateur de Winslett $\triangleleft_W$ satisfait (E6) mais pas (E7).

Il n'est guère surprenant que nous puissions énoncer un théorème de représentation pour les opérateurs d'effacement définis par des préordres totaux qui est la contrepartie du théorème de représentation obtenu par Caridroit *et al.* [4] pour la contraction. La preuve nécessite les deux lemmes suivants.

Le premier lemme explicite le résultat de l'effacement par une formule qui n'a qu'un seul contre-modèle et n'utilise que les postulats de base.

Lemma 1 *Soit $\triangleleft$ un opérateur d'effacement qui satisfait les postulats* (E1)–(E5) *et* (E8)*, ψ une formule satisfaisable, w une interpretation et α_w une formule ayant w comme unique modèle, alors $\psi \triangleleft \neg\alpha_w \equiv \psi \vee \alpha_w$.*

Preuve: Puisque l'opérateur satisfait (E8) il est suffisant de prouver le lemme quand ψ est une formule complète ayant, disons w_0 comme unique modèle.

Si $w_0 = w$, alors $\psi \models \alpha_w$ et selon (E2), $\psi \triangleleft \neg\alpha_w \equiv \psi \equiv \psi \vee \alpha_w$.

Si $w_0 \neq w$, d'après (E5) $(\psi \triangleleft \neg\alpha_w) \wedge \neg\alpha_w \models \psi$, et donc $(\psi \triangleleft \neg\alpha_w) \models \psi \vee \alpha_w$. Par (E1), $\psi \models (\psi \triangleleft \neg\alpha_w)$. De plus, selon (E3), $(\psi \triangleleft \neg\alpha_w) \not\models \neg\alpha_w$. Ainsi nous obtenons $\alpha_w \models (\psi \triangleleft \neg\alpha_w)$ et donc $\psi \triangleleft \neg\alpha_w \equiv \psi \vee \alpha_w$. □

Le deuxième lemme met en jeu les deux postulats additionnels qui traitent de la minimalité du changement.

Lemma 2 *Soit $\triangleleft$ un opérateur d'effacement qui satisfait les postulats* (E1)–(E8)*, ψ une formule complète, et α et β deux formules qui ne sont pas des tautologies, alors $\psi \triangleleft (\alpha \wedge \beta) \equiv \psi \triangleleft \alpha$ ou $\psi \triangleleft \beta$ ou $(\psi \triangleleft \alpha) \vee (\psi \triangleleft \beta)$.*

Preuve: Sous les hypothèses faites sur les formules la preuve est similaire à celle donnée dans [4, preuve de la Proposition 9]. □

Dans tout ce qui suit, étant donné une interprétation w, ψ_w (resp. α_w) représente une formule complète dont l'unique modèle est w. Egalement, étant donné une interprétation w_i, on note α_i une formule complète dont l'unique modèle est w_i.

4.3 L'effacement en termes de préordres totaux

Nous sommes maintenant en mesure de prouver le théorème de représentation suivant, qui montre que les postulats capturent tous les opérateurs d'effacement définis par un préordre total.

Theorem 5 *Un opérateur d'effacement $\triangleleft$ satisfait les postulats* (E1)–(E8) *si et seulement si il existe une affectation fidèle ponctuelle qui associe à chaque interprétation w un préordre total $\leq_w$ tel que* $\mathrm{Mod}(\psi \triangleleft \mu) = \mathrm{Mod}(\psi) \cup \bigcup_{w \in \mathrm{Mod}(\psi)} min_{\leq_w}(\mathrm{Mod}(\neg\mu))$.

Preuve: $\Leftarrow$) Suposons que nous avons une affectation fidèle ponctuelle qui associe à chaque interprétation w un préordre total $\leq_w$.

Considérons l'opérateur d'effacement $\triangleleft$ défini par

$$\mathrm{Mod}(\psi \triangleleft \mu) = \mathrm{Mod}(\psi) \cup \bigcup_{w \in \mathrm{Mod}(\psi)} min_{\leq_w}(\mathrm{Mod}(\neg\mu)).$$

Prouvons dans un premier temps que $\triangleleft$ satisfait les postulats (E1)–(E8). Il est évident que $\triangleleft$ satisfait (E1), (E3), (E4), (E5) et (E8). Si ψ est insatisfaisable alors (E2), (E6) et (E7) sont trivialement vérifiés. Nous supposons donc dans la suite que ψ est satisfaisable.

Il découle de la définition d'une affectation fidèle ponctuelle que si w est un modèle de $\neg\mu$, alors $\psi_w \triangleleft \mu$ est équivalent à ψ_w. On obtient donc (E2) en utilisant (E8).

Puisque $min_{\leq_w}\mathrm{Mod}(\neg\mu_1 \vee \neg\mu_2) \subseteq min_{\leq_w}\mathrm{Mod}(\neg\mu_1) \cup min_{\leq_w}\mathrm{Mod}(\neg\mu_2)$, (E6) est vérifié.

Soit ψ une formule complète telle que $\mathrm{Mod}(\psi) = \{w_0\}$. Supposons que $\psi \triangleleft (\mu_1 \wedge \mu_2) \not\models \mu_1$. Cela signifie qu'il existe $w \in \mathrm{Mod}(\psi) \cup min_{\leq_{w_0}}(\mathrm{Mod}(\neg(\mu_1 \wedge \mu_2)))$ tel que $w \in \mathrm{Mod}(\neg\mu_1)$. Si $w \in \mathrm{Mod}(\psi \wedge \neg\mu_1)$, alors $\mathrm{Mod}(\psi) = \{w\}$ et dans ce cas puisque l'affectation est fidèle ponctuelle $\mathrm{Mod}(\psi \triangleleft \mu_1) = \mathrm{Mod}(\psi)$. Le fait que (E7) est valide découle alors de (E1). Si $w \notin \mathrm{Mod}(\psi)$, pour vérifier que (E7) est valide il suffit de montrer que $min_{\leq_{w_0}}(\mathrm{Mod}(\neg\mu_1)) \subseteq min_{\leq_{w_0}}(\mathrm{Mod}(\neg(\mu_1 \wedge \mu_2))$. Soit $w' \in min_{\leq_{w_0}}(\mathrm{Mod}(\neg\mu_1))$. Puisque $\leq_{w_0}$ est un préordre total et que $w \in \mathrm{Mod}(\neg\mu_1)$, nous avons $w' \leq_{w_0} w$. Supposons qu'il existe w" $\in \mathrm{Mod}(\neg(\mu_1 \wedge \mu_2))$ tel que w" $<_{w_0} w'$, alors on a également w" $<_{w_0} w$, ce qui contredit le fait que $w \in min_{\leq_{w_0}}(\mathrm{Mod}(\neg(\mu_1 \wedge \mu_2)))$. Donc (E7) est bien vérifié.

$\Rightarrow$) Soit $\triangleleft$ un opérateur d'effacement qui satisfait les postulats(E1)–(E8). Définissons une relation binaire $\leq_w$ sur les interprétations comme suit :

$$w_1 \leq_w w_2 \text{ si ou bien } w_1 = w \text{ ou } w_1 \in \mathrm{Mod}(\psi_w \triangleleft \neg(\alpha_1 \vee \alpha_2)).$$

Montrons tout d'abord que $\leq_w$ est un préordre total. Il découle du postulat (E3) que ou bien w_1 ou w_2 appartient à $\mathrm{Mod}(\psi_w \triangleleft \neg(\alpha_1 \vee \alpha_2))$, prouvant ainsi que la relation binaire est totale. Le fait que $\leq_w$ est reflexive découle du Lemme 1.

La preuve que $\leq_w$ est transitive est complètement similaire à la preuve donnée par Caridroit *et al.* dans le cas de la contraction (voir [4, Preuve du Theorem 14]). Cette dernière s'appuie sur un lemme analogue au Lemme 2 pour la contraction et utilise alors seulement les postulats (C1), (C6) et (C7), qui sont analogues à (E1), (E6) et (E7) (quand le dernier est restreint à une formule complète).

Il découle de (E2) que l'application $w \mapsto \leq_w$ est une affectation fidèle ponctuelle.

Il reste à montrer que

$$\mathrm{Mod}(\psi \lhd \mu) = \mathrm{Mod}(\psi) \cup \bigcup_{w \in \mathrm{Mod}(\psi)} min_{\leq_w}(\mathrm{Mod}(\neg\mu)).$$

Si ψ est insatisfaisable, alors les deux côtés de l'équation sont vides et l'égalité est vérifiée. Si ψ est satisfaisable, alors selon (E8) étant donné une interprétation w il est suffisant de prouver que $\mathrm{Mod}(\psi_w \lhd \mu) = \{w\} \cup min_{\leq_w}(\mathrm{Mod}(\neg\mu))$. Si $w \in \mathrm{Mod}(\neg\mu)$ alors il découle de (E2) que $\psi_w \lhd \mu \equiv \psi$ et l'égalité est valide puisque nous utilisons une affectation fidèle. Nous supposons donc dans la suite que $w \notin \mathrm{Mod}(\neg\mu)$. Si μ est une tautologie, selon (E1) et (E5) $\psi_w \lhd \mu \equiv \psi$ et l'égalité est valide. Supposons maintenant qu'il existe $w_1 \in \mathrm{Mod}(\psi_w \lhd \mu)$ qui est dans $\mathrm{Mod}(\neg\mu)$ mais qui n'appartient pas à $min_{\leq_w}(\mathrm{Mod}(\neg\mu))$. Alors il existe $w_2 \in \mathrm{Mod}(\neg\mu)$ tel que $w_2 <_w w_1$. Nous avons alors $w_1 \notin \mathrm{Mod}(\psi_w \lhd \neg(\alpha_1 \vee \alpha_2))$. Considérons maintenant la formule $\beta = \neg\mu \wedge \neg\alpha_1 \wedge \neg\alpha_2$. Clairement nous avons $\neg\mu \equiv \beta \vee (\alpha_1 \vee \alpha_2)$. Puisque (E4) est satisfait, $\psi_w \lhd \mu \equiv \psi_w \lhd (\neg\beta \wedge \neg\alpha_1 \wedge \neg\alpha_2)$. Et donc selon (E6), $\psi_w \lhd \mu \models (\psi_w \lhd \neg\beta) \vee (\psi_w \lhd \neg(\alpha_1 \vee \alpha_2))$. Nous avons supposé que $w_1 \in \mathrm{Mod}(\psi_w \lhd \mu)$ et $w_1 \notin \mathrm{Mod}(\psi_w \lhd \neg(\alpha_1 \vee \alpha_2))$, donc $w_1 \in \mathrm{Mod}(\psi_w \lhd \neg\beta)$. Selon (E5) nous avons $(\psi_w \lhd \neg\beta) \wedge \neg\beta \models \psi_w$. Puisque nous avons également $w_1 \in \mathrm{Mod}(\neg\beta)$, nous obtenons $w_1 \in \mathrm{Mod}(\psi_w)$, c'est-à-dire, $w_1 = w$, contradiction.

Cela prouve que $\mathrm{Mod}(\psi_w \lhd \mu) \subseteq \{w\} \cup min_{\leq_w}(\mathrm{Mod}(\neg\mu))$.

Montrons maintenant l'inclusion inverse. Selon (E1), $w \in \mathrm{Mod}(\psi_w \lhd \mu)$. Considérons w_1 appartenant à $min_{\leq_w}(\mathrm{Mod}(\neg\mu))$ et dans le but d'obtenir une contradiction supposons $w_1 \notin \mathrm{Mod}(\psi_w \lhd \mu)$. Dans ce cas μ n'est pas une tautologie et selon (E3) $\psi_w \lhd \mu \not\models \mu$, ainsi il existe w_2 un modèle de $\neg\mu$ qui est dans $\mathrm{Mod}(\psi_w \lhd \mu)$. Si $w_2 = w$ alors $w_2 <_w w_1$, ce qui contredit la minimalité de w_1.

Si $w_2 \neq w$. Puisque $w_2 \in \mathrm{Mod}(\psi_w \lhd \mu)$ nous avons $\psi_w \lhd \mu \not\models \neg(\alpha_1 \vee \alpha_2)$. Puisque w_1 et w_2 sont tous les deux des modèles de $\neg\mu$ observons que $\mu \wedge \neg(\alpha_1 \vee \alpha_2) \equiv \mu$, et donc selon (E7), $\psi_w \lhd \neg(\alpha_1 \vee \alpha_2) \models \psi_w \lhd \mu$. Puisque par hypothèse $w_1 \notin \mathrm{Mod}(\psi_w \lhd \mu)$, nous avons $w_1 \notin \mathrm{Mod}(\psi_w \lhd \neg(\alpha_1 \vee \alpha_2))$. Donc, d'après le Lemme 2 et le Lemme 1, $\mathrm{Mod}(\psi_w \lhd \neg(\alpha_1 \vee \alpha_2)) = \{w, w_2\}$, ce qui contredit la minimalité de w_1. □

4.4 L'effacement en termes de préordres partiels

Rappelons que l'opérateur d'effacement de Winslett ne satisfait pas (E7), donc cet opérateur n'est pas pris en compte par le théorème précédent. Cet opérateur est défini par un préordre partiel et non total. Nous pouvons modifier les postulats de l'effacement de façon à ce qu'ils s'adaptent aux préordres partiels.

Dans la preuve du Théorème 5 étant donné un opérateur d'effacement associé à un préordre total, seul le postulat (E7) exige que le préordre soit total. Par conséquent, comme l'ont fait Katsuno et Mendelzon pour la révision, afin d'obtenir un théorème de représentation par le biais de préordres partiels, nous supprimons le postulat (E7) et le remplaçons par deux postulats plus faibles.

(E9) Si $\psi \models \mu_1 \wedge \mu_2$, $(\psi \lhd \mu_1) \models \psi \vee \neg\mu_2$ et $(\psi \lhd \mu_2) \models \psi \vee \neg\mu_1$, alors $(\psi \lhd \mu_1) \equiv (\psi \lhd \mu_2)$.

(E10) Si ψ est complète, alors $(\psi \lhd \mu_1) \wedge (\psi \lhd \mu_2) \models \psi \lhd (\mu_1 \vee \mu_2)$.

Les opérateurs d'effacement $\lhd_F$ et $\lhd_W$ satisfont tous les deux postulats (E9) et (E10).

La définition de ces postulats permet de concevoir une classe d'opérateurs d'effacement basés sur des préordres partiels. Le théorème suivant montre que les opérateurs d'effacement basés sur des préordres partiels sont complètement caractérisés par les postulats (E1)–(E6) et (E8)–(E10).

Theorem 6 *Un opérateur d'effacement $\lhd$ satisfait les postulats* (E1)–(E6) *et* (E8)–(E10) *si et seulement si il existe une affectation fidèle ponctuelle qui associe à chaque interprétation w un préordre partiel $\leq_w$ tel que* $\mathrm{Mod}(\psi \lhd \mu) = \mathrm{Mod}(\psi) \cup \bigcup_{w \in \mathrm{Mod}(\psi)} min_{\leq_w}(\mathrm{Mod}(\neg\mu))$.

Preuve: $\Leftarrow$) Supposons que nous avons affectation fidèle ponctuelle qui associe à chaque interprétation w un préordre partiel $\leq_w$. Considérons l'opérateur d'effacement $\lhd$ défini par $\mathrm{Mod}(\psi \lhd \mu) = \mathrm{Mod}(\psi) \cup \bigcup_{w \in \mathrm{Mod}(\psi)} min_{\leq_\psi}(\mathrm{Mod}(\neg\mu))$.

Prouvons dans un premier temps que l'opérateur $\lhd$ satisfait les postulats (E1)–(E6) et (E8)–(E10). La preuve que $\lhd$ satisfait les postulats (E1)–(E6) et (E8) est similaire à la preuve du Théorème 5 ci-dessus, le fait que le préordre est partiel (et non nécessairement total) n'a pas d'impact sur la preuve de ces sept postulats.

Supposons que $\psi \models \mu_1 \wedge \mu_2$, $(\psi \lhd \mu_1) \models \psi \vee \neg\mu_2$ et $(\psi \lhd \mu_2) \models \psi \vee \neg\mu_1$. Dans le but d'obtenir une contradiction supposons qu'il existe une interprétation w telle $w \in \mathrm{Mod}(\psi \lhd \mu_1)$ et $w \notin \mathrm{Mod}(\psi \lhd \mu_2)$. Observons que $w \notin \mathrm{Mod}(\psi)$. Puisque $(\psi \lhd \mu_1) \models \psi \vee \neg\mu_2$, nous avons $w \in \mathrm{Mod}(\neg\mu_2)$. Par définition de l'opérateur, puisque $w \notin \mathrm{Mod}(\psi \lhd \mu_2)$, pour chaque modèle w_i de ψ il existe $w'_i \in min_{\leq_{w_i}}(\mathrm{Mod}(\neg\mu_2))$ tel que $w'_i <_{w_i} w$ et w'_i est minimal dans $\mathrm{Mod}(\neg\mu_2)$. Alors chaque w'_i est un modèle de $(\psi \lhd \mu_2)$. Or $(\psi \lhd \mu_2) \models \psi \vee \neg\mu_1$, et donc ou bien $w'_i \in \mathrm{Mod}(\psi)$ ou $w'_i \in \mathrm{Mod}(\neg\mu_1)$. Si $w'_i \in \mathrm{Mod}(\psi)$, alors $w'_i \in \mathrm{Mod}(\psi) \cap \mathrm{Mod}(\neg\mu_2)$, ce qui contredit le fait que $\psi \models \mu_2$. Si $w'_i \in \mathrm{Mod}(\neg\mu_1)$, alors $w'_i <_{w_i} w$ ce qui contredit le fait que $w \in min_{\leq_{w_i}}(\mathrm{Mod}(\neg\mu_1))$. Ainsi dans les deux cas nous obtenons une contradiction et nous avons prouvé que $\mathrm{Mod}(\psi \lhd \mu_1) \subseteq \mathrm{Mod}(\psi \lhd \mu_2)$. L'inclusion inverse

est prouvée de la même façon ce qui montre que (E9) est satisfait.

Notons que $\text{Mod}((\psi_w \triangleleft \mu_1) \wedge (\psi_w \triangleleft \mu_2)) = \{w\} \cup min_{\leq_w}(\text{Mod}(\neg\mu_1)) \cap min_{\leq_w}(\text{Mod}(\neg\mu_2)) \subseteq \{w\} \cup min_{\leq_w}\text{Mod}(\neg\mu_1 \wedge \neg\mu_2)$. Donc $(\psi_w \triangleleft \mu_1) \wedge (\psi_w \triangleleft \mu_2) \models \psi_w \triangleleft (\mu_1 \vee \mu_2)$, ce qui prouve que (E10) est satisfait.

$\Rightarrow$) Soit $\triangleleft$ un opérateur d'effacement qui satisfait les postulats (E1)–(E6) et (E8)–(E10). Rappelons que dans le suite étant donné une interprétation w_i, nous notons α_i une formule dont le seul modèle est w_i.

Pour chaque interprétation w nous définissons une relation binaire $\leq_w$ sur les interprétations par :

$$w_1 \leq_w w_2 \text{ si } \text{Mod}(\psi_w \triangleleft \neg(\alpha_1 \vee \alpha_2)) = \{w\} \cup \{w_1\}.$$

Montrons dans un premier temps que $\leq_w$ est un préordre. D'après le Lemme 1, $\text{Mod}(\psi_w \triangleleft \neg\alpha_1) = \text{Mod}(\psi_w) \cup \{w_1\}$, et donc $\leq_w$ est réflexive.

Prouvons maintenant que la relation est transitive. Considérons trois interprétations deux à deux distinctes w_1, w_2 et w_3 telles que $w_1 \leq_w w_2$ et $w_2 \leq_w w_3$. Nous avons donc $\text{Mod}(\psi_w \triangleleft \neg(\alpha_1 \vee \alpha_2)) = \text{Mod}(\psi_w) \cup \{w_1\}$, c'est-à-dire $\psi \triangleleft \neg(\alpha_1 \vee \alpha_2) \equiv \psi_w \vee \alpha_1$ et $\text{Mod}(\psi_w \triangleleft \neg(\alpha_2 \vee \alpha_3)) = \text{Mod}(\psi_w) \cup \{w_2\}$, c'est-à-dire $\psi_w \triangleleft \neg(\alpha_2 \vee \alpha_3) \equiv \psi_w \vee \alpha_2$. Supposons tout d'abord qu'une de ces interprétations est égale à w. Si $w_1 = w$, alors par (E2), $\text{Mod}(\psi_w \triangleleft \neg(\alpha_1 \vee \alpha_3)) = \text{Mod}(\psi_w)$ et $w_1 \leq_w w_3$. Si $w_2 = w$, alors selon les hypothèses et par (E2), on a également $w_1 = w$ et donc $w_1 = w_2$. Si $w_3 = w$ alors selon les hypothèses et par (E2) nous avons $w_1 = w_2 = w_3 = w$. Supposons maintenant qu'aucune des interprétations w_1, w_2 et w_3 n'est égale à w, et donc que $\psi_w \models (\neg\alpha_1 \wedge \neg\alpha_2 \wedge \neg\alpha_3)$. D'une part $\psi_w \triangleleft (\neg\alpha_1 \wedge \neg\alpha_2) \equiv \psi_w \vee \alpha_1$, d'où $\psi_w \triangleleft (\neg\alpha_1 \wedge \neg\alpha_2) \models \psi_w \vee (\alpha_1 \vee \alpha_2 \vee \alpha_3)$. D'autre part d'après (E6), $\psi_w \triangleleft (\neg\alpha_1 \wedge \neg\alpha_2 \wedge \neg\alpha_3) \models (\psi_w \triangleleft \neg\alpha_1) \vee (\psi_w \triangleleft (\neg\alpha_2 \wedge \neg\alpha_3))$. Donc sous nos hypothèses $\psi_w \triangleleft (\neg\alpha_1 \wedge \neg\alpha_2 \wedge \alpha_3) \models (\psi_w \triangleleft \neg\alpha_1) \vee \psi_w \vee \alpha_2$. Selon le Lemme 1, $\psi_w \triangleleft \neg\alpha_1 \equiv \psi_w \vee \alpha_1$. D'où $\psi_w \triangleleft (\neg\alpha_1 \wedge \neg\alpha_2 \wedge \neg\alpha_3) \models \psi_w \vee \alpha_1 \vee \alpha_2$. Donc par (E9) $\psi_w \triangleleft (\neg\alpha_1 \wedge \neg\alpha_2) \equiv \psi_w \triangleleft (\neg\alpha_1 \wedge \neg\alpha_2 \wedge \neg\alpha_3)$. En conséquence, d'une part, $\text{Mod}(\psi_w \triangleleft (\neg\alpha_1 \wedge \neg\alpha_2 \wedge \neg\alpha_3)) = \{w, w_1\}$, et donc $\psi_w \triangleleft (\neg\alpha_1 \wedge \neg\alpha_2 \wedge \neg\alpha_3) \models \psi_w \vee \alpha_1 \vee \alpha_3$. D'autre part par (E5), $(\psi_w \triangleleft (\neg\alpha_1 \wedge \neg\alpha_3)) \wedge (\neg\alpha_1 \wedge \neg\alpha_3)) \models \psi_w$, et donc $(\psi_w \triangleleft (\neg\alpha_1 \wedge \neg\alpha_3)) \models \psi_w \vee (\alpha_1 \vee \alpha_3) \models \psi_w \vee (\alpha_1 \vee \alpha_2 \vee \alpha_3)$. Par (E9) nous obtenons que $\psi_w \triangleleft (\neg\alpha_1 \wedge \neg\alpha_3) \equiv \psi_w \triangleleft (\neg\alpha_1 \wedge \neg\alpha_2 \wedge \neg\alpha_3)$. On en déduit que $\psi_w \triangleleft (\neg\alpha_1 \wedge \neg\alpha_3) \equiv \psi_w \triangleleft -(\neg\alpha_1 \wedge \neg\alpha_2)$. Donc $\text{Mod}(\psi_w \triangleleft \neg(\alpha_1 \vee \alpha_3)) = \text{Mod}(\psi_w) \cup \{w_1\}$, c'est-à-dire $w_1 \leq_w w_3$, ce qui prouve la transitivité de la relation $\leq_w$.

Il découle du postulat (E2) que l'application $w \mapsto \leq_w$ est une assignation fidèle ponctuelle.

Il reste à prouver que $\text{Mod}(\psi \triangleleft \mu) = \text{Mod}(\psi) \cup \bigcup_{w \in \text{Mod}(\psi)} min_{\leq_w}(\text{Mod}(\neg\mu))$. Si ψ is insatisfaisable, alors les deux membres de l'équation sont vides et l'égalité est trivialement vérifiée. Si ψ est satisfaisable, alors puisque le postulat (E8) est vérifié, étant donné une interprétation w il est suffisant de prouver que $\text{Mod}(\psi_w \triangleleft \mu) = \{w\} \cup min_{\leq_w}(\text{Mod}(\neg\mu))$. Si $w \in \text{Mod}(\neg\mu)$ alors il découle de (E2) que $\psi_w \triangleleft \mu \equiv \psi_w$ et l'égalité est vérifiée puisque l'assignation est fidèle. Si μ est une tautologie alors selon (E5) $\psi_w \triangleleft \mu \equiv \psi_w$ et l'égalité est vérifiée. Nous supposons désormais que $w \notin \text{Mod}(\neg\mu)$ et que μ n'est pas une tautologie.

Prouvons tout d'abord que $\text{Mod}(\psi_w) \cup min_{\leq_w}(\text{Mod}(\neg\mu)) \subseteq \text{Mod}(\psi_w \triangleleft \mu)$. On sait par (E1) que $w \in \text{Mod}(\psi_w \triangleleft \mu)$. Considérons maintenant $w_0 \neq w$ et $w_0 \in min_{\leq_w}(\text{Mod}(\neg\mu))$.

Supposons que $\text{Mod}(\neg\mu) = \{w_0, w_1, \ldots, w_n\}$. Pour toute interprétation $w_i \in \text{Mod}(\neg\mu)$, puisque ni $\neg\alpha_0$ ni $\neg\alpha_i$ ne sont des tautologies il découle du Lemme 2 que $\psi_w \triangleleft (\neg\alpha_0 \wedge \neg\alpha_i) \equiv \psi_w \vee \alpha_0$ ou $\psi_w \vee \alpha_i$ ou $\psi_w \vee \alpha_0 \vee \alpha_i$. Puisque $w_0 \in min_{\leq_w}(\text{Mod}(\neg\mu))$ il n'existe aucun $w_i \in \text{Mod}(\neg\mu)$ tel que $\psi_w \triangleleft (\neg\alpha_0 \wedge \neg\alpha_i) \equiv \psi_w \vee \alpha_i$, et donc $w_0 \in \psi_w \triangleleft (\neg\alpha_0 \wedge \neg\alpha_i)$.

Observons que $\mu \equiv \bigvee_{i=1}^{n}(\alpha_0 \vee \alpha_i)$. Donc selon (E4) et en appliquant de façon répétée (E10) nous obtenons que $w_0 \in \text{Mod}(\psi_w \triangleleft \mu)$, prouvant ainsi que

$\text{Mod}(\psi_w) \cup min_{\leq_\psi}(\text{Mod}(\neg\mu)) \subseteq \text{Mod}(\psi_w \triangleleft \mu)$.

Montrons maintenant l'inclusion inverse, $\text{Mod}(\psi_w \triangleleft \mu) \subseteq \text{Mod}(\psi_w) \cup min_{\leq_w}(\text{Mod}(\neg\mu))$. Considérons $w_0 \in \text{Mod}(\psi_w \triangleleft \mu)$ tel que $w_0 \neq w$. Par (E5) nous avons $w_0 \in \text{Mod}(\neg\mu)$. Afin d'obtenir une contradiction supposons que $w_0 \notin min_{\leq_w}(\text{Mod}(\neg\mu))$. Cela signifie qu'il existe $w_1 \in \text{Mod}(\neg\mu)$ tel que $w_1 <_w w_0$, c'est-à-dire, $\psi_w \triangleleft (\neg\alpha_0 \wedge \neg\alpha_1)) \equiv \psi_w \vee \alpha_1$. Considèrons maintenant la formule $\beta = \neg\mu \wedge \neg\alpha_0 \wedge \neg\alpha_1$. Clairement nous avons $\neg\mu \equiv \beta \vee \alpha_0 \vee \alpha_1$. Selon (E4), $\psi_w \triangleleft \mu \equiv \psi_w \triangleleft (\neg\beta \wedge \neg\alpha_0 \wedge \neg\alpha_1)$. Par (E6), $\psi_w \triangleleft (\neg\beta \wedge \neg\alpha_0 \wedge \neg\alpha_1) \models (\psi_w \triangleleft \neg\beta) \vee (\psi_w \triangleleft (\neg\alpha_0 \wedge \neg\alpha_1))$. Puisque $w_1 <_w w_0$, $w_0 \notin \text{Mod}(\psi_w \triangleleft (\neg\alpha_0 \wedge \neg\alpha_1)$, et donc $w_0 \in \text{Mod}(\psi_w \triangleleft \neg\beta)$. On a également $w_0 \in \text{Mod}(\neg\mu)$ et *a fortiori* $w_0 \in \text{Mod}(\neg\beta)$. Alors selon (E5), $(\psi_w \triangleleft \neg\beta) \wedge (\neg\beta) \models \psi_w$, et nous obtenons $w_0 = w$, ce qui fournit une contradiction. □

4.5 La contraction en termes de préordres partiels

Comme nous l'avons observé dans la Section 3.2, il manque toujours un théorème de représentation pour les opérateurs de contraction par le biais de préordres partiels. Notre objectif est de combler cette lacune. De la même manière que pour l'effacement, nous sommes en mesure de donner une version des postulats de contraction qui prend en compte les préordres partiels et nous pouvons ainsi concevoir une classe d'opérateurs de contraction basés sur les préordres partiels.

Nous supprimons le postulat (C7) et le remplaçons par deux postulats plus faibles, (C8) et (C9). Ils sont similaires

aux postulats (E9) et (E10), à l'exception du dernier qui n'est plus restreint aux formules complètes.

(C8) Si $\psi \models \mu_1 \wedge \mu_2$, $(\psi - \mu_1) \models \psi \vee \neg\mu_2$ et $(\psi - \mu_2) \models \psi \vee \neg\mu_1$, alors $(\psi - \mu_1) \equiv (\psi - \mu_2)$.

(C9) $(\psi - \mu_1) \wedge (\psi - \mu_2) \models \psi - (\mu_1 \vee \mu_2)$.

Les opérateurs de contraction $-_D$ et $-_S$ satisfont (C8) et (C9).

Le théorème suivant montre que les opérateurs de contraction basés sur des préordres partiels sont parfaitement caractérisés par les postulats (C1)–(C6), (C8) et (C9).

Theorem 7 *Un opérateur de contraction $-$ satisfait les postulats* (C1)–(C6) *et* (C8)–(C9) *si et seulement si il existe une affectation fidèle qui associe à chaque formule ψ un préordre partiel $\leq_\psi$ tel que* $\text{Mod}(\psi - \mu) = \text{Mod}(\psi) \cup min_{\leq_\psi}(\text{Mod}(\neg\mu))$.

La preuve de ce thèorème suit exactement les mêmes lignes que la preuve du théorème 6.

5 Conclusion

Dans cet article consacré à l'effacement des croyances en logique propositionnelle nous poursuivons et complètons les travaux initiés par Katsuno et Mendelzon [14]. Ils ont formellement défini l'effacement des croyances dans un cadre sémantique et ont proposé un ensemble de postulats basiques. Nous proposons de nouveaux postulats capturant le principe de changement minimal pour l'effacement, plus précisément (E6) et (E7), qui permettent d'établir un premier théorème de représentation montrant qu'un opérateur d'effacement satisfaisant l'ensemble des postulats (E1)–(E8) se traduit par un préordre total sur les interprétations. De plus, le remplacement du postulat (E7) par deux postulats plus faibles (E9) et (E10) nous permet d'établir un deuxième théorème de représentation montrant qu'un opérateur de d'effacement satisfaisant l'ensemble des postulats (E1)–(E6) et (E8)–(E10) se traduit par un préordre partiel sur les interprétations.

De plus, pour l'opération de contraction, nous montrons qu'en remplaçant le postulat (C7) par deux postulats plus faibles (C8) et (C9), nous pouvons établir un théorème de représentation, qui jusqu'à ce jour manquait, montrant qu'un opérateur de contraction satisfaisant l'ensemble des (C1)–(C6) et (C8)–(C9) se traduit par un préordre partiel sur les interprétations.

Ainsi notre contribution permet de mettre une touche finale au panorama complet des opérations de révision, mise à jour, contraction et effacement dans un même cadre sémantique unifié, en termes de postulats et de théorèmes de représentation.

Une suite naturelle à ce travail serait l'étude de l'opération, appelée *Forget*, proposée par Winslett [21], qu'elle compare à la contraction. Soit ψ et μ deux formules propositionnelles et soit $\diamond$ un opérateur de mise à jour, l'opération de Forget est équivalente à $(\psi \diamond \mu) \vee (\psi \diamond \neg\mu)$.

Une autre perspective serait l'étude de la contraction et de l'effacement itérés. Alors que de nombreux travaux ont été développés sur la révision itérée suite aux travaux de Darwiche et Pearl [7], la contraction itérée n'a suscité que peu d'intérêt jusqu'à présent.

Par ailleurs, une étude plus ambitieuse serait l'étude de la complexité de problèmes de décision comme la vérification de modèles pour les opérateurs de contraction et d'effacement.

Références

[1] C.E. Alchourrón, P. Gärdenfors, and D. Makinson. On the logic of theory change : Partial meet contraction and revision functions. *Journal of Symbolic Logic*, 50 :510–530, 1985.

[2] C.E. Alchourrón and D. Makinson. On the logic of theory change : Safe contraction. *Studia Logica*, 44(4) :14–37, 1985.

[3] T. Caridroit, S. Konieczny, and P. Marquis. Contraction in propositional logic. In *Proceedings of ECSQARU'15*, pages 186–196, 2015.

[4] T. Caridroit, S. Konieczny, and P. Marquis. Contraction in propositional logic. *Int. J. Approx. Reason.*, 80 :428–442, 2017.

[5] N. Creignou, R. Ktari, and O. Papini. Belief contraction and erasure in fragments of propositional logic. *J. Log. Comput.*, 32(7) :1436–1468, 2022.

[6] M. Dalal. Investigations into a theory of knowledge base revision : preliminary report. In *Proceedings of AAAI'88*, pages 475–479, 1988.

[7] A. Darwiche and J. Pearl. On the logic of iterated belief revision. *Artif. Intell.*, 89(1-2) :1–29, 1997.

[8] T. Eiter and G. Gottlob. On the complexity of propositional knowledge base revision, updates, and counterfactuals. *Artificial Intelligence*, 57(2–3) :227–270, 1992.

[9] K. D. Forbus. Introducing actions into qualitative simulation. In *Proceedings International Joint Conferences on Artificial Intelligence Organization (IJCAI)*, pages 1273–1278, 1989.

[10] P. Gärdenfors. Knowledge in flux. In *Cambridge University Press, Cambridge UK*, 1988.

[11] A. Herzig and O. Rifi. Propositional belief base update and minimal change. *Artificial Intelligence*, 115(1) :107–138, 1999.

[12] H. Katsuno and A. O. Mendelzon. A unified view of propositional knowledge base updates. In *Proceedings of IJCAI'89*, pages 1413–1419, 1989.

[13] H. Katsuno and A.O. Mendelzon. Propositional knowledge base revision and minimal change. *Artificial Intelligence*, 52(3) :263–294, 1991.

[14] H. Katsuno and A.O. Mendelzon. On the difference between updating a knowledge base and revising it. In P. Gärdenfors, editor, *Belief revision*, pages 183–203. Cambridge University Press, 1992.

[15] A.M. Keller and M. Winslett. On the use of an extended relational model to handle changing incomplete information. *IEEE Trans. Software Eng.*, 11(7) :620–633, 1985.

[16] R. Ktari. *Changement de croyances dans des fragments de la logique propositionnelle*. PhD thesis, Aix-Marseille Université, 5 2016.

[17] P. Marquis, H. Prade, and O. Papini, editors. *Panorama de l'intelligence artificielle : Volume 1 : Représentation des connaissances et formalisation des raisonnements*. Cepaduès, 2014.

[18] P. Marquis, H. Prade, and O. Papini, editors. *A Guided Tour of Artificial Intelligence Research : Volume I : Knowledge Representation, Reasoning and Learning*. Springer, 2020.

[19] K. Satoh. Nonmonotonic reasoning by minimal belief revision. In *Proceedings of FGCS'88*, pages 455–462, Tokyo, 1988.

[20] M. Winslett. Reasoning about action using a possible models approach. In *Proc. AAAI*, pages 89–93, 1988.

[21] M. Winslett. Sometimes updates are circumscription. In *Proceedings International Joint Conferences on Artificial Intelligence Organization (IJCAI)*, pages 859–863, 1989.

Actes JIAF-JFPDA 2024

Appliquer la logique des variations propositionnelles à la représentation de règles d'adaptation pour le raisonnement à partir de cas *

Nicolas François[1] **Jean Lieber**[2]

1 Lycée public Chopin, Nancy
2 Université de Lorraine, CNRS, Inria, LORIA, Nancy
nicolas.francois@free.fr jean.lieber@loria.fr

Résumé

La logique des variations propositionnelles a été étudiée dans un article précédent. Ce formalisme est issu d'une notation issue du raisonnement à partir de cas (RàPC), avec l'ajout d'une sémantique. Cet article développe cette étude avec l'objectif de formaliser à l'aide de cette logique la notion de règles d'adaptation en RàPC et leur utilisation.

Abstract

The logic of propositional variations was studied in a previous article. This formalism is based on a notation derived from case-based reasoning (CBR), with the addition of a semantics. This article develops this study with the aim of using this logic to formalize the notion of adaptation rules in CBR and their use.

1 Introduction

Le raisonnement à partir de cas (RàPC [12]) consiste à résoudre un problème en s'appuyant sur une base de cas, où un cas est la représentation d'un épisode de résolution de problème. Le modèle de processus du RàPC consiste généralement, étant donné un problème cible (problème à résoudre) en la sélection dans la base de cas d'un (parfois de plusieurs) cas jugé(s) similaire(s) au problème cible (étape de *remémoration*) et dans la modification de ce (ou ces) cas dans l'optique de la résolution du problème cible (étape d'*adaptation*).

L'adaptation peut s'appuyer sur des règles d'adaptation indiquant comment une variation entre problèmes peut se répercuter en variations entre solutions. Des notations ont été introduites pour coder ces variations, en vue de processus d'apprentissages de connaissances d'adaptation (voir, par exemple, [4]). Ces notations ont été dotées d'une sémantique pour obtenir une logique, qui a été étudiée, pour elle-même, dans [7].

L'objectif de cet article est d'étudier des liens entre cette logique et l'adaptation en RàPC, avec l'idée que les notions relatives aux variations (et aux différences, et aux (dis)similarités), fréquemment utilisées dans ce domaine, puissent être représentées explicitement, i.e. qu'on puisse les coder et raisonner avec. Plus particulièrement, elle permet d'étudier la représentation de règles d'adaptation.

La section 2 décrit la logique des variations propositionnelles (et résume ainsi [7]) et les notations et notions générales relatives au RàPC. Quelques études sur cette logique se sont avérées nécessaires pour l'étude de son application au RàPC : elles sont introduites dans la section 3. La section 4 étudie comment les logiques de variations peuvent être utilisées pour représenter des connaissances d'adaptation. La section 5 présente une discussion en lien avec d'autres travaux.

Les preuves complètes des résultats de cet article se trouvent dans le rapport [8], version étendue de cet article.

2 Préliminaires

Ces préliminaires sont en deux parties : l'une concerne la logique $(\Delta\mathcal{LP}, \models)$, l'autre, le RàPC.

*Les auteurs tiennent à remercier Tiago de Lima qui leur a conseillé des lectures sur les logiques dynamiques qui ont des liens avec la logique des variations propositionnelles, ainsi que cela est mis en évidence dans la section 5.2 de cet article. Ils veulent également exprimer leur gratitude envers Pierre Marquis pour ses conseils judicieux. Que leurs chemins soient pavés de fleurs de cerisiers. Les auteurs tiennent également à remercier les relecteurs de cet article pour leurs remarques.

2.1 La logique des variations propositionnelles

Cette section reprend les points principaux sur la logique des variations propositionnelles $(\Delta\mathcal{LP}, \models)$ introduite et décrite dans [7].

La logique propositionnelle finie $(\mathcal{LP}, \models)$ est la logique de base sur laquelle va être définie $(\Delta\mathcal{LP}, \models)$.

On se donne un ensemble fini de symboles $\mathcal{V}$: un élément de $\mathcal{V}$ est appelé *variable propositionnelle* (ou simplement *variable*). Une formule de $(\mathcal{LP}, \models)$ est soit une variable propositionnelle, soit d'une des formes suivantes : $\neg\beta$, $\beta_1 \wedge \beta_2$ et $\beta_1 \vee \beta_2$, où β, β_1 et β_2 sont des formules propositionnelles. On utilisera les abréviations usuelles suivantes (où $\underline{euad}$ se lit « est une abréviation de », avec a, une variable choisie arbitrairement et $\beta_1, \beta_2 \in \mathcal{LP}$) :

$$\top \underline{euad}\ a \vee \neg a \qquad \bot \underline{euad}\ a \wedge \neg a$$
$$\beta_1 \rightarrow \beta_2 \underline{euad}\ \neg\beta_1 \vee \beta_2$$

Un littéral de $(\mathcal{LP}, \models)$ est une formule de la forme a ou de la forme $\neg a$ ($a \in \mathcal{V}$). Un cube de cette logique est une conjonction de littéraux dans laquelle une variable n'apparaît au plus qu'une fois. Un cube est complet si toutes les variables apparaissent dans ce cube.

L'ensemble des variables ayant des occurrences dans $\alpha \in \mathcal{LP}$ est noté $\mathcal{V}(\alpha)$.

On définit la sémantique en théorie des modèles comme suit. Une *interprétation* dans la logique propositionnelle est une fonction $\mathcal{I} : \mathcal{V} \rightarrow \{\mathbb{0}, \mathbb{1}\}$ où $\{\mathbb{0}, \mathbb{1}\}$ est l'ensemble des booléens. On note Ω l'ensemble des interprétations. Soit $\mathcal{I} \in \Omega$ et $\alpha \in \mathcal{LP}$. On définit la relation « $\mathcal{I}$ satisfait α » ($\mathcal{I} \models \alpha$, i.e. $\mathcal{I}$ est un modèle de α) de la façon suivante :

— $\mathcal{I} \models a$ si $\mathcal{I}(a) = 1$ pour une variable a ;
— $\mathcal{I} \models \neg\alpha$ si $\mathcal{I} \not\models \alpha$ pour $\alpha \in \mathcal{LP}$;
— $\mathcal{I} \models \alpha_1 \wedge \alpha_2$ si $\mathcal{I} \models \alpha_1$ et $\mathcal{I} \models \alpha_2$ pour $\alpha_1, \alpha_2 \in \mathcal{LP}$;
— $\mathcal{I} \models \alpha_1 \vee \alpha_2$ si $\mathcal{I} \models \alpha_1$ ou $\mathcal{I} \models \alpha_2$ pour $\alpha_1, \alpha_2 \in \mathcal{LP}$.

L'ensemble des modèles de α est noté $\mathcal{M}(\alpha)$. Si $\mathcal{B}$ est un ensemble fini de formules, $\mathcal{M}(\mathcal{B})$ est l'intersection des $\mathcal{M}(\alpha)$ pour $\alpha \in \mathcal{B}$, ce qui permet d'assimiler $\mathcal{B}$ à la conjonction $\bigwedge \mathcal{B}$ de ses éléments. On définit alors la relation $\mathcal{B} \models \beta$ ($\mathcal{B}$ *entraîne* β) par $\mathcal{M}(\mathcal{B}) \subseteq \mathcal{M}(\beta)$ et la relation $\alpha \models_{\mathcal{B}} \beta$ par $\mathcal{B} \cup \{\alpha\} \models \beta$. Une formule α est *satisfiable* si $\mathcal{M}(\alpha) \neq \emptyset$ (on le notera souvent $\alpha \not\models \bot$ dans l'article). Enfin, avec $\alpha, \beta \in \mathcal{LP}$, $\alpha \equiv \beta$ si $\mathcal{M}(\alpha) = \mathcal{M}(\beta)$.

La syntaxe de la logique $(\Delta\mathcal{LP}, \models)$ diffère légèrement de celle qui était donnée dans [7], au sens où certains constructeurs dans cet article sont ici des abréviations et inversement. Une formule $\varphi \in \Delta\mathcal{LP}$ est une expression d'une des formes suivantes : $\alpha \triangleright \beta$ (ce qu'on peut lire « α devient β »), $\neg\psi$, $\psi_1 \wedge \psi_2$ et $\psi_1 \vee \psi_2$, où $\alpha, \beta \in \mathcal{LP}$ et $\psi, \psi_1, \psi_2 \in \Delta\mathcal{LP}$.

On utilisera les abréviations suivantes :

$$\top_\Delta \underline{euad}\ \top \triangleright \top \qquad \bot_\Delta \underline{euad}\ \bot \triangleright \bot$$
$$\psi_1 \rightarrow \psi_2 \underline{euad}\ \neg\psi_1 \vee \psi_2$$
$$\alpha^{=\mathbb{1}} \underline{euad}\ \alpha \triangleright \alpha \qquad \alpha^{=\mathbb{0}} \underline{euad}\ \neg\alpha \triangleright \neg\alpha$$
$$\alpha^{+} \underline{euad}\ \neg\alpha \triangleright \alpha \qquad \alpha^{-} \underline{euad}\ \alpha \triangleright \neg\alpha$$
$$\alpha^{=} \underline{euad}\ \alpha^{=\mathbb{1}} \vee \alpha^{=\mathbb{0}} \qquad \alpha^{\neq} \underline{euad}\ \alpha^{+} \vee \alpha^{-}$$
$$\alpha^{\mathbb{0}\bullet} \underline{euad}\ \alpha^{=\mathbb{0}} \vee \alpha^{+} \qquad \alpha^{\mathbb{1}\bullet} \underline{euad}\ \alpha^{=\mathbb{1}} \vee \alpha^{-}$$
$$\alpha^{\bullet\mathbb{0}} \underline{euad}\ \alpha^{=\mathbb{0}} \vee \alpha^{-} \qquad \alpha^{\bullet\mathbb{1}} \underline{euad}\ \alpha^{=\mathbb{1}} \vee \alpha^{+}$$

Quand le contexte ne sera pas ambigu, $\top_\Delta$ et $\bot_\Delta$ seront notés simplement $\top$ et $\bot$. On note $\mathcal{D}_0 = \{\mathtt{=}\mathbb{1}, \mathtt{=}\mathbb{0}, \mathtt{+}, \mathtt{-}\}$ et ses éléments sont appelés *symboles de variations primitifs*. L'ensemble $\mathcal{D}_1 = \{\mathtt{=}, \mathtt{\neq}, \mathbb{0}\bullet, \mathbb{1}\bullet, \bullet\mathbb{0}, \bullet\mathbb{1}\}$ est celui des *symboles de variation non primitifs*. $\mathcal{D} = \mathcal{D}_0 \cup \mathcal{D}_1$ est donc l'ensemble des symboles de variations.

Un littéral de $(\Delta\mathcal{LP}, \models)$ est une formule de la forme a^v où $a \in \mathcal{V}$ et $v \in \mathcal{D}$. Un *cube* de cette logique est une conjonction de littéraux dans laquelle une variable n'apparaît au plus qu'une fois. Un cube est *primitif* si tous les symboles de variation qui y apparaissent sont primitifs. Un cube est *complet* s'il est primitif et si toutes les variables apparaissent dans ce cube.

Pour $\varphi \in \Delta\mathcal{LP}$, $x \in \mathcal{V}$ et $\alpha \in \mathcal{LP}$, $\varphi[x \backslash \alpha]$ est le résultat de la substitution de x par α dans φ. L'ensemble des variables ayant des occurrences dans φ est noté $\mathcal{V}(\varphi)$. La taille de φ, notée $|\varphi|$, est le nombre d'occurrences de connecteurs qu'elle contient.

Sémantique de $(\Delta\mathcal{LP}, \models)$**.** Soit $\Delta\Omega = \Omega \times \Omega$. Un élément $(\mathcal{I}, \mathcal{J})$ de $\Delta\Omega$ est appelé interprétation (pour cette logique) et sera noté simplement $\mathcal{I}\mathcal{J}$. Pour $\varphi \in \Delta\mathcal{LP}$, la relation $\mathcal{I}\mathcal{J} \models \varphi$ est définie comme suit :

— $\mathcal{I}\mathcal{J} \models \alpha \triangleright \beta$ si $\mathcal{I} \models \alpha$ et $\mathcal{J} \models \beta$.
— $\mathcal{I}\mathcal{J} \models \neg\psi$ si $\mathcal{I}\mathcal{J} \not\models \psi$.
— $\mathcal{I}\mathcal{J} \models \psi_1 \wedge \psi_2$ si $\mathcal{I}\mathcal{J} \models \psi_1$ et $\mathcal{I}\mathcal{J} \models \psi_2$.
— $\mathcal{I}\mathcal{J} \models \psi_1 \vee \psi_2$ si $\mathcal{I}\mathcal{J} \models \psi_1$ ou $\mathcal{I}\mathcal{J} \models \psi_2$.

On définit alors $\mathcal{M}(\varphi) = \{\mathcal{I}\mathcal{J} \in \Delta\Omega \mid \mathcal{I}\mathcal{J} \models \varphi\}$. On note en particulier que $\mathcal{M}(\alpha \triangleright \beta) = \mathcal{M}(\alpha) \times \mathcal{M}(\beta)$. On définit les notions de satisfiabilité, de conséquence logique ($\models$) et d'équivalence logique ($\equiv$) de la même façon qu'en logique propositionnelle.

Pour $\varphi \in \Delta\mathcal{LP}$, on notera $\mathsf{G}(\varphi)$ et $\mathsf{D}(\varphi)$ des formules propositionnelles (uniques à l'équivalence logique près) telles que

$$\mathcal{M}(\mathsf{G}(\varphi)) = \{\mathcal{I} \mid \mathcal{I}\mathcal{J} \in \mathcal{M}(\varphi)\}$$
$$\mathcal{M}(\mathsf{D}(\varphi)) = \{\mathcal{J} \mid \mathcal{I}\mathcal{J} \in \mathcal{M}(\varphi)\}$$

En particulier, pour $\alpha, \beta \in \mathcal{LP}$, $\mathsf{G}(\alpha \triangleright \beta) \equiv \alpha$ et $\mathsf{D}(\alpha \triangleright \beta) \equiv \beta$ (G comme « gauche », D comme « droite »).

Quelques résultats. Les points suivants résument et complètent certains résultats de [7] :

- Tout sous-ensemble A de $\Delta\Omega$ est *représentable* dans la logique : il existe $\varphi \in \Delta\mathcal{LP}$ telle que $\mathcal{M}(\varphi) = A$.
- On a la plupart des résultats classiques de la logique propositionnelle : théorème de la déduction, lois de De Morgan, formes normales conjonctives et disjonctives, etc.
- Toute formule φ peut se mettre sous la forme d'une formule obtenue à partir d'une formule de logique propositionnelle, en substituant à ses variables des formules de la forme a^v où $a \in \mathcal{V}$ et $v \in \mathcal{D}_0$. Par exemple :

$$a \wedge b \triangleright \neg a \wedge c \quad \equiv \quad a^{-} \wedge (b^{=1} \vee b^{-}) \wedge (c^{=1} \vee c^{+})$$

- En particulier, on peut écrire toute formule de $(\Delta\mathcal{LP}, \models)$ sous forme normale disjonctive (FND), i.e. sous la forme de disjonction de cubes.
- On a la propriété de distributivité de $\triangleright$ sur $\wedge$ (modulo l'équivalence) :

$$(\alpha_1 \wedge \alpha_2) \triangleright \beta \equiv (\alpha_1 \triangleright \beta) \wedge (\alpha_2 \triangleright \beta)$$
$$\alpha \triangleright (\beta_1 \wedge \beta_2) \equiv (\alpha \triangleright \beta_1) \wedge (\alpha \triangleright \beta_2)$$

  pour $\alpha, \alpha_1, \alpha_2, \beta, \beta_1, \beta_2 \in \mathcal{LP}$. On a également la distributivité de $\triangleright$ sur $\vee$.
- Pour $a \in \mathcal{V}$, soit $\ell(a)$ et $m(a)$ deux littéraux construits sur a ($\ell(a), m(a) \in \{a, \neg a\}$). On a :

$$\bigwedge_{a\in\mathcal{V}} \ell(a) \triangleright \bigwedge_{a\in\mathcal{V}} m(a) \quad \equiv \quad \bigwedge_{a\in\mathcal{V}} \ell(a) \triangleright m(a) \quad (1)$$

  et chaque $\ell(a) \triangleright m(a)$ peut s'écrire a^v où $v \in \mathcal{D}_0$.
- On peut définir un système formel correct et complet pour $(\Delta\mathcal{LP}, \models)$.
- Le problème de décision $\mathcal{B} \models \varphi$ est NP-complet et plusieurs approches pour l'implanter ont été envisagées.
- On a le résultat suivant (avec $\varphi, \psi \in \Delta\mathcal{LP}$) :

$$\mathsf{D}(\varphi \vee \psi) \equiv \mathsf{D}(\varphi) \vee \mathsf{D}(\psi) \quad (2)$$

- Pour $\varphi \in \Delta\mathcal{LP}$, on peut définir facilement $\varphi^{-1} \in \Delta\mathcal{LP}$ telle que $\mathcal{IJ} \models \varphi^{-1}$ ssi $\mathcal{JI} \models \varphi$ (il suffit d'inverser les paramètres de $\triangleright$).

2.2 Rappels sur le raisonnement à partir de cas

Généralités. Le RàPC est un type de raisonnement s'appuyant sur une base de cas où un cas est une représentation d'un épisode de résolution de problème.

Un domaine d'application pour le RàPC est donné par un triplet $(\mathcal{P}, \mathcal{S}, \rightsquigarrow)$ où $\mathcal{P}$ et $\mathcal{S}$ sont des ensembles et $\rightsquigarrow$ est une relation binaire sur $\mathcal{P} \times \mathcal{S}$. Un élément de $\mathtt{x}$ est appelé *problème*, un élément de $\mathtt{y}$, *solution* et $\mathtt{x} \rightsquigarrow \mathtt{y}$ se lit « $\mathtt{x}$ a pour solution $\mathtt{y}$ ».

En général, la relation $\rightsquigarrow$ n'est pas connue du système de RàPC dont l'objectif est de construire une *hypothèse de solution* $\mathtt{y}^{\mathtt{cible}}$ à un problème donné, le *problème cible*, noté $\mathtt{x}^{\mathtt{cible}}$.

On considère souvent (et cela sera le cas dans cet article) qu'un cas est donné par un couple $(\mathtt{x}, \mathtt{y}) \in \mathcal{P}\times\mathcal{S}$ où $\mathtt{x} \rightsquigarrow \mathtt{y}$: la relation $\rightsquigarrow$ est connue pour un ensemble fini qui constitue la *base de cas*. On note $(\mathtt{x}^s, \mathtt{y}^s)$ un élément de la base de cas et on l'appelle *cas source* ($\mathtt{x}^s$ est un *problème source*).

Le processus de RàPC le plus courant consiste en une étape de remémoration (sélection de k éléments de la base de cas jugés similaires au problème cible) et d'adaptation (réutilisation des cas remémorés pour résoudre $\mathtt{x}^{\mathtt{cible}}$). Dans cet article, ne seront considérés que des processus de remémoration et d'adaptation simples, i.e. $k = 1$. Pour l'adaptation, cela signifie qu'un *problème d'adaptation* sera donné par un cas source et un problème cible : on le notera $((\mathtt{x}^s, \mathtt{y}^s), \mathtt{x}^{\mathtt{cible}})$.

L'approche de l'adaptation considérée dans cet article, parfois appelée *adaptation transformationnelle* (voir [2]) vise à résoudre le problème d'adaptation en le lisant sous la forme « La solution cherchée, $\mathtt{y}^{\mathtt{cible}}$, est à $\mathtt{y}^s$ ce que $\mathtt{x}^{\mathtt{cible}}$ est à $\mathtt{x}^s$. » On peut l'interpréter ainsi : des variations entre les problèmes $\mathtt{x}^s$ et $\mathtt{x}^{\mathtt{cible}}$, notées $\Delta\mathtt{x}$, on infère des variations entre la solution $\mathtt{y}^s$ et l'inconnue $\mathtt{y}^{\mathtt{cible}}$, notée $\Delta\mathtt{y}$, puis, on infère $\mathtt{y}^{\mathtt{cible}}$ à partir de $\mathtt{y}^s$ et $\Delta\mathtt{y}$.

Modèle de connaissances du RàPC. On considère généralement qu'une base de connaissances d'un système de RàPC est constitué de quatre « conteneurs de connaissances » : la base de cas BC, les connaissances du domaine CD, les connaissances de remémoration CR et les connaissances d'adaptation CA [11]. BC et CA ont été évoqués ci-dessus. CR est souvent représenté par une mesure de similarité ou une distance entre problèmes. CD peut être vu comme une conjonction de contraintes d'intégrité qui donnent des conditions nécessaires pour qu'un cas soit licite (exemple en cuisine : « Les salsifis, c'est pas bon [1]. »). Les inférences dans le formalisme représentant les cas se feront en général sur la base de ces connaissances du domaine (on utilisera la relation $\models_{\mathtt{CD}}$ plutôt que simplement $\models$).

Représentation des cas en logique propositionnelle. Pour certaines applications, on peut représenter les cas dans $(\mathcal{LP}, \models)$: on considère un partitionnement de $\mathcal{V}$ en $\{\mathcal{V}_\mathcal{P}, \mathcal{V}_\mathcal{S}\}$ et un problème $\mathtt{x}$ (resp. une solution $\mathtt{y}$) sera représenté(e) par une formule dont les variables appartiennent à $\mathcal{V}_\mathcal{P}$ (resp. à $\mathcal{V}_\mathcal{S}$). Un cas source $(\mathtt{x}^s, \mathtt{y}^s)$ sera représenté par la conjonction $\mathtt{x}^s \wedge \mathtt{y}^s$ ([2]).

1. Cette connaissance du domaine, quelque peu subjective, nous a été fournie par une des sœurs d'un des auteurs.

2. Parfois, comme on le verra pour l'exemple suivi, le partitionnement entre variables problèmes et variables solutions se fait une fois le problème d'adaptation spécifié : connaissant $\mathtt{x}^{\mathtt{cible}}$ et le cas source $\mathtt{c}^s$, on détermine

Dans beaucoup d'applications, les cas sources et le problème cible sont considérés comme *spécifiques*, i.e. issus d'une expérience particulière complètement instanciée. Dans ce cas, pour un cas source $(\mathbf{x}^s, \mathbf{y}^s)$ et un problème cible $\mathbf{x}^{\mathtt{cible}}$, les problèmes $\mathbf{x}^s$ et $\mathbf{x}^{\mathtt{cible}}$ peuvent s'écrire sous la forme $\bigwedge_{a\in\mathcal{V}_\mathcal{P}} \ell(a)$ et la solution $\mathbf{y}^s$ sous la forme $\bigwedge_{a\in\mathcal{V}_\mathcal{S}} \ell(a)$, où $\ell(a) \in \{a, \neg a\}$. La formule $\mathbf{x}^s \wedge \mathbf{y}^s$ est alors un cube complet et donc $|\mathcal{M}(\mathbf{x}^s \wedge \mathbf{y}^s)| = 1$.

L'apprentissage de connaissances d'adaptation. L'adaptation peut s'appuyer sur des connaissances d'adaptation. Ces connaissances peuvent être acquises auprès d'un expert (voir p. ex. [5]). Elles peuvent être aussi apprises à partir de la base de cas, selon le principe appelé *difference heuristics* dans [10] et défini initialement dans [9]. Ce principe s'inscrit dans le cadre de l'adaptation transformationnelle : étant donné deux cas sources différents $(\mathbf{x}^i, \mathbf{y}^i)$ et $(\mathbf{x}^j, \mathbf{y}^j)$, on construit les variations $\Delta\mathbf{x}^{ij}$ de $\mathbf{x}^i$ à $\mathbf{x}^j$ et $\Delta\mathbf{y}^{ij}$ de $\mathbf{y}^i$ à $\mathbf{y}^j$. L'apprentissage de connaissances d'adaptation utilise un ensemble de couples $(\Delta\mathbf{x}^{ij}, \Delta\mathbf{y}^{ij})$ et permet de construire un modèle d'une fonction $\Delta\mathbf{x} \mapsto \Delta\mathbf{y}$, qui permet, à partir d'une variation entre problèmes d'obtenir une variation entre solution, ce qui est utile pour un processus d'adaptation transformationnelle.

Des travaux dans ce cadre se sont appuyés sur une notation a^v, où a est un attribut d'un problème ou d'une solution et v représente une relation entre valeurs du domaine de cet attribut. Cette notation permet de coder ces variations, dans une optique d'apprentissage de règles d'adaptation, utilisant des techniques telles que l'extraction de motifs fermés fréquents [4]. Par exemple, $\hat{\mathtt{age}}^{\mathtt{ajouter}(5)}$ indique une variation de 5 pour l'attribut $\hat{\mathtt{age}}$. Le résultat d'une extraction de motifs fermés fréquents avec des $\Delta\mathbf{x}^{ij}$ et des $\Delta\mathbf{y}^{ij}$ est un ensemble de motifs, chaque motif étant un ensemble d'expressions a^v. En se limitant à des attributs booléens, les a^v correspondaient dans ce travail à ceux de la section 2.1 : c'est ce travail qui a d'ailleurs motivé au départ l'étude de la logique $(\Delta\mathcal{LP}, \models)$.

Plus précisément, si les cas sources sont supposés spécifiques, d'après le résultat (1), on peut écrire $\Delta\mathbf{x}^{ij}$ et $\Delta\mathbf{y}^{ij}$ sous les formes $\bigwedge_{a\in\mathcal{V}_\mathcal{P}} a^{v(a)}$ et $\bigwedge_{a\in\mathcal{V}_\mathcal{S}} a^{v(a)}$, où $v(a) \in \mathcal{D}_0$. Un algorithme de recherche de motifs fréquents permettra alors de calculer des motifs M, ensemble d'éléments de la forme a^v, motifs qui peuvent être interprétés en règles d'adaptation.

3 Nouveaux résultats sur $(\Delta\mathcal{LP}, \models)$

Cette section introduit et étudie des notions relatives à la logique $(\Delta\mathcal{LP}, \models)$ utiles pour la suite de l'article et qui étaient peu ou pas introduits dans [7].

$\mathcal{V}_\mathcal{P}$ et $\mathcal{V}_\mathcal{S}$ pour décomposer $\mathbf{c}^s$ en $\mathbf{x}^s \wedge \mathbf{y}^s$.

3.1 Résultats généraux

Le premier résultat énonce le fait que seules les variables apparaissant dans une formule de $(\Delta\mathcal{LP}, \models)$ ont une influence dans les inférences déductives (ce qui peut sembler une évidence mais mérite peut-être d'être énoncé) :

$$\begin{aligned} &\text{si } \mathcal{I}_1\mathcal{J}_1, \mathcal{I}_2\mathcal{J}_2 \in \Delta\Omega \text{ vérifient} \\ &\quad \mathcal{I}_1(x) = \mathcal{I}_2(x) \text{ et } \mathcal{J}_1(x) = \mathcal{J}_2(x) \quad \text{pour } x \in \mathcal{V}(\varphi) \\ &\text{alors } \mathcal{I}_1\mathcal{J}_1 \models \varphi \text{ ssi } \mathcal{I}_2\mathcal{J}_2 \models \varphi \end{aligned} \tag{3}$$

pour $\varphi \in \Delta\mathcal{LP}$. La preuve de ce résultat se fait par induction structurelle.

Le résultat (2) ne se généralise pas en remplaçant $\vee$ par $\wedge$. Cependant, on a le résultat suivant :

$$\begin{aligned} &\text{si } \varphi, \psi \in \Delta\mathcal{LP} \text{ vérifient } \mathcal{V}(\varphi) \cap \mathcal{V}(\psi) = \emptyset \\ &\text{alors } \mathsf{G}(\varphi \wedge \psi) \equiv \mathsf{G}(\varphi) \wedge \mathsf{G}(\psi) \\ &\quad \text{et } \mathsf{D}(\varphi \wedge \psi) \equiv \mathsf{D}(\varphi) \wedge \mathsf{D}(\psi) \end{aligned} \tag{4}$$

Preuve. La preuve ne sera faite que pour l'opérateur D (elle se transpose immédiatement pour l'opérateur G). Soit $\mathcal{J} \in \mathcal{M}(\mathsf{D}(\varphi \wedge \psi))$. Il existe donc $\mathcal{I} \in \Omega$ telle que $\mathcal{I}\mathcal{J} \models \varphi \wedge \psi$. Donc, il existe $\mathcal{I} \in \Omega$ telle que $\mathcal{I}\mathcal{J} \models \varphi$. Donc $\mathcal{J} \models \mathsf{D}(\varphi)$. Le même raisonnement conduit à $\mathcal{J} \models \mathsf{D}(\psi)$. Donc $\mathcal{J} \models \mathsf{D}(\varphi) \wedge \mathsf{D}(\psi)$. Par conséquent, $\mathsf{D}(\varphi \wedge \psi) \models \mathsf{D}(\varphi) \wedge \mathsf{D}(\psi)$.

Pour ce sens direct, l'hypothèse $\mathcal{V}(\varphi) \cap \mathcal{V}(\psi) = \emptyset$ n'était pas utile. Elle l'est pour la réciproque.

Supposons que $\mathcal{J} \models \mathsf{D}(\varphi) \wedge \mathsf{D}(\psi)$. Par conséquent, il existe $\mathcal{I}_1 \in \Omega$ telle que $\mathcal{I}_1\mathcal{J} \models \varphi$. De même, il existe $\mathcal{I}_2 \in \Omega$ telle que $\mathcal{I}_2\mathcal{J} \models \psi$. Comme $\mathcal{V}(\varphi) \cap \mathcal{V}(\psi) = \emptyset$ on peut définir $\mathcal{I} \in \Omega$ par

$$\mathcal{I}(x) = \begin{cases} \mathcal{I}_1(x) & \text{si } x \in \mathcal{V}(\varphi) \\ \mathcal{I}_2(x) & \text{si } x \in \mathcal{V}(\psi) \\ 1 & \text{sinon} \end{cases} \qquad (\text{pour tout } x \in \mathcal{V})$$

D'après le résultat (3), $\mathcal{I}_1\mathcal{J} \models \varphi$ entraîne $\mathcal{I}\mathcal{J} \models \varphi$ (puisque pour $x \in \mathcal{V}(\varphi)$, $\mathcal{I}_1(x) = \mathcal{I}(x)$ et, évidemment, $\mathcal{J}(x) = \mathcal{J}(x)$). De la même façon, on prouve que $\mathcal{I}\mathcal{J} \models \psi$. Donc, $\mathcal{I}\mathcal{J} \models \varphi \wedge \psi$ et, par conséquent, $\mathcal{J} \models \mathsf{D}(\varphi \wedge \psi)$. On en déduit que $\mathsf{D}(\varphi) \wedge \mathsf{D}(\psi) \models \mathsf{D}(\varphi \wedge \psi)$ ce qui conclut la preuve. ■

3.2 Caractérisation alternative de $\mathcal{B} \models \varphi$

La relation $\models$ entre une base de connaissances $\mathcal{B}$ et une formule φ de la logique des variations propositionnelles peut être redéfinie en s'appuyant sur quatre « valeurs de vérité » sur les variations, données par les éléments de $\mathcal{D}_0 = \{\mathtt{=1}, \mathtt{=0}, \mathtt{+}, \mathtt{-}\}$.

Soit $\Delta\Omega_{\text{alt}}$ l'ensemble des fonctions $\omega : \mathcal{V} \to \mathcal{D}_0$. On définit la relation $\models_{\text{alt}}$ entre $\omega \in \Delta\Omega_{\text{alt}}$ et $\varphi \in \Delta\mathcal{LP}$ par :

$$\omega \models_{\text{alt}} \varphi \text{ si } \bigwedge_{a\in\mathcal{V}} a^{\omega(a)} \models \varphi$$

On note alors $\mathcal{M}_{\text{alt}}(\varphi) = \{\omega \in \Delta\Omega_{\text{alt}} \mid \omega \models_{\text{alt}} \varphi\}$. Par exemple, si $\mathcal{V} = \{a, b\}$, $\mathcal{M}_{\text{alt}}(a^{=} \wedge b^{+}) = \{\omega_1, \omega_2\}$ où $\omega_1(a) = \texttt{=1}$, $\omega_2(a) = \texttt{=0}$ et $\omega_1(b) = \omega_2(b) = \texttt{+}$. Pour une base de connaissances $\mathcal{B}$, on notera $\mathcal{M}_{\text{alt}}(\mathcal{B}) = \bigcap_{\varphi \in \mathcal{B}} \mathcal{M}_{\text{alt}}(\varphi)$. On définit alors la relation $\models_{\text{alt}}$ entre une base de connaissances $\mathcal{B}$ et une formule φ par $\mathcal{B} \models_{\text{alt}} \varphi$ si $\mathcal{M}_{\text{alt}}(\mathcal{B}) \subseteq \mathcal{M}_{\text{alt}}(\varphi)$.

Cette relation $\models_{\text{alt}}$ ne définit pas une nouvelle sémantique de la logique des variations comme le montre la proposition ci-dessous ; elle permet juste une caractérisation alternative de cette sémantique.

Proposition 1. *Pour toute base de connaissances $\mathcal{B}$ de $(\Delta\mathcal{LP}, \models)$ et toute $\varphi \in \Delta\mathcal{LP}$, $\mathcal{B} \models_{alt} \varphi$ ssi $\mathcal{B} \models \varphi$.*

Preuve. À $\mathcal{I}\mathcal{J} \in \Delta\Omega$, on associe $\omega_{\mathcal{I}\mathcal{J}} \in \Delta\Omega_{\text{alt}}$ de la façon suivante, pour $a \in \mathcal{V}$

$$\omega_{\mathcal{I}\mathcal{J}}(a) = \begin{cases} \texttt{=1} & \text{si } \mathcal{I}(a) = \mathcal{J}(a) = 1 \\ \texttt{=0} & \text{si } \mathcal{I}(a) = \mathcal{J}(a) = \texttt{0} \\ \texttt{+} & \text{si } \mathcal{I}(a) = \texttt{0} \text{ et } \mathcal{J}(a) = 1 \\ \texttt{-} & \text{si } \mathcal{I}(a) = 1 \text{ et } \mathcal{J}(a) = \texttt{0} \end{cases}$$

La fonction $\mathcal{I}\mathcal{J} \in \Delta\Omega \mapsto \omega_{\mathcal{I}\mathcal{J}} \in \Delta\Omega_{\text{alt}}$ est une bijection dont la fonction inverse est définie ainsi : à $\omega \in \Delta\Omega_{\text{alt}}$ elle associe $\mathcal{I}\mathcal{J} \in \Delta\Omega$ définie ainsi, pour $a \in \mathcal{V}$:

$$\mathcal{I}(a) = \begin{cases} \texttt{0} & \text{si } \omega(a) \in \{\texttt{=0}, \texttt{+}\} \\ 1 & \text{sinon} \end{cases}$$

$$\mathcal{J}(a) = \begin{cases} \texttt{0} & \text{si } \omega(a) \in \{\texttt{=0}, \texttt{-}\} \\ 1 & \text{sinon} \end{cases}$$

(on montre que c'est une bijection en utilisant le fait que $|\Delta\Omega_{\text{alt}}| = |\Delta\Omega| = 4^{|\mathcal{V}|}$ et en vérifiant que, pour $\omega \in \Delta\Omega_{\text{alt}}$, en y associant $\mathcal{I}\mathcal{J}$ comme ci-dessus, on a $\omega_{\mathcal{I}\mathcal{J}} = \omega$).

On montre ensuite que pour $\mathcal{I}\mathcal{J} \in \Delta\Omega$ et $\varphi \in \Delta\mathcal{LP}$ on a $\mathcal{I}\mathcal{J} \models \varphi$ ssi $\omega_{\mathcal{I}\mathcal{J}} \models_{\text{alt}} \varphi$.

Enfin, pour une base de connaissances $\mathcal{B}$ et $\varphi \in \Delta\mathcal{LP}$, on montre que $\mathcal{B} \models \varphi$ ssi $\mathcal{B} \models_{\text{alt}} \varphi$ en s'appuyant sur la bijection introduite ci-dessus et l'équivalence entre satisfaction dans $\Delta\Omega$ et dans $\Delta\Omega_{\text{alt}}$. ■

3.3 Variables invariantes d'une formule de variations

Soit $\varphi \in \Delta\mathcal{LP}$, on s'intéresse (et ce sera justifié par la suite) aux variables a dont le changement laisse φ inchangée. On va définir $\mathsf{Inv}(\varphi)$ en s'appuyant sur la définition alternative de la sémantique présentée à la section précédente. L'idée est que $a \in \mathsf{Inv}(\varphi)$ signifie que la valeur d'$\omega(a)$ n'influe pas sur la satisfaction de φ par ω. Formellement, cela mène à la définition suivante :

Définition 1. *Pour $\omega \in \Delta\Omega_{alt}$, $a \in \mathcal{V}$ et $v \in \mathcal{D}_0$, on définit $\omega[a^v] \in \Delta\Omega_{alt}$ par*

$$\omega[a^v] : x \in \mathcal{V} \mapsto \begin{cases} v & \textit{si } x = a \\ \omega(x) & \textit{sinon} \end{cases}$$

Pour $\varphi \in \Delta\mathcal{LP}$, l'ensemble des variables invariantes de φ est

$$\mathsf{Inv}(\varphi) = \left\{ a \in \mathcal{V} \;\middle|\; \begin{array}{l} \textit{pour toute } \omega \in \mathcal{M}_{alt}(\varphi) \\ \textit{et tout } v \in \mathcal{D}_0,\ \omega[a^v] \models_{alt} \varphi \end{array} \right\}$$

Par exemple,

$$\text{si } \mathcal{V} = \{a, b, c\} \text{ et } \varphi = (a^{+} \wedge b^{=}) \vee (a^{+} \wedge b^{+}) \vee (a^{+} \wedge b^{-}) \\ \text{alors } \mathsf{Inv}(\varphi) = \{b, c\} \tag{5}$$

Cette notion est indépendante de la syntaxe : si $\varphi \equiv \psi$ alors $\mathsf{Inv}(\varphi) = \mathsf{Inv}(\psi)$. On montre, en s'appuyant sur (3) que si une variable x n'apparaît pas dans φ alors elle est invariante. Donc :

$$\mathcal{V} \setminus \mathcal{V}(\varphi) \subseteq \mathsf{Inv}(\varphi) \subseteq \mathcal{V} \tag{6}$$

Comme le montre l'exemple (5), une variable apparaissant dans φ peut très bien être invariante pour φ.

Le cas particulier de la proposition suivante est utile dans la suite de l'article.

Proposition 2. *Si φ est un cube de $(\Delta\mathcal{LP}, \models)$ alors $\mathsf{Inv}(\varphi) = \mathcal{V} \setminus \mathcal{V}(\varphi)$.*

Preuve. Comme $\mathsf{Inv}(\varphi) \supseteq \mathcal{V} \setminus \mathcal{V}(\varphi)$ (cf. (6)), il suffit de montrer que si $a \in \mathsf{Inv}(\varphi)$ alors $a \notin \mathcal{V}(\varphi)$, ce qu'on va montrer par contraposée. Soit $a \in \mathcal{V}(\varphi)$. Soit $v \in \mathcal{D}$ tel que a^v est l'atome de φ ayant a comme variable. Soit $v_0 \in \mathcal{D}_0$ tel que $a^{v_0} \models a^v$ (si v est un symbole de variation primitif, $v_0 = v$, sinon, il y a plusieurs choix possibles pour v_0, p. ex., si $v = \neq$ alors on peut choisir $v_0 = \texttt{+}$ ou $v_0 = \texttt{-}$). Soit $w \in \mathcal{D}_0 \setminus \{v_0\}$. On a donc $\varphi[a^{v_0}] \models a^{v_0}$ et $\varphi[a^w] \not\models a^{v_0}$ (puisque, par hypothèse sur les cubes, une variable n'apparaît qu'une seule fois dans un cube et donc, les cubes sont satisfiables). Par conséquent, $\varphi[a^{v_0}] \not\equiv \varphi[a^w]$ et donc $a \notin \mathsf{Inv}(\varphi)$. ■

3.4 Opérateur *ceteris paribus*

Considérons la formule a^{+} : elle indique que a « passe » de $\texttt{0}$ à 1, les autres variables étant libres de suivre toutes les variations. Ainsi, si $\mathcal{V} = \{a, b\}$, $\mathcal{M}(a^{+}) = \{(\overline{a}\,\overline{b}, a\overline{b}), (\overline{a}\,\overline{b}, \overline{a}b), (\overline{a}b, a\overline{b}), (\overline{a}b, ab)\}$. Il apparaît utile dans certains cas d'exprimer par exemple que a « passe » de $\texttt{0}$ à 1, le reste des variables restant *inchangées*. On va exprimer cela par une formule $\mathsf{cp}(a^{+})$ qui, sur l'exemple avec deux variables, donnera $\mathcal{M}(\mathsf{cp}(a^{+})) = \{(\overline{a}\,\overline{b}, a\overline{b}), (\overline{a}b, ab)\}$: dans les modèles de ces formules, b reste à $\texttt{0}$ ou reste à 1 (cp pour *ceteris paribus*), en d'autres termes, $\mathsf{cp}(a^{+}) \models b^{=}$. Dans cet exemple, b est une variable dont l'interprétation ne change pas l'interprétation de φ : $b \in \mathsf{Inv}(\varphi)$.

Plus généralement, on cherche un opérateur cp tel que, pour $\varphi, \psi \in \Delta\mathcal{LP}$, on ait

$$\mathsf{cp}(\varphi) \models \varphi \tag{7}$$

$$\text{pour toute } x \in \mathsf{Inv}(\varphi),\ \mathsf{cp}(\varphi) \models x^{=} \tag{8}$$

$$\text{si } \varphi \equiv \psi \text{ alors } \mathsf{cp}(\varphi) \equiv \mathsf{cp}(\psi) \tag{9}$$

$$\text{si } \varphi \text{ est satisfiable alors } \mathsf{cp}(\varphi) \text{ l'est aussi} \tag{10}$$

On propose la définition suivante de cp et on prouve ensuite qu'elle vérifie ces propriétés.

Définition 2. cp *est la fonction de* $\Delta\mathcal{LP}$ *dans lui-même définie, pour* $\varphi \in \Delta\mathcal{LP}$ *par*

$$\mathsf{cp}(\varphi) = \varphi \wedge \bigwedge \{x^{=} \mid x \in \mathsf{Inv}(\varphi)\}$$

Proposition 3. cp *vérifie (7), (8), (9) et (10).*

Preuve. (7), (8) et (9) découlent immédiatement de la définition.

Pour (10), on le prouve de la façon suivante. Supposons que φ soit satisfiable. Elle l'est donc aussi au sens de la définition alternative de la sémantique et donc, il existe $\omega \in \Delta\Omega_{\text{alt}}$ telle que $\omega \models_{\text{alt}} \varphi$. Soit a_1, a_2, ..., a_p telles que $\mathsf{Inv}(\varphi) = \{a_1, a_2, \ldots, a_p\}$. Soit $\omega' = \omega[a_1^{=1}][a_2^{=1}]\ldots[a_p^{=1}]$. On a, par définition de $\mathsf{Inv}(\varphi)$, $\omega' \models_{\text{alt}} \varphi$. Par ailleurs, $\omega' \models_{\text{alt}} a_k^{=1}$ pour tout $k \in \{1, 2, \ldots, p\}$ et donc $\omega' \models_{\text{alt}} \bigwedge\{x^{=1} \mid x \in \mathsf{Inv}(\varphi)\}$. Or, pour toute variable a, $a^{=1} \models a^{=}$. Par conséquent, $\omega' \models_{\text{alt}} \bigwedge\{x^{=} \mid x \in \mathsf{Inv}(\varphi)\}$. Donc, $\omega' \models_{\text{alt}} \mathsf{cp}(\varphi)$ et donc, $\mathsf{cp}(\varphi)$ est satisfiable. ■

Une conséquence immédiate de la proposition 2 est la suivante :

$$\text{si } \varphi \text{ est un cube alors } \mathsf{cp}(\varphi) = \varphi \wedge \bigwedge_{x \in \mathcal{V} \setminus \mathcal{V}(\varphi)} x^{=} \tag{11}$$

Par ailleurs, les modèles de $\mathsf{cp}(\top)$ sont les $\mathcal{II}$ pour $\mathcal{I} \in \Omega$.

Notons que cp est non monotone, dans le sens où on peut avoir $\varphi \models \psi$ et $\mathsf{cp}(\varphi) \not\models \mathsf{cp}(\psi)$. Par exemple, $a^{+} \models \top$ et $\mathsf{cp}(a^{+}) \not\models \mathsf{cp}(\top)$.

Un problème pratique se pose avec cet opérateur : pour $\varphi \in \Delta\mathcal{LP}$, $|\mathsf{cp}(\varphi)|$ est en $O(|\varphi| + n)$ et la manipulation de formules longues prend du temps de calcul. Ce problème devient théorique si on étend le cadre logique à un ensemble de variables infini : la définition de cp ci-dessus ne s'applique pas, les formules étant nécessairement de tailles finies. Une perspective pour pallier ce problème serait l'étude d'une possible extension de la logique par un *connecteur* cp (plutôt que par un opérateur).

Il apparaît parfois utile de restreindre le *ceteris paribus* à un ensemble prédéfini de variables, d'où la définition suivante.

Définition 3. *Soit* $\mathcal{W} \subseteq \mathcal{V}$ *et* $\varphi \in \Delta\mathcal{LP}$. *Le* ceteris paribus *de* φ *restreint à l'ensemble de variables* $\mathcal{W}$ *est :*

$$\mathsf{cp}_{\mathcal{W}}(\varphi) = \varphi \wedge \bigwedge \{x^{=} \mid x \in \mathsf{Inv}(\varphi) \cap \mathcal{W}\}$$

Et on a, évidemment, $\mathsf{cp}_{\mathcal{V}}(\varphi) = \mathsf{cp}(\varphi)$.

3.5 Composition des variations

La composition sur $(\Delta\mathcal{LP}, \models)$ a été introduite dans [7] mais peu étudiée. Comme une formule $\varphi \in \Delta\mathcal{LP}$ représente un sous-ensemble $\mathcal{M}(\varphi)$ de $\Delta\Omega = \Omega \times \Omega$, on peut la considérer comme une représentation d'une relation binaire sur Ω. Les relations binaires sur un ensemble peuvent se composer et la composition des formules de variations correspond à la composition entre relations binaires.

Plus formellement, soit $\varphi, \psi \in \Delta\mathcal{LP}$. La composition de φ et ψ est une formule $\varphi \,;\, \psi$ définie à la syntaxe près par

$$\mathcal{M}(\varphi \,;\, \psi) = \left\{ \mathcal{IK} \in \Delta\Omega \;\middle|\; \begin{array}{l}\text{il existe } \mathcal{J} \in \Omega \text{ telle que} \\ \mathcal{IJ} \models \varphi \text{ et } \mathcal{JK} \models \psi\end{array} \right\}$$

Le résultat suivant est une conséquence directe de l'associativité de la composition de relations binaires :

$$\text{pour } \varphi, \psi, \chi \in \Delta\mathcal{LP},\ (\varphi \,;\, \psi) \,;\, \chi \equiv \varphi \,;\, (\psi \,;\, \chi)$$

Cela permet d'omettre des parenthèses dans les compositions.

La recherche d'une formule équivalente à $\varphi \,;\, \psi$ va être étudiée via les propositions ci-dessous.

On commence par le cas particulier où les formules à composer sont de la forme $\alpha \triangleright \beta$.

Proposition 4. *Soit* $\alpha, \beta, \gamma, \delta \in \mathcal{LP}$. *On a :*

$$(\alpha \triangleright \beta) \,;\, (\gamma \triangleright \delta) \equiv \begin{cases} \bot & \text{si } \beta \wedge \gamma \models \bot \\ \alpha \triangleright \delta & \text{sinon} \end{cases}$$

On s'intéresse ensuite aux liens entre composition et disjonction :

Proposition 5. *Soit* $\varphi, \varphi_1, \varphi_2, \psi, \psi_1, \psi_2 \in \Delta\mathcal{LP}$. *On a les équivalences suivantes :*

$$\varphi \,;\, (\psi_1 \vee \psi_2) \equiv (\varphi \,;\, \psi_1) \vee (\varphi \,;\, \psi_2) \tag{12}$$

$$(\varphi_1 \vee \varphi_2) \,;\, \psi \equiv (\varphi_1 \,;\, \psi) \vee (\varphi_2 \,;\, \psi) \tag{13}$$

Définition 4. *On introduit deux symboles de variations supplémentaires* $\bot_{\mathcal{D}}$ *et* $\top_{\mathcal{D}}$ *tels que, pour tout* $\alpha \in \mathcal{LP}$, $\alpha^{\bot_{\mathcal{D}}} = \bot \triangleright \bot = \bot_{\Delta}$ *et* $\alpha^{\top_{\mathcal{D}}} = \top \triangleright \top = \top_{\Delta}$.

À un cube φ *de* $(\Delta\mathcal{LP}, \models)$ *on associe* $\mathtt{var}(x, \varphi) \in \mathcal{D} \cup \{\top_{\mathcal{D}}\}$ *de la façon suivante :*

$$\mathtt{var}(x, \varphi) = \begin{cases} \top_{\mathcal{D}} & \text{si } x \notin \mathcal{V}(\varphi) \\ v & \text{si } x^v \text{ est un des termes de } \varphi \end{cases}$$

La proposition suivante permet de calculer la composition de deux atomes de $(\Delta\mathcal{LP}, \models)$ construits sur la même variable.

Proposition 6. *Il existe une opération binaire* $\,;\,$ *sur* $\mathcal{D} \cup \{\bot_{\mathcal{D}}, \top_{\mathcal{D}}\}$ *telle que pour une variable a, on a* $a^v \,;\, a^w \equiv a^{v;w}$.

Principe de la preuve. À chaque symbole de variation v on associe A^v, une matrice 2×2 de booléens telle que les lignes et les colonnes sont indexées par $(i, j) \in \{0, 1\}^2$ et tel que $A^v = [A^v_{ij}]_{ij}$ est défini comme suit (où a est une variable arbitraire et $\mathcal{IJ} \in \Delta\Omega$) :

$$A^v_{\mathcal{I}(a)\mathcal{J}(a)} = \begin{cases} 1 & \text{si } \mathcal{IJ} \models a^v \\ 0 & \text{sinon} \end{cases}$$

Ainsi, $A^{\bot_\mathcal{D}} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}$, $A^{+} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$ et $A^{\neq} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$.

On montre ensuite que, pour $v, w \in \mathcal{D} \cup \{\bot_\mathcal{D}, \top_\mathcal{D}\}$, le produit des matrices $A^v \times A^w$ (où la somme et le produit correspondent aux `ou` et au `et` sur les booléens) donne une matrice A^u où $u \in \mathcal{D} \cup \{\bot_\mathcal{D}, \top_\mathcal{D}\}$ (et on notera $u = v \ ; \ w$).

Il reste à montrer que le produit des matrices correspond bien à une composition au sens où, pour $a \in \mathcal{V}$, $a^v \ ; \ a^w = a^{v;w}$. Pour ce faire, un programme Python a été développé pour automatiser le calcul d'une table de composition. ■

Cela permet alors de calculer la composition de cubes :

Proposition 7. *Soit φ et ψ, deux cubes de $(\Delta\mathcal{LP}, \models)$. On a :*

$$\varphi \ ; \ \psi \equiv \bigwedge_{a \in \mathcal{V}} a^{\mathtt{var}(a,\varphi);\mathtt{var}(a,\psi)} \tag{14}$$

Pour composer deux cubes, on applique la proposition 7 et on simplifie en se rappelant que $x^{\top_\mathcal{D}} \equiv \top$ et $x^{\bot_\mathcal{D}} \equiv \bot$. Par exemple, si l'ensemble des variables est $\{a, b, c, d, e\}$ alors

$$\begin{aligned} & a^{+} \wedge b^{=} \wedge d^{=0} \ ; \ a^{-} \wedge c^{-} \wedge d^{+} \\ & \equiv a^{=0} \wedge b^{\top_\mathcal{D}} \wedge c^{\bullet 0} \wedge d^{+} \wedge e^{\top_\mathcal{D}} \\ & \equiv a^{=0} \wedge c^{\bullet 0} \wedge d^{+} \end{aligned}$$

(en s'appuyant sur une table de composition sur $\mathcal{D}$ disponible dans [8]).

De façon générale, une façon d'avoir une expression syntaxique pour $\varphi \ ; \ \psi$ consiste à mettre ces deux formules de $(\Delta\mathcal{LP}, \models)$ sous FND puis à appliquer les équivalences de la proposition 5 tant que c'est possible pour enfin appliquer la proposition 7 et se débarrasser complètement du symbole `;`.

4 Représenter des règles d'adaptation

Cette section vise de façon générale à formaliser dans $(\Delta\mathcal{LP}, \models)$ ce qu'est une règle d'adaptation et comment raisonner avec.

4.1 Introduction d'un exemple

On considère un exemple dans le domaine culinaire : un problème représente une requête de recette et une solution, une recette. On suppose que les problèmes et les solutions s'expriment en logique propositionnelle. Dans cet exemple, on suppose que l'ensemble des connaissances du domaine est vide : $\mathtt{CD} = \top$. Le problème d'adaptation est $(\mathtt{c}^s, \mathtt{x}^{\mathtt{cible}})$ où

— $\mathtt{c}^s$ est le cas source correspondant à une recette de salade, avec de la batavia, du magret de canard, des tomates, de l'huile d'olive et du vinaigre :

$$\begin{aligned} \mathtt{c}^s = \ & \mathit{rSalade} \wedge \mathit{iBatavia} \wedge \mathit{iMagret} \wedge \mathit{iTomate} \\ & \wedge \mathit{iHuileDOlive} \wedge \mathit{iVinaigre} \wedge \mathit{rien\ d'autre} \end{aligned}$$

où $\mathit{rSalade}$ se lit « recette de salade », $\mathit{iMachin}$, « recette avec l'ingrédient machin » et *rien d'autre* est une notation pour la conjonction des littéraux négatifs $\neg a$ tels que a ne peut être déduit du début de la formule : $\alpha \wedge \mathit{rien\ d'autre} \equiv \alpha \wedge \bigwedge\{\neg a \mid a \in \mathcal{V} \text{ et } \alpha \not\models_{\mathtt{CD}} a\}$. En particulier, $\mathtt{c}^s \models_{\mathtt{CD}} \neg \mathit{iAil}$. Notons que $\mathtt{c}^s$ est un cube complet.

— $\mathtt{x}^{\mathtt{cible}}$ est le problème cible « Je veux une recette de salade avec des tomates, du jus de citron, mais pas d'ail. » :

$$\mathtt{x}^{\mathtt{cible}} = \mathit{rSalade} \wedge \mathit{iTomate} \wedge \mathit{iJusCitron} \wedge \neg \mathit{iAil}$$

La connaissance du problème d'adaptation $(\mathtt{c}^s, \mathtt{x}^{\mathtt{cible}})$ permet de partitionner $\mathcal{V}$ en $\{\mathcal{V}_\mathcal{P}, \mathcal{V}_\mathcal{S}\}$: $\mathcal{V}_\mathcal{P} = \{\mathit{rSalade}, \mathit{iTomate}, \mathit{iJusCitron}, \mathit{iAil}\}$ (variables de $\mathtt{x}^{\mathtt{cible}}$) et $\mathcal{V}_\mathcal{S} = \mathcal{V} \setminus \mathcal{V}_\mathcal{P}$. Par conséquent, $\mathtt{c}^s$ s'écrit $\mathtt{c}^s = \mathtt{x}^s \wedge \mathtt{y}^s$ avec

$$\begin{aligned} \mathtt{x}^s &= \mathit{rSalade} \wedge \mathit{iTomate} \wedge \neg \mathit{iJusCitron} \wedge \neg \mathit{iAil} \\ \mathtt{y}^s &= \mathit{iBatavia} \wedge \mathit{iMagret} \wedge \mathit{iHuileDOlive} \\ & \quad \wedge \mathit{iVinaigre} \wedge \ldots \end{aligned}$$

(les points de suspension correspondant à la conjonction des littéraux $\neg a$ où $a \in \mathcal{V}_\mathcal{S}$ et a n'est pas un littéral positif de $\mathtt{y}^s$).

On considère également un motif M obtenu par un système d'extraction de motifs (p. ex., de motifs fermés fréquents) pour l'acquisition de connaissances d'adaptation. Un tel motif contient un ensemble fini d'atomes a^v indiquant des variations « simultanées » entre cas et s'interprète par la conjonction φ de ces éléments. φ peut s'écrire comme un cube de $(\Delta\mathcal{LP}, \models)$: si plusieurs atomes $a^{v_1}, a^{v_2}, \ldots$ construits sur la même variable a apparaissent dans φ, on peut montrer qu'ils peuvent être remplacés par un seul a^v. Dans cet exemple, on va considérer la formule suivante :

$$\varphi = \mathit{rSalade}^{=1} \wedge \mathit{iVinaigre}^{-} \wedge \mathit{iJusCitron}^{+} \wedge \mathit{iSel}^{+}$$

Le motif M dont est issu φ traduit le fait qu'on a observé un nombre suffisant de couples de recettes de salades avec du vinaigre dans la première mais pas dans la seconde, et du jus de citron et du sel dans la seconde mais pas dans la première. Par conséquent, si on considère que φ

est une règle d'adaptation, alors, on va considérer l'applicabilité et l'application de φ sur un problème d'adaptation $((\mathbf{x}^s, \mathbf{y}^s), \mathbf{x}^{\mathtt{cible}})$.

4.2 Représenter une règle d'adaptation

Une règle d'adaptation est définie comme étant un couple $R = (\varphi, \mathsf{C})$ où $\varphi \in \Delta\mathcal{LP}$ et C est un réel positif appelé *coût* de R.

Que signifie que R est *applicable* sur un problème d'adaptation $((\mathbf{x}^s, \mathbf{y}^s), \mathbf{x}^{\mathtt{cible}})$ et comment définir le résultat de cette application ? Pour répondre à cette question, on va s'appuyer sur une fonction qui à $(\alpha, \varphi, \gamma) \in \mathcal{LP} \times \Delta\mathcal{LP} \times \mathcal{LP}$ associe $\mathsf{appl}(\alpha, \varphi, \gamma) \in \mathcal{LP}$, l'application sur α de φ sous contrainte γ, fonction qui va être définie et étudiée dans la section 4.3. Comme les cas sources et le problème cible doivent être considérés avec les connaissances du domaine CD, on calculera la formule suivante :

$$\beta = \mathsf{appl}(\mathsf{CD} \wedge \mathbf{x}^s \wedge \mathbf{y}^s, \varphi, \mathsf{CD} \wedge \mathbf{x}^{\mathtt{cible}}) \qquad (15)$$

Si $\beta \not\models \bot$ alors on dira que R n'est pas applicable sur le problème d'adaptation. Sinon, ce résultat s'écrira $\beta \equiv \mathsf{CD} \wedge \mathbf{x}^{\mathtt{cible}} \wedge \mathbf{y}^{\mathtt{cible}}$ où $\mathbf{y}^{\mathtt{cible}}$ sera la solution proposée pour $\mathbf{x}^{\mathtt{cible}}$ par application de R sur le problème d'adaptation.

Le RàPC est généralement hypothétique et l'adaptation ne constitue pas nécessairement une solution correcte au problème cible. Ainsi, une règle d'adaptation $R = (\varphi, \mathsf{C})$ peut produire une telle solution incorrecte. Pour choisir entre plusieurs règles d'adaptation, on utilise C : le coût associé à cette règle est indicatif de la qualité de R. Plus C est élevé, moins la confiance en l'application de la règle sera grande ; on pourrait, par exemple, associer C a une mesure de la probabilité P que la règle donnera un résultat correct par $\mathsf{C} = -\log P$ (sous des hypothèses de distribution données).

4.3 Appliquer une formule des variations sur une formule propositionnelle

Cette section vise à définir une notion d'application d'une formule $\varphi \in \Delta\mathcal{LP}$ sur une formule $\alpha \in \mathcal{LP}$, en présence d'une contrainte γ sur le résultat.

On va noter $\mathsf{appl}(\alpha, \varphi, \gamma)$ l'application sur α de φ sous contrainte γ. Pour arriver à sa définition, on s'intéressera d'abord à la transformation de α par φ sous contrainte γ, une formule $\mathsf{transf}(\alpha, \varphi, \gamma) \in \Delta\mathcal{LP}$ dont le résultat (la partie droite) sera $\mathsf{appl}(\alpha, \varphi, \gamma) = \mathsf{D}(\mathsf{transf}(\alpha, \varphi, \gamma))$.

On propose les trois conditions suivantes sur $\mathcal{IJ} \in \Delta\Omega$ pour que $\mathcal{IJ} \models \mathsf{transf}(\alpha, \varphi, \gamma)$: $\mathcal{I} \models \alpha$, $\mathcal{IJ} \models \varphi$ et $\mathcal{J} \models \gamma$. Ces trois conditions peuvent s'écrire en une seule :

$$\mathcal{IJ} \models (\alpha \triangleright \gamma) \wedge \varphi \qquad (16)$$

Cette condition est-elle suffisante ? Considérons l'exemple suivant : $\alpha = a \wedge \neg b \wedge c$, $\varphi = a^- \wedge b^+$ et $\gamma = \top$. Si (16) était une condition nécessaire et suffisante pour que $\mathcal{IJ} \models \mathsf{transf}(\alpha, \varphi, \gamma)$ alors $\mathsf{appl}(\alpha, \varphi, \gamma)$ serait équivalente à $\mathsf{D}(\varphi)$ donc à $\neg a \wedge b$. Donc, que $\alpha \models c$ ou que $\alpha \models \neg c$ ne changerait pas l'application de φ sur α. Nous supposerons au contraire que si une variable (dans cet exemple c) est invariante dans φ, sa valeur doit être inchangée par la transformation, d'où l'utilisation du *ceteris paribus* (et la justification *a posteriori* de son étude) : on supposera donc que $\mathcal{IJ} \models \mathsf{cp}(\varphi)$. L'idée est que la formule φ représente de façon compacte une transformation et que quand elle ne dit rien sur une variable x, celle-ci est inchangée par la transformation : si $x \in \mathsf{Inv}(\varphi)$ et $\alpha \models x$ (resp. $\alpha \models \neg x$) alors, avec $\beta = \mathsf{appl}(\alpha, \varphi, \gamma)$, $\beta \models x$ (resp. $\beta \models \neg x$). On aboutit ainsi à la définition suivante :

Définition 5. *Pour $\varphi \in \Delta\mathcal{LP}$ et $\alpha, \gamma \in \mathcal{LP}$, on définit la transformation d'α par φ sous contrainte γ ainsi :*

$$\mathsf{transf}(\alpha, \varphi, \gamma) = (\alpha \triangleright \gamma) \wedge \mathsf{cp}(\varphi)$$

L'application de φ sur α sous contrainte γ est le résultat de cette transformation :

$$\mathsf{appl}(\alpha, \varphi, \gamma) = \mathsf{D}(\mathsf{transf}(\alpha, \varphi, \gamma))$$

On dira que φ est applicable sur α sous contrainte γ si $\mathsf{appl}(\alpha, \varphi, \gamma) \not\models \bot$. Deux conditions nécessaires (mais non suffisantes) d'applicabilité sont $\alpha \wedge \mathsf{G}(\varphi) \not\models \bot$ et $\mathsf{D}(\varphi) \wedge \gamma \not\models \bot$.

Malgré l'usage du *ceteris paribus*, si α, φ et γ contiennent peu de variables (relativement à $|\mathcal{V}|$) on peut calculer $\mathsf{appl}(\alpha, \varphi, \gamma)$ sans avoir à considérer toutes les variables, comme la proposition suivante le montre :

Proposition 8. *Soit $\alpha, \gamma \in \mathcal{LP}$ et $\varphi \in \Delta\mathcal{LP}$ telles que φ peut s'appliquer sur α sous contrainte γ. Soit $\mathcal{W} = \mathcal{V}(\alpha) \cup \mathcal{V}(\varphi) \cup \mathcal{V}(\gamma)$. On a :*

$$\mathsf{appl}(\alpha, \varphi, \gamma) \equiv \mathsf{D}((\alpha \triangleright \gamma) \wedge \mathsf{cp}_{\mathcal{W}}(\varphi))$$

Preuve. Soit $\beta_1 = \mathsf{appl}(\alpha, \varphi, \gamma) = \mathsf{D}((\alpha \triangleright \gamma) \wedge \mathsf{cp}(\varphi))$ et $\beta_2 = \mathsf{D}((\alpha \triangleright \gamma) \wedge \mathsf{cp}_{\mathcal{W}}(\varphi))$. On cherche à montrer que $\beta_1 \equiv \beta_2$.

Comme $\mathsf{cp}(\varphi) \models \mathsf{cp}_{\mathcal{W}}(\varphi)$, on en déduit que $\beta_1 \models \beta_2$ (en particulier parce que si $\psi \models \psi'$ alors $\mathsf{D}(\psi) \models \mathsf{D}(\psi')$ pour toute $\psi \in \Delta\mathcal{LP}$).

Montrons que $\beta_2 \models \beta_1$. Soit $\mathcal{J} \in \mathcal{M}(\beta_2)$: il reste donc à montrer que $\mathcal{J} \models \beta_1$. Par définition de β_2, il existe $\mathcal{I} \in \Omega$ telle que $\mathcal{IJ} \models (\alpha \triangleright \gamma) \wedge \mathsf{cp}_{\mathcal{W}}(\varphi)$. Soit alors $\mathcal{I}' \in \Omega$ définie (pour $x \in \mathcal{V}$) par $\mathcal{I}'(x) = \begin{cases} \mathcal{I}(x) & \text{si } x \in \mathcal{W} \\ \mathcal{J}(x) & \text{sinon} \end{cases}$. On va montrer que $\mathcal{I}'\mathcal{J} \models (\alpha \triangleright \gamma) \wedge \mathsf{cp}(\varphi)$ ce qui impliquera que $\mathcal{J} \models \beta_2$. Pour cela, il suffit de montrer les assertions suivantes :

(A1) $\mathcal{I}'\mathcal{J} \models \alpha \triangleright \gamma$;

(A2) $\mathcal{I}'\mathcal{J} \models \varphi$;

(A3) Pour toute $x \in \mathsf{Inv}(\varphi)$, $\mathcal{I}'\mathcal{J} \models x^{=}$.

Comme la seule différence entre $\mathcal{I}$ et $\mathcal{I}'$ concerne des variables qui ne sont pas dans $\mathcal{V}(\alpha)$ (puisque hors de $\mathcal{W}$) et que $\mathcal{I} \models \alpha$, on a donc $\mathcal{I}' \models \alpha$. Comme $\mathcal{J} \models \gamma$, on a donc (A1).

De la même façon, la seule différence entre $\mathcal{I}\mathcal{J}$ et $\mathcal{I}'\mathcal{J}$ concerne des variables qui ne sont pas dans $\mathcal{V}(\varphi)$ (puisque hors de $\mathcal{W}$) et que $\mathcal{I}\mathcal{J} \models \varphi$, on a donc $\mathcal{I}'\mathcal{J} \models \varphi$, d'où (A2).

Soit $x \in \mathsf{Inv}(\varphi)$. Pour montrer (A3), il suffit de montrer que $\mathcal{I}'(x) = \mathcal{J}(x)$. Si $x \notin \mathcal{W}$, par définition de $\mathcal{I}'$, c'est vérifié. Si $x \in \mathcal{W}$, on a donc $x \in \mathsf{Inv}(\varphi) \cap \mathcal{W}$, donc $\mathsf{cp}_{\mathcal{W}}(\varphi) \models x^{=}$ (d'après la définition 3). Comme $\mathcal{I}\mathcal{J} \models \mathsf{cp}_{\mathcal{W}}(\varphi)$ on a donc $\mathcal{I}\mathcal{J} \models x^{=}$, i.e. $\mathcal{I}(x) = \mathcal{J}(x)$. Or, $\mathcal{I}'(x) = \mathcal{I}(x)$ puisque $x \in \mathcal{W}$. On a donc $\mathcal{I}'(x) = \mathcal{J}(x)$, ce qui permet de prouver (A3) et qui conclut la preuve. ■

La proposition suivante permet de calculer en $O(|\mathcal{V}(\alpha)|+|\mathcal{V}(\varphi)|+|\mathcal{V}(\gamma)|)$ l'applicabilité et l'application de φ sur α dans le cas où α et β sont des cubes et φ est un cube primitif.

Proposition 9. *Soit α et γ deux cubes de $(\mathcal{LP}, \models)$ et φ, un cube primitif de $(\Delta\mathcal{LP}, \models)$. Soit $\mathcal{W} = \mathcal{V}(\alpha) \cup \mathcal{V}(\varphi) \cup \mathcal{V}(\gamma)$. On peut donc écrire $\alpha \equiv \bigwedge_{a\in\mathcal{W}} \ell(a)$, $\gamma \equiv \bigwedge_{a\in\mathcal{W}} p(a)$, et $\varphi \equiv \bigwedge_{a\in\mathcal{W}} m(a) \triangleright n(a)$ où les $\ell(a)$, $m(a)$, $n(a)$ et $p(a)$ appartiennent à $\{a, \neg a, \top\}$. On a donc $a \in \mathcal{V}(\alpha)$ ssi $\ell(a) \neq \top$, $a \in \mathcal{V}(\gamma)$ ssi $p(a) \neq \top$ et $a \in \mathcal{V}(\varphi)$ ssi $m(a) \neq \top$ ou (de façon équivalente) $n(a) \neq \top$ (puisque φ est un cube primitif).*

Pour $a \in \mathcal{W}$, on définit $f(a) \in \{a, \neg a, \bot\}$ de la façon suivante. Si $\ell(a) \wedge m(a) \models \bot$, $n(a) \wedge p(a) \models \bot$ ou ($\ell(a) \wedge p(a) \models \bot$ et $m(a) = n(a) = \top$) alors $f(a) = \bot$. Sinon,

$$f(a) = \begin{cases} \ell(a) & \text{si } \ell(a) \neq \top \text{ et } m(a) = n(a) = \top \\ p(a) & \text{si } \ell(a) = \top \text{ et } m(a) = n(a) = \top \\ n(a) & \text{sinon} \end{cases}.$$

On a alors : $\mathsf{appl}(\alpha, \varphi, \gamma) \equiv \bigwedge_{a\in\mathcal{W}} f(a)$

Par conséquent, φ est applicable sur α sous contrainte γ ssi pour toute $a \in \mathcal{W}$, $f(a) \neq \bot$.

La preuve de cette proposition (détaillée dans [8]) consiste essentiellement à détailler tous les cas possibles selon l'appartenance ou non de a à $\mathcal{V}(\alpha)$, à $\mathcal{V}(\varphi)$ et à $\mathcal{V}(\gamma)$ ($2^3 = 8$ cas à considérer, donc).

Enfin, la proposition suivante permet de calculer $\mathsf{appl}(\alpha, \varphi, \gamma)$ où φ est un cube primitif mais où α et γ sont quelconques, en s'appuyant sur des mises sous forme normale disjonctive de α et γ et sur la proposition 9.

Proposition 10. *Soit $\alpha, \alpha_1, \alpha_2, \beta, \beta_1, \beta_2 \in \mathcal{LP}$ et $\varphi \in \Delta\mathcal{LP}$. On a les équivalences suivantes :*

$$\mathsf{appl}(\alpha_1 \vee \alpha_2, \varphi, \gamma) \equiv \mathsf{appl}(\alpha_1, \varphi, \gamma) \vee \mathsf{appl}(\alpha_2, \varphi, \gamma)$$
$$\mathsf{appl}(\alpha, \varphi, \gamma_1 \vee \gamma_2) \equiv \mathsf{appl}(\alpha, \varphi, \gamma_1) \vee \mathsf{appl}(\alpha, \varphi, \gamma_2)$$

Preuve. On a :

$$\begin{aligned} &\mathsf{transf}(\alpha_1 \vee \alpha_2, \varphi, \gamma) = ((\alpha_1 \vee \alpha_2) \triangleright \gamma)) \wedge \mathsf{cp}(\varphi) \\ &\equiv ((\alpha_1 \triangleright \gamma) \wedge \mathsf{cp}(\varphi)) \vee ((\alpha_2 \triangleright \gamma) \wedge \mathsf{cp}(\varphi)) \\ &\equiv \mathsf{transf}(\alpha_1, \varphi, \gamma) \vee \mathsf{transf}(\alpha_2, \varphi, \gamma) \end{aligned}$$

D'après (2), on en déduit la première équivalence de la proposition. La deuxième équivalence se prouve de façon similaire. ■

4.4 Application d'une règle d'adaptation sur l'exemple

On considère le problème d'adaptation $((\mathtt{x}^s, \mathtt{y}^s), \mathtt{x}^{\mathtt{cible}})$ et la formule φ de la section 4.1, ainsi que la règle d'adaptation $R = (\varphi, \mathtt{C})$ où $\mathtt{C}$ est choisi arbitrairement. En appliquant (15) sur cet exemple, on obtient un $\beta \in \mathcal{LP}$ qui est satisfiable et tel que $\beta \equiv \mathtt{CD} \wedge \mathtt{x}^{\mathtt{cible}} \wedge \mathtt{y}^{\mathtt{cible}}$ avec

$$\mathtt{y}^{\mathtt{cible}} = \mathit{iBatavia} \wedge \mathit{iMagret} \wedge \mathit{iHuileDOlive} \wedge \mathit{iJusCitron} \wedge \mathit{iSel} \wedge \ldots$$

(qui est le résultat attendu sur cet exemple).

4.5 Un algorithme pour le RàPC

Une approche du RàPC utilisée dans le système Taaable [3], notamment, consiste à effectuer une remémoration s'appuyant sur une transformation minimale du problème cible afin qu'il s'apparie exactement avec au moins un cas source puis à utiliser une « transformation inverse » sur un cas remémoré pour obtenir une solution.

Dans le cadre de représentation de cet article, pour implanter cette approche, les transformations considérées seront celles associées aux règles d'adaptation $R = (\varphi, \mathtt{C})$: on considérera la transformation par la règle $R^{-1} = (\varphi^{-1}, \mathtt{C})$ (pour garantir un résultat, on pourra ajouter les règles $(a^{\neq}, \mathtt{C})$, pour $a \in \mathcal{V}$). En supposant que les coûts sont additifs, la remémoration retournera non seulement un (ou plusieurs) $(\mathtt{x}^s, \mathtt{y}^s)$ mais également une séquence de règles ${R_1}^{-1}$, ${R_2}^{-1}$, ..., ${R_q}^{-1}$ telle que, avec $\mathtt{x}^0 = \mathtt{CD} \wedge \mathtt{x}^{\mathtt{cible}}$ et, pour $i \in \{1, 2, \ldots, q\}$, $\mathtt{x}^i = \mathsf{appl}(\mathtt{x}^{i-1}, {\varphi_i}^{-1}, \top)$ (où $R_i = (\varphi_i, \mathtt{C}_i)$), avec $\mathtt{x}^s \wedge \mathtt{y}^s \models_{\mathtt{CD}} \mathtt{x}^q$. La transformation sera de coût $\sum_{i=1}^{q} \mathtt{C}_i$.

Une fois un cas $(\mathtt{x}^s, \mathtt{y}^s)$ remémoré avec une séquence de règles d'adaptation ${R_1}^{-1}$, ${R_2}^{-1}$, ..., ${R_q}^{-1}$ associée, l'idée est d'appliquer la séquence inverse de ces règles sur $\mathtt{x}^s \wedge \mathtt{y}^s$ pour résoudre $\mathtt{x}^{\mathtt{cible}}$: avec $\mathtt{y}^q$ tel que $\mathtt{x}^s \wedge \mathtt{y}^s \equiv \mathtt{x}^q \wedge \mathtt{y}^q$, on calcule successivement $\mathtt{y}^{q-1}$, ..., $\mathtt{y}^0$ avec, pour $i \in \{q-1, q-2, \ldots, 0\}$, $\mathtt{x}^i \wedge \mathtt{y}^i \equiv \mathsf{appl}(\mathtt{x}^{i+1} \wedge \mathtt{y}^{i+1}, \varphi_{i+1}, \mathtt{x}^i)$. Avec $\mathtt{y}^{\mathtt{cible}}$ telle que $\mathtt{CD} \wedge \mathtt{x}^{\mathtt{cible}} \wedge \mathtt{y}^{\mathtt{cible}} \equiv \mathtt{x}^0 \wedge \mathtt{y}^0$, on a alors une proposition de solution de $\mathtt{x}^{\mathtt{cible}}$, adaptée en q étapes à partir du cas remémoré.

Pour qu'une telle adaptation soit possible, il faut que les règles d'adaptation R_i soient applicables. Cela est garanti

par la proposition suivante, dans le cas où $\mathbf{x}^s \wedge \mathbf{y}^s$, $\mathbf{x}^{\mathtt{cible}}$ et φ_i sont des cubes :

Proposition 11. *Soit* α *et* γ*, deux cubes de* $(\Delta\mathcal{LP}, \models)$ *et* φ*, un cube de* $(\Delta\mathcal{LP}, \models)$*. On a :*

$$\mathsf{appl}(\alpha, \varphi, \gamma) \not\models \top \quad \textit{ssi} \quad \mathsf{appl}(\gamma, \varphi^{-1}, \top) \wedge \alpha \not\models \bot$$

La preuve de cette proposition (détaillée dans [8]) consiste à reprendre le découpage en 8 cas de la preuve de la proposition 9.

On montre que $\mathbf{x}^i \wedge \mathbf{y}^i$ est satisfiable pour tout $i \in \{0, 1, \ldots, q\}$ en partant de la fin :

- $(i = q)$ Comme, par définition, $\mathbf{x}^q \wedge \mathbf{y}^q$ est équivalente au cube complet $\mathbf{x}^s \wedge \mathbf{y}^s$ et que les cubes sont satisfiables, $\mathbf{x}^q \wedge \mathbf{y}^q \not\models \bot$.
- $(i = q-1)$ $\mathbf{x}^{q-1} \wedge \mathbf{y}^{q-1} \equiv \mathsf{appl}(\mathbf{x}^q \wedge \mathbf{y}^q, \varphi_q, \mathbf{x}^{q-1})$ qui, d'après la proposition 11, est satisfiable ssi β est satisfiable avec $\beta = \mathsf{appl}(\mathbf{x}^{q-1}, {\varphi_q}^{-1}, \top) \wedge \mathbf{x}^q \wedge \mathbf{y}^q$. Or, par définition de $\mathbf{x}^q$, $\beta \equiv \mathbf{x}^q \wedge \mathbf{x}^q \wedge \mathbf{y}^q \equiv \mathbf{x}^q \wedge \mathbf{y}^q$ qui est satisfiable (cf. cas $(i = q)$), donc β est satisfiable et par conséquent $\mathbf{x}^{q-1} \wedge \mathbf{y}^{q-1} \not\models \bot$.
- $(i = q-2)$ $\mathbf{x}^{q-2} \wedge \mathbf{y}^{q-2} \equiv \mathsf{appl}(\mathbf{x}^{q-1} \wedge \mathbf{y}^{q-1}, \varphi_{q-1}, \mathbf{x}^{q-2})$ qui, d'après la proposition 11, est satisfiable ssi β est satisfiable avec $\beta = \mathsf{appl}(\mathbf{x}^{q-2}, {\varphi_{q-1}}^{-1}, \top) \wedge \mathbf{x}^{q-1} \wedge \mathbf{y}^{q-1}$. Or, par définition de $\mathbf{x}^{q-1}$, $\beta \equiv \mathbf{x}^{q-1} \wedge \mathbf{x}^{q-1} \wedge \mathbf{y}^{q-1} \equiv \mathbf{x}^{q-1} \wedge \mathbf{y}^{q-1}$ qui est satisfiable (cf. cas $(i = q-1)$), donc β est satisfiable et par conséquent $\mathbf{x}^{q-2} \wedge \mathbf{y}^{q-2} \not\models \bot$.
- $(i = 0)$ $\mathbf{x}^0 \wedge \mathbf{y}^0 \equiv \mathsf{appl}(\mathbf{x}^1 \wedge \mathbf{y}^1, \varphi_1, \mathbf{x}^0)$ qui, d'après la proposition 11, est satisfiable ssi β est satisfiable avec $\beta = \mathsf{appl}(\mathbf{x}^0, {\varphi_1}^{-1}, \top) \wedge \mathbf{x}^1 \wedge \mathbf{y}^1$. Or, par définition de $\mathbf{x}^1$, $\beta \equiv \mathbf{x}^1 \wedge \mathbf{x}^1 \wedge \mathbf{y}^1 \equiv \mathbf{x}^1 \wedge \mathbf{y}^1$ qui est satisfiable (cf. cas $(i = 1)$), donc β est satisfiable et par conséquent $\mathbf{x}^0 \wedge \mathbf{y}^0 \not\models \bot$.

Comme $\mathtt{CD} \wedge \mathbf{x}^{\mathtt{cible}} \wedge \mathbf{y}^{\mathtt{cible}} \equiv \mathbf{x}^0 \wedge \mathbf{y}^0$, $\mathtt{CD} \wedge \mathbf{x}^{\mathtt{cible}} \wedge \mathbf{y}^{\mathtt{cible}}$ est satisfiable : l'adaptation donne bien un résultat cohérent.

Notons enfin que cet algorithme peut être utilisé pour résoudre un problème d'adaptation $((\mathbf{x}^s, \mathbf{y}^s), \mathbf{x}^{\mathtt{cible}})$ (quand un cas $(\mathbf{x}^s, \mathbf{y}^s)$ a déjà été choisi). Il suffit pour cela de l'appliquer à une base de cas singleton $\mathtt{BC} = \{(\mathbf{x}^s, \mathbf{y}^s)\}$.

4.6 Vers une composition des règles d'adaptation

L'algorithme de la section 4.5 s'appuie sur une séquence d'applications de règles d'adaptation. La question ouverte qui est posée ici est le calcul d'une composition de règles d'adaptation : étant donnés deux telles règles R_1 et R_2, peut-on définir une règle R telle que l'application de R équivaut à l'application successive de R_1 et de R_2 ? *A priori*, la composition des formules présentée à la section 3.5 devrait être utile ici, mais nous n'avons pas de réponse définitive à cette question pour l'heure.

Un intérêt de définir une telle composition de règles serait dans le post-traitement de l'apprentissage de ces règles. Si on utilise, par exemple, une extraction de motifs fermés fréquents, elle produira souvent un grand nombre de motifs M interprétables en règles R. S'il était possible de trouver une famille génératrice minimale pour la composition de ces règles, cela diminuerait leur nombre (et le temps de calcul) sans altérer l'inférence.

5 Discussion et liens avec d'autres travaux

Dans [7], $(\Delta\mathcal{LP}, \models)$ était comparée avec d'autres formalismes. Cette comparaison est complétée en section 5.1, sous l'angle des deux types d'inférences associées aux règles d'adaptation. La section 5.2 traite plus en détail du lien entre $(\Delta\mathcal{LP}, \models)$ et une logique dynamique.

5.1 Observations et actions

Les mécanismes liés à l'adaptation par transformation présentés dans la section 4 peuvent être partagés en mécanismes d'*observations* et mécanismes d'*actions*.

Les mécanismes d'observations apparaissent lors de l'apprentissage de connaissances d'adaptation (évoquée dans la section 2.2) dans laquelle sont observées des variations entre cas sources.

Étant donné une règle d'adaptation $R = (\varphi, \mathtt{C})$ et un problème d'adaptation $((\mathbf{x}^s, \mathbf{y}^s), \mathbf{x}^{\mathtt{cible}})$, l'application de R sur $((\mathbf{x}^s, \mathbf{y}^s), \mathbf{x}^{\mathtt{cible}})$ peut être apparentée à un mécanisme d'action : on crée une hypothèse de solution $\mathbf{y}^{\mathtt{cible}}$ de $\mathbf{x}^{\mathtt{cible}}$, partant de $(\mathbf{x}^s, \mathbf{y}^s)$. Donc, en assimilant un cas à un état et une règle d'adaptation à une action, on peut considérer l'application d'une règle d'adaptation comme l'application d'une action sur un état, pour obtenir un nouvel état (sachant que ce nouvel état est contraint par $\mathbf{x}^{\mathtt{cible}}$).

L'application de règles d'adaptation invite donc à se pencher sur les formalismes et inférences relatives aux actions (voir la synthèse [6]). En l'occurrence, les actions considérées sont déterministes : si une action est applicable sur un état, elle génère un état et un seul. Dans la section suivante, nous nous intéresserons à une logique dynamique et à ses liens avec $(\Delta\mathcal{LP}, \models)$.

Une dernière remarque peut être faite. On pourrait imaginer faire le lien entre les observations et les actions épistémiques (également évoquées dans [6]). Cependant, le lien avec les observations dont il est question ici (observer des variations entre deux cas) et les actions épistémiques (agir pour observer et acquérir/modifier des connaissances sur l'état actuel) ne s'est pas révélé fructueux (ou pas encore).

5.2 $(\Delta\mathcal{LP}, \models)$ et DL-PA

La logique DL-PA (*dynamic logic of propositional assignments*) est une logique dynamique dont les programmes atomiques sont des affectations de variables propositionnelles, i.e. des expressions $+a$ et $-a$ pour $a \in \mathcal{V}$ correspondant à une affectation de a par respectivement $\mathtt{1}$ et $\mathtt{0}$ [1].

Les formules de DL-PA sont des variables propositionnelles, des formules construites à partir d'autres formules en utilisant les connecteurs de la logique propositionnelle et les formules de la forme $\langle\pi\rangle\alpha$ où π est un programme et α est une formule. Les programmes sont soit des programmes atomiques, soit d'une des formes $\pi_1 \,;\, \pi_2$, $\pi_1 \cup \pi_2$, π^* et $\alpha?$ (où π, π_1 et π_2 sont des programmes et où α est une formule).

La sémantique d'une formule α est donnée par un sous-ensemble $\mathcal{M}(\alpha)$ de Ω et celle d'un programme π par un sous-ensemble $\mathcal{M}(\pi)$ de $\Omega \times \Omega = \Delta\Omega$, ce qui suggère d'étudier les liens entre les programmes de DL-PA et les formules de $(\Delta\mathcal{LP}, \models)$. Cette sémantique est définie de façon habituelle pour les variables propositionnelles et les formules construites sur les connecteurs, et pour le reste on a (pour $a \in \mathcal{V}$, α, une formule, π, π_1 et π_2, des programmes) :

$$\mathcal{M}(\langle\pi\rangle\alpha) = \left\{\mathcal{I} \in \Omega \;\middle|\; \begin{array}{l}\text{il existe } \mathcal{J} \in \Omega \text{ telle que}\\ \mathcal{I}\mathcal{J} \models \pi \text{ et } \mathcal{J} \models \alpha\end{array}\right\}$$

$$\mathcal{M}(+a) = \left\{\mathcal{I}\mathcal{J} \in \Delta\Omega \;\middle|\; \begin{array}{l}\mathcal{J} \models a\\ \text{et, pour toute } b \in \mathcal{V} \setminus \{a\},\\ \mathcal{I}(b) = \mathcal{J}(b)\end{array}\right\}$$

$$\mathcal{M}(-a) = \left\{\mathcal{I}\mathcal{J} \in \Delta\Omega \;\middle|\; \begin{array}{l}\mathcal{J} \models \neg a\\ \text{et, pour toute } b \in \mathcal{V} \setminus \{a\},\\ \mathcal{I}(b) = \mathcal{J}(b)\end{array}\right\}$$

$$\mathcal{M}(\pi_1 \,;\, \pi_2) = \left\{\mathcal{I}\mathcal{K} \in \Delta\Omega \;\middle|\; \begin{array}{l}\text{il existe } \mathcal{J} \in \Omega \text{ telle que}\\ \mathcal{I}\mathcal{J} \models \pi_1 \text{ et } \mathcal{J}\mathcal{K} \models \pi_2\end{array}\right\}$$

$$\mathcal{M}(\pi_1 \cup \pi_2) = \mathcal{M}(\pi_1) \cup \mathcal{M}(\pi_2)$$

$$\mathcal{M}(\alpha?) = \{\mathcal{I}\mathcal{I} \mid \mathcal{I} \in \mathcal{M}(\alpha)\}$$

Enfin, $\mathcal{I}\mathcal{J} \models \pi^*$ s'il existe une séquence $\mathcal{I}_0, \mathcal{I}_1, \ldots, \mathcal{I}_p$ ($p \geq 0$) telle que $\mathcal{I}_0 = \mathcal{I}$, $\mathcal{I}_p = \mathcal{J}$ et $\mathcal{I}_k\mathcal{I}_{k+1} \models \pi$ pour tout $k \geq 0$ et $k < p$.

Le *ceteris paribus* sur des formules de $(\Delta\mathcal{LP}, \models)$ va être utile pour faire le lien de formules de cette logique vers des programmes de DL-PA. On peut aussi définir une opération sur les programmes de DL-PA permettant de faire un lien dans l'autre sens. Cela suppose que l'ensemble des variables propositionnelles est fini et fixé, contrairement à l'hypothèse d'un ensemble dénombrable de variables qui est faite dans [1]. On fera donc cette restriction dans la suite de cette section.

On part de l'observation que le programme $+a \cup -a$ (pour $a \in \mathcal{V}$) fait l'inverse d'une affectation : même si a est affecté à une valeur ($\mathbb{0}$ ou $\mathbb{1}$) dans l'état de départ, l'exécution de ce programme fait que a n'a pas d'affection après l'exécution de ce programme. On généralise cela à un ensemble $\mathcal{W} = \{a_1, \ldots, a_p\}$ de variables :

$$\mathsf{désaff}(\{a_1, \ldots, a_p\}) = (+a_1 \cup -a_1) \,;\, \ldots \,;\, (+a_p \cup -a_p)$$

La table 1 donne des égalités entre ensembles de modèles de certaines formules de $(\Delta\mathcal{LP}, \models)$ et ensembles de modèles de certains programmes de DL-PA. On notera que dans cette table ne sont pas distingués les opérations sur les logiques des connecteurs de cette logique : par exemple, dans $\pi_1 \,;\, \pi_2$, le ; appartient à la syntaxe de la logique alors que dans $\varphi_1 \,;\, \varphi_2$, le ; est un opérateur. Par ailleurs, pour $\alpha \in \mathcal{LP}$ (qui est donc un cas particulier de formule de DL-PA), on a $\mathcal{M}(\alpha?) = \mathcal{M}(\widehat{\alpha})$ où $\widehat{\alpha}$ est défini dans [7] comme étant le plongement de α dans $(\Delta\mathcal{LP}, \models)$ défini syntaxiquement en remplaçant toutes les occurrences de $a \in \mathcal{V}(\alpha)$ par $a^{=\mathbb{1}}$. Enfin, pour $\alpha \in \mathcal{LP}$, $\varphi \in \Delta\mathcal{LP}$ et un programme π de DL-PA tel que $\mathcal{M}(\varphi) = \mathcal{M}(\pi)$, on a $\mathcal{M}(\langle\pi\rangle\alpha) = \mathsf{G}(\varphi \wedge \alpha^{\bullet\mathbb{1}})$.

Pour $a \in \mathcal{V}$, $\varphi_1, \varphi_2 \in \Delta\mathcal{LP}$ et π_1 et π_2 des programmes de DL-PA tels que $\mathcal{M}(\varphi_1) = \mathcal{M}(\pi_1)$ et $\mathcal{M}(\varphi_2) = \mathcal{M}(\pi_2)$, on a le tableau suivant, dans lequel, pour chaque ligne, $\mathcal{M}(\varphi) = \mathcal{M}(\pi)$:

$\varphi \in \Delta\mathcal{LP}$	π : programme de DL-PA
$\mathsf{cp}(a^{\bullet\mathbb{1}})$	$+a$
$\mathsf{cp}(a^{\bullet\mathbb{0}})$	$-a$
$a^{=\mathbb{1}}$	$a? \,;\, +a \,;\, \mathsf{désaff}(\mathcal{V} \setminus \{a\})$
$a^{=\mathbb{0}}$	$\neg a? \,;\, -a \,;\, \mathsf{désaff}(\mathcal{V} \setminus \{a\})$
a^{+}	$\neg a? \,;\, +a \,;\, \mathsf{désaff}(\mathcal{V} \setminus \{a\})$
a^{-}	$a? \,;\, -a \,;\, \mathsf{désaff}(\mathcal{V} \setminus \{a\})$
$\varphi_1 \vee \varphi_2$	$\pi_1 \cup \pi_2$
$\varphi_1 \,;\, \varphi_2$	$\pi_1 \,;\, \pi_2$

TABLE 1 – Des correspondances entre formules de $(\Delta\mathcal{LP}, \models)$ et programmes de DL-PA.

De façon générale, on peut traduire toute $\varphi \in \Delta\mathcal{LP}$ en un programme π de DL-PA de façon à ce que $\mathcal{M}(\varphi) = \mathcal{M}(\pi)$, comme nous le verrons ci-dessous, du moins sous l'hypothèse d'un ensemble fini de variables. Notons néanmoins que la traduction proposée n'est pas efficace et qu'il y a peu d'espoir d'en trouver une qui le soit puisque la satisfiabilité dans $(\Delta\mathcal{LP}, \models)$ est NP-complète alors que la complexité de la satisfiabilité dans DL-PA est EXPTIME-difficile (même si cette complexité concerne les formules et pas les programmes, c'est une indication de la complexité relative aux inférences sur les programmes).

Une preuve simple du fait que $\varphi \in \Delta\mathcal{LP}$ puisse se traduire en π tient au fait que tout sous-ensemble de $\Delta\Omega$ est représentable par un programme π (sous l'hypothèse d'un ensemble fini de variables). Une preuve plus constructive consiste à associer d'abord à chaque cube complet un programme de DL-PA. Soit ψ un cube complet de $(\Delta\mathcal{LP}, \models)$: $\psi = \bigvee_{a \in \mathcal{V}} a^{\omega(a)}$ pour une $\omega \in \Delta\Omega_{\text{alt}}$. Pour $a \in \mathcal{V}$, soit le programme

$$\pi_a = \begin{cases} a? \,;\, +a & \text{si } \omega(a) = \mathbb{1} \\ \neg a? \,;\, -a & \text{si } \omega(a) = \mathbb{0} \\ \neg a? \,;\, +a & \text{si } \omega(a) = \mathbb{+} \\ a? \,;\, -a & \text{si } \omega(a) = \mathbb{-} \end{cases}$$

Avec $\mathcal{V} = \{a_1, a_2, \ldots, a_n\}$ soit $\pi = \pi_{a_1} ; \pi_{a_2} ; \ldots ; \pi_{a_n}$. On peut montrer alors que $\mathcal{M}(\pi) = \mathcal{M}(\psi)$. De façon générale, toute φ peut s'écrire comme une disjonction de cubes complets : $\varphi = \psi_1 \vee \psi_2 \vee \ldots \vee \psi_p$. En associant à chaque cube ψ_k un programme π_k comme ci-dessus, on peut construire le programme $\pi = \pi_1 \cup \pi_2 \cup \ldots \cup \pi_p$. On a alors $\mathcal{M}(\pi) = \bigcup_{k=1}^{p} \mathcal{M}(\pi_k) = \bigcup_{k=1}^{p} \mathcal{M}(\psi_k) = \mathcal{M}(\varphi)$.

Il y a donc des liens forts entre $(\Delta\mathcal{LP}, \models)$ et DL-PA qui doivent être étudiés davantage. Il semblerait que $(\Delta\mathcal{LP}, \models)$ soit davantage utile pour *observer* des variations (et c'est pour cela que cette logique a été définie initialement) alors que DL-PA qui, en quelque sorte, intègre naturellement le *ceteris paribus* soit appropriée pour *agir* avec ces variations (en considérant l'application d'une règle d'adaptation comme l'exécution d'un programme). Ainsi, raisonner avec des règles d'adaptation pourrait se faire avec $(\Delta\mathcal{LP}, \models)$ pour leur apprentissage et avec DL-PA (ou un de ses fragments) pour leur application.

6 Conclusion

Cet article a abordé l'usage de la logique des variations propositionnelles au raisonnement à partir de cas, en se penchant sur une première problématique : celle de la représentation des règles d'adaptation.

De nombreuses perspectives suivent cette étude, en particulier celles mentionnées dans [7] et pas traitées dans cet article et celles qui apparaissent dans ce présent article. Nous considérerons en particulier l'étude de l'extension de la logique des variations propositionnelles à d'autres logiques, en particulier, des logiques avec des variables de types entier ou réel. Cela suppose de définir un langage de symboles de variations qui sera nécessairement incomplet : une logique étant supposée s'appuyer sur un langage dénombrable (du moins, c'est une hypothèse que nous considérons) et l'ensemble des relations binaires sur $\mathbb{N}$ étant indénombrable... Une façon de conduire cette étude passe par les logiques dynamiques au-delà de DL-PA, en tentant de résoudre l'équation analogique « Une telle logique des variations serait à $(\Delta\mathcal{LP}, \models)$ ce qu'une logique dynamique serait à DL-PA. »

Références

[1] Balbiani, Ph., A. Herzig et N. Troquard: *Dynamic logic of propositional assignments : a well-behaved variant of PDL*. Dans *28th Annual ACM/IEEE Symposium on Logic in Computer Science*, pages 143–152. IEEE, 2013.

[2] Carbonell, J. G.: *Learning by analogy : Formulating and generalizing plans from past experience*. Dans Michalski, R. S., J. G. Carbonell et T. M. Mitchell (rédacteurs) : *Machine Learning, An Artificial Intelligence Approach*, chapitre 5, pages 137–161. Morgan Kaufmann, Inc., 1983.

[3] Cordier, A., V. Dufour-Lussier, J. Lieber, E. Nauer, F. Badra, J. Cojan, E. Gaillard, L. Infante-Blanco, P. Molli, A. Napoli et H. Skaf-Molli: *Taaable : a Case-Based System for personalized Cooking*. Dans Montani, S. et L. C. Jain (rédacteurs) : *Successful Case-based Reasoning Applications-2*, tome 494 de *Studies in Computational Intelligence*, pages 121–162. Springer, 2014.

[4] d'Aquin, M., F. Badra, S. Lafrogne, J. Lieber, A. Napoli et L. Szathmary: *Case base mining for adaptation knowledge acquisition*. Dans Veloso, M. M. (rédacteur) : *Proc. of the 20th Int. Joint Conf. on Artificial Intelligence (IJCAI'07)*, pages 750–755. Morgan Kaufmann, Inc., 2007.

[5] d'Aquin, M., J. Lieber et A. Napoli: *Adaptation Knowledge Acquisition : a Case Study for Case-Based Decision Support in Oncology*. Computational Intelligence (an International Journal), 22(3/4) :161–176, 2006.

[6] Dupin de Saint-Cyr, F., A. Herzig, J. Lang et P. Marquis: *Raisonnement sur l'action et le changement*. Dans *Panorama de l'intelligence artificielle — ses bases méthodologiques, ses développements*, tome 1, chapitre 12, pages 363–392. Cépaduès, 2014.

[7] François, N., Th. Laure et J. Lieber: *Une logique pour représenter des variations propositionnelles*. Dans Bouraoui, Z., F. Schwarzentruber et A. Wilczynski (rédacteurs) : *Journées d'intelligence artificielle fondamentale – plateforme AFIA 2023*, page 11, Strasbourg, France, juillet 2023. Z. Bouraoui and F. Schwarzentruber and A. Wilczynski.

[8] François, N. et J. Lieber: *Appliquer la logique des variations propositionnelles à la représentation de règles d'adaptation pour le raisonnement à partir de cas (version étendue)*. rapport technique, Loria, 2024. (Ce rapport se trouve à l'adresse `https://hal.science/hal-04497632`).

[9] Hanney, K. et M. T. Keane: *Learning adaptation rules from a case-base*. Dans Smith, I. et B. Faltings (rédacteurs) : *Advances in Case-Based Reasoning – Proc. of the Third Eur. Workshop, EWCBR'96*, LNAI 1168, pages 179–192. Springer Verlag, Berlin, 1996.

[10] Jalali, V., D. Leake et N. Forouzandehmehr: *Learning and applying adaptation rules for categorical features : An ensemble approach*. AI Communications, 30(3-4) :193–205, 2017.

[11] Richter, M. M. et R. O. Weber: *Case-based reasoning, a textbook*. Springer, 2013.

[12] Riesbeck, C. K. et R. C. Schank: *Inside Case-Based Reasoning*. Lawrence Erlbaum Associates, Inc., Hillsdale, New Jersey, 1989. Available on line.

Session 2 : Analyse comportementale en planification

Actes JIAF 2024

Un cadre pour la planification consciente d'un observateur sous observabilité partielle

Salomé Lepers Vincent Thomas Olivier Buffet

Université de Lorraine, CNRS, Inria, LORIA, F-54000 Nancy, France
prénom.nom@loria.fr

Résumé

Dans cet article, nous nous intéressons à des problèmes de planification où l'agent est conscient de la présence d'un observateur et où cet observateur est en situation d'observabilité partielle. L'agent doit donc choisir sa stratégie dans le but d'optimiser les informations qu'il transmet à travers les observations. Nous proposons un cadre qui permet de traiter ce type de problème et de travailler avec différentes propriétés telles que la prédictibilité, la lisibilité et l'explicabilité. Notre travail s'appuie sur le cadre des processus de décision markoviens conscients d'un observateur (OAMDP). Étendre les OAMDP en observabilité partielle permet d'une part de travailler sur des problèmes plus réalistes (des situations où l'observateur n'aurait pas accès à l'ensemble des données de l'environnement), mais permet aussi de considérer des variables cibles dynamiques. Ces types dynamiques permettent de traiter la prédictibilité telle que présentée dans les pOAMDP (predictable OAMDP) ainsi que des problèmes de lisibilité à objectifs multiples où l'objectif de l'agent pourrait changer au cours du temps.

Abstract

In this article, we are interested in planning problems where the agent is aware of the presence of an observer, and where this observer is in a partial observability situation. The agent has to choose its strategy to optimize the information transmitted by observations. We build a framework to handle those kinds of problems and work with various properties such as predictability, legibility and explicability. Our work is based on the Observer Aware Markov Decision Process (OAMDP) framework. The extension of OAMDPs to partial observability can handle more realistic problems (situations where the observer doesn't have access to all of the environment information) but also allows to consider dynamic types. Those dynamic target variables allow to work with predictability as presented in the pOAMDP (predictable OAMDP) framework and with legibility problems with multiple goals where the goal might change during the task.

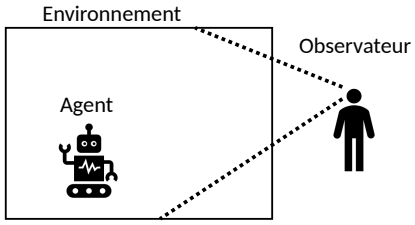


FIGURE 1 : Agent dans son environnement en présence d'un observateur passif

1 Introduction

Dans des situations de collaboration homme-robot, certaines propriétés du comportement du robot peuvent être appréciées de l'humain, voire permettre une meilleure collaboration. Divers travaux récents ont porté sur l'obtention automatique de comportements dotés de telles propriétés, en particulier dans le cas où l'humain ne fait qu'observer l'agent dans son environnement, et où l'agent, conscient de cet observateur, cherche à adopter un comportement qui permette de contrôler au mieux les informations acquises par l'humain (cf. figure 1).

CHAKRABORTI, KULKARNI, SREEDHARAN et al. [1] proposent une taxonomie des différents concepts rencontrés dans ces travaux, certains cherchant 1. à transmettre de l'information, tels que la *lisibilité* (lorsque l'agent essaye de communiquer son but à travers ses choix d'actions), l'*explicabilité* (un comportement explicable est conforme aux attentes de l'observateur), et la *predictabilité* (un comportement est prédictible si il est facile de deviner la fin d'une trajectoire en cours), ou 2. d'autres à cacher de l'information, par exemple l'*obscurcissement*, quand le comportement vise à cacher la tâche réelle de l'agent. Ils formalisent aussi ces différents problèmes de manière unifiée

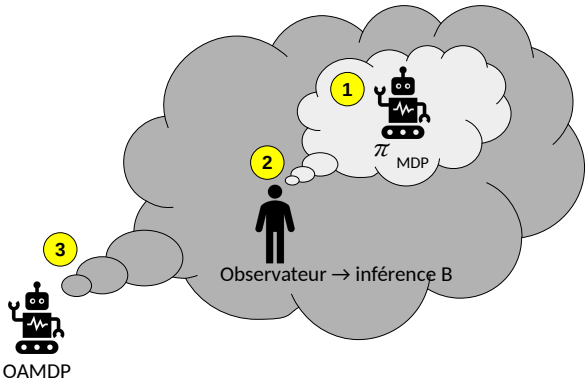


FIGURE 2 : Un agent OAMDP (3) fait l'hypothèse que l'observateur s'attend (2) à ce que l'agent se comporte de manière à accomplir une certaine tâche (1).

sous l'hypothèse que les transitions sont déterministes, raisonnant donc principalement sur des plans (une séquence d'actions induisant une unique séquence d'états). Dans leur approche, le robot modélise l'humain comme ayant un certain modèle du système robot+environnement (y compris de la ou les tâches possibles du robot), et pouvant ainsi anticiper les comportements possibles du robot. Chacune de ces propriétés peut être intéressante dans certaines situations et transmet différentes informations à l'observateur.

MIURA et ZILBERSTEIN [2], pour leur part, proposent un formalisme générique analogue (voir figure 2), mais sous l'hypothèse de transitions stochastiques, d'où le nom de *processus de décision markovien conscient d'un observateur* (OAMDP pour *observer-aware Markov decision process*). Ils font l'hypothèse que, du point de vue de l'observateur, l'agent effectue sa tâche en ignorant la présence de l'observateur. C'est une hypothèse réaliste dans un grand nombre de contextes. En outre, faire l'hypothèse contraire (l'observateur suppose que l'agent essaye d'aider son inférence) induirait un problème de poule et d'œuf, les deux cherchant à se modéliser l'un l'autre. Entre autres choses, ils travaillent aussi sur l'explicabilité, la lisibilité et la prédictibilité.

LEPERS, THOMAS et BUFFET [3] ont plus récemment proposé une nouvelle façon de modéliser la prédictibilité en s'inspirant du cadre OAMDP, et en proposant une approche plus adaptée aux environnements incertains. La variable cible n'étant plus un type statique, mais la prochaine action ou le prochain état de l'agent, donc une variable dynamique, il a fallu introduire un nouveau cadre, celui des pOAMDP (predictable OAMDP). L'objectif de cet article est de proposer un modèle qui permette de traiter à la fois des problèmes avec un type statique (lisibilité, explicabilité pour les OAMDP), des problèmes avec des variables cibles dynamiques (prédictibilité des pOAMDP) et des problèmes en observabilité partielle. Dans cette dernière situation, l'observateur n'a alors plus forcement accès à l'état et à l'action de l'agent mais à une observation liée à la transition suivie par le système. L'introduction d'observabilité partielle permet de travailler sur des problèmes plus divers et plus proches de la réalité tels que des situations où l'observateur n'aurait pas accès à l'ensemble des informations de l'environnement. Nous pouvons par exemple considérer des situations où l'agent PO-OAMDP n'est pas toujours visible et doit choisir d'utiliser certains passages pour être vu par l'observateur et lui permettre de mieux inférer la situation courante.

La section 2 introduit des pré-requis sur le processus de décision markoviens (MDP) et les MDP conscients d'un observateur. Notre approche général est décrite en section 3, la résolution des PO-OAMDP est décrite en section section 4 avant de conclure en section 5.

2 Pré-requis

2.1 Processus de décision markovien

Un *processus de décision markovien* (MDP) est un 6-uplet $\langle \mathcal{S}, \mathcal{A}, T, R, \gamma, \mathcal{S}_f \rangle$ où :

- $\mathcal{S}$ est l'ensemble des états ;
- $\mathcal{A}$ est l'ensemble des actions ;
- $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0; 1]$, la fonction de transition, donne la probabilité $T(s, a, s')$ d'aller dans un état s' depuis un état s en exécutant l'action a ;
- $R : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$, la fonction de récompense, donne la récompense reçue $R(s, a, s')$ lors d'une transition (s, a, s') ;
- $\gamma \in [0, 1]$ est le facteur d'actualisation ; et
- $\mathcal{S}_f \subset \mathcal{S}$ est l'ensemble des états terminaux : pour tout $s, a \in \mathcal{S}_f \times \mathcal{A}$, $T(s, a, s) = 1$ et $R(s, a, s) = 0$.

Une politique $\pi_{\text{OBS}} : \mathcal{S} \to \Delta(\mathcal{A})$ détermine un comportement en associant à chaque état une action à effectuer. Elle peut éventuellement être stochastique, $\pi_{\text{OBS}}(a|s)$ étant alors la probabilité d'effectuer a dans l'état s. Considérant un *MDP actualisé*, c'est-à-dire tel que $\gamma < 1$, la valeur d'une politique π_{OBS} en un état s est l'espérance de la somme des récompenses actualisées sur un horizon infini :

$$V^{\pi_{\text{OBS}}}(s) \stackrel{\text{def}}{=} \mathbb{E}_{\pi_{\text{OBS}}}\left[\sum_{t=0}^{\infty} \gamma^t r(S_t, A_t) | S_0 = s\right].$$

Il existe toujours au moins une politique ${\pi_{\text{OBS}}}^*$, dite optimale, telle que, pour tout s, $V^{{\pi_{\text{OBS}}}^*}(s) = \max_{\pi_{\text{OBS}}} V^{\pi_{\text{OBS}}}(s)$. L'algorithme d'*itération sur la valeur* (VI) calcule cette fonction de valeur optimale, notée V^*, en itérant le calcul suivant jusqu'à atteindre une précision suffisante (où k

désigne l'itération courante) :

$$V_{k+1}(s) \leftarrow \max_a \sum_{s'} T(s,a,s') \cdot (R(s,a,s') + \gamma V_k(s')) \,.$$

On interrompt les calculs quand le *résidu de Bellman* est inférieur à un seuil fonction de l'erreur ϵ souhaitée et de γ :

$$\underbrace{\max_s |V_{k+1}(s) - V_k(s)|}_{\text{résidu de Bellman}} \leq \frac{1-\gamma}{\gamma}\epsilon,$$

une politique déterministe ϵ-optimale étant alors obtenue en agissant de "manière gourmande" dans tout état s avec :

$$\pi_{\text{OBS}}{}^*(s) \leftarrow \arg\max_a \sum_{s'} T(s,a,s') \cdot (R(s,a,s') + \gamma V^*(s')) \,.$$

Les propriétés ci-dessus restent valident avec $\gamma = 1$ si

1. $\mathcal{S}_f$ non vide ; et
2. R est telle qu'il existe des politiques atteignant $\mathcal{S}_f$ avec probabilité 1 depuis tout état s, et que la valeur des autres politiques diverge vers $-\infty$ dans les états depuis lesquels on ne peut pas être sûr de pouvoir atteindre un état terminal.

On parle alors de problème de type *chemin stochastique le plus court* (SSP). On a un SSP en particulier, si, pour tout $(s,a,s') \in (\mathcal{S} \setminus \mathcal{S}_f) \times \mathcal{A} \times \mathcal{S}$, on a $r(s,a,s') < 0$, c'est-à-dire si on cherche à atteindre un état terminal à "moindre coût" (en moyenne).

Note : On peut transformer tout MDP actualisé en un SSP dans lequel, à chaque instant, on a une probabilité $1-\gamma$ de transiter vers un état terminal. Le cas SSP est donc plus général.

2.2 Processus de décision markovien conscient d'un observateur

Un *MDP conscient d'un observateur* (OA-MDP pour *observer-aware MDP*) décrit une situation dans laquelle un agent interagit avec son environnement en ayant conscience de la présence d'un observateur, et en cherchant à maximiser un critère de performance lié aux croyances de cet observateur. Il est défini formellement par un 8-uplet $\langle \mathcal{S}, \mathcal{A}, T, \gamma, \mathcal{S}_f, \Theta, B, R \rangle$ où :

- $\langle \mathcal{S}, \mathcal{A}, T, \gamma, \mathcal{S}_f \rangle$ est un MDP sans fonction de récompense ;
- Θ est un ensemble fini de *types* possibles de l'agent, représentant une caractéristique de celui-ci telle que sa tâche réelle ou ses capacités ;
- $B : H^* \to \Delta^{|\Theta|}$ donne la croyance que l'observateur a sur le type de l'agent (la *croyance* sur une variable aléatoire est la distribution de probabilité sur ses valeurs possibles étant données les informations disponibles) en fonction de l'historique des états et des actions ($H \stackrel{\text{def}}{=} \mathcal{S} \times \mathcal{A}$) ;
- $R : \mathcal{S} \times \mathcal{A} \times \Delta^{|\Theta|} \to \mathbb{R}$ est la fonction de récompense.

Dans la plupart des cas considérés par Miura et Zilberstein, B est obtenue en s'appuyant sur la définition de la mise-à-jour de croyance bayésienne BST de Baker, Saxe et Tenenbaum, c'est-à-dire en considérant que, du point de vue de l'agent, l'observateur modélise le comportement de l'agent pour une tâche donnée à travers un MDP :

1. en utilisant une fonction de récompense R_{OBS} appropriée ;
2. en résolvant le MDP $\langle \mathcal{S}, \mathcal{A}, T, \gamma, \mathcal{S}_f \rangle$ (où tous les composants, exceptée la fonction de récompense R_{OBS}, émanent de la définition de l'OAMDP) pour obtenir V^*_{OBS} ; et
3. en construisant une politique "softmax", c'est-à-dire telle que, pour chaque couple (s,a),

$$\pi_{\text{OBS}}(a|s) = \frac{e^{\frac{1}{\tau}Q^*_{\text{OBS}}(s,a)}}{\sum_{a'} e^{\frac{1}{\tau}Q^*_{\text{OBS}}(s,a')}}, \text{ où}$$
$$Q^*_{\text{OBS}}(s,a) = \sum_{s'} T(s,a,s') \cdot \left(r(s,a,s') + \gamma V^*_{\text{OBS}}(s')\right),$$

$\tau > 0$ représentant le niveau de rationalité de l'agent (considéré par l'observateur) afin de pouvoir raisonner sur des politiques plus ou moins proches de la politique optimale.

La croyance de l'observateur sur les types peut ensuite être obtenue par inférence bayésienne en utilisant π_{OBS}.

Miura et Zilberstein formalisent ainsi, entre autres, des problèmes de lisibilité, d'explicabilité, et de prédictibilité.

Note : Comme déjà fait ci-dessus, on indicera souvent par "OBS" les quantités liées au point de vue de l'observateur (tel que perçu par l'agent). Entre autres, certaines probabilités seront calculées du point de vue de l'observateur, et notées P_{OBS}. Aussi, on écrira parfois une fonction $f(X,Y)$ décrivant une distribution de probabilité conditionnelle sous la forme $f(Y|X)$ pour faire ressortir les dépendances entre variables.

3 Contribution : MDP conscient d'un observateur en observabilité partielle

Comme vu en introduction, on souhaite proposer un modèle OAMDP en observabilité partielle, lequel permettrait à la fois de traiter plus de scénarios (situations où l'observateur ne voit pas toujours le robot) et traiter des propriétés qui nécessitent l'utilisation de variables cibles dynamiques.

3.1 Formalisme

Dans le cadre PO-OAMDP, l'agent a accès à l'état complet du système, et l'observateur n'en a désormais qu'une perception partielle. Sa construction part de l'ajout au formalisme OAMDP d'un ensemble d'observations et d'une fonction d'observation. Nous allons expliquer cette construction avant de donner une définition formelle. Dans ce contexte, le type est désormais nommé *variable cible* et peut évoluer au cours du temps, contrairement au type statique des OAMDP. Afin de rester souple et générique dans la définition de ce qu'est la variable cible, sa valeur à chaque pas de temps est le résultat d'une fonction prenant en entrée la transition suivie par le système. La variable cible peut donc être juste une sous-partie de l'état du système (par exemple une variable non observable par l'observateur), mais elle peut aussi être liée à l'action émise par l'agent (pour des problèmes de prédictibilité) ou à l'évolution de l'état plus qu'à l'état lui même. Évidemment, cette variable cible peut regrouper en son sein plusieurs variables différentes. Nous ne parlons que d'une unique variable que par commodité mais sans perte de généralité.

En outre, on suppose que, en plus de l'état complet du système, l'agent a accès aux observations reçues par l'observateur (ce qui reste réaliste dans de nombreuses situations, en particulier si le processus d'observation est déterministe, auquel cas les observations reçues par l'observateur sont facilement prédictibles). L'agent peut ainsi construire l'état interne de l'observateur au fur et à mesure de l'exécution de son comportement.

En ayant accès à toutes les informations du problème (l'état du système, les actions effectuées et les observations perçues par l'observateur), l'agent va planifier ses actions pour chercher à contrôler l'inférence faite par l'observateur sur sa variable cible.

Formellement, un PO-OAMDP est ainsi défini par un n-uplet $\langle \mathcal{S}, \mathcal{A}, T, \gamma, \mathcal{S}_f, R_{\text{OBS}}, \Theta, \Omega, O, B, R_{\text{AG}}, \phi \rangle$, où :

- $\langle \mathcal{S}, \mathcal{A}, T, \gamma, \mathcal{S}_f, R_{\text{OBS}} \rangle$ est un MDP ;
- Θ désigne une *variable cible* (dynamique), mais aussi l'ensemble fini de valeurs qu'elle peut prendre ; nous changeons de terminologie pour souligner la différence entre cette variable (dynamique) et la variable *type* des OAMDP ;
- $\phi : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \Theta$ est une fonction qui donne la valeur de la variable cible en fonction de la transition : $\theta_t = \phi(s_t, a_t, s_{t+1})$;
- Ω est un ensemble fini d'observations ;
- $O : \mathcal{A} \times \mathcal{S} \to \Omega$ est la fonction d'observation ; $O(a, s', o)$ est la probabilité d'émission d'une observation o si l'action a conduit dans l'état s' ;
- $B : \Omega^* \to \Delta^{|\mathcal{S}|}$ donne la croyance que l'observateur a sur l'état en fonction de l'historique des observations ; la croyance sur la variable cible pourra en être déduite (voir section 3.2) ; on notera $b_t \stackrel{\text{def}}{=} B(o_1, \ldots, o_{t-1})$, en appelant croyance initiale $b_0 \stackrel{\text{def}}{=} B()$;
- $R_{\text{AG}} : \mathcal{S} \times \Delta^{|\Theta|} \times \mathcal{A} \times \mathcal{S} \times \Delta^{|\Theta|} \to \mathbb{R}$ est la fonction de récompense de l'agent sous sa forme la plus générale : $R_{\text{AG}}(s_t, \beta_t, a_t, s_{t+1}, \beta_{t+1})$.

Alors que dans un POMDP, la mise à jour de la croyance dépend nécessairement des actions effectuées choisies par l'agent lui même (incluses dans l'historique), dans un PO-OAMDP, les actions ne sont pas forcément connues de l'observateur. Les observations peuvent inclure les actions selon le scénario considéré.

Nous ferons ici l'hypothèse que l'observation reçue permet à l'observateur de savoir à chaque instant si un état terminal a été atteint ou non, sans nécessairement indiquer duquel il s'agit (ce qui pourra dépendre du problème).

On notera que le modèle PO-OAMDP repose sur un seul MDP sous-jacent, contrairement au modèle OAMDP, qui associe un MDP à chaque type possible. Dans notre cadre à observabilité partielle, n'employer qu'un seul MDP sous-jacent n'est toutefois pas restrictif. On pourrait montrer formellement que tout OAMDP peut se ré-écrire comme un PO-OAMDP.

On notera aussi que l'introduction de la variable cible n'a rien de nécessaire. On pourrait obtenir un formalisme équivalent en écrivant une fonction de récompense directement sur la croyance sur les états au lieu de la croyance sur la variable cible. L'introduction de la variable cible est une commodité pour afficher le lien avec les OAMDP et pour faciliter la modélisation des problèmes.

La suite de cette section décrit comment la croyance de l'observateur (sur l'état) est mise à jour, et comment en déduire la croyance sur la variable cible, laquelle permet de calculer la récompense de l'agent lors d'une transition. Elle illustre ensuite les usages possibles du cadre PO-OAMDP sur différents scénarios.

3.2 Calcul des croyances sur l'état et la variable cible

Comme dans le cadre OAMDP, l'observateur calcule la politique de l'agent étant donnée la fonction de récompense qu'il connaît, R_{OBS}. L'observateur modélise le comportement de l'agent pour une tâche donnée à travers un MDP :

1. en utilisant une fonction de récompense R_{OBS} ;
2. en résolvant le MDP sous-jacent ; et
3. en construisant une politique softmax.

On peut noter que, étant données la dynamique (transitions+observations) du PO-OAMDP et la politique π_{OBS} de l'agent, l'observateur fait face à un HMM : il résout un

problème de *filtrage* en devant estimer la croyance sur l'état s_t en fonction de l'historique d'observation $o_{1:t}$.

La croyance de l'agent peut ensuite être construite à partir de la politique de l'agent, de la fonction de transition du MDP sous-jacent et de la fonction d'observation :

$$\begin{aligned} B(s_{t+1}|o_{1:t+1}) = P(s_{t+1}|o_{1:t+1}) &= \frac{P(s_{t+1}, o_{1:t}, o_{t+1})}{P(o_{1:t+1})} \\ &= \frac{P(s_{t+1}, o_{1:t+1})}{\sum_{s_{t+1}} P(s_{t+1}, o_{1:t+1})} \\ &= \frac{K(s_{t+1}, o_{1:t+1})}{\sum_{s_{t+1}} K(s_{t+1}, o_{1:t+1})}, \text{ avec} \\ K(s_{t+1}, o_{1:t+1}) &\stackrel{\text{def}}{=} \sum_{a_t} O(o_{t+1}|a_t, s_{t+1}) \sum_{s_t} T(s_{t+1}|s_t, a_t) \cdot \\ &\quad \pi_{\text{OBS}}(a_t|s_t) \cdot B(s_t|o_{1:t}). \end{aligned}$$

Croyance sur la variable cible Pour déterminer la récompense reçue lors d'une transition, il est aussi nécessaire de calculer la croyance β sur la valeur que va prendre la variable cible : $\Theta_t = \phi(S_t, A_t, S_{t+1})$. Cela peut être réalisé en partant de la croyance b de l'observateur sur l'état courant, S_t, comme suit :

$$\begin{aligned} \beta(\theta) &= \sum_{s,a,s'} \mathbb{1}_{\theta=\phi(s,a,s')} \cdot P_{obs}(s, a, s'|b) \\ &= \sum_{s,a,s'} \mathbb{1}_{\theta=\phi(s,a,s')} \cdot P_{obs}(s'|s, a) \cdot P_{obs}(a|s) \cdot P_{obs}(s|b) \\ &= \sum_{s,a,s'} \mathbb{1}_{\theta=\phi(s,a,s')} \cdot T(s, a, s') \cdot \pi_{\text{OBS}}(a|s) \cdot b(s). \end{aligned} \quad (1)$$

3.3 Mise en œuvre sur divers scénarios

Le modèle PO-OAMDP permet de construire différents comportements en faisant varier Θ et R et donc d'aborder différents types de problèmes.

Nous allons commencer par montrer comment des OAMDP peuvent être reformulés comme des PO-OAMDP. Dans un OAMDP, l'agent a un *type* statique qui le caractérise. Dans le cadre PO-OAMDP, cela peut se traduire par une variable d'état (cachée) dont la valeur est extraite par $\theta = \phi(s)$.

3.3.1 Expression de B et R' pour différentes propriétés

Lisibilité La lisibilité réduit l'ambiguïté sur les buts possibles de l'agent. Dans cette situation, l'agent a plusieurs buts possibles inconnus de l'observateur. Un comportement lisible transmet le but (ou, plus généralement, le critère de performance) de l'agent à travers ses choix d'actions.

Θ **:** Dans le cas de la propriété de lisibilité, le type va donc caractériser ce critère de performance parmi un ensemble fini de critères possibles. Cela va se traduire typiquement par le fait que la fonction de récompense dépend de ce type, mais pas nécessairement la fonction de transition ou la fonction d'observation.

R_{AG} **:** Pour la fonction de récompense, Miura et Zilberstein utilisent l'opposé de la distance euclidienne à la "croyance idéale". La croyance idéale étant définie par : $\beta^*(s) = (0, \ldots, 0, 1, 0, \ldots, 0)$ (avec un 1 en composante $\theta = \phi(s)$), on a alors :

$$R_{\text{AG}}(s, \beta, a, s', \beta') \stackrel{\text{def}}{=} -\sqrt{\|\beta - \beta^*(s)\|_2}.$$

Explicabilité Un comportement explicable est un comportement cohérent avec les attentes de l'observateur.

Θ **:** Pour traduire cette idée, Miura et Zilberstein (suivant Sreedharan, Kulkarni, Chakraborti et al. [5]) proposent de minimiser la probabilité que le comportement observé soit celui d'un comportement aléatoire, même si plusieurs autres comportements restent probables. Ils introduisent ainsi un type "virtuel" θ_0 qui représente un comportement (une politique) aléatoire en plus des autres types (réels).

R_{AG} **:** Pour traduire le critère d'explicabilité susmentionné, on va prendre

$$R_{\text{AG}}(s, \beta, a, s'\beta') \stackrel{\text{def}}{=} -\beta(\theta_0).$$

Prédictibilité Un comportement prédictible est un comportement dont la fin de trajectoire est plus facile à prédire pour l'observateur. On propose ici une définition de la prédictibilité inspirée des travaux de Lepers, Thomas et Buffet [3], mais mieux fondée d'un point de vue théorique. On essaye donc de prédire soit l'action, soit l'état de l'agent,

Si nous partons comme Miura et Zilberstein et d'autres de cette définition, nous nous inspirons plutôt des travaux de Lepers, Thomas et Buffet [3], lesquels sont plus adaptés aux problèmes à dynamique stochastique, mais en faisant ici une proposition dont la sémantique est plus claire.

Θ **:** L'idée de départ est que l'observateur cherche, à chaque instant, à prédire la prochaine action ou le prochain état, d'où deux types de prédictibilité différents. Pour la prédictibilité sur l'action, on pose $\Theta = A$ et $\phi(s, a, s') = a$. Pour la prédictibilité sur l'état, on pose $\Theta = S$ et $\phi(s, a, s') = s'$. Dans les deux cas, pour agir optimalement, l'observateur doit parier sur une des prochaines valeurs cibles les plus probables, et donc choisir une valeur dans l'ensemble

$$\psi_\Theta(\beta_t) \stackrel{\text{def}}{=} \arg\max_\theta \beta_t(\theta).$$

R_{AG} **:** On considère que l'observateur échantillonne sa prédiction de façon uniforme, on peut alors définir :

$$\text{pred}(\theta|\beta_t) \stackrel{\text{def}}{=} \begin{cases} \frac{1}{|\psi_\Theta(\beta_t)|} & \text{si } \theta \in \psi_\Theta(\beta_t), et \\ 0 & \text{sinon.} \end{cases}$$

En définissant

$$R_{\text{AG}}(s, \beta, a, s', \beta') \stackrel{\text{def}}{=} \begin{cases} \text{pred}(a|\beta) - 1 & \text{si } \Theta = A, \\ \text{pred}(s'|\beta) - 1 & \text{si } \Theta = S, \end{cases}$$

la récompense immédiate est l'opposé de la probabilité que le pari (d'un observateur rationnel) échoue : $R_{\text{AG}}(s, \beta, a, s', \beta') = -P(\text{pari perdu})$.

Obscurcissement La problématique inverse peut également être considérée. L'agent essaye alors de cacher des informations telles que son but à l'observateur. Dans cette situation, l'agent a plusieurs buts possibles et essaye de ne pas révéler son "vrai" but à l'observateur. L'obscurcissement avec le modèle PO-OAMDP présente les mêmes difficultés que celles rencontrées par le modèle OAMDP :

- si l'objectif ne porte que sur l'obscurcissement, mais pas sur la réalisation de la tâche, l'agent peut simplement ne rien faire pour dissimuler son but, et
- pour construire la croyance de l'observateur sur les buts, on fait l'hypothèse que l'observateur ne sait pas qu'on essaye de le tromper.

3.3.2 Élargissement des types de problèmes couverts

Les sections précédentes ont montré comment étendre les problèmes déjà modélisés dans les cadres OAMDP et p-OAMDP en considérant l'observabilité partielle de l'observateur. Il faut cependant noter que les PO-OAMDP permettent la formalisation de nouveaux problèmes dans lesquels l'agent ne va pas simplement chercher à exhiber un comportement prédictible, lisible ou explicable, mais pourra chercher à transmettre au mieux de l'information sur l'état du monde partiellement observé par l'observateur.

Scénario 1 : Si on considère un environnement de type bureau (cf. figure 3) avec des portes soit ouvertes, soit fermées à clef, on peut imaginer un agent cherchant à faire comprendre à un observateur extérieur l'état des portes à travers ses actions. Cela peut bien entendu se faire en ouvrant des portes visibles pour l'observateur mais aussi en se montrant dans certaines zones qui ne peuvent être atteintes que par l'ouverture de certaines portes. Ainsi, même si ces portes ne sont jamais vues par l'observateur, la présence de l'agent peut lui permettre d'inférer que certaines portes sont ouvertes. Dans l'exemple représenté en figure 3, en

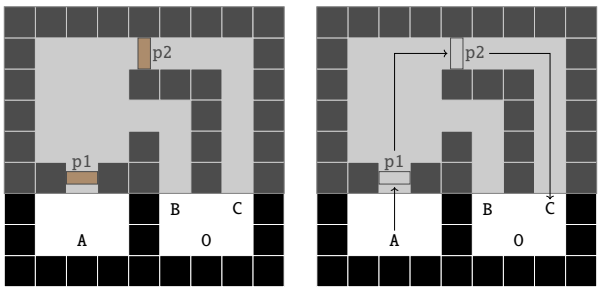


FIGURE 3 : Représentation d'un environnement avec des portes pouvant être verrouillées. Les murs sont représentés pas des cases noires, les portes par des rectangles bruns. La zone grisée correspond à une zone non visible de l'observateur. L'agent débute en `A` et doit se diriger en `0`.

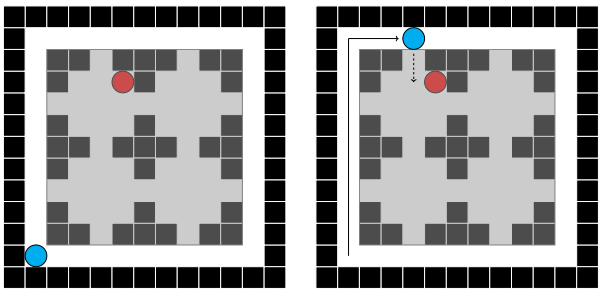


FIGURE 4 : Environnement à grille avec un intrus dont l'observateur cherche à connaître la localisation. Les murs sont représentés par des cases noires. La zone grisée correspond à une zone non visible de l'observateur. L'intrus est représenté par un cercle rouge et l'agent par un cercle bleu.

choisissant un chemin plus long dans la zone cachée et en redevenant visible en `C`, il informe l'observateur que les portes `p1` et `p2` ne sont pas verrouillées. Se rendre visible en `B` permettrait d'atteindre l'objectif et augmenterait un peu la probabilité que la porte `p2` soit fermée. Résoudre ce problème nécessite que l'agent raisonne sur 1. les conséquences de ses actions, 2. sa visibilité en fonction de sa localisation, 3. les inférences que pourra faire l'observateur et 4. les portes dont l'observateur cherche à estimer l'état.

Scénario 2 : Dans une seconde situation, on peut considérer un agent en charge de détecter des intrus dans un environnement inaccessible pour l'observateur (cf. figure 4). En utilisant un modèle que l'observateur a de son comportement, cet agent peut tirer parti des attentes de l'observateur pour agir, se rendre visible à certains endroits et faire comprendre à l'observateur la présence effective d'un intrus et sa localisation. Dans l'exemple illustré par la figure 4, l'observateur estime que l'agent cherche à se rapprocher

de l'intrus. L'agent peut ainsi informer l'observateur de la position de l'intrus en choisissant, parmi les trajectoires possibles l'amenant au plus proche de l'intrus, un chemin souvent visible de l'observateur.

Scénario 3 : Enfin, dans des tâches complexes qui requièrent plusieurs étapes intermédiaires, l'agent peut chercher à transmettre l'état de la tâche en cours en empruntant des chemins plus longs mais 1. qui sont partiellement visibles par l'observateur et 2. qui laissent moins d'ambiguïté sur son objectif intermédiaire.

En cherchant à faciliter l'inférence des objectifs intermédiaires qu'il cherche à atteindre, l'agent peut ainsi transmettre l'état de la tâche à réaliser, ce qui peut s'avérer crucial dans une situation collaborative (qui attendrait en retour une action particulière de l'humain).

Ces différentes situations montrent que le cadre des OAMDP permet d'élargir les problèmes à d'autres types d'échanges d'information que simplement des informations sur le comportement de l'agent lui-même. Le cadre permet de modéliser des problèmes proches des problèmes de recherche active d'information tels que formalisés par les ρ-POMDP [6]. Dans le cadre ρ-POMDP, un agent est dans un environnement partiellement observé et doit agir au mieux pour acquérir des observations pertinentes et maximiser une mesure d'information sur des variables cibles (par exemple sa localisation). La principale différence avec le cadre PO-OAMDP est que, dans ce dernier, c'est l'information acquise par un tiers (l'observateur) que l'agent cherche à contrôler, pas la sienne propre (qui est complète). Cela nécessite en particulier un modèle de l'observateur dont l'agent va tirer parti pour chercher à contrôler indirectement les croyances de l'observateur.

4 Résolution

Nous allons maintenant discuter des approches possibles pour la résolution de PO-OAMDP, c'est-à-dire de politiques maximisant (au moins à $\epsilon > 0$ près) la somme des récompenses atténuées. On fait l'hypothèse dans cette section qu'on dispose d'un couple état-croyance initial, les algorithmes discutés en faisant tous usage.

Dans la suite, on considère d'abord les solutions prenant la forme naturelle de politiques dépendant de l'historique, avant de voir que l'on peut aussi raisonner avec des croyances à la place de ces historiques.

4.1 Recherche d'une politique historique-dépendante

4.1.1 Données pertinentes pour la politique

Au premier abord, à un instant donné, le choix d'action de l'agent dépend au plus des données spécifiques à sa trajectoire actuelle, c'est-à-dire l'historique des états, actions et observations.

On a toutefois besoin pour prendre des décisions que d'informations nécessaires à la prédiction des récompenses. Or, par définition, la récompense reçue à un instant t dépend de la transition $s_t, b_t, a_t, s_{t+1}, b_{t+1}$ (la croyance sur la variable cible se déduisant de la croyance sur l'état comme on l'a déjà vu). Comme, dans ce tuple, 1. l'état s_t évolue de manière markovienne (indépendamment de l'historique d'états et d'actions antérieurs), et 2. la croyance sur l'état b_t ne dépend que des observations passées, alors le choix d'action ne dépend que de l'état courant s_t et de l'historique d'observations $o_1, \dots, o_{t-1}$. On va donc pouvoir ne considérer que les politiques de la forme $\pi_{\text{AG}} : \mathcal{S} \times (\Omega)^* \to \mathcal{A}$ (en notant que, dans cet espace "d'états", parmi les politiques optimales, certaines sont déterministes).

4.1.2 Résolution

Dans le cas $\gamma < 1$, étant donné $\epsilon > 0$, on peut trouver une solution ϵ-optimale en se ramenant à un problème à horizon fini et en employant un algorithme tel que la programmation dynamique, AO* [7] ou MCTS [8] (comme l'ont fait Miura et Zilberstein [2] pour les OAMDP). L'opérateur d'optimalité de Bellman va alors s'écrire, pour $t < H - 1$,

$$V_t^*(s, o_{1:t}) = \max_a \sum_{s', o_{t+1}} T(s'|a, s) \cdot O(o_{t+1}|s', a) \cdot \\ [R'(s, b_{[o_{1:t}]}, a, s', b_{[o_{1:t+1}]}) + \gamma V_{t+1}^*(s', b_{[o_{1:t+1}]})],$$

et

$$V_H^*(s, o_{1:H}) = 0.$$

Dans le cas $\gamma = 1$, sauf cas particulier, on ne peut pas se ramener à un horizon temporel fini. Les politiques historique-dépendantes ne paraissent donc pas adaptées.

4.2 Recherche d'une politique croyance-dépendante

Nous allons voir ici que la résolution d'un PO-OAMDP est équivalente à celle d'un MDP dans un espace continu particulier. Cela va permettre d'envisager l'emploi de politiques croyances-dépendantes.

4.2.1 Belief-MDP équivalent

Statistique suffisante Nous avons vu précédemment comment calculer l'état de croyance de l'observateur sur l'état, au vu de la séquence d'observations qu'il a reçues, par filtrage bayésien. En fait, l'état d'information (s, b) (l'état courant couplé avec la croyance de l'observateur sur l'état) constitue une statistique suffisante pour la planification puisqu'elle

1. est markovienne (on peut prédire son évolution sans avoir recours à des informations antérieures) puisque l'état s est markovien par définition, et la croyance b l'est aussi d'après nos calculs (elle peut être mise à jour en ne connaissant que la dernière observation reçue) ; et

2. permet d'estimer la récompense à chaque pas de temps, par définition de la fonction de récompense dans un PO-OAMDP.

Formalisation du belief-MDP On obtient donc un MDP valide $\langle \mathcal{I}, \mathcal{A}, T', R', \gamma, \mathcal{I}_f \rangle$, où :

- $\mathcal{I} \stackrel{\text{def}}{=} \mathcal{S} \times B$ est l'ensemble des états ;
- $\mathcal{A}$ est l'ensemble des actions, identique à l'ensemble des actions du PO-OAMDP ;
- $T' : \mathcal{I} \times \mathcal{A} \times \mathcal{I} \to [0; 1]$ est une nouvelle fonction de transition (voir plus bas) ;
- $R' : \mathcal{I} \times \mathcal{A} \times \mathcal{I} \to \mathbb{R}$ est une nouvelle fonction de récompense (voir plus bas) ;
- $\gamma \in [0, 1]$ est le facteur d'actualisation ; et
- $\mathcal{I}_f \subset \mathcal{I}$ est l'ensemble des éléments $\iota \equiv (s, b)$ de $\mathcal{I}$ tels que $s \in \mathcal{S}_f$; on pourra vérifier ultérieurement que, pour tout $\iota, a \in \mathcal{I} \times \mathcal{A}, T(\iota, a, \iota) = 1$ et $R'(\iota, a, \iota) = 0$.

Fonction de transition T' : La fonction de transition du belief-MDP est définie par :

$$\begin{aligned} T'(\iota'|a, \iota) &\stackrel{\text{def}}{=} T(s', b'|a, s, b) \\ &= \sum_o \mathbb{1}_{b'=B(b,o)} O(o|s', a) P(s'|a, s). \end{aligned}$$

Fonction de récompense R' : La fonction de récompense du belief-MDP est définie par :

$$R'(s, b, a, s', b') \stackrel{\text{def}}{=} R_{\text{AG}}(s, \beta(b), a, s', \beta(b')),$$

où $\beta(b)$ dénote la croyance β que l'on peut dériver de b comme vu dans l'équation (1).

4.2.2 Résolution

Ce nouveau MDP est défini sur un espace d'états continu. Sauf cas particuliers, l'ensemble des états accessibles depuis l'état initial est donc infini.

Dans le cas $\gamma < 1$, on pourrait à nouveau se ramener à un problème à horizon fini, avec l'avantage que deux trajectoires différentes peuvent conduire à la même paire (ι, t), ce qui permettrait de réduire la quantité de calculs. L'opérateur d'optimalité de Bellman s'écrit alors (en développant $\iota = (s, b)$)

$$\begin{aligned} V_t^*(s, b) &= \max_a \sum_{s'} \int_{b'} T(s', b'|a, s, b) \cdot \\ &\qquad [R'(s, b, a, s', b') + \gamma V_{t+1}^*(s', b')] db' \\ &= \max_a \sum_{s', o} O(o|s', a) \cdot T(s'|a, s) \cdot \\ &\qquad [R'(s, b, a, s', b^o) + \gamma V_{t+1}^*(s', b^o)], \end{aligned}$$

où b^o est la croyance sur l'état mise à jour après observation de o, et

$$V_H^*(s, b) = 0.$$

On peut aussi se demander si, comme dans le cadre des POMDP résolus via des bMDP, la fonction de valeur optimale a des propriétés de continuité particulières (en l'occurrence de convexité) qui permettraient d'approcher celle-ci. Des résultats préliminaires montrent toutefois qu'il peut y avoir des discontinuités sur les bords de la fonction de valeur d'une politique, autour de points dont l'état de croyance est impossible. On ne pourra donc pas re-transcrire directement les approches POMDP reposant sur des approximateurs généralisants de V^* (tels que HSVI [9], PBVI [10] et SARSOP [11]). Mais des versions spécifiques tenant compte des localisations des discontinuités sont peut-être envisageables.

Dans le cas $\gamma = 1$, un premier problème est de savoir si le problème obtenu est un SSP valide. Il faudrait ainsi vérifier, par exemple, si c'est toujours le cas avec les fonctions de récompenses proposées pour les propriétés de lisibilité, explicabilité et prédictibilité.

À supposer que le SSP obtenu soit valide, on se retrouve sur des problèmes proches des POSSP (ou Goal-POMDP) pour lesquels peu de travaux ont été développés à part, par exemple, ceux de Patek [12] ou, plus récemment, de Horák, Bošanský et Chatterjee [13].

Approche bi-critère Un problème déjà présent dans la résolution des OAMDP est que, si l'on emploie un critère lié à une fonction de récompense "observer-aware", il est possible que la performance liée à la fonction de récompense classique du MDP sous-jacent soit fortement dégradée. Une première approche peut être de combiner linéairement deux tels critères, mais cela soulève la question de la bonne pondération de ceux-ci. Une autre approche, aussi évoquée par Miura et Zilberstein [2], est d'optimiser un critère observer-aware sous la contrainte que le

critère “classique” doit atteindre au minimum une certaine valeur [14]-[18]. Il peut alors être nécessaire que la politique optimale soit stochastique.

5 Conclusion

Nous avons introduit un nouveau formalisme, celui des OAMDP en observabilité partielle (PO-OAMDP), lequel permet de travailler sur des problèmes de lisibilité, d'explicabilité et de prédictibilité en observabilité partielle. La complexité des PO-OAMDP est au moins celle des OAMDP telle qu'étudiée par Miura et Zilberstein [2]. Selon eux, il n'est pas nécessairement bénéfique de se ramener un POMDP pour le résoudre. Différents ensembles de variables cibles et fonctions de récompense ont été proposés pour construire des comportements avec des propriétés différentes. Ce nouveau modèle permet aussi de traiter des problèmes plus proches de la réalité où un agent agit en prenant en compte un observateur qui peut ne pas avoir accès à l'ensemble des informations de l'environnement (porte fermée, champ de vision bloqué). Nous avons également discuté de la résolution des PO-OAMDP.

Comme expliqué dans la partie résolution, nous proposons de transformer les PO-OAMDP en belief-MDP équivalents pour les résoudre avec des approches génériques comme MCTS. Une première perspective est d'étudier les performances de ces algorithmes pour résoudre les PO-OAMDP. Une seconde direction de travail est d'étudier les propriétés théoriques du modèle PO-OAMDP pour proposer des algorithmiques tirant parti des propriétés de ce cadre. Une étape sera de proposer des problèmes pertinents pour tester notre modèle et évaluer les algorithmes proposés.

Références

[1] T. Chakraborti, A. Kulkarni, S. Sreedharan, D. E. Smith et S. Kambhampati, “Explicability ? Legibility ? Predictability ? Transparency ? Privacy ? Security ? The Emerging Landscape of Interpretable Agent Behavior”, in *Proceedings of the Twenty-Ninth International Conference on Automated Planning and Scheduling (ICAPS)*, 2019. adresse : https://ojs.aaai.org/index.php/ICAPS/article/view/3463.

[2] S. Miura et S. Zilberstein, “A unifying framework for observer-aware planning and its complexity”, in *Proceedings of the Thirty-Seventh Conference on Uncertainty in Artificial Intelligence*, t. 161, juill. 2021, p. 610-620. adresse : https://proceedings.mlr.press/v161/miura21a.html.

[3] S. Lepers, V. Thomas et O. Buffet, “Comment rendre des comportements plus prédictibles”, in *JIAF-JFPDA - Journées d'Intelligence Artificielle Fondamentale*, juill. 2023. adresse : https://hal.science/hal-04212452.

[4] C. L. Baker, R. Saxe et J. B. Tenenbaum, “Action understanding as inverse planning”, *Cognition*, t. 113, n° 3, p. 329-349, déc. 2009. doi : 10.1016/j.cognition.2009.07.005.

[5] S. Sreedharan, A. Kulkarni, T. Chakraborti, D. E. Smith et S. Kambhampati, *A Bayesian Account of Measures of Interpretability in Human-AI Interaction*, 2020. arXiv : 2011.10920 [cs.AI].

[6] M. Araya-López, O. Buffet, V. Thomas et F. Charpillet, “A POMDP Extension with Belief-dependent Rewards”, in *Advances in Neural Information Processing Systems 23*, Vancouver, Canada, 2010.

[7] N. Nilsson, *Principles of Artificial Intelligence*. Morgan Kaufmann Publishers, 1980.

[8] L. Kocsis et C. Szepesvari, “Bandit based Monte-Carlo Planning”, in *Proceedings of the Sixteenth European Conference on Machine Learning*, 2006.

[9] T. Smith et R. G. Simmons, “Point-Based POMDP Algorithms : Improved Analysis and Implementation”, in *Proceedings of the Twenty-First Conference on Uncertainty in Artificial Intelligence*, 2005, p. 542-549.

[10] J. Pineau, G. Gordon et S. Thrun, “Anytime point-based approximations for large POMDPs”, *Journal of Artificial Intelligence Research*, t. 27, p. 335-380, 2006.

[11] H. Kurniawati, D. Hsu et W. S. Lee, “SARSOP : Efficient point-based POMDP planning by approximating optimally reachable belief spaces”, in *Robotics : Science and Systems IV*, 2008.

[12] S. Patek, “On partially observed stochastic shortest path problems”, in *Proceedings of the 40th IEEE Conference on Decision and Control*, t. 5, 2001, p. 5050-5055. doi : 10.1109/CDC.2001.981011.

[13] K. Horák, B. Bošanský et K. Chatterjee, “Goal-HSVI : Heuristic Search Value Iteration for Goal-POMDPs”, in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, 2018, p. 4764-4770.

[14] E. Altman, *Constrained Markov Decision Processes*. Chapman et Hall/CRC, 1999.

[15] D. Kim, J. Lee, K.-E. Kim et P. Poupart, “Point-based value iteration for constrained POMDPs”, in *IJCAI*, t. 11, 2011, p. 1968-1974.

[16] F. Dufour et T. Prieto-Rumeau, "Stochastic approximations of constrained discounted Markov decision processes", *Journal of Mathematical Analysis and Applications*, t. 413, n° 2, p. 856-879, 2014. doi : 10.1016/j.jmaa.2013.12.016.

[17] F. Trevizan, S. Thiébaux, P. Santana et B. Williams, "Heuristic Search in Dual Space for Constrained Stochastic Shortest Path Problems", *Proceedings of the International Conference on Automated Planning and Scheduling*, t. 26, n° 1, mars 2016. doi : 10.1609/icaps.v26i1.13768.

[18] J. Lee, G.-H. Kim, P. Poupart et K.-E. Kim, "Monte-Carlo tree search for constrained POMDPs", *Advances in Neural Information Processing Systems*, t. 31, 2018.

Actes JIAF 2024

Learning Interpretable Behaviour Classifiers for PDDL Planning

Arnaud Lequen

IRIT, Université Toulouse III - Paul Sabatier
arnaud.lequen@irit.fr

Résumé

On cherche à synthétiser des modèles interprétables reconnaissant le comportement d'un agent parmi d'autres, et ce sur tout un domaine de planification, exprimé en PDDL. Nous proposons d'apprendre des formules logiques, à partir d'un ensemble d'exemples de taille réduite, qui montrent la solution apportée par l'agent à un ensemble de petites instances. Ces formules sont exprimées dans une variante de la Logique Temporelle du Premier Ordre (FTL) adaptée au formalisme de la planification automatique. De telles formules sont lisibles par un humain, et peuvent être vues comme des explications (partielles) de la politique mise en œuvre par un agent. Notre méthode consiste à apprendre de tels classifieurs de comportements au travers d'une compilation vers MaxSAT topologiquement guidée qui nous permet d'apprendre une grande variété de formules. Une étude expérimentale montre que notre implémentation peut apprendre des formules intéressantes en temps raisonnable.

Abstract

We consider the problem of synthesizing interpretable models that recognize the behaviour of an agent out of many others, on a whole set of planning problems expressed in PDDL. Our approach consists in learning logical formulas, from a set of small examples that show how an agent solved small planning instances. These formulas are expressed in a version of First-Order Temporal Logic (FTL) tailored to our planning formalism. Such formulas are human-readable, and serve as (partial) explanations of an agent's policy. We propose to learn such behaviour classifiers through a topology-guided compilation to MaxSAT, which allows us to generate a wide range of different formulas. Experiments show that interesting formulas can be learned in reasonable time.

1 Introduction

One of the main strengths of PDDL planning models is that they are succinct and human-readable, but can nonetheless express general, complex problems, whose state search spaces are exponential in the size of the encoding – as can be the solutions. As a consequence, given a set of examples of the behaviour of an agent (called traces), understanding and recognizing this behaviour can be tedious.

In order to summarize the behaviour of a planning agent in a concise, interpretable way, we propose to learn properties that are specific to the solutions proposed by this agent. Such properties, expressed in a temporal logic tailored to fit PDDL planning models, are not only human-readable, but are also general, and can be evaluated against different instances of the same planning problem. This allows them to recognize the behaviour of an agent on instances that are substantially different from the ones used in the set of examples.

More specifically, the problem we tackle is the one where, given a set of positive example traces (the ones of the agent we seek to recognize) and negative examples traces (the ones of other agents), we wish to learn a model that can discriminate as well as possible between positive and negative traces. A wide variety of techniques and models of different natures have been proposed in the literature. Among these, the learning of finite-state automata (DFA) is a well-studied problem [1, 20, 21], but DFAs can grow quickly (thus becoming harder to interpret) and do not generalize to instances not in the example set. More recently, neural network-based architectures such as LSTMs [22] have shown very promising results, but lack interpretability, and the rationale for their decision is rarely clear.

In the past decade, significant efforts have been made towards learning logical formulas expressed in (a form of) temporal logic. Such works [18, 19, 8, 15, 3, 4] often leverage symbolic methods to learn Linear Temporal Logic (LTL) formulas [17] that fit the example traces, and thus share some similarities with our work. Some other authors propose other techniques, such as Latent Dirichlet Allocation [12], which stems from the field of natural language processing.

However, in all of these cases, the knowledge extracted from the sets of examples has the major drawback of not generalizing well to unknown instances. This is due to the

choice of the language used to express these properties. For instance, since LTL formulas are built over a set of propositional variables, they do not generalize to models that do not share the same variables.

To address this issue, we propose to learn properties in a version of First-Order Temporal Logic (FTL). When tailored to the PDDL planning formalism, FTL can express a wide range of properties that generalize from one planning instance to the other, given that they model similar problems. This was shown in [2], who proposed to express *search control knowledge* in a language similar to ours, albeit with the aim of guiding the search of a planner designed to use such knowledge. In [4], the authors proposed to synthesize such control knowledge automatically, and thus address the problem of learning properties expressed in a fragment of FTL.

In this paper, we show that it is possible to learn richer and more expressive properties, using the whole range of FTL operators and modalities. The properties we wish to learn should describe the behaviour of a given planning agent, without being true for the behaviour of other agents. We show that learning such formulas is computationally intractable, as the associated decision problem is NP-hard. This is why the core of our approach consists in encoding the learning problem into a MaxSAT instance, which has the added benefit of showing resilience to any potential noise in the set of training examples. To make the search more efficient, we fix the general topology of the target formula before the encoding. In addition to alleviating the load on the MaxSAT solver and rendering the algorithm more parallelizable, this also increases the diversity in the formulas learned by our algorithm, thus providing varied descriptions of the behaviour of the agent of interest.

Our article is organised as follows: Section 2 introduces the planning formalism as well as the FTL language. Section 3 formally introduces the learning problem we tackle in this paper, and shows that the associated decision problem is intractable. Sections 4 and 5 present some technical choices that we made to solve our problem in reasonable time in practice. In Section 6, we describe our reduction of the problem to MaxSAT, and in Section 7, we present our experimental results, as well as a few examples of formulas that are within reach of our implementation.

2 Background

2.1 Planning with PDDL

This section introduces the model that we use to describe planning tasks. Our definition of a PDDL planning task differs from [9], for instance, as we require the organization of the objects of our instances into types. The model we use resembles the one defined in [11]

Definition 1 (Type tree) *A* type tree $\mathcal{T}$ *is a non-empty tree where each node is labeled by a symbol, called a* type*. For any type $\tau \in \mathcal{T}$, we call* strict subtype *any descendant τ' of τ. τ' is a subtype of τ (denoted $\tau' \preceq \tau$) when τ' is a strict subtype of τ or when $\tau' = \tau$.*

Definition 2 (Object class) *Let $\mathcal{O}$ be a set of elements called* objects*. We call* object class *any subset of $\mathcal{O}$. A class c_i is said to be a* subclass *of type c_j if $c_i \subseteq c_j$.*

Definition 3 (Type hierarchy) *A* type hierarchy $\mathcal{H}$ *over type tree $\mathcal{T}$ is a set of object classes such that $\mathcal{O} \in \mathcal{H}$, and such that each object class of $\mathcal{H}$ is mapped to a unique type of $\mathcal{T}$. This mapping $\tau : \mathcal{H} \to \mathcal{T}$ is such that for any pair c_i, c_j of object classes:*

- *c_i is a subclass of c_j iff $\tau(c_i)$ is a subtype of $\tau(c_j)$ (and conversely);*
- *$c_i \cap c_j = \emptyset$ iff $\tau(c_i)$ is not a subtype of $\tau(c_j)$ (and conversely).*

We say that object $o \in \mathcal{O}$ is of type $\tau(o) := \tau(c)$ where c is the smallest (for inclusion $\subseteq$) class of $\mathcal{H}$ to which o belongs.

Definition 4 (Predicate, atoms and fluents) *A* predicate *p is a symbol, which is associated with:*

- *An arity $ar(p) \in \mathbb{N}$*
- *A type for each of its arguments. For $i \in \{1, \dots, ar(p)\}$, the type of its argument at position i is denoted $\tau_p(i) \in \mathcal{T}$*

An atom *is a predicate for which each argument is associated with a symbol, which can be a variable symbol, or an object of $\mathcal{O}$. When the i-th argument of the atom is an object o (associated to type hierarchy $\mathcal{H}$), then we require that $\tau(o) = \tau_p(i)$. The atom consisting of predicate p and symbols $x_1, \dots, x_{ar(p)}$ is denoted $p(x_1, \dots, x_{ar(p)})$.*

A fluent *is an atom where each argument is associated an object of $\mathcal{O}$. A* state *is a set of fluents.*

Definition 5 (Action schema and operators) *An* action schema *is a tuple $a = \langle \mathsf{pre}(a), \mathsf{add}(a), \mathsf{del}(a) \rangle$, such that $\mathsf{pre}(a)$, $\mathsf{add}(a)$ and $\mathsf{del}(a)$ are sets of* atoms *instantiated with variables only.*

An operator *o is akin to an action schema, except that the sets $\mathsf{pre}(o)$, $\mathsf{add}(o)$ and $\mathsf{del}(o)$ are sets of fluents.*

This leads us to the main definition of this section, which crystallizes the elements above.

Definition 6 (PDDL planning problem) *A PDDL planning problem is a pair $\Pi = \langle \mathcal{D}, \mathcal{I} \rangle$ where $\mathcal{D} = \langle \mathcal{P}, \mathcal{A}, \mathcal{T} \rangle$ is the domain and $\mathcal{I} = \langle \mathcal{O}, \mathcal{H}, I, G \rangle$ is the instance.*

The domain $\mathcal{D}$ consists of a set $\mathcal{P}$ of predicates, a set of actions schemas $\mathcal{A}$, and a type hierarchy $\mathcal{T}$.

The instance $\mathcal{I}$ consists of a set of objects $\mathcal{O}$ and an associated type hierarchy $\mathcal{H}$, as well as two states, I and G, which are respectively the initial state and goal.

An operator o is applicable in a state s if $\mathsf{pre}(\mathsf{o}) \subseteq s$. The state that results of the application of o on s is $s[\mathsf{o}] = (s \setminus \mathsf{del}(\mathsf{o})) \cup \mathsf{add}(\mathsf{o})$.

A sequence of operators $\mathsf{o}_1, \ldots, \mathsf{o}_n$ is called a *plan* for Π if there exists a sequence of states $s_0, \ldots, s_n$ where $s_0 = I$, and which is such that, for all $i \in \{1, \ldots, n\}$, $s_i = s_{i-1}[\mathsf{o}_i]$ and o_i is applicable in s_{i-1}. Such a sequence of states (which is unique for each plan) is called a *trace*. A plan is called a *solution-plan* if, in addition to this, $G \subseteq s_n$.

We will say that a fluent $p(o_1, \ldots, o_{ar(p)})$ is true in state s iff $p(o_1, \ldots, o_{ar(p)}) \in s$.

2.2 First-Order Temporal Logic

This section introduces the language in which are expressed the formulas that we attempt to learn throughout this paper.

Syntax Let $\mathcal{X}$ be a set of variable symbols, $\mathcal{P}$ a set of predicates, and $\mathcal{T}$ a type tree. We define our language $\mathcal{L}_{\text{FTL}}$ such that:

$$\psi := \exists x \in \tau.\psi \mid \forall x \in \tau.\psi \mid \varphi$$

where $\varphi \in \mathcal{L}_{\text{TL}}$, and $\mathcal{L}_{\text{TL}}$ is such that:

$$\varphi := \top \mid p(x, \ldots, x) \mid \neg\varphi \mid \bigcirc\varphi \mid \Diamond\varphi \mid \Box\varphi \mid \overline{\bigcirc}\varphi \mid \overline{\Diamond}\varphi \mid \overline{\Box}\varphi \mid \varphi \,\mathsf{U}\, \varphi \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \varphi \Rightarrow \varphi$$

where x is a variable of $\mathcal{X}$, p a predicate of $\mathcal{P}$, and τ a type. In the following, we will denote $\Lambda = \{\wedge, \vee, \Rightarrow, \mathsf{U}, \bigcirc, \Diamond, \Box\}$ the set of all logical operators. For each operator $\lambda \in \Lambda$, we also note $ar(\lambda) \in \{1, 2\}$ the arity of the operator.

This formulation is akin to Linear Temporal Logic on finite traces (LTL_f) [17], where propositional variables are replaced with first-order predicates and variables. Notice that we only work with formulas in prenex normal form.

Semantics Any quantifier-free formula φ of $\mathcal{L}_{\text{TL}}$ can be evaluated against a trace $t = (s_0, \ldots, s_n)$, at any step. When $i \in [\![0, n]\!]$, we write $t, i \models \varphi$ to denote that formula φ is true at state s_i of trace t. In that case, temporal modalities, such as $\bigcirc$, $\Diamond$, $\Box$, etc., are used to reason over the states that follow or precede the current state s_i.

For instance, $\bigcirc\varphi$ intuitively means that property φ is true in the next state, while $\Diamond\varphi$ means that φ is eventually true, in one of the (iterated) successors of the current state. $\Box\varphi$ means that φ is true from this state on, until the end of the trace, and $\varphi_1 \mathsf{U} \varphi_2$ means that φ_2 is true in some successor state, and until then, φ_1 is true. Operators $\overline{\bigcirc}$, $\overline{\Diamond}$ and $\overline{\Box}$ are the respective *past* counterparts of the previous connectors: $\overline{\bigcirc}\varphi$ means that φ is true in the previous state, $\overline{\Diamond}\varphi$ that φ is true in some previous state, and $\overline{\Box}\varphi$ that φ is true in every previous state.

To illustrate the language, we introduce the Childsnack problem, which originates from the International Planning Competition (IPC). It consists in making sandwiches and serving them to a group of children, some of whom are allergic to gluten. Sandwiches can only be prepared in the kitchen, and then have to be put on trays, which is the only way they can be brought to the children for service. Among the following FTL formulas, the first indicates that "All children will eventually be served" (and will be satisfied by any solution-plan). The second formula indicates that every sandwich x will eventually be put on some tray, at a moment $t+1$. For every moment that precedes moment t, x will not be prepared yet (which indicates that the sandwich is actually put on the tray right after being prepared).

$$\forall x \in \text{Child}.\, \Diamond\, \text{served}(x) \tag{1}$$

$$\forall x \in \text{Sandwich}.\, \exists y \in \text{Tray}.\ \text{notprepared}(x)\ \mathsf{U}\ \bigcirc\, \text{on}(x, y) \tag{2}$$

Temporal modalities can be expressed in terms of one another - the same way propositional connectors can be expressed in terms of one another. For any quantifier-free formula φ, we have $\Diamond\varphi \equiv \top \mathsf{U} \varphi$, $\Box\varphi \equiv \neg\Diamond\neg\varphi$ and $\overline{\Box}\varphi \equiv \neg\overline{\Diamond}\neg\varphi$. This leads us to an inductive definition of the semantics of our language, for quantifier-free formulas of $\mathcal{L}_{\text{TL}}$:

$t, i \models p(x, \ldots, x)$	iff $p(x, \ldots, x) \in s_i$
$t, i \models \neg\varphi$	iff $t, i \not\models \varphi$
$t, i \models \varphi_1 \wedge \varphi_2$	iff $t, i \models \varphi_1$ and $t, i \models \varphi_2$
$t, i \models \bigcirc\varphi$	iff $i < n$ and $t, (i+1) \models \varphi$
$t, i \models \overline{\bigcirc}\varphi$	iff $i > 0$ and $t, (i-1) \models \varphi$
$t, i \models \overline{\Diamond}\varphi$	iff $\exists j \in [\![0, i]\!]$ s.t. $t, j \models \varphi$
$t, i \models \varphi_1 \mathsf{U} \varphi_2$	iff $\exists j \in [\![i, n]\!]$ s.t. $t, j \models \varphi_2$ and $\forall k \in [\![i, j-1]\!], t, k \models \varphi_1$

We write $t \models \varphi$ as a shorthand for $t, 0 \models \varphi$, which means that trace t satisfies the formula φ, since it is true in the initial state of t.

A formula $\psi \in \mathcal{L}_{\text{FTL}}$ is evaluated against instantiated traces, as defined below:

Definition 7 (Instantiated trace) *An* instantiated trace *is a pair $\langle t, \mathcal{I} \rangle$ such that t is a trace where fluents are built on the objects of the planning instance $\mathcal{I}$.*

For any formula ϕ of $\mathcal{L}_{\text{FTL}}$, let us denote $\phi\,[x/y]$ the formula of $\mathcal{L}_{\text{FTL}}$ where each occurrence of y is replaced by x. The semantics of $\mathcal{L}_{\text{FTL}}$ is defined as follows:

$\langle t, \mathcal{I} \rangle \models \forall x \in \tau.\psi$	iff for all $o \in \mathcal{O}$ s.t. $\tau(o) = \tau$, $\langle t, \mathcal{I} \rangle \models \psi\,[o/x]$
$\langle t, \mathcal{I} \rangle \models \exists x \in \tau.\psi$	iff there exists $o \in \mathcal{O}$ s.t. $\tau(o) = \tau$, $\langle t, \mathcal{I} \rangle \models \psi\,[o/x]$
$\langle t, \mathcal{I} \rangle \models \varphi$	iff $t \models \varphi$

where x is a variable, and φ is a formula of $\mathcal{L}_{\text{TL}}$ (thus quantifier-free).

Note that it is well known that the past modalities do not change the expressivity of LTL. As a consequence, our language could have expressed the same properties without modalities $\overline{\bigcirc}$, $\overline{\Diamond}$ or $\overline{\Box}$. However, as these modalities can make some properties exponentially more succinct to express [14], we chose to include them in our language.

2.3 The MaxSAT problem

Let Var be a set of propositional variables. The Boolean satisfiability problem (SAT) is concerned with finding a valuation that satisfies a propositional formula ϕ. Propositional formulas are defined as follows, where $x \in \text{Var}$ is a propositional variable:

$$\phi := \top \mid x \mid \neg\phi \mid \phi \vee \phi \mid \phi \wedge \phi$$

The maximum Boolean satisfiability problem (MaxSAT) is a variant of SAT, in which a valuation of the variables Var of a set of formulas $\{\phi_1, \ldots, \phi_n\}$ is sought. Each formula ϕ_i is assigned a *weight* $w(\phi_i) \in \mathbb{R} \cup \{\infty\}$. The MaxSAT problem consists in finding a valuation v of Var such that the sum of the weights of the formulas that are not satisfied by v is minimal.

3 The $\mathcal{L}_{\text{FTL}}$ learning problem

In this section, we introduce the main problem we are tackling in this paper. In the following, we use $[\langle t, \mathcal{I}\rangle \models \psi]$ as a shorthand for the function equal to 1 if $\langle t, \mathcal{I}\rangle \models \psi$ and equal to 0 otherwise.

Problem 1 *$\mathcal{L}_{FTL}$ learning*

Input: *$\mathcal{D}$ a domain*
T a set of instantiated traces
$r \in \mathbb{N}$ the maximum number of logical operators in the output formula
$q \in \mathbb{N}$ the maximum number of quantifiers
$\sigma : \mathsf{T} \to \mathbb{R}$ a function called the score function

Output: *A formula $\psi \in \mathcal{L}_{FTL}$ such that ψ has at most r logical operators, and q quantifiers, and*
$$\sum_{\langle t, \mathcal{I}\rangle \in \mathsf{T}} \sigma(\langle t, \mathcal{I}\rangle)\,[\langle t, \mathcal{I}\rangle \models \psi]$$
is maximal

Even though the problem above is expressed as an optimization problem, various associated decision problems can be of interest. For example, a problem of interest is the one where ψ must be satisfied by all instantiated traces $\langle t, \mathcal{I}\rangle$ such that $\sigma(\langle t, \mathcal{I}\rangle) \geq 0$ and falsified by all other instantiated traces given in input.

Conjecture 1 *The decision problem associated to the $\mathcal{L}_{FTL}$ learning problem is NP-hard.*

Various authors tried to settle the complexity of problems related to ours, without always succeeding, even when dealing with simpler languages. Notable works include [6], where the authors show interest in the problem of learning various fragments of LTL. Even though the learning problems associated to several fragments were shown to be NP-complete, the complexity of the problem associated to the whole language is still open.

Membership in PSPACE Given an environment e, a trace t, and a formula $\varphi \in \mathcal{L}_{\text{TL}}$, checking that $t, e \models \varphi$ can be done in space polynomial in $|t|$, $|e|$ and $|\varphi|$ (ex. [7]). The model-checking of $\psi \in \mathcal{L}_{\text{FTL}}$ against some $\langle t, \mathcal{I}\rangle$ can be done by enumerating all relevant environments $e \in O^q$, and checking that $t, e \models \varphi$, where φ is the quantifier-free part of ψ. As a consequence, the $\mathcal{L}_{\text{FTL}}$ learning problem is in PSPACE. Even though this shows membership, the potential PSPACE-hardness of our problem is still an open problem.

Score function The choice of the score function allows us to express preferences on which traces are the most important to capture in the output formula, and which traces are the most important to avoid. In the rest of this article, we will say that an instantiated trace $\langle t, \mathcal{I}\rangle$ is *positive* iff $\sigma(\langle t, \mathcal{I}\rangle) \geq 0$. Otherwise, the instantiated trace is said to be *negative*.

4 Planning problem preprocessing

We present in this section the transformations we bring to the PDDL planning problem before it is passed to our algorithm for learning $\mathcal{L}_{\text{FTL}}$ formulas. As our algorithm is based on a compilation of the $\mathcal{L}_{\text{FTL}}$ learning problem into MaxSAT, reducing the size of the compiled form is crucial for it to run in reasonable time.

Predicate splitting Each predicate is split into several predicates of size 2, in order to curb the number of fluents while conserving the links between pairs of objects. This allows us to synthesize formulas containing predicates of high arity, while keeping the number of quantifiers of the formula low.

Concretely, a predicate of the form $p(x, y, z)$ will be split into newly-created predicates $p_{12}(x, y)$, $p_{13}(x, z)$, and $p_{23}(y, z)$. Notice that mathematically, predicate splitting leads to significantly fewer fluents than if the task was to be grounded as is: for a predicate of arity $n \geq 2$, to be grounded with instance $\mathcal{I}$, there are $O(n^2|O|^2)$ associated fluents, while there would be $O(|O|^n)$ if the predicate was not split.

Even though the planning model thus obtained is less rich than the original one, we argue that predicate splitting allows us to learn formulas that would be otherwise out of computational reach of our procedure.

Goal predicates The language $\mathcal{L}_{\text{FTL}}$ naturally allows us to reason on the initial state. However, in its current state, it does not allow reasoning on the goal conditions, which depend on the instance and are trace-agnostic.

We fix this issue by introducing *goal predicates*. For every predicate $p \in \mathcal{P}$, we introduce the predicate p^G. Then, for each instance $\mathcal{I}$, we introduce the *latent state* $s_{\mathcal{I}}$, which is intuitively a set of fluents that are true in every state of every trace associated to $\mathcal{I}$.

For every fluent $p(o_1, \ldots, o_{ar(p)})$ of the goal state G of $\mathcal{I}$, we add the fluent $p^G(o_1, \ldots, o_{ar(p)})$ to $s_{\mathcal{I}}$.

For readability reasons, rather than writing $p^G(o_1, \ldots, o_{ar(p)})$, we will often denote this using a (fictitious) modality named Goal. We thus write $\mathsf{Goal}\big(p(o_1, \ldots, o_{ar(p)})\big)$, as this is a clearer way to indicate that $p(o_1, \ldots, o_{ar(p)})$ is true in any goal state.

Equality predicates In addition to goal predicates, we also add to the latent state *equality predicates*. These are simply predicates of the form $=_\tau\ (x, y)$, where τ is a type. During the preprocessing of instantiated trace $\langle t, \mathcal{I} \rangle$, we add the fluents $=_{\tau(o)}\ (o, o)$ to the latent state, for every object $o \in O$.

5 Topology-based guiding

TL chains An interesting representation for formulas φ of $\mathcal{L}_{\text{TL}}$ is a representation as *TL chains*. They are the adaptation to our language of the notion of chain [13, 19], which is useful for representing formulas of modal or propositional logic.

A *TL chain* is a Directed Acyclic Graph (DAG) which has three types of nodes: logical connector nodes (represented as $\circ$ in the example of Figure 1), predicate nodes (represented as $\diamond$) and variable nodes (represented as $\square$). In order to represent a correct $\mathcal{L}_{\text{TL}}$ formula, logical connector nodes can only be children of logical connector nodes, predicate nodes children of logical connector nodes, and variable nodes children of predicate nodes. We also impose that every leaf is a variable node. In addition, to stay consistent with our language and the choices we made in Section 4, we only work with TL chains that are binary trees, whose inner nodes have exactly two children.

By assigning a symbol of the correct type (i.e., a logical connector, a predicate symbol or a variable) to each node, we end up with a representation of a $\mathcal{L}_{\text{TL}}$ formula. Figure 1 shows the representation as a TL chain of the formula $(q(v, u) \wedge r(z, y)) \,\mathsf{U}\, p(t, x)$.

For each connector node i of the TL chain, we will denote $succ_L(i)$ (resp. $succ_R(i)$) the left (resp. right) child of node i. It is guaranteed to exist, even though it might sometimes be a predicate node. In the case of connectors $\alpha \in \{\neg, \bigcirc, \overline{\bigcirc}, \Diamond, \overline{\Diamond}, \Box, \overline{\Box}\}$ that have arity 1, we will use the convention that the value of the right successor is ignored (and will not appear in the $\mathcal{L}_{\text{FTL}}$ formula that ensues), and the left successor will be the root of the formula under the operator α.

In order to alleviate the pressure on the MaxSAT solver, we impose the topology of the output quantifier-free formula before encoding the problem into a propositional formula. This idea was first introduced in [19], in an attempt to speed up the search for an LTL formula. In addition, we also fix the quantifiers of the formula before the encoding, as well as the types they quantify on. At the end of the day, all that is left to the MaxSAT solver is to "fill in the blanks" in the TL chains that it is given, so that the associated $\mathcal{L}_{\text{FTL}}$ formula fits the input as well as possible.

An interesting aspect of the constraints we impose on the form of the output formula, is that they force the algorithm to produce a wide diversity of formulas.

Quantifiers In the rest of the article, for practical reasons, we restrict ourselves to learning formulas of the form $\forall x_1 \cdots \forall x_k \exists x_{k+1} \cdots \exists x_b \varphi$, where φ is a formula of $\mathcal{L}_{\text{TL}}$, for which every argument of every predicate is a variable x_i. This choice makes some properties impossible to express, and it was made with the aim of limiting the number of MaxSAT intances to solve, as well as to curb the size of the MaxSAT encoding. However, as will be shown by our experimental evaluation, interesting formulas can still be learnt. Our encoding could be easily modified so that any sequence of quantifiers can be worked with, but we leave the experimental evaluation of such a program to future work.

6 Reduction to MaxSAT

6.1 Learning algorithm

Algorithm 1 summarizes the procedure that we use to learn $\mathcal{L}_{\text{FTL}}$ formulas out of our input. The subroutines work as follows: gen_TLchains(r) enumerates every TL chain having exactly r connectors. gen_quantifiers(q) enumerates sequences of quantifier symbols of size q, such that all universal quantifiers $\forall$ appear before existential quantifiers $\exists$. gen_types($\mathcal{D}$, q) enumerates every q-combination of types in the type tree $\mathcal{T}$ of $\mathcal{D}$. Finally, the main subroutine, find_formula($\mathcal{D}, \mathsf{T}, \rho, \{Q_i\}, \{\tau_i\}, \sigma$), encodes the problem of finding an $\mathcal{L}_{\text{FTL}}$ formula fitting the instantiated traces of T, with the constraints imposed by the TL chain ρ, the quantifiers $\{Q_i\}$, and the types $\{\tau_i\}$. find_formula then returns (one of) the best formula(s) it finds, or the token FAIL is none is found.

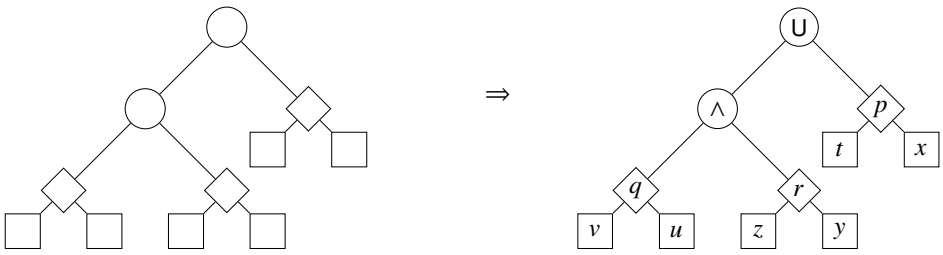


Figure 1: A TL chain example, as well as a possible assignation of symbols to its nodes. The TL chain on the right has been assigned symbols to every one of its nodes, and represents the formula $(q(v,u) \wedge r(z,y)) \,\mathsf{U}\, p(t,x)$

Algorithm 1: $\mathcal{L}_{\text{FTL}}$ learning

Input: Domain $\mathcal{D}$, traces T, parameters r, q, and function σ
Output: A set of $\mathcal{L}_{\text{FTL}}$ formulas
found_formulas := [ ]
for $\rho \in$ *gen_TLchains(r)* **do**
  for $Q_1, \ldots, Q_q \in$ *gen_quantifiers(q)* **do**
    for $\tau_1, \ldots, \tau_q \in$ *gen_types($\mathcal{D}$, q)* **do**
      $\psi \leftarrow$ find_formula($\mathcal{D}, \mathsf{T}, \rho, \{Q_i\}, \{\tau_i\}, \sigma$);
      if $\psi \neq$ *FAIL* **then**
        found_formulas.add(ψ);
    end
  end
end
return *found_formulas*

6.2 Preliminaries to the encoding

Notion of environment The constraints require us to reason on which combinations of objects may satisfy a formula or not. Indeed, as we work with a logic reminiscent of first-order logic (on finite domains), we are required to reason on various sets of objects at once. We now introduce the concept of environment, which allows us to work on assignations of (sets of) objects to variables of the formula.

Let us suppose that the formula we try to synthesize ranges over the variables $\mathcal{X} = (x_1, \ldots, x_q)$. In addition, let $\mathcal{I}$ be an instance, with objects $O = \{o_1, \ldots, o_{|O|}\}$. We call a *partial* environment any assignation of some of the variables $x_1, \ldots, x_q$ to an object of O. Let us denote $\mathsf{var}(e)$ the variables that are assigned an object within the partial environment e. When $\mathsf{var}(e) = \mathcal{X}$, we simply say that e is an environment.

We denote any (partial) environment $e = \{x_1 := o_{i_1}, \ldots, x_q := o_{i_q}\}$, where $i_1, \ldots, i_q \in [\![1, |O|]\!]$.

We also denote $p(x,y)[e]$ the grounding of an atom $p(x,y)$ by an environment e such that $x, y \in \mathsf{var}(e)$. If $e = \{x := o_1, y := o_2, \ldots\}$, then $p(x,y)[e] = p(o_1, o_2)$. By extension, the formula obtained when grounding each atom of φ with e is written $\varphi[e]$.

The main difficulty that rises when working with environments is that there are $|O|^q$ different environments. As O can be large, the number of environments being exponential in the number of quantifiers quickly makes the problem intractable, if no restriction is posed. This is why we ensure that each variable of the $\mathcal{L}_{\text{FTL}}$ formula to learn is assigned a type, so that not every environment has to be considered during search: the types are chosen before proceeding to the encoding.

Variables Our MaxSAT encoding is built on the set of variables that follows. When possible, we use the following conventions, as closely as possible: nodes of the FL-chain are denoted by i when they are logical connectors (represented by ◯ in Figure 1), by ℓ when they are predicate nodes (represented by ◊), and by v when they are first-order variable nodes (represented by □). A trace is denoted by t, and a position in this trace is denoted by k (i.e., the k-th state). Moreover, j is an index for a variable of the quantifiers, and p is a predicate.

This leads us to the following variables, as will be used in the MaxSAT encoding. Greek letters denote decision variables while latin characters are for "technical" variables.

- $y_i^{t,k}[e]$: In position k of trace t, with environment e, the formula rooted at node i is true.
- $\delta_{j,v}^{\ell}$: The v-th variable of predicate node ℓ is the variable of quantifier j.
- θ_{ℓ}^{p}: The predicate of node ℓ is p.
- λ_i^q: The logical connector at node i is q.
- s_t: Trace t is currently satisfied by the first order formula

"Exactly one" constraints In the encoding of a problem into SAT, some situations require that *at most one* variable, out of a set of variables, is true. There exist encodings that are more efficient than the naive one to define such "at most one" constraints: see for instance [10, 16] for a

survey on these encodings. Like many other solvers, the MaxSAT solver that we use proposes a built-in function for this. More specifically, it offers a built-in function for *exactly one* constraints, where exactly one variable of a set must be true in a model of the formula.

In the following, we will denote $\mathsf{ExactlyOne}_{s\in S}(v_s)$ the set of propositional constraints enforcing that at most one of the variables of $\{v_s \mid s \in S\}$ is true.

6.3 Core constraints

Some of the constraints below are adapted from [8, 19, 15], which are concerned with LTL. Our main contribution is the adaptation of the encoding to our language $\mathcal{L}_{\text{FTL}}$, which differs from LTL by its tighter links with PDDL planning models through first-order components.

In the following, we suppose that an empty TL chain ρ has been computed, and that the associated quantifiers and types have been decided. We will denote n its number of connector nodes, and m its number of predicate nodes. As a consequence, there are $2m$ variables nodes. As previously, the number of quantifiers is denoted q. The first $b \leq q$ quantifiers are universal, while the other are existential.

We also suppose that the types on which the quantifiers range, denoted $\tau_1, \ldots, \tau_q$, are already chosen. As a consequence, in this section, the set of relevant environments for instance $\mathcal{I}_t$ associated to trace t, denoted $E_{\mathcal{I}_t}$, only consists of environments of the form $\{x_u := o_u\}_{1\leq u\leq q}$ where, $\tau(o_u) = \tau_u$, for $u \in [\![1, q]\!]$.

Syntactic constraints This section describes the constraints that ensure that the formula is syntactically well-formed.

The following constraints ensure that every logical connector node has exactly one logical connector assigned, and every predicate node has exactly one predicate, respectively. Recall that Λ is the set of all logical operators.

$$\bigwedge_{i\leq n} \mathsf{ExactlyOne}_{c\in\Lambda}(\lambda_i^c) \tag{3}$$

$$\bigwedge_{\ell\leq m} \mathsf{ExactlyOne}_{p\in\mathcal{P}}(\theta_\ell^p) \tag{4}$$

Finally, the following constraints force each argument of each predicate to be bound to a variable on which the formula quantifies.

$$\bigwedge_{\ell\leq m}\bigwedge_{s\in\{1,2\}} \mathsf{ExactlyOne}_{j\leq b}(\delta_{j,s}^\ell) \tag{5}$$

Semantic constraints These constraints ensure that the formula found by the solver is consistent with the traces. It mimics the model-checking algorithm for modal logic.

The following clauses ensure that the formula ψ that is synthesized is consistent with the traces of T. This is made in accordance with the environments imposed by the quantifier, which are iterated upon. The variable s_t is true iff for every required environment e, $\varphi[e]$ is satisfied by t (where $\varphi[e]$ is the evaluation of formula φ in environment e, and φ is the quantifer-free part of the formula we synthesize). Thus, for every trace $t \in \mathsf{T}$, we add the following:

$$s_t \Leftrightarrow \left(\bigwedge_{\substack{o_1\in O_1\\ \cdots\\ o_k\in O_k}} \bigvee_{\substack{o_{k+1}\in O_{k+1}\\ \cdots\\ o_q\in O_q}} y_1^{t,1}[\{x_u := o_u\}_{1\leq u\leq q}] \right) \tag{6}$$

The following constraints ensure that formulas that consist of a single literal (i.e., a positive or negative fluent) are consistent with the y variables, that give the truth value of a trace at a certain position in the trace, at each node of the TL chain.

Such constraints appear once for every trace $t \in \mathsf{T}$, for every position $k \leq |t|$ of this trace, for every predicate node $\ell \leq m$ and every predicate $p \in \mathcal{P}$, for every pair of quantifiers (positions) $j_1, j_2 \leq q$, and for each relevant environment $e \in E_{\mathcal{I}_t}$.

$$\theta_\ell^p \wedge \delta_{j_1,1}^\ell \wedge \delta_{j_2,2}^\ell \Rightarrow \begin{cases} y_\ell^{t,k}[e] & \text{if } t[k] \models p(x_{j_1}, x_{j_2})[e] \\ \neg y_\ell^{t,k}[e] & \text{otherwise} \end{cases} \tag{7}$$

The constraints sketched in equations (8) to (11) appear once for each connector node $i \leq n$ of the formula, each position $k \leq |t|$ of each trace $t \in \mathsf{T}$, and for each environment $e \in E_{\mathcal{I}_t}$. They ensure that the logical operators are correctly interpreted.

In the case where the logical connector at node i is a negation $\neg$, we have:

$$\lambda_i^\neg \Rightarrow \left(y_i^{t,k}[e] \Leftrightarrow \neg y_{succ_L(i)}^{t,k}[e] \right) \tag{8}$$

In the case of $\Delta \in \{\wedge, \vee, \Rightarrow\}$:

$$\lambda_i^\Delta \Rightarrow \left(y_i^{t,k}[e] \Leftrightarrow \left(y_{succ_L(i)}^{t,k}[e] \;\Delta\; y_{succ_R(i)}^{t,k}[e] \right) \right) \tag{9}$$

In the case of the next operator $\bigcirc$, we have the following:

$$\lambda_i^\bigcirc \Rightarrow \left(y_i^{t,k}[e] \Leftrightarrow y_{succ_L(i)}^{t,k+1}[e] \right) \tag{10}$$

with the convention that $y_{succ_L(i)}^{t,|t|+1}[e]$ is replaced by $\bot$ during the encoding itself.

In the case of the finally operator $\Diamond$:

$$\lambda_i^\Diamond \Rightarrow \left(y_i^{t,k}[e] \Leftrightarrow \bigvee_{\substack{k'\\ k\leq k'\leq |t|}} y_{succ_L(i)}^{t,k'}[e] \right) \tag{11}$$

The case of the temporal operators $\Box$, $\overline{\bigcirc}$, $\overline{\Diamond}$, $\overline{\Box}$ and U can be encoded in a way that is similar to the constraints above.

Well-formed fluents constraints The following constraints ensure that, in the output formula ψ, there is a consistency between the types of the variables and the arguments of predicates are assigned to. Otherwise said, when a variable x of type τ is chosen to be the v-th argument of a predicate p that occurs in ψ, we require that $\tau = \tau_p(v)$. This can be done through the following constraints:

$$\bigwedge_{j \leq q} \bigwedge_{\ell \leq m} \bigwedge_{p \in \mathcal{P}} \bigwedge_{j \leq q} \bigwedge_{\substack{v \leq 2 \\ \tau_p(v) \neq \tau_j}} \neg\theta^p_\ell \vee \neg\delta^\ell_{j,v} \tag{12}$$

Weights for the MaxSAT solver Recall that we wish to find a formula ψ that maximizes the following function:

$$\sum_{\langle t, \mathcal{I} \rangle \in \mathsf{T}} \sigma(\langle t, \mathcal{I} \rangle) \, [\langle t, \mathcal{I} \rangle \models \psi]$$

The objective of the MaxSAT solver is to minimize the total weight of the falsified soft clauses. As such, for each instantiated trace $\langle t, \mathcal{I} \rangle$, we add the clause s_t, with weight $\sigma(\langle t, \mathcal{I} \rangle)$. This penalizes formulas that falsify traces with a positive score, while rewarding formulas that falsify traces with a negative score.

Pruning non-discriminatory formulas With a given configuration of TL chain, quantifiers and types, it is not guaranteed that there exists a formula ψ that captures (some of) the positive traces while falsifying (some of) the negative traces. Without further precautions, our algorithm can output formulas that are true on all traces, or false on all traces. These formulas are often tautologies or unsatisfiable, and still have a non-zero score as they completely capture the positive or negative traces.

The following clauses ensure that at least one positive trace and one negative trace are captured:

$$\bigvee_{\substack{t \in \mathsf{T} \\ \sigma(t) \geq 0}} s_t \wedge \bigvee_{\substack{t \in \mathsf{T} \\ \sigma(t) < 0}} \neg s_t \tag{13}$$

6.4 Formula quality enhancement

The constraints presented in this section filter the solutions so that less interesting formulas, or formulas that could be computed by a run of our algorithm with smaller parameters, are barred from being output.

Syntactic redundancies prevention These constraints prevent idempotent and involutive modalities and operators from being chained in the output formula. These include the negation $\neg$, as well as the temporal operators $\Diamond$ (for which $\Diamond\Diamond\varphi \equiv \Diamond\varphi$) and $\Box$ (which is, likewise, idempotent). In order to prevent the operator $\alpha \in \{\neg, \Diamond, \overline{\Diamond}, \Box, \overline{\Box}\}$ from appearing in a node of the TL chain and its left-successor, we add the following constraints, when possible (i.e. when both i and $succ_L(i)$ are defined):

$$\neg\lambda^\alpha_i \vee \neg\lambda^\alpha_{succ_L(i)} \tag{14}$$

In addition, we prevent redundancies of the form $p(x, y) \, \Delta \, p(x, y)$, where $\Delta \in \{\wedge, \vee, \mathsf{U}, \Rightarrow\}$ is a binary operator. In every case, there exists a smaller (sub-)formula that can be found and that expresses the same thing, without the redundant atom. For each connector node i which has two predicate nodes as children, denoted as $\ell_l := succ_L(i)$ and $\ell_r := succ_R(i)$, we add the following clauses:

$$\bigwedge_{p \in \mathcal{P}} \bigwedge_{\substack{j_1, j_2 \leq q \\ j_2 \neq j_1}} \neg \left(\theta^p_{\ell_l} \wedge \delta^1_{j_1, \ell_l} \wedge \delta^2_{j_2, \ell_l} \wedge \theta^p_{\ell_r} \wedge \delta^1_{j_1, \ell_r} \wedge \delta^2_{j_2, \ell_r} \right)$$

The constraints above actually prevent two fluents that are adjacent (i.e. that are connected by a binary operator) to be equal, regardless of the actual operator that links them.

Variable visibility As the size of the encoding is exponential in the number q of expected quantifiers, we wish to ensure that every variable that we quantify upon in the output formula ψ also appears in an atom of ψ. Otherwise, an equivalent formula could be found by running the algorithm with fewer quantifiers. This is why we force each variable to appear at least once in some atom. For space reasons, we skip the presentation of the constraints.

7 Experiments

This section presents the experiments we ran in order to assess the performances of our method. The implementation was done in Python 3.10, using the MaxSAT solver Z3 [5]. Experiments were conducted on a machine running Rocky Linux 8.5, powered by an Intel Xeon E5-2667 v3 processor, with a 24-hours cutoff and using at most 16GB of memory per run. The code of our implementation can be found online[1]. The repository also includes tools to evaluate formulas against a dataset and a planning model.

7.1 Performances of the learning algorithm

To assess the performances of our algorithms, we considered 6 domains from various editions of the International Planning Competition (IPC), some of which are described in Section 7.2. For each of these domains, we generated 10 instances that model problems with similar goals. We then used 5 planners from the IPC to generate plans for each instance, that our algorithm converted to traces. On average, plans had 10.2 operators, but some instances include plans of size up to 23.

[1] https://github.com/arnaudlequen/LearningEngine

Table 1: Average amortized time in seconds (s) required to learn a single formula from our dataset, depending on the quantifiers imposed and the maximum number of logical operators. Values in the table represent the total running time of the algorithm, divided by the total number of formulas found. This represents the average time between two formula outputs. Entries labeled by a dot (.) represent training instances that reached the time cutoff.

$\|\varphi\|$	Quantifiers				
	$\forall$	$\exists$	$\forall\forall$	$\forall\exists$	$\exists\exists$
2	4.2	4.3	36.5	41.0	34.8
3	9.8	8.7	53.8	56.9	48.7
4	22.2	20.5	.	.	.

We built our training instances by selecting 3 planning instances of each domain, and the associated traces for each planner - for a total of 15 instantiated traces per training instance. We then created the tasks of finding a formula recognizing the behavior of each planner, and ran our algorithm on each such task. The remaining instances were used for our test set.

The aim of these experiments was to test our algorithm in a setting where it would struggle. Indeed, planners often output plans that are similar to each others, as they are close to optimal, and do not exhibit particularly distinctive behaviors. This makes finding a formula that perfectly recognizes the behavior of a planner hard. Nonetheless, our algorithm still managed to output formulas that *imperfectly* recognize a planner's behaviour.

The average amortized times to synthesize a formula on our dataset are summarized in Table 1. In our tests, a formula was found in 87.5% of the solving attempts of our algorithm. Since 73.2% of the running time is dedicated to the encoding, we ask the MaxSAT solver to output multiple solutions for each encoding.

7.2 Examples of learnt formulas

In this section, we present some formulas that have been learnt by our algorithm. We considered three domains among the ones used in our data set. For the first two domains, since the plans found by the 5 planners were all very similar, we handcrafted various agents that tackle the problem in a distinctive way. We kept off-the-shelf planners for the last domain. We then built sets of plans in a similar way as in the previous section.

Spanner Instances of the Spanner domain involve an operator that has to go from a shed to a gate to tighten some nuts, passing through a sequence of locations where single-use spanners can be picked up. Once a location is left, it can not be returned to. Thus, collecting enough spanners before reaching the gate is seminal for solving the problem.

We developed three different behaviours for this domain. Agent ALL picks every possible spanner on its way to the gate, while agent SME picks exactly as many spanners as are needed to tighten the nuts at the gate. Agent SGL takes a single spanner and rushes to the gate, and can then only tighten one nut.

Among the formulas that perfectly recognize plans belonging to agent ALL, we have the following:

$$\forall x \in \text{Spanner}.\ \exists y \in \text{Operator}.\ \Diamond\,\Box\,\text{carrying}(y, x) \tag{15}$$

$$\forall x \in \text{Spanner}.\ \exists y \in \text{Location}.\ \text{at}(x, y) \wedge \Diamond\neg\text{at}(x, y) \tag{16}$$

Formula (15) expresses that every spanner will be picked up by the (only) operator and carried for the rest of the plan, and Formula (16) expresses that every spanner will be moved from its initial position at some point.

Even though we also managed to learn a formula that perfectly discriminates agent SGL from the others, we failed to learn a formula completely capturing SME's behaviour.

When searching formulas with a single variable, we split the predicates so that the maximum arity of a fluent is 1. Our algorithm output the following formulas, which were learnt in a few seconds, and completely characterize the behavior of agent ALL:

$$\forall x \in \text{Spanner}.\ \Diamond\,\text{carrying}_2(x) \tag{17}$$

$$\forall x \in \text{Spanner}.\ \text{useable}(x)\ \mathsf{U}\ \text{carrying}_2(x) \tag{18}$$

Predicate splitting allows us to obtain a concise formula, where the focus is clear (every spanner x is eventually carried), as uninformative elements (who carries the spanner) are omitted.

Childsnacks Domain Childsnacks, as introduced in Section 2, consists in making sandwiches and serving them to a group of children, some of whom are allergic to gluten. Sandwiches can only be prepared in the kitchen, and then have to be put on trays, which is the only way they can be brought to the children.

We designed three different agents that solve Childsnacks instances. Agents NGF and NGL compute solution plans of minimal size, and differ in that agent NGF makes sandwiches with *no gluten first*, and agent NGL makes sandwiches with *no gluten last*. Both agents make all sandwiches, put them on a tray, then serve the children. Agent GS greedily serves children: as soon as a sandwich is made, it is put on a tray and brought to a child. It also prioritizes gluten-free sandwiches.

In every instance, 2 trays are initially in the kitchen. The only difference between instances is in the number of children to serve, the smallest having 2. This is, however, enough to learn a wide variety of formulas that perfectly recognize the behavior of agent GS (among others) on our

test set. Such formulas include, for instance, the following:

$$\forall x \in \text{Kitchen}.\, \exists y \in \text{Tray}.\, \Diamond(\text{at}(y,x) \wedge \overline{\Diamond}\neg\text{at}(y,x)) \quad (19)$$

$$\forall x \in \text{Kitchen}.\, \exists y \in \text{Tray}.\Diamond(\neg\text{at}(y,x) \wedge \bigcirc\text{at}(y,x)) \quad (20)$$

Formula (19) expresses that Agent GS eventually comes back to the kitchen with some tray y, even though the tray was brought out of the kitchen at some point in the past. Formula (20) expresses the same idea, but pinpoints the moment when a tray is brought back to the kitchen.

Both formulas manage to perfectly capture our test set, but do not perfectly capture our *training* set. This is due to the fact that the smallest instance of our training set contains as many children as there are trays, and thus, no tray has to be brought back to the kitchen. Our use of a reduction to MaxSAT allows us to be resilient to this kind of edge cases, and the formulas that are learnt are satisfactory despite not perfectly fitting the training set.

Even though our algorithm managed to learn concise formulas that perfectly capture agent NGL's behaviour, it failed to find in reasonable time a formula that discriminates agent NGF's behaviour with reasonable accuracy.

Rovers Domain Rovers simulates a planetary exploration mission, where a fleet of mobile rovers has to navigate between various waypoints on a planet to collect data or samples, and to transmit the data back to a lander. The rovers have instruments, which have to be calibrated before they can collect data.

The set of instances we designed all required a single rover to collect two kinds of samples, and to take a picture of an object. The only differences lie in the topology of the planet and the positions of objectives.

For this domain, we resorted to five different planners from the IPC 2018 and 2023 to generate plans, namely BFWS, Odin, TFTM-ArgMax, LAMA and DecStar. Our algorithm managed to learn the following formula (among others), which recognizes the plans of planner BFWS with 93.3% accuracy. The only traces that have been wrongfully recognized are traces of TFTM-ArgMax.

$$\begin{aligned}&\forall x \in \text{Rover}.\forall y \in \text{Waypoint}.\\ &\quad \mathsf{Goal}(\text{communicated\_rock\_data}(y))\\ &\quad \Rightarrow \bigcirc\text{have\_rock\_analysis}(x,y)\end{aligned} \quad (21)$$

$$\forall x \in \text{Rover}.\exists y \in \text{Store}.\text{store\_of}(y,x) \wedge \bigcirc\text{full}(y) \quad (22)$$

The formulas above express the fact that BFWS consistently collects samples as fast as possible, and starts by collecting the rock samples. By opposition, other planners tend to start by calibrating the instruments required to take the picture, before proceeding to explore the planet.

8 Conclusion

In this paper, we have presented a method to learn temporal logic formulas that recognize agents based on examples of their behaviours. We showed that such formulas can be learned using an algorithm that boils down to a reduction to MaxSAT, and that very few examples are sometimes enough to perfectly capture the behaviour of an agent on instances that can differ from the ones used in the training set. This justifies the cost of resorting to a first-order language, which generalizes to new instances.

In future works, we wish to tailor our algorithm and our datasets so that they can generate domain-specific control knowledge. Some other authors [2] have expressed search control knowledge in a language similar to ours, with the aim of guiding the search of a planner designed to use such knowledge. While this knowledge must be written by a human operator, previous works show that it could also be generated automatically [4].

Acknowledgements The author would like to thank Martin C. Cooper, for his insightful suggestions and his careful proofreading of this manuscript; and the reviewers of this paper, for their numerous helpful comments which helped us improve this article.

References

[1] Dana Angluin. Learning regular sets from queries and counterexamples. *Information and Computation*, 75(2):87–106, 1987.

[2] Fahiem Bacchus and Froduald Kabanza. Using temporal logics to express search control knowledge for planning. *Artificial Intelligence*, 116(1):123–191, 2000.

[3] Alberto Camacho and Sheila A. McIlraith. Learning interpretable models expressed in linear temporal logic. *Proceedings of the International Conference on Automated Planning and Scheduling*, 29(1):621–630, May 2021.

[4] Tomás de la Rosa and Sheila McIlraith. Learning domain control knowledge for TLPlan and beyond. In *Proceedings of the ICAPS-11 Workshop on Planning and Learning (PAL)*, 2011.

[5] Leonardo de Moura and Nikolaj Bjørner. Z3: An efficient SMT solver. In *Tools and Algorithms for the Construction and Analysis of Systems*, pages 337–340. Springer Berlin Heidelberg, 2008.

[6] Nathanaël Fijalkow and Guillaume Lagarde. The complexity of learning linear temporal formulas from examples. In Jane Chandlee, Rémi Eyraud, Jeff Heinz,

Adam Jardine, and Menno van Zaanen, editors, *Proceedings of the Fifteenth International Conference on Grammatical Inference*, volume 153 of *Proceedings of Machine Learning Research*, pages 237–250. PMLR, 23–27 Aug 2021.

[7] Valeria Fionda and Gianluigi Greco. The complexity of LTL on finite traces: Hard and easy fragments. In *AAAI Conference on Artificial Intelligence*, 2016.

[8] Jean-Raphaël Gaglione, Daniel Neider, Rajarshi Roy, Ufuk Topcu, and Zhe Xu. Learning linear temporal properties from noisy data: a MaxSAT-based approach. In *Automated Technology for Verification and Analysis: 19th International Symposium, ATVA 2021, Gold Coast, QLD, Australia, October 18–22, 2021, Proceedings 19*, pages 74–90. Springer, 2021.

[9] Malte Helmert. Concise finite-domain representations for PDDL planning tasks. *Artificial Intelligence*, 173(5-6):503–535, 2009.

[10] Steffen Hölldobler and Van-Hau Nguyen. An efficient encoding of the at-most-one constraint. *Technical Report. Technische Universitäat Dresden*, 2013.

[11] Rostislav Horčík and Daniel Fišer. Endomorphisms of lifted planning problems. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 31, pages 174–183, 2021.

[12] Joseph Kim, Christian Muise, Ankit Jayesh Shah, Shubham Agarwal, and Julie A Shah. Bayesian inference of linear temporal logic specifications for contrastive explanations. In *International Joint Conference on Artificial Intelligence*, 2019.

[13] Donald Ervin Knuth. *The Art of Computer Programming*, volume 4A. Addison Wesley, 2020.

[14] Nicolas Markey. Temporal logic with past is exponentially more succinct. *Bull. EATCS*, 79:122–128, 2003.

[15] Daniel Neider and Ivan Gavran. Learning linear temporal properties. In *2018 Formal Methods in Computer Aided Design (FMCAD)*, pages 1–10. IEEE, 2018.

[16] Van-Hau Nguyen and Son T. Mai. A new method to encode the at-most-one constraint into SAT. In *Proceedings of the 6th International Symposium on Information and Communication Technology*, SoICT '15, page 46–53, New York, NY, USA, 2015. Association for Computing Machinery.

[17] Amir Pnueli. The temporal logic of programs. *18th Annual Symposium on Foundations of Computer Science (SFCS 1977)*, pages 46–57, 1977.

[18] Ritam Raha, Rajarshi Roy, Nathanaël Fijalkow, and Daniel Neider. Scalable anytime algorithms for learning fragments of linear temporal logic. In Dana Fisman and Grigore Rosu, editors, *Tools and Algorithms for the Construction and Analysis of Systems*, pages 263–280, Cham, 2022. Springer International Publishing.

[19] Heinz Riener. Exact synthesis of LTL properties from traces. In *2019 Forum for Specification and Design Languages*, pages 1–6. IEEE, 2019.

[20] Maayan Shvo, Andrew C Li, Rodrigo Toro Icarte, and Sheila A McIlraith. Interpretable sequence classification via discrete optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 9647–9656, 2021.

[21] Frits Vaandrager, Bharat Garhewal, Jurriaan Rot, and Thorsten Wißmann. A new approach for active automata learning based on apartness. In *International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, pages 223–243. Springer, 2022.

[22] Chunting Zhou, Chonglin Sun, Zhiyuan Liu, and Francis Lau. A C-LSTM neural network for text classification. *arXiv*, 11 2015.

Session 3 : Choix social

Actes JIAF 2024

Agrégation de Jugements avec une Fiabilité Variable Inconnue

Quentin Elsaesser[1] **Patricia Everaere**[2] **Sébastien Konieczny**[1]

[1] CRIL, CNRS, Université d'Artois
[2] CRIStAL, CNRS, Université de Lille
elsaesser@cril.fr,
patricia.everaere-caillier@univ-lille.fr
konieczny@cril.fr

Résumé

Pour choisir une solution, les méthodes d'agrégation de jugements s'appuient souvent sur le nombre de votes que chaque formule reçoit de la part des agents. Cela suppose implicitement que tous les agents ont la même fiabilité et que tous les votes ont la même importance. Dans ce travail, nous considérons une vision épistémique de l'agrégation de jugements, où nous considérons qu'il existe une vérité sous-jacente. Trouver la vérité en utilisant ce que dit la majorité des agents peut conduire à une mauvaise solution, i.e. une solution où certaines formules n'ont pas la bonne valeur de vérité. L'idée dans ce travail est de suivre les opinions des agents les plus fiables pour trouver la vérité. À cette fin, nous proposons une nouvelle famille de méthodes d'agrégation de jugements qui évaluent la fiabilité des agents et des formules. Cette évaluation est ensuite utilisée pour prendre une décision et trouver la vérité, au lieu de considérer simplement le nombre de votes. Nous présentons une étude expérimentale montrant que ces méthodes donnent de meilleurs résultats pour la recherche de la vérité par rapport aux approches existantes dans la littérature. Nous étudions également les propriétés théoriques de ces méthodes.

Abstract

To choose an outcome, judgment aggregation methods often rely on the number of votes each formula receives from agents. This implicitly assumes that all agents possess equal reliability and that all votes carry identical weight. In this work we consider an epistemic view of judgment aggregation, where we consider that there is an underlying truth. Finding the truth by using what the majority of agents says can lead to a wrong solution, i.e. a solution where some of the formulae do not have the correct truth value. The idea of this work is to follow the opinions of the most reliable agents to find the truth. To this aim, we propose a new family of judgment aggregation methods that evaluate the reliability of the agents and issues. This evaluation is then used to take a decision and find the truth, instead of simply consider the number of votes. We provide an experimental study showing that these methods yield superior results in the truth-tracking task compared to existing approaches in the literature. We also study the theoretical properties of these methods.

1 Introduction

Il existe deux façons d'agréger les croyances de différents agents/sources. La première consiste à définir une nouvelle base de croyances (représentée par une formule ou un ensemble de formules) qui synthétise les croyances du groupe d'agents. Cette façon de faire correspond à la fusion des croyances [29, 19, 20, 9]. La deuxième possibilité consiste à demander à ces agents de répondre à un ensemble de questions, chaque question étant représentée par une formule logique, et chaque agent donnant une réponse oui/non en fonction du fait qu'il croit que la formule est vraie ou fausse. Ceci correspond à l'agrégation de jugements [25, 13, 22].

Même s'il existe des liens entre ces deux domaines [11, 28], ils sont principalement utiles dans des contextes différents. On ne peut donc pas choisir d'utiliser l'agrégation de jugements ou la fusion de croyances. C'est l'application qui dicte si nous avons simplement besoin d'une réponse à certaines questions ou si nous devons considérer une base de croyances complète.

Dans cet article, nous nous concentrerons sur l'agrégation de jugements. Les méthodes d'agrégation de jugements visent à former une décision collective en combinant les jugements individuels des agents qui reflètent de manière appropriée les opinions du groupe sur un ensemble de questions [25, 13].

Suivre les opinions de la majorité des agents peut conduire à des résultats incohérents. Ce dilemme, appelé *paradoxe doctrinal* ou *dilemme discursif* [21, 14, 27, 24] réside dans le fait que l'utilisation du vote majoritaire avec

des agents rationnels (cohérents) peut aboutir à un résultat irrationnel (incohérent). De nombreuses règles d'agrégation ont été proposées pour résoudre ce problème, telles que les méthodes basées sur les *prémisses* [6] qui divisent l'agenda (i.e. l'ensemble des formules propositionnelles) en prémisses et en conclusions et utilisent les prémisses pour prendre une décision, les méthodes dites *séquentielles* [5] qui examinent de manière séquentielle les éléments de l'agenda, les règles basées sur les *quotas* [5] qui associent un quota à chaque proposition de l'agenda et acceptent les propositions qui dépassent ce quota. Une autre famille de règles d'agrégation de jugements utilise les distances et/ou la minimisation pour résoudre le conflit. Par exemple, les règles basées sur la *distance* [28], utilisent la distance entre les ensembles de jugements et le profil pour choisir un résultat collectif. [22] introduisent des méthodes basées sur la *minimisation*, qui minimisent la perte d'information dans le profil. Les méthodes basées sur le *soutien* [10] utilisent le soutien obtenu par les questions pour sélectionner les ensembles de jugements.

Nombre de ces méthodes utilisent le nombre de voix attribuées aux questions pour prendre une décision collective, et la plupart des méthodes, notamment celles basées sur la *minimisation* choisissent l'ensemble de jugements majoritaires (i.e. l'opinion majoritaire sur chaque question) s'il est cohérent. Mais suivre l'opinion de la majorité des agents peut conduire à une mauvaise solution d'un point de vue épistémique, même si le résultat sélectionné est cohérent. Dans l'agrégation de jugement standard, tous les agents sont considérés comme ayant la même fiabilité. Cependant, nous considérons ici le cas plus général (et plus réaliste) où les agents ont des fiabilités différentes et où la fiabilité de chaque agent est initialement inconnue. C'est ce que nous avons appelé la *fiabilité variable inconnue* dans le titre. Et notre objectif est de suivre l'opinion des agents fiables. Nous devons donc évaluer cette fiabilité et tenir compte de cette évaluation pour trouver le bon résultat.

Nous adoptons donc dans cet article une vision épistémique de l'agrégation de jugements. En général, on parle peu de la nature des jugements, qui peuvent souvent être des croyances (quelqu'un est-il coupable ou non ?) ou des objectifs/préférences (devrions-nous aller en Italie pour nos prochaines vacances ?) [1]. Dans ce travail, nous nous concentrons sur les croyances, donc sur les jugements épistémiques. Cela signifie que les agents fournissent leurs croyances dans leurs *ensembles de jugements* et qu'il existe une *vérité*, que nous aimerions identifier avec la méthode d'agrégation de jugements, comme dans la formulation originale du paradoxe doctrinal.

Illustrons notre propos par un exemple. Considérons les formules propositionnelles $\varphi_1 = p, \varphi_2 = p \wedge r, \varphi_3 = r \vee s, \varphi_4 = p \wedge q, \varphi_5 = t$. La vérité est lorsque les cinq formules sont vraies. La Table 1 rapporte les opinions des onze agents sur les cinq formules où 1 signifie que l'agent vote pour la formule et 0 pour la négation de la formule.

	p	$p \wedge r$	$r \vee s$	$p \wedge q$	t
A_1, A_2, A_3	1	1	1	1	1
A_4	1	0	0	1	1
A_5	1	1	1	1	0
A_6	1	1	1	0	1
A_7, A_8, A_9, A_{10}	0	0	0	0	0
A_{11}	0	0	1	0	1
Majorité	1	0	1	0	1
Vrité	1	1	1	1	1

TABLE 1 – Profile

Il convient de noter que les formules de l'agenda sont généralement liées par une contrainte, i.e. que certaines formules s'impliquent mutuellement ou qu'il existe des incohérences entre certaines formules. Nous supposons que chaque agent respecte cette contrainte de cohérence (liée aux formules propositionnelles) lorsqu'il donne son avis. Une méthode de base pour prendre une décision collective consiste à utiliser le vote majoritaire sur une question donnée. Le résultat de ce vote est nommé *Majorité* dans la Table 1. Dans notre cas, l'objectif des agents est de trouver la *Vérité*. Chaque agent donne *sincèrement* son évaluation pour chaque formule, qui est la valeur de vérité de chaque formule en fonction de ses croyances. Sincèrement signifie que les agents ne sont pas censés être malveillants. Le résultat donné par le vote majoritaire est cohérent dans cet exemple, alors de nombreuses règles d'agrégation de jugements prendront ce que la majorité des agents dit comme résultat (cette propriété de sélection de l'ensemble de jugements majoritaires lorsqu'il est cohérent, appelée "Préservation de la Majorité", est souvent considérée comme une propriété attrayante). Néanmoins, ici, le vote majoritaire est différent de la vérité sur les formules φ_2 et φ_4. Les méthodes qui évaluent la fiabilité des agents et la confiance des formules à l'aide de méthodes itératives peuvent aider à trouver correctement la vérité. Dans l'exemple, les quatre agents A_7, A_8, A_9, A_{10} se trompent sur les cinq formules. Ils influencent négativement le résultat du vote majoritaire en obtenant $\neg\varphi_2$ et $\neg\varphi_4$ au lieu de la vérité φ_2 et φ_4 (i.e. $\varphi_2 = 0$ et $\varphi_4 = 0$ au lieu de $\varphi_2 = 1$ et $\varphi_4 = 1$). Si la fiabilité des agents A_7, A_8, A_9, A_{10} est correctement évaluée, la confiance dans les formules soutenues par les agents A_7, A_8, A_9, A_{10} sera faible. Inversement, la fiabilité des agents A_1, A_2, A_3 soutenant la vérité devrait être élevée et aider à trouver collectivement la vérité. L'utilisation de (l'évaluation estimée de la) confiance des formules peut aider à trouver la vérité plus efficacement qu'en se basant uniquement sur le nombre de votes que chaque question reçoit, comme le font les méthodes qui satisfont la Préservation de la Majorité.

1. Cette distinction entre *agrégation de jugements* et *agrégation de préférences* est faite au début de l'article original [21].

Dans cet article, nous définissons des méthodes qui permettent de trouver correctement la vérité sur ce genre d'exemple. Nous montrons qu'elles satisfont les propriétés logiques attendues pour les méthodes d'agrégation de jugements, et qu'elles sont plus performantes que les méthodes de la littérature pour la recherche de la vérité.

Un résultat surprenant et important de ce travail est que la propriété de Préservation de la Majorité, qui est généralement considérée comme une propriété très attrayante du point de vue de la rationalité, va à l'encontre de l'efficacité de la recherche de la vérité, comme illustré dans l'exemple ci-dessus et aussi dans nos expérimentations. La recherche de la vérité devrait être l'objectif principal des méthodes qui agrègent des croyances, ce qui signifie que les méthodes qui satisfont la Préservation de la Majorité devraient être considérées avec précaution. Plus précisément, une conséquence du théorème du jury de Condorcet (et de ses généralisations) [4, 26, 33, 12, 18] est que si nous disposons d'un nombre suffisant de sources fiables, la majorité trouvera la bonne réponse. Ainsi, dans les applications comportant des milliers de sources, la Préservation de la Majorité n'irait pas à l'encontre de la recherche de la bonne solution. Mais dans les nombreuses applications où nous ne disposons que d'un nombre relativement faible de sources, ces méthodes sont moins efficaces que les méthodes que nous proposons dans ce travail (et ces méthodes seront aussi efficaces que les méthodes satisfaisant la Préservation de la Majorité sur un grand nombre de sources).

Notons que l'étude de la recherche de la vérité avec des règles d'agrégation de jugements a été traité dans [17, 16, 1, 2]. Mais ces travaux portaient sur des structures d'agenda très particulières. [1, 2] effectue une analyse théorique des jeux en supposant un comportement stratégique des agents sur un agenda composé de 2 questions liées. [16] étudie les performances des règles basées sur les prémisses, les conclusions et la fusion sur l'agenda classique de 3 questions du paradoxe doctrinal. [17] étudie les règles basées sur les prémisses et les conclusions. Ces règles sont très particulières et nécessitent des informations supplémentaires pour savoir quelles sont les prémisses et quelles sont les conclusions, informations qui ne sont pas souvent disponibles. Leur principal objectif est d'identifier les bonnes raisons (prémisses) qui sont utilisées pour prendre la bonne décision (conclusions), et elles fournissent un résultat d'optimalité pour une classe particulière de méthodes qui satisfont une condition *d'impartialité*. Dans cet article, nous travaillons sur des agendas généraux (générés aléatoirement), et nos résultats sont donc plus robustes à cet égard. De plus, nous sommes les seuls à considérer que la fiabilité des agents peut être différente et à en tenir compte au cours du processus d'agrégation.

En ce qui concerne le plan de cet article, la section 2 présente quelques préliminaires. Nous rappelons quelques méthodes de la littérature dans la section 3. Nous expliquons comment évaluer la fiabilité des agents et la confiance des formules dans la section 4. Les *méthodes Itératives d'Évaluation de la Confiance* sont définies dans la section 5. Dans la section 6, nous vérifions quelles propriétés sont satisfaites par nos nouvelles méthodes et nous fournissons une étude expérimentale dans la section 7. La section 8 conclut.

2 Préliminaires

Cette section rappelle quelques définitions sur l'agrégation de jugement et introduit des définitions pour l'évaluation de la fiabilité.

Un *agenda* X est un ensemble fini, non vide et totalement ordonné de formules propositionnelles non triviales (i.e. non contradictoires et non tautologiques) $X = \{\varphi_1, \ldots, \varphi_m\}$. Nous désignons par $\underline{X}$ l'agenda étendu qui contient les formules de X et leur négation, i.e. $\underline{X} = \{\varphi_1, \neg\varphi_1, \ldots, \varphi_m, \neg\varphi_m\}$.

Un *jugement* sur une formule φ_k de X est un élément de $D = \{1, 0\}$, où 1 signifie que φ_k est soutenue, 0 signifie que $\neg\varphi_k$ est soutenue.

Un *ensemble de jugements* sur X est une application J de X dans D^m, qui peut également être considéré comme l'ensemble des formules $\{\varphi_k \mid \varphi_k \in X \text{ avec } J(\varphi_k) = 1\} \cup \{\neg\varphi_k \mid \varphi_k \in X \text{ avec } J(\varphi_k) = 0\}$.

Nous pouvons étendre les ensembles de jugements sur X à l'agenda étendu $\underline{X}$ directement par $J(\neg\varphi_k) = \neg J(\varphi_k)$, où $\neg J(\varphi_k) = 1$ si $J(\varphi_k) = 0$, et $\neg J(\varphi_k) = 0$ si $J(\varphi_k) = 1$.

On attend souvent des ensembles de jugements qu'ils soient cohérents et résolus : Un ensemble de jugements J sur X est *cohérent* si $\bigwedge_{\{\varphi_k \in X \mid J(\varphi_k)=1\}} \varphi_k \wedge \bigwedge_{\{\varphi_k \in X \mid J(\varphi_k)=0\}} \neg\varphi_k$ est cohérent. Il est *résolu* ssi $\forall \varphi_k \in X$, $J(\varphi_k) = 0$ ou $J(\varphi_k) = 1$.

Nous désignons par $\mathcal{A} = \{a_1, \ldots, a_n\}$ l'ensemble de tous les agents.

Un profil $P = \langle J_1, \ldots, J_n \rangle$ sur X est un vecteur d'ensembles de jugements sur X où J_i est l'ensemble de jugements de l'agent a_i. P est *cohérent* (resp. *résolu*) lorsque chaque ensemble de jugements qu'il contient est cohérent (resp. *résolu*).

Remarque. Par souci de simplification, nous ne considérerons désormais que les profils résolus et cohérents, i.e. tous les agents doivent donner leur avis (cohérent) sur toutes les formules. Nous désignons par $\mathcal{J}$ l'ensemble des ensembles de jugements cohérents et résolus.

Pour chaque formule φ_k de l'agenda étendu $\underline{X}$, nous définissons $\mathcal{A}_P(\varphi_k) = \{a_i \in \mathcal{A} \mid J_i(\varphi_k) = 1\}$ comme les agents qui soutiennent φ_k dans le profil P.

Pour un agenda X, une *méthode d'agrégation de jugements* R assigne à un profil P sur X un ensemble non vide $R(P)$ d'ensembles de jugements J sur X. Cela signifie que les méthodes d'agrégation de jugements ne doivent pas nécessairement renvoyer un résultat unique, mais tous les résultats admissibles (i.e. tous les ensembles de jugements

ayant la même évaluation). Cela est nécessaire car, dans de nombreux cas, il n'existe aucun moyen de se débarrasser des liens entre certains ensembles de jugements.

3 Méthodes d'Agrégation de Jugements

Dans cette section, nous donnons quelques définitions des méthodes d'agrégation de jugements issues de la littérature. Nous rappelons les définitions et les méthodes de [22, 10].

d_H est la *distance de Hamming* : la distance de Hamming $d_H(J, J')$ entre deux ensembles de jugements J et J' est le nombre de questions sur lesquelles J et J' sont en désaccord, i.e. $d_H(J, J') = |\{\varphi_k \mid J(\varphi_k) \neq J'(\varphi_k)\}|$. La distance de Hamming $d_H(P, P')$ entre deux profils est $d_H(P, P') = \sum_{i=1}^{n} d_H(J_i, J'_i)$.

$N_P(\varphi) = |\mathcal{A}_P(\varphi)|$ est le nombre d'agents dans P dont l'ensemble de jugements comprend φ.

Tout sous-ensemble cohérent de l'agenda peut être étendu afin d'obtenir un ensemble de jugement résolu. Pour tout $S \subseteq X$ cohérent, l'ensemble des extensions rationnelles de S est $ext(S) = \{J \mid J \in \mathcal{J} \text{ et } S \subseteq J\}$.

L'ensemble de jugements majoritaires associé à un profil P contient tous les éléments de l'agenda qui sont soutenus par une majorité d'ensembles de jugements dans P : $m(P) = \{\varphi \in X \mid N_P(\varphi) > \frac{n}{2}\}$. Un profil P est majoritairement cohérent si $m(P)$ est cohérent.

Étant donné un ensemble de formule S et une formule φ, $S' \subseteq S$ est dit φ-cohérent si $S' \cup \{\varphi\}$ est cohérent. S' est un sous-ensemble maximal φ-cohérent de S si S' est φ-cohérent et s'il n'existe pas de $S'' \subseteq S$ qui soit φ-cohérent et que $S' \subset S''$. $max(S, \subseteq)$ désigne les sous-ensembles maximaux cohérent de S. L'ensemble $S' \subseteq S$ est un sous-ensemble *maxcard* φ-cohérent de S si S' est φ-cohérent et qu'il n'existe pas d'ensemble φ-cohérent $S'' \subseteq S$ tel que $|S'| < |S''|$. Le sous-ensemble cohérent *maxcard* de S est noté $max(S, |.|)$.

Il existe deux méthodes basées sur des ensembles de jugements cohérents maximaux : La règle utilisant les *sous-ensembles d'agenda maximaux* R_{MSA} et la règle utilisant les *sous-ensembles d'agenda maxcard* R_{MCSA}. Elles sont définies dans [22] comme suit :

Définition 1. *Pour tout agenda X, pour tout profil P,* $R_{MSA}(P) = \{ext(S) \mid S \in \max(m(P), \subseteq)\}$ *et* $R_{MCSA}(P) = \{ext(S) \mid S \in \max(m(P), |.|)\}$.

La règle basée sur l'ensemble des jugements majoritaires est appelée règle du *sous-agenda à poids maximal* R_{MWA} et est définie comme suit :

Définition 2. $R_{MWA}(P) = \underset{J \in \mathcal{J}}{\operatorname{argmax}} \sum_{\varphi \in J} N_P(\varphi)$.

Une méthode inspirée de la règle de vote "avec rangement de paires" [30], nommé R_{RA}, est définie comme suit :

Définition 3. *Soit* $\geq_P$ *le pré-ordre total sur* $\underline{X}$ *définie par* $\varphi \geq_P \phi$ *si* $N_P(\varphi) \geq N_P(\phi)$. R_{RA} *est défini par la procédure algorithmique suivante :*

réorganiser les éléments de $\underline{X}$ *de telle sorte que* $\varphi_1 \geq_P \ldots \geq_P \varphi_{2m}$
$D := \emptyset$
pour $k = 1, \ldots, 2m$ *faire*
  si $D \cup \{\varphi_k\}$ *est cohérent* ***alors***
    $D \leftarrow D \cup \{\varphi_k\}$
  fin si
fin pour
$R_{RA}(P) := D$.

La règle basée sur la distance $R^{d_H,max}$ est définie comme suit :

Définition 4. $R^{d_H,max}(P) = \underset{J \in \mathcal{J}}{\operatorname{argmin}} \max_{i=1}^{n}(d_H(J_i, J))$

Enfin, nous rappelons la définition des méthodes basées sur le soutien $\delta^{RM\oplus}$ venant de [10].

Un ensemble de jugements $J = \{\varphi_1, \ldots, \varphi_m\}$, avec $\varphi_i \in \underline{X}$, est évalué en termes de soutien de ses éléments. Plus précisément, $s(J)$ est le vecteur de score associé à l'ensemble de jugement J tel que $s(J) = (N_P(\varphi_1), \ldots, N_P(\varphi_m))$.

Une méthode d'agrégation par jugement majoritaire ordonnée $\delta^{RM\oplus}$ (où $\oplus$ peut être $leximax$, $leximin$ ou Σ) est définie comme suit :

Définition 5. *Une méthode d'agrégation par jugement majoritaire ordonnée* $\delta^{RM\oplus}$ *associe à chaque profil P sur l'agenda X l'ensemble de jugements J avec le vecteur de score le plus élevé s par rapport à la fonction d'agrégation* $\oplus$*, i.e.* $\delta^{RM\oplus}(P) = \{J \in \mathcal{J} \mid \nexists J' \in \mathcal{J} t.q. \oplus(s(J')) > \oplus(s(J))\}$.

4 Évaluation de la confiance

La plupart des méthodes de la littérature sur l'agrégation de jugements utilisent le nombre de votes reçus par chaque formule pour établir un jugement collectif et considèrent que tous les agents ont la même fiabilité.

Ici, nous considérons que les agents ont des fiabilités différentes et que ces fiabilités sont inconnues. Cette hypothèse semble plus réaliste dans de nombreux contextes.

Pour avoir un meilleur jugement, nous voulons utiliser une méthode qui calcule la fiabilité des agents, représentée par $r(a_i) \in [0, 1]$ et la confiance de la formule $\varphi_k \in \underline{X}$, représentée par $c(\varphi_k) \in \mathbb{R}^+$. Nous utilisons une méthode itérative pour évaluer la fiabilité des agents. Au lieu de voter simplement pour la formule (ou sa négation) que l'agent estime correcte, nous mettons à jour le poids (i.e. la fiabilité) des agents à chaque itération et calculons la confiance des formules et des négations.

L'évaluation de la fiabilité est basée sur l'algorithme de découverte de la vérité proposé dans [7] (d'autres algo-

rithmes de découverte de la vérité peuvent être trouvés dans [15, 32, 31]).

Dans ces travaux, les méthodes de découverte de la vérité considèrent des faits, objets et sources. Les objets sont plus ou moins des questions ; les faits sont les réponses possibles à ces questions (liées à une seule question) ; et les sources sont des agents qui donnent la réponse qu'ils choisissent aux questions (au plus une par objet). Dans notre cas, les formules sont les objets et les faits sont la valeur de vérité attribuée à chaque formule (0 si l'agent soutient la négation de la formule, i.e. $\neg\varphi$ et 1 si l'agent soutient la formule, i.e. φ). Un agent donne une évaluation pour toutes les formules de l'agenda, le profil est donc résolu.

Nous normalisons la fiabilité d'un agent par le nombre de formules dans l'agenda X car les profils sont résolus. L'étape de normalisation nous permet de considérer le poids associé à chaque agent comme une probabilité, qui représente la confiance empirique que nous pouvons avoir dans cet agent. Plus la fiabilité est proche de 1, plus nous considérons l'agent comme fiable.

La méthode originale est simplifiée dans le cas de l'agrégation de jugements car il n'y a que deux faits par objet (les deux valeurs possibles pour une formule) et le résultat des deux normalisations proposées dans l'article original est le même avec les profils résolus.

Nous décrivons maintenant la procédure d'évaluation de la fiabilité des agents et de la confiance dans les formules. Nous initialisons la fiabilité des agents à 1 car nous n'avons aucune information *a priori* sur les agents et nous considérons donc qu'ils sont tous *a priori* aussi fiables les uns que les autres.

Une itération se déroule comme suit : Nous commençons par évaluer la confiance des formules. Pour chaque formule, nous additionnons la fiabilité des agents qui soutiennent cette formule :

$$c(\varphi_k) = \sum_{a_i \in \mathcal{A}_P(\varphi_k)} r(a_i)$$

Ensuite, pour chaque formule, nous comparons la confiance de la formule positive $c(\varphi_k)$ et la confiance de sa négation $c(\neg\varphi_k)$. Celle qui a la plus grande confiance donne un score de 1 aux agents qui contiennent cette formule dans leur ensemble de jugement individuel et de 0 pour la formule qui a le moins de confiance. En cas d'égalité, la formule et sa négation donnent un score de 0,5 aux agents :

$$V(\varphi_k) = \begin{cases} 1 & si\ c(\varphi_k) > c(\neg\varphi_k) \\ 0 & si\ c(\varphi_k) < c(\neg\varphi_k) \\ 0.5 & si\ c(\varphi_k) = c(\neg\varphi_k) \end{cases}$$

Ensuite, pour calculer la fiabilité d'un agent, nous agrégeons les scores $V(\varphi_k)$ et nous normalisons la fiabilité de chaque agent par le nombre de formules m pour nous assurer que nous avons une fiabilité entre 0 et 1 :

$$r(a_i) = \frac{\sum_{\{\varphi_k | J_i(\varphi_k)=1\}} V(\varphi_k)}{|X|}$$

Algorithme 1 Évaluation de la confiance

Entrée : Le profil P et l'agenda X.
Sortie : La fiabilité r des agents et la confiance c des formules.

1: $r(a_i) = 1\ \forall a_i \in \mathcal{A}$ #fiabilité initiales des agents
2: $c(\varphi_k) = 0\ \forall \varphi_k \in \underline{X}$ #confiance initiale des formules
3: #ta (resp. ta^{-1}) est le vecteur de fiabilité des agents pour l'itération courante (resp. précédente), i.e. $ta = \langle r(a_i) : \forall a_i \in \mathcal{A} \rangle$.
4: **tant que** distance_euclidean$(ta, ta^{-1}) > 0.001$ et
5: nombre_d'itérations < 30 **faire**
6: # Évaluation de la fiabilité des formules
7: **pour chaque** $\varphi_k \in \underline{X}$ **faire**
8: $c(\varphi_k) = \sum_{a_i \in \mathcal{A}_P(\varphi_k)} r(a_i)$
9: **fin pour**
10: # Évaluation du score V pour chaque formule
11: **pour chaque** $\varphi_k \in \underline{X}$ **faire**
12: **si** $c(\varphi_k) > c(\neg\varphi_k)$ **alors**
13: $V(\varphi_k) = 1$
14: **sinon si** $c(\varphi_k) < c(\neg\varphi_k)$ **alors**
15: $V(\varphi_k) = 0$
16: **sinon si** $c(\varphi_k) = c(\neg\varphi_k)$ **alors**
17: $V(\varphi_k) = 0.5$
18: **fin si**
19: **fin pour**
20: **pour chaque** $a_i \in \mathcal{A}$ **faire**
21: # Évaluation de la fiabilité des agents
22: $r(a_i) = \frac{\sum_{\{\varphi_k | J_i(\varphi_k)=1\}} V(\varphi_k)}{|X|}$
23: **fin pour**
24: **fin tant que**

L'algorithme (voir Algorithme 1) s'arrête après 30 itérations ou lorsque le processus converge, i.e. lorsque la distance euclidienne entre la fiabilité des agents de l'itération précédente et l'itération actuelle est inférieure à ϵ avec $\epsilon = 0.001$ [2]. Lorsque l'algorithme s'arrête, la confiance des formules devient le support des formules (la somme de la fiabilité des agents qui soutiennent les formules).

Une idée naturelle est de considérer l'ensemble de jugements obtenu en sélectionnant une formule ou sa négation en fonction de la meilleure confiance à la fin du calcul (ou les ensembles de jugements en cas d'égalité). Cet ensemble est représenté par l'ensemble $H(P)$:

Définition 6. *Nous désignons par $H(P)$ l'ensemble des ensembles de jugements cohérents dont les élé-*

2. Dans les expériences de la section 7, le nombre d'itérations n'excède jamais 7.

ments ont une confiance plus élevée que leur négation : $H(P) = \{\langle\varphi_1, \ldots, \varphi_m\rangle \mid \bigwedge \varphi_i \nvdash \bot$ *et* $\forall k, c(\varphi_k) \geq c(\neg\varphi_k)$*, avec* $\varphi \in \underline{X}\}$.

Notez que la définition de $H(P)$ est faite après le calcul de la confiance de l'algorithme 1, et que $H(P)$ peut être vide si jamais la conjonction des formules n'est pas cohérente. Si $H(P)$ n'est pas vide, alors $H(P)$ sera le résultat de la méthode d'agrégation utilisant la confiance. Si $H(P)$ est vide, nous choisirons le meilleur ensemble de jugements possible, en fonction de la fiabilité des formules qu'il contient. La signification de « meilleur » conduit à des choix différents, selon des stratégies différentes. Ces stratégies seront expliquées dans la section suivante.

5 Méthodes Itératives d'Évaluation de la Confiance

Nous proposons une nouvelle famille de méthodes d'agrégation de jugements basées sur l'évaluation de la fiabilité des agents et de la confiance des formules, appelées méthodes *ICE* pour Méthodes Itératives d'Évaluation de la Confiance (*Iterated Confidence Evaluation Methods* en anglais).

Dans notre contexte, il n'existe qu'une seule *vraie* solution . Cette solution fait partie de l'ensemble de jugements $\mathcal{J}$. Les agents ne savent pas quelle solution est la vérité et l'objectif des méthodes ICE est de la trouver. A cette fin, nous définissons un vecteur de confiances (calculé avec la méthode détaillée dans la section 4) pour tous les ensembles de jugements.

Définition 7. *Étant donné l'agenda étendu* $\underline{X}$ *et un profil* P *sur* $\underline{X}$*, nous désignons par* $s(J) = \langle c(\varphi_k) \mid \varphi_k \in J\rangle$ *le vecteur de fiabilité pour un ensemble de jugements* $J \in \mathcal{J}$.

Une méthode d'agrégation de jugements utilisant la confiance avec la fonction d'agrégation $\oplus$ est définie comme suit :

Définition 8. *Étant donné un profil* P *sur un agenda* X*. Soit* $R^{\oplus}(P)$ *le(s) ensemble(s) de jugements dans* $\mathcal{J}$ *qui maximise(nt) la fiabilité des formules :*

$$R^{\oplus}(P) = \{J_1 \in \mathcal{J} \mid \oplus s(J_1) \geq \oplus s(J_2)\ \forall J_2 \in \mathcal{J}\}$$

Il est important de souligner que si $H(P)$ n'est pas vide, quelle que soit la fonction d'agrégation $\oplus$ considérée, les méthodes ICE sélectionneront $H(P)$ (voir la propriété de cohérence de la fiabilité dans la section 6).

Pour définir les méthodes ICE, nous utilisons $\Sigma, \times$ et *leximax* comme fonctions d'agrégation.

R^{Σ} maximise la somme de la fiabilité de chaque formule, ce qui est lié à une maximisation de la fiabilité globale. $R^{\times}$ maximise le produit de la fiabilité des formules. Le produit est un moyen naturel de calculer la probabilité d'un ensemble d'événements. R^{lex} consiste à trier, dans un ordre décroissant, la fiabilité des formules et à choisir la meilleure (maximum) pour l'ordre lexicographique (*leximax*). C'est une façon de maximiser la meilleure formule (du point de vue de la fiabilité) dans les ensembles de jugements sélectionnés.

Notons que le *leximin* (qui maximise la pire formule) n'est pas utilisé ici, car il a été prouvé dans [10] que le résultat de *leximax* et de *leximin* est le même pour un profil résolu. Nous écrivons *lex* au lieu de *leximax* et *leximin*.

Illustrons le comportement de nos méthodes sur l'exemple de l'introduction, et montrons que, contrairement aux autres méthodes de la littérature, elles parviennent à trouver la vérité.

Exemple 1. *Considérons le profil* P *de la Table 1. Nous évaluons tout d'abord la fiabilité. Nous pouvons voir l'évolution à chaque itération de la fiabilité des agents dans la Table 2 et l'évolution de la confiance des formules dans la Table 3. L'algorithme s'arrête après 4 itérations dans cet exemple. On constate que notre algorithme évalue correctement la fiabilité des agents. Les agents qui affirment la vérité* (A_1, A_2, A_3) *ont la plus grande fiabilité alors que les agents qui se trompent complètement* (A_7, A_8, A_9, A_{10}) *ont la plus faible confiance et n'influencent donc pas négativement la décision.*

	A_1, A_2, A_3	A_4	A_5	A_6	A_7, A_8, A_9, A_{10}	A_{11}
It1	1	1	1	1	1	1
It2	0.6	0.6	0.4	0.8	0.4	0.8
It3	0.7	0.5	0.5	0.9	0.3	0.7
It4	1	0.6	0.8	0.8	0	0.4

Table 2 – Fiabilité des agents

	φ_1	$\neg\varphi_1$	φ_2	$\neg\varphi_2$	φ_3	$\neg\varphi_3$	φ_4	$\neg\varphi_4$	φ_5	$\neg\varphi_5$
It1	6	5	5	6	6	5	5	6	6	5
It2	3.6	2.4	3	3	3.8	2.2	2.8	3.2	4	2
It3	4	1.9	3.5	2.4	4.2	1.7	3.1	2.8	4.2	1.7
It4	5.2	0.4	4.6	1	5	.6	4.4	1.2	4.8	0.8

Table 3 – Confiance des formules

Nous pouvons maintenant utiliser les méthodes ICE. La solution proposée est : $R^{\Sigma}(P) = R^{\times}(P) = R^{lex}(P) = \{\varphi_1, \varphi_2, \varphi_3, \varphi_4, \varphi_5\}$ *ce qui correspond à la vérité. Pour cet ensemble de jugements, le vecteur de fiabilité est* $s(\{\varphi_1, \varphi_2, \varphi_3, \varphi_4, \varphi_5\}) = \langle 5.2, 4.6, 5, 4.4, 4.8\rangle$ *Le score de* $R^{\Sigma}(P)$ *est 24, le score de* $R^{\times}(P)$ *est 2525.95 et pour* $R^{lex}(P)$ *dans l'ordre nous avons :* φ_1*,* φ_3*,* φ_5*,* φ_2 *puis* φ_4*.*

La plupart des méthodes de la littérature ($R_{RA}, R_{MWA}, R_{MCSA}, R_{MSA}$, $\delta^{RM_{leximax}}$ *et* $\delta^{RM_{\Sigma}}$) *satisfont la Préservation de la Majorité, alors ils donneront l'ensemble de jugements majoritaires comme solution (i.e.* $\{\varphi_1, \neg\varphi_2, \varphi_3, \neg\varphi_4, \varphi_5\}$*), ce qui est faux.* $R^{d_H,max}$ *ne satisfait pas à la Préservation de la Majorité et donne comme solutions* $\{\{\varphi_1, \neg\varphi_2, \varphi_3, \varphi_4, \neg\varphi_5\}, \{\varphi_1, \neg\varphi_2, \varphi_3, \neg\varphi_4, \varphi_5\}, \{\varphi_1, \neg\varphi_2, \varphi_3, \neg\varphi_4, \neg\varphi_5\}, \{\varphi_1, \neg\varphi_2, \neg\varphi_3, \varphi_4, \varphi_5\}\}$*, qui sont quatre solutions erronés.*

6 Propriétés

Dans cette section, nous étudions les propriétés théoriques des méthodes ICE $R^{\oplus}$.

La première propriété est *Universalité* et stipule que tous les profils cohérents doivent être une solution possible.

Universalité (Uni.). Le domaine de R est l'ensemble de tous les profils cohérents.

Proposition 1. *Les méthodes $R^{\oplus}$ satisfont **Universalité**.*

Démonstration. Les solutions de R sont choisis parmi les ensembles de jugements $\mathcal{J}$. Tous les ensembles de jugements de $\mathcal{J}$ sont cohérents. □

La propriété suivante est *Anonymat*. Nous voulons que les méthodes d'agrégation de jugements soient impartiales et ne favorisent aucun agent en particulier.

Anonymat (Ano.). Pour deux profils $P = (J_1, \ldots, J_n)$ et $P' = (J'_1, \ldots, J'_n)$ dans le domaine de R qui sont des permutations l'un de l'autre, nous avons $R(P) = R(P')$.

Proposition 2. *Les méthodes $R^{\oplus}$ satisfont **Anonymat**.*

Démonstration. La fiabilité d'un agent est évaluée en fonction de ses affirmations et n'est pas liée à l'ordre donné dans le profil. Dans notre algorithme (Algorithme 1), la fiabilité des agents sera la même en prenant le profil P ou le profil P' avec la permutation, alors les solutions données par les méthodes ICE sont les mêmes : $R(P) = R(P')$. □

Pour la propriété suivante, nous devons rappeler la *règle d'agrégation majoritaire* R^{maj} qui est définie comme suit : Pour tout agenda X et tout profil P sur X, nous avons $R^{maj}(P) = \{R_P^{maj}\}$, où pour toute question $\varphi_k \in X$:

$$R_P^{maj}(\varphi_k) = \begin{cases} 1 & \text{si } N_P(\varphi_k) > \frac{|\mathcal{A}|}{2} \\ 0 & \text{si } N_P(\neg\varphi_k) > \frac{|\mathcal{A}|}{2} \\ \star & \text{sinon} \end{cases}$$

La propriété suivante est la *Préservation de la Majorité*. Elle stipule que le résultat du vote majoritaire doit être la solution proposée par la méthode si le résultat du vote majoritaire est cohérent et résolu.

Préservation de la Majorité (Maj.). Si R_P^{maj} est cohérent et résolu alors $R(P) = \{R^{maj}(P)\}$

Proposition 3. *Les méthodes $R^{\oplus}$ ne satisfont pas **Préservation de la Majorité**.*

Démonstration. Considérons le profil de la Table 1, l'évaluation de la fiabilité des agents de la Table 2 et la confiance des formules de la Table 3.

Dans cet exemple, l'ensemble de jugements majoritaires cohérents est $(\{\varphi_1, \neg\varphi_2, \neg\varphi_3, \neg\varphi_4, \neg\varphi_5\})$, mais les méthodes ICE donnent $\{\varphi_1, \varphi_2, \varphi_3, \varphi_4, \varphi_5\}$ comme solution. Le score de $\{\varphi_1, \varphi_2, \varphi_3, \varphi_4, \varphi_5\}$ maximise le score pour R^{Σ} et $R^{\times}$. R^{lex} ordonne les vecteurs de confiance pour obtenir le maximum $\{5.2, 5, 4.8, 4.6, 4.4\}$ (dans un ordre décroissant) et sélectionne ensuite $\{\varphi_1, \varphi_2, \varphi_3, \varphi_4, \varphi_5\}$.

Soulignons que dans ce cas, une méthode qui satisfait Préservation de la Majorité ne trouvera pas la vérité, contrairement à la nôtre. □

Si nous considérons l'exemple de la Table 1, nous voyons que suivre les opinions de la majorité des agents peut conduire à une mauvaise solution. Cet exemple montre donc que cette propriété n'est pas adéquate dans ce cadre de fiabilité à variable inconnue, où nous ne voulons pas nous concentrer sur le plus grand nombre de votes, mais sur la plus grande confiance (le plus grand nombre de votes provenant de sources fiables). Par conséquent, nous ne voulons pas que nos méthodes ICE satisfassent cette propriété.

Les méthodes ICE ne satisfont pas *Préservation de la Majorité*, mais lorsque $H(P)$ est cohérent (l'ensemble des ensembles de jugements cohérents dont les éléments ont une confiance plus élevée que leur négation), le résultat des méthodes ICE sera les formules de $H(P)$. Nous définissons une nouvelle propriété appelée *Cohérence de la fiabilité*. Cette propriété garantit qu'une méthode sélectionne les résultats cohérents les plus fiables, si possible.

Cohérence de la fiabilité (Coh.). Si $H(P)$ n'est pas vide, alors la solution de R est $H(P)$, i.e. $R(P) = H(P)$.

Proposition 4. *Les méthodes $R^{\oplus}$ satisfont **Cohérence de la fiabilité**.*

Démonstration. Supposons que $H(P)$ n'est pas vide.

Soit $J^R \in H(P)$. Par définition de $H(P)$, $\forall\varphi_k \in J^R, c(\varphi_k) \geq c(\neg\varphi_k)$. Supposons que $R^{\oplus}$ ne sélectionne pas J^R. Cela signifie que $\exists J \notin J^R$ t.q. $\oplus_{\varphi_k \in J}(c(\varphi_k)) > \oplus_{\varphi'_k \in J^R}(c(\varphi'_k))$. Nécessairement, cela implique que pour au moins un k, $c(\varphi_k) > c(\varphi'_k)$, i.e que $c(\neg\varphi'_k) > c(\varphi'_k)$ avec $\varphi'_k \in J^R$: contredit la définition de $H(P)$. Donc $H(P) \subseteq R^{\oplus}(P)$.

Soit $J \in R^{\oplus}(P)$. Supposons que $J \notin H(P)$. Comme $J \notin H(P)$, il y a au moins un k t.q. $c(\varphi_k) < c(\neg\varphi_k)$. Comme $H(P)$ n'est pas vide, il existe au moins un ensemble de jugement $J^R \in H(P)$ t.q. $\neg\varphi_k \in J^R$. Nous savons également que $\forall\varphi_i \in J^R, i \neq k, c(\varphi_i) \geq c(\neg\varphi_i)$. Alors $\oplus_{\varphi_i \in J}(c(\varphi_i)) < \oplus_{\varphi'_i \in J^R}(c(\varphi'_i))$: J ne peut être sélectionné par $R^{\oplus}$. Nous avons aussi une contradiction et $R^{\oplus}(P) \subseteq H(P)$.

Enfin, $R^{\oplus}(P) = H(P)$. □

Unanimité est la propriété suivante. Lorsqu'une formule est présente dans tous les ensembles de jugements individuels, elle doit l'être dans tous les résultats.

Unanimité (Una.). Pour tout $\varphi_k \in X$, si $J_i(\varphi_k) = x$ avec $x \in \{0, 1\}$, $\forall J_i \in P$, alors pour tout $J \in R(P)$, nous avons $J(\varphi_k) = x$.

Proposition 5. *$R^{\times}$ et R^{lex} satisfont **Unanimité**. R^{Σ} ne satisfait pas **Unanimité**.*

Avec R^{Σ}, la combinaison de la fiabilité de la négation de la formule unanime avec la fiabilité des formules qui maximisent leur confiance par rapport à leur négation peut dans certains cas compenser la confiance nulle de la négation de la formule unanime.

Démonstration. Considérons φ_k la formule assignée à chaque ensemble de jugement individuel et $\neg\varphi_k$ la formule qui n'est dans aucun ensemble de jugement.

Nous savons que la confiance de φ_k sera la plus élevée car tous les agents affirment que cette formule est la vérité ($c(\neg\varphi_k) = \sum_{a_i \in \mathcal{A}_P(\varphi_k)} r(a_i)$) et que la confiance en φ_k sera égale à zéro ($c(\neg\varphi_k) = \sum_{a_i \in \emptyset} 0$). De plus, nous savons qu'il y a au moins un ensemble de jugements qui contient φ_k car φ_k est dans chaque ensemble de jugements individuel.

Tous les vecteurs de fiabilité maximale pour lex contiennent φ_k. R^{lex} choisira son résultat dans ces ensembles. Nous savons que φ_k a le niveau de confiance le plus élevé et que φ_k figure au moins dans un ensemble de jugement (cohérent). Donc φ_k sera dans tous les résultats de R^{lex}.

Tous les ensembles de jugement qui contiennent $\neg\varphi_k$ obtiennent une confiance de 0 avec $R^{\times}$. Il existe au moins un ensemble de jugements cohérents, disons J, qui contient φ_k, car φ_k appartient aux ensembles de jugements de tous les agents, qui sont cohérents. Comme il y a unanimité sur φ_k, tous les agents ont une fiabilité strictement supérieure à 0. Alors toutes les formules de J ont une confiance strictement supérieure à 0. En conséquence, tout ensemble de jugement sélectionné par $R^{\times}$ contiendra φ_k.

Donc $R^{\times}$ et R^{lex} satisfont *Unanimité*.

R^{Σ} ne satisfait pas *Unanimité*. Considérons les formules propositionnelles $\varphi_1 = a$, $\varphi_2 = b$, $\varphi_3 = c$, $\varphi_4 = d$, $\varphi_5 = \neg a$, $\varphi_6 = \neg b$, $\varphi_7 = \neg c$, $\varphi_8 = \neg d$, $\varphi_9 = a \vee b$, $\varphi_{10} = \neg(a \wedge b \wedge c \wedge d)$. Nous avons quatre agents dans le profil P, avec les agents qui affirment $\{\varphi_1, \varphi_2, \varphi_3, \neg\varphi_4, \neg\varphi_5, \neg\varphi_6, \neg\varphi_7, \varphi_8, \varphi_9, \varphi_{10}\}$, $\{\varphi_1, \varphi_2, \neg\varphi_3, \varphi_4, \neg\varphi_5, \neg\varphi_6, \varphi_7, \neg\varphi_8, \varphi_9, \varphi_{10}\}$, $\{\varphi_1, \neg\varphi_2, \varphi_3, \varphi_4, \neg\varphi_5, \varphi_6, \neg\varphi_7, \neg\varphi_8, \varphi_9, \varphi_{10}\}$ et $\{\neg\varphi_1, \varphi_2, \varphi_3, \varphi_4, \varphi_5, \neg\varphi_6, \neg\varphi_7, \neg\varphi_8, \varphi_9, \varphi_{10}\}$.

φ_1	$\neg\varphi_1$	φ_2	$\neg\varphi_2$	φ_3	$\neg\varphi_3$	φ_4	$\neg\varphi_4$	φ_5	$\neg\varphi_5$
2.4	0.8	2.4	0.8	2.4	0.8	2.4	0.8	0.8	2.4
φ_6	$\neg\varphi_6$	φ_7	$\neg\varphi_7$	φ_8	$\neg\varphi_8$	φ_9	$\neg\varphi_9$	φ_{10}	$\neg\varphi_{10}$
0.8	2.4	0.8	2.4	0.8	2.4	3.2	0	3.2	0

TABLE 4 – Confiance des formules du profil P

Nous avons la confiance des formules dans la Table 4. Les solutions proposées par R^{Σ} sont $R^{\Sigma} = \{ \{\varphi_1, \varphi_2, \varphi_3, \varphi_4, \neg\varphi_5, \neg\varphi_6, \neg\varphi_7, \neg\varphi_8, \varphi_9, \neg\varphi_{10}\}$, $\{\varphi_1, \varphi_2, \varphi_3, \neg\varphi_4, \neg\varphi_5, \neg\varphi_6, \neg\varphi_7, \varphi_8, \varphi_9, \varphi_{10}\}$, $\{\varphi_1, \varphi_2, \neg\varphi_3, \varphi_4, \neg\varphi_5, \neg\varphi_6, \varphi_7, \neg\varphi_8, \varphi_9, \varphi_{10}\}$, $\{\varphi_1, \neg\varphi_2, \varphi_3, \varphi_4, \neg\varphi_5, \varphi_6, \neg\varphi_7, \neg\varphi_8, \varphi_9, \varphi_{10}\}$, $\{\neg\varphi_1, \varphi_2, \varphi_3, \varphi_4, \varphi_5, \neg\varphi_6, \neg\varphi_7, \neg\varphi_8, \varphi_9, \varphi_{10}\}$ $\}$

Les quatre solutions ont un score de 22,4 avec Σ. Les agents sont unanimes sur φ_{10} mais dans l'une des solutions de R^{Σ} nous avons $\neg\varphi_{10}$. Donc la méthode R^{Σ} ne satisfait pas *Unanimité*. □

Maintenant nous parlons de *Systématicité*, une propriété critiquée dans la littérature [3, 8, 10, 23]. Elle stipule que toutes les formules ayant le même nombre de voix doivent être traitées de la même manière. La principale critique de cette propriété est qu'elle ignore le reste du profil lorsqu'une décision doit être prise pour une formule. C'est déjà un problème dans l'agrégation standard des jugements. Mais c'est encore plus un problème dans notre cadre de fiabilité variable inconnue.

Systématicité (Syst.). Pour deux profils quelconques $P = (J_1, \ldots, J_n)$ et $P' = (J'_1, \ldots, J'_n)$ dans le domaine de R, et deux propositions quelconques φ_k et φ_l de X, tel que $J_i(\varphi_k) = J'_i(\varphi_l) \forall i$, si $J_P(\varphi_k) = x$ pour tous $J_P \in R(P)$, alors $J'_{P'}(\varphi_l) = x$ pour tous $J'_{P'} \in R(P')$.

Proposition 6. *Les méthodes $R^{\oplus}$ ne satisfont pas* ***Systematicity****.*

Cette propriété ne tient pas compte des autres formules. Mais l'avis des agents sur toutes les formules doit être pris en compte pour trouver le bon résultat. C'est ainsi que la fiabilité des agents est estimée.

Démonstration. Considérerons le profil P et P' respectivement de la Table 5 et de la Table 6 avec $\varphi_1 = p, \varphi_2 = p \wedge r, \varphi_3 = r \vee s, \varphi_4 = p \wedge q, \varphi_5 = t$. Considérerons $k = 3$ et $l = 4$, les agents expriment donc le même avis sur φ_3 et φ_4. Les agents A_6 et A_7 changent leurs opinions sur φ_1 et l'agent A_{10} sur φ_5.

Nous avons la confiance des formules dans la Table 7 pour le profil P et dans la Table 8 pour le profil P'.

	φ_1	φ_2	φ_3	φ_4	φ_5
A_1	1	1	1	1	1
A_2	1	1	1	1	1
A_3	1	1	1	1	0
A_4	1	0	0	1	1
A_5	1	0	0	1	0
A_6	0	0	1	0	0
A_7	0	0	1	0	1
A_8	0	0	0	0	0
A_9	0	0	0	0	0
A_{10}	0	0	0	0	0

TABLE 5 – Profile P

	φ_1	φ_2	φ_3	φ_4	φ_5
A_1	1	1	1	1	1
A_2	1	1	1	1	1
A_3	1	1	1	1	0
A_4	1	0	0	1	1
A_5	1	0	0	1	0
A_6	1	0	1	0	0
A_7	1	0	1	0	1
A_8	0	0	0	0	0
A_9	0	0	0	0	0
A_{10}	0	0	0	0	1

TABLE 6 – Profile P'

φ_1	$\neg\varphi_1$	φ_2	$\neg\varphi_2$	φ_3	$\neg\varphi_3$	φ_4	$\neg\varphi_4$	φ_5	$\neg\varphi_5$
1.2	4.4	0.2	5.4	1.6	4	1.2	4.4	1	4.6

TABLE 7 – Confiance des formules du profil P

φ_1	$\neg\varphi_1$	φ_2	$\neg\varphi_2$	φ_3	$\neg\varphi_3$	φ_4	$\neg\varphi_4$	φ_5	$\neg\varphi_5$
5	0.8	2.2	3.6	3.6	2.2	3.6	2.2	3.6	2.2

TABLE 8 – Confiance des formules du profil P'

Avec le profil P, la solution de nos méthodes avec Σ, $\times$ et lex est la même qu'avec le profil P et P' et est $R^{\oplus}(P) =$

$\{\neg\varphi_1, \neg\varphi_2, \neg\varphi_3, \neg\varphi_4, \neg\varphi_5\}$. Avec le profil P', nous avons $R^{\Sigma}(P') = \{\varphi_1, \neg\varphi_2, \varphi_3, \varphi_4, \varphi_5\}$, qui est différent sur φ_3 et φ_4 malgré le fait que les opinions des agents n'ont pas changé. Nous avons $R(P) \neq R(P')$ même si $\forall i, J_i(\varphi_k) = J'_i(\varphi_l)$.

□

La propriété suivante est *Neutralité*. Les éléments de l'agenda doivent être considérés de la même manière.

Neutralité (Neut.). Si $X = \{\varphi_1, \ldots, \varphi_m\}$ et $X' = \{\varphi'_1, \ldots, \varphi'_m\}$ sont deux agendas tels qu'il existe une permutation σ sur $\{1, \ldots, m\}$ satisfaisant $\varphi_k \equiv \varphi'_{\sigma(k)}$ pour tous $k \in \{1, \ldots, m\}$ alors pour tout profil $P = (J_1, \ldots, J_n)$ sur X et $P' = (J'_1, \ldots, J'_n)$ sur X' tel que pour chaque $i \in \{1, \ldots, n\}$, pour tous $k \in \{1, \ldots, m\}$, $J_i(\varphi_k) = J'_i(\varphi'_{\sigma(k)})$, nous avons $R(P) = R(P')$.

Proposition 7. *Les méthodes $R^{\oplus}$ satisfont **Neutralité**.*

Démonstration. Une permutation sur X ne changera pas les ensembles de jugements dans P de sorte que la confiance des formules sera la même avec P' parce que la confiance n'est pas évaluée en fonction de l'ordre des formules dans l'agenda. Une permutation sur X ne changera pas les résultats de R : $R(P) = R(P')$.

Nos méthodes satisfont à la règle *Neutralité*. □

Les résultats sont résumé dans la Table 9.

	Uni.	Ano.	Maj.	Coh.	Una.	Syst.	Neut.
R^{Σ}	✓	✓	✗	✓	✗	✗	✓
$R^{\times}$	✓	✓	✗	✓	✓	✗	✓
R^{lex}	✓	✓	✗	✓	✓	✗	✓

TABLE 9 – Propriétés satisfaites par les méthodes ICE

Les méthodes $R^{\oplus}$ satisfont donc les propriétés importantes et standard tel que Universalité, Anonymat, Neutralité et Unanimité, à l'exception de R^{Σ} qui ne satisfait pas à Unanimité. Cette méthode est donc peut-être moins intéressante que les autres pour cette raison. Aucune de ces méthodes ne satisfait Systématicité, qui est une propriété très critiquée. Elles ne satisfont pas à la propriété de Préservation de la Majorité, qui est généralement considérée comme une propriété souhaitable, mais nous avons montré que dans ce cadre, suivre la majorité des votes n'est pas toujours approprié. Nous développerons davantage ce point dans la section suivante, où nous montrerons les meilleures performances de nos méthodes en matière de recherche de la vérité par rapport à celles qui satisfont la Préservation de la Majorité.

7 Expérimentations

Nous avons procédé à une évaluation expérimentale de nos méthodes ICE afin de tester leurs performances en matière de recherche de vérité, i.e. de savoir si elles trouvent correctement la vérité pour chaque formule. [3]

À notre connaissance, il n'existe pas de jeux de données disponibles pour la recherche de la vérité dans l'agrégation de jugements. L'utilisation de jeux de données tels que ceux venant de *Preflib* [4] n'est pas possible pour plusieurs raisons. Tout d'abord, ces jeux de données concernent les préférences et sont utiles pour tester les règles de vote. La deuxième raison est qu'il n'y a pas de vérité dans Preflib, car il s'agit de préférences et non de jugements épistémiques (l'objectif de nos méthodes est de trouver la vérité). Même si nous pouvons traduire les profils de préférences en profils d'agrégation de jugements, nous ne pouvons pas utiliser ces jeux de données pour évaluer la fiabilité moyenne des profils, ce qui est un paramètre important dans nos expériences. Pour ces raisons, nous générons des agendas et des profils aléatoires afin de tester les performances des méthodes.

Pour tester toutes les méthodes, nous générons l'agenda, composé d'un ensemble de formules propositionnelles. Dans cet article, nous considérons un langage basé sur 10 propositions atomiques et chaque agenda est composé de $m = 10$ formules propositionnelles.

Pour créer aléatoirement un agenda X, nous générons chaque formule une par une. Pour générer une formule propositionnelle φ de X, nous sélectionnons aléatoirement une distance d et une interprétation ω. Pour chaque interprétation ω', nous disons que ω' est un modèle de φ si la distance de Hamming entre ω et ω' est inférieure à d : $d_H(\omega, \omega') \leq d$.

Une fois l'agenda fixé, nous pouvons choisir la vérité et générer le profil. Pour s'assurer que l'ensemble des jugements d'un agent est résolu et cohérent, nous choisissons au hasard l'un des ensembles de jugements dans $\mathcal{J}$ pour être son ensemble de jugements. La vérité, noté T, est également choisie au hasard dans $\mathcal{J}$.

Avec l'ensemble des jugements des agents, nous pouvons calculer la fiabilité moyenne des agents, i.e. la probabilité *a posteriori* de trouver les valeurs correctes pour toutes les formules en comparant l'ensemble des jugements de tous les agents et la vérité T. La fiabilité moyenne d'un agent $J_i \in P$ est $\frac{|T \cap J_i|}{|T|}$.

Chaque méthode produit un ensemble de solutions, et différentes mesures peuvent être utilisées pour estimer à quel point cet ensemble de solutions est proche de la vérité. Nous avons choisi celle qui est directement liée à la prise de décision, i.e. le nombre de décisions qui peuvent être prises à l'unanimité sur chaque formule du profil. Ainsi, toutes les solutions doivent avoir la même valeur pour une formule pour qu'il y ait une décision unanime sur cette formule. Ensuite, nous évaluons ces décisions par rapport à la vérité T choisie précédemment.

3. Le code et les agendas utilisés dans les expériences sont disponibles sur `https://github.com/QuentinElsaesser/JA`.

4. `www.preflib.org`

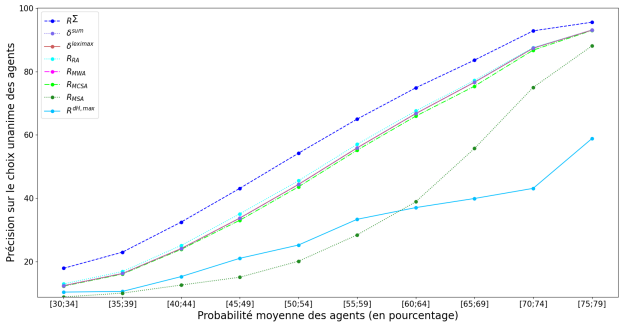


FIGURE 1 – Précision - 10 agents

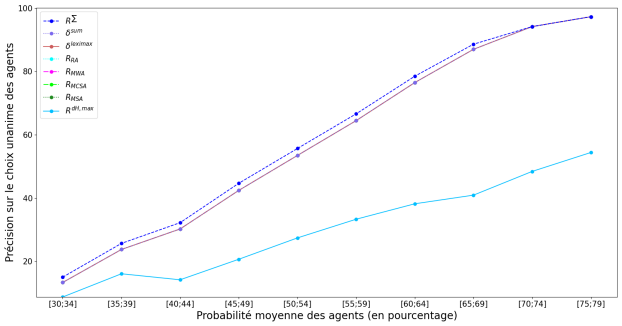


FIGURE 2 – Précision - 11 agents - Ensemble de jugements majoritaires

Définition 9. *Nous désignons par $U(P)$ l'ensemble des formules qui se trouvent dans tous les ensemble de jugement de $R(P)$: $U(P) = \{\varphi_k \in \underline{X} \mid \varphi_k \in \bigcap R(P)\}$*

Pour évaluer la performance, nous regardons la proportion de valeurs correctement prédites par la méthode et nous l'appelons *Précision* i.e. $Precision = \frac{|\{\varphi_k \in U(P) | \varphi_k \in T\}|}{|X|}$.

Chaque point sur les graphiques est la moyenne obtenue avec la génération de 1000 expériences. Nous classons les expériences en fonction de la fiabilité moyenne des agents.

Nous comparons les méthodes ICE avec les méthodes R_{MSA}, R_{MCSA}, R_{MWA}, R_{RA}, $R^{d_H,max}$ dans [22] et les méthodes de [10] $\delta^{RM_{leximax}}$ et δ^{RM_Σ}. Nous ne présentons que les résultats de R^Σ (et non ceux de $R^\times$ et R^{lex}) car les solutions proposées par les trois méthodes ICE sont identiques à plus de 95% dans nos expérimentations.

Pour les expériences, nous nous concentrons sur deux paramètres qui ont un impact sur les résultats des méthodes. Le premier est la cohérence ou l'incohérence de l'ensemble de jugements majoritaires. S'il est cohérent, la plupart des méthodes de la littérature donneront l'opinion majoritaire comme résultat. Mais que se passe-t-il lorsque l'ensemble des jugements majoritaires n'est pas cohérent pour les méthodes de la littérature et comment les méthodes ICE traitent-elles ces deux cas différents ? Le deuxième paramètre est le nombre d'agents, plus précisément si le nombre d'agents est pair ou impair. Dans la littérature, le nombre d'agents est souvent impair car cela garantit une majorité d'opinions sur toutes les formules, alors que dans l'autre cas, il peut y avoir des égalités (i.e. le même soutien entre la formule et sa négation), ce qui rend les méthodes de la littérature moins décisives.

Sur la Figure 1, nous avons les résultats de la *Précision* avec 10 agents sous une fiabilité moyenne variable. Nous avons entre 10 et 50% de profils pour lesquels l'ensemble des jugements majoritaires est cohérent. Ce cas est le moins optimal pour les méthodes de la littérature. Les méthodes ICE surpassent les méthodes de la littérature de plus de 5% et lorsque la fiabilité moyenne est supérieure à 75%, les méthodes ICE sont 3% plus efficaces et sont proches de la vérité, i.e. que la *Précision* est proche de 100%. La raison pour laquelle les méthodes de ICE sont plus performantes que les méthodes de la littérature est qu'elles ne proposent en moyenne qu'une seule solution (entre 1,01 et 1,055 en fonction de la fiabilité moyenne des agents dans nos expériences) alors que les autres méthodes proposent plusieurs solutions, entre 2 et 3 et pour $R^{d_H,max}$ plus de 8. Les méthodes ICE sont donc plus décisives que les méthodes de la littérature lorsque la plupart des ensembles de jugements majoritaires sont incohérents. L'une des raisons de ce manque de décision est que les méthodes de la littérature ne savent pas comment gérer correctement les égalités entre une formule et sa négation, et donnent deux solutions possibles, l'une contenant la formule et l'autre sa négation (notons que nous n'avons pas d'égalités sur tous les profils dans cette expérience). Leur *Précision* sera donc plus faible car les résultats ne sont pas unanimes pour toutes les formules. Grâce à l'évaluation itérée de la fiabilité, les méthodes ICE permettent une meilleure gestion des égalités et donne de meilleurs résultats.

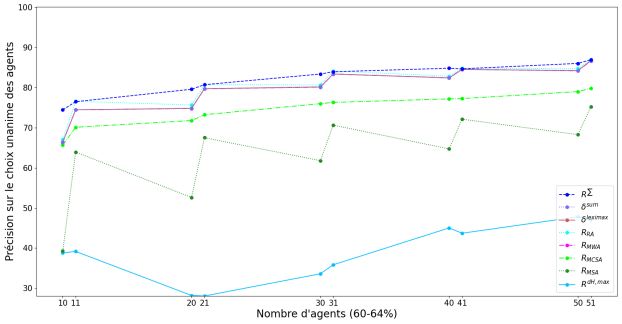


FIGURE 3 – Précision

Dans la Figure 2, le nombre d'agents est impair, de sorte qu'il n'y a plus de cas avec une égalité. De plus, nous ne sélectionnons que les profils ayant un ensemble de jugements majoritaires cohérents pour cette expérience, car il s'agit du cas le plus optimal pour les méthodes de la littérature, ils donnent l'ensemble de jugements majoritaires et sont décisifs. Nous voulons donc voir si le caractère décisif des méthodes ICE est la seule raison des bons résultats de la Figure 1. Et nous voyons que la Précision entre les méthodes ICE et les méthodes de la littérature est plus proche. Pour les

méthodes ICE, il n'y a pas de changement significatif dans les résultats, et les méthodes ICE ont toujours de meilleurs résultats. Dans ces expérimentations, l'opinion d'une majorité d'agents fiables, lorsqu'elle est cohérente, est souvent proche de la vérité.

Dans la Figure 3, nous voyons les résultats dans le cas général et augmentons le nombre d'agents pour les profils avec une fiabilité fixe, entre 60 et 64%. Nous examinons le cas avec un nombre pair et impair d'agents. Dans cette expérience, plus de 76% des profils ont un ensemble de jugements majoritaires cohérents avec un nombre impair d'agents et moins de 50% avec un nombre pair d'agents. Avec un ensemble de jugements majoritaires cohérents, les méthodes ICE sont légèrement plus performantes que les méthodes de la littérature, mais ont plus de 5% avec un ensemble de jugements majoritaires incohérents. Avec un plus grand nombre d'agents, les performances de toutes les méthodes augmentent.

Nous avons observé différents cas. Lorsque l'ensemble des jugements majoritaires est cohérent, ce qui est le cas optimal pour les méthodes d'agrégation de jugements dans la littérature, les méthodes ICE sont légèrement meilleures. Cependant, lorsque l'ensemble de jugements majoritaires n'est pas cohérent, les méthodes ICE sont plus performantes. Elles sont plus décisives que les autres méthodes et peuvent traiter des cas plus complexes tels que des profils où la majorité conduit à des égalités. En moyenne, les méthodes ICE obtiennent donc de meilleurs résultats que les méthodes de la littérature pour la recherche de la vérité avec différents paramètres. L'évaluation de la fiabilité a un réel impact pour prendre la bonne décision lorsque le nombre d'agents est pair et lorsqu'il y a égalité entre le support d'une formule et sa négation.

8 Conclusion

Dans cet article, nous présentons un cadre pour l'agrégation de jugements où la fiabilité de l'agent est variable mais inconnue. Dans ce cadre, il est possible d'estimer cette fiabilité en confrontant les ensembles de jugements des agents. Nous présentons la famille des méthodes ICE, qui utilisent la confiance des formules au lieu du nombre de votes pour trouver le résultat correct. Nous montrons que ces méthodes satisfont les propriétés standard des méthodes d'agrégation de jugements et nous effectuons une évaluation montrant qu'elles sont plus performantes que les méthodes de la littérature pour la tâche de recherche de la vérité. Il est intéressant de noter qu'avec un nombre suffisant d'agents fiables, la majorité donnera la vérité (ce qui peut être considéré comme une conséquence du théorème du Jury de Condorcet), de sorte que les méthodes de la littérature qui satisfont la propriété de la Préservation de la Majorité donneront de bons résultats. Mais nous discutons du fait que la Préservation de la Majorité va également à l'encontre de la recherche de la vérité dans certains cas (les moins décisifs). Et nos méthodes ICE (qui ne satisfont pas la propriété Préservation de la Majorité) sont plus performantes que les autres méthodes lorsque cette majorité n'est pas suffisamment claire. En particulier, nous avons montré que pour un nombre pair d'agents, ou lorsqu'il n'y a pas d'ensemble de jugements majoritaires cohérent, les méthodes ICE sont nettement plus performantes.

Pour les travaux futurs, il serait possible de commencer l'évaluation de la fiabilité des agents à partir d'une fiabilité *a priori* donnée, qui pourrait provenir par exemple de la réputation de chaque agent, ou d'autres types de mesures de confiance, puis d'ajuster ces fiabilités *a priori* à l'aide de nos méthodes.

Remerciements

Ce travail a bénéficié du support de la Chaire IA BE4musIA (ANR-20-CHIA-0028).

Références

[1] Irem Bozbay. Truth-tracking judgment aggregation over interconnected issues. *Social Choice and Welfare*, 53 :337–370, 2019.

[2] Irem Bozbay, Franz Dietrich, and Hans Peters. Judgment aggregation in search for the truth. *Games and Economic Behavior*, 87 :571–590, 2014.

[3] Bruce Chapman. Rational aggregation. *politics, philosophy & economics*, 1(3) :337–354, 2002.

[4] Marquis de Condorcet. *Essai sur l'application de l'analyse à la probabilité des décisions rendues à la pluralité des voix.* Imprimerie royale Paris, 1785.

[5] Franz Dietrich and Christian List. Judgment aggregation by quota rules. *Journal of Theoretical Politics*, 2005.

[6] Franz Dietrich and Philippe Mongin. The premiss-based approach to judgment aggregation. *Journal of Economic Theory*, 145(2) :562–582, 2010.

[7] Quentin Elsaesser, Patricia Everaere, and Sébastien Konieczny. Voting-based methods for evaluating sources and facts reliability. In *2023 IEEE 35th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 178–185, 2023.

[8] Ulle Endriss, Umberto Grandi, and Daniele Porello. Complexity of judgment aggregation. *Journal of Artificial Intelligence Research*, 45 :481–514, 2012.

[9] Patricia Everaere, Chouaib Fellah, Sébastien Konieczny, and Ramón Pino Pérez. Weighted merging of propositional belief bases. In *Proceedings of the*

20th International Conference on Principles of Knowledge Representation and Reasoning, KR'23, Rhodes, Greece, September 2-8, 2023, pages 219–228, 2023.

[10] Patricia Everaere, Sébastien Konieczny, and Pierre Marquis. Counting votes for aggregating judgments. In Ana L. C. Bazzan, Michael N. Huhns, Alessio Lomuscio, and Paul Scerri, editors, *International conference on Autonomous Agents and Multi-Agent Systems, AAMAS '14, Paris, France, May 5-9, 2014*, pages 1177–1184. IFAAMAS/ACM, 2014.

[11] Patricia Everaere, Sébastien Konieczny, and Pierre Marquis. Belief merging versus judgment aggregation. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems, AAMAS'15, Istanbul, Turkey, May 4-8, 2015*, pages 999–1007, 2015.

[12] Patricia Everaere, Sébastien Konieczny, and Pierre Marquis. The epistemic view of belief merging : can we track the truth ? In *Nineteenth European Conference on Artificial Intelligence (ECAI'10)*, pages 621–626, 2010.

[13] Davide Grossi and Gabriella Pigozzi. Introduction to judgment aggregation. In *European Summer School in Logic, Language and Information*, volume 7388 of *Lecture Notes in Computer Science*, pages 160–209. Springer, 2010.

[14] G. Th. Guilbaud. Theories of the general interest, and the logical problem of aggregation. *Readings in Mathematical Social Science*, pages 262–307, 1966.

[15] Manish Gupta and Jiawei Han. Heterogeneous network-based trust analysis : a survey. *SIGKDD Explorations Newsletter*, 13(1) :54–71, 2011.

[16] Stephan Hartmann and Gabriella Pigozzi. Judgment aggregation and the problem of truth-tracking. In *Proceedings of the 11th Conference on Theoretical Aspects of Rationality and Knowledge (TARK XI). S*, volume 248, pages 248–252, 2007.

[17] Stephan Hartmann and Jan Sprenger. Judgment aggregation and the problem of tracking the truth. *Synthese*, 187 :209–221, 2012.

[18] Patrick Hummel. Jury theorems with multiple alternatives. *Social Choice and Welfare*, 34(1) :65–103, 2010.

[19] Sébastien Konieczny and Ramón Pino Pérez. Merging information under constraints : a logical framework. *Journal of Logic and Computation*, 12(5) :773–808, 2002.

[20] Sébastien Konieczny and Ramón Pino Pérez. Logic based merging. *Journal of Philosophical Logic*, 40(2) :239–270, 2011.

[21] Lewis A Kornhauser and Lawrence G Sager. Unpacking the court. *Yale Law Journal*, 96 :82, 1986.

[22] Jérôme Lang, Gabriella Pigozzi, Marija Slavkovik, and Leendert W. N. van der Torre. Judgment aggregation rules based on minimization. In Krzysztof R. Apt, editor, *Proceedings of the 13th Conference on Theoretical Aspects of Rationality and Knowledge (TARK-2011), Groningen, The Netherlands, July 12-14, 2011*, pages 238–246. ACM, 2011.

[23] Jérôme Lang, Marija Slavkovik, and Srdjan Vesic. Agenda separability in judgment aggregation. In Dale Schuurmans and Michael P. Wellman, editors, *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 1016–1022. AAAI Press, 2016.

[24] Christian List. The discursive dilemma and public reason. *Ethics*, 116(2) :362–402, 2006.

[25] Christian List. The theory of judgment aggregation : an introductory review. *Synthese*, 187(1) :179–207, 2012.

[26] Christian List and Robert E. Goodin. Epistemic democracy : Generalizing the Condorcet jury theorem. *Journal of Political Philosophy*, 9(3) :277–306, 2001.

[27] Christian List and Clemens Puppe. Judgement aggregation : A survey. In Paul Anand, Prasanta K. Pattanaik, and Clemens Puppe, editors, *The Handbook of Rational and Social Choice - an overview of new foundations and applications*, pages 457–482. Oxford University Press, 2009.

[28] Gabriella Pigozzi. Belief merging and the discursive dilemma : an argument-based account to paradoxes of judgment aggregation. *Synthese*, 152 :285–298, 2006.

[29] P. Z. Revesz. On the semantics of arbitration. *International Journal of Algebra and Computation*, 7(2) :133–160, 1997.

[30] Markus Schulze. A new monotonic and clone-independent single-winner election method. *Voting Matters*, 17 :9–19, 2003.

[31] Joseph Singleton and Richard Booth. Towards an axiomatic approach to truth discovery. *Journal of Autonomous Agents and Multi-Agent Systems*, 36(2) :1–49, 2022.

[32] Dalia Attia Waguih and Laure Berti-Equille. Truth discovery algorithms : An experimental evaluation. *arXiv preprint arXiv :1409.6428*, 2014.

[33] Hobart Peyton Young and Arthur Levenglick. A consistent extension of Condorcet's election principle. *SIAM Journal on Applied Mathematics*, 35(2) :285–300, 1978.

Actes JIAF 2024

Equité dans le problème d'allocation répétée de maisons

Karl Jochen Micheel[1] **Anaëlle Wilczynski**[2]

[1] Heinrich-Heine-Universität Düsseldorf, Germany
[2] MICS, CentraleSupélec, Université Paris-Saclay, France
karl.micheel@hhu.de anaelle.wilczynski@centralesupelec.fr

Résumé

Cet article considère le problème d'allocation de maisons–le problème d'allocation de ressources indivisibles où chaque agent doit recevoir exactement un objet–dans un contexte répété, où le même problème d'allocation est posé à plusieurs pas de temps. L'équité pouvant rarement être atteinte dans le cadre d'une décision unique, nous étudions si le cadre répété nous permet de l'obtenir. En particulier, nous introduisons plusieurs critères d'équité et étudions s'ils peuvent être satisfaits dans notre cadre d'allocation répétée de maisons. Bien que les problèmes de décision associés soient globalement difficiles d'un point de vue computationnel, nous identifions des restrictions pour lesquelles ces problèmes peuvent être résolus efficacement.

Abstract

This article considers a house allocation setting–where exactly one object has to be assigned to each agent–in a repeated context, where the same allocation problem is decided multiple times. Since fairness can be rarely achieved in a one-shot decision, we study whether fairness over time can be reached. In particular, we introduce several fairness criteria and investigate whether they can be satisfied in our repeated house allocation setting. While we show that most related decision problems are computationally hard in general, we identify restricted positive cases.

1 Introduction

L'allocation de maisons, où chaque agent reçoit exactement un objet, est l'un des problèmes de partage équitable les plus simples, qui capture néanmoins de nombreux problèmes du monde réel. Garantir l'équité est généralement difficile dans le problème d'allocation de maisons. Il existe cependant de nombreux contextes dans lesquels la décision d'allocation doit être répétée fréquemment, par exemple lors de l'attribution de cours à des enseignants ou de créneaux horaires à des travailleurs. Cette répétition peut être l'occasion de contourner les impossibilités d'équité qui surviennent dans un cadre de décision déterministe ponctuelle.

Dans cet article, nous étudions comment mesurer et atteindre l'équité dans le problème d'allocation répétée de maisons. Concrètement, nous examinons s'il est possible de construire les prochaines allocations, avec exactement un objet par agent, pour un nombre *donné fini* d'étapes, et éventuellement des allocations précédentes, de telle sorte que la séquence d'allocation globale atteigne l'équité dans le temps. Idéalement, nous aimerions pouvoir utiliser des solutions du problème d'affectation randomisé [1], mais celles-ci ne sont pas forcément implémentables en un horizon fini et fixé à l'avance, comme ce que nous considérons. Alors que plusieurs travaux récents ont étudié des problèmes de partage équitable dans le temps [2, 3], ils supposent des fonctions d'utilité additives pour les préférences des agents. En revanche, nous nous concentrons sur le problème standard, *ordinal*, d'allocation de maisons, ce qui induit des notions d'équité pertinentes bien différentes. Nous étudions trois critères d'équité qui se concentrent sur différents aspects de l'équité au fil du temps : l'*envie en miroir*, qui impose une symétrie dans l'envie entre les agents, le *traitement égal entre égaux*, qui exige que des agents identiques soient identiquement traités, et la minimisation de l'*envie cumulée maximale* entre toute paire d'agents.

2 Mesures d'équité

On considère un ensemble N de n agents et un ensemble M de n objets. Chaque agent $i \in N$ a des préférences ordinales strictes $\succ_i$ sur M. Un profil de préférence $\succ$ est l'ensemble des ordres $\succ_i$ pour tous les agents $i \in N$. Une allocation A est une bijection $A : N \to M$. Une allocation A est *Pareto-optimale* s'il n'existe pas d'autre allocation A' telle que $A'(i) \succeq_i A(i)$ pour chaque agent $i \in N$ et il existe un agent i tel que $A'(i) \succ_i A(i)$. Une séquence d'allocations $\mathcal{A}$ sur T étapes est notée $\mathcal{A} = \langle A^1, \ldots, A^T \rangle$. Nous notons

$\mathcal{A}_1 + \mathcal{A}_2$ la concaténation de deux séquences d'allocations $\mathcal{A}_1$ et $\mathcal{A}_2$. Un agent *i envie* l'agent j dans l'allocation A si $A(j) \succ_i A(i)$. Pour une séquence $\mathcal{A} = \langle A^1, \ldots, A^T \rangle$, l'envie accumulée de l'agent i envers l'agent j est $e^{\mathcal{A}}(i,j) := |\{t \in [T] : A^t(j) \succ_i A^t(i)\}|$. Notre première notion impose la symétrie de l'envie accumulée entre les paires d'agents.

Definition 1 (Envie en miroir)**.** *Une séquence d'allocation $\mathcal{A} = \langle A^1, \ldots, A^T \rangle$ satisfait l'envie en miroir si $e^{\mathcal{A}}(i,j) = e^{\mathcal{A}}(j,i)$, pour chaque paire d'agents i et j.*

Une autre façon d'envisager l'équité consiste à traiter de manière égale des agents considérés comme égaux.

Definition 2 (Traitement égal des égaux)**.** *Une séquence $\mathcal{A} = \langle A_1, \ldots, A^T \rangle$ satisfait le traitement égal des égaux si $\biguplus_{t\in[T]} A^t(i) = \biguplus_{t\in[T]} A^t(j)$ pour tous agents i et j avec les mêmes préférences, où $\biguplus$ représente l'union multiensemble.*

Nous explorons aussi la possibilité de garantir aux agents un nombre maximum de fois où ils envient le même agent.

Definition 3 (Envie cumulée)**.** *L'envie cumulée maximale d'une séquence $\mathcal{A}$ est donnée par* $\max_{(i,j)\in N^2} e^{\mathcal{A}}(i,j)$.

3 Résultats computationnels

Nous étudions les problèmes computationnels liés à nos concepts d'équité combinés avec l'optimalité de Pareto.

Mirrored Envy Completion (MirEnvyPO)	
Instance :	$(N, M, \succ)$, séquence passée $\mathcal{A}_1$, entier T
Question :	Existe-t-il une séquence d'allocations $\mathcal{A}_2 = \langle A_2^1, \ldots, A_2^T \rangle$ telle que $\mathcal{A}_1 + \mathcal{A}_2$ satisfait l'envie en miroir et chaque allocation A_2^t est Pareto-optimale pour $t \in [T]$?

Nous montrons que MirEnvyPO est résoluble en temps polynomial lorsque nous n'avons qu'une seule prochaine étape ($T \leq 1$). En revanche, nous montrons qu'à partir d'un horizon de deux étapes, le problème devient NP-complet même s'il n'y a aucune allocation passée ($\mathcal{A}_1$ est vide). Néanmoins, nous exhibons un algorithme polynomial dans le cas d'un horizon de deux étapes, lorsque les agents ont les mêmes préférences. Le problème MirEnvyPO reste ouvert pour $T = 3$ et des agents avec les mêmes préférences, mais nous le conjecturons difficile.

Equal Treatment of Equals Completion (ETEPO)	
Instance :	$(N, M, \succ)$, séquence passée $\mathcal{A}_1$, entier T
Question :	Existe-t-il une séquence d'allocations $\mathcal{A}_2 = \langle A_2^1, \ldots, A_2^T \rangle$ telle que $\mathcal{A}_1 + \mathcal{A}_2$ satisfait un traitement égal des égaux et que chaque allocation A_2^t est Pareto-optimale pour $t \in [T]$?

Soit EQ l'ensemble des sous-ensembles maximaux d'agents avec les mêmes préférences. Pour garantir un traitement égal entre égaux, pour chaque $C \in EQ$, chaque agent de C doit obtenir chaque objet le même nombre de fois que les autres agents de C. En l'absence de séquence passée, le traitement égal des égaux ne peut pas être assuré s'il existe un groupe $C \in EQ$ tel que $|C| > T$, et peut toujours être garanti si chaque taille de groupe dans EQ divise T. Il s'ensuit que ETEPO est résoluble en temps polynomial lorsqu'il n'y a que deux prochaines étapes et pas d'allocation passée. Bien que la complexité exacte de ETEPO reste ouverte, nous soupçonnons le problème d'être NP-complet, même pour trois étapes et pas d'allocation passée.

En revanche, nous montrons que le problème ETE, où l'optimalité de Pareto n'est plus requise, est NP-complet, même sans allocation passée. Néanmoins, ETE est résoluble efficacement lorsqu'il n'y a pas d'allocation passée et $|EQ|$ est fixé. De plus, nous dérivons un algorithme polynomial pour ETE si $T < \min\{|C| : C \in EQ \text{ et } |C| > 1\}$.

Max Cumulative Envy Bound (MaxCumEnvyPO)	
Instance :	$(N, M, \succ)$, entier T, entier B
Question :	Existe-t-il une séquence $\mathcal{A} = \langle A^1, \ldots, A^T \rangle$ telle que $e^{\mathcal{A}}(i,j) \leq B$, pour tous agents i, j, et A^t est Pareto-optimale pour tout $t \in [T]$?

Il est toujours possible de construire une séquence d'allocation où chaque agent envie un autre au maximum $\lceil \frac{T}{2} \rceil$ fois. Cette borne est atteinte lorsque les agents ont les mêmes préférences. En général, nous prouvons que MaxCumEnvyPO est NP-complet même pour $B = \frac{T}{3}$ et $T = 3$.

4 Perspectives

Il serait intéressant de relâcher certains de nos critères d'équité. On pourrait par exemple assouplir l'envie en miroir en laissant une certaine marge dans la différence d'envie entre les agents. De plus, on pourrait redéfinir l'égalité de traitement entre égaux pour permettre plus de flexibilité dans le traitement similaire tout en élargissant le champ des agents qui devraient être traités de la même manière. Enfin, une possibilité serait de permettre une certaine variabilité dans l'ensemble des agents ou des objets, ou de supposer que les préférences peuvent être modifiées au fil du temps.

Références

[1] Anna Bogomolnaia and Hervé Moulin. A new solution to the random assignment problem. *Journal of Economic theory*, 100(2) :295–328, 2001.

[2] Ioannis Caragiannis and Shivika Narang. Repeatedly matching items to agents fairly and efficiently. In *Proc. of SAGT-23*, pages 347–364, 2023.

[3] Ayumi Igarashi, Martin Lackner, Oliviero Nardi, and Arianna Novaro. Repeated fair allocation of indivisible items. In *Proc. of AAAI-24*, pages 9781–9789, 2024.

Session 4 : IA explicable

Actes JIAF 2024

Explications et caractérisation de décisions équitables

Hénoïk Willot[1] **Khaled Belahcene**[2] **Sébastien Destercke**[1]

[1] Heudiasyc, Université de Technologie de Compiègne, France
[2] MICS, CentraleSupélec, Université Paris-Saclay, France
henoik.willot@hds.utc.fr
khaled.belahcene@centralesupelec.fr
sebastien.destercke@hds.utc.fr

Résumé

Les *Ordered weighted averaging* (*OWA*), aussi connues sous le nom *Generalised Gini Index*, sont régulièrement utilisées pour obtenir des décisions équitables. Cependant, bien qu'elles assurent un certain niveau d'équité dans leurs résultats, deux questions subsistent, pourquoi recommandent-elles une alternative donnée et à quel point cette décision est-elle robuste. Nous apportons des outils pratiques et théorisés pour répondre à ces questions, à l'aide d'un moteur d'explications pour les *OWA* robustes qui consiste en une chaîne transitive d'arguments normalisés et considérés évidents, eux-même basés sur les propriétés normatives du modèle. À travers une étude théorique du moteur, nous montrons qu'il est correct (*sound*) et complet par rapport au modèle, et donnons une borne théorique sur la longueur des explications ainsi qu'un algorithme efficace (bien que minimiser la longueur soit NP-difficile). Nous fournissons aussi des éléments montrant que le moteur fonctionne bien sur des données synthétiques. Ainsi, nous garantissons qu'une explication peut toujours être trouvée, et que le raisonnement produit par le moteur explicatif est valide. De plus, ces explications permettent de questionner les fondements du modèle, donc de permettre sa validation et d'établir sa redevabilité, qui sont des composants clés d'une IA de confiance.

Abstract

Ordered weighted averaging (OWA) functions, a.k.a. Generalised Gini Index, are routinely used to obtain fair solutions. However, while they ensure some level of fairness in the result, two remaining questions are why they recommend a given alternative, and how robust is this recommendation. We bring practical and theoretically grounded solutions to these questions, by providing an explanation engine for robust OWA that consists in a normative transitive chain of self-evident arguments, themselves based on the normative properties of the model. We provide a thorough theoretical study of the engine, showing that it is sound and complete with respect to the model, with a theoretical upper bound on the length of the explanation and a tractable algorithm (even though minimizing the length is NP-hard). We also provide experimental evidence that the engine performs well on synthetic data. Thus, we guarantee that an explanation can always be found, and that reasoning according to the provided scheme always produces a valid statement. Moreover, the explanations allow to probe the normative requirements of the model, so as to allow validation, accountability and recourse, that are key components of trustworthy AI.

1 Introduction

Équité, Robustesse et Explicabilité sont trois piliers de l'IA de confiance. Les *Ordered Weighted Averaging* (*OWA*) [39], aussi connues sous le nom *generalized Gini indices* [38], sont communément utilisées pour assurer le pilier de l'équité dans différents contextes, par exemple en économie et en choix social computationnel [2], en optimisation multi-objectifs [8], ou en apprentissage de préférences [17] pour n'en citer que quelques uns. Cependant, nous ne connaissons pas de travaux cherchant à expliquer les *OWA* équitables robustes, les *OWA* à poids décroissants définis à partir d'informations partielles. Cela contraste avec d'autres modèles de complexité similaire comme les sommes pondérées robustes, pour lesquelles des moteurs explicatifs corrects (*sound*), complets et efficaces existent.

Nous répondons à ces problèmes, tout d'abord en caractérisant les *OWA* robustes avec des propriétés normatives, puis en utilisant cette caractérisation pour définir un moteur explicatif dont les explications peut être prouvées correctes, complètes et calculables efficacement. Nous avons plusieurs raisons d'adopter ce point de vue normatif et logique :

— il permet un raisonnement réfutable, de construire un système certifié et redevable, aux tiers préjudiciés de contester une décision sur une base formelle ;
— les *OWA* ne sont pas compatibles avec les outils explicatifs comme les *Shapley values* [28] (à cause de

leur symétrie) ou basées sur les gradients [34] (car elles sont hautement non linéaires) ;
— les explications produites reposent uniquement sur l'information donnée et sur les propriétés normatives, sans hypothèse supplémentaire ni divulgation des paramètres du modèle. Ainsi le processus sera plus résilient aux brèches de données personnelles ou à la manipulation, et minimise le biais inductif.

Concrètement, nous souhaitons expliquer des prédictions du type "$\mathbf{x}$ est moins désirable que $\mathbf{y}$", où $\mathbf{x}, \mathbf{y}$ sont des alternatives, ou des états du monde, et où notre fonction de décision doit satisfaire un nombre de propriétés normatives désirables, en plus des préférences données. De plus, nous basons nos conclusions et explications sur les déductions valides pour chaque modèle possible, consistant avec l'information observée. Une telle inférence sceptique est communément utilisée en logique [18] ou dans la décision multi-objectifs [1] et dans l'incertain [35, 29], assurant robustesse dans le sens fort du terme.

Notre proposition commencera avec les propriétés normatives les plus fondamentales, puis nous allons progressivement augmenter leur complexité jusqu'à définir les *OWA* robustes (qui, à leur limite, incluent les *OWA* précis). Pendant ces étapes, nos résultats vont montrer pourquoi expliquer les *OWA* équitables robustes a reçu peu voir aucune attention : le problème est loin d'être trivial, du point de vue théorique ou algorithmique, et n'est pas une simple adaptation de résultats existants, comme ceux de la somme pondérée. Notre point de départ est le problème de la comparaison d'alternatives, décrites par un nombre de points de vue exprimés sur une échelle commensurable à l'aide d'une structure de préférences qui vérifie des propriétés fortes de la théorie de la décision (transitivité, symétrie, redistributivité et monotonie) qui sont normativement désirables pour le procédé de décision étudié. De telles structures de préférences sont fortement liées à la notion de dominance de Lorenz généralisée, introduite il y a longtemps dans le domaine de l'économie du bien-être [36].

Dans la section 4, nous nous intéressons aux OWA équitables robustes (c.-à-d. avec des poids décroissants), qui raffinent les préférences de la dominance de Lorenz généralisée, résolvant le problème qu'a cette dernière à régulièrement ne pas pouvoir comparer les alternatives, c.-à-d. être trop indécisive. Pour ce faire, nous considérons qu'en plus de devoir satisfaire les propriétés théoriques précédemment mentionnées, un utilisateur a émis des préférences, sous forme de paire comparative obtenues par exemple avec de l'apprentissage actif [7], mais qui ne permettent d'identifier qu'un sous-ensemble de modèles possibles.

Nos principales contributions sont les suivantes :

— A partir de la littérature sur les explications de la dominance de Lorenz, nous montrons que trouver l'explication optimale (la plus courte) sous forme d'un ensemble successif de transferts est un problème NP-difficile et nous donnons une heuristique ayant de meilleures performances empiriques que les précédentes.
— Nous proposons, ce qui a notre connaissance est, une nouvelle axiomatique pour des ensembles convexes d'OWA (que nous nommons OWA équitables robustes), qui inclue la dominance de Lorenz et l'index de Gini généralisé dans un cadre unique. De plus, ces axiomes étant plutôt naturels nous permettent de définir des méchanismes explicatifs. Cela contraste avec les axiomatiques qui ont besoin d'axiomes techniques comme la continuité qui sont difficiles à utiliser dans une explication. Nous montrons aussi que nos explications sont logiquement correctes et complètes, c.-à-d. que toutes les préférences peuvent être expliquées et que toutes les préférences expliquées sont vraies. Nous proposons aussi des heuristiques pour fournir rapidement des explications.

2 Préliminaires

Nous étudions les préférences entre *alternatives*, décrite par plusieurs attributs mesurés sur une échelle commune : nous noterons $[n] = \{1, \dots, n\}$ l'ensemble des attributs et par $\mathbb{X}$ cette échelle commune, étant un intervalle non trivial des réels. Les alternatives $\mathbf{x} \in \mathbb{X}^n$ sont décrites par une minuscule. $[n]$ peut représenter des points de vues dans différents contextes, comme la décision multi-critère ou multi-agent, et les alternatives sont des choix possibles décrits par ces points de vues, par exemple la distribution de richesse dans un groupe d'agents. Nous noterons $(\mathbf{e}^1, \dots, \mathbf{e}^n)$ pour la base canonique de $\mathbb{R}^n$ et $\widehat{\mathbb{X}^n}$ le sous-ensemble de $\mathbb{X}^n$ des n-uplets triés par ordre croissant.

Les préférences sont représentées par une relation binaire $\mathcal{R}$ sur $\mathbb{X}^n$. Étant donné deux alternatives $\mathbf{x}, \mathbf{y}$ de $\mathbb{X}^n$, la *paire comparative* $(\mathbf{x}, \mathbf{y}) \in \mathcal{R}$ (ou $\mathbf{x}\ \mathcal{R}\ \mathbf{y}$) dénote que l'alternative $\mathbf{x}$ est au plus aussi désirable que l'alternative $\mathbf{y}$. Par conséquence, il existe quatre possibilités quand nous comparons $\mathbf{x}$ à $\mathbf{y}$: *(i)* $\mathbf{y}$ est strictement préféré à $\mathbf{x}$ si $(\mathbf{x}, \mathbf{y}) \in \mathcal{R}$ et $(\mathbf{y}, \mathbf{x}) \notin \mathcal{R}$; *(ii)* $\mathbf{x}$ est strictement préféré à $\mathbf{y}$ si $(\mathbf{x}, \mathbf{y}) \notin \mathcal{R}$ et $(\mathbf{y}, \mathbf{x}) \in \mathcal{R}$; *(iii)* $\mathbf{x}, \mathbf{y}$ sont indifférents quand $(\mathbf{x}, \mathbf{y}) \in \mathcal{R}$ et $(\mathbf{y}, \mathbf{x}) \in \mathcal{R}$ [1] ; *(iv)* $\mathbf{x}, \mathbf{y}$ sont *incomparables* quand $(\mathbf{x}, \mathbf{y}), (\mathbf{y}, \mathbf{x}) \notin \mathcal{R}$. Quelques ensembles de préférences nous intéressent tout particulièrement :

Définition 1 (réarrangement $\mathcal{S}$)**.** Soit $\mathcal{S}$ l'ensemble de paires comparatives $(\mathbf{x}, \mathbf{y})$ t.q. $\mathbf{y}$ est une permutation de $\mathbf{x}$. $\mathcal{S}$ est une relation d'équivalence et chaque alternative $\mathbf{x} \in \mathbb{X}^n$ a un unique équivalent $\widehat{\mathbf{x}}$ par $\mathcal{S}$ dans l'ensemble $\widehat{\mathbb{X}^n}$.

Définition 2 (transferts $\mathcal{T}$)**.** Soient $t \in \mathbb{R}$, $i, j \in [n]$ et la paire comparative $(\widehat{\mathbf{x}}, \widehat{\mathbf{y}})$, où les deux alternatives sont triées dans l'ordre croissant, et où $\widehat{\mathbf{y}}$ est la situation où, en

1. Pour ce faire, $\mathcal{R}$ est évidemment *reflexive*, c.-à-d. $(\mathbf{x}, \mathbf{x}) \in \mathcal{R}$. Nous faisons cette hypothèse tout au long de l'article.

commençant par $\widehat{\mathbf{x}}$, la quantité t est prise de l'agent j et donnée à l'agent i, c.-à-d. $\widehat{\mathbf{y}} = \widehat{\mathbf{x}} + t\mathbf{e}^i - t\mathbf{e}^j$, noté $\tau^t_{j\to i}$. Quand $t > 0$ et $i < j$, ce transfert est dit *redistributif*[2], et notons $\mathcal{T}$ l'ensemble de tous les transferts redistributifs, c.-à-d. $\mathcal{T} := \bigcup_{t>0} \bigcup_{1\le i<j\le n} \tau^t_{j\to i}$.

Définition 3 (dons $\mathcal{G}$)**.** Soit $\mathcal{G} := \{(\mathbf{x}, \mathbf{y}) : \forall i \in [n]\ \mathbf{x}_i \le \mathbf{y}_i\}$, l'ensemble des paires comparatives où les préférences des agents de $[n]$ sont unanimes [3]

Nous donnons un exemple général, où une autorité centrale doit redistribuer des revenus d'investissements, permettant d'illustrer le problème considéré, les notions impliquées et d'appliquer notre solution. Nous invitons le lecteur à revenir vers lui au cours de sa lecture.

Exemple 1 (Exemple illustratif)**.**

Inv.	*Alice*	*Bob*	*Charlie*	*David*	*Emma*
a	*31*	*83*	*70*	*16*	*51*
b	*28*	*98*	*25*	*2*	*84*
c	*22*	*23*	*76*	*82*	*34*
d	*96*	*6*	*18*	*17*	*88*

Nous commençons par affirmer que les alternatives doivent être anonymisées et triées par ordre croissant de satisfaction.

Inv.	*#1*	*#2*	*#3*	*#4*	*#5*
$\widehat{\mathbf{a}}$	*16*	*31*	*51*	*70*	*83*
$\widehat{\mathbf{b}}$	*2*	*25*	*28*	*84*	*98*
$\widehat{\mathbf{c}}$	*22*	*23*	*34*	*76*	*82*
$\widehat{\mathbf{d}}$	*6*	*17*	*18*	*88*	*96*

Supposons que l'autorité centrale a déjà statué sur le fait que **b** *est au plus aussi désirable que* **d** *et qu'elle utilise un OWA précis* [4] $O_{WA_{\{\omega^*\}}}$ *avec* $\omega^* = (.70, .10, .10, .05, .05)$*. Comment peut-on expliquer pourquoi* **c** *est préféré à* **a** *tout en gardant* ω^* *secret ?* [5]

Pour prouver cette préférence, nous construisons une chaîne d'alternatives $(\mathbf{a}, \widehat{\mathbf{a}}, \mathbf{x}^1, \mathbf{x}^2, \widehat{\mathbf{c}}, \mathbf{c})$*, avec* $\mathbf{x}^1 :=$ (16 35 47 70 83) *et* $\mathbf{x}^2 :=$ (22 23 32 76 80)*.* $\mathbf{x}^1$ *doit être considéré meilleur que* $\widehat{\mathbf{a}}$*, car il est obtenu en transférant* 4 *unités du troisième agent le moins satisfait vers le second agent le moins satisfait. La situation* $\mathbf{x}^2$ *doit être considérée meilleure que* $\mathbf{x}^1$*, vu que le changement* $(+6, -12, -15, +6, -3)$ *de* $\mathbf{x}^1$ *vers* $\mathbf{x}^2$ *doit être considéré positif, comme ce dernier correspond* ceteris paribus *à une fois et demie le changement de* $\widehat{\mathbf{b}}$ *vers* $\widehat{\mathbf{d}}$*. Enfin,* $\widehat{\mathbf{c}}$ *doit être considéré meilleur que* $\mathbf{x}^2$*, car chaque agent est au moins aussi satisfait. Ainsi, la transitivité des préférences permet de dire que* **c** *est préféré par rapport à* **a***. Techniquement, les préférences* $(\mathbf{a}, \widehat{\mathbf{a}})$ *et* $(\widehat{\mathbf{c}}, \mathbf{c})$ *sont des réarrangements,* $(\widehat{\mathbf{a}}, \mathbf{x}^1)$ *est un transfert redistributif et* $(\mathbf{x}^2, \widehat{\mathbf{c}})$ *est un don.*

3 Expliquer la dominance de Lorenz

Notre but est de caractériser nos inférences sceptiques et moteurs explicatifs pour des règles de décision équitables, à commencer par la dominance de Lorenz restreinte puis sa version généralisée, définies par :

Définition 4 (Dominances de Lorenz $\mathcal{L}$ et $\mathcal{L}^*$)**.** La *dominance de Lorenz généralisée* est la relation binaire $\mathcal{L}$ définie sur $\mathbb{X}^n$ telle que $(\mathbf{x}, \mathbf{y}) \in \mathcal{L}$ ssi $\forall i \in [n]$, $\sum_{k=1}^{i} \widehat{\mathbf{x}}_k \le \sum_{k=1}^{i} \widehat{\mathbf{y}}_k$. La *dominance de Lorenz restreinte* est le sous-ensemble $\mathcal{L}^*$ de $\mathcal{L}$ où $\mathbf{x}, \mathbf{y}$ ont le même revenu total, c.-à-d. $\sum_{k\in[n]} \mathbf{x}_k = \sum_{k\in[n]} \mathbf{y}_k$.

3.1 Sémantique des préférences sceptiques

Dans cet article, nous nous intéressons aux relations de préférences qui vérifient un certain nombre de propriétés issues de la théorie de la décision.

Propriété (t). $\mathcal{R}$ est *transitive* quand, pour tous $\mathbf{x}, \mathbf{y}, \mathbf{z} \in \mathbb{X}^n$, si $(\mathbf{x}, \mathbf{y}) \in \mathcal{R}$ et $(\mathbf{y}, \mathbf{z}) \in \mathcal{R}$, alors $(\mathbf{x}, \mathbf{z}) \in \mathcal{R}$.

Propriété (s). $\mathcal{R}$ est *symétrique* quand elle inclut (ou raffine) $\mathcal{S}$, c.-à-d. $\mathcal{R} \supseteq \mathcal{S}$ contient tous les réarrangements.

Propriété (r). $\mathcal{R}$ est *redistributive* quand elle inclut $\mathcal{T}$, c.-à-d. $\mathcal{R} \supseteq \mathcal{T}$ contient tous les transferts redistributif.

Notons $\mathfrak{P}_{(t,s,r)}$ l'ensemble de toutes les relations binaires réflexives sur $\mathbb{X}^n$ vérifiant simultanément ces trois propriétés. En assumant que cet ensemble est non vide (montré dans la prochaine section), soit $\mathcal{P}_{(t,s,r)}$ la relation de préférence définie comme l'intersection de toutes les relations de $\mathfrak{P}_{(t,s,r)}$. Comme les trois propriétés sont stables par intersection [6], $\mathcal{P}_{(t,s,r)}$ appartient à $\mathfrak{P}_{(t,s,r)}$ et est son plus petit élément du point de vue de l'inclusion. Par conséquent, du point de vue sémantique, $\mathfrak{P}_{(t,s,r)}$ peut être vu comme l'ensemble des mondes possibles et $\mathcal{P}_{(t,s,r)}$ comme l'ensemble des décisions qui peuvent être inférées sceptiquement, c.-à-d. qui se produisent dans tous les mondes possibles. $\mathfrak{P}_{(t,s,r)}$ peut aussi être vu comme un jury, où les jurés sont toutes les relations de préférences qui respectent les propriétés normatives, et $\mathcal{P}_{(t,s,r)}$ est l'ensemble des décisions unanimes en leur sein. Ces décisions prudentes sont nécessaires du

2. À interpréter : "prendre une quantité $t > 0$ de l'agent j et la donner à l'agent plus pauvre i" (en conservant l'ordre social). De tels transferts sont aussi appelés transferts de *Pigou-Dalton* ou de *Robin des Bois*.

3. Il s'agit simplement de la dominance de Pareto de $\mathbf{y}$ sur $\mathbf{x}$, où $\mathbf{y}_i = \mathbf{x}_i + t_i$, $t_i > 0$ étant le don fait à l'agent i

4. La définition de sa représentation numérique est rappelée au début de la section 4.1.

5. Nous pouvons voir que calculer l'importance individuelle des agents, par exemple de Bob avec la dérivée partielle $\partial_{\text{Bob}} O_{WA_{\{\omega^*\}}}(\mathbf{a}) = .70$, attribue une haute importance à Bob ce qui est trompeur, tandis que les valeurs de Shapley des joueurs Alice, Bob, Charlie, David et Emma sont égales.

6. dans le sens où si $\mathcal{R}_1$ et $\mathcal{R}_2$ satisfont simultanément cette propriété, alors $\mathcal{R}_1 \cap \mathcal{R}_2$ aussi.

point de vue des propriétés normatives et ne sont donc pas arbitraires ou contingentes. En se concentrant sur ces décisions, nous offrons de la robustesse à l'utilisateur, et nous espérons les accompagner de preuves et d'explications basées sur ces propriétés normatives.

3.2 Une représentation numérique de $\mathcal{L}^*$

Il est facile de vérifier que $\mathcal{L}$ et $\mathcal{L}^*$, définies dans Def. 4, satisfont les propriétés (t), (s) et (r). $\mathfrak{P}_{(t,s,r)}$ est donc non vide, et, comme nous le verrons, $\mathcal{L}^*$ est son plus petit élément, soit $\mathcal{P}_{(t,s,r)} = \mathcal{L}^*$. C'est un résultat fort, permettant de dire qu'une paire comparative $(\mathbf{x}, \mathbf{y})$ est obtenue par toutes les relations de préférences vérifiant (t), (s) et (r) simplement en triant $\mathbf{x}$ et $\mathbf{y}$ et en calculant leur sommes cumulatives et comparant (au plus) n paires d'éléments.

3.3 Un calcul correct et complet des préférences

Nous nous intéressons à créer un système déductif formel qui reflète ces propriétés normatives, et qui infère des paires comparatives à partir d'autres.

Inférence. Nous associons la propriété (t) à la règle

$$\textbf{Règle T :}\ \frac{(\mathbf{a},\mathbf{b}),(\mathbf{b},\mathbf{c})}{(\mathbf{a},\mathbf{c})}\ \text{(transitivité)}$$

Vérités premières [7]**.** Pour refléter les propriétés (s) et (r), nous considérons les réarrangements $\mathcal{S}$ et les transferts redistributifs $\mathcal{T}$ comme des vérités premières.

Soit $cl_T(\mathcal{S} \cup \mathcal{T})$ la clôture déductive [8] de l'ensemble des vérités premières $\mathcal{S} \cup \mathcal{T}$ par l'opérateur T, c.-à-d. l'ensemble de toutes les paires comparatives qui peuvent être prouvées à partir de prémisses dans $\mathcal{S}$ ou dans $\mathcal{T}$ en enchaînant les déductions à partir de la règle T. La *correction* (*soundness*) du système formel par rapport à la sémantique, c.-à-d. $cl_T(\mathcal{S} \cup \mathcal{T}) \subseteq \mathcal{P}_{(t,s,r)}$, signifiant que toute décision qui peut être prouvée peut aussi être sceptiquement inférée, est obtenue immédiatement par la construction des règles et vérités premières qui reflètent les propriétés des préférences. La *complétude*, c.-à-d. $\mathcal{P}_{(t,s,r)} \subseteq cl_T(\mathcal{S} \cup \mathcal{T})$, signifiant que toute paire qui ne peut être réfutée empiriquement peut être prouvée, provient du fait que $cl_T(\mathcal{S} \cup \mathcal{T})$ satisfait (t) car elle est fermée sous T, et évidemment (s) et (r).

3.4 Explications schématiques

Bien que le résultat de complétude soit satisfaisant, les preuves résultantes seront toujours sous forme d'arbres, qui ne seront sûrement pas assez concis ou simples pour être cognitivement acceptées par des agents. Nous proposons donc des explications sous forme d'une *chaîne transitive d'arguments* formant une séquence "d'actes de langage".

Définition 5 (Moteur ATX)**.** Soit $\mathcal{R}$ une relation binaire sur $\widehat{\mathbb{X}^n}$. Une *explication anonyme et transitive basée sur les vérités dans* $\mathcal{R}$ de longueur ℓ ($\mathcal{R}$-ATX$^\ell$) est une paire (s, c) où le *support* s est un ℓ-uplet de paires comparatives $s = ((\mathbf{x}^1, \mathbf{y}^1), \ldots, (\mathbf{x}^\ell, \mathbf{y}^\ell)) \in \mathcal{R}^\ell$ et *l'affirmation* c est une paire comparative $c = (\mathbf{x}, \mathbf{y})$ vérifiant $\mathbf{x}^1 = \widehat{\mathbf{x}}$, $\mathbf{y}^\ell = \widehat{\mathbf{y}}$ et, pour tout entier k entre 2 et ℓ, $\mathbf{x}^k = \mathbf{y}^{k-1}$.

Soit $\ell \in \mathbb{N}^*$ et soit $\mathcal{E}(\mathcal{T}\text{-ATX}^\ell)$ l'ensemble de toutes les *décisions explicables*, c.-à-d. des affirmations associées à une chaîne de vérités premières de $\mathcal{T}$ par une ATX de taille ℓ. Cet ensemble est inclus dans $cl_T(\mathcal{S} \cup \mathcal{T})$, car une ATX justifiant la conclusion $(\mathbf{x}, \mathbf{y})$ peut être vue comme une chaîne transitive entre $\mathbf{x}$ et $\mathbf{y}$, où la première (c.-à-d. $(\mathbf{x}, \widehat{\mathbf{x}})$) et la dernière (c.-à-d. $(\widehat{\mathbf{y}}, \mathbf{y})$) paire comparative appartiennent à $\mathcal{S}$, et toutes les autres à $\mathcal{T}$: les $\mathcal{T}$-ATX sont correctes par rapport à $cl_T(\mathcal{S} \cup \mathcal{T})$. Sont-elles complètes ? En effet, elles sont même complètes par rapport à $\mathcal{L}^*$, d'après ce résultat bien connu des années 1930.

Lemme 1 (Hardy et al. 1929)**.** $\mathcal{L}^* \subseteq \bigcup_{\ell=1}^n \mathcal{E}(\mathcal{T}\text{-}ATX^\ell)$

Nous rappelons rapidement la preuve constructive, étant donné qu'elle forme un algorithme que nous améliorerons.

Ébauche de preuve. Initialiser $\mathbf{x}^0 := \widehat{\mathbf{x}}$; A l'étape k trouver i^*, j^*, t^* t.q. $j^* = \arg\min_j \mathbf{x}_j^{k-1} > \widehat{\mathbf{y}}_j$, $i^* = \arg\max_{i:i<j} \mathbf{x}_i^{k-1} < \widehat{\mathbf{y}}_i$ et $t^* = \min(\widehat{\mathbf{y}}_{i^*} - \mathbf{x}_{i^*}^{k-1}, \mathbf{x}_{j^*}^{k-1} - \widehat{\mathbf{y}}_{j^*})$ définir $\mathbf{x}^k$ t.q. $(\mathbf{x}^{k-1}, \mathbf{x}^k) \in \tau_{j^* \to i^*}^{t^*}$. Le nombre d'agents $i \in [n]$ t.q. $\mathbf{x}_i^k \neq \widehat{\mathbf{y}}_i$ décroît strictement avec k, d'où la terminaison de l'algorithme qui délivre une $\mathcal{T}$-ATX de taille au plus n pour $(\mathbf{x}, \mathbf{y})$. □

Comme les $\mathcal{T}$-ATX sont correctes et complètes, nous obtenons le résultat suivant

Théorème 1 (Explicabilité de la dominance de Lorenz restreinte)**.**

$$\bigcup_{\ell=1}^n \mathcal{E}(\mathcal{T}\text{-}ATX^\ell) = cl_T(\mathcal{S} \cup \mathcal{T}) = \mathcal{P}_{(t,s,r)} = \mathcal{L}^*$$

Exemple 2. *Considérons les alternatives* $\widehat{\mathbf{c}}$ *et* $\widehat{\mathbf{b}}$ *de l'exemple 1. Calculer leur sommes cumulées montre que* $(\mathbf{b}, \mathbf{c}) \in \mathcal{L}^*$. *La plus petite explication supportant* $(\mathbf{b}, \mathbf{c})$ *est de longueur 4 :*

$$\widehat{\mathbf{b}}\ \tau_{4\to1}^{18}\ (\overline{20}\ 25\ 28\ \underline{66}\ 98)\ \tau_{5\to4}^{10}\ (20\ 25\ 28\ \overline{76}\ \underline{88})$$

$$\tau_{2\to1}^{2}\ (\overline{22}\ \underline{23}\ 28\ 76\ 88)\ \tau_{5\to3}^{6}\ \widehat{\mathbf{c}}$$

7. Aussi appelées *axiomes*, mais nous évitons cette dénomination qui change de sens selon l'utilisation en logique ou en théorie de la décision.

8. C'est aussi la clôture transitive vu que le seul opérateur est la transitivité, ce qui est contingent aux propriétés de la dominance de Lorenz.

$$\mathcal{D}_k^* = \left\{ j \in [n] \mid \mathbf{x}_j^{k-1} \geq \widehat{\mathbf{y}}_j \text{ et } \mathbf{x}_j^{k-1} - \mathbf{x}_{j-1}^{k-1} \geq \mathbf{x}_j^{k-1} - \widehat{\mathbf{y}}_j \right\},$$
$$\mathcal{R}_k^* = \left\{ i \in [n] \mid \mathbf{x}_i^{k-1} \leq \widehat{\mathbf{y}}_i \text{ et } \mathbf{x}_{i+1}^{k-1} - \mathbf{x}_i^{k-1} \geq \widehat{\mathbf{y}}_i - \mathbf{x}_i^{k-1} \right\},$$
$$t_k^* = t(i_k^*, j_k^*) = \max_{i \in \mathcal{R}_k^*, j \in \mathcal{D}_k^*, i<j} \min \left(\mathbf{x}_j^k - \widehat{\mathbf{y}}_j, \widehat{\mathbf{y}}_i - \mathbf{x}_i^k \right)$$

FIGURE 1 – Notre algorithme – le choix du donneur j^*, receveur i^* et de la quantité t^* pour le transfert à l'étape k.

3.5 Aspects computationnels

Ainsi, étant donné une paire comparative $(\mathbf{x}, \mathbf{y})$, il est équivalent de *(i)* décider si elle est obtenue pour toute relation de préférence vérifiant (t), (s) et (r) ; *(ii)* chercher pour une preuve déductive utilisant la règle T avec les prémisses issues de $\mathcal{S}$ et $\mathcal{T}$; *(iii)* résoudre le problème de trouver une explication ; ou *(iv)* ordonner - sommer - comparer à l'aide de la représentation numérique de $\mathcal{L}^*$. Évidemment, la dernière est la plus facile d'un point de vue computationnel. En effet, nous prouvons que le problème de trouver une explication d'une taille donnée est difficile.

Théorème 2 (Difficulté de trouver des explications courtes)**.** *Étant donné* $(\mathbf{x}, \mathbf{y}) \in \mathcal{L}^*$ *et un entier positif* k*, décider s'il existe une* $\mathcal{T}$*-ATX de taille au plus* k *pour* $(\mathbf{x}, \mathbf{y})$ *est NP-difficile.*

Ébauche de preuve. Réduction depuis le problème de 3-partition [19]. Soit S un ensemble d'entiers t.q. $|S| = 3m$, $\sum_{s \in S} s = mT$ et $\frac{T}{4} < s < \frac{T}{2}$ $\forall s \in S$, entrée du problème de 3-partition. Nous construisons les alternatives $\mathbf{x}$ et $\mathbf{y} \in \widehat{\mathbb{X}^{4m}}$ t.q. $(\mathbf{x}, \mathbf{y}) \in \mathcal{L}^*$, définies par $\mathbf{x}_i = \sum_{j=1}^{i-1} \widehat{S}_j$ si $1 \leq i \leq 3m$ et $\mathbf{x}_i = iT$ si $3m < i \leq 4m$ et par $\mathbf{y}_i = \sum_{j=1}^{i} \widehat{S}_j$ si $1 \leq i \leq 3m$ et $\mathbf{y}_i = (i-1)T$ si $3m < i \leq 4m$. □

La difficulté de résolution nous amène à considérer :

— Une formulation en programmation linéaire mixte (MILP) sous la forme d'un problème de planification continue [9] dont la solution est l'explication la plus courte.

— Une heuristique gloutonne, semblable à l'algorithme HLP [22] sous-jacent au Lemme 1, mais orienté vers des explications courtes, est décrite par ses choix de valeurs pour i^*, j^*, t^* à l'étape k dans la figure 1.

Les résultats expérimentaux menés sur des données synthétiques sont donnés dans la table 1. Pour un nombre donné n d'agents, un ensemble de 10 alternatives est échantillonné de $\mathbb{X}^n := [1000]^n$ t.q. la richesse totale de chaque alternative soit égale à $200n$. Les résultats données, obtenus sur un ordinateur portable standard, sont moyennés sur 10 répétitions indépendantes. La table 1 montre que notre heuristique est très rapide et plutôt proche de l'optimum. Les résultats suggèrent que la taille optimale croît selon $0.6n$, tandis que notre heuristique croît selon $0.7n$.

3.6 Le cas de la dominance de Lorenz généralisée

La machinerie que nous avons construit ne permet pas de comparer des populations qui ont une richesse totale différente. Pour cette raison, les économistes ont proposé de considérer les *dons* comme désirables.

Propriété (m). $\mathcal{R}$ est *monotone* quand $\mathcal{R} \supseteq \mathcal{G}$.

Cette propriété est une sorte de garantie *d'efficacité* : dépenser le surplus est préférable à ne pas le dépenser [10]

Sémantique et représentation. Nous pouvons observer que $\mathcal{L}$ vérifie (m) (alors que $\mathcal{L}^*$ non), ainsi l'ensemble $\mathfrak{P}_{(t,s,r,m)}$ contenant toutes les relations binaires réflexives satisfaisant conjointement (t), (s), (r) et (m) est non vide. Soit $\mathcal{P}_{(t,s,r,m)}$ l'intersection de toutes les relations de $\mathfrak{P}_{(t,s,r,m)}$, qui est le plus petit élément de $\mathfrak{P}_{(t,s,r,m)}$, comme la monotonie est aussi stable par intersection. $\mathfrak{P}_{(t,s,r,m)}$ est un sous-ensemble (strict) de $\mathfrak{P}_{(t,s,r)}$, ainsi $\mathcal{P}_{(t,s,r,m)}$ raffine $\mathcal{P}_{(t,s,r)}$: ajouter une nouvelle propriété réduit le spectre de mondes possibles et réduit la possibilité de trouver un contre-argument à une préférence, permettant donc d'augmenter le nombre de décision sceptiques.

Déduction. Prendre en compte (m) dans le système déductif est simple : il suffit de considérer les dons $\mathcal{G}$ comme vérités premières, en plus des réarrangements $\mathcal{S}$ et des transferts $\mathcal{T}$. La clôture déductive par T est notée $cl_T(\mathcal{S} \cup \mathcal{T} \cup \mathcal{G})$.

Explications. Nous gardons la structure des explications anonymes et transitives, et augmentons leur pouvoir expressif en ajoutant les éléments de l'ensemble $\mathcal{G}$ des dons en plus des transferts redistributifs de l'ensemble $\mathcal{T}$.

Résultats structurels. Comme la dominance de Lorenz généralisée $\mathcal{L}$ appartient à $\mathfrak{P}_{(t,s,r,m)}$, elle raffine $\mathcal{P}_{(t,s,r,m)}$. $cl_T(\mathcal{S} \cup \mathcal{T} \cup \mathcal{G})$ est clairement correct et complet par rapport à $\mathcal{P}_{(t,s,r,m)}$. Les $(\mathcal{T} \cup \mathcal{G})$-ATX sont correctes *by design* vis à vis de $cl_T(\mathcal{S} \cup \mathcal{T} \cup \mathcal{G})$. Cette inclusion des relations de préférences se réduit grâce au résultat suivant.

Lemme 2 (Chong, 1976)**.** $\mathcal{L} \subseteq \bigcup_{\ell=1}^{n+1} \mathcal{E}(\mathcal{T} \cup \mathcal{G}\text{-}ATX^\ell)$

Ébauche de preuve. Il suffit d'appliquer un don en première ou en dernière position, de manière à avoir $\mathbf{x}^1$ et $\mathbf{y}$ avec le même revenu total, et ensuite de chercher une séquence de transferts redistributifs de $\mathbf{x}^1$ vers $\mathbf{y}$ via le Lemme 1. □

Théorème 3 (Explicabilité de la dominance de Lorenz généralisée)**.**

$$\bigcup_{\ell=1}^{n+1} \mathcal{E}(\mathcal{T} \cup \mathcal{G}\text{-}ATX^\ell) = cl_T(\mathcal{S} \cup \mathcal{T} \cup \mathcal{G}) = \mathcal{P}_{(t,s,r,m)} = \mathcal{L}$$

9. *L'espace d'états* est $\widehat{\mathbb{X}^n}$, les états *initial, objectif* et *actuels* sont respectivement $\widehat{\mathbf{x}}, \widehat{\mathbf{y}}$ et $\mathbf{x}^k$, et les *actions* sont les vérités premières.

10. Dans certains contextes, cette notion est connue pour être aux antipodes de l'équité [12].

	Long. Moy.			% d'égal./vict./déf. entre les longueurs des méthodes					Temps Moy. (s)			Time-out
n	◊	□	△*	◊ = □	◊ < □	◊ > □	□ = △*	□ > △*	◊	□	△	△ %
5	3.93	3.92	3.92	99	0	1	100	0	10^{-3}	10^{-3}	.15	0
10	8.36	8.21	8.05	83	1	16	59	41	10^{-3}	10^{-3}	16.65	1.4
20	14.65	13.81	13.71	34	0	66	90	10	10^{-3}	10^{-3}	139.71	89
50	26.79	24.98	24.98	12	2	86	100	0	10^{-3}	10^{-3}	150	100

◊ Algorithme HLP □ Notre algorithme △ Optimum

* la valeur de l'heuristique est choisie comme référence si le temps de calcul de l'optimum a dépassé le timeout (150s par explication).

TABLE 1 – Comparaison entre notre heuristique et l'algorithme HLP pour $\mathcal{L}^*$

Aspects computationnels. Du point de vue théorique, le problème de trouver des explications courtes pour la dominance de Lorenz généralisée est au moins aussi dur que pour la dominance restreinte. Curieusement, du point de vue algorithmique, l'heuristique découlant du Lemme 2, c.-à-d. systématiquement se réduire au problème avec un revenu total égal par un don initial (ou final) est sous-optimal, comme le montre l'exemple suivant.

Exemple 3. *Prenons les alternatives* $\widehat{\mathbf{d}}$ *et* $\widehat{\mathbf{c}}$ *de l'exemple 1. Calculer leurs cumulées montre que* $(\mathbf{d}, \mathbf{c}) \in \mathcal{L}$*. L'explication la plus courte supportant* $(\mathbf{d}, \mathbf{c})$ *est de taille 3 :*

$$\widehat{\mathbf{d}}\ \tau^{12}_{4\to 3}\ (6\ 17\ \overline{30}\ \underline{76}\ 96)\ \mathcal{G}\ (\overline{8}\ \overline{23}\ \overline{34}\ 76\ 96)\ \tau^{14}_{5\to 1}\ \widehat{\mathbf{c}}$$

Elle est strictement plus courte que toutes les explications plaçant le don en première ou dernière position.

4 Expliquer les décisions des *OWA* équitables et robustes

Bien qu'étant des relations de préférences équitables fondamentales, les dominances de Lorenz restent très indécises. Dans une situation de prise de décision [13] nous avons besoin d'une structure de préférences plus résolue, par exemple pour faire un choix (sélectionner une alternative favorite) ou donner un classement des alternatives.

De plus, la dominance de Lorenz est par définition non paramétrée, mais il peut être utile de considérer des raffinement paramétrés, capturant des motifs de préférences plus spécifiques tout en permettant des explications simples. Ensuite, nous allons complémenter les principes normatives avec *de l'information préférentielle*, permettant à la fois de restreindre l'ensemble des mondes possibles et d'augmenter la base du raisonnement et des explications.

4.1 Préférences basées sur un ensemble d'*OWA*

Les préférences représentées par la fonction de score nommée *Ordered weighted averaging* proviennent de [38] dans le contexte des indices d'inégalité et de [39] dans le contexte de la décision multi-critères. L'*OWA* est paramétrisée par un n-uplet $\boldsymbol{\omega}$ et associe à l'alternative $\mathbf{x}$ au score $\sum_{i\in[n]} \boldsymbol{\omega}_i \widehat{\mathbf{x}}_i$, qui est une somme pondérée ordonnée.

Le même modèle est parfois appelé *generalized Gini index* (*GGI*), comme une valeur spécifique du paramètre $\boldsymbol{\omega}$ produit l'indice de Gini. Nous donnons une définition basée sur la structure de préférence plutôt que sur le score, qui représente intrinsèquement l'inférence sceptique sur un ensemble Ω de valeurs du paramètre qui représente l'information préférentielle incomplète.

Définition 6 (Préférences basées sur les *OWA* robustes)**.** Soit Ω un sous-ensemble non vide de la sphère unité L_1 [11] de $\mathbb{R}^n$ et soit O_{WA_Ω} la relation binaire définie par $(\mathbf{x}, \mathbf{y}) \in O_{\mathrm{WA}_\Omega}$ ssi $\sum_{i\in[n]} \boldsymbol{\omega}_i \widehat{\mathbf{x}}_i \leq \sum_{i\in[n]} \boldsymbol{\omega}_i \widehat{\mathbf{y}}_i$ pour tout $\boldsymbol{\omega} \in \Omega$.

4.2 Propriétés des préférences basées sur les *OWA*

Nous observons que $O_{\mathrm{WA}_{\{\boldsymbol{\omega}\}}}$ est :
- — réflexive, transitive et symétrique qu'importe $\boldsymbol{\omega}$;
- — monotone quand toutes les valeurs de $\boldsymbol{\omega}$ sont non négative, reflétant l'attrait de tous les critères ;
- — redistributive quand les valeurs de $\boldsymbol{\omega}$ sont décroissantes : les agents les moins satisfaits sont plus importants ;

Soit [12] $\Omega^{\emptyset}$ l'ensemble de tous les n-uplets non-négatifs, décroissants, non-nuls de la sphère unité de $\mathbb{R}^n$.

Lemme 3 (Argyris et al. 2022)**.** $\mathcal{L} = O_{\mathrm{WA}_{\Omega^{\emptyset}}}$

De plus, plusieurs caractérisations des préférences basées sur des *OWA* ont été proposées (par exemple [38, 6, 32]). Elles diffèrent légèrement, mais elles requièrent toutes que la relation soit décisive et continue d'une certaine façon pour assurer qu'elle soit représentée par une fonction de valeur réelles, et impose une condition pour assurer que cette fonction est additive sur l'ensemble $\widehat{\mathbb{X}^n}$. Nous détaillons les résultats obtenus par Ben-Porath et Gilboa.

Propriété (d). $\mathcal{R}$ est *décisive* quand pour toutes alternatives $\mathbf{x}, \mathbf{y}$, $(\mathbf{x}, \mathbf{y})$ ou $(\mathbf{y}, \mathbf{x})$ (ou les deux, où dans ce cas $\mathbf{x}$ et $\mathbf{y}$ sont considérées aussi désirable l'un que l'autre) appartiennent à $\mathcal{R}$.

11. Cette condition assure la non trivialité des préférences (vu que le paramètre nul correspond à l'indifférence totale) et la non redondance des paramètres (comme une transformation linéaire des paramètres induit la même relation), tandis que le choix de la norme L_1 assure la calculabilité.
12. Cette notation est consistante avec la Def. 8.

$O\text{WA}_\Omega$ est décisive ssi Ω est un singleton.

Propriété (c). $\mathcal{R}$ est *continue* quand, pour toute alternative $\mathbf{z}$, les ensembles $\{\mathbf{x} : (\mathbf{x}, \mathbf{z}) \in \mathcal{R}\}$ et $\{\mathbf{y} : (\mathbf{z}, \mathbf{y}) \in \mathcal{R}\}$ sont fermés.

Propriété (i). $\mathcal{R}$ est *invariante* (par rapport aux dons préservant l'ordre social) quand, pour toutes alternatives $\mathbf{x}, \mathbf{x}', \mathbf{y}, \mathbf{y}'$, s'il existe un agent $i \in [n]$ et $t \in \mathbb{R}$ t.q. $\widehat{\mathbf{x}}' = \widehat{\mathbf{x}} + t\mathbf{e}^i$ et $\widehat{\mathbf{y}}' = \widehat{\mathbf{y}} + t\mathbf{e}^i$, alors $(\widehat{\mathbf{x}}, \widehat{\mathbf{y}}) \in \mathcal{R} \iff (\widehat{\mathbf{x}}', \widehat{\mathbf{y}}') \in \mathcal{R}$.

En conjonction à (t), la propriété (i) induit une propriété clé des relations linéaires : la capacité de raisonner *ceteris paribus*– c.-à-d. toutes choses étant égales par ailleurs. La préférence de $\mathbf{y}$ sur $\mathbf{x}$ dépend uniquement de l'acceptabilité du *compromis* $\mathbf{y} - \mathbf{x}$, peu importe qu'il modifie $\mathbf{x}$ ou un autre $\mathbf{x}' \in \widehat{\mathbb{X}^n}$ (tant que $\mathbf{y}' := \mathbf{x}' + (\mathbf{y} - \mathbf{x})$ appartient à $\widehat{\mathbb{X}^n}$).

Définition 7 (Équivalence *ceteris paribus*)**.** Deux paires comparatives $(\mathbf{x}, \mathbf{y})$ et $(\mathbf{x}', \mathbf{y}')$ sont équivalentes *ceteris paribus* quand les alternatives $\mathbf{x}, \mathbf{x}', \mathbf{y}, \mathbf{y}'$, appartiennent à $\widehat{\mathbb{X}^n}$ et les vecteurs $\mathbf{x} - \mathbf{y}$ et $\mathbf{x}' - \mathbf{y}'$ sont égaux. Dans ce cas, $(\mathbf{x}, \mathbf{y})$ et $(\mathbf{x}', \mathbf{y}')$ représentent le même *compromis*.

Quand une relation de préférence $\mathcal{R}$ vérifie (t) et (i), elle est compatible à l'équivalence *ceteris paribus*, dans le sens où deux paires équivalentes sont soit toutes les deux dans $\mathcal{R}$, soit aucune. Ainsi, $\mathcal{R}$ peut être définie par l'ensemble des *compromis acceptables* $to(\mathcal{R}) := \{\widehat{\mathbf{x}} - \widehat{\mathbf{y}}, (\widehat{\mathbf{x}}, \widehat{\mathbf{y}}) \in \mathcal{R}\}$.

Lemme 4 (Ben-Porath et Gilboa, 1995)**.** *Une relation binaire réflexive $\mathcal{R}$ satisfait (t), (s), (r), (m), (d), (c) et (i) ssi il existe un n-uplet $\boldsymbol{\omega}$ de réels non négatifs et décroissants tel que $\mathcal{R} = O\text{WA}_{\{\boldsymbol{\omega}\}}$.*

C'est un théorème de représentation très puissant, cependant ses propriétés sont contradictoire vis à vis de notre programme de représenter le raisonnement sous-jacent par des règles déductives.

4.3 Au-delà de la décisivité

La propriété (d) est peu satisfaisante pour deux raisons. D'un point de vue raisonnement, elle équivaut à introduire la règle du *tiers exclus* dans notre système formel, ce qui permet d'introduire les preuves par réfutation [13] qui sont un outil puissant de déduction. Cependant, cela ne s'aligne pas avec notre besoin d'explications intelligibles, qui semble mieux correspondre à la logique intuitionniste. Du point de vue de la représentation de préférences, être décisif entre en conflit avec le besoin de capturer une information nécessairement incomplète. Un bon indicateur de la fragilité des décisions prises sous cette condition est que cette propriété est la seule dans cet article à ne pas être stable par intersection. Afin de proposer graduellement une transition entre la dominance de Lorenz généralisée (obtenue quand $\Omega = \Omega^\emptyset$) et une relation décisive (obtenue quand Ω est un singleton), nous adoptons le paradigme de *l'apprentissage robuste des préférences* [21, 20]. Supposons que l'on ait accès à de *l'information préférentielle* $\mathcal{I}$ sous la forme d'une relation binaire sur des alternatives, c.-à-d. un ensemble de référence de paires comparatives qui doivent être vérifiées par la relation de préférences recherchée.

Propriété ($\mathbf{pi}^\mathcal{I}$)**.** Avec $\mathcal{I}$ une relation binaire sur des alternatives, $\mathcal{R}$ est compatible avec *l'information préférentielle* $\mathcal{I}$ quand $\mathcal{R} \supseteq \mathcal{I}$.

Du point de vue de la déduction, la compatibilité avec l'information préférentielle est obtenue naturellement en considérant les paires de $\mathcal{I}$ comme évidentes. Du point de vue de la représentation numérique, nous devons considérer l'ensemble $\Omega^\mathcal{I}$ contenant exactement tous les paramètres tels que $O\text{WA}_{\Omega^\mathcal{I}}$ est compatible avec $\mathcal{I}$.

Définition 8 (ensemble des paramètres compatibles)**.** Avec $\mathcal{I}$ une relation binaire sur des alternatives, soit $\Omega^\mathcal{I}$ l'ensemble de tous les n-uplets $\boldsymbol{\omega}$ non-négatifs, décroissants, non-nuls de la sphère unité de $\mathbb{R}^n$ t.q. $O\text{WA}_{\{\boldsymbol{\omega}\}} \supseteq \mathcal{I}$. Quand $\Omega^\mathcal{I} \neq \emptyset$, $\mathcal{I}$ est dite *consistante* (avec l'*OWA* équitable).

L'inférence sceptique est souvent confronté à des problèmes computationnels [18], mais dans notre cas elle reste polynomiale, vu que l'*OWA* est linéaire par rapport à $\boldsymbol{\omega}$.

Lemme 5 (inspiré par [21])**.** *Étant donné une relation binaire sur des alternatives $\mathcal{I}$, l'ensemble $\Omega^\mathcal{I}$ est le polytope de $\mathbb{R}^n$ défini par les contraintes linéaires sur la variable $\boldsymbol{\omega}$: $\omega_i \geq 0$ pour tout $i \in [n]$; $\omega_i - \omega_{i+1} \geq 0$ pour tout $i \in [n-1]$; $\sum_{i\in[n]} \omega_i = 1$ et $\sum_{i\in[n]} (\widehat{\mathbf{b}}_i - \widehat{\mathbf{a}}_i)\omega_i \geq 0$ pour tout $(\mathbf{a}, \mathbf{b}) \in \mathcal{I}$. De plus, vérifier si $\mathcal{I}$ est consistante, ou si une paire comparative est dans $O\text{WA}_{\Omega^\mathcal{I}}$ se réduit à un problème d'optimisation linéaire soluble en temps polynomial.*

Exemple 4. *Considérons l'information préférentielle $(\mathbf{b}, \mathbf{d})$ donnée par l'autorité centrale dans l'exemple 1. Le score d'OWA paramétré par $\boldsymbol{\omega} \in \Omega^\mathcal{I}$ donné à l'alternative $\mathbf{d}$ doit être plus quand que celui donné à $\mathbf{b}$. Ainsi $\widehat{\mathbf{d}} \cdot \boldsymbol{\omega} \geq \widehat{\mathbf{b}} \cdot \boldsymbol{\omega}$, ou de manière équivalente $(\widehat{\mathbf{d}} - \widehat{\mathbf{b}}) \cdot \boldsymbol{\omega} \geq 0$. Cette contrainte peut être interprétée comme le trade-off $(\widehat{\mathbf{d}} - \widehat{\mathbf{b}}) = (+4, -8, -10, +4, -2)$ étant désirable.*

4.4 Caractérisation des préférences d'un *OWA* robuste

Notre prochain objectif est de caractériser la structure de préférences induite par l'ensemble d'*OWA* à l'aide de propriétés actionables amenant à un moteur explicatif correct et complet. Nous aimerions garder (t), (s), (r), (m) et (i), mais retirer (d), (c). Nous avons déjà commenté (d), que nous souhaiterions remplacer par ($\text{pi}^\mathcal{I}$), et bien que (c) soit un outil mathématique fantastique, elle est inutilisable du point de vue cognitif. Elle permet de prendre la limite

13. En effet, pour prouver que $\mathbf{y}$ est préférée à $\mathbf{x}$, il suffit de prouver que $\mathbf{x}$ ne peut pas être préféré à $\mathbf{y}$.

à gauche et à droite dans une séquence de paires comparatives mais comment définir de telles séquences, vérifier leur convergence et calculer leur limites. Il sera difficile pour un non-expert de comprendre et de commenter une telle propriété.

Cependant, comme nous le verrons plus tard, (t), (s), (r), (m), (c), (i) et (pi$^{\mathcal{I}}$) ne sont pas suffisantes pour caractériser les *OWA* équitables et robustes. C'est pourquoi nous introduisons une nouvelle notion, proche de celle de l'équivalence *ceteris paribus*, mais nettement plus puissante.

En partant de l'équivalence *ceteris paribus*, nous souhaitons incorporer la symétrie et relâcher l'égalité entre les compromis par l'existence d'un lien non négatif, nous amenant à la propriété plus forte suivante

Définition 9 (congruence entre paires comparatives)**.** Deux paires comparatives $(\mathbf{x}, \mathbf{y})$ et $(\mathbf{x}', \mathbf{y}')$ sont congruentes quand $\widehat{\mathbf{x}} - \widehat{\mathbf{y}}$ et $\widehat{\mathbf{x}}' - \widehat{\mathbf{y}}'$ sont non-négativement liées [14]. Dans ce cas, nous écrivons $(\mathbf{x}, \mathbf{y}) \equiv (\mathbf{x}', \mathbf{y}')$.

Propriété (cc). $\mathcal{R}$ est compatible avec la *congruence* quand, pour toutes alternatives $\mathbf{x}, \mathbf{x}', \mathbf{y}, \mathbf{y}'$, si $(\mathbf{x}, \mathbf{y}) \equiv (\mathbf{x}', \mathbf{y}')$ alors $(\mathbf{x}, \mathbf{y}) \in \mathcal{R} \iff (\mathbf{x}', \mathbf{y}') \in \mathcal{R}$.

Il est à noter que la propriété (cc) implique (i) et (s). Cela joue un rôle important dans la caractérisation des *OWA* décisifs [15]. Nous pouvons maintenant nous demander si cette nouvelle structure des compromis acceptables peut être dérivée, ou est une conséquence logique, des propriétés (t), (s), (r), (m), (c), (i) et (pi$^{\mathcal{I}}$). La réponse à ces questions est non.

Théorème 4. *Quand $\Omega^{\mathcal{I}}$ n'est pas un singleton, la propriété (cc) ne peut être déduite de (t), (s), (r), (m), (c), (i) et (pi$^{\mathcal{I}}$).*

Contre-exemple. Soit $\Gamma := \{(z_1, z_2, z_3) \in \mathbb{R}^3$ vérifiant (1) $3z_1 + 2z_2 + z_3 \geq 3$ ou (2) $z_1 \geq 0$ et $z_1 + z_2 \geq 0$ et $z_1 + z_2 + z_3 \geq 0\}$, et $\mathcal{R}$ la relation binaire sur $\mathbb{R}^3$ définie par $(\mathbf{x}, \mathbf{y}) \in \mathcal{R}$ ssi $\widehat{\mathbf{y}} - \widehat{\mathbf{x}} \in \Gamma$. $\mathcal{R}$ satisfait (i) et (s) par construction. Elle est continue car l'ensemble Γ de compromis acceptables est fermé (comme l'union de l'intersection de préimages d'ensembles fermés par des fonctions continues). Elle est transitive car Γ est stable par rapport à l'addition (la somme de deux vecteurs vérifiant (1) ou (2) vérifient respectivement (1) or (2), et la somme d'un vérifiant (1) et de l'autre (2) vérifie (2)). (r) et (m) sont obtenues par la condition (2). Cependant, bien que le compromis $t_1 := (2, -4, 6)$ est acceptable (correspondant par exemple à la paire comparative $(0, 8, 10)$ contre $(2, 4, 16)$), le compromis $\frac{1}{2}t_1 = (1, -2, 3)$ ne l'est pas (alors qu'il correspond par exemple à la paire $(3, 8, 9)$ contre $(4, 6, 12)$). □

14. c.-à-.d au moins un des vecteurs est nul, ou il existe $\lambda > 0$ t.q. $(\widehat{\mathbf{x}} - \widehat{\mathbf{y}}) = \lambda(\widehat{\mathbf{x}}' - \widehat{\mathbf{y}}')$.

15. Une étape importante de la preuve établie l'équation quand le coefficient λ est l'inverse d'un entier positif, en raisonnant par transitivité et le tiers exclus.

Comme corollaire [16], les propriétés (t), (s), (r), (m), (c), (i) et (pi$^{\mathcal{I}}$) ne sont pas suffisantes pour caractériser les préférences d'*OWA* robuste $O_{\text{WA}_{\Omega^{\mathcal{I}}}}$. Nous introduisons une nouvelle règle d'inférence

$$\textbf{Règle CC :}\ \frac{(\mathbf{a}, \mathbf{b}), (\mathbf{c}, \mathbf{d}) \equiv (\mathbf{a}, \mathbf{b})}{(\mathbf{c}, \mathbf{d})}\ \text{(compatibilité à la congruence)}$$

correspondant à la propriété (cc). Nous pouvons donc introduire notre résultat principal.

Théorème 5. *Pour toute information préférentielle $\mathcal{I}$ consistante avec l'OWA :*

$$cl_{T,CC}(\mathcal{T} \cup \mathcal{G} \cup \mathcal{I}) = \mathcal{P}_{(t,r,m,cc,pi^{\mathcal{I}})} = O_{\text{WA}_{\Omega^{\mathcal{I}}}}$$

Le théorème 5 caractérise sémantiquement et déductivement les préférences basées sur un *OWA* équitable et robuste : $O_{\text{WA}_{\Omega^{\mathcal{I}}}}$ est la seule relation binaire réflexive satisfaisant (t), (s), (r), (m), (cc) et (pi$^{\mathcal{I}}$); de plus $\mathbf{x}$ est moins préféré que $\mathbf{y}$ selon cette relation si, et seulement si, il existe une preuve établissant cette préférence en utilisant uniquement les règles déductives T et CC et les vérités de $\mathcal{T}$, $\mathcal{G}$ ou $\mathcal{I}$. Comme décrit en section 3 dans le cas de la dominance de Lorenz, la chaîne d'inclusions de la gauche vers la droite dénote la correction, c.-à-d. $cl_{T,CC}(\mathcal{T} \cup \mathcal{G} \cup \mathcal{I}) \subseteq \mathcal{P}_{(t,r,m,cc,pi^{\mathcal{I}})} \subseteq O_{\text{WA}_{\Omega^{\mathcal{I}}}}$ est structurellement valide car les propriétés mises en avant correspondent à celles de l'OWA robuste et des règles. Nous allons maintenant concevoir un moteur explicatif implémentant une restriction de $cl_{T,CC}(\mathcal{T} \cup \mathcal{G} \cup \mathcal{I})$ et *complet* par rapport à $O_{\text{WA}_{\Omega^{\mathcal{I}}}}$, ce qui cloturera la preuve du théorème 5.

4.5 Moteurs explicatifs

L'exemple 1 illustrait le cas d'une paire comparative expliquée par une ATX basée sur les vérités premières de $\mathcal{S}$, $\mathcal{G}$, $\mathcal{T}$ et de $cl_{CC}(\mathcal{I})$ –les compromis qui sont congruents à un de ceux présents dans l'information préférentielle– mais ce moteur explicatif n'est peut être pas complet par rapport à $O_{\text{WA}_{\Omega^{\mathcal{I}}}}$, et est clairement difficilement traitable computationnellement car la contrainte de rester dans $\widehat{\mathbb{X}^n}$ est difficile à satisfaire, surtout quand les deux alternatives comparées sont proches du bord.

Nous proposons donc de relâcher le besoin de chercher un chemin de $\widehat{\mathbf{x}}$ vers $\widehat{\mathbf{y}}$ en la recherche d'un chemin de $\mathbf{x}'$ vers $\mathbf{y}'$ avec $(\mathbf{x}', \mathbf{y}')$ congruent à $(\mathbf{x}, \mathbf{y})$.

Définition 10 (Moteur CTX)**.** Soit $\mathcal{R}$ une relation binaire sur $\widehat{\mathbb{X}^n}$. Une *explication congruente et transitive basée sur les vérités dans $\mathcal{R}$* de longueur ℓ ($\mathcal{R}$-CTX$^{\ell}$) est une paire (s, c) où le *support* s est un ℓ-uplet de paires comparatives $s = ((\mathbf{x}^1, \mathbf{y}^1), \ldots, (\mathbf{x}^{\ell}, \mathbf{y}^{\ell})) \in cl_{CC}(\mathcal{R})^{\ell}$ et *l'affirmation* c est une paire comparative $c = (\mathbf{x}, \mathbf{y})$ vérifiant $(\mathbf{x}^1, \mathbf{y}^{\ell}) \equiv (\mathbf{x}, \mathbf{y})$ et, pour tout entier k entre 2 et ℓ, $\mathbf{x}^k = \mathbf{y}^{k-1}$.

16. Le contre-exemple se focalise sur la famille de compromis $\{\delta : \omega \cdot \delta \geq K\}$, avec $\omega = (3, 2, 1)$, mais peut être modifié pour incorporer toute information préférentielle consistante mais non décisive.

Comme elles n'utilisent que des règles déductives et des vérités premières, il est clair que les CTX sont correctes par rapport aux règles déductives T et CC. Nous établissons un autre résultat principal, la complétude du moteur à l'aide d'une preuve constructive qui peut être appliquée par un algorithme efficace. Il conclut la preuve du théorème 5 (l'inclusion dans l'autre sens est obtenue via la correction).

Théorème 6. $O_{WA_{\Omega^{\mathcal{I}}}} \subseteq \bigcup_{\ell=1}^{n+|\mathcal{I}|} \mathcal{E}(\mathcal{T} \cup \mathcal{G} \cup \mathcal{I}\text{-}CTX^{\ell})$

Démonstration. Soit $(\mathbf{x}, \mathbf{y}) \in O_{WA_{\Omega^{\mathcal{I}}}}$. Par le lemme de Farkas appliqué au programme linéaire du Lemme 5, il existe des coefficients non-négatifs $\langle \lambda^*_{(\mathbf{a},\mathbf{b})} \rangle_{(\mathbf{a},\mathbf{b})\in\mathcal{I}}$, $\langle \mu^*_i \rangle_{i\in[n]}$ et $\langle \nu^*_{i,j} \rangle_{1\le i<j\le n}$ t.q.

$$\widehat{\mathbf{y}} - \widehat{\mathbf{x}} = \sum_{(\mathbf{a},\mathbf{b})\in\mathcal{I}} \lambda^*_{(\mathbf{a},\mathbf{b})}(\widehat{\mathbf{b}} - \widehat{\mathbf{a}}) + \sum_{i\in[n]} \mu^*_i \mathbf{e}^i + \sum_{i<j} \nu^*_{i,j}(\mathbf{e}^i - \mathbf{e}^j)$$

Cette équation décompose additivement le compromis correspondant à la paire comparative en 3 termes, chacun étant la somme de compromis correspondant aux vérités premières de $cl_{CC}(\mathcal{I})$, $\mathcal{G}$ et $\mathcal{T}$. Pour chaque agent i, nous séparons cette équation en deux parties, une contenant la somme des valeurs positives Δ^+_i et l'autre des valeurs négatives Δ^-_i, t.q. $\widehat{\mathbf{y}}_i - \widehat{\mathbf{x}}_i = \Delta^+_i + \Delta^-_i$. Si nous définissons $\widehat{\mathbf{x}}'$ et $\widehat{\mathbf{y}}'$ t.q. $\forall i \in [n] : \widehat{\mathbf{x}}'_i = \widehat{\mathbf{x}}_i + \sum_{j=1}^{i} \Delta^+_{j-1} - \sum_{j=1}^{i} \Delta^-_j$ et $\widehat{\mathbf{y}}'_i = \widehat{\mathbf{y}}_i + \sum_{j=1}^{i} \Delta^+_{j-1} - \sum_{j=1}^{i} \Delta^-_j$, nous avons *(i)* $\widehat{\mathbf{y}}_i - \widehat{\mathbf{x}}_i = \widehat{\mathbf{y}}'_i - \widehat{\mathbf{x}}'_i$, donc $(\mathbf{x}, \mathbf{y})$ et $(\mathbf{x}', \mathbf{y}')$ sont congruents ; et *(ii)* $\forall i \in [n]\ \widehat{\mathbf{x}}'_i - \widehat{\mathbf{x}}'_{i-1} \ge \Delta^+_{i-1} - \Delta^-_i$, ainsi la séparation entre les critères permet $\widehat{\mathbf{x}}'_{i-1}$ d'être augmenté et $\widehat{\mathbf{x}}'_i$ d'être réduit par les compromis donnés par le certificat de Farkas tout en vérifiant $\widehat{\mathbf{x}}'_{i-1} \le \widehat{\mathbf{x}}'_i$. Ainsi il est possible d'appliquer dans n'importe quel ordre les compromis de $\mathcal{T}$, $\mathcal{G}$ et $cl_{CC}(\mathcal{I})$ décrits par $\langle \lambda^*_{(\mathbf{a},\mathbf{b})} \rangle_{(\mathbf{a},\mathbf{b})\in\mathcal{I}}$, $\langle \mu^*_i \rangle_{i\in[n]}$ et $\langle \nu^*_{i,j} \rangle_{1\le i<j\le n}$ pour construire la chaîne transitive entre $\mathbf{x}'$ et $\mathbf{y}'$. Il en résulte que l'on peut toujours trouver une $(\mathcal{T} \cup \mathcal{G} \cup \mathcal{I})$-CTX de taille au plus $|\mathcal{I}| + n$ en temps polynomial. □

L'utilisation d'un certificat d'infaisabilité pour obtenir une explication peut aussi être trouvé dans [23, 3]. Obtenir ce certificat avec les résultats de la dualité forte (ici, le lemme de Farkas) peut aussi être trouvé dans [25, 4, 5].

Exemple 5. *Nous proposons de calculer une CTX, tel qu'expliqué dans la preuve du Théorème 6 pour la paire comparative* $(\mathbf{a}, \mathbf{c})$*, avec* $\widehat{\mathbf{c}} = (22\ 23\ 34\ 76\ 82)$ *et* $\widehat{\mathbf{a}} = (16\ 31\ 51\ 70\ 83)$*. Le certificat de Farkas obtenu est :* $\lambda^*_{(\mathbf{b},\mathbf{d})} = 1.5$*,* $\mu^* = (0\ 0\ 2\ 0\ 2)$ *et un unique transfert redistributif* $\nu^*_{3,2} = 4$*. Nous pouvons nous assurer de sa validité en calculant* $\widehat{\mathbf{c}} - \widehat{\mathbf{a}} = (+6, -8, -17, +6, -1) = 1.5 * (+4, -8, -10, +4, -2) + (0, 0, +2, 0, +2) + (0, +4, -4, 0, 0)$*. Nous générons les alternatives* $\mathbf{x}'$ *et* $\mathbf{y}'$ *comme décrit dans la preuve. Pour l'agent 1, nous avons* $\Delta^+_1 = 6$ *et* $\Delta^-_1 = 0$*, ainsi* $\mathbf{y}'_1 = \widehat{\mathbf{c}}_1 - \Delta^-_1 = \widehat{\mathbf{c}}_1 = 22$ *et* $\mathbf{x}'_1 = \widehat{\mathbf{a}}_1 - \Delta^-_1 = \widehat{\mathbf{a}}_1 = 16$*. Pour l'agent 2, nous avons* $\Delta^+_2 = 4$ *(du transfert redistributif) et* $\Delta^-_2 = -12$ *(issu de la* $\mathcal{I}$*-congruence), ainsi* $\mathbf{y}'_2 = \widehat{\mathbf{c}}_2 + \Delta^+_1 - \Delta^-_2 - \Delta^-_1 = 23 + 6 - (-12) = 41$ *et* $\mathbf{x}'_2 = \widehat{\mathbf{a}}_2 + \Delta^+_1 - \Delta^-_2 - \Delta^-_1 = 31 + 6 - (-12) = 49$*. Pour l'agent agent 3, nous avons* $\Delta^+_3 = -15 - 4 = -19$ *(du transfert redistributif et de la* $\mathcal{I}$*-congruence) et* $\Delta^-_3 = 20$ *(du don), ainsi* $\mathbf{y}'_3 = \widehat{\mathbf{c}}_3 + \Delta^+_2 + \Delta^+_1 - \Delta^-_3 - \Delta^-_2 - \Delta^-_1 = 34 + 4 + 6 - (-19) - (-12) = 75$ *et* $\mathbf{x}'_3 = \widehat{\mathbf{a}}_3 + \Delta^+_2 + \Delta^+_1 - \Delta^-_3 - \Delta^-_2 - \Delta^-_1 = 51 + 4 + 6 - (-19) - (-12) = 92$*. Nous continuons pour les agents 4 et 5 et obtenons ainsi* $\mathbf{y}' = (22\ 41\ 75\ 119\ 134)$ *et* $\mathbf{x}' = (16\ 49\ 92\ 113\ 135)$*. Nous avons bien* $\mathbf{y}' - \mathbf{x}' = \widehat{\mathbf{c}} - \widehat{\mathbf{a}}$*, donc les deux paires sont congruentes, nous pouvons donc construire la CTX de longueur 3 à partir de la chaîne d'alternatives* $(\mathbf{x}^1, \mathbf{x}^2, \mathbf{x}^3, \mathbf{x}^4)$ *avec* $\mathbf{x}^1 := \mathbf{x}'$ *et* $\mathbf{x}^4 := \mathbf{y}'$*. Toutes les permutations du transfert redistributif, du don et de la PI-congruence sont possibles, si nous optons pour cet ordre-ci nous avons* $\mathbf{x}^2 := (16\ 53\ 88\ 113\ 135)$ *and* $\mathbf{x}^3 := (22\ 41\ 75\ 119\ 134)$.

Nous pouvons nous demander si une ATX, plutôt qu'une CTX, peut être trouvée pour expliquer une affirmation appartenant à $O_{WA_{\Omega^{\mathcal{I}}}}$. C'est en effet possible sous certaines conditions. Soit $\mathbb{F} := \{\mathbf{z} \in \widehat{\mathbb{X}^n} : \exists i\ \mathbf{z}_i = 0 \text{ ou } \exists j \neq i\ \mathbf{z}_i = \mathbf{z}_j\}$. $\mathbb{F}$ est la frontière de $\widehat{\mathbb{X}^n}$.

Théorème 7. *Soit* $\mathcal{I}$ *une relation binaire sur des alternatives consistante avec l'OWA et* $(\mathbf{x}, \mathbf{y}) \in O_{WA_{\Omega^{\mathcal{I}}}}$*. Si :*

i. aucune alternative $\mathbf{x}$ *ou* $\mathbf{y}$ *n'appartient à* $\mathbb{F}$ *; ou*

ii. $\sup_{\omega\in\Omega^{\mathcal{I}}} \boldsymbol{\omega} \cdot (\widehat{\mathbf{y}} - \widehat{\mathbf{x}}) > 0$

alors il existe une $[\mathcal{T} \cup \mathcal{G} \cup cl_{CC}(\mathcal{I})]$*-ATX supportant la conclusion* $(\mathbf{x}, \mathbf{y})$.

Cependant, notre preuve réside sur la construction d'une ATX de taille non bornée. Nous supposons donc l'existence de paires comparatives qui ne sont pas explicables par une ATX.

5 Conclusion et perspectives

Nous avons définis un moteur explicatif correct et complet pour les *OWA* robustes en utilisant un modèle logique formel. Nous ne sommes pas les premiers à suivre une telle approche, analogue à celle initiée par [15] et étendues par [33, 10, 30, 11]. Nous nous assurons que nos explications sont narrativement et cognitivement acceptables en évitant l'utilisation de propriétés purement techniques (comme la continuité), sous la forme d'une explication étape-par-étape, enchaînant les arguments, de taille bornée [9].

Nous pensons que nos résultats comblent une importante lacune : les *OWA* sont souvent utilisées pour des problèmes d'équités en optimisation combinatoire [31, 26], choix social computationnel [2, 27], apprentissage de préférences ou par renforcement [14, 17], il est raisonnable de penser que, quand l'équité est un aspect important, nous souhaitons aussi pouvoir examiner les décisions obtenues et éviter autant que faire se peut les biais non contrôlés ou les instabilités dues au choix de paramètres spécifiques. En donnant

des explications prouvablement correctes et lisibles nous pallions ce besoin. Nos explications peuvent ainsi permettre l'évaluation de la régularité de la procédure et l'adéquation des décisions algorithmiques, ou même le recours [24, 16].

Du point de vue IA de confiance, la prochaine évolution serait de soumettre les explications à l'approbation des propriétés sous-jacentes, avec des *questions critiques* [37] comme "est-ce raisonnable d'être symétrique ? (c'est l'anniversaire de Charlie)" ou "est-on sûrs que les utilités s'expriment sur une échelle commune ?". Cela requerrait d'intégrer le moteur explicatif dans un agent dialectique capable de raisonnements non monotones. Du point de vue théorie de la décision, la prochaine étape serait de transférer notre approche à des modèles plus complexes comme l'intégrale de Choquet, requérant un gros travail axiomatique.

Références

[1] Silvia Angilella, Salvatore Greco, and Benedetto Matarazzo. Non-additive robust ordinal regression : A multiple criteria decision model based on the Choquet integral. *European Journal of Operational Research*, 201(1) :277–288, 2010.

[2] Nikolaos Argyris, Özlem Karsu, and Mirel Yavuz. Fair resource allocation : Using welfare-based dominance constraints. *European journal of operational research*, 297(2) :560–578, 2022.

[3] Khaled Belahcène, Yann Chevaleyre, Christophe Labreuche, Nicolas Maudet, Vincent Mousseau, and Wassila Ouerdane. Accountable approval sorting. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI*, pages 70–76. ijcai.org, 2018.

[4] Khaled Belahcene, Christophe Labreuche, Nicolas Maudet, Vincent Mousseau, and Wassila Ouerdane. Explaining robust additive utility models by sequences of preference swaps. *Theory and Decision*, 82 :151–183, 2017. Number : 2 Publisher : Springer.

[5] Khaled Belahcène, Christophe Labreuche, Nicolas Maudet, Vincent Mousseau, and Wassila Ouerdane. Comparing options with argument schemes powered by cancellation. In Sarit Kraus, editor, *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI*, pages 1537–1543, 2019.

[6] Elchanan Ben-Porath and Itzhak Gilboa. Linear measures, the gini index and the income-equality tradeoff. *Journal of Economic Theory*, 64 :443–467, 1994.

[7] Nawal Benabbou, Christophe Gonzales, Patrice Perny, and Paolo Viappiani. Minimax regret approaches for preference elicitation with rank-dependent aggregators. *EURO journal on Decision processes*, 3 :29–64, 2015.

[8] Nawal Benabbou, Cassandre Leroy, Thibaut Lust, and Patrice Perny. Combining local search and elicitation for multi-objective combinatorial optimization. In *Algorithmic Decision Theory : 6th International Conference, ADT*, pages 1–16. Springer, 2019.

[9] Ignace Bleukx, Jo Devriendt, Emilio Gamba, Bart Bogaerts, and Tias Guns. Simplifying Step-Wise Explanation Sequences. In *29th International Conference on Principles and Practice of Constraint Programming (CP 2023)*, volume 280 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 11 :1–11 :20. Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2023.

[10] Arthur Boixel, Ulle Endriss, and Ronald de Haan. A calculus for computing structured justifications for election outcomes. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022*, pages 4859–4866. AAAI Press, 2022.

[11] Arthur Boixel, Ulle Endriss, and Oliviero Nardi. Displaying justifications for collective decisions. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI*, pages 5892–5895, 2022.

[12] Sylvain Bouveret, Yann Chevaleyre, and Nicolas Maudet. Fair allocation of indivisible goods. In Felix Brandt, Vincent Conitzer, Ulle Endriss, Jérôme Lang, and Ariel D. Procaccia, editors, *Handbook of Computational Social Choice*, pages 284–310. Cambridge University Press, 2016.

[13] Denis Bouyssou, Thierry Marchant, Marc Pirlot, Alexis Tsoukiàs, and Philippe Vincke. *Evaluation and decision models with multiple criteria : Stepping stones for the analyst*. International Series in Operations Research and Management Science, Volume 86. Springer, 2006.

[14] Róbert Busa-Fekete, Balázs Szörényi, Paul Weng, and Shie Mannor. Multi-objective bandits : Optimizing the generalized gini index. In *International Conference on Machine Learning*, pages 625–634. PMLR, 2017.

[15] Olivier Cailloux and Ulle Endriss. Arguing about voting rules. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, pages 287–295. ACM, 2016.

[16] Raja Chatila, Virginia Dignum, Michael Fisher, Fosca Giannotti, Katharina Morik, Stuart Russell, and Karen Yeung. Trustworthy AI. In Bertrand Braunschweig and Malik Ghallab, editors, *Reflections on Artificial Intelligence for Humanity*, volume 12600 of *Lecture Notes in Computer Science*, pages 13–39. Springer, 2021.

[17] Virginie Do, Sam Corbett-Davies, Jamal Atif, and Nicolas Usunier. Two-sided fairness in rankings via

lorenz dominance. *Advances in Neural Information Processing Systems*, 34 :8596–8608, 2021.

[18] Thomas Eiter and Georg Gottlob. On the complexity of propositional knowledge base revision, updates, and counterfactuals. *Artificial Intelligence*, 57(2-3), 1992.

[19] Michael R Garey and David S Johnson. Computers and intractability : a guide to the theory of np-hardness, 1979.

[20] Salvatore Greco, Benedetto Matarazzo, and Roman Slowinski. Rough sets theory for multicriteria decision analysis. *European Journal of Operational Research*, 129(1) :1–47, 2001.

[21] Salvatore Greco, Vincent Mousseau, and Roman Słowiński. Ordinal regression revisited : Multiple criteria ranking using a set of additive value functions. *European Journal of Operational Research*, 191(2) :416–436, 2008.

[22] G.H. Hardy, J.E. Littlewood, and G. Pólya. Some simple inequalities satisfied by convex functions. *Messenger of Mathematics*, 58 :145–152, 1929.

[23] Ulrich Junker. QUICKXPLAIN : preferred explanations and relaxations for over-constrained problems. In Deborah L. McGuinness and George Ferguson, editors, *Proceedings of the Nineteenth National Conference on Artificial Intelligence, Sixteenth Conference on Innovative Applications of Artificial Intelligence*, pages 167–172. AAAI Press / The MIT Press, 2004.

[24] Joshua A. Kroll, Joanna Huey, Solon Barocas, Edward W. Felten, Joel R. Reidenberg, David G. Robinson, and Harlan Yu. Accountable algorithms. *University of Pennsylvania Law Review*, 165(3) :633–705, 2017.

[25] Christophe Labreuche, Nicolas Maudet, and Wassila Ouerdane. Justifying dominating options when preferential information is incomplete. In *Proceedings of the 20th European Conference on Artificial Intelligence*, pages 486–491, 2012.

[26] Julien Lesca, Michel Minoux, and Patrice Perny. The fair owa one-to-one assignment problem : Np-hardness and polynomial time special cases. *algorithmica*, 81 :98–123, 2019.

[27] Jing Wu Lian, Nicholas Mattei, Renee Noble, and Toby Walsh. The conference paper assignment problem : Using order weighted averages to assign indivisible goods. In *proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[28] Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, pages 4765–4774, 2017.

[29] Radu Marinescu, Debarun Bhattacharjya, Junkyu Lee, Fabio Cozman, and Alexander Gray. Credal marginal map. In *Annual Conference on Neural Information Processing Systems*, 2023.

[30] Oliviero Nardi, Arthur Boixel, and Ulle Endriss. A graph-based algorithm for the automated justification of collective decisions. In *21st International Conference on Autonomous Agents and Multiagent Systems, AAMAS*, pages 935–943. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS), 2022.

[31] Wlodzimierz Ogryczak. Fair optimization–methodological foundations of fairness in network resource allocation. In *2014 IEEE 38th International Computer Software and Applications Conference Workshops*, pages 43–48. IEEE, 2014.

[32] Patrice Perny, Olivier Spanjaard, and Louis-Xavier Storme. A decision-theoretic approach to robust optimization in multivalued graphs. *Ann. Oper. Res.*, 147(1) :317–341, 2006.

[33] Dominik Peters, Ariel D. Procaccia, Alexandros Psomas, and Zixin Zhou. Explainable voting. In *Advances in Neural Information Processing Systems NeurIPS*, 2020.

[34] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should I trust you ?" : Explaining the predictions of any classifier. In *Proceedings of the Demonstrations Session, NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics : Human Language Technologies*, pages 97–101. The Association for Computational Linguistics.

[35] Teddy Seidenfeld, Mark J Schervish, and Joseph B Kadane. Coherent choice functions under uncertainty. *Synthese*, 172 :157–176, 2010.

[36] Anthony F. Shorrocks. Ranking Income Distributions. *Economica*, 50(197) :3–17, 1983.

[37] Douglas Walton, Chris Reed, and Fabrizio Macagno. *Argumentation Schemes*. Cambridge University Press, 2008.

[38] John A. Weymark. Generalized gini inequality indices. *Mathematical Social Sciences*, 1(4) :409–430, 1981.

[39] Ronald R. Yager. On Ordered Weighted Averaging Aggregation Operators in Multicriteria Decisionmaking. In Didier Dubois, Henri Prade, and Ronald R. Yager, editors, *Readings in Fuzzy Sets for Intelligent Systems*, pages 80–87. Morgan Kaufmann, 1993.

Actes JIAF 2024

Backward explanation via redefinition of predicates

Léo Saulières **Martin C. Cooper** **Florence Dupin de Saint-Cyr**

IRIT, University of Toulouse III, France
{first name}.{last name}@irit.fr

Résumé

Les 'History eXplanations based on Predicates' (HXPs) étudient les historiques de comportement d'un agent ayant appris par renforcement à travers le prisme de prédicats [21]. Pour ce faire, un score d'importance est calculé pour chaque action de l'historique. Les actions les plus importantes sont alors affichées à l'utilisateur. Le calcul de score d'importance étant difficile (#W[1]-dur), il est nécessaire pour des historiques longs d'approximer les scores, au détriment de leur qualité. Aussi, nous proposons une autre méthode intitulée 'Backward-HXP' afin de fournir des explications pour ces historiques, et ce sans avoir à approximer les scores. Les expérimentations confirment la pertinence des 'Backward-HXP'.

Abstract

History eXplanation based on Predicates (HXP), studies the behavior of a Reinforcement Learning (RL) agent in a sequence of agent's interactions with the environment (a history), through the prism of an arbitrary predicate [21]. To this end, an action importance score is computed for each action in the history. The explanation consists in displaying the most important actions to the user. As the calculation of an action's importance is #W[1]-hard, it is necessary for long histories to approximate the scores, at the expense of their quality. We therefore propose a new HXP method, called Backward-HXP, to provide explanations for these histories without having to approximate scores. Experiments confirm the usefulness of Backward-HXP.

1 Introduction

Nowadays, Artificial Intelligence (AI) models are used in a wide range of tasks in different fields, such as medicine, agriculture and education [12, 8, 4]. Most of these models cannot be explained or interpreted without specific tools, mainly due to the use of neural networks which are effectively black-box functions. Numerous institutions [17, 9] and researchers [7, 15] have emphasized the importance of providing comprehensible models to end users. This is why the eXplainable AI (XAI) research field, which consists in providing methods to explain AI behavior, is flourishing. In this context, we propose a method for explaining AI models that have learned using Reinforcement Learning (RL).

In RL, the agent learns by trial and error to perform a task in an environment. At each time step, the agent chooses an action from a state, arrives in a new state and receives a reward. The dynamics of the environment are defined by the non-deterministic transition function and the reward function. The agent learns a policy π to maximize its reward; this policy assigns an action to each state (defining a deterministic policy). Our eXplainable Reinforcement Learning (XRL) method is restricted to the explanation of deterministic policies.

Various works focus on explaining RL agents using a notion of importance. To provide a visual summary of the agent's policy, Amir and Amir [2] select a set of interactions of the agent with the environment

(sequences) using the "state importance" [5]. From a set of sequences, Sequeira et Gervasio propose to learn a set of information, to deduce interesting elements to show the user in the form of a visual summary [22]. Using a self-explainable model, Guo et al. determine the critical time-steps of a sequence for obtaining the agent's final reward [11].

To explain an RL agent, explanation must capture concepts of RL [16]. To this end, the HXP method [21], consists of studying a history of agent interactions with the environment through the prism of a certain predicate. A predicate d can represent any partial description of states. This XRL method answers the question: *"Which actions were important to ensure that d was achieved, given the agent's policy π?"*. This paper continues the work on this method, by proposing a new way of defining the important actions of a history, called Backward-HXP (B-HXP). Specially, B-HXP was investigated because of the limits of (forward) HXP in explaining long histories.

The paper is structured as follows. The theoretical principle of HXP is outlined in Section 2, before defining B-HXP in Section 3. Section 4 presents the experimental results carried out on 3 problems. Section 5 presents related works and Section 6 concludes.

2 HXP

An RL problem is modeled using a Markov Decision Process [23], which is a tuple $\langle \mathcal{S}, \mathcal{A}, R, p \rangle$. $\mathcal{S}$ represents the state space and $\mathcal{A}$ the action space. $A(s)$ denotes the set of available actions to perform from s. $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ and $p : \mathcal{S} \times \mathcal{A} \to Pr(\mathcal{S})$ are respectively the reward function and the transition function of the environment. $p(s'|s, a)$ represents the probability of reaching state s', having performed action a from state s. $\pi : \mathcal{S} \to \mathcal{A}$ denotes a deterministic policy that maps an action a to each state s; thus, $\pi(s)$ is the action performed by the agent in state s. Starting by doing an action a from a state s, the probability of a state s' is the product of the probabilities along the current path reaching s' according to π and p. In the following, we use the function *next* to compute the next possible states (associated with their probabilities) given a set of (state, probability) pairs S: $next_{\pi,p}(S) = \{(s', pr \times p(s'|s, a))$ such that $(s, pr) \in S$, $a = \pi(s)$ and $p(s'|s, a) \neq 0\}$. In order to compute the set of final states reachable at horizon k using the agent's policy π from a set of states S, the function $succ^k_{\pi,p}$ is defined recursively by $succ^0_{\pi,p}(S) = S$ and $succ^{n+1}_{\pi,p}(S) = next_{\pi,p}(succ^n_{\pi,p}(S))$.

HXPs provide to the user important actions for the respect of a predicate d, given an agent's policy π, by computing an importance score for each action in the history [21]. The language used for the predicate is based on the features that characterize a state. We consider a set of features $\mathcal{F} = \{f_1, ..., f_n\}$, where each feature f_i has a range of values defined by a domain D_i. The feature space is therefore $\mathbb{F} = D_1 \times ... \times D_n$. The state space $\mathcal{S}$ is a subset of $\mathbb{F}$. A predicate is given by a propositional formula with literals of the form $l_{i,j}$, where $l_{i,j}$ means that the feature f_i takes the value j in domain D_i. The importance score represents the benefit of performing an action a from s rather than another action $a' \in A(s)\backslash\{a\}$, where this benefit is the probability of reaching a state at a horizon of k that satisfies d. To evaluate an action we first require the notion of utility of a set of (state, probability) pairs.

Definition 1 (utility)**.** Given a predicate d, *the utility u_d of a set of (state, probability) pairs S* is:

$$u_d(S) = \sum_{(s,pr) \in S, s \models d} pr$$

Finally, the importance of an action a from a state s is defined as follows.

Definition 2 (importance)**.** Given a predicate d, an agent's policy π, a transition function p, the *importance score of a* from s at horizon k is defined by:

$$imp^k_{d,\pi,p}(s, a) = u_d(succ^k_{\pi,p}(S_{(s,a)})) - \operatorname*{avg}_{a' \in A(s)\backslash\{a\}} u_d(succ^k_{\pi,p}(S_{(s,a')})) \quad (1)$$

where avg is the average and $S_{(s,a)}$ is the set of reachable states (along with their probabilities) from s by performing action a. Formally, we have:

$$S_{(s,a)} = \{(s', p(s'|s,a)) \mid p(s'|s,a) \neq 0\}$$

The importance score lies in the range $[-1, 1]$, where a positive (negative) score denotes an important (resp. not important) action in comparison with other possible actions. Its computation is $\#W[1]$-hard [21], so it is necessary to approximate it, in particular by generating only part of the length-k scenarios with the *succ* function.

Approximate HXP consists in considering the last n time steps as deterministic over a horizon k, taking only the most probable transition into account. Thus, with b denoting the maximum number of transitions from a state-action couple (s, a) of a given RL problem, we produce at most b^{k-n} scenarios. As a note, the scenarios generated are not necessarily the most probable ones, since taking the most probable transition at each time step does not ensure that the most probable sequence is obtained.

To handle long histories on problems where the number of possible transitions is large, i.e. k and b large, it is necessary to use approximate methods to provide explanations in reasonable time, at the expense of only approximating the importance scores. In the next section, we propose a new way of computing HXP in a step-by-step backward approach, which allows us to provide explanations in reasonable time for long histories, without having to approximate the scores calculation. As we will see, this leads to other computational difficulties. The result is thus a novel method for the explanation of histories with different pros and cons compared to forward-based HXP.

3 Backward-HXP (B-HXP)

The idea of B-HXP is to iteratively look for the most important action in the near past of the state that respects the predicate under study. When an important action is found, we look at its associated state to define the new predicate to be studied. Indeed, by observing only a subset of the actions in the history (near past), the horizon for calculating importance scores is relatively small. In this sense, importance scores can be calculated exhaustively. The predicate is then modified so that actions can be evaluated with respect to a predicate that they can achieve within a shorter horizon. The following example will be used throughout this section to illustrate the method.

Example 1. *Consider the end of Bob's day. The history of Bob's actions is: [work, shop, watch TV, nap, eat, water the plants, read]. Bob's state is represented by* 5 *binary features: hungry, happy, tired, fridge, fuel. Fridge and fuel means respectively that the fridge is full and that the car's fuel level is full. Bob's last state is: (¬hungry, happy, tired, ¬fridge, ¬fuel) (for the sake of conciseness, Bob's states are represented by a boolean 5-tuple. Thus, Bob's last state is:* $(0, 1, 1, 0, 0)$*) The environment is deterministic and the predicate under study is "Bob is not hungry". We are looking for the* 2 *most important actions for Bob not to be hungry. Starting from the final state, the most important action in the near past is 'eat'. We are interested in its associated state, i.e. the state before doing the action 'eat', which is assumed to be* $(1, 0, 0, 1, 0)$*. The new predicate deduced from this state is "Bob is hungry and has a full fridge". In the near past of* $(1, 0, 0, 1, 0)$*, the 'shop' action is the most important one (among work, shop, watch TV and nap) for respecting this new predicate. To sum up, we can say that the reason that Bob is not hungry in the final state is that he went shopping (to fill his fridge) and then ate.*

Before describing in detail the B-HXP method, we introduce some notation. $H = (s_0, a_0, s_1, ..., a_{k-1}, s_k)$ denotes a length-k history, with $H_i = (s_i, a_i)$ denoting the state and action performed at time i, and for $i < j$, $H_{(i,j)}$ denotes the sub-sequence $H_{(i,j)} = (s_i, a_i, ..., s_j)$. In this section, we employ the term 'utility of a state s' to express the utility of the agent's action associated with s (this action being unique since we assume a deterministic policy). To define the near past of a state in H, it is necessary to introduce the maximum length of sub-sequences: l. This length must be sufficiently short to allow importance scores to be calculated in a reasonable time. The value of l depends on the RL problem being addressed, and specifically on the maximal number of possible transitions from any observable state-action pair,

namely b. It follows that the lower b is, the higher l can be chosen to be.

To provide explanations for long histories, we need a way of defining new intermediate predicates (such as Bob is hungry and the fridge is full in Example 1). For this we use Probabilistic Abductive eXplanations, shortened to PAXp [13].The aim of this formal explanation method is to explain the prediction of a class c by a classifier κ by providing an important set of features among $\mathcal{F}$. Setting these features guarantees (with at probability at least δ) that the classifier outputs class c, whatever the value of the other features. A classifier maps the feature space into the set of classes: $\kappa : \mathbb{F} \rightarrow \mathcal{K}$. We represent by $\mathbf{x} = (x_1, ..., x_n)$ an arbitrary point of the feature space and $\mathbf{v} = (v_1, ..., v_n)$ a specific point, where each v_i has a fixed value of domain D_i. [13] defines a weak PAXp as a subset of features for which the probability of predicting the class $c = \kappa(\mathbf{v})$ is above a given threshold δ when these features are fixed to the values in $\mathbf{v}$. A PAXp is simply a subset-minimal weak PAXp.

Definition 3 (PAXp [13])**.** Given a threshold $\delta \in [0, 1]$, a specific point $\mathbf{v} \in \mathbb{F}$ and the class $c \in \mathcal{K}$ such that $\kappa(\mathbf{v}) = c$, $\mathcal{X} \subseteq \mathcal{F}$ *is a weak PAXp* if:

$$Prop(\kappa(\mathbf{x}) = c \mid \mathbf{x}_{\mathcal{X}} = \mathbf{v}_{\mathcal{X}}) \geq \delta$$

where $\mathbf{x}_{\mathcal{X}}$ and $\mathbf{v}_{\mathcal{X}}$ are the projection of $\mathbf{x}$ and $\mathbf{v}$ onto features $\mathcal{X}$ respectively and $Prop(\kappa(\mathbf{x}) = c \mid \mathbf{x}_{\mathcal{X}} = \mathbf{v}_{\mathcal{X}})$ is the proportion of the states $\mathbf{x} \in \mathbb{F}$ satisfying $\mathbf{x}_{\mathcal{X}} = \mathbf{v}_{\mathcal{X}}$, that the classifier maps to c, in other words $|\{\mathbf{x} \in \mathbb{F} \mid \mathbf{x}_{\mathcal{X}}{=}\mathbf{v}_{\mathcal{X}} \text{ and } \kappa(\mathbf{x}){=}c\}|/|\{\mathbf{x} \in \mathbb{F} \mid \mathbf{x}_{\mathcal{X}}{=}\mathbf{v}_{\mathcal{X}}\}|$.

The *set of all WeakPAXp* for $\kappa(\mathbf{v}) = c$ wrt the threshold δ is denoted $\text{WeakPAXp}(\kappa, \mathbf{v}, c, \delta, \mathbb{F})$.

$\mathcal{X} \subseteq \mathcal{F}$ *is a PAXp* if it is a subset-minimal weak PAXp. The *set of all PAXp* for $\kappa(\mathbf{v}) = c$ wrt the threshold δ is denoted $\text{PAXp}(\kappa, \mathbf{v}, c, \delta, \mathbb{F})$.

The idea of using PAXp is to redefine the predicate to be studied for the next sub-sequence as we progress backwards. In order to fit the PAXp framework we define the classifier $\kappa_{s,\pi,p,d,k}$ as a binary classifier based on the utility of the state s. We note $u^k_{d,\pi,p}(s) = u_d(succ^k_{\pi,p}(\{(s, 1)\}))$, *the utility of* s wrt d given an horizon k, a policy π and a transition function p. The class of any state $\mathbf{x}$ is the result of a comparison between the utility of s and the utility of $\mathbf{x}$ for the respect of d.

Definition 4 (B-HXP classifier)**.** Given a state s, a policy π, a transition function p, a predicate d and a horizon k. The *B-HXP classifier*, denoted $\kappa_{s,\pi,p,d,k}$, is a function such that: for all $\mathbf{x} \in \mathcal{S}$,

$$\kappa_{s,\pi,p,d,k}(\mathbf{x}) = \begin{cases} True & \text{if } u^k_{d,\pi,p}(\mathbf{x}) \geq u^k_{d,\pi,p}(s) \\ False & \text{otherwise} \end{cases}$$

This classifier is specific to B-HXP. The utility threshold value depends on the state s which is the state associated with the most important action in the sub-sequence studied. It is used to generate a predicate d' which reflects a set of states at least as useful as s (with a probability of at least δ) for the respect of d. The predicate d' can then be seen as a sub-goal for the agent in order to satisfy d.

To assess whether a subset $\mathcal{X} \subseteq \mathcal{F}$ is a PAXp, it is necessary to calculate the utility of each state having this subset of features, which involves using the agent's policy π and the environment transition function p. A weak PAXp is then a sufficient subset of state features which ensures that a state utility is greater than or equal to the utility of s with probability at least δ. The new predicate is defined as the disjunction of every possible PAXp from s.

Definition 5. Given a state $s = (s_1, \ldots, s_n) \in \mathcal{S}$, a B-HXP classifier κ on $\mathcal{S}$, *the predicate PAXpred associated with* s for a given threshold δ is:

$$\text{PAXpred}_{\kappa}(s, \delta) = \bigvee_{\mathcal{X} \in \text{PAXp}(\kappa, s, True, \delta, \mathcal{S})} \left(\bigwedge_{f_i \in \mathcal{X}} f_i = s_i \right)$$

Example 1 (Cont)**.** *For this history,* δ *is set to* 1 *and* l *is* 4*. The first sub-sequence studied is: [nap, eat, water the plants, read]. The most important action relative to the achievement of "Bob is not hungry" is 'eat', with a score of, say,* 0.5 *and its associated state, say,* $(1, 0, 0, 1, 0)$*. We extract a PAXp, which includes the features* fridge *and* hungry *set to* 1*. This means that, whatever the values of the other features, a state with* fridge *and* hungry *set to* 1 *has (with* 100% *probability) a utility greater than or equal to* 0.5*. Now, we study the sub-sequence [work, shop,*

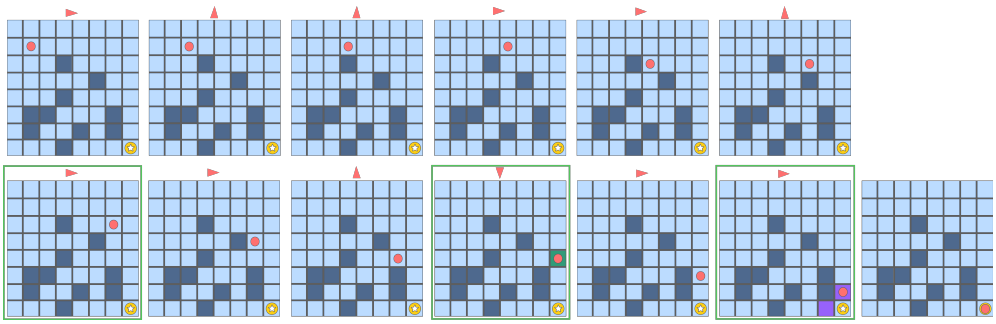


Figure 1: B-HXP for the *win* predicate. The agent is symbolized by a red dot, the dark blue cells are holes and the destination cell is marked by a star. Actions identified as important are highlighted by a green frame.

watch TV, nap], in which the most important action to achieve the new intermediate predicate is 'shop'.

In the backward analysis of H, the change of predicate allows us to look at a short-term objective to be reached, thus keeping the calculation of HXP reasonable. Our method is explained in pseudo-code in Algorithm 1. This algorithm allows us to go backwards through the history H, successively determining in each sub-sequence studied, the important action and its associated state predicate. The argmax function is used to find, in a given sub-sequence $H_{(i,j)}$, the most important action a, its associated state s, and its index in H. The latter is used to determine the next sub-sequence to consider. The *PAXpred* function is used to generate the new predicate to study in the next sub-sequence, based on Definition 5. The process stops when all actions have been studied at least once, or when the utility of the most important action in the current sub-sequence is 0. Finally, the algorithm returns a list of important actions and the different predicates found.

With B-HXPs, it is interesting to note that the number of actions to be presented to the user is not fixed. In the worst-case scenario, a user could end up with an explanation that refers to all actions as important. This would happen if it was always the last action in each subsequence which is the most important for achieving the current predicate. However, this problem was not observed in our experiments.

The computationally hard part of this approach is the predicate generation. Enumerating all the

Algorithm 1 B-HXP algorithm

Input: history H, maximal sub-sequence length l, agent's policy π, predicate d, transition function p, probability threshold δ, state space $\mathcal{S}$
Output: important actions A, predicates D

$A \leftarrow [] \; ; D \leftarrow []; u \leftarrow 1$
$i_{max} \leftarrow len(H) \; ; i_{min} \leftarrow \max(0, i_{max} - l)$
while $i_{min} \neq 0$ and $u \neq 0$ **do**
  $i, s, a \leftarrow \underset{\substack{i \in [i_{min}, i_{max}] \\ (s,a) = H_i}}{\text{argmax}} \; imp^{l}_{d,\pi,p}(s,a)$
  $u \leftarrow u^{l}_{d,\pi,p}(s)$
  $d \leftarrow \text{PAXpred}_{\kappa_{s,\pi,p,d,l}}(s, \delta)$
  $A.append(a) \; ; D.append(d)$
  $i_{max} \leftarrow i \; ; i_{min} \leftarrow \max(0, i_{max} - l)$
end while
return A, D

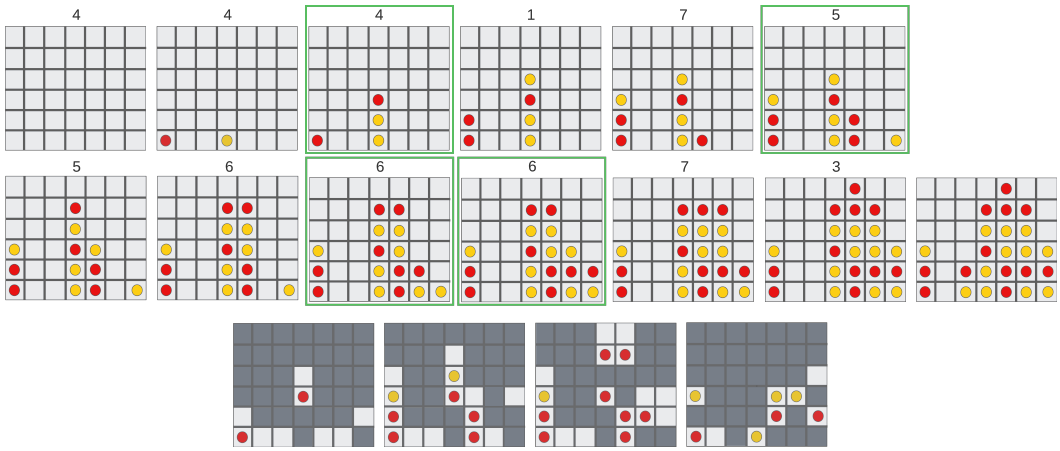


Figure 2: B-HXP for the *win* predicate. Above: the input history, showing 12 moves (where move = choice of column) of the agent (yellow) to which the environment (red) responds. Below: the predicates found by B-HXP corresponding to the four important moves it finds in the history, each highlighted by a green frame.

PAXp's turns out to be intractable. To support this assertion, even finding a single AXp (which is a PAXp with $\delta = 1$) is in general NP-hard, for example in the case of a DNF classifier [6]. Also, finding a single PAXp when $\delta < 1$ is NP-hard even for decision trees [3]. A further computational difficulty, specific to our problem, is that our classifier $\kappa_{s,\pi,p,d,k}$ requires, at each call, the computation of the action utility, which is a #W[1]-hard problem [21] w.r.t. the parameter k.

Thus, we decided to limit the definition of a predicate d to the generation of one weak PAXp. To obtain a predicate d in reasonable time, we need to look at a particular class of weak PAXp, the *locally-minimal* PAXps, which are not necessarily subset-minimal. Formally, a set of features $\mathcal{X} \subseteq \mathcal{F}$ is a *locally-minimal* PAXp if $\mathcal{X} \in \text{WeakPAXp}(\kappa, \mathbf{v}, c, \delta, \mathbb{F})$ and for all $j \in \mathcal{X}$, $\mathcal{X} \setminus \{j\} \notin \text{WeakPAXp}(\kappa, \mathbf{v}, c, \delta, \mathbb{F})$ The *findLmPAXp* algorithm [13] is used to calculate a *locally-minimal* PAXp.

In short, B-HXP keeps the calculation of importance scores exhaustive, by cutting the length-k history into sub-sequences of length l, starting with the end. The most important action of a sub-sequence is retained, and its associated state is used to define d', the new predicate to study, using *locally-minimal* PAXp. d' is then studied in a new sub-sequence. This process is iterated throughout the history. The next section presents examples of B-HXP.

4 Experiments

The experiments were carried out on 3 RL problems: Frozen Lake (FL), Connect4 (C4) and Drone Coverage (DC) [20]. Q-learning was used to solve the FL problem, and Deep-Q-Network for C4 and DC. The agents' training was performed using a Nvidia GeForce GTX 1080 TI GPU, with 11 GB of RAM. The B-HXP examples were run on an HP Elitebook 855 G8 with 16GB of RAM (source code available at: `https://github.com/lsaulier/HXP`).

The first part of this section describes the problems and the studied predicates (for more details, see [21]). The second part presents B-HXP examples.

In the figures, the history is displayed over two lines. The action taken by the agent from a state is shown above it. Important actions and their states are highlighted by a green frame. The third line of figures 2 and 3 corresponds to the predicates generated during the B-HXP, where a dark grey cell means that this feature is not part of the predicate. In Tables 1,

2 and 3, the importance scores given are w.r.t. either the initial predicate or the intermediate ones.

4.1 Description of the problems

In FL, the agent moves on the surface of a frozen lake (2D grid) to reach a certain goal position, avoiding falling into the holes. The agent can move in any of the 4 cardinal directions. However, due to the slippery surface of the frozen lake, the agent may slip and not end up in the position induced by the chosen action. Indeed, if the agent chooses a direction (e.g. *up* as in the second state of Figure 1), it has 0.6 probability to go in this direction and 0.2 to go towards each remaining direction except the opposite one (e.g., for *up*, 0.2 to go *left* and 0.2 to go *right*, as occurred in the scenario of Figure 1 where the agent moved right in the third state after performing *up* in the second one). The agent's state is composed of 5 features: its position (P) and previous position (PP) on the map, the position of one of the two holes closest to the agent (HP), the Manhattan distance between the agent's initial position and his current position (PD), and the total number of holes on the map (HN). Predicates *win*, *holes* and *region* were studied. They respectively determine whether the agent reaches the goal, falls into a hole or reaches a pre-defined set of map positions.

The C4 game is played on a 6 by 7 vertical board, where the goal is to align 4 tokens in a row, column or diagonal. Two players play in turn. An agent's state is the whole board. 5 predicates were studied: *win*, *lose*, *3 in a row*, *prevent 3 in a row* and *control mid-column*.

In DC, four drones must cover (observe) the largest area of a windy 2D map, while avoiding crashing into a tree or another drone. A drone can move in any of the 4 cardinal directions or not. A drone cover is a 3×3 square centered on it. A cover for a drone is optimal when it does not contain any trees and there is no overlap with the cover of the other drones. An agent's state is made up of its view, a 5×5 image centered on it, and its position, represented by (x, y) coordinates. Ten predicates for the DC problem were studied (local and global versions of): *perfect cover, maximum reward, no drones, crash* and *region*. Local versions concern a single agent, whereas the global versions concern all agents.

Table 1: Importance scores in the FL history 1

Predicate	Time-step / Importance score			
win	8 -0.001	9 0.04	10 0.012	11 **0.114**
$\text{PAXpred}_\kappa(s_{11}, 0.7)$ (a.k.a. *purple)*	7 0.006	8 -0.008	9 **0.102**	10 0.087
$\text{PAXpred}_\kappa(s_9, 0.7)$ (a.k.a. *green)*	5 -0.0003	6 **0.0**	7 -0.001	8 -0.0003

4.2 B-HXP examples

To provide B-HXPs in reasonable time, the $sample$ parameter, which corresponds to the maximum number of states observed for a feature evaluation in the *findLmPAXp* algorithm [13], i.e. the predicate generation, was set to 10 in the following examples. In other words, to avoid an exhaustive search over $\mathbb{F}$, the proportion in Definition 3 was computed based on 10 samples.

A B-HXP (computed in 2 seconds) for a FL history is shown in Figure 1, with l=4, δ=0.7. Importance scores are presented in Table 1. The *right* action linked to the state $s_{11} = \{P = (7,8), PP = (6,8), HP = (6,7), PD = 13, HN = 10\}$ is the most important in the first sub-sequence studied in order to *win*. The predicate, named *purple*, computed from s_{11} with $\delta = 0.7$ is $PAXpred_\kappa(s_{11}, 0.7) = \{PD = 13\}$. The states described by the predicate are shown in purple in Figure 1. In the following sub-sequence, the *down* action linked to state $s_9 = \{P = (5,8), PP = (5,7), HP = (6,7), PD = 11, HN = 10\}$ is the most important to respect *purple*. The predicate, named *green*, computed based on s_9 is $PAXpred_\kappa(s_9, 0.7) = \{P = (5,8), PP = (5,7), HP = (6,7)\}$ (the states described are shown in green in Figure 1). A predicate is generic if it is respected by a large number of different states. We note that *purple* describes more states than *green*. The latter is not generic enough, which is reflected in the importance scores, which are close to 0: whatever the action, it is unlikely to respect this predicate after 4 time steps. The entire history is not explored when calculating the B-HXP, as the utility

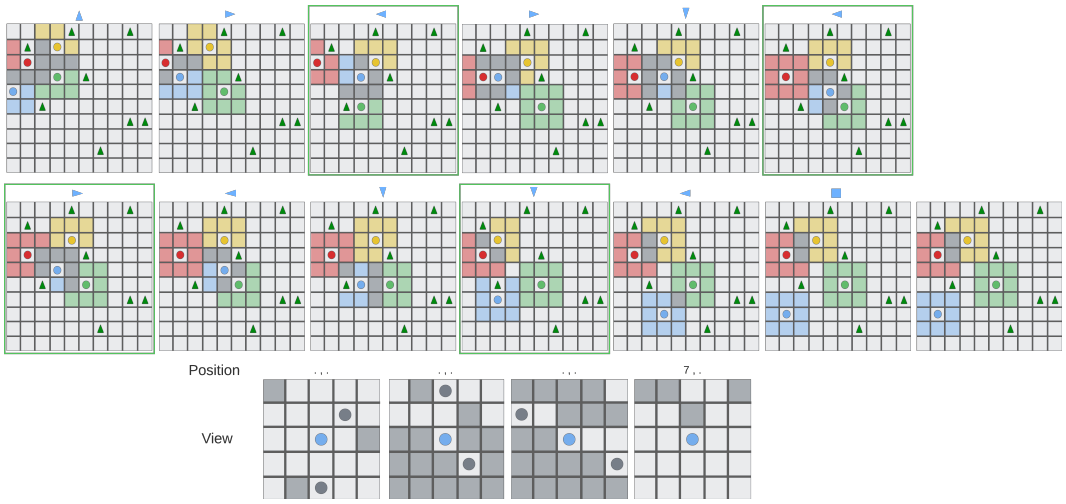


Figure 3: B-HXP for the *perfect cover* predicate. The intermediate predicates are shown in the last row. The rightmost of these states that the blue drone is in row 7 and that the light grey squares are free. The other intermediate predicates impose the relative positions of two other drones and that some squares are free. The drones are represented by dots and the trees by green triangles. In the history, a colored cell means that the area is covered by the drone of the same color and a dark grey cell indicates an overlap of the coverage of different drones.

Table 2: Importance scores in the C4 history 2

Predicate	Time-step / Importance score		
win	9 **0.726**	10 0.099	11 0.16
$\text{PAXpred}_{\kappa}(s_9, 0.8)$	6 -0.006	7 0.03	8 **0.113**
$\text{PAXpred}_{\kappa}(s_8, 0.8)$	5 **0.003**	6 0.0	7 0.0
$\text{PAXpred}_{\kappa}(s_5, 0.8)$	2 **0.003**	3 0.0	4 0.0

of the last state selected s_6 is 0. The selected actions form a meaningful explanation when we look at the predicates studied. However, the redefined predicates fairly quickly become very specific and probably of little help in explaining why the agent won.

With l set to 3 and δ to 0.8, a B-HXP (computed in 10 seconds) for a C4 history is shown in Figure 2. A large part of the board is ignored in the predicates, which gives the user an intuition of the type of states that the agent must reach. Importance scores are presented in Table 2. Almost all the actions returned are related to setting up the token alignment leading to victory, which is interesting because the predicate *win* is only studied on the last three states of the history. The other predicates provide actions linked to achieving a *win*. The first redefined predicate (the rightmost image on the third line of Figure 2) describes a partial board configuration: from positions satisfying this predicate, an agent following the learnt policy has at least 80% chance of achieving a win in the final position. However, as with FL, apart from the predicate defined at time-step 9, the other predicates generated seem to be not sufficiently generic, which can be seen in the scores. Indeed, the second redefined predicate (the second-from-right image in the last line of Figure 2) is so specific as to uniquely determine the exact board configuration (given the height of columns and the number of tokens played).

Table 3: Importance scores in the DC history 3

Predicate	Time-step / Importance score		
perfect cover	9 **0.114**	10 0.063	11 0.056
$\text{PAXpred}_{\kappa}(s_9, 1)$	6 **0.008**	7 0.006	8 0.002
$\text{PAXpred}_{\kappa}(s_6, 1)$	3 0.053	4 -0.013	5 **0.09**
$\text{PAXpred}_{\kappa}(s_5, 1)$	2 **0.034**	3 -0.036	4 0.023

The value of l has an important impact on the computation of importance scores. This point is explored in more detail in the conclusion.

A B-HXP (calculated in 13 seconds) for a DC history is shown in Figure 3, with l=3 and δ=1. The explanation is for the blue drone with the original predicate that this agent has a perfect cover. The importance scores are presented in Table 3. The predicates give a good intuition of the type of state (position and 5×5 view) the agent is trying to reach.

5 Related Work

XRL methods can be clustered according to the scope of the explanation (e.g. explaining a decision in a given situation or the policy in general), the key RL elements used to produce the explanation (e.g. states [10, 18], rewards [14, 1]), or the form of the explanation (e.g. saliency maps [10] or sequence-based visual summaries [2, 22, 20]).

One approach consists in generating counterfactual trajectories (state-action sequences) and comparing them with the agent's trajectory. In [1], reward influence predictors are learnt to compare trajectories. The counterfactual one is generated based on the user's suggestion. In [25], a contrastive policy based on the user's question is produced to generate the counterfactual trajectory. In the MDP context, Tsirtsis et al. generate optimal counterfactual trajectories that differ at most by k actions [24].

EDGE [11] is a self-explainable model. Like HXP, it identifies the important elements of a sequence. However, EDGE is limited to importance based on the final reward achieved, whereas HXPs allow the study of various predicates. In addition, HXP relies on the transition function (which is assumed to be known) and the agent's policy to explain, whereas EDGE [11] requires the learning of a predictive model of the final-episode reward.

6 Conclusion

Our paper is a follow-up to the work carried out in [21]. HXP is a method that makes it possible to answer, for a given history, the question: *"Which actions were important to ensure that the predicate d was achieved, given the agent's policy π?"*. To do this, an importance score is computed for each action in the history. To provide explanations for long histories, without importance score approximation, we defined an approach named Backward-HXP. Starting from the end of the history, this involves iteratively studying a subsequence, highlighting the most important action in it and defining a new intermediate predicate to study for the next sub-sequence. The intermediate predicate is a *locally-minimal PAXp* associated to the state where the most important action took place.

In the experiments, we observed that the genericity of a predicate d and the search horizon l influence the importance scores. The more generic the predicate d, the greater the probability of finding states with a l horizon that respect d. Thus, the importance scores generated are significant, regardless of l. Conversely, a less generic predicate makes it more difficult to evaluate an action. In this case, the value of l is discriminating. A too specific predicate d can lead to insignificant importance scores for the respect of d, as the utility of actions is close to 0. In several histories, notably C4 ones, the predicates generated are non generic, leading to less interesting explanations.

Although in the examples the important actions are often related to the respect of the initial predicate, this is not always the case. If we consider the first redefined predicate as a possible cause of the predicate being satisfied in the final state, then the second redefined predicate is a possible cause of a possible cause. The notion of causality can quickly

become highly diluted (due to the fact for computational reasons, we study a single cause at each step). The user must be aware of this effect when computing B-HXPs.

HXP and B-HXP offer the user great diversity in the study of agent behavior through its notion of predicate. Furthermore, this approach is agnostic with regard to the agent learning algorithm. As described in [21], the strong assumption for the use of HXP and B-HXP is the knowledge of the transition function. It must be known during the explanation phase (not necessarily during training), or at least approximated, for example using an RL model-based method.

More experiments are needed to ensure the quality and scalability of B-HXP, specially in environments with a large number of transitions. When calculating a *locally-minimal* PAXp, the order in which features are processed is important. A future work would be to direct the generation of *locally-minimal* PAXp using a feature ordering heuristic, such as LIME [19]. In this way, it would be possible to compare the intermediate predicates and check whether this changes the important actions returned.

Our experiments have shown the feasibility of the finding important actions in a long sequence of actions by redefining predicates, working backwards from the end of the sequence. However, we found that the intermediate predicates can quickly become very specific leading to the difficulty of calculating the importance scores of actions w.r.t. these very specific predicates. Further research is required to investigate this point.

Acknowledgement

We would like to thank the reviewers for their pertinent comments, which helped to improve the paper quality.

References

[1] Amal Alabdulkarim and Mark O. Riedl. Experiential explanations for reinforcement learning. *CoRR*, abs/2210.04723, 2022.

[2] Dan Amir and Ofra Amir. HIGHLIGHTS: summarizing agent behavior to people. In *AAMAS*, pages 1168–1176. ACM, 2018.

[3] Marcelo Arenas, Pablo Barceló, Miguel A. Romero Orth, and Bernardo Subercaseaux. On computing probabilistic explanations for decision trees. In *NeurIPS*, 2022.

[4] Lijia Chen, Pingping Chen, and Zhijian Lin. Artificial intelligence in education: A review. *IEEE Access*, 8:75264–75278, 2020.

[5] Jeffery A. Clouse. *On integrating apprentice learning and reinforcement learning*. PhD thesis, UMass Amherst, 1996.

[6] Martin C. Cooper and João Marques-Silva. Tractability of explaining classifier decisions. *Artif. Intell.*, 316:103841, 2023.

[7] Adnan Darwiche. Human-level intelligence or animal-like abilities? *Commun. ACM*, 61(10):56–67, 2018.

[8] Ngozi Clara Eli-Chukwu. Applications of artificial intelligence in agriculture: A review. *Engineering, Technology & Applied Science Research*, 9(4), 2019.

[9] European Commission. Artificial Intelligence Act, 2021.

[10] Samuel Greydanus, Anurag Koul, Jonathan Dodge, and Alan Fern. Visualizing and understanding Atari agents. In *ICML*, pages 1787–1796. PMLR, 2018.

[11] Wenbo Guo, Xian Wu, Usmann Khan, and Xinyu Xing. EDGE: explaining deep reinforcement learning policies. In *NeurIPS*, pages 12222–12236, 2021.

[12] Pavel Hamet and Johanne Tremblay. Artificial intelligence in medicine. *Metabolism*, 69:S36–S40, 2017.

[13] Yacine Izza, Xuanxiang Huang, Alexey Ignatiev, Nina Narodytska, Martin C. Cooper,

and João Marques-Silva. On computing probabilistic abductive explanations. *Int. J. Approx. Reason.*, 159:108939, 2023.

[14] Zoe Juozapaitis, Anurag Koul, Alan Fern, Martin Erwig, and Finale Doshi-Velez. Explainable reinforcement learning via reward decomposition. In *IJCAI/ECAI workshop on explainable artificial intelligence*, page 7, 2019.

[15] Zachary C. Lipton. The mythos of model interpretability. *Commun. ACM*, 61(10):36–43, 2018.

[16] Stephanie Milani, Nicholay Topin, Manuela Veloso, and Fei Fang. A survey of explainable reinforcement learning. *CoRR*, abs/2202.08434, 2022.

[17] White House Office of Science and Technology. Blueprint for an AI Bill of Rights. 2022.

[18] Matthew L. Olson, Lawrence Neal, Fuxin Li, and Weng-Keen Wong. Counterfactual states for Atari agents via generative deep learning. *CoRR*, abs/1909.12969, 2019.

[19] Marco Túlio Ribeiro, Sameer Singh, and Carlos Guestrin. "Why should I trust you?": Explaining the predictions of any classifier. In *SIGKDD*, pages 1135–1144. ACM, 2016.

[20] Léo Saulières, Martin C. Cooper, and Florence Bannay. Reinforcement learning explained via reinforcement learning: Towards explainable policies through predictive explanation. In *ICAART, Vol. 2*, pages 35–44, 2023.

[21] Léo Saulières, Martin C Cooper, and Florence Dupin de Saint Cyr. Predicate-based explanation of a Reinforcement Learning agent via action importance evaluation. In *ECML/PKDD workshop AIMLAI*, 2023.

[22] Pedro Sequeira and Melinda T. Gervasio. Interestingness elements for explainable reinforcement learning: Understanding agents' capabilities and limitations. *Artif. Intell.*, 288:103367, 2020.

[23] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[24] Stratis Tsirtsis, Abir De, and Manuel Rodriguez. Counterfactual explanations in sequential decision making under uncertainty. In *NeurIPS*, pages 30127–30139, 2021.

[25] Jasper van der Waa, Jurriaan van Diggelen, Karel van den Bosch, and Mark A. Neerincx. Contrastive explanations for reinforcement learning in terms of expected consequences. *CoRR*, abs/1807.08706, 2018.

Actes JIAF 2024

Explicabilité en Apprentissage par Renforcement : vers une Taxinomie Unifiée

Maxime Alaarabiou[1] **Nicolas Delestre**[1] **Laurent Vercouter**[1]

[1] INSA Rouen Normandie, Normandie Univ, LITIS UR 4108, F-76000 Rouen, France

maxime.alaarabiou@insa-rouen.fr
nicolas.delestre@insa-rouen.fr
laurent.vercouter@insa-rouen.fr

Résumé

La problématique de l'explicabilité est à l'heure actuelle un enjeu important en intelligence artificielle, et plus spécifiquement en apprentissage par renforcement. Dans cet article, nous proposons une nouvelle classification des techniques d'explicabilité pour l'apprentissage par renforcement. Pour se faire, nous nous appuyons sur les aspects spécifiques de l'apprentissage par renforcement en définissant les concepts d'explication, de source d'explication et de technique d'explicabilité. En utilisant cette nouvelle classification, nous effectuons une analyse de l'état de l'art des techniques existantes dans ce domaine. Enfin, nous proposons une rétrospective de l'évolution des systèmes de classification, mettant en lumière les avancées et les tendances récentes en apprentissage par renforcement explicable.

Abstract

The issue of explainability is a current important concern in artificial intelligence, specifically in reinforcement learning. In this article, we propose a new classification of explainable reinforcement learning techniques. To do so, we rely on specific aspects of reinforcement learning by defining the concepts of explanation, source of explanation, and explicability technique. Using this new classification, we conduct an analysis of the state-of-the-art techniques existing in this field. Finally, we offer a retrospective on the evolution of classification systems, highlighting recent advances and trends in this domain.

1 Introduction

L'apprentissage par renforcement (*Reinforcement Learning*, RL) est un champ de l'intelligence artificielle (*Artificial Intelligence*, AI), et plus particulièrement de l'apprentissage machine, dans lequel on cherche à créer un agent qui maximise sa récompense dans un environnement [30]. L'intervention humaine se limite au strict minimum dans le processus d'apprentissage. L'agent doit donc développer une stratégie dans l'environnement en passant par une phase d'exploration non guidée. Pendant cette phase d'exploration, l'agent expérimente par essais-erreurs afin de mettre au point une fonction politique pertinente. Cette fonction a pour objectif d'établir une correspondance entre l'observation et les actions de l'agent, de sorte qu'une succession de décisions suivant cette politique maximise l'espérance de la somme des récompenses. Pour améliorer cette politique, on apprend la fonction de critique qui pour une observation donnée, prédit l'espérance de la somme des récompenses.

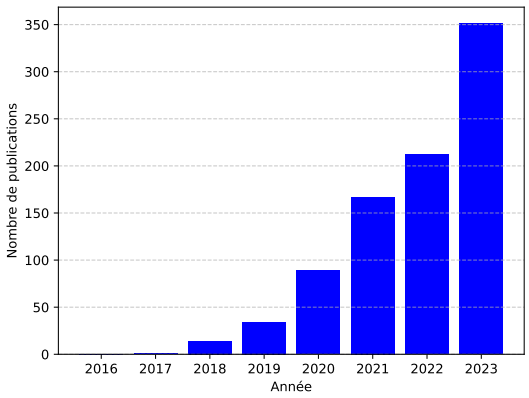


FIGURE 1 – Évolution du nombre de publications sur arXiv ayant le mot-clé "XAI" dans leur résumé.

Cette méthode d'entraînement a obtenu beaucoup de succès depuis l'émergence de l'apprentissage profond, car les réseaux de neurones permettent d'approximer efficacement les fonctions de politique et de critique [6, 19]. L'un des apports les plus marquants de cette méthode est la capacité d'un agent obtenu par RL à affronter une équipe de joueurs professionnels du jeu Dota [2].

L'utilisation à grande échelle d'agents ayant appris par renforcement rencontre toujours plusieurs défis, malgré les progrès significatifs réalisés dans le domaine [19, 10, 25, 16, 2, 28]. Cette réticence à adopter ces technologies repose principalement sur le manque de confiance que les humains accordent aux systèmes qu'ils ne peuvent

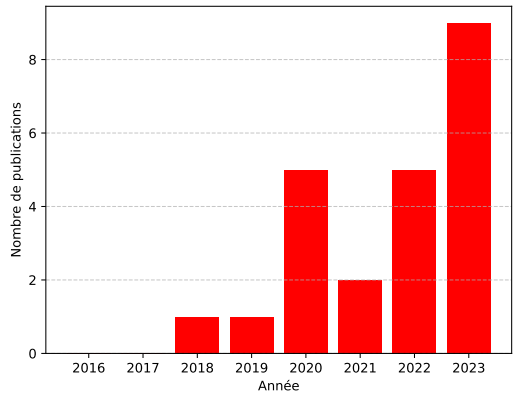


FIGURE 2 – Évolution du nombre de publications sur arXiv ayant le mot-clé “XRL” dans leur résumé.

pas expliquer [9]. Pour résoudre ce problème, un nouveau champ de recherche a émergé : l'intelligence artificiel explicable (*eXplainable Artificial Intelligence*, XAI [9]). Dans le cadre de l'apprentissage par renforcement, on parle plus spécifiquement d'apprentissage par renforcement explicable (*eXplainable Reinforcement Learning*, XRL).

En raison de la pertinence du sujet et pour espérer une plus large utilisation du RL dans le monde réel un nombre croissant d'équipes de recherche travaillent à mettre au point de nouvelles méthodes d'explicabilité. Cet engouement pour l'XAI se traduit ces dernières années par une rapide augmentation du nombre de publications, comme illustré dans les figures 1 et 2. Face à cette croissance rapide, il est apparu crucial pour ce champ de se structurer et de proposer une taxinomie unifiée. Cela permettrait d'assurer la comparaison des futurs résultats entre eux, la compréhension des chercheurs via l'utilisation d'un vocabulaire unifié dans la communauté et une organisation structurée.

Pour permettre aux agents d'évoluer dans le monde réel, il est nécessaire qu'ils puissent interagir à travers des environnements partiellement observables et impliquant un grand nombre d'états. C'est pourquoi nous nous positionnons, dans le cadre de l'apprentissage par renforcement profond (*Deep Reinforcement Learning*, DRL) ainsi que dans un contexte d'environnements décisionnels markoviens partiellement observables (*Partially Observable Markov Decision Processes*, POMDP).

Dans cet article notre objectif est de nous appuyer sur deux articles de revue de l'état de l'art [18, 24] afin d'en synthétiser les informations et de produire une taxinomie unifiée. Celle-ci aura pour but de classifier de manière précise les méthodes existantes ainsi que celles à venir.

2 Présentation du système de classification

Les **explications** [1] décrivent à un **observateur** le fonctionnement d'un système d'intelligence artificielle. Cela lui permet de construire un **modèle mental** du système. Un modèle mental représente la compréhension de l'observateur du système ainsi que sa capacité à anticiper les comportements du système dans de nouvelles situations [8]. Pour obtenir des explications, en RL on utilise des **sources d'explication** sur lesquelles des **techniques d'explicabilité** sont appliquées, comme l'illustre la figure 3.

2.1 Sources d'explication

Nous dénombrons trois **sources d'explication** en RL : les politiques, les trajectoires et les fonctions de valeurs. Ces sources sont obtenues pendant ou après l'entraînement. Dans tous les cas, elles capturent des informations sur l'agent.

Les **politiques** sont des fonctions qui, pour une observation donnée, génèrent une distribution dans l'espace des actions. L'objectif de l'apprentissage par renforcement est de trouver une politique qui, pour chaque état, maximise la somme des récompenses à venir. Les politiques sont améliorées de manière itérative pendant la phase d'apprentissage [30].

Les **trajectoires** représentent une séquence d'interactions successives entre l'environnement et l'agent. En fonction de l'observation, l'agent sélectionne une action qui influence l'environnement, produisant ainsi une récompense et une nouvelle observation. Ce processus se répète jusqu'à la fin de la trajectoire [30]. Nous distinguons trois types de trajectoires : les trajectoires d'entraînement, les trajectoires post-entraînement et les trajectoires fictives. Les trajectoires d'*entraînement* sont celles obtenues par les interactions entre une politique en phase d'apprentissage et l'environnement. Les trajectoires *post-entraînement* sont générées par les politiques finales. Les trajectoires *fictives* résultent de l'interaction entre une politique et une approximation de l'environnement.

Les **fonctions de valeurs** génèrent une prédiction sur la suite de la trajectoire pour une observation donnée [31]. La fonction de valeur la plus couramment utilisée est la fonction de critique, qui fournit une prédiction sur la somme des récompenses obtenues dans le futur de la trajectoire. Cette fonction de critique joue un rôle important dans le RL. L'agent va apprendre à orienter ses actions afin de maximiser cette prédiction. Cependant, d'autres fonctions de valeurs peuvent être considérées afin de produire de l'explicabilité (voir la sous-section 3.3).

2.2 Explication

Une **explication** est une information interprétable, c'est-à-dire une information qui peut être observée, comprise et étudiée par un observateur [1]. C'est à partir de ces informations que l'observateur construit son modèle mental du fonctionnement du système. Nous proposons de caractériser une explication en RL par sa **forme** et son **sujet**.

Nous définissons la **forme** d'une explication comme la caractérisation de ce qu'elle décrit. Selon nos connaissances actuelles [18, 24], nous définissons que les explications peuvent prendre les formes suivantes :

Modèle de la politique : modèle interprétable de la fonction de politique d'un agent.

Modèle de l'interaction : modèle interprétable des interactions entre l'agent et l'environnement.

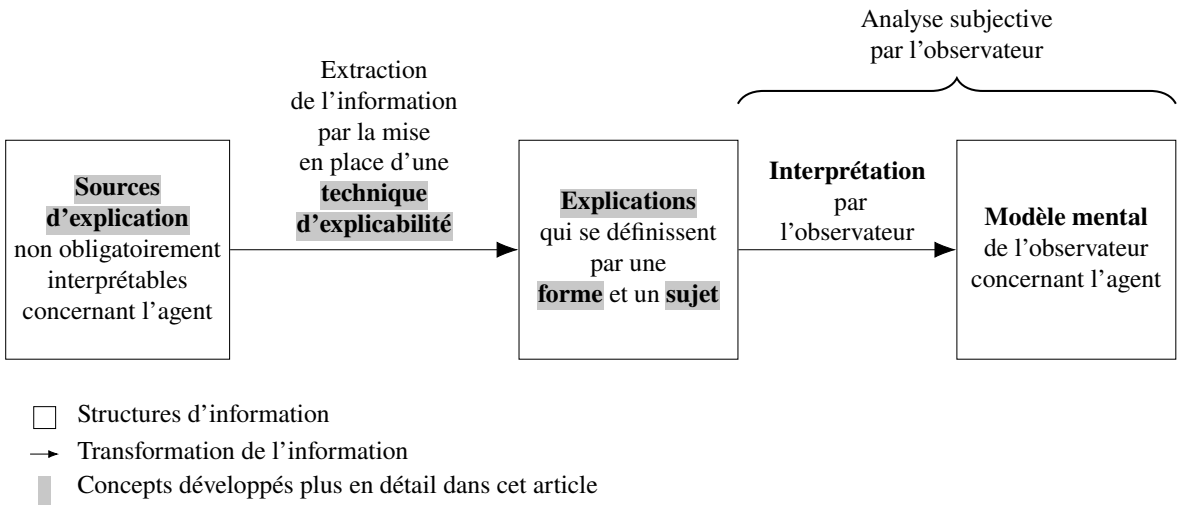


FIGURE 3 – Processus d'explication en apprentissage par renforcement explicable.

Trajectoire de l'agent : représentation d'un ensemble d'interactions successives entre l'environnement et l'agent.

Prédiction sur une trajectoire : prédiction produite par une fonction de valeur sur le futur de la trajectoire.

Contrefactuelle : représentation de l'observation obtenue par la perturbation de l'observation originale dans le but de produire une action différente de la part de l'agent.

Carte de chaleur : pondération de chaque caractéristique des données d'entrée qui représente l'impact sur la sortie de la fonction de politique.

Le **sujet** d'une explication caractérise quelle facette du RL est expliquée. Le sujet d'une explication est entièrement déterminé par sa forme. Il existe trois catégories de sujets : l'*apprentissage*, la *décision* et la *stratégie* [18].

Une explication portant sur l'*apprentissage* offre des informations sur les évolutions itératives de la politique. Par exemple, elle pourrait détailler quelles expériences significatives ont permis à une politique de converger vers une stratégie spécifique ainsi que les grandes étapes de son développement.

Concernant la compréhension de la *décision* d'un agent, cela revient à identifier les éléments de l'observation qui sont déterminants dans le choix de l'action. Ce type d'explication se rapproche de l'apprentissage supervisé, car il ne tient pas compte de la dimension séquentielle de la prise de décision.

Enfin, une explication axée sur la *stratégie* fournit des informations sur une séquence de décisions. Elle cherche ainsi à résumer les choix faits par l'agent en analysant les interactions entre un ensemble de décisions et l'environnement.

2.3 Technique d'explicabilité

Une technique d'explicabilité est un **algorithme** qui utilise une ou plusieurs sources afin de générer des explications pour un humain. Il existe une grande variété de techniques, et leur présentation sera l'objet de la section 3. Ces techniques sont déterminantes dans le processus d'explication car elles définissent les sources utilisées ainsi que le sujet des explications.

3 Analyse des techniques d'explicabilité

Le but de cette section est de détailler le fonctionnement des techniques d'explicabilité (cf. tableau 1). Elles sont présentées en fonction des formes d'information qu'elles produisent. Chaque sous-section présente une **forme d'explication** ainsi que les **techniques** qui lui sont associées.

3.1 Modèles interprétables de la politique

Il existe une catégorie de modèles qui sont compréhensibles via la simple analyse de leur représentation ; on dit de ces modèles qu'ils sont intéprétables [1]. Ces modèles sont utilisés pour représenter la fonction de politique. Les principaux modèles considérés comme intéprétables sont : les arbres de décision [32], les algorithmes [33], la logique formelle [22] et les modèles linéaires [5]. Il est possible de mettre à profit l'interprétabilité intrinsèque de ces modèles pour produire de l'explicabilité dans le carde du RL.

Cette représentation directement intéprétable permet à l'observateur d'extraire de l'information. Une explication issue d'un modèle intépretable fournit principalement des informations sur le processus de décision, car ce qui est modélisé est la correspondance entre une observation et une action. Si le modèle est suffisamment simple ou bien organisé, il est possible pour l'observateur de se faire une représentation d'une séquence de décisions à travers l'environnement et ainsi, fournir des informations sur la stratégie de l'agent.

Pour obtenir un modèle interprétable de la politique, nous disposons de deux techniques d'explication : l'*apprentissage d'une politique interprétable* et l'*approximation de la politique par un modèle interprétable*.

Sujet de l'explication	**Forme de l'explication**	**Technique d'explicabilités**	**Source de l'explication**			
			Politique	Trajectoires d'entraînements	Trajectoires post-entraînements	Fonctions de valeurs
Apprentissage	Trajectoires d'entraînements	*Extraction par comparaisons multi-critiques*		✓		✓
Décision	Modèles interprétables de la politique	*Apprentissage d'une politique interprétable*	✓			
		Approximation de la politique par un modèle interprétables	✓			
	Cartes de chaleur de l'observation	*Perturbation de l'observation*	✓			
		Analyse du gradient de l'observation	✓			
		Architecture à base d'attention	✓			
	Contrefactuelles de l'observation	*Utilisation de l'espace latent pour la génération de contrefactuelles*	✓			
Stratégie	Prédictions sur la trajectoire de l'agent	*Fonctions de valeurs comme observation*				✓
		Enrichissement du critique				✓
	Modèles de l'interaction agent-environnement	*Apprentissage d'un réseau bayésien*			✓	
		Exploitation de l'espace latent pour la génération d'un MDP	✓			
	Trajectoires post-entraînement	*Décomposition hiérarchique*	✓			
		Extraction par analyse du critique			✓	✓

TABLE 1 – Taxinomie des techniques d'explications en apprentissage par renforcement

3.1.1 *Apprentissage d'une politique interprétable*

Cette technique a pour objectif l'apprentissage d'une politique représentée par un modèle **interprétable** [32, 33]. Il s'agit de l'approche la plus directe pour résoudre les problématiques posées par l'XRL. Ce type d'entraînement nécessite une modification du MDP afin de l'adapter à l'espace de recherche et au type de modélisation de la politique. De plus, cet espace de recherche doit souvent être minimisé afin de rendre l'apprentissage possible. Cette **réduction** nécessite l'intégration de **connaissances expertes** dans le processus d'apprentissage. Pour cette technique, c'est la *fonction de politique interprétable* qui est utilisée pour produire de l'explication à destination de l'observateur.

3.1.2 *Approximation de la politique par un modèle interprétable*

À l'aide d'un modèle interprétable, il est possible d'**approximer** une politique non interprétable [27]. Dans ce cas, on se rapporte à un problème d'**apprentissage supervisé**, où l'on utilise la politique non interprétable pour étiqueter chaque observation différente avec l'action choisie par la politique. En utilisant le jeu de données, on effectue un apprentissage supervisé à l'aide du modèle interprétable. Le *modèle interprétable* ainsi créé sera alors utilisé comme explication pour l'observateur. Si cette approximation est suffisamment fidèle, elle peut être utilisée dans l'environnement à la place de la politique, permettant ainsi d'avoir un maximum de contrôle sur l'agent.

3.2 Cartes de chaleur de l'observation

Les cartes de chaleur permettent d'identifier quelles parties du vecteur d'entrée sont déterminantes dans la sélection de la sortie (comme l'illustre la figure 4). Appliqué au cadre de l'apprentissage par renforcement, cela met en valeur les parties de l'espace d'observation déterminantes dans le choix de l'action. De par sa nature, cette forme d'explication fournit des informations sur la prise de décision de la politique. La compréhension simple de ce type de représentation permet une utilisation par un large public. Il existe trois techniques d'explicabilité pour obtenir cette explication : *perturbation de l'observation*, *analyse du gradient* et *architecture à base d'attention*.

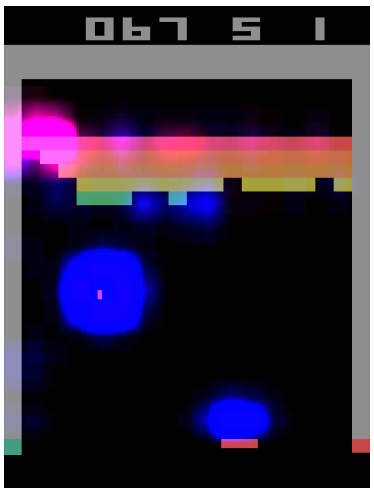


FIGURE 4 – Carte de chaleur par perturbation pour le jeu “Breakout” d’Atari [7]. En surbrillance, les zones les plus déterminantes pour le choix de la fonction de politique.

3.2.1 *Perturbation de l’observation*

L’analyse par perturbation consiste à appliquer différentes **modifications** à l’observation pour étudier leur **impact** sur la sortie de la **fonction de politique** [7]. Plus la variation de la sortie due à une perturbation sur une des composantes du vecteur d’entrée est **importante**, plus cette composante se voit attribuer une **pondération élevée**. La mesure de la perturbation s’effectue en calculant la distance entre la sortie originale et la sortie avec une perturbation en entrée. En utilisant cet ensemble de pondérations, on crée une *carte de chaleur* qui permet à l’observateur de visualiser quelles parties du vecteur d’observation sont sensibles pour le choix de l’action.

3.2.2 *Analyse du gradient de l’observation*

L’analyse du **gradient** de l’observation est un ensemble de techniques qui nécessitent une fonction de politique différenciable pour en extraire de l’information [29, 35, 12]. Pour se faire, on calcule le gradient de la fonction de politique pour l’ensemble des **composants d’un vecteur d’entrée**. Pour chaque composant du vecteur d’entrée, on obtient avec ce gradient une pondération qui représente son importance dans la sortie de la fonction politique. À partir de ces pondérations, on crée une *carte de chaleur* pour permettre à l’observateur d’identifier visuellement les parties du vecteur d’entrée les plus influentes dans la sortie de la fonction de politique.

3.2.3 *Architecture à base d’attention*

Les blocs d’**attention** constituent une architecture pour les réseaux de neurones permettant de pondérer l’importance relative d’une partie d’un vecteur par rapport aux autres parties de ce même vecteur [34]. La pondération se calcule en fonction du vecteur lui-même et est apprise par le réseau de neurones lors de sa phase d’apprentissage. Pour une observation spécifique, chaque partie du vecteur d’observation se voit attribuer un scalaire. On peut donc estimer que les parties avec les pondérations les plus élevées sont les plus importantes pour le choix de la sortie [20, 13]. C’est cette *carte de chaleur* des blocs d’attention qui sera communiquée à l’observateur comme explication.

3.3 Prédictions sur la trajectoire de l’agent

Effectuer des prédictions sur le futur d’une trajectoire permet de se représenter ce à quoi le système s’attend, fondé sur ses expériences d’entraînement (cf. figure 5). Ces prédictions sont réalisées à l’aide de fonctions de valeur. Ces fonctions permettent d’éclairer l’observateur sur le devenir de la trajectoire de l’agent. Par leur nature, cette forme d’explication renseigne sur la stratégie adoptée par l’agent.

La problématique de ce type d’approche réside dans la nécessité de s’assurer que les valeurs prédites par ces fonctions sont réellement déterminantes dans les actions sélectionnées par l’agent. Pour cela, on utilise les techniques suivantes : l’utilisation de *fonctions de prédiction comme observation* et l’*enrichissement du critique*.

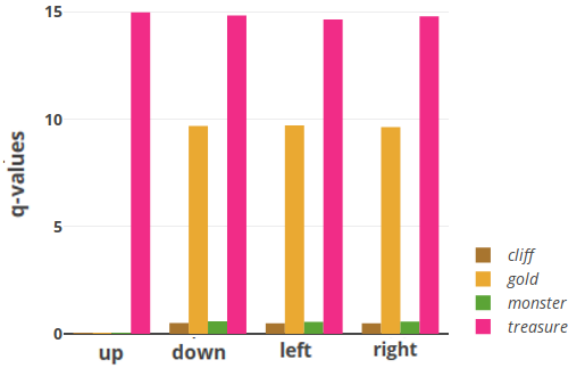


FIGURE 5 – Prédiction des sous-récompenses en fonction de l’action choisie par l’agent pour un états donné [14]. En fonction de l’action choisie par l’agent on peut déterminer quelle sous-récompense il cherche à maximiser.

3.3.1 *Fonctions de valeurs comme observation*

L’objectif est d’entraîner des **fonctions de valeurs interprétrables** afin de les utiliser comme observations pour l’agent. Si les caractéristiques prédites par les fonctions de valeurs ont du **sens** pour l’observateur, celui-ci sera capable de déterminer les aspects que l’agent cherche à maximiser par ses actions [15].

3.3.2 *Enrichissement du critique*

Afin d’obtenir des informations interprétables à partir des prédictions du **critique**, il est possible de **diviser** ce critique en **sous-fonctions** [14]. Ainsi, on décompose la fonction de récompense en sous-fonctions de récompenses, représentant ainsi des **sous-objectifs** de la tâche à accomplir. De manière similaire, pour chacune de ces sous-fonctions de récompenses, on crée des sous-fonctions de critiques qui leur sont associées, ainsi qu’une fonction de combinaison des critiques. L’agent sélectionnera ses actions de manière à maximiser cette fonction de combinaison des

sous-critiques lors de son apprentissage. Grâce à cette méthode, pour chaque observation et action de l'agent, on obtiendra une prédiction sur les sous-objectifs qu'il tente de maximiser. C'est cette *prédiction* qui servira d'explication pour l'observateur.

3.4 Trajectoires de l'agent

Les trajectoires représentent une séquence d'interactions entre l'agent et son environnement. En analysant ces trajectoires, l'observateur peut déduire la dynamique sous-jacente de l'évolution de l'agent au sein de son environnement. Selon le type de trajectoire utilisé (voir 2.2), le sujet de l'explication varie.

L'utilisation des trajectoires post-entraînement pour expliquer la stratégie est systématiquement employée afin d'obtenir une explication sur la stratégie de l'agent, même en dehors du cadre du XRL. Cette méthode simple, peut être perfectionnée grâce à des techniques telles que la *décomposition hiérarchique* et l'*extraction par analyse du critique*.

Pour expliquer le processus d'apprentissage d'un agent, on utilise des trajectoires d'entraînement ayant recours à la technique d'*extraction par comparaisons multi-critiques*.

3.4.1 *Décomposition hiérarchique*

Les agents hiérarchiques sont des agents composés d'un ensemble de **sous-politiques** spécialisées pour des tâches précises dans des sous-ensembles d'états de l'environnement. Cette méthode permet de répartir l'optimisation de la fonction de récompense sur plusieurs modèles et simplifie grandement l'apprentissage. Par ailleurs, ce **découpage** de la fonction de politique peut également apporter de l'explicabilité dans la stratégie de l'agent [3]. Si une sous-politique est choisie, cela signifie que l'agent va mettre en place un comportement spécialisé. Cette spécialisation dans le comportement nous donne des informations sur la stratégie mise en place par l'agent pour maximiser la fonction de récompense. La *trajectoire* de l'agent, ainsi que la *sous-politique* sélectionnée pour chaque action sont utilisées comme informations interprétables.

3.4.2 *Extraction par analyse du critique*

Dans cette technique, on utilise les prédictions du critique pour déterminer quelle **trajectoire présenter à l'observateur** [26]. Ainsi, on sélectionne un ensemble d'états provenant de multiples trajectoires. Pour un même état, on examine l'ensemble des actions possibles. L'état se voit attribuer comme valeur la différence entre la prédiction du critique avec l'action la plus basse et la prédiction du critique avec l'action la plus haute. Ensuite, on choisit de présenter à l'observateur les trajectoires qui contiennent l'état avec la valeur la plus élevée selon cette procédure. On considère que ces trajectoires représentent les moments **critiques** de l'agent et sont donc pertinents à soumettre à l'analyse de l'observateur. Pour cette technique les *trajectoires* sélectionnées représentent l'information intepretable pour l'observateur.

3.4.3 *Extraction par comparaisons multi-critiques*

Dans cette approche, l'objectif principal est de **pondérer les interactions** d'entraînement entre l'agent et l'environnement, selon leurs influences sur le processus d'apprentissage. Pour atteindre cet objectif, plusieurs fonctions de critique sont crées à l'aide de l'apprentissage *off-policy*. Un apprentissage *off-policy* est un processus d'apprentissage par renforcement qui permet à une politique d'apprendre à partir de données générées par une politique différente [30]. Chaque fonction de critique est entraînée sur un sous-ensemble du jeu de données.

La fonction de critique de référence est élaborée en exploitant l'intégralité du jeu de données. Ensuite, pour chaque interaction agent-environnement, une fonction de critique est de nouveau entraînée en excluant spécifiquement cette interaction. La **disparité de prédiction** entre une fonction de critique et la fonction de critique de référence représente l'impact de la suppression de cette interaction sur l'apprentissage. Par ce processus, une valeur est attribuée à chaque interaction en fonction de cette différence : plus la disparité est grande, plus la valeur attribuée à l'interaction est élevée.

Une valeur plus élevée indique que l'interaction est considérée comme plus importante dans le processus d'apprentissage. Ces *valeurs* sur les *interactions* agissent comme des explications pour l'observateur, offrant un moyen savoir l'importance de l'interaction agent-environnement dans le cadre du processus d'apprentissage.

3.5 Contrefactuelles de l'observation

Dans le contexte de l'apprentissage par renforcement, les explications contrefactuelles sont utilisées pour générer de nouvelles observations (cf. figure 6). Cette nouvelle observation distincte de la première conduit l'agent à adopter un comportement alternatif. Une explication contrefactuelle vise à expliquer le processus décisionnel de l'agent. La modification de l'observation renseigne l'observateur sur les ajustements nécessaires pour que l'agent adopte un comportement différent. La méthode dont on dispose pour créer des contrefactuelles est l'*utilisation de l'espace latent pour la génération de contrefactuelles*.

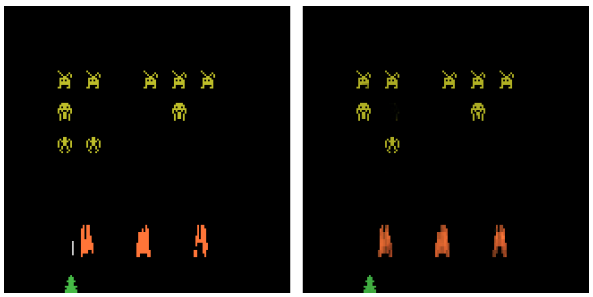


FIGURE 6 – Contrefactuelle sur le comportement de l'agent pour le jeu Space Invader [21]. Sur l'image de gauche l'observation provoque l'action "LEFT", tandis que pour l'observation de l'image de droite, l'action sélectionnée par l'agent est "RIGHT_FIRE".

Dans cette technique, on a recours à des **modèles génératifs** [4] pour exploiter la représentation latente. Il s'agit d'un type de modèle qui apprend à générer de nouvelles données en imitant les données d'entraînement. Dans cette technique, les modèles génératifs cherchent à produire des observations alternatives susceptibles de modifier les actions de l'agent [21].

La difficulté dans ce type de procédé réside dans le fait que pour servir d'explication, les contrefactuelles doivent être à la fois **proches** et **réalisables**. Une contrefactuelle proche signifie une contrefactuelle qui ne diffère pas trop de l'observation d'origine. Une observation contrefactuelle réalisable est une contrefactuelle qui fait sens comme observation dans l'environnement considéré.

3.6 Modèles de l'interaction agent-environnement

La **modélisation** des interactions entre un agent et son environnement permet de comprendre la manière dont les décisions de l'agent sont prises et quels impacts elles ont sur l'environnement (cf. figure 7). En incorporant ces interactions dans un modèle explicatif, on peut mieux appréhender les relations complexes entre l'agent et son environnement, identifiant ainsi les facteurs déterminants dans le processus décisionnel. De par la gestion de l'interaction **agent-environnement**, ce type d'explication a pour objectif d'expliquer la stratégie de l'agent. Les méthodes utilisées afin d'avoir une explication sous forme d'interactions agent-environnement sont : *apprentissage d'un réseau bayésien* et *exploitation de l'espace latent pour la génération d'un MDP*.

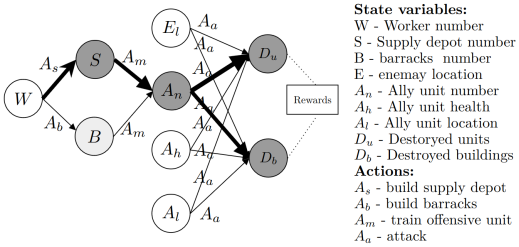


FIGURE 7 – Modèle de l'interaction agent-environnement pour le jeu vidéo Starcraft 2 [17]. Les sommets représentent des variables aléatoires. Les liens représentent les causalités entre les variables aléatoires.

3.6.1 *Apprentissage d'un réseau bayésien*

Il est possible d'observer l'évolution d'un agent dans un environnement sous la forme d'un ensemble de **variables aléatoires** liées entre elles par un **réseau bayésien** [17]. On définit des variables aléatoires comme représentant les aspects externes (l'environnement) et les aspects internes à l'agent (les actions). En étudiant les trajectoires, il est alors possible de déduire des liens de causalité entre toutes ces variables aléatoires. À la fin du processus, on obtient un graphe de liens de causalité pondérés qui servira d'explication pour l'observateur (cf. figure 7). C'est ce *réseau bayésien* représentant les interactions entre l'agent et l'environnement qui sera utilisé comme explication.

3.6.2 *Utilisation de l'espace latent pour la génération d'un MDP*

Chaque couche d'un réseau de neurones fonctionne comme une projection d'un espace vers un autre [36]. En parcourant les couches successives d'un réseau de neurones, l'information prend une forme de plus en plus abstraite. On peut supposer que deux vecteurs proches dans cet espace de projection sont également proches en terme de perception pour l'agent. De ce constat, il est possible de **redéfinir un MDP** en passant par la **représentation abstraite** des dernières couches d'un réseau de neurones. Une fois le nouveaux MDP finalisé, on obtient un modèle d'interaction entre l'agent et son environnement. Ce modèle représenté sous forme d'un *graphe*, sera utilisé comme explication.

4 Discussion

Pour l'XRL, les premières tentatives de classification datent du début des années 2020 [23, 11]. Pendant cette période, les techniques de classification reposaient en grande partie sur les approches XAI généralistes, à savoir la division entre deux catégories d'explication : "locale" et "globale". Les explications locales fournissent des informations spécifiques à une entrée particulière du système et répondent à la question suivante : quel est l'élément déterminant, dans cette entrée, pour le choix du système ? Les explications globales ont quant à elles pour objectif de fournir des informations sur la manière dont le système effectue ses choix, mais cette fois-ci sur l'ensemble des entrées possibles.

Cette première catégorisation ne saisit pas les spécificités de l'apprentissage par renforcement, créant ainsi des imprécisions dans le système de classification. Pour remédier à cette lacune, une nouvelle génération d'articles, parus en 2023 [18, 24], a intégré les particularités de l'apprentissage par renforcement dans son système de classification.

L'article de 2023 proposé par Milani et al. [18], suggère que la classification doit être fondée sur ce que les méthodes cherchent à expliquer. Trois catégories de sujets d'explication sont décrites : "importance des caractéristiques" (*Feature Importance*, FI), "explication au niveau politique" (*Policy Level*, PL), et "explication du processus d'apprentissage et du processus décisionnel de markov" (*Learning Process and markov decision process*). Ce type de classification est en grande partie une adaptation de l'approche XAI classique pour le domaine du XRL : les catégories "locale" et "globale" sont remplacées dans cette classification par FI et PL. Comme différence majeure, on peut noter l'ajout d'une nouvelle catégorie "processus d'apprentissage et du MDP" qui regroupe un ensemble disparate d'éléments, à savoir : les explications de l'apprentissage ainsi que toutes les méthodes qui tirent leur explication du critique. Cette dernière catégorie s'inscrit difficilement dans la logique initialement proposée par les auteurs puisqu'elle contient en elle-même des explications se référant au MDP, alors que jusqu'à présent, la classification se faisait uniquement par rapport au sujet des explications.

L'article de Qing et al. [24] quant à lui utilise des méthodes de classification fondées sur les sources. Pour cette équipe de recherche, les grandes catégories à considérer

sont : les explications du "modèle de l'agent" (*Agent Model-Explaining*), les explications des "récompenses" (*Reward-Explaining*), les explications des "états" (*State-Explaining*) et les explications des "tâches" (*Task-Explaining*). Cependant, encore une fois, des approximations fragilisent la structure de classification. Par exemple, la catégorie "explication des états" n'est pas réellement une source d'explication : on ne peut rien expliquer de la représentation d'un état en lui-même. C'est l'interprétation faite par la politique ou par le critique de l'état qui produit de l'explicabilité.

Ces deux modèles présentent quelques imprécisions dans leurs classifications lorsqu'ils sont examinés individuellement. Cependant, cette situation s'explique par le fait qu'ils ont adopté chacun une seule approche pour leurs classifications. En intégrant les différentes approches proposées par ces deux articles, nous avons élaboré une classification plus complète qui prend en compte et intègre les points de vue respectifs de chacun.

Une observation notable émanant de l'analyse de l'état de l'art, à travers le prisme de notre système de classification, réside dans le fait que très peu de techniques se donnent pour mission d'expliciter le processus d'apprentissage lui-même. Cette lacune peut être attribuée à l'idée que, dans une perspective d'application concrète dans le monde réel, de telles explications peuvent sembler peu pertinentes, étant donné qu'elles ne fournissent pas d'informations détaillées sur le comportement de la politique adoptée. Cependant, nous jugeons pertinent d'explorer cet aspect. En effet, identifier les stratégies viables dans un environnement donné et comprendre les raisons pour lesquelles une stratégie est privilégiée par rapport aux autres peut offrir des informations significatives. Bien que ces informations ne soient pas directement utilisables pour l'intégration opérationnelle des agents, elles peuvent servir à comparer les différents algorithmes d'apprentissage par renforcement entre eux.

5 Conclusion

Notre article a mis en lumière les enjeux cruciaux à propos de l'apprentissage par renforcement explicable, dont le but est de faciliter l'intégration d'agents formés par renforcement dans des situations du monde réel. Il souligne également la nécessité de structurer ce domaine de manière efficiente, compte tenu de sa dynamique croissante. À cette fin, nous avons élaboré une nouvelle taxinomie en fusionnant deux perspectives existantes [18, 24].

Cette classification repose sur des aspects spécifiques de l'apprentissage par renforcement, mettant en exergue les concepts de **source d'explication**, d'**explication** et de **sujet d'explicabilité**. L'état de l'art présenté dans cet article démontre la pertinence de notre nouveau système de classification. Cet outil méthodologique offre la possibilité d'organiser de manière plus efficace les futures recherches dans ce domaine.

Cependant, un élément central demeure à discuter en détail : le type d'audience auquel une explication est destinée. Comme illustré dans la figure 3, une explication est élaborée pour être interprétée par un observateur. Généralement, on peut classer un observateur dans l'une des trois grandes catégories d'audience : les experts en intelligence artificielle, les spécialistes du domaine d'application du modèle d'IA, et les non-initiés, c'est-à-dire des observateurs qui ne possèdent aucune connaissance ni en intelligence artificielle ni dans le domaine d'application [18].

Pour évaluer l'impact d'un processus explicatif sur l'un de ses publics cibles, il est impératif que la communauté de chercheurs établisse des protocoles d'évaluation visant à mesurer la pertinence d'une explication. Ce domaine a été exploré par Milani et al. [18], dont le travail permet l'introduction d'une classification des différentes méthodes d'évaluation ainsi que diverses métriques permettant de quantifier l'efficacité d'une explication. Des recherches approfondies dans cette direction sont nécessaires, notamment pour comparer différentes approches d'explicabilité et évaluer leur pertinence en fonction du public visé.

Références

[1] Matthieu Bellucci, Nicolas Delestre, Nicolas Malandain, and Cecilia Zanni-Merk. Towards a terminology for a fully contextualized xai. *Procedia Computer Science*, 192 :241–250, 2021. **Article**.

[2] Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv :1912.06680*, 2019. **Article**. **Vidéo**.

[3] Benjamin Beyret, Ali Shafti, and A. Aldo Faisal. Dot-to-dot : Explainable hierarchical reinforcement learning for robotic manipulation. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, November 2019. **Article**.

[4] Sam Bond-Taylor, Adam Leach, Yang Long, and Chris G Willcocks. Deep generative modelling : A comparative review of vaes, gans, normalizing flows, energy-based and autoregressive models. *IEEE transactions on pattern analysis and machine intelligence*, 2021. **Article**.

[5] Damien Garreau and Ulrike von Luxburg. Explaining the explainer : A first theoretical analysis of lime. In Silvia Chiappa and Roberto Calandra, editors, *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pages 1287–1296. PMLR, 26–28 Aug 2020. **Article**.

[6] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. **Livre**.

[7] Sam Greydanus, Anurag Koul, Jonathan Dodge, and Alan Fern. Visualizing and understanding atari agents, 2018. **Article**. **Code**.

[8] David Gunning and David Aha. Darpa's explainable artificial intelligence (xai) program. *AI magazine*, 40(2) :44–58, 2019. **Article**.

[9] David Gunning, Eric Vorm, Yunyan Wang, and Matt Turek. Darpa's explainable ai (xai) program : A retrospective. *Authorea Preprints*, 2021. **Article**.

[10] Matteo Hessel, Joseph Modayil, Hado van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow : Combining improvements in deep reinforcement learning, 2017. **Article**.

[11] Alexandre Heuillet, Fabien Couthouis, and Natalia Díaz-Rodríguez. Explainability in deep reinforcement learning, 2020. **Article**.

[12] Tobias Huber, Dominik Schiller, and Elisabeth André. Enhancing explainability of deep reinforcement learning through selective layer-wise relevance propagation. In *KI 2019 : Advances in Artificial Intelligence : 42nd German Conference on AI, Kassel, Germany, September 23–26, 2019, Proceedings 42*, pages 188–202. Springer, 2019. **Article**.

[13] Hidenori Itaya, Tsubasa Hirakawa, Takayoshi Yamashita, Hironobu Fujiyoshi, and Komei Sugiura. Visual explanation using attention mechanism in actor-critic-based deep reinforcement learning, 2021. **Article**.

[14] Zoe Juozapaitis, Anurag Koul, Alan Fern, Martin Erwig, and Finale Doshi-Velez. Explainable reinforcement learning via reward decomposition. In *IJCAI/ECAI Workshop on explainable artificial intelligence*, 2019. **Article**.

[15] Zhengxian Lin, Kim-Ho Lam, and Alan Fern. Contrastive explanations for reinforcement learning via embedded self predictions, 2021. **Article**. **Code**.

[16] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments, 2020. **Article**.

[17] Prashan Madumal, Tim Miller, Liz Sonenberg, and Frank Vetere. Explainable reinforcement learning through a causal lens. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 2493–2500, 2020. **Article**.

[18] Stephanie Milani, Nicholay Topin, Manuela Veloso, and Fei Fang. Explainable reinforcement learning : A survey and comparative review. *ACM Computing Surveys*, 2023. **Article**.

[19] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning, 2013. **Article**.

[20] Alexander Mott, Daniel Zoran, Mike Chrzanowski, Daan Wierstra, and Danilo Jimenez Rezende. Towards interpretable reinforcement learning using attention augmented agents. *Advances in neural information processing systems*, 32, 2019. **Article**.

[21] Matthew L Olson, Roli Khanna, Lawrence Neal, Fuxin Li, and Weng-Keen Wong. Counterfactual state explanations for reinforcement learning agents via generative deep learning. *Artificial Intelligence*, 295 :103455, 2021. **Article**. **Code**.

[22] Ali Payani and Faramarz Fekri. Inductive logic programming via differentiable deep neural logic networks, 2019. **Article**. **Code**.

[23] Erika Puiutta and Eric MSP Veith. Explainable reinforcement learning : A survey, 2020. **Article**.

[24] Yunpeng Qing, Shunyu Liu, Jie Song, Huiqiong Wang, and Mingli Song. A survey on explainable reinforcement learning : Concepts, algorithms, challenges. *arXiv preprint arXiv :2211.06665*, 2022. **Article**. **GitHub**.

[25] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. **Article**.

[26] Pedro Sequeira and Melinda Gervasio. Interestingness elements for explainable reinforcement learning : Understanding agents' capabilities and limitations. *Artificial Intelligence*, 288 :103367, November 2020. **Article**.

[27] Alexander Sieusahai and Matthew Guzdial. Explaining deep reinforcement learning agents in the atari domain through a surrogate model, 2021. **Article**.

[28] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Simonyan, and Demis Hassabis. Mastering chess and shogi by self-play with a general reinforcement learning algorithm, 2017. **Article**.

[29] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks : Visualising image classification models and saliency maps, 2014. **Article**. **Code**.

[30] Richard S Sutton and Andrew G Barto. *Reinforcement learning : An introduction*. MIT press, 2018. **Livre**.

[31] Richard S Sutton, Joseph Modayil, Michael Delp, Thomas Degris, Patrick M Pilarski, Adam White, and Doina Precup. Horde : A scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction. In *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pages 761–768, 2011. **Article**.

[32] Nicholay Topin, Stephanie Milani, Fei Fang, and Manuela Veloso. Iterative bounding mdps : Learning interpretable policies via non-interpretable methods. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 9923–9931, 2021. **Article**.

[33] Dweep Trivedi, Jesse Zhang, Shao-Hua Sun, and Joseph J. Lim. Learning to synthesize programs as interpretable and generalizable policies, 2022. **Article**. **Code**.

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023. **Article**.

[35] Laurens Weitkamp, Elise van der Pol, and Zeynep Akata. Visual rationalizations in deep reinforcement learning for atari games, 2019. **Article**.

[36] Tom Zahavy, Nir Ben Zrihem, and Shie Mannor. Graying the black box : Understanding dqns, 2017. **Article**.

Session 5 : Argumentation

Actes JIAF 2024

Raisonnement Approximatif pour l'Acceptabilité des Arguments en Argumentation Abstraite

Jérôme Delobelle[1] **Jean-Guy Mailly**[2] **Julien Rossit**[1]

[1] Université Paris Cité, LIPADE, F-75006 Paris, France
[2] Université Toulouse Capitole, IRIT, Toulouse, France
{jerome.delobelle,julien.rossit}@u-paris.fr, jean-guy.mailly@irit.fr

Résumé

Définir des algorithmes efficaces pour des tâches de raisonnement complexe est un enjeu majeur dans l'argumentation abstraite. Une approche récente propose de définir des algorithmes approximatifs fournissant une réponse qui peut ne pas toujours être correcte, mais qui surpasse les algorithmes exacts en termes de temps de calcul. Plus particulièrement, elle propose d'utiliser la sémantique de base, qui est calculable en temps polynomial, comme point de départ pour déterminer si les arguments sont acceptés (de manière crédule ou sceptique) par rapport à différentes sémantiques. Dans cet article, nous allons plus loin dans cette idée en définissant différentes approches permettant d'évaluer l'acceptabilité des arguments qui ne sont ni dans l'extension de base, ni attaqués par celle-ci. Nous avons implémenté ces approches afin d'établir une évaluation empirique de celles-ci. Ce papier est un résumé du papier publié à la conférence ECSQARU 2023 [2].

1 État de l'art

Formellement, un **système d'argumentation** (AF, pour *Argumentation Framework*) [3] est un couple $\mathcal{F} = \langle \mathcal{A}, \mathcal{R} \rangle$ où $\mathcal{A}$ est un ensemble fini d'arguments et $\mathcal{R} \subseteq \mathcal{A} \times \mathcal{A}$ est la relation d'attaque entre arguments. Des *sémantiques à base d'extensions* ont été définies pour déterminer l'acceptabilité des arguments. Elles sont en général basées sur deux principes : un ensemble acceptable $E \subseteq \mathcal{A}$ doit 1) être *sans conflit* ($\texttt{cf}(\mathcal{F})$) : $\forall a, b \in E, (a, b) \notin \mathcal{R}$; 2) *défendre tous ses éléments* : $\forall a \in E, \forall b \in \mathcal{A}$, si $(b, a) \in \mathcal{R}$ alors $\exists c \in E$ tel que $(c, b) \in \mathcal{R}$. Un ensemble qui satisfait ces deux propriétés est appelé *admissible* ($\texttt{ad}(\mathcal{F})$). Les extensions *complètes* ($\texttt{co}(\mathcal{F})$) sont les ensembles admissibles qui contiennent tous les arguments qu'ils défendent ; l'extension *de base* ($\texttt{gr}(\mathcal{F})$) est l'extension complète minimale pour l'inclusion ; les extensions *préférées* ($\texttt{pr}(\mathcal{F})$) sont les extensions complètes maximales pour l'inclusion ; les extensions *stables* ($\texttt{stb}(\mathcal{F})$) sont les ensembles sans conflit qui attaquent tous les arguments n'étant pas dans l'ensemble. Étant donné une sémantique à base d'extensions σ, un argument est crédulement (resp. sceptiquement) accepté par rapport à σ s'il appartient à au moins une (resp. toutes les) extension(s). Les problèmes liés au fait de savoir si un argument est crédulement ou sceptiquement accepté sont DC-σ et DS-σ respectivement.

Notons que le calcul de la sémantique de base ($\texttt{gr}$) est réalisé en temps polynomial, alors que les autres sémantiques ont une complexité plus importante [4]. Ceci a conduit à une évaluation empirique montrant la similarité non-négligeable entre la sémantique de base et les autres sémantiques à base d'extensions [1]. Cela à amené au développement du solver approximatif Harper++ [5] pour les problèmes DC-σ et DS-σ. Pour les sémantiques usuelles, un argument qui appartient à l'extension de base sera nécessairement dans toutes les extensions, tandis qu'un argument attaqué par un élément de l'extension de base ne sera dans aucune extension. Il convient donc de se concentrer sur le troisième type d'arguments, qu'on notera UNDEC($\mathcal{F}$) et qui correspond à l'ensemble des arguments n'étant ni dans $\texttt{gr}(\mathcal{F})$, ni attaqué par celle-ci. Étant donné un AF $\mathcal{F} = \langle \mathcal{A}, \mathcal{R} \rangle$ et $a \in \mathcal{A}$, l'algorithme de Harper++ répond donc :YES si $a \in \texttt{gr}(\mathcal{F})$, NO si a est attaqué par $b \in \texttt{gr}(\mathcal{F})$, et pour les éléments de UNDEC($\mathcal{F}$) l'algorithme répond YES si le problème est DC-σ, et NO si le problème est DS-σ.

2 ARIPOTER

Notre but est d'étendre Harper++, notamment sur l'évaluation des arguments appartenant à UNDEC($\mathcal{F}$), en introduisant deux familles de solveurs approximatifs nommées ARIPOTER (ARgumentatIon apPrOximaTE Reasoning).

Le solveur ARIPOTER-degrees capture l'idée qu'un argument qui attaque plus d'arguments que le nombre de ses attaquants directs a de bonnes chances de se défendre, et donc d'être accepté. Ainsi, un argument de UNDEC($\mathcal{F}$) sera considéré comme accepté si le nombre d'arguments qu'il attaque (i.e. *out-degree*) est au moins k fois plus élevé que le nombre d'arguments qui l'attaquent (i.e. *in-degree*).

Définition 1 *Étant donné un AF $\mathcal{F} = \langle \mathcal{A}, \mathcal{R} \rangle$, $a \in \mathcal{A}$ et $k \in \mathbb{R}$, la fonction $Acc^{Out/In}$ est définit comme suit :*

$$Acc^{Out/In}(\mathcal{F}, a, k) = \begin{cases} YES & si\ a \in gr(\mathcal{F})\ ou \\ & (a \in UNDEC(\mathcal{F})\ et \\ & |a^+| \geq k \times |a^-|), \\ NO & sinon. \end{cases}$$

avec $a^+ = \{b \in \mathcal{A} \mid (a, b) \in \mathcal{R}\}$ et $a^- = \{b \in \mathcal{A} \mid (b, a) \in \mathcal{R}\}$ l'ensemble des arguments attaqués par a ou attaquant a respectivement.

Notre seconde approche, ARIPOTER-hcat, accepte les arguments de UNDEC($\mathcal{F}$) dont le score retourné par la sémantique h-categorizer est supérieur à une valeur τ. Cette sémantique graduée utilise une fonction dont le but est d'attribuer une valeur qui capture la force relative d'un argument en tenant compte de la force de ses attaquants. Formellement, étant donné un AF $\mathcal{F} = \langle \mathcal{A}, \mathcal{R} \rangle$ et $a \in \mathcal{A}$, $\texttt{h-cat}(\mathcal{F}, a) = \frac{1}{1+\sum_{b \in a^-} \texttt{h-cat}(\mathcal{F}, b)}$.

Définition 2 *Étant donné un AF $\mathcal{F} = \langle \mathcal{A}, \mathcal{R} \rangle$, $a \in \mathcal{A}$ et $\tau \in [0, 1]$, la fonction Acc^{h-cat} est définit comme suit :*

$$Acc^{h-cat}(\mathcal{F}, a, \tau) = \begin{cases} YES & si\ a \in gr(\mathcal{F})\ ou \\ & (a \in UNDEC(\mathcal{F})\ et \\ & h\text{-}cat(\mathcal{F}, a) \geq \tau), \\ NO & sinon. \end{cases}$$

avec $h\text{-}cat(\mathcal{F}, a) = \frac{1}{1+\sum_{b \in a^-} h\text{-}cat(\mathcal{F}, b)}$.

3 Analyse Empirique

Concernant les instances utilisés pour évaluer et comparer nos deux approches et Harper++, nous avons sélectionné les deux familles d'instances suivantes :

- `iccma19` : ensemble des instances difficiles "2019" de la compétition ICCMA 2021, composé de 107 AF, avec un nombre d'arguments entre 102 et 8034 arguments.
- `randomAF` : ensemble de 9460 AF réparties entre trois familles de graphes (Erdös-Rényi, Barabási-Albert et Watts-Strogatz) utilisés lors des compétitions ICCMA. Le nombre d'attaques, de cycles et d'arguments (entre 10 et 100) varie selon la famille étudiée.

Notre étude s'est concentrée sur quatre problèmes de décision : DC-`stb`, DS-`stb`, DC-`pr` and DS-`pr`. La table 1 résume la précision des différents solveurs approximatifs sur `randomAF` et `iccma19`. Observons que les deux solveurs ARIPOTER retournent une bien meilleure précision que Harper++ pour les instances `randomAF` avec des résultats à plus de 93% pour tous les problèmes étudiés excepté pour DS-`stb` (78%). Pour les instances `iccma19`, les résultats de nos deux solveurs sont très proches pour les quatre problèmes étudiés. En comparaison avec les résultats précédents, la précision est légèrement plus faible pour les problèmes DC, mais reste de l'ordre de 80% de réponses correctes. Cependant, cette diminution est plus importante pour le problème DS-`stb` où nos solveurs obtiennent une précision qui est de l'ordre de 54% contrairement à Harper++ qui est de plus de 82%.

Instances	Solveur	DC-pr	DC-stb	DS-pr	DS-stb
randomAF	Harper++	0.4161	0.414	0.9794	0.4738
	ARIPOTER-degrees	0.8306 (8)	0.8306 (10)	**0.9797** (8)	**0.7802** (0.1)
	ARIPOTER-hcat	**0.9321** (0.5)	**0.9304** (0.5)	0.9794 (1)	0.7648 (0.1)
iccma19	Harper++	0.7549	0.757	**0.9712**	**0.8269**
	ARIPOTER-degrees	**0.7941** ($\lvert\mathcal{A}\rvert$)	**0.8131** ($\lvert\mathcal{A}\rvert$)	**0.9712** ($\lvert\mathcal{A}\rvert$)	0.5481 (0.1)
	ARIPOTER-hcat	**0.7941** (0.5)	**0.8131** (0.5)	**0.9712** (1)	0.5385 (0.1)

TABLE 1 – Comparaison de la précision des trois solveurs approximatifs pour `randomAF` et `iccma19`, avec les valeurs de k (pour ARIPOTER-degrees) ou de τ (pour ARIPOTER-hcat) retournant les meilleurs résultats entre parenthèses.

Après ces résultats très encourageants, nous prévoyons d'étendre notre approche à d'autres sémantiques graduées et d'étudier plus en détails la précision de ces solveurs sur d'autres familles de graphes.

Remerciements. Ce travail est supporté par l'ANR (AGGREEY ANR-22-CE23-0005 et AIDAL ANR-22-CPJ1-0061-01).

Références

[1] Federico Cerutti, Matthias Thimm, and Mauro Vallati. An experimental analysis on the similarity of argumentation semantics. *Arg. Comput.*, 11(3) :269–304, 2020.

[2] Jérôme Delobelle, Jean-Guy Mailly, and Julien Rossit. Revisiting approximate reasoning based on grounded semantics. In *Proc. of ECSQARU'23*, 2023.

[3] Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artif. Intell.*, 77(2) :321–358, 1995.

[4] Wolfgang Dvorák and Paul E. Dunne. Computational problems in formal argumentation and their complexity. In *Handbook of Formal Argumentation*, pages 631–688. College Publications, 2018.

[5] Matthias Thimm. Harper++ : Using grounded semantics for approximate reasoning in abstract argumentation. `http://argumentationcompetition.org/2021/downloads/harper++.pdf`, 2021.

Actes JIAF 2024

Gestion des supports dans les systèmes d'argumentation incomplets

Marie-Christine Lagasquie-Schiex[1] **Jean-Guy Mailly**[2] **Antonio Yuste-Ginel**[3]

[1] Université Paul Sabatier, IRIT, Toulouse, France
[2] Université Toulouse Capitole, IRIT, Toulouse, France
[3] Complutense University of Madrid, Madrid, Espagne
lagasq@irit.fr jean-guy.mailly@irit.fr antoyust@ucm.es

Résumé

L'intérêt croissant de la communauté pour les généralisations du cadre d'argumentation abstraite de Dung a mené récemment à la combinaison de deux de ces généralisations : les systèmes d'argumentation bipolaires (BAF pour *Bipolar Argumentation Frameworks*), où une relation de support entre arguments est ajoutée, et les systèmes d'argumentation incomplets (IAF pour *Incomplete Argumentation Frameworks*), où l'existence des arguments et attaques peut être incertaine, dont la combinaison a mené à la définition des systèmes d'argumentation bipolaires incomplets (IBAF pour *Incomplete Bipolar Argumentation Frameworks*). Cet article poursuit l'étude de cette combinaison, en (i) proposant une analyse des notions de complétions existantes (c'est-à-dire les possibles suppressions d'incertitude utilisées dans les IBAF pour raisonner sur l'acceptabilité des arguments) ; (ii) proposant, motivant et étudiant de nouvelles notions de complétions ; (iii) fournissant des résultats de complexité concernant les problèmes d'acceptabilité des arguments associés aux IBAF ; (iv) encodant ces problèmes de raisonnement en logique dynamique.

Les systèmes d'argumentation abstraits ont reçu un intérêt croissant depuis le travail originel de Dung [2], qui définit un système d'argumentation comme un graphe orienté $\mathcal{F} = \langle \mathcal{A}, \mathcal{R} \rangle$ tel que $\mathcal{R}$ représente une relation d'attaque entre les arguments $\mathcal{A}$, et tel que l'acceptabilité des arguments est évaluée au moyen de la notion d'extensions, c'est-à-dire d'ensembles d'arguments collectivement acceptables. On considère les sémantiques classiques comme admissible (ad), complète (co), préférée (pr), stable (st) et fondée (gr). De nombreuses généralisations de ce cadre ont été proposées afin d'en enrichir l'expressivité, notamment à travers l'addition de nouvelles interactions entre les arguments comme une relation de support qui peut être, notamment, nécessaire (nec) [8] ou déductif (ded) [1], ou encore l'intégration d'incertitude sur la présence des arguments et attaques [7]. Récemment, deux travaux indépendant ont simultanément proposé de combiner ces deux généralisations, menant à la définition des systèmes d'argumentation bipolaires incomplets [3, 5]. Cet article est un résumé de [6].

1 Système d'Argumentation Bipolaire Incomplet

Definition 1 *Un* Système d'Argumentation Bipolaire Incomplet (IBAF) *est un tuple* $\mathcal{IB} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{R}, \mathcal{R}^?, \mathcal{S}, \mathcal{S}^? \rangle$*, où* $\mathcal{A}, \mathcal{A}^?$ *sont des ensembles disjoints d'arguments et* $\mathcal{R}, \mathcal{R}^?, \mathcal{S}, \mathcal{S}^?$ *sont des relations disjointes entre arguments. Les éléments de* $\mathcal{R}$ *et* $\mathcal{S}$ *(resp.* $\mathcal{R}^?$ *et* $\mathcal{S}^?$*) représentent les attaques et les supports certains (resp. incertains).*

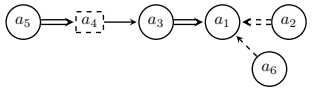


FIGURE 1 – $\mathcal{IB} = \langle \{a_1, a_2, a_3, a_5, a_6\}, \{a_4\}, \{(a_4, a_3)\}, \{(a_6, a_1)\}, \{(a_3, a_1), (a_5, a_4)\}, \{(a_2, a_1)\} \rangle$.

Pour raisonner avec un IBAF, on peut adapter la notion de complétion des systèmes d'argumentation incomplets (IAF), qui sont des systèmes d'argumentation (classiques) correspondant aux différentes façons de « résoudre » l'incertitude présente dans l'IAF. Cependant, la relation de support entre arguments a des spécificités qui nous amènent à proposer différentes versions de la notion de complétion pour les IBAF.

Definition 2 *Étant donnés* $\mathcal{IB} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{R}, \mathcal{R}^?, \mathcal{S}, \mathcal{S}^? \rangle$ *et* σ*, un BAF* $\mathcal{B} = \langle \mathcal{A}_c, \mathcal{R}_c, \mathcal{S}_c \rangle$ *est :*

1. *une* pla-complétion de $\mathcal{IB}$ *(*plain completion*) ssi* $\mathcal{A} \subseteq \mathcal{A}_c \subseteq \mathcal{A} \cup \mathcal{A}^?$, $\mathcal{R} \cap (\mathcal{A}_c \times \mathcal{A}_c) \subseteq \mathcal{R}_c \subseteq (\mathcal{R} \cup \mathcal{R}^?) \cap (\mathcal{A}_c \times \mathcal{A}_c)$, $\mathcal{S} \cap (\mathcal{A}_c \times \mathcal{A}_c) \subseteq \mathcal{S}_c \subseteq (\mathcal{S} \cup \mathcal{S}^?) \cap (\mathcal{A}_c \times \mathcal{A}_c)$ *;*
2. *une* t-complétion de $\mathcal{IB}$ selon σ avec $\mathsf{t} \in \{\mathsf{nec}, \mathsf{ded}\}$ *(complétion sémantique –* **déd***uctive ou* **néc***essaire) ssi* $\mathcal{B}$ *est une* pla*-complétion et* $\forall (a,b) \in \mathcal{S}, \forall E \in \sigma^{\mathsf{t}}(\mathcal{B})$ *: (i)* $\mathsf{t} = \mathsf{nec}$ *et* $b \in E \Rightarrow a \in E$ *et (ii)* $\mathsf{t} = \mathsf{ded}$ *et* $a \in E \Rightarrow b \in E$ *;*
3. *une* t-complétion de $\mathcal{IB}$ avec $\mathsf{t} \in \{c\mathsf{ded}, c\mathsf{nec}\}$ *(complétion* **c***lose –* **déd***uctive ou* **néc***essaire) ssi* $\mathcal{B}$ *est une* pla*-complétion et* $\forall (a,b) \in \mathcal{S}$ *: (i) si* $\mathsf{t} = c\mathsf{nec}$ *et* $b \in \mathcal{A}_c$*, alors* $a \in \mathcal{A}_c$ *et (ii) si* $\mathsf{t} = c\mathsf{ded}$ *et* $a \in \mathcal{A}_c$*, alors* $b \in \mathcal{A}_c$*.*

Pour $\mathsf{t} \in \{\mathsf{pla}, \mathsf{nec}, \mathsf{ded}, c\mathsf{ded}, c\mathsf{nec}\}$, $\mathsf{comp}^{\mathsf{t}}_{\sigma}(\mathcal{IB})$ est l'ensemble des t-complétions de $\mathcal{IB}$ pour la sémantique σ ; si σ n'est pas nécessaire pour ce type de complétion, la notation est simplifiée en $\mathsf{comp}^{\mathsf{t}}(\mathcal{IB})$. On note σ^{t_2}-$\mathsf{t}_1(\mathcal{IB})$ l'ensemble des extensions de $\mathcal{IB}$ avec l'interprétation t_2 pour le support ($\mathsf{t}_2 \in \{\mathsf{nec}, \mathsf{ded}\}$), la sémantique σ ($\sigma \in \{\mathsf{pr}, \mathsf{gr}, \mathsf{co}, \mathsf{st}\}$) et le type de complétions t_1 ($\mathsf{t}_1 \in \{\mathsf{pla}, \mathsf{nec}, \mathsf{ded}, c\mathsf{nec}, c\mathsf{ded}\}$). Donc σ^{t_2}-$\mathsf{t}_1(\mathcal{IB}) = \{E \subseteq \mathcal{A} \cup \mathcal{A}^? | \exists \mathcal{B} \in \mathsf{comp}^{\mathsf{t}_1}_{\sigma}(\mathcal{IB})$ et $E \in \sigma^{\mathsf{t}_2}(\mathcal{B})\}$.

Étant donné la nature de la relation de support, il semble intéressant de tenir compte de cette relation dans la définition des complétions. En effet, supposons qu'il y a un support déductif (ded) certain de a vers b ($(a,b) \in \mathcal{S}$), mais que b est un argument incertain ($b \in \mathcal{A}^?$). En utilisant la définition la plus simple des complétions (pla), on obtient des complétions où a est présent (et potentiellement accepté dans les extensions) mais b n'existe pas (et donc n'est forcément pas membre des extensions). Cela semble contradictoire avec la notion de support déductif. Cette situation contre-intuitive est corrigée grâce aux conditions supplémentaires des complétions sémantiques et closes.

2 Complexité et encodages logiques

Pour des raisons de place, nous ne présentons pas en détail les résultats obtenus, mais nous invitons le lecture à se référer à l'article de conférence [6]. Nous avons adapté aux IBAF les problèmes de décision liés à l'acceptabilité des arguments [7], c'est-à-dire :

PCA $\exists \mathcal{B} \in \mathsf{comp}^{\mathsf{t}_1}_{\sigma}(\mathcal{IB})$, $\exists E \in \sigma^{\mathsf{t}_2}(\mathcal{B})$ t.q. $a \in E$?
NCA $\forall \mathcal{B} \in \mathsf{comp}^{\mathsf{t}_1}_{\sigma}(\mathcal{IB})$, $\exists E \in \sigma^{\mathsf{t}_2}(\mathcal{B})$ t.q. $a \in E$?
PSA $\exists \mathcal{B} \in \mathsf{comp}^{\mathsf{t}_1}_{\sigma}(\mathcal{IB})$, $\forall E \in \sigma^{\mathsf{t}_2}(\mathcal{B})$ t.q. $a \in E$?
NSA $\forall \mathcal{B} \in \mathsf{comp}^{\mathsf{t}_1}_{\sigma}(\mathcal{IB})$, $\forall E \in \sigma^{\mathsf{t}_2}(\mathcal{B})$ t.q. $a \in E$?

Une observation générale est que la prise en compte des supports n'augmente pas la complexité par rapport aux IAF dans le cas des complétions *plain* et closes, mais peut avoir un impact non négligeable pour le raisonnement avec les complétions sémantiques.

Enfin, nous avons tiré parti de la littérature riche sur les encodages de problèmes d'argumentation en DL-PA (logique dynamique des assignements propositionnels) [4, 9] afin de proposer de nouveaux encodages adaptés à nos différentes méthodes de raisonnement.

3 Perspectives de recherche

Nous souhaitons poursuivre ces travaux en prenant en compte d'autres généralisation de l'argumentation abstraite (comme d'autres types de supports ou des interactions d'ordre supérieur entre les arguments). Une autre piste intéressante consiste à étudier des instanciations des IBAF sous forme de cadres structurés.

Remerciements J.-G. Mailly a reçu un support de l'ANR (AGGREEY ANR-22-CE23-0005 et AIDAL ANR-22-CPJ1-0061-01).

Références

[1] G. Boella, D. Gabbay, L. van der Torre, and S. Villata. Support in abstract argumentation. In *Proc. of COMMA'10*, pages 111–122, 2010.

[2] P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artif. Intell.*, 77(2) :321–358, 1995.

[3] B. Fazzinga, S. Flesca, and F. Furfaro. Incomplete bipolar argumentation frameworks. In *Proc. of ECAI'23*, pages 684–691, 2023.

[4] A. Herzig and A. Yuste-Ginel. Abstract argumentation with qualitative uncertainty : An analysis in dynamic logic. In *CLAR'21*, 2021.

[5] M.-C. Lagasquie-Schiex, J.-G. Mailly, and A. Yuste-Ginel. Incomplete Bipolar Argumentation Frameworks. Technical Report IRIT/RR–2023–01–FR, IRIT, May 2023.

[6] M.-C. Lagasquie-Schiex, J.-G. Mailly, and A. Yuste-Ginel. How to manage supports in incomplete argumentation. In *FoIKS 2024*, 2024.

[7] J.-G. Mailly. Yes, no, maybe, I don't know : Complexity and application of abstract argumentation with incomplete knowledge. *Argument Comput.*, 13(3) :291–324, 2022.

[8] F. Nouioua and V. Risch. Argumentation frameworks with necessities. In *SUM 2011*, 2011.

[9] Antonio Yuste-Ginel and Andreas Herzig. Qualitative uncertainty and dynamics of argumentation through dynamic logic. *J. Log. Comput.*, 2023.

Actes JIAF 2024

Sémantique agrégative graduelle pour les systèmes d'argumentation bipolaires pondérés

Yann Munro[1] **Isabelle Bloch**[1] **Mohamed Chetouani**[2]
Catherine Pelachaud[3] **Marie-Jeanne Lesot**[1]

[1]Sorbonne Université, CNRS, LIP6, Paris, France
[2]Sorbonne Université, CNRS, ISIR, Paris, France
[3]CNRS, Sorbonne Université, ISIR, Paris, France
{prenom.nom}@sorbonne-universite.fr

Résumé

Dans cet article, une instanciation particulière de la notion de sémantique graduelle pour les systèmes d'argumentation bipolaires pondérés (QBAF), appelée sémantique agrégative graduelle, est présentée. Contrairement aux sémantiques modulaires, nous séparons l'opération d'agrégation des attaquants de celle des supports. De cette façon, un poids global pour les attaquants et un autre pour les supports sont calculés, avant d'agréger ces deux valeurs avec le poids intrinsèque de l'argument. Cette méthode permet d'exploiter la bipolarité permise par les QBAF en gérant indépendamment et de manière potentiellement asymétrique les deux relations. Une discussion sur les propriétés requises pour ces fonctions d'agrégation ainsi qu'une comparaison formelle avec les axiomes classiques pour les sémantiques graduelles sont également proposées.

Abstract

In this paper, a particular instantiation of the notion of gradual semantics for quantitative bipolar argumentation framework (QBAF), called gradual aggregative semantics, is presented. Unlike modular semantics, we propose to agregate separately attackers on the one hand and defenders on the other hand. Doing so, a global weight for attackers and another for defenders are computed, before aggregating these two values with the intrinsic weight of the argument. This method makes it possible to exploit the bipolarity feature of QBAFs by managing the two relationships independently and potentially asymmetrically. A discussion about the properties required for these aggregation functions and a formal comparison with the classical axioms for gradual semantics are also proposed.

1 Introduction

L'argumentation formelle est un domaine très utilisé en intelligence artificielle pour modéliser et raisonner sur des dialogues argumentatifs et plus généralement des informations contradictoires. Ces informations sont appelées des *arguments* et une relation binaire entre les arguments appelée *relation d'attaque* permet de représenter ces incompatibilités. Depuis les travaux de Dung [9], des extensions ont été proposées pour permettre de modéliser des situations plus complexes. Tout d'abord une deuxième relation binaire, indépendante de la première et de nature opposée, appelée *support*, permet de représenter une défense directe d'un argument [6]. On se place alors dans un cadre bipolaire [8], associant des informations positives de support et des informations négatives d'attaque, les deux pouvant être asymétriques. De plus, une pondération sur chaque argument représentant la force intrinsèque d'un argument permet de modifier le degré d'acceptabilité initial de chaque argument et donc également l'impact d'un argument sur ceux qu'il attaque ou supporte. Un tel formalisme est appelé système d'argumentation bipolaire pondéré (Quantitative Bipolar Argumentation Framework, QBAF) [2].

Pour déterminer le statut de chaque argument dans un tel système, il existe deux types d'approches : des sémantiques reposant sur la notion d'extension [6] qui recherchent des ensembles d'arguments acceptables collectivement, et des sémantiques graduelles [5] qui associent une valeur d'acceptabilité à chaque argument. Dans cet article, nous considérons cette deuxième catégorie de sémantique. En particulier, nous proposons une sémantique graduelle que nous appelons sémantique *agrégative* qui agrège indépendamment attaques et supports en deux valeurs distinctes

puis combine ces deux valeurs avec le poids de l'argument pour calculer l'acceptabilité de celui-ci. Contrairement aux sémantiques graduelles existantes, cette approche permet de conserver la bipolarité plus longtemps en agrégeant de manière potentiellement asymétrique attaques et supports, et possède un comportement intrinsèquement explicable. De plus, en faisant le lien avec le domaine des opérateurs d'agrégation, elle ouvre la voie vers de nouvelles sémantiques pour l'argumentation.

Dans la suite, nous rappelons d'abord dans la section 2 quelques éléments de base sur les QBAF et les opérateurs d'agrégation, avant de présenter dans la section 3 notre proposition de sémantique agrégative. Ensuite, nous proposons une discussion en deux temps sur les différentes propriétés souhaitables pour les trois fonctions d'agrégation, selon les contextes considérés. Nous examinons d'abord dans la section 4 les propriétés classiques d'un opérateur d'agrégation et discutons de leur intérêt dans le cadre de l'argumentation. Puis, dans la section 5, nous étudions les axiomes usuels d'une sémantique modulaire en argumentation pour déterminer leurs liens avec les propriétés précédentes. Enfin, dans la section 6 nous discutons des sémantiques modulaires utilisées couramment dans la littérature.

2 Préliminaires

2.1 Argumentation et sémantique graduelle

Définition 1 (Quantitative Bipolar Argumentation Framework [2])**.** *Un QBAF est un quadruplet $A = (\mathcal{A}, \mathcal{R}, \mathcal{S}, w)$ tel que :*

- *$\mathcal{A}$ est un ensemble fini d'arguments ;*
- *$\mathcal{R}$ est l'ensemble des relations d'attaque ($\mathcal{R} \subseteq \mathcal{A} \times \mathcal{A}$) ;*
- *$\mathcal{S}$ est l'ensemble des relations de support ($\mathcal{S} \subseteq \mathcal{A} \times \mathcal{A}$) ;*
- *$w : \mathcal{A} \to I$ est une fonction qui associe un poids intrinsèque à chaque argument dans un ensemble de valeurs préordonné I.*

On note WAG l'ensemble des QBAF.

Comme $\mathcal{R}$ et $\mathcal{S}$ sont des relations binaires définies sur un ensemble fini, un QBAF peut être représenter sous la forme d'un graphe orienté étiqueté dans lequel les noeuds représentent les arguments, les étiquettes leur poids intrinsèque et les arcs les relations d'attaque et de support.

Pour déterminer le statut de chaque argument dans un graphe d'argumentation, différents types de sémantiques peuvent être utilisés. Dans cet article, nous nous intéressons aux sémantiques graduelles [5] qui associent une valeur d'acceptabilité à chaque argument :

Définition 2 (Sémantique graduelle)**.** *Une sémantique graduelle S est une fonction définie sur WAG et qui à $A \in WAG$ associe une fonction $\mathrm{Deg}_A^S : \mathcal{A} \to I$, où $\mathrm{Deg}_A^S(a)$ est le degré d'acceptabilité de l'argument a.*

Dans la suite, on utilise Deg_A^S pour parler à la fois du degré d'acceptabilité et de la sémantique S.

Il s'agit d'une définition très générale n'imposant aucune contrainte sur Deg_A^S. Pour cette raison, comme pour le cas des extensions, des axiomes ont été énoncés pour ces fonctions afin d'obtenir le comportement souhaité [1, 12, 14]. Ces derniers sont présentés dans la table 3 en annexe.

En plus de cette définition, voici quelques notations usuelles qui sont utilisées dans la suite de cet article :

- $\mathrm{Att}_a = \{b \in \mathcal{A} \mid (b, a) \in \mathcal{R}\}$ est l'ensemble des arguments qui attaquent l'argument a ;
- $\mathrm{sAtt}_a = \{b \in \mathrm{Att}_a \mid Deg_A^S(b) \neq 0\}$;
- $\mathrm{Supp}_a = \{c \in \mathcal{A} \mid (c, a) \in \mathcal{S}\}$ est l'ensemble des arguments qui supportent l'argument a ;
- $\mathrm{sSupp}_a = \{c \in \mathrm{Supp}_a \mid Deg_A^S(c) \neq 0\}$;
- Soit A, A' des QBAF. $A'' = A \oplus A'$ est le QBAF défini par : $\mathcal{A}'' = \mathcal{A} \cup \mathcal{A}', \mathcal{R}'' = \mathcal{R} \cup \mathcal{R}', \mathcal{S}'' = \mathcal{S} \cup \mathcal{S}'$;
- On appelle un isomorphisme de A vers A', une fonction f bijective qui conserve les relations et les étiquettes des graphes : $\forall a, b \in \mathcal{A}, w(a) = w'(f(a)), (b, a) \in \mathcal{R}$ ssi $(f(b), f(a)) \in \mathcal{R}'$ et $(b, a) \in \mathcal{S}$ ssi $(f(b), f(a)) \in \mathcal{S}'$.

Att_a et Supp_a sont des ensembles fixes définis en même temps que le QBAF auquel ils sont associés. Inversement, sAtt_a et sSupp_a peuvent changer selon le choix de la sémantique choisie mais également évoluent avec l'acceptabilité des arguments du graphe.

Parmi les différents axiomes de la littérature, Mossakowski et Neuhaus [12] proposent la modularité qui définit une sémantique graduelle particulière, appelée *sémantique modulaire*. Formellement, $Deg_A^S(a)$ est définie comme : $Deg_A^S(a) = i(\alpha(g_a, s), w(a))$ où α est une fonction d'agrégation qui combine les relations (que ce soit d'attaque ou de support) d'un argument (g_a) avec le degré d'acceptabilité (s) de tous les arguments du graphe et i une fonction d'influence qui agrège le poids intrinsèque $w(a)$ de l'argument avec la valeur de α. Cette définition permet de décomposer le calcul de l'acceptabilité d'un argument en deux étapes créant un processus considéré comme intrinsèquement explicable et offrant la possibilité de contraindre chacune des étapes différemment.

Il faut noter qu'il s'agit d'une définition récursive. En effet, le calcul de l'acceptabilité d'un argument nécessite d'avoir calculé l'acceptabilité des attaquants et supports de ce dernier. De fait, dans le cas d'un graphe d'argumentation cyclique, la convergence de ce calcul n'est pas garantie et dépend du choix de α et de i [14].

2.2 Fonction d'agrégation et propriétés usuelles

Une fonction d'agrégation est une fonction qui combine un ensemble de valeurs en une unique valeur et qui peut prendre de multiples formes, voir par exemple [4, 7, 10, 15, 16]. De nombreuses propriétés permettant de caractériser

leur comportement ont été proposées dans la littérature [3, 7], certaines d'entre elles sont présentées dans la table 2 en annexe.

3 Sémantique agrégative graduelle

Dans cette section, nous introduisons une nouvelle définition de sémantiques graduelles en argumentation que nous appelons *sémantique agrégative* pour souligner le rôle qu'y jouent les fonctions d'agrégation. Nous discutons également de son lien avec les sémantiques modulaires.

3.1 Architecture globale

Une sémantique agrégative suit un principe similaire à celui des sémantiques modulaires, en considérant une modularité accrue : elle exploite la bipolarité des relations d'attaque et de support des QBAF en traitant séparément l'agrégation des poids des attaquants et celle des supports, ce qui constitue une différence fondamentale avec ce qui est fait dans la littérature. En effet, cela permet une prise en compte indépendante et potentiellement asymétrique de ces deux composantes. Les deux valeurs ainsi obtenues sont ensuite agrégées avec le poids intrinsèque de l'argument concerné. Cette idée est formalisée dans la définition suivante.

Définition 3 (Sémantique agrégative graduelle)**.** *Soit* $A = (\mathcal{A}, \mathcal{R}, \mathcal{S}, w) \in WAG$, $a \in \mathcal{A}$ *et* J *un ensemble préordonné.*

On définit le degré d'acceptabilité de a *comme* $\mathrm{Deg}^S_A(a) = \varphi_f(\pi_\mathcal{R}(a), \pi_\mathcal{S}(a), w(a))$ *avec :*

- $\pi_\mathcal{R} : \mathcal{A} \to J$*, le poids global des attaquants d'un argument ;*
- $\pi_\mathcal{S} : \mathcal{A} \to J$ *le poids global des supports d'un argument ;*
- $\varphi_f : J^2 \times I \to I$ *une fonction d'agrégation.*

Les fonctions π_* $(* \in \{\mathcal{R}, \mathcal{S}\})$ *représentent le résultat de l'agrégation de l'acceptabilité de tous les attaquants ou de tous les supports d'un argument. Formellement, en notant* $b_1, \ldots, b_n$ *(respectivement* $c_1, \ldots, c_m$*) les éléments de* Att_a *(respectivement* $Supp_a$*) et* $\varphi_\mathcal{R}$*,* $\varphi_\mathcal{S}$ *deux fonctions d'agrégation, on a*

- $\pi_\mathcal{R}(a) = \varphi_\mathcal{R}(\mathrm{Deg}^S_A(b_1), \ldots, \mathrm{Deg}^S_A(b_n))$ *;*
- $\pi_\mathcal{S}(a) = \varphi_\mathcal{S}(\mathrm{Deg}^S_A(c_1), \ldots, \mathrm{Deg}^S_A(c_m))$.

Dans le cas où $\mathrm{Att}_a = \emptyset$, on adopte la convention que $\pi_\mathcal{R}(a) = 0$. De même, si $\mathrm{Supp}_a = \emptyset$, $\pi_\mathcal{S}(a) = 0$. Dans la suite et sans hypothèse supplémentaire on prend $I = J = [0, 1]$.

Cette formalisation ouvre la voie vers de nouvelles sémantiques d'acceptabilité pour les QBAF, selon les fonctions d'agrégation choisies pour instancier la définition 3. La section suivante examine, dans le cadre du calcul du degré d'acceptabilité, le sens des propriétés classiques qu'elles peuvent présenter. La section 5 examine, réciproquement, la correspondance des axiomes classiques des sémantiques argumentatives avec ces propriétés.

Remarque – Le nom de certaines propriétés des fonctions d'agrégation est également utilisé pour parler de propriétés parfois très différentes en argumentation [1]. Afin de pouvoir les distinguer, nous utilisons le mot postulat pour parler des propriétés souhaitables provenant du domaine des fonctions d'agrégation et nous gardons le mot axiome pour les propriétés issues de l'argumentation.

Dans la suite, $A = (\mathcal{A}, \mathcal{R}, \mathcal{S}, w)$ est un QBAF et Deg^S_A une sémantique agrégative graduelle avec $\varphi_f, \varphi_\mathcal{R}, \varphi_\mathcal{S}$ les fonctions d'agrégation associées.

3.2 Comparaison avec les sémantiques modulaires

La définition de sémantique agrégative que nous proposons repose sur le même principe que les sémantiques modulaires, à savoir décomposer le calcul de l'acceptabilité d'un argument en différentes étapes afin d'une part de pouvoir contraindre chacune d'entre elles indépendamment et d'autre part obtenir un processus explicable. Contrairement au cas des sémantiques modulaires, nous décomposons le calcul de la fonction α en deux étapes. Attaque et support étant deux relations indépendantes l'une de l'autre, nous souhaitons conserver cette indépendance. Bien qu'il soit possible par l'intermédiaire de α d'agréger les attaquants (respectivement supports) de la même manière que $\varphi_\mathcal{R}$ (respectivement $\varphi_\mathcal{S}$), ces deux valeurs doivent ensuite être combinées avant d'être agrégées avec w, empêchant un traitement asymétrique entre eux. Un exemple d'un tel scénario est donné dans la section 6.

À l'inverse, nous pensons qu'il existe une réécriture des sémantiques modulaires en sémantique agrégative. Une telle réécriture est proposée dans la table 1 dans le cas particulier des fonctions utilisées dans la littérature. Toutefois, la preuve formelle de ce résultat est un travail en cours. Comme les propriétés présentées dans [12, 14] ont été écrites spécifiquement pour les sémantiques modulaires, elles ne sont pas non plus discutées dans cet article.

Comme dans le cas modulaire, la définition de sémantique agrégative que nous proposons est récursive, aussi la question de la convergence du calcul de l'acceptabilité dans le cas des graphes cycliques se pose également. Nous ne la considérons pas dans cet article, en la laissant pour des travaux futurs, et en concentrant ici la discussion sur l'impact et la pertinence des propriétés des fonctions d'agrégation.

4 Discussion sur les propriétés souhaitables des différentes fonctions d'agrégation

Dans cette section, nous examinons différentes propriétés des fonctions d'agrégation dans le cadre de l'argumentation.

En particulier, nous discutons de la pertinence de celles-ci et cherchons à identifier dans quel contexte ces propriétés sont pertinentes, voire requises. Leur formalisation est regroupée dans la table 2 en annexe.

4.1 Les cas de $\varphi_{\mathcal{R}}$ et $\varphi_{\mathcal{S}}$

Cette section s'intéresse à la définition du poids global des attaquants et des supports d'un argument et plus particulièrement aux propriétés de $\varphi_{\mathcal{R}}$ et $\varphi_{\mathcal{S}}$. Dans toute la suite, ces deux fonctions sont notées φ_* avec $* \in \{\mathcal{R}, \mathcal{S}\}$. De plus, à part si la distinction entre attaquants et supports est nécessaire, nous utilisons le cas des attaquants de façon systématique pour expliquer et discuter des postulats à venir. Dans la suite, nous considérons un argument $a \in \mathcal{A}$.

(P1) : Conditions aux limites de φ_* – En plus des conditions aux limites présentées dans la table 2, il y a un cas limite supplémentaire à considérer en argumentation : celui de l'ensemble vide. Si a n'est pas attaqué, $\varphi_{\mathcal{R}}$ n'est pas défini. On adopte alors la convention que le poids global des attaquants de a est égal à 0.

(P2) : Croissance de φ_* – Ce deuxième postulat traduit que si l'acceptabilité d'un des attaquants de a augmente alors le poids agrégé des attaquants de ce dernier augmente ou reste constant. Il s'agit du comportement le plus souvent attendu quel que soit le contexte.

(P3) : Continuité de φ_* – Considérons une situation dans laquelle l'acceptabilité d'un des attaquants de a change un petit peu et nous voulons savoir quelle est l'évolution du poids global des attaquants. Une première possibilité est que cela n'entraîne qu'une légère modification sur le poids global des attaquants, ce que le postulat de continuité garantit. Néanmoins, dans certains scénarios, on peut imaginer que des effets de seuil apparaissent : au-delà ou en deçà d'un certain poids global pour les attaquants, il peut y avoir par exemple saturation entraînant un saut brusque qui correspond à une discontinuité. En faisant l'hypothèse d'un nombre au plus dénombrable de tels seuils, ce postulat devient alors que les fonctions φ_* doivent être continues par morceaux.

(P4) : Commutativité de φ_* – La pertinence du postulat de commutativité pour φ_* dépend du contexte considéré. En effet, si une fonction vérifie ce postulat alors l'ordre des variables de la fonction n'est pas important. C'est le cas par exemple lors de combinaisons de critères d'importance égale ou bien dans des procédures de vote. En argumentation, si la temporalité doit être prise en compte dans le calcul de l'acceptabilité d'un argument, les attaquants ne jouent pas un rôle symétrique. En effet, l'ordre d'énonciation joue un rôle crucial dans la construction d'un débat : par exemple un argument énoncé au tout début ou à la toute fin d'un débat a généralement plus d'impact que ceux au milieu [11]. Dans ce cas, les arguments ne peuvent pas être permutés et un poids pourrait être ajouté pour l'agrégation en fonction de la proximité avec le début et la fin du débat. La propriété de commutativité n'est alors pas souhaitée.

(P6) : Associativité de φ_* – Il s'agit principalement d'une caractéristique structurelle dont l'intérêt est de pouvoir agréger les données déjà présentes puis ensuite de mettre à jour avec les nouvelles données qui arrivent. En argumentation, dans le cas où l'ordre d'énonciation des arguments n'est pas pris en compte, toutes les informations sont déjà présentes dans le graphe au moment de l'agrégation. De plus, il s'agit d'une propriété très restrictive [10], c'est pourquoi sans autre contrainte cette propriété n'est en général pas requise.

(P5) : Idempotence de φ_* – La propriété d'idempotence est généralement souhaitable dans le cas où les sources des valeurs à agréger ne sont pas indépendantes. Dans ce cas, il y a redondance de l'information et donc elle est résumée en une valeur. À l'inverse, si les sources sont indépendantes, alors des informations redondantes se renforcent ou s'affaiblissent. Nous discutons de cette propriété au point suivant. En argumentation, dans le cas où l'on souhaite tenir compte du nombre d'arguments, c'est-à-dire de la redondance, cette propriété n'est donc pas souhaitable. En effet, si l'on impose l'idempotence, un argument attaqué par un seul argument d'acceptabilité x et un autre attaqué par plusieurs arguments de même acceptabilité x auraient alors le même poids global pour ses attaquants.

De plus, si φ_* est associative et idempotente, la répétition d'un argument devient inutile. En effet, $\varphi_*(x, x, y) = \varphi_*(\varphi_*(x, x), y) = \varphi_*(x, y)$. De fait, si l'on souhaite donner un sens à la répétition d'un argument, la négation d'au moins un de ces postulats devient nécessaire.

(P7-8) : Affaiblissement et renforcement de φ_* – Les postulats d'affaiblissement et de renforcement sont très reliés à la gestion des valeurs *faibles* et des valeurs *fortes*. Dans [1], l'axiome de monotonie exprime le fait qu'un argument moins attaqué ou plus soutenu, toutes choses égales par ailleurs, a une acceptabilité plus forte. Cela n'est pas évident et dépend notamment de l'attaquant supplémentaire : un argument "stupide" pourrait desservir son propre camp en faisant diminuant la confiance envers le camp des attaquants.

Lorsque les axiomes de monotonie et neutralité [1] sont vérifiés, le comportement à l'égard de ces arguments "stupides" est l'ignorance. Une autre possibilité est que l'ajout d'un argument de ce type affaiblisse l'ensemble des attaquants, c'est-à-dire diminue la valeur du poids global de l'ensemble d'arguments. À l'inverse, un ensemble d'arguments "brillants" peut se renforcer encore plus donnant un poids global encore plus important.

Ces deux comportements ne sont pas compatibles entre eux. Néanmoins, il existe des opérateurs comme par exemple les uninormes qui présentent un comportement hybride. Le domaine de définition est alors découpé de sorte que pour un ensemble d'arguments "faibles" la valeur

agrégée soit encore plus faible et que pour un ensemble d'arguments "forts" elle soit encore plus forte.

Enfin, dans le cas ou l'opérateur est également monotone et idempotent alors les seules fonctions possibles sont les fonctions min et max. En effet, en notant $\mathbf{x_n} = (x_1, \ldots, x_n)$ et x (respectivement X) le minimum (respectivement le maximum) de $\mathbf{x_n}$, on a pour l'affaiblissement $x = \varphi(\mathbf{x}) \leq \varphi(\mathbf{x}_n) \leq x$ c'est-à-dire $\varphi(\mathbf{x}_n) = x = \min_{i\in[1,n]}(x_i)$. De la même façon avec le postulat de renforcement, si φ est monotone et idempotente alors $\varphi(\mathbf{x}_n) = X = \max_{i\in[1,n]}(x_i)$.

(P9) : Composition de φ_* – Aucun des postulats précédents n'impose de comportement dans le cas où un argument est ajouté. De fait, si par exemple un argument c attaquant à la fois un argument a et un argument b est ajouté alors rien ne garantit qu'il ne va pas avoir pour effet d'inverser le rapport de forces entre l'ensemble des attaquants de chaque argument. Dans le cas où tous les arguments ne sont pas déjà présents dans le graphe et que ces derniers sont ajoutés séquentiellement, ce n'est pas un comportement souhaitable. En effet, dans le scénario le plus extrême c pourrait faire augmenter le poids global des attaquants de a jusqu'à celui de b mais jamais le dépasser si à la base il était inférieur. Cet argument étant présent dans les deux ensembles d'attaquants et l'ensemble d'attaquants de b étant à la base plus fort que celui de a, l'ensemble d'attaquants de b doit rester au moins aussi fort que celui de a. Cette propriété est traduite par le postulat de composition.

(P10) : Décomposition de φ_* – De façon identique au postulat de composition, la propriété de décomposition précise que si un attaquant de deux arguments est retiré alors l'ordre entre le poids global des attaquants de chacun de ces deux arguments est conservé. Dans le cas où la temporalité est prise en compte, un tel scénario apparaît si l'on souhaite modéliser un mécanisme d'oubli. Un argument ayant été énoncé "longtemps" auparavant est oublié au bout d'un certain temps si celui-ci n'a pas été réénoncé. Dans ce cas, l'argument disparaît et est donc retiré des valeurs à agréger. Un autre contexte dans lequel cette situation peut se présenter est celui où l'on considère 0 comme élément neutre des fonctions φ_*. Avec cette hypothèse, rendre un argument non acceptable, c'est-à-dire que son acceptabilité devient nulle, revient à retirer l'argument de l'agrégation.

4.2 Le cas de φ_f

Cette sous-section discute des postulats de la fonction d'agrégation φ_f qui calcule le degré d'acceptabilité d'un argument en fonction du poids global de ses attaquants, de ses supports et de son poids intrinsèque.

(P2) : Monotonie de φ_f – Contrairement aux deux autres fonctions d'agrégations, φ_f admet un nombre fixe de variables possédant chacune une sémantique différente. La première correspond au poids des attaquants d'un argument : plus celle-ci est élevée, plus l'acceptabilité de l'argument est faible. À l'inverse, la deuxième et la troisième variables de la fonction correspondent au poids des supports et à la force intrinsèque de l'argument. De fait plus l'une d'elles est élevée, plus l'acceptabilité augmente. Ainsi, φ_f est décroissante selon sa première variable et croissante selon les deux dernières.

(P1) : Conditions aux limites de φ_f – En raison de la monotonie de φ_f, ses conditions aux limites diffèrent de celles présentées dans la table2. Elles s'écrivent : $\varphi_f(0,1,1) = 1$ et $\varphi_f(1,0,0) = 0$. Nous ajoutons également une condition aux limites supplémentaire : $\varphi_f(0,0,w) = w$. Il s'agit d'une propriété élémentaire : un argument qui n'est ni attaqué, ni soutenu a une acceptabilité égale à son poids intrinsèque.

(P3) : Continuité de φ_f – Tout comme pour $\varphi_\mathcal{R}$ et $\varphi_\mathcal{S}$, la continuité de φ_f dépend du choix ou non d'inclure des effets de seuil et de saturation dans la modélisation.

(P4) : Commutativité de φ_f – Les arguments de φ_f n'étant pas symétriques et ayant chacun sa propre sémantique, la commutativité n'est pas une propriété désirable. En effet, si permuter le poids des attaquants avec le poids des supports n'a aucun effet sur l'acceptabilité d'un argument alors, sauf cas particulier, cela signifie que faire une distinction entre attaquant et support n'apporte rien à la modélisation et donc la bipolarité n'a aucun intérêt.

(P6-9-10) : Associativité, composition et décomposition de φ_f – Contrairement à $\varphi_\mathcal{R}$ $\varphi_\mathcal{S}$, certaines fonctions d'agrégation, comme φ_f, possèdent un nombre fixe de variables et leur place dans l'expression de φ_f est fixe également. Dans ce cas là, les postulats d'associativité, de composition et de décomposition n'ont alors pas de sens.

(P5) : Idempotence de φ_f – La propriété d'idempotence appliquée à φ_f est un cas particulier d'une propriété plus large correspondant à une forme de symétrie entre les attaquants et les supports. En effet, considérer que $\varphi_f(x,x,x) = x$ peut être justifié par le fait que les attaquants et les supports jouent un rôle symétrique et donc se compensent. Dans ce cas, cela se généralise sous la forme : $\forall x, w \in I, \varphi_f(x,x,w) = w$.

(P7-8) : Affaiblissement et Renforcement de φ_f – En raison de la monotonie de φ_f, le postulat d'affaiblissement devient : un poids global des attaquants "élevé" et des valeurs "faibles" pour le poids des supports et le poids de l'argument s'affaiblissent pour donner une acceptabilité encore plus faible. De la même manière, un argument "fort", "faiblement" attaqué et "fortement" soutenu aura une valeur d'acceptabilité renforcée, c'est-à-dire encore plus forte. Avec un comportement hybride, φ_f étale les valeurs d'acceptabilité permettant d'obtenir plus de contraste entre arguments forts et faibles. Dans les cas où une décision précise est requise, ce comportement peut permettre de limiter le nombre de choix possibles.

5 Comparaison formelle avec les axiomes classiques des QBAF

Dans cette section, nous revenons sur les axiomes des QBAF les plus couramment utilisés à notre connaissance dans la littérature [1, 12, 14], et présentés dans la table 3. Tout d'abord nous examinons leurs liens avec les différents postulats précédents. Certains axiomes n'ayant pas de lien avec ces postulats, nous proposons ensuite d'adapter ces derniers au cadre d'une sémantique agrégative.

Les preuves des propositions sont dans la plupart des cas directes et ne sont pas détaillées ici.

5.1 Lien entres les axiomes argumentatifs et les propriétés des opérateurs d'agrégation

La définition d'une sémantique agrégative impose une structure particulière pour calculer le degré d'acceptabilité d'un argument. En effet, seuls les degrés d'acceptabilité des attaquants directs et des supports directs d'un argument ainsi que le poids de l'argument lui-même sont pris en compte pour ce calcul. Pour cette raison, certains axiomes de l'argumentation sont vérifiés sans aucune hypothèse supplémentaire sur les fonctions d'agrégation.

Proposition 1. *Soit A un QBAF acyclique et* Deg_A^S *une sémantique agrégative. Alors* Deg_A^S *vérifie les axiomes (A1), (A2), (A3), (A5).*

L'axiome d'équivalence (A4) garantit que l'acceptabilité d'un argument ne dépend que de son propre poids et du poids global de ses attaquants et supports. Dans le cadre d'une sémantique agrégative dont les fonctions $\varphi_{\mathcal{R}}$ et $\varphi_{\mathcal{S}}$ sont non commutatives, l'acceptabilité d'un argument dépend également de l'ordre des attaquants et des supports. Pour satisfaire cet axiome il faut donc imposer la commutativité de ces deux fonctions.

Proposition 2 (Équivalence)**.** *Soit A, un QBAF acyclique et* Deg_A^S *une sémantique agrégative. Si* $\varphi_{\mathcal{R}}$ *et* $\varphi_{\mathcal{S}}$ *sont commutatives,* Deg_A^S *vérifie l'axiome d'équivalence.*

L'axiome suivant est celui de neutralité (A6). Il s'intéresse à l'effet des attaquants et des supports d'acceptabilité nulle. Ils sont qualifiés d'inutiles (*worthless*) [1] et ne doivent donc pas avoir d'impact sur l'acceptabilité d'un argument : dans le cadre d'une sémantique agrégative, ils ne doivent pas modifier la valeur du poids des attaquants ou des supports. Ce n'est pas le cas par défaut et nécessite d'imposer que 0 est un élément neutre pour $\varphi_{\mathcal{R}}$ et $\varphi_{\mathcal{S}}$.

Proposition 3 (Neutralité)**.** *Soit A un QBAF acyclique et* Deg_A^S *une sémantique agrégative. Si* 0 *est un élément neutre pour* $\varphi_{\mathcal{R}}$ *et* $\varphi_{\mathcal{S}}$*,* Deg_A^S *vérifie l'axiome de neutralité.*

Les deux axiomes suivants sont la monotonie (A7) et le renforcement (A8) tous les deux au sens de l'argumentation. Ils concernent l'évolution du degré d'acceptabilité quand on ajoute des attaquants ou des supports (axiome de monotonie) ou quand on modifie l'acceptabilité de l'un d'entre eux (axiome de renforcement).

Proposition 4 (Renforcement)**.** *Soit* $A \in WAG$*, un QBAF acyclique et* Deg_A^S *une sémantique agrégative. Si* $\varphi_{\mathcal{R}}$*, et* $\varphi_{\mathcal{S}}$ *sont croissantes (respectivement strictement croissantes) et* φ_f *est croissante selon x et décroissante selon y (respectivement strictement monotone) alors* Deg_A^S *vérifie l'axiome de renforcement (respectivement renforcement strict) au sens de l'argumentation.*

Pour l'axiome de monotonie, une hypothèse supplémentaire est nécessaire : 0 doit être élément neutre. En effet, il n'y a aucun postulat des fonctions d'agrégation qui explicite le comportement dans tous les cas quand un argument est ajouté. En effet, le postulat de composition (P9) concerne l'ajout d'un même ensemble d'arguments à deux ensembles. D'autre part, les postulats d'affaiblissement et de renforcement donnent une borne inférieure ou supérieure pour le résultat mais ne fournissent pas d'informations sur l'évolution par rapport à l'absence de cet argument. Or si 0 est élément neutre, on peut écrire $\forall z \in [0,1], \varphi_{\mathcal{R}}(\mathbf{x}_n) = \varphi_{\mathcal{R}}(\mathbf{x}_n, 0) \leq \varphi_{\mathcal{R}}(\mathbf{x}_n, z)$.

Proposition 5 (Monotonie)**.** *Soit A un QBAF acyclique et* Deg_A^S *une sémantique agrégative. Si* 0 *est un élément neutre pour* $\varphi_{\mathcal{R}}$ *et* $\varphi_{\mathcal{S}}$ *et que les trois fonctions d'agrégation sont monotones, alors* Deg_A^S *vérifie l'axiome de monotonie.*

5.2 Reformulation de certains axiomes de l'argumentation pour une sémantique agrégative

Résilience (A9) – Cet axiome donne un rôle particulier aux valeurs extrêmes des poids intrinsèques et finaux d'un argument. En effet, ces valeurs d'acceptabilité ne peuvent être atteintes que par des arguments ayant déjà cette valeur comme poids. Ainsi, un argument d'acceptabilité égale à 0.99 mais dont le poids intrinsèque est différent de 1 ne pourra jamais être totalement accepté peu importe la qualité de ses supports. Il s'agit d'un principe optionnel dont l'intérêt dépend de la nature des arguments [1].

Les postulats sur les fonctions d'agrégation n'imposent aucun comportement à propos de l'impact des attaquants et des supports sur le poids d'un argument. Selon quelle relation est la plus forte, aucune règle n'est spécifiée sur la valeur de $Deg_A^S(a)$ par rapport à son poids intrinsèque. Les axiomes de Franklin, d'affaiblissement et de consolidation proposent de telles règles, constituant les premières règles entremêlant les trois fonctions d'agrégation.

Principe de Franklin (A10) – Le principe de Franklin est décrit dans [1] comme un principe indiquant la manière dont les attaquants et les supports d'un argument doivent être agrégés. Il s'agit de la première propriété qui relie les différentes fonctions d'agrégation. En effet, cet axiome régit les situations dans lesquelles un attaquant et un

$\alpha = \varphi_S - \varphi_R$		(P1)	(P2)	(P3)	(P4)	(P5)	(P6)	(P7)	(P8)	(P9)	(P10)
Somme	$\varphi_{R/S}(b_1, \ldots, b_n) = \sum_i b_i$	×	✓	✓	✓	×	✓	×	✓	✓	✓
Produit	$\varphi_{R/S}(b_1, \ldots, b_n) = 1 - \prod_i (1 - b_i)$	✓	✓	✓	✓	×	✓	×	✓	✓	✓
Maximum	$\varphi_{R/S}(b_1, \ldots, b_n) = \max_i(b_i)$	✓	✓	✓	✓	✓	✓	×	✓	✓	×
φ_f											
Linear(κ)	$\varphi_f(x, y, w) = w - \frac{w}{\kappa}\max(0, -s) + \frac{1-w}{\kappa}\max(0, s)$ avec $s = y - x$	✓	✓	✓	×	✓		×	×		
Euler-based	$\varphi_f(x, y, w) = 1 - \frac{1-w^2}{1+we^s}$ avec $s = y - x$	✓	✓	✓	×	✓		×	×		
p-Max(κ)	$\varphi_f(x, y, w) = w \times (1 + h(\frac{s}{\kappa}) - h(-\frac{s}{\kappa}))$ avec $h(x) = \frac{\max(0,x)^p}{1+\max(0,x)^p}$ et $s = y - x$	✓	✓	✓	×	✓		×	×		
	$\varphi_f(x, y, z) = \begin{cases} \min(\max(0, \frac{z-x}{1-y}), 1) & \text{si } y < 1 \\ \frac{1+\text{signe}(z-x)}{2} & \text{si } y = 1 \end{cases}$	✓	✓	✓	×	×		×	×		

TABLE 1 – Exemples de fonctions d'agrégation pour les sémantiques de la littérature. À l'exception de la somme ou $I = \mathbb{R}$, les autres fonctions sont définies de $[0, 1]$ dans $[0, 1]$.

support de même acceptabilité sont ajoutés. Dans un cas, l'attaquant domine le support et donc l'acceptabilité globale diminue (principe de Franklin). Dans l'autre, attaquant et support se compensent et donc la valeur d'acceptabilité ne change pas (principe de Franklin strict). Dans notre formalisme où l'asymétrie est permise, l'ajout d'un attaquant et d'un support de même force peut ne pas avoir le même impact sur le poids global des attaquants et des supports. Néanmoins, peu importe ces nouvelles valeurs, nous avons une information sur les nouvelles acceptabilités. En réécrivant ce principe avec notre formalisme et en reprenant les notations de la table 2 nous avons : $\varphi_f(\varphi_R(\mathbf{x_n}, x), \varphi_S(\mathbf{y_m}, x), w) \leq \varphi_f(\varphi_R(\mathbf{x_n}), \varphi_S(\mathbf{y_m}), w)$ et égalité dans le cas strict.

Affaiblissement (A11) – Informellement, l'axiome d'affaiblissement s'intéresse au cas où les attaquants "dominent" les supports. Dans cette situation, le degré d'acceptabilité doit être inférieur strictement au poids de l'argument. Il s'agit de la deuxième propriété qui impose un comportement entremêlant les trois fonctions d'agrégation. En effet, en réécrivant formellement cet axiome dans notre formalisme on a : soit $a \in \mathcal{A}$ tel que $w(a) > 0$, et sous réserve d'existence soit f une fonction injective de Supp_a vers Att_a telle que $\forall x \in \text{Supp}_a$, $Deg_A^S(x) \leq Deg_A^S(f(x))$. Avec $\text{Supp}_a = \mathbf{x}_m$ et $\text{Att}_a = \mathbf{y}_n$, si $\exists x_i$ tel que $Deg_A^S(x_i) < Deg_A^S(f(x_i))$ ou $\exists y_j$ tel que $f^{-1}(y_j) = \emptyset$ et $Deg_A^S(y_j) > 0$ alors on a : $\varphi_f(\varphi_R(\mathbf{y}_n), \varphi_S(\mathbf{x}_m), w(a)) < w(a)$.

Pour tenter de séparer cet axiome en deux propriétés indépendantes, on peut ajouter un deuxième résultat : $\varphi_S(\mathbf{x}_m) < \varphi_R(\mathbf{y}_n)$. Avec ce résultat, l'axiome d'affaiblissement devient alors que si le poids global des attaquants est plus fort que celui des supports, alors l'acceptabilité est inférieure au poids initial de l'argument. Cependant, cette proposition n'est plus équivalente à l'axiome de base. En effet, en prenant comme fonction d'agrégation pour φ_R et φ_S la fonction min, alors malgré l'existence d'une telle injection entre les deux ensembles d'argument, si on a $\min(\mathbf{y}_j) < \min(\mathbf{x}_i)$, le poids global des attaquants sera plus faible que celui des supports. Si par ailleurs on choisit pour φ_f une moyenne pondérée qui accorde plus de poids à l'attaque qu'au support, alors on peut avoir malgré tout $Deg_A^S(a) < w(a)$ ce qui vérifie l'axiome d'affaiblissement mais contredit la propriété de "découplage" $\varphi_S(\mathbf{x}_m) < \varphi_R(\mathbf{y}_n)$.

Consolidation (A12) – De même, l'axiome de consolidation régit le cas où les supports "dominent" les attaquants et impose que l'acceptabilité de l'argument soit supérieure strictement à son poids intrinsèque. Il s'agit de la troisième propriété liant les trois fonctions d'agrégation entre elles.

6 Discussion sur les sémantiques graduelles de la littérature

Dans cette section, nous reprenons les sémantiques graduelles les plus couramment utilisées dans la littérature et discutons de leur implémentation dans le formalisme des sémantiques agrégatives. Les résultats sont présentés dans la table 1.

Tout d'abord dans le cas des sémantiques modulaires, attaquants et supports sont agrégés collectivement. Avec les sémantiques usuelles il s'agit toujours d'une différence entre ce que nous appelons le poids des supports et celui des attaquants. De plus, l'agrégation des attaquants et celle des supports sont identiques. Cela correspond donc à $\varphi_R = \varphi_S$. Enfin, bien que dans tout l'article nous ayons pris $I = J = [0, 1]$, dans le cas de la fonction somme pour α (φ_R et φ_S dans notre formalisme), on a $I = \mathbb{R}$.

Contrairement aux cas précédents et pour illustrer le genre d'asymétrie qu'il est possible d'obtenir grâce à la définition de sémantique agrégative, prenons la fonction $\varphi_f : [0, 1]^3 \to [0, 1]$ définie par : $\varphi_f(x, y, z) = \begin{cases} \min(\max(0, \frac{z-x}{1-y}), 1) & \text{si } y < 1 \\ \frac{1+\text{signe}(z-x)}{2} & \text{si } y = 1 \end{cases}$

En agrégeant de cette façon, les supports (y) peuvent affecter l'acceptabilité de l'argument uniquement dans le cas où le poids des attaquants (x) est plus faible que le poids intrinsèque de l'argument (z). Dans ce cas, à l'inverse des attaquants pour lesquels c'est l'écart relatif avec le poids de l'argument qui compte, pour les supports il s'agit de leur valeur agrégée y en elle-même. Cette fonction correspond à un cas extrême où si $x \geq z$ alors le résultat final est égal à 0. Enfin, il s'agit d'une sémantique qu'il n'est pas possible d'écrire sous forme de sémantique modulaire.

7 Conclusion

Dans cet article, nous avons proposé une formalisation d'un cas particulier de sémantiques graduelles que nous appelons sémantique agrégative. Elle repose sur la décomposition du processus de calcul de l'acceptabilité d'un argument en trois étapes d'agrégation distinctes : calcul du poids global des attaquants, calcul du poids global des supports et agrégation de ces deux poids avec le poids intrinsèque de l'argument. Grâce à la diversité des fonctions d'agrégation qui existent dans la littérature et à la discussion que nous avons menée sur les postulats et axiomes désirables pour ces fonctions, il est possible de construire de nouvelles sémantiques dépendant du contexte. Une première perspective pour ce travail consiste précisément à proposer une instanciation des sémantiques modulaires de [12] quand cela est possible, ainsi que quelques sémantiques originales afin de les comparer dans des contextes bien définis et variés. Des travaux en cours portent également sur l'extension de ce formalisme en incluant des argument de poids intrinsèque non évalué, ou encore l'ajout de nouvelles propriétés. L'objectif est d'étendre des travaux précédents [13] à ce cadre plus large pour la génération d'explications dans un cadre d'interaction humain-agent.

Références

[1] L. Amgoud and J. Ben-Naim. Weighted bipolar argumentation graphs : Axioms and semantics. In *27th International Joint Conference on Artificial Intelligence-IJCAI*, pages 5194–5198, 2018.

[2] P. Baroni, A. Rago, and F. Toni. From fine-grained properties to broad principles for gradual argumentation : A principled spectrum. *International Journal of Approximate Reasoning*, 105 :252–286, 2019.

[3] I. Bloch. Information Combination Operators for Data Fusion : A Comparative Review with Classification. *IEEE Transactions on Systems, Man, and Cybernetics*, 26(1) :52–67, 1996.

[4] F. Bobillo and U. Straccia. Aggregation operators for fuzzy ontologies. *Applied Soft Computing*, 13(9) :3816–3830, 2013.

[5] C. Cayrol and M.-C. Lagasquie-Schiex. Gradual valuation for bipolar argumentation frameworks. In *European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty(ECSQARU)*, pages 366–377. Springer, 2005.

[6] C. Cayrol and M.-C. Lagasquie-Schiex. On the acceptability of arguments in bipolar argumentation frameworks. In *European Conference on Symbolic and Quantitative Approaches to Reasoning and Uncertainty (ECASQARU)*, pages 378–389. Springer, 2005.

[7] D. Dubois and H. Prade. A review of fuzzy set aggregation connectives. *Information Sciences*, 36(1-2) :85–121, 1985.

[8] D. Dubois and H. Prade. An introduction to bipolar representations of information and preference. *International Journal of Intelligent Systems*, 23(8) :866–877, 2008.

[9] P. M. Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77 :321–357, 1995.

[10] M. Grabisch, J.-L. Marichal, R. Mesiar, and E. Pap. *Aggregation functions*, volume 127. Cambridge University Press, 2009.

[11] N. Miller and D. T Campbell. Recency and primacy in persuasion as a function of the timing of speeches and measurements. *The Journal of Abnormal and Social Psychology*, 59(1) :1, 1959.

[12] T. Mossakowski and F. Neuhaus. Modular semantics and characteristics for bipolar weighted argumentation graphs. *arXiv preprint arXiv :1807.06685*, 2018.

[13] Y. Munro, I. Bloch, M. Chetouani, M.-J. Lesot, and C. Pelachaud. Argumentation and causal models in human-machine interaction : a round trip. In *8th International Workshop on Artificial Intelligence and Cognition*, 2022.

[14] N. Potyka. Extending modular semantics for bipolar weighted argumentation. In *KI 2019 : Advances in Artificial Intelligence : 42nd German Conference on AI*, pages 273–276. Springer, 2019.

[15] V. Torra and Y. Narukawa. *Modeling decisions : information fusion and aggregation operators*. Springer Science & Business Media, 2007.

[16] R. R Yager and A. Rybalov. Full reinforcement operators in aggregation techniques. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 28(6) :757–769, 1998.

8 Annexe

Nom	Propriété de φ
(P1) : Conditions aux limites	$\varphi(0,\dots,0)=0$ et $\varphi(1,\dots,1)=1$
(P2) : Monotonie (croissance)	si $x'_{n+1}\leq x'_{n+1}$, $\quad\varphi(\mathbf{x}_n,x'_{n+1})\leq\varphi(\mathbf{x}_n,x_{n+1})$
(P3) : Continuité	$\lim\limits_{\mathbf{x}'_n\to\mathbf{x}_n}\varphi(\mathbf{x}'_n)=\varphi(\mathbf{x}_n)$
(P4) : Commutativité	soit σ une permutation de $[1,n]$, $\quad\varphi(\mathbf{x}_n)=\varphi(x_{\sigma(1)},\dots,x_{\sigma(n)})$
(P5) : Idempotence	$\varphi(x,\dots,x)=x$
(P6) : Associativité	$\varphi(\varphi(\mathbf{x}_n),\varphi(\mathbf{y}_m))=\varphi(\mathbf{x}_n,\mathbf{y}_m)$
(P7) : Affaiblissement	$\varphi(\mathbf{x}_n)\leq\min(\mathbf{x}_n)$
(P8) : Renforcement	$\varphi(\mathbf{x}_n)\geq\max(\mathbf{x}_n)$
(P9) : Composition	si $\varphi(\mathbf{x}_n)\leq\varphi(\mathbf{y}_m)$, $\varphi(\mathbf{x}_n,\mathbf{z})\leq\varphi(\mathbf{y}_m,\mathbf{z})$ quelle que soit la position de $\mathbf{z}$
(P10) : Décomposition	si $\varphi(\mathbf{x}_n,\mathbf{z})\leq\varphi(\mathbf{y}_m,\mathbf{z})$, $\varphi(\mathbf{x}_n)\leq\varphi(\mathbf{y}_m)$ quelle que soit la position de $\mathbf{z}$

Table 2 – Propriétés classiques des fonctions d'agrégation. Notations : les variables écrites en gras et indicées par n, comme $\mathbf{x}_n=(x_1,\dots x_n)$, sont des éléments de $[0,1]^n$, les autres variables sont des valeurs de $[0,1]$.

Nom	Définition
(A1) : Anonymat	$\forall f$ isomorphisme de A vers A', $\forall a\in\mathcal{A}, Deg^S_A(a)=Deg^S_{A'}(f(a))$
(A2) : Indépendance	Si $\mathcal{A}\cap\mathcal{A}'=\emptyset$ alors $\forall a\in\mathcal{A}, Deg^S_A(a)=Deg^S_{A\oplus A'}(a)$
(A3) : Directionnalité	Si $\mathcal{A}=\mathcal{A}', w=w', \mathcal{R}\subseteq\mathcal{R}'$ et $\mathcal{S}\subseteq\mathcal{S}'$ alors $\forall a,b,x\in\mathcal{A}$ si $\mathcal{R}'\cup\mathcal{S}'=\mathcal{R}\cup\mathcal{S}\cup\{(a,b)\}$ et qu'il n'y a pas de chemin entre b et x, $Deg^S_A(x)=Deg^S_{A'}(x)$
(A4) : Équivalence	$\forall a,b\in\mathcal{A}$ si : $w(a)=w(b)$ et $\exists f:\mathrm{Att}_a\to\mathrm{Att}_b$ et $\exists f':\mathrm{Supp}_a\to\mathrm{Supp}_b$ bijectives tel que : $\forall x\in\mathrm{Att}_a, Deg^S_A(x)=Deg^S_A(f(x))$ et $\forall y\in\mathrm{Supp}_a, Deg^S_A(y)=Deg^S_A(f'(y))$ Alors $Deg^S_A(a)=Deg^S_A(b)$
(A5) : Stabilité	$\forall a\in\mathcal{A}$, si $\mathrm{Att}_a=\mathrm{Supp}_a=\emptyset$ alors $Deg^S_A(a)=w(a)$
(A6) : Neutralité	$\forall a,b,x\in\mathcal{A}$, si $w(a)=w(b), Deg^S_A(x)=0, \mathrm{Att}_a\subseteq\mathrm{Att}_b, \mathrm{Supp}_a\subseteq\mathrm{Supp}_b$ et $\mathrm{Att}_b\cup\mathrm{Supp}_b=\mathrm{Att}_a\cup\mathrm{Supp}_a\cup\{x\}$ alors $Deg^S_A(a)=Deg^S_A(b)$
(A7) : Monotonie	Si $\forall a,b\in\mathcal{A}$ tel que $w(a)=w(b), \mathrm{Att}_a\subseteq\mathrm{Att}_b$ et $\mathrm{Supp}_b\subseteq\mathrm{Supp}_a$ alors **Monotonie :** $Deg^S_A(a)\geq Deg^S_A(b)$ **Monotonie stricte :** Si $Deg^S_A(a)>0$ et $\mathrm{sAtt}_a\subset\mathrm{sAtt}_b$ ou $Deg^S_A(b)<1$ et $\mathrm{sSupp}_b\subset\mathrm{sSupp}_a$ alors $Deg^S_A(a)>Deg^S_A(b)$
(A8) : Renforcement	Si $\forall a,b\in\mathcal{A},\forall C,C'\subseteq\mathcal{A}$ et $\forall x,x',y,y'\in\mathcal{A}\setminus(C\cup C')$ tel que $w(a)=w(b)$, $\mathrm{Att}_a=C\cup\{x\}, \mathrm{Att}_b=C\cup\{y\}, \mathrm{Supp}_a=C'\cup\{x'\}, \mathrm{Supp}_b=C'\cup\{y'\}$ et $Deg^S_A(x)\leq Deg^S_A(y), Deg^S_A(x')\geq Deg^S_A(y')$ alors **Renforcement :** $Deg^S_A(a)\geq Deg^S_A(b)$ **Renforcement strict :** Si $Deg^S_A(a)>0$ et $Deg^S_A(x)<Deg^S_A(y)$ ou $Deg^S_A(b)<1$ et $Deg^S_A(x')>Deg^S_A(y')$ alors $Deg^S_A(a)>Deg^S_A(b)$
(A9) : Résilience	$\forall a\in\mathcal{A}$, si $0<w(a)<1$ alors $0<Deg^S_A(a)<1$
(A10) : Principe de Franklin	$\forall a,b,x,y\in\mathcal{A}$, si $w(a)=w(b), \mathrm{Att}_a=\mathrm{Att}_b\cup\{x\}, \mathrm{Supp}_a=\mathrm{Supp}_b\cup\{y\}$ et $Deg^S_A(x)=Deg^S_A(y)$ **Franklin** : $Deg^S_A(a)\leq Deg^S_A(b)$ **Franklin Strict** : $Deg^S_A(a)=Deg^S_A(b)$
(A11) : Affaiblissement	$\forall a\in\mathcal{A}, w(a)>0$ et $f:\mathrm{Supp}_a\to\mathrm{Att}_a$ injective tel que $\forall x\in\mathrm{Supp}_a, Deg^S_A(x)\leq Deg^S_A(f(x))$ et $\mathrm{sAtt}_a\setminus\{f(x)\|x\in\mathrm{Supp}_a\}\neq\emptyset$ ou $\exists x\in\mathrm{Supp}_a$ tel que $Deg^S_A(x)<Deg^S_A(f(x))$ Alors $Deg^S_A(a)<w(a)$
(A12) : Consolidation (*Strengthening*)	$\forall a\in\mathcal{A}, w(a)>0$ et $f:\mathrm{Att}_a\to\mathrm{Supp}_a$ injective tel que $\forall x\in\mathrm{Att}_a, Deg^S_A(x)\leq Deg^S_A(f(x))$ et $\mathrm{sSupp}_a\setminus\{f(x)\|x\in\mathrm{Att}_a\}\neq\emptyset$ ou $\exists x\in\mathrm{Att}_a$ tel que $Deg^S_A(x)<Deg^S_A(f(x))$ Alors $Deg^S_A(a)>w(a)$

Table 3 – Quelques axiomes des sémantiques graduelles en argumentation avec A et A' des WAG quelconques.

Session 6 : Raisonnement par analogie

Actes JIAF 2024

Analogie et moyenne généralisée

Yves Lepage[1] **Miguel Couceiro**[2,3]

[1] IPS, université Waseda, Japon
[2] LORIA, université de Lorraine, France
[3] INESC-ID, IST, Universidade de Lisboa, Portugal
yves.lepage@waseda.jp
miguel.couceiro@loria.fr

Résumé

Nous étudions le lien entre analogie et moyennes, ou, plus exactement, moyennes généralisées. Les moyennes généralisées constituent une classe de fonctions d'agrégation définies en terme d'un seul paramètre, la puissance. En particulier, nous montrons que, quels que soient quatre nombres réels positifs, il est toujours possible de voir une analogie entre eux, grâce à une puissance bien choisie, qui est unique. Nous montrons aussi que toute analogie peut se réduire à une analogie arithmétique équivalente.

Abstract

We study the relationship between analogy and means, or, more precisely, generalized means, which are aggregation functions defined in terms of a power parameter. In particular, we show that whatever four positive real numbers are, it is always possible to establish an analogy between them by choosing a suitable power. We show that for each analogy this power is unique. In addition, we show that any analogy can be reduced to an equivalent arithmetic analogy.

1 Introduction

En intelligence artificielle, les termes d'inférence analogique ou de raisonnement analogique sont souvent utilisés. Les travaux s'y rattachant s'intéressent à l'étude des rapports entre deux couples d'éléments A et B d'une part et C et D de l'autre. Il y a plusieurs exploitations possibles d'une telle compréhension. Typiquement on peut juger si le rapport de C à D est le même que celui qui existe entre A et B. On évaluera alors la qualité de la similarité de tels rapports et l'on pourra discuter de similarité d'attributs ou de similarité de relations [32] [1] à la suite des travaux plus récents de Gentner [15]. On peut aussi voir A et C comme des problèmes, B comme une solution au problème A, et l'on se demandera si la transposition du rapport de A à B sur C permet de produire un D et dans quelle mesure ce D est une solution du problème C. C'est l'approche en raisonnement à partir de cas [3, 4, 22].

Le principe sous-jacent est celui de l'*inférence analogique* qui a été intégré dans diverses tâches d'apprentissage automatique, notamment, en apprentissage de préférences et en recommandation [13, 14, 24] et, plus généralement, en classification [11]. En outre, l'extrapolation analogique (inférence) peut résoudre des tâches de raisonnement difficiles telles que les tests d'aptitude scolaire ou de réponse à des questions visuelles [31, 27, 5] et de vérification du sens d'une phrase cible [34]. Elle peut également prendre en charge l'augmentation des ensembles de données (extension analogique et extrapolation) [9]. Cela peut également être réalisé à un méta-niveau pour un apprentissage par transfert [8, 1] où l'idée est de profiter de ce qui a été appris sur un domaine source afin d'améliorer le processus d'apprentissage dans un domaine cible lié à un domaine source. Enfin, la création d'analogies peut fournir des explications utiles qui s'appuient sur l'exemple-contre-exemple parallèle [17] et guident la génération contrefactuelle [18].

Dans certains domaines particuliers comme la traduction automatique, ce type de raisonnement analogique a été utilisé [26] et le rapprochement avec le raisonnement à partir de cas y est bien reconnu [7]. En traitement automatique des langues toujours, mais cette fois-ci par exemple pour la tâche de production de mots, une telle approche a été proposée [25] : la minimisation de la complexité algorithme du programme décrivant le rapport de A à B, c'est-à-dire la transformation de A en B, suivie de l'application du programme de complexité minimale à C, permet la production de D, c'est-à-dire d'un mot nouveau, avec des performances

1. Types de similarités déjà présentes dans l'Encyclopédie où la « raison de la dénomination commune » est soit « d'attribution » soit de « proportion » [6].

élevées sur un jeu de plusieurs millions d'analogies morphologiques en une dizaine de langues.

Mais le raisonnement analogique n'est pas exactement l'analogie. Dans la vie de tous les jours, hélas, le mot analogie prend facilement le sens de raisonnement par analogie, voire de simple comparaison, et est assez souvent lié à des raisonnements fallacieux [2]. L'analogie, dans son acception mathématique [3], n'est pas tout à fait la même chose. Le présent article traite de l'analogie mathématique. Il s'agit d'une relation sur un quadruplet qui ne privilégie pas un rapport en particulier, par exemple celui de A à B de préférence à celui de C à D, ni ne privilégie une direction particulière, par exemple celle de A à B de préférence à celle de B à A. Cette vue ne privilégie pas non plus l'un des termes de préférence aux autres, même s'ils ne sont pas tous interchangeables. Les propriétés d'interchangeabilité seront rappelées dans les sections 5.2 et 5.3. Elles ont déjà fait l'objet d'observations depuis l'antiquité en passant par l'Encyclopédie [6] pour continuer à exploiter cette référence.

Le présent article est organisé comme suit. La section 3 discute le lien entre analogies mathématiques et certaines moyennes telles que la moyenne géométrique ou la moyenne arithmétique. Ceci motive la notion de moyenne généralisée définie en terme d'un paramètre (puissance) p que nous rappelons en section 4. Munis de cette notion, nous introduisons dans la section 5 *l'analogie pour la puissance p* et étudions ses propriétés telles que la réflexivité, la symétrie et la transitivité de la conformité, ainsi que les propriétés d'interchangeabilité des termes. Nous poursuivons avec des résultats de réductions, notamment, à une forme canonique en section 6. Nous examinons l'existence du paramètre p pour un quadruplet donné dans la section 7. Nous discutons de l'unicité de p et de l'absence d'unicité dans certains cas particuliers dans les sections 7.2 et 8. L'article se termine sur des perspectives dans la section 9.

2 Notations et terminologie

Dans la suite, les lettres a, b, c et d (et même e et f là où elles sont utiles) désignent, sauf indication du contraire, des éléments de $\mathbb{R}^+ \setminus \{0\}$, autrement dit $]0;+\infty[$.

Nous utilisons les majuscules A, B, C et D lorsque nous parlons de l'analogie en général. On note classiquement une analogie par $A : B :: C : D$. On lira le symbole :: *conformité* pour éviter les termes d'égalité ou d'identité, trop liés à une notion de relation d'équivalence. Le symbole : est celui du *rapport*. On lira *conformité en p* le symbole $::^p$ introduit plus bas.

Dans une analogie $A : B :: C : D$, les termes A et D sont appelés les *extrêmes* et les termes B et C sont appelés les *moyens*.

3 Arrière-plan sur les moyennes et l'analogie

Nous étudions le lien entre analogie et moyennes, ou, plus exactement, moyennes généralisées.

Pour faire un rappel historique rapide sur la notion mathématique d'analogie dans l'antiquité grecque, et lier la notion à celle de moyenne, rappelons que, selon certains chercheurs (cf. [33]), le mot ἀναλογία (« encore le (même) rapport ») aurait peut-être émergé à la suite d'une période d'indécision sur la dénomination de l'analogie continue (το ἀναλογον) c'est-à-dire $A : B :: B : D$ (noter la répétition de B) et de l'analogie discrète c'est-à-dire $A : B :: C : D$. Pour bien faire la différence, l'analogie discrète aurait même été appelée ἀνακολουθία sous le calame d'un certain Speusippe [23]. L'analogie étudiée primitivement par Euclide, qui empruntait à Eudoxe, aurait donc peut-être été plutôt l'analogie continue.

Dans l'*Éthique à Nicomaque*, Aristote répète ce fait qu'il existe deux types d'analogies appelées chacune respectivement discrète et continue, la première impliquant quatre termes, la seconde trois termes, dont l'un est répété.

> ἡ γὰρ ἀναλογία ἰσότης ἐστὶ λόγων, καὶ ἐν τέτταρσιν ἐλαχίστοις. ἡ μὲν οὖν διῃρημένη ὅτι ἐν τέτταρσιν, δῆλον. ἀλλὰ καὶ ἡ συνεχής [...] δὶς οὖν ἡ τοῦ β εἴρηται [...] ([2], 1131 a29 – 1131 b2) [4]

Dans le cas d'une analogie continue, avec la division comme rapport ou raison [5], on peut calculer b en fonction de a et d :

$$a \div b = b \div d \quad \Leftrightarrow \quad b^2 = a \times d \quad \Leftrightarrow \quad b = \sqrt{a \times d}.$$

On sait que cette formule pour b est celle de la moyenne géométrique de a et d.

Si le rapport est la soustraction, on a alors, toujours pour une analogie continue :

$$a - b = b - d \quad \Leftrightarrow \quad 2 \times b = a + d \quad \Leftrightarrow \quad b = \frac{1}{2}(a + d)$$

On voit que b est la moyenne arithmétique de a et d.

De ce qui précède, on comprend que l'analogie continue est intimement liée à la notion de moyenne.

2. Voir dans l'Encyclopédie [6], à l'article Analogie, l'exemple des enfants nés sous le signe du Lion supposés d'humeur martiale par simple rapprochement du nom de la constellation et des attributs de l'animal : « s'imaginer que les enfans qui naissoient sous cette constellation étoient d'humeur martiale : c'est une erreur. »

3. L'Encyclopédie, encore : « Analogie, en Mathématique, est la même chose que proportion, ou égalité de rapport. Voyez Proportion, Rapport, Raison. (O) »

4. « En effet la proportion est une égalité de rapports et en quatre [termes] au moins. Et que la [proportion] disjointe soit effectivement en quatre [termes], c'est évident. Mais aussi la [proportion] continue ; [...] de sorte que si B est posée deux fois, [...] »

5. Le rapport appelé raison jusqu'à récemment (lat. *rationem* > fr. raison) désigne originellement la division. Voir la traduction du xvii[e] siècle des Élements d'Euclide due à Henrion [12].

Analogie	Moyenne	Réécriture	⇔	Expr. de b	⇔	Formule de moyenne
$(a-b):(b-c)=a:a$	arithmétique	$(a-b):(b-c)=1$	$\Leftrightarrow$	$b=\frac{1}{2}(a+c)$	$\Leftrightarrow$	$b=\frac{1}{2}(a+c)$
$(a-b):(b-c)=a:b$	géométrique	$(ab-b^2):(ab-ac)=ab:ab$	$\Leftrightarrow$	$b^2=ac$	$\Leftrightarrow$	$\log b=\frac{1}{2}(\log a+\log c)$
$(a-b):(b-c)=a:c$	harmonique	$(ca-cb):(ab-ac)=ac:ac$	$\Leftrightarrow$	$b=\frac{2ac}{a+c}$	$\Leftrightarrow$	$\frac{1}{b}=\frac{1}{2}(\frac{1}{a}+\frac{1}{c})$

Tableau 1 – Quelques médiétés tirées de [23]. La conformité est ici l'égalité (=) et le rapport la division (:). On obtient les trois moyennes classiques, arithmétique, géométrique et harmonique, à partir d'analogies dont les trois premiers membres sont les mêmes. Seul le quatrième terme varie et prend successivement les valeurs a, b et c.

4 Moyennes généralisées

L'étude des liens entre les différentes moyennes imaginables et l'analogie a donné lieu à des travaux par les Pythagoriciens et les mathématiciens du Moyen-Âge. Il faut citer en particulier les études sur les médiétés (gr. $\mu\varepsilon\sigma\acute{o}\tau\eta\varsigma$, lat. medietas ou mediocritas) (cf. [23]). Le tableau 1 illustre une manière, entre autres, d'extraire les trois moyennes arithmétique, géométrique et harmonique à partir d'analogies similaires, mais discrètes.

Mais la généralisation de la notion de moyenne, sans relation avec l'analogie, au-delà des seules moyennes arithmétique, géométrique ou harmonique, a été donnée en 1889 dans un article signé d'Hölder [16]. Le propos du présent article est de relier la notion mathématique d'analogie à cette généralisation de la notion de moyenne.

4.1 Définition

La moyenne généralisée de plusieurs nombres $x_1, \dots x_N$ est la valeur

$$m_p(x_1, \dots x_N) = \lim_{r \to p} \sqrt[r]{\frac{1}{N}\sum_{i=1}^{N} x_i^r}$$

pour tout $p \in]-\infty, +\infty[$. En particulier, nous retrouvons :

- la moyenne arithmétique pour $p = 1$;
- la moyenne harmonique pour $p = -1$;
- la moyenne quadratique pour $p = 2$;
- la moyenne géométrique quand p tend vers 0 car

$$m_0(x_1, \dots, x_N) = \lim_{p \to 0} m_p(x_1, \dots x_N) = \sqrt[N]{\prod_{i=1}^{N} x_i}.$$

Enfin, quand p tend vers $+\infty$, c'est le maximum des nombres, $\max(x_1, \dots x_N)$, et quand p tend vers $-\infty$, c'est le minimum des nombres, $\min(x_1, \dots x_N)$. Nous définissons $m_{+\infty}$ et $m_{-\infty}$ comme ces valeurs limites.

4.2 Spécialisation à deux nombres

La moyenne généralisée de deux nombres a et d est la valeur

$$m_p(a, d) = \lim_{r \to p} \sqrt[r]{\frac{1}{2}(a^r + d^r)}$$

pour tout $p \in]-\infty, +\infty[$. La figure 1 illustre la courbe obtenue pour les valeurs particulières $a = 2$ et $d = 5$.

Il est connu que la fonction moyenne généralisée est une fonction continue et strictement croissante. Il est tentant de croire qu'elle possèderait une symétrie centrale par rapport à l'un de ses points, c'est-à-dire par rapport à une certaine valeur de p. Mais attention, en général, il n'en est rien. La figure 1 illustre ce fait.

Avant de passer aux résultats, mentionnons que, dans la suite, nous ne détaillons pas toutes les démonstrations car elles adoptent toujours la même structure. Dans le cas général où p n'est ni 0, ni infini, la limite n'est pas nécessaire, la prise de la racine p-ième non plus, le facteur un demi peut être éliminé et les démonstrations peuvent alors exploiter directement la formule $a^p + d^p$ ou bien l'égalité $a^p + d^p = b^p + c^p$. Dans le cas $p = 0$, les formules de la moyenne géométrique sont utilisées : $a \times d$ ou $a \times d = b \times c$. Enfin dans les deux cas infinis, les formules avec les min et max sont utilisées.

5 Analogie par moyennes généralisées

5.1 Définition

Sur quatre nombres réels positifs, on définit *l'analogie pour la puissance p* de la façon qui suit :

$$\begin{aligned} a : b ::^p c : d &\overset{\text{déf.}}{\Leftrightarrow} m_p(a, d) = m_p(b, c) \\ &\Leftrightarrow \lim_{r \to p} \sqrt[r]{\frac{1}{2}(a^r + d^r)} = \lim_{r \to p} \sqrt[r]{\frac{1}{2}(b^r + c^r)} \end{aligned}$$

On dira qu'il y a une analogie entre quatre nombre réels positifs a, b, c et d, si et seulement il existe un p tel que la moyenne généralisée en p des extrêmes a et d soit égale à

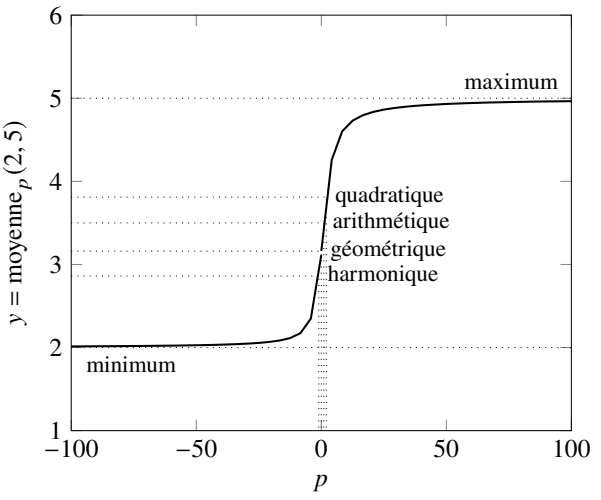


FIGURE 1 – Valeurs des moyennes généralisées de a et d pour $a = 2$ et $d = 5$, avec p en abscisse. On y voit les valeurs des moyennes harmonique (2,86), géométrique ($\sqrt{2 \times 5} = \sqrt{10} \simeq 3{,}16$), arithmétique (3,5) et quadratique (3,81). De même, les limites en $-\infty$ et $+\infty$ sont les minimum et maximum, c'est-à-dire respectivement 2 et 5. Noter que la courbe ne possède pas de symétrie centrale par rapport à l'un de ses points.

la moyenne généralisée en p des moyens b et c. La question est de savoir premièrement à quelle condition on peut trouver un p tel que l'analogie tienne et deuxièmement de déterminer ce p.

Mais il faut d'abord vérifier que la définition donnée ci-dessus correspond bien aux idées que l'on se fait ordinairement de l'analogie mathématique. Il faut examiner si l'on a la réflexivité et la symétrie [6] pour $::^p$ et si les huit formes équivalentes de l'analogie existent bien ici.

5.2 Réflexivité, symétrie et transitivité de $::^p$

Pour la réflexivité de $::^p$, il est trivial de constater que, pour deux nombres positifs quelconques a et b, pour tout p, on a toujours

$$a : b ::^p a : b, \tag{1}$$

c'est-à-dire,

$$\lim_{r \to p} \sqrt[r]{\frac{1}{2}(a^r + b^r)} = \lim_{r \to p} \sqrt[r]{\frac{1}{2}(b^r + a^r)}. \tag{2}$$

Nous visualisons par des grisés le fait que, si le premier a de (1) correspond au premier a de (2), en revanche, le premier b à gauche de la conformité en p dans (1) correspond au deuxième b à droite du signe égal dans l'équation (2).

6. La transitivité n'est pas requise d'habitude pour l'analogie en général. Une simple relation de ressemblance et non d'équivalence suffit. De là notre insistance à dire conformité pour :: et non pas égalité ou identité.

La symétrie de $::^p$ s'énonce comme suit :

$$a : b ::^p c : d \quad \Leftrightarrow \quad c : d ::^p a : b.$$

Elle est vérifiée car :

$$\begin{aligned} a : b ::^p c : d \quad &\Leftrightarrow \quad m_p(a, d) = m_p(b, c) \\ &\Leftrightarrow \quad m_p(b, c) = m_p(a, d) \\ &\Leftrightarrow \quad m_p(c, b) = m_p(d, a) \\ &\Leftrightarrow \quad c : d ::^p a : b. \end{aligned}$$

Répétons que la transitivité n'est pas nécessaire en général pour l'analogie. Il se trouve qu'elle existe pour $::^p$. Ce qui s'énonce :

$$\begin{cases} a : b :: c : d \\ c : d :: e : f \end{cases} \Rightarrow \quad a : b :: e : f. \tag{3}$$

Pour le cas général où p est dans $\mathbb{R}^*$, c'est-à-dire différent de 0, on a :

$$\begin{aligned} \begin{cases} m_p(a, d) = m_p(b, c) \\ m_p(c, f) = m_p(d, e) \end{cases} \quad &\Leftrightarrow \quad \begin{cases} a^p - b^p = c^p - d^p \\ c^p - d^p = e^p - f^p \end{cases} \\ &\Rightarrow \quad a^p - b^p = e^p - f^p \\ &\Rightarrow \quad m_p(a, f) = m_p(b, e). \end{aligned}$$

Dans le cas où p tend vers zéro on sait que

$$m_0(a, d) = \sqrt{a \times d}.$$

On a donc facilement, à condition qu'aucun des termes ne soit nul :

$$\begin{aligned} \begin{cases} m_0(a, d) = m_0(b, c) \\ m_0(b, c) = m_0(e, f) \end{cases} \quad &\Leftrightarrow \quad \begin{cases} \sqrt{a/b} = \sqrt{c/d} \\ \sqrt{c/d} = \sqrt{e/f} \end{cases} \\ &\Rightarrow \quad \sqrt{a/b} = \sqrt{e/f} \\ &\Leftrightarrow \quad m_0(a, f) = m_0(b, e). \end{aligned}$$

Dans le cas où p tend vers moins l'infini, on voudrait faire le raisonnement suivant qui bloque à l'endroit mentionné ci-dessous par l'absence d'implication.

$$\begin{aligned} \begin{cases} m_{-\infty}(a, d) = m_{-\infty}(b, c) \\ m_{-\infty}(c, f) = m_{-\infty}(d, e) \end{cases} \quad &\Leftrightarrow \quad \begin{cases} \min(a, d) = \min(b, c) \\ \min(c, f) = \min(d, e) \end{cases} \\ &\nRightarrow \quad \min(a, f) = \min(b, e) \\ &\Leftrightarrow \quad m_{-\infty}(a, f) = m_{-\infty}(b, e). \end{aligned}$$

En général on n'a pas l'implication. Mais si l'on suppose que a, b, c, d sont rangés par ordre croissant et que e et f sont supérieurs ou égaux à d, alors on peut montrer la transitivité. Il en va de même pour plus l'infini pour lequel il faudrait simplement remplacer min par max et inverser le sens des inégalités.

5.3 Huit formes équivalentes de l'analogie

Les formalisations classiques de l'analogie en mathématique autorisent huit formes d'écritures équivalentes de la même analogie en jouant sur la position des termes dans l'analogie.

$$\begin{array}{ll} a : b :: c : d & c : a :: d : b \\ a : c :: b : d & c : d :: a : b \\ b : a :: d : c & d : b :: c : a \\ b : d :: a : c & d : c :: b : a \end{array}$$

L'article Proportion de l'Encyclopédie [30] mentionne les deux plus connues : *invertendo*, l'inversion des rapports $b : a :: d : c$ et *permutando*, la permutation des moyens $a : c :: b : d$, jugée intrinsèquement caractéristique de l'analogie par les Anciens (cf. [2]). L'équivalence entre la forme originale $a : b :: c : d$ et ces deux formes implique les cinq autres. [7]

Pour ce qui est de l'analogie par moyenne généralisée, l'inversion des rapports est donnée par la symétrie de l'égalité dans $m_p(b, c) = m_p(a, d)$. La commutativité de l'addition dans $(b^p + c^p)$, de la multiplication dans $b \times c$ et dans $\min(b, c)$ ou $\max(b, c)$ donne la permutation des moyens. Les huit formes équivalentes de l'analogie sont donc bien là pour l'analogie par moyenne généralisée.

6 Réductions des analogies par moyennes généralisées

Nous allons maintenant montrer que la définition donnée en 5.1 non seulement unifie les notions classiques d'analogie arithmétique et géométrique, mais encore entraîne des équivalences entre une infinité d'analogies en puissance p. Autrement dit, il existe une forme canonique à laquelle on peut réduire toute analogie en puissance p (sauf pour $p = -\infty$ et $p = +\infty$). On peut faire le choix, par exemple, de réduire à l'analogie arithmétique.

6.1 Prise des inverses des termes

On remarque facilement que, pour une analogie en puissance p donnée, l'analogie en la puissance opposée est valable sur les inverses des termes.

$$a : b ::^p c : d \quad \Leftrightarrow \quad \frac{1}{a} : \frac{1}{b} ::^{-p} \frac{1}{c} : \frac{1}{d}$$

6.2 Cas général de la multiplication par un nombre positif

Il est facile d'observer que, pour quatre nombre réels positifs quelconques, l'analogie en p entre ces nombres peut être transformée en une autre analogie de même puissance p, entre ces mêmes nombres multipliés par un nombre positif quelconque. C'est-à-dire :

$$\forall \lambda \in \mathbb{R}^+ \setminus \{0\}, \quad a : b ::^p c : d \;\Leftrightarrow\; \lambda a : \lambda b ::^p \lambda c : \lambda d.$$

Il est en effet facile de montrer que, d'abord dans le cas où p est différent de 0 :

$$\begin{aligned} & a : b ::^p c : d \\ \Leftrightarrow \quad & \sqrt[p]{\frac{1}{2}(a^p + d^p)} = \sqrt[p]{\frac{1}{2}(b^p + c^p)} \\ \Leftrightarrow \quad & \lambda \times \sqrt[p]{\frac{1}{2}(a^p + d^p)} = \lambda \times \sqrt[p]{\frac{1}{2}(b^p + c^p)} \\ \Leftrightarrow \quad & \sqrt[p]{\frac{1}{2}((\lambda a)^p + (\lambda d)^p)} = \sqrt[p]{\frac{1}{2}((\lambda b)^p + (\lambda c)^p)} \\ \Leftrightarrow \quad & \lambda a : \lambda b ::^p \lambda c : \lambda d. \end{aligned}$$

Dans le cas où p est égal à 0, il est aussi facile de montrer que :

$$\begin{aligned} a : b ::^0 c : d \quad \Leftrightarrow \quad & \sqrt{ad} = \sqrt{bc} \\ \Leftrightarrow \quad & \lambda \times \sqrt{ad} = \lambda \times \sqrt{bc} \\ \Leftrightarrow \quad & \sqrt{\lambda^2 ad} = \sqrt{\lambda^2 bc} \\ \Leftrightarrow \quad & \sqrt{(\lambda a)(\lambda d)} = \sqrt{(\lambda b)(\lambda c)} \\ \Leftrightarrow \quad & \lambda a : \lambda b ::^0 \lambda c : \lambda d. \end{aligned}$$

Dans le cas où p tend vers $-\infty$, on a :

$$\begin{aligned} a : b ::^{-\infty} c : d \quad \Leftrightarrow \quad & \min(a, d) = \min(b, c) \\ \Leftrightarrow \quad & \min(\lambda a, \lambda d) = \min(\lambda b, \lambda c) \\ \Leftrightarrow \quad & \lambda a : \lambda b ::^{-\infty} \lambda c : \lambda d. \end{aligned}$$

Et de même pour p tendant vers $+\infty$, en remplaçant min par max.

6.3 Cas particulier de la division par d

Puisque a, b, c et d sont des réels positifs, à condition qu'ils ne soient pas nuls, leurs inverses aussi. On déduit donc de la propriété précédente que l'on peut multiplier par l'inverse de l'un des termes s'il est non nul. Ce terme divisé par lui-même devient 1. Conséquemment, toute analogie en puissance p peut se réduire, en divisant tous les termes par l'un d'entre eux, à une analogie de même puissance où l'un des termes est 1.

Par utilisation des huit formes équivalentes de l'analogie (voir 5.3), il est toujours possible de placer le plus grand terme en dernière position en tant que d. Dans ce cas, en divisant par d, on obtient une analogie de même puissance où le dernier terme est 1, et où tous les autres termes sont inférieurs à 1. Tous les termes sont donc dans $]0; 1]$. On a donc l'équivalence suivante, dans le cas où d est supérieur ou égal aux autres termes :

$$a : b ::^p c : d \quad \Leftrightarrow \quad \frac{a}{d} : \frac{b}{d} ::^p \frac{c}{d} : 1.$$

7. Nous ne nous servirons pas de ce résultat dans la suite de cet article, mais rappelons que les huit formes équivalentes de l'analogie établissent une correspondance avec le groupe des transformations des coins du carré, c'est-à-dire le groupe diédral, noté D_8.

6.4 Réduction à l'analogie arithmétique

Toute analogie en puissance p, avec p différent de $-\infty$ ou $+\infty$, peut être transposée en analogie arithmétique, c'est-à-dire en puissance 1. La dernière colonne du Tableau 1 suggérait déjà ce résultat en exprimant les moyennes géométrique et harmonique au moyen de la moyenne arithmétique par prise du logarithme ou de l'inverse.

Dans le cas général où $p \neq 0$, en prenant la puissance p des termes d'une analogie, on obtient le résultat voulu. L'équivalence suivante énonce ce fait :

$$a : b ::^p c : d \quad \Leftrightarrow \quad a^p : b^p ::^1 c^p : d^p. \tag{4}$$

Cela est prouvé facilement comme suit :

$$\begin{aligned} a : b ::^p c : d \quad &\Leftrightarrow \quad \frac{1}{2}(a^p + d^p) = \frac{1}{2}(b^p + c^p) \\ &\Leftrightarrow \quad a^p : b^p ::^1 c^p : d^p. \end{aligned}$$

Une autre manière de faire est d'écrire simplement les équivalences suivantes, en ayant en tête que l'analogie en puissance 1 est l'analogie arithmétique dans laquelle le rapport est la soustraction et la conformité l'égalité :

$$\begin{aligned} a : b ::^p c : d \quad &\Leftrightarrow \quad a^p - b^p = c^p - d^p \\ &\Leftrightarrow \quad a^p : b^p ::^1 c^p : d^p. \end{aligned}$$

Dans le cas où $p = 0$, la transformation à appliquer pour passer de l'analogie géométrique à l'analogie arithmétique est la prise du logarithme, c'est-à-dire :

$$\begin{aligned} a : b ::^0 c : d \quad &\Leftrightarrow \quad \sqrt{a \times d} = \sqrt{b \times c} \\ &\Leftrightarrow \quad \frac{1}{2}(\ln a + \ln d) = \frac{1}{2}(\ln b + \ln c) \\ &\Leftrightarrow \quad \ln a : \ln b ::^1 \ln c : \ln p. \end{aligned}$$

La propriété n'est pas valable dans les cas de $-\infty$ et $+\infty$. Il n'y a pas de transformation permettant de passer à une analogie arithmétique à partir de la simple égalité des min ou des max des extrêmes et des moyens.

6.5 Réduction à une forme canonique

En combinant les deux réductions précédentes, pour un quadruplet (a, b, c, d) de nombres positifs non-nuls rangés par ordre croissant, s'il existe une analogie de puissance $p \neq 0$ entre ces nombres, alors on peut réduire cette analogie à la forme canonique suivante :

$$a : b ::^p c : d \quad \Leftrightarrow \quad \left(\frac{a}{d}\right)^p : \left(\frac{b}{d}\right)^p ::^1 \left(\frac{c}{d}\right)^p : 1$$

dans laquelle les termes sont rangés comme suit si p est positif :

$$\left(\frac{a}{d}\right)^p \leq \left(\frac{b}{d}\right)^p \leq \left(\frac{c}{d}\right)^p \leq 1,$$

mais comme suit si p est négatif :

$$\left(\frac{a}{d}\right)^p \geq \left(\frac{b}{d}\right)^p \geq \left(\frac{c}{d}\right)^p \geq 1.$$

7 Existence et unicité de p pour un quadruplet quelconque

Nous nous interrogeons à présent sur l'existence d'analogies en p pour un quadruplet donné (a, b, c, d) de nombres réels. Nous supposons ces nombres positifs et non nuls c'est-à-dire dans $\mathbb{R}^+ \setminus \{0\}$. En plus, et cela est important, nous les supposons rangés dans l'ordre $a < b < c < d$, avec inégalité stricte. Nous allons montrer qu'il existe alors un unique p pour lequel il existe une analogie entre a, b, c et d, c'est-à-dire,

$$a < b < c < d \quad \Rightarrow \quad \exists! p \in \mathbb{R} \ : \quad m_p(a, d) = m_p(b, c)$$

autrement dit,

$$\exists! p \in \mathbb{R} \ : \quad \lim_{r \to p} \sqrt[r]{\frac{1}{2}(a^r + d^r)} = \lim_{r \to p} \sqrt[r]{\frac{1}{2}(b^r + c^r)}. \tag{5}$$

Comme, étant donnés quatre nombre réels différents deux à deux, on peut toujours les ordonner, la proposition précédente s'interprète de la façon suivante :

> Quels que soient quatre nombres réels positifs non nuls, différents deux à deux, on peut toujours voir une analogie entre eux et cette analogie est unique.

On peut paraphraser :

> Il y a toujours un angle, et il est unique, sous lequel on peut voir une analogie entre quatre nombres quelconques (positifs, non nuls, différents deux à deux) à condition de les ordonner par ordre croissant.

7.1 Existence de p

Pour prouver les propositions énoncées ci-dessus, on considère la fonction en p suivante :

$$\delta(p) = \lim_{r \to p} \sqrt[r]{\frac{1}{2}(a^r + d^r)} - \lim_{r \to p} \sqrt[r]{\frac{1}{2}(b^r + c^r)}.$$

Cette fonction est simplement la différence entre la partie gauche et la partie droite de l'équation (5).

On établit d'abord que δ est continue. La fonction δ étant la différence de deux fonctions $m_p(a, d)$ et $m_p(b, c)$ toutes deux continues sur $\mathbb{R}$, est continue sur $\mathbb{R}$. En 0, la limite est $\sqrt{ad} - \sqrt{bc}$.

Examinons maintenant les valeurs extrêmes de δ. Considérons le cas où p tend vers $-\infty$. Rappelons que l'on a

supposé a, b, c et d ordonnés strictement de façon croissante. La fonction δ tend donc vers la valeur $\min(a,d) - \min(b,c) = a - b$. Maintenant, a étant inférieur à b, la valeur limite en $-\infty$ est négative :

$$\lim_{p\to-\infty} \delta(p) = a - b < 0.$$

Pour le cas où p tend vers $+\infty$, on a le même raisonnement en remplaçant min par max : $\max(a,d) - \max(b,c) = d - c$. La valeur obtenue, $d - c$ est positive parce que $c < d$:

$$\lim_{p\to+\infty} \delta(p) = d - c > 0.$$

Le fait que δ soit continue et comprise entre deux valeurs l'une négative à gauche, l'autre positive à droite, implique par le théorème des valeurs intermédiaires de Cauchy que δ s'annule, autrement dit qu'il existe au moins une valeur de p définissant une analogie entre les quatre termes a, b, c et d.

7.2 Unicité de p

On montre d'abord que s'il y a deux analogies, alors il y en a en réalité trois. Soient donc p et q les abscisses de deux points d'intersection des courbes des moyennes généralisées de a, d et de b, c. Comme a et d encadrent b et c, la courbe pour b, c passe en p sous celle de a, d en venant de $-\infty$ et au dessus en q en venant de $+\infty$. Elle est donc à la fois en dessous et au dessus entre les deux valeurs p et q. Il existe donc une valeur r entre p et q correspondant à un troisième point d'intersection. Soit r est négatif et l'on a deux valeurs négatives p et r ; soit r est positif et l'on a deux valeurs positives r et q. Ci-dessous on montre que chacun de ces deux cas mène à une contradiction.

Supposons qu'il existe deux valeurs $0 < r < q$ telles que l'on ait l'analogie. On se restreint au cas $0 < a < b < c < 1$ en divisant par d grâce au résultat vu en 6.3.

$$a : b ::^r c : 1 \quad \Leftrightarrow \quad b^r + c^r - 1 = a^r$$

$$\begin{aligned} a : b ::^q c : 1 \quad &\Leftrightarrow \quad b^q + c^q - 1 = a^q \\ &\Leftrightarrow \quad b^q + c^q - 1 = a^r \times a^{q-r} \\ &\Leftrightarrow \quad b^q + c^q - 1 = (b^r + c^r - 1) \times a^{q-r} \\ &\Leftrightarrow \quad \underbrace{\frac{1 - (b^q + c^q)}{1 - (b^r + c^r)}}_{>1} = \underbrace{a^{q-r}}_{<1} \end{aligned}$$

La dernière ligne se justifie de la manière suivante. D'une part $b^q + c^q < b^r + c^r$ car $0 < b < c < 1$ et $0 < r < q$, d'où le rapport supérieur à 1. D'autre part $a^{q-r} < 1$ car $a < 1$ et $q - r > 0$. Cette ligne énonce une contradiction : il ne peut donc exister deux analogies pour le même quadruplet.

Pour deux valeurs négatives $p < r < 0$, la même démonstration vaut en utilisant l'analogie équivalente vue en 6.1 $1/a : 1/b ::^{-p} 1/c : 1/d$.

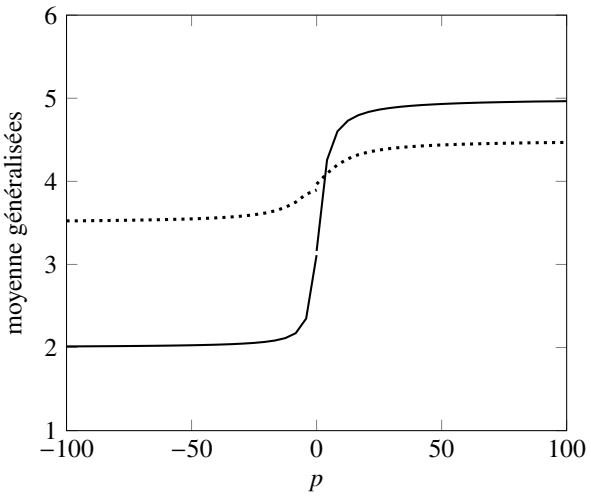


FIGURE 2 – En plein, valeurs des moyennes généralisées de a et d pour $a = 2$ et $d = 5$, avec p en abscisse. En pointillé, même chose avec deux nombres $b = 3{,}5$ et $c = 4{,}5$. La valeur de p telle que l'on ait l'analogie $a : b ::^p c : d$ est donnée par l'intersection des courbes pleine et pointillée. En l'occurrence $p \simeq 3{,}06$.

Pour être complet, comme cas particulier, se demander si $p = 0$ revient seulement à vérifier l'unicité du cas $bc = ad$.

La figure 2 visualise l'unicité de p pour des valeurs particulières de a, b, c et d.

7.3 Calcul de p en pratique

En pratique, le calcul de p pour un quadruplet de nombres positifs peut s'implémenter par recherche dichotomique et p peut être calculé avec une précision fixée à l'avance qui servira de critère d'arrêt. On lancera bien sûr cette recherche dichotomique après s'être assuré que le quadruplet ne correspond pas à un cas particulier de p nul ou infini.

8 Remarques

8.1 Résolution d'analogie

Soient a, b et c des nombres réels positifs non-nuls rangés par ordre croissant et p un nombre réel quelconque, il est trivial qu'il existe une solution unique à l'équation suivante dans le cas où $p \neq 0$:

$$a^p + x^p = b^p + c^p,$$

ou à l'équation suivante correspondant au cas où $p = 0$:

$$\sqrt{a \times x} = \sqrt{b \times c}.$$

Autrement dit, il existe une solution unique à l'équation analogique

$$a : b ::^p c : x.$$

Dans le cas $p = -\infty$ ou $+\infty$, il faut détailler les cas. Dans certains cas, la solution est unique car devant être égale au min ou au max, dans d'autres cas tout nombre supérieur au min ou inférieur au max faisant l'affaire, la solution ne sera pas unique.

Il est clair que la même chose peut être dite pour les cas où l'inconnue n'est pas d mais a ou b ou c.

8.2 Condition à l'existence de p

D'après les huit formes équivalentes de l'analogie, les deux extrêmes sont interchangeables. De même pour les deux moyens. En plus, les moyens et les extrêmes jouent aussi le même rôle par inversion des rapports.

La démonstration de l'existence de p pour le cas où les quatre nombres donnés (différents deux à deux, soulignons) sont ordonnés repose sur la possibilité que les courbes des moyennes généralisées des extrêmes et des moyens s'intersectent. Autrement dit, p existe si et seulement si les extrêmes encadrent les moyens, ou inversement les moyens encadrent les extrêmes.

8.3 Réordonnancement de quatre termes quelconques

Dans les développements donnés dans la sous-section 7.2, nous avons supposé les quatre nombres rangés par ordre croissant. (En fait, comme vu juste au-dessus, nous pouvons nous contenter que les extrêmes encadrent les moyens ou bien l'inverse.) Tout quadruplet n'étant pas forcément ordonné, il faut s'interroger sur les différentes possibilités. Un raisonnement de combinatoire élémentaire (24 possibilités), combiné aux huit formes équivalentes de l'analogie (24 / 8 = 3), montre qu'il n'y a en fait que trois réordonnancements pertinents du point de vue de l'analogie (voir, par exemple [20, p. 119]) :

$$a : b :: c : d \quad \text{ou} \quad a : c :: d : b \quad \text{ou} \quad a : d :: b : c.$$

Ce qui a été dit dans la section 7, à savoir que l'on peut toujours voir une analogie entre quatre nombres, est donc vrai pour un quadruplet quelconque non nécessairement ordonné, à condition de précise quel réordonnancement y est appliqué.

Notons que le cas $a = b = c = d$ est très particulier, car il n'oblige à aucun réordonnancement. Dans ce cas, les 24 différentes écritures sont toutes possibles et équivalentes. Nous poursuivons le traitement de tels cas ci-dessous.

8.4 Cas d'égalité

Avec la remarque ci-dessus, nous venons de rencontrer un cas d'égalité. Nous les étudions maintenant.

Cas d'égalité $b = c$ seulement. Supposons les termes de l'analogie rangés dans l'ordre a, b, c, d. Il est possible que b soit égal à c. C'est l'analogie continue. Dans ce cas, il est clair que, si $a \neq d$ (et a et d tous deux différents de b), p est unique et est donné par l'intersection de la droite horizontale $y = b = c$ avec la courbe des moyennes généralisées de a et d, puisque c'est une fonction continue et strictement croissante.

Cas d'égalité $a = b$ et $c = d$. Dans ce cas, les deux courbes de moyennes généralisées pour a et d et pour b et c se superposent. La puissance pour l'analogie n'est pas unique puisque tout p de $\mathbb{R}$ permet d'écrire l'égalité de la moyenne quelle qu'elle soit. Les valeurs $-\infty$ et $+\infty$ sont, elles aussi, possibles. Il n'y a donc pas unicité de p dans ce cas.

Cas d'égalité $a = d$. Les termes a et d étant les extrêmes, c'est-à-dire les max et les min des termes de l'analogie on a alors nécessairement $a = b = c = d$. Dans ce cas, l'analogie en p est vraie pour tout p dans $]-\infty;+\infty[$ et même pour $p = -\infty$ ou $+\infty$. Il n'y a donc pas unicité de p dans ce cas.

Dans la section 6, nous avions divisé par d, ce qui suppose qu'il soit différent de 0. Si d tend vers 0 et qu'il est le maximum des quatre nombres positifs a, b, c et d, on aura forcément à la limite $a = b = c = d = 0$. Comme vu ci-dessus, l'analogie sera alors vraie pour toute valeur de p, et même pour $p = 0$ en définissant de manière adéquate, par limite, la valeur que l'on attribue à 0^0. Par souci de continuité, on posera à la limite que $0^0 = 0$.

8.5 Cas particulier des booléens

L'analogie entre booléens a été présentée et étudiée dans plusieurs travaux, comme par exemple [28, 10]. Nous établissons ici le lien avec l'analogie en puissance p.

Pour établir le lien entre booléens et nombres réels, nous notons évidemment les valeurs faux et vrai par 0 et 1. Il y a fondamentalement trois analogies possibles entre booléens :

$$0 : 0 :: 0 : 0 \quad \text{ou} \quad 0 : 0 :: 1 : 1 \quad \text{ou} \quad 1 : 1 :: 1 : 1$$

La deuxième analogie peut en particulier être réécrite de façon équivalente en $0 : 1 :: 0 : 1$, ou en $1 : 1 :: 0 : 0$, ou encore en $1 : 0 :: 1 : 0$ grâce aux formes équivalentes.

Les trois analogies fondamentales ci-dessus correspondent au cas $a = b = c = d$ pour la première et la troisième et au cas $a = b$ et $c = d$ pour la deuxième. Il est donc intéressant de noter que dans tous ces cas, de par ce qui a été vu plus haut pour les cas d'égalité, il n'y a pas unicité de p, et toutes les valeurs, y compris les valeurs infinies sont valables.

Mais il est aussi possible d'envisager l'analogie

$$0 : 1 :: 1 : 0$$

qui est équivalente à $1 : 0 :: 0 : 1$ par inversion des rapports (cf. [19] pour une occurrence en pratique de ce cas).

Ces deux formes équivalentes de la même analogie s'expliquent en considérant que le rapport est la négation logique. L'égalité des rapports fait alors leur validité.

Or cette analogie ne constitue pas un prolongement du modèle qui étend les formules d'analogie entre ensembles aux booléens [21] ou encore ne permet pas une justification fondée sur la minimalité de complexité algorithmique [29]. Cette analogie n'entre pas non plus dans le modèle d'analogie en puissance p pour la simple raison que les termes ne sont pas rangés dans un ordre croissant quelles que soient les formes équivalentes considérées.

9 Conclusion et perspectives

Nous avons étudié le lien entre analogie et moyennes, ou plus exactement, moyennes généralisées, et nous nous sommes servis de cette notion comme levier pour généraliser et définir les analogies en puissance p. Les analogies classiques, arithmétique et géométrique, ne sont évidemment que les cas particuliers en $p = 0$ et en $p = 1$ d'analogies en puissance p.

En particulier, nous avons montré que, quels que soient quatre nombres réels positifs, il est toujours possible de voir une analogie entre eux, grâce à un réordonnancement et à une puissance bien choisie unique si les nombres sont différents deux à deux. On peut paraphraser et dire qu'il y a généralement une unique perspective – un réordonnancement et une puissance – sous laquelle on peut voir une analogie entre quatre nombres positifs quelconques.

Pour un problème particulier à traiter, la question se posera de savoir quels sens prennent le réordonnancement nécessaire pour trier les nombres et la valeur numérique de la puissance. Il ne semble pas *a priori* que la valeur de la puissance exprime une force quelconque de l'analogie, mais son interprétation devra sans doute se faire en fonction des données du problème traité.

Nous avons aussi montré que toute analogie peut se réduire (par bijection) à une forme canonique, ce qui donne lieu à une infinité d'analogies équivalentes. En particulier, cela montre que toutes les analogies, y compris les analogies classiques, peuvent être traitées au moyen d'une analogie arithmétique équivalente.

Ce travail ouvre de nouvelles pistes. Tout d'abord nous nous sommes restreints au cas de valeurs positives, et il est naturel de s'interroger sur l'extension à tout les réels pour ouvrir la voie au traitement de représentations comprenant des valeurs négatives. On peut même envisager une extension aux nombres complexes.

Un autre piste, déjà mentionnée ci-dessus, concerne la sémantique du paramètre p qui, d'une certaine façon, résume l'information implicite dans a, b, c et d à leur analogie en p, soit $a : b ::^p c : d$. On peut donc s'interroger sur la possibilité de fusionner l'information contenue dans un quadruplet de nombres par la seule donnée de la puissance de leur analogie.

Remerciements

Les travaux présentés dans cet article sont le fruit d'une collaboration rendue possible par le séjour en sabbatique du premier auteur auprès de l'organisme du second auteur. Les auteurs adressent leurs remerciements à la section pour la promotion de la recherche de l'université Waseda et à la chaire internationale du LORIA pour leurs financements.

Les travaux présentés dans cet article ont aussi bénéficié d'une subvention de recherche de la Société japonaise pour la promotion de la science, de type Kiban C, n° 21K12038, intitulée « Algorithmes théoriquement fondés pour l'extraction automatique de jeux de tests d'analogie en traitement automatique des langues ».

Aussi, ces travaux de recherche ont été partiellement soutenues par le projet ANR *Analogies : from theory to tools and applications* (AT2TA), ANR-22-CE23-0023.

Références

[1] Alsaidi, Safa, Amandine Decker, Puthineath Lay, Esteban Marquer, Pierre Alexandre Murena et Miguel Couceiro: *On the transferability of neural models of morphological analogies*. Dans *Workshop on Advances in Interpretable Machine Learning and Artificial Intelligence (AIMLAI 2021)*, tome 1 de *PKDD/ECML Workshops*, pages 76–89, Bilbao/Virtual, Spain, 2021. https://inria.hal.science/hal-03313591.

[2] Aristote: *Éthique à Nicomaque*. Librairie philosophique J. Vrin, Paris, [1er tirage 1990] édition, 1997. Trad. J. Tricot.

[3] Badra, Fadi et Marie-Jeanne Lesot: *CoAT-APC: When analogical proportion-based classification meets case-based reasoning*. Dans Reuss, Pascal et Jakob Michael Schönborn (rédacteurs): *Workshop Proceedings of the 30th International Conference on Case-Based Reasoning (ICCBR 2022)*, tome 3389 de *CEUR Workshop Proceedings*, pages 43–56, Nancy, France, September 12-15 2022.

[4] Badra, Fadi, Marie Jeanne Lesot, Esteban Marquer et Miguel Couceiro: *Some perspectives on similarity learning for case-based reasoning and analogical transfer*. Dans *Workshop on the Interactions between Analogical Reasoning and Machine Learning (IARML@IJCAI'2023*, tome 3492, pages 16–29, Macao, China, 2023. CEUR-WS.org. https://inria.hal.science/hal-04208969.

[5] Bitton, Yonatan, Ron Yosef, Eliyahu Strugo, Dafna Shahaf, Roy Schwartz et Gabriel Stanovsky: *VASR: visual analogies of situation recognition*. Dans Williams, Brian, Yiling Chen et Jennifer Neville (rédacteurs): *(AAAI 2023) and (IAAI 2023) and (EAAI*

2023), pages 241–249, Washington, DC, USA, February 7-14 2023. AAAI Press.

[6] Chesneau dit Du Marsais, César et Claude Yvon: *Analogie*. Dans *Encyclopédie ou Dictionnaire raisonné des sciences, des arts et des métiers, par une société de gens de lettres*. Chez Briasson, David, Le Breton ou Durand, Paris, 1751–1772.

[7] Collins, Brona et Harold Somers: *Recent Advances in Example-Based Machine Translation*, chapitre EBMT seen as case-based reasoning, pages 115–153. Springer Netherlands, Dordrecht, 2003, ISBN 978-94-010-0181-6. https://doi.org/10.1007/978-94-010-0181-6_4.

[8] Cornuéjols, A., P. A. Murena et R. Olivier: *Transfer learning by learning projections from target to source*. Dans *IDA 2020*, tome 12080 de *LNCS*, pages 119–131, Springer, 2020.

[9] Couceiro, Miguel, Nicolas Hug, Henri Prade et Gilles Richard: *Analogy-preserving functions: A way to extend boolean samples*. Dans *26th International Joint Conference on Artificial Intelligence (IJCAI 2017)*, pages 1–7, Melbourne, Australia, 2017. https://inria.hal.science/hal-01668230.

[10] Couceiro, Miguel, Nicolas Hug, Henri Prade et Gilles Richard: *Analogy-preserving functions: A way to extend boolean samples.* Dans *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI 2017)*, pages 1575–1781, Melbourne, Australia, August 2017.

[11] Couceiro, Miguel et Erkko Lehtonen: *Galois theory for analogical classifiers*. Annals of Mathematics and Artificial Intelligence, 2023. https://inria.hal.science/hal-03665625.

[12] Euclide: *Les quinze livres des éléments géométriques d'Euclide : plus le livre des donnez. . .trad. en françois*. I. Dédin, Paris, 1632.

[13] Fahandar, Mohsen Ahmadi et Eyke Hüllermeier: *Learning to rank based on analogical reasoning*. Dans *AAAI-18*, pages 2951–2958, 2018.

[14] Fahandar, Mohsen Ahmadi et Eyke Hüllermeier: *Analogical embedding for analogy-based learning to rank*. Dans *IDA 2021*, tome 12695 de *LNCS*, pages 76–88, Springer, 2021.

[15] Gentner, Derdre: *Structure mapping: A theoretical model for analogy*. Cognitive Science, 7(2):155–170, 1983.

[16] Hölder, Otto Ludwig: *Ueber einen Mittelwerthssatz*. Nachrichten von der königlichen Gesellschaft der Wissenschaft zu Göttingen und der Georg-Augusts-Universität zu Göttingen, 1889(2):38–47, Jan 1889.

[17] Hüllermeier, Eyke: *Towards analogy-based explanations in machine learning*. Dans *MDAI 2020*, tome 12256 de *LNCS*, pages 205–217, Springer, 2020.

[18] Keane, Mark T. et Barry Smyth: *Good counterfactuals and where to find them: A case-based technique for generating counterfactuals for explainable AI (XAI)*. Dans *ICCBR 2020*, tome 12311 de *LNCS*, pages 163–178, Springer, 2020.

[19] Klein, Sheldon: *Culture, mysticism and social structure and the calculation of behavior*. Dans *Proceedings of the European Conference on Artificial Intelligence (ECAI 1982)*, pages 141–146, 1982.

[20] Lepage, Yves: *De l'analogie rendant compte de l'analogie en linguistique*. Mémoire d'abilitation à diriger les recherches, Université Joseph Fourier, Grenoble, mai 2003. https://theses.hal.science/tel-00004372.

[21] Lepage, Yves: *Formulae for the solution of an analogical equation between Booleans using the Sheffer stroke (NAND) or the Pierce arrow (NOR)*. Dans Couceiro, Miguel, Pierre Alexandre Murena et Stergos Afantenos (rédacteurs): *Proceedings of the Workshop Interactions between analogies and machine learning, colocated with IJCAI 2023 (IARML@ IJCAI 2023)*, pages 3–14, August 2023. https://ceur-ws.org/Vol-3492/paper1.pdf.

[22] Lieber, Jean, Emmanuel Nauer et Henri Prade: *When revision-based case adaptation meets analogical extrapolation*. Dans *ICCBR 2021*, tome 12877 de *LNCS*, pages 156–170, Springer, 2021.

[23] Michel, Paul Henri: *Les médiétés*. Revue d'histoire des sciences et de leurs applications, 2(2) :139–178, 1949.

[24] Mitchell, Melanie: *Abstraction and analogy-making in artificial intelligence*. CoRR, abs/2102.10717, 2021. https://arxiv.org/abs/2102.10717.

[25] Murena, Pierre Alexandre, Marie Al-Ghossein, Jean Louis Dessalles et Antoine Cornuéjols: *Solving analogies on words based on minimal complexity transformation*. Dans Bessière, Christian (rédacteur): *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20*, pages 1848–1854, juillet 2020. https://doi.org/10.24963/ijcai.2020/256, Main track.

[26] Nagao, Makoto: *A framework of a mechanical translation between Japanese and English by analogy principle*. Dans Elithorn, Alick et Ranan Banerji (rédacteurs): *Proceedings of the international NATO symposium on Artificial and human intelligence*, pages 173–180. Elsevier Science Publishers, NATO, 1984. http://www.mt-archive.info/Nagao-1984.pdf.

[27] Peyre, Julia, Ivan Laptev, Cordelia Schmid et Josef Sivic: *Detecting unseen visual relations using analogies*. Dans *IEEE ICCV 2019*, pages 1981–1990, 2019.

[28] Prade, Henri et Gilles Richard: *Analogical proportions and analogical reasoning - an introduction*. Dans Aha, David W. et Jean Lieber (rédacteurs): *Case-Based Reasoning Research and Development*, pages 16–32, Cham, 2017. Springer International Publishing, ISBN 978-3-319-61030-6.

[29] Prade, Henri et Gilles Richard: *Boolean analogical proportions - axiomatics and algorithmic complexity issues*. Dans Antonucci, Alessandro, Laurence Cholvy et Odile Papini (rédacteurs): *Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, pages 10–21, Cham, 2017. Springer International Publishing, ISBN 978-3-319-61581-3.

[30] Rallier des Ourmes, Jean Joseph: *Proportion*. Dans *Encyclopédie ou Dictionnaire raisonné des sciences, des arts et des métiers, par une société de gens de lettres*. Chez Briasson, David, Le Breton ou Durand, Paris, 1751–1772.

[31] Sadeghi, Fereshteh, C. Lawrence Zitnick et Ali Farhadi: *Visalogy: Answering visual analogy questions*. Dans *NIPS 2015*, pages 1882–1890, 2015.

[32] Turney, Peter D.: *Similarity of semantic relations*. Computational Linguistics, 32(2):379–416, 2006.

[33] Vitrac, Bernard: *La Définition V. 8 des Eléments d'Euclide.* Centaurus, 38(2–3) :97–121, 1996.

[34] Zervakis, Georgios, Emmanuel Vincent, Miguel Couceiro, Marc Schoenauer et Esteban Marquer: *An analogy based approach for solving target sense verification*. Dans *Proceedings of the 6th International Conference on Natural Language Processing and Information Retrieval (NLPIR 2022)*, pages 144–151, Bangkok, Thailand, December 16–18 2022. ACM. https://doi.org/10.1145/3582768.3582794.

Actes JIAF 2024

Proportions analogiques et créativité: une étude préliminaire

Stergos Afantenos[1] **Henri Prade**[1] **Gilles Richard**[1] **Leonardo Cortez Bernardes**[1]

[1]IRIT – CNRS, 118, route de Narbonne, 31062 Toulouse Cedex 9, France
stergos.afantenos@irit.fr
henri.prade@irit.fr
gilles.richard@irit.fr
leonardo.cortez_bernardes@irit.fr

Résumé

Les proportions analogiques sont des énoncés de la forme "a est à b comme c est à d", qui expriment que la comparaison des éléments de la paire (a, b) et de la paire (c, d) donne des résultats semblables. Les proportions analogiques sont créatives dans le sens où, étant donné 3 éléments distincts, on peut calculer la représentation d'un quatrième élément d, distinct des éléments précédents, qui forme avec eux une proportion analogique, si certaines conditions sont remplies. Après une introduction aux proportions analogiques et à leurs propriétés, l'article rapporte les résultats d'une expérience réalisée avec une base de données de descriptions d'animaux et de leur classes, où nous tentons de "créer" de nouveaux animaux à partir de ceux existants, ou de retrouver des animaux rares tels que l'ornithorynque. Nous effectuons une série d'expériences en utilisant des plongements lexicaux de mots ainsi que des caractéristiques booléennes afin de proposer de nouveaux animaux basés sur des proportions analogiques, en comparant l'approche symbolique avec des plongements lexicaux non contextuels, montrant que de tels plongements lexicaux de mots obtiennent également de bons résultats. En outre, nous fournissons une série d'expériences avec des plongements lexicaux de phrases utilisant des plongements lexicaux contextuels qui sont moins concluants. Enfin, nous proposons également un processus créatif plus sophistiqué basé sur les proportions analogiques où l'ensemble des paires (a, b) utilisées est choisi et élargi au moyen d'une opération de préservation des propriétés.

Abstract

Analogical proportions are statements of the form "a is to b as c is to d", which express that the comparison of the elements in pair (a, b) and in pair (c, d) yield similar results. Analogical proportions are creative in the sense that given 3 distinct items, the representation of a 4th item d, distinct from the previous items, which forms an analogical proportion with them, can be calculated, provided certain conditions are met. After an introduction to analogical proportions and their properties, the paper reports the results of an experiment made with a database of animal descriptions and their class, where we try to 'create' new animals from existing ones, retrieving rare animals such as platypus. We perform a series of experiments using word embeddings as well as Boolean features in order to propose novel animals based on analogical proportions, comparing the symbolic approach with non-contextual embeddings showing that such word embeddings obtain also good results. Furthermore we provide a series of experiments with sentence embeddings using contextual embeddings which are less conclusive. Finally, we also propose a more sophisticated creative process based on analogical proportions where the set of pairs (a, b) used is chosen and enlarged by means of a property-preserving operation.

Introduction

La créativité a suscité un grand intérêt depuis longtemps dans les sciences informatiques et en intelligence artificielle [2, 47, 6, 23] avec des applications dans de nombreux domaines. Le raisonnement analogique a toujours été reconnu pour favoriser la créativité, notamment dans la pensée créative et la résolution de problèmes [20, 16, 50]. En effet, le raisonnement analogique établit un parallèle entre deux situations, suggérant que ce qui est vrai ou applicable dans la première situation pourrait également être vrai ou applicable dans la seconde situation qui présente une certaine similitude avec la première.

Les proportions analogiques [38] (PA) sont des relations quaternaires notées $a : b :: c : d$ entre quatre éléments a, b, c, d, qui se lisent "« a est à b comme c est à d ». Dans la suite, a, b, c, d sont représentés au moyen de vecteurs ; ces vecteurs peuvent être constitués soit de valeurs d'attributs soit de plongements lexicaux de mots en anglais "word embeddings") [34]. Les proportions analogiques peuvent être

considérées comme des éléments constitutifs du raisonnement analogique. En effet, elles établissent des parallèles entre les paires ordonnées (a, b) et (c, d). Par exemple, « le veau est à la vache comme le poulain est à la jument » met les bovidés sur un pied d'égalité avec les équidés. Dans de telles proportions analogiques, les quatre éléments sont décrits au moyen du même ensemble d'attributs. Les proportions analogiques ont été appliquées avec succès à la classification [32, 5, 4], à la prédiction de préférences [12, 3], ou pour résoudre des tests de QI de type Raven [8, 43].

Les travaux sur les plongements lexicaux [34] ont montré que les modèles de langage où les mots sont représentés par des vecteurs réels ont le potentiel de respecter les proportions analogiques dans un espace vectoriel, bien que des approches ultérieures aient montré que cela était dû au corpus limité d'analogies qui était utilisé, proposant de meilleures ressources pour tester les analogies [15, 52]. Récemment, [21] ont montré que les grands modèles de langage ont la capacité de résoudre des problèmes de Raven, pourvu qu'ils leur soient présentés dans une description en langage naturel, atteignant au moins le même niveau que les êtres humains. À notre connaissance, les capacités créatives des analogies à produire quelque chose de nouveau n'ont pas été explorées, notamment en langage naturel.

Cet article présente une étude du pouvoir créatif des proportions analogiques. Ce pouvoir repose sur leur capacité à produire un quatrième élément à partir de trois éléments (à condition que certaines conditions soient remplies). L'article est organisé en deux sections principales. La première rappelle les définitions et propriétés de base des proportions analogiques, ainsi qu'une nouvelle procédure avancée pour la créativité guidée. La seconde propose une série d'expériences utilisant l'ensemble de données Zoo [1]. Initialement, nous générons des descriptions symboliques d'animaux à partir de celles existantes et vérifions si ces descriptions existent déjà dans la base de données. Nous menons ensuite deux autres séries d'expériences : l'une utilisant des représentations vectorielles de mots, l'autre traitant de phrases descriptives. Ces expériences utilisent des incorporations de mots, soit par le biais d'incorporations statiques GloVe [35], soit par des incorporations de phrases contextuelles avec BERT [10, 45]. Cela est complété par un exemple de matrice de Raven résolu au moyen de proportions analogiques, ainsi qu'une discussion conclusive.

Proportions Analogiques

Cette section est structurée en cinq sous-parties : i) rappelant la modélisation logique booléenne, les postulats et les propriétés des proportions analogiques (PA), ii) fournissant un exemple montrant leur pouvoir créatif, iii) traitant des valeurs d'attributs nominaux, iv) introduisant des PA imbriquées, et v) traitant des PA de mots.

Tables de vérité, postulats et propriétés

Un modèle logique d'une PA «a est à b comme c est à d» où a, b, c, d sont des variables booléennes est donné par la formule suivante qui dit que *a diffère* de *b* comme *c diffère* de *d* et que *b diffère* de *a* comme *d diffère* de *c*" [33] :

$$a : b :: c : d = ((a \wedge \neg b) \equiv (c \wedge \neg d)) \wedge ((\neg a \wedge b) \equiv (\neg c \wedge d))$$

Cette expression est vraie pour les 6 valuations données dans la Table 1 et fausse pour les $2^4 - 6 = 10$ autres valuations possibles. C'est le modèle booléen minimal qui

a	b	c	d
0	0	0	0
1	1	1	1
0	0	1	1
1	1	0	0
0	1	0	1
1	0	1	0

TABLE 1 – Valuations booléennes validant $a : b :: c : d$

satisfait les trois postulats de base suivants (inspirés des proportions numériques) qu'une PA doit respecter [37] :

— *réflexivité* : $a : b : a : b$;
— *symétrie* : $a : b :: c : d \Rightarrow c : d :: a : b$;
— *permutation centrale* : $a : b :: c : d \Rightarrow a : c :: b : d$.

En conséquence, nous avons :

— $a : a : b : b$ (*identité*),
— $a : b :: c : d \Rightarrow b : a :: d : c$ (*inversion interne*), et
— $a : b :: c : d \Rightarrow d : b :: c : a$ (*permutation des extrêmes*).

Remarquablement, les proportions analogiques booléennes sont *indépendantes du code*, c.-à-d. que $a : b :: c : d \Rightarrow \neg a : \neg b :: \neg c : \neg d$. Ainsi, toute propriété utilisée pour décrire des éléments peut être encodée de manière positive ou négative.

Nous supposons que les éléments considérés sont représentés par des *vecteurs* booléens avec n composantes correspondant à n valeurs de caractéristiques, c'est-à-dire $\vec{a} = (a_1, ..., a_n)$, etc. Une proportion analogique " $\vec{a}$ est à $\vec{b}$ comme $\vec{c}$ est à $\vec{d}$ ", notée $\vec{a} : \vec{b} :: \vec{c} : \vec{d}$, est définie composante par composante :
$\vec{a} : \vec{b} :: \vec{c} : \vec{d}$ si et seulement si $\forall i \in [1, n], a_i : b_i :: c_i : d_i$

Les proportions analogiques sont créatives

À titre d'illustration, considérons le problème d'analogie géométrique de la Figure 1 ci-après [8, 38]. Il peut être codé avec cinq prédicats booléens : *Rectangle (R), PointNoir (PN), Triangle (T), Cercle (C), Ellipse (E)*, dans cet ordre

1. https ://archive.ics.uci.edu/dataset/111/zoo

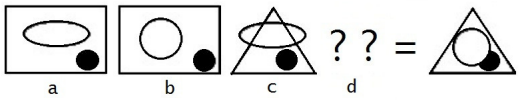


FIGURE 1 – Un problème d'analogie géometrique

dans le Tableau 2, où a, b, c sont codés. Chaque colonne est une équation de PA : $a_i : b_i :: c_i : x_i$.

Une équation $a : b :: c : x$ *n'a pas toujours une solution*. En effet, les équations $1 : 0 :: 0 : x$ et $0 : 1 :: 1 : x$ n'ont pas de solution. Si $a : b :: c : x$ a une solution, elle est *unique*. Elle est donnée par $x = c \equiv (a \equiv b)$, où $\equiv$ représente le connecteur d'équivalence. C'est le cas dans l'exemple, où $\vec{x} = \vec{d} = (01110)$ dans la Table 2, représenté sur la droite de la Figure 2.

Il convient de souligner que dans ce cas, la description de $\vec{d}$ est calculée directement à partir de celles de $\vec{a}, \vec{b}, \vec{c}$, ce qui contraste avec ce genre de tests de QI où la réponse doit être trouvée parmi un ensemble de solutions candidates. En effet, le premier programme pour résoudre de tels problèmes [11], dans les premières années de l'IA, consistait à sélectionner la solution parmi les solutions candidates sur la base d'une distance tenant compte de la similarité des transformations pour passer de $\vec{a}$ à $\vec{b}$ et de $\vec{c}$ à $\vec{x}$ (où $\vec{x}$ est la solution candidate considérée). Le fait que la description de $\vec{x}$ soit maintenant directement calculée à partir de $\vec{a}, \vec{b}, \vec{c}$ montre le *pouvoir créatif* de la proportion analogique. De plus, $\vec{d} = \vec{x}$ *est distinct de* $\vec{a}$, $\vec{b}$ *et* $\vec{c}$. C'est général pour $\vec{a} : \vec{b} :: \vec{c} : \vec{d}$ dès que $\vec{a}$ est distinct de $\vec{b}$ et distinct de $\vec{c}$ (sur au moins deux attributs).

	R	PN	T	C	E
$\vec{a}$	1	1	0	0	1
$\vec{b}$	1	1	0	1	0
$\vec{c}$	0	1	1	0	1
$\vec{x}$	?	?	?	?	?

TABLE 2 – Codage de l'exemple. $\vec{x} = (01110)$

Attributs nominaux

La description des éléments peut impliquer des attributs nominaux, c'est-à-dire des attributs avec un domaine fini ayant une cardinalité supérieure à 2. Alors $a : b :: c : d$ est vrai pour les variables nominales si et seulement si :

$$(a, b, c, d) \in \{(s, s, s, s), (s, t, s, t), (s, s, t, t) | \, s, t \in \mathcal{A}, s \neq t\}$$

où s, t représentent n'importe quelle valeur du domaine d'attributs $\mathcal{A}$ (comme suggéré pour la première fois dans [36], voir aussi [5]). Cela généralise le cas booléen et préserve toutes les propriétés rapportées pour ce cas.

Un exemple de proportions analogiques imbriquées

Nous illustrons maintenant les valeurs nominales avec un exemple de PA (de [25]) entre quatre phrases $\boldsymbol{a}$ = "les filles détestent la lumière", $\boldsymbol{b}$ = "les garçons aiment la lumière", $\boldsymbol{c}$ = "les femmes détestent l'obscurité", $\boldsymbol{d}$ = "les hommes aiment l'obscurité", vues comme des tuples ordonnés de valeurs nominales, par rapport aux 3 attributs "sujet", "verbe", "complément". Les 4 phrases $\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}, \boldsymbol{d}$ sont donc représentées par des vecteurs à 3 composantes $\vec{a}, \vec{b}, \vec{c}, \vec{d}$:

	sujet	verbe	complément
$\vec{a}$	filles	détestent	lumière
$\vec{b}$	garçons	aiment	lumière
$\vec{c}$	femmes	détestent	obscurité
$\vec{d}$	hommes	aiment	lumière

TABLE 3 – Proportion analogique entre phrases

Évidemment, les valeurs dans la colonne "sujet" du Tableau 3 ne forment pas une proportion analogique par elles-mêmes, puisque quatre valeurs distinctes sont impliquées (alors que `déteste : aime :: déteste : aime`, et `lumière : lumière :: obscurité : obscurité` sont des proportions analogiques claires en termes de valeurs nominales). Cependant, `filles : garçons :: femmes : hommes` peut être considéré comme une écriture compacte d'une PA entre des descriptions en termes d'une collection de caractéristiques booléennes comme dans le Tableau 4, où une PA est vérifiée dans chaque colonne. Ainsi, $\vec{a} : \vec{b} :: \vec{c} : \vec{d}$, et donc $\boldsymbol{a} : \boldsymbol{b} :: \boldsymbol{c} : \boldsymbol{d}$ sont vraies. La PA $\vec{a} : \vec{b} :: \vec{c} : \vec{d}$ est une PA *imbriquée* car elle implique un attribut dont les quatre valeurs ne forment pas l'un des trois schémas de base d'une proportion analogique nominale, alors que ces valeurs sont associées à des descriptions vectorielles qui forment elles-mêmes une PA.

	male	*femelle*	*jeune*	*adulte*	*humain*
`filles`	0	1	1	0	1
`garçons`	1	0	1	0	1
`femmes`	0	1	0	1	1
`hommes`	1	0	0	1	1

TABLE 4 – Les filles sont aux garçons ce que les femmes sont aux hommes

Proportions analogiques sur les mots

Dans l'exemple du Tableau 3, les valeurs des composantes du vecteur sont des mots. Les mots peuvent eux-mêmes être représentés par des plongements lexicaux [34]. Les proportions analogiques peuvent être directement définies en termes de telles représentations vectorielles, comme cela a été anticipé dès [46]. Alors, la proportion analogique $\vec{a} : \vec{b} :: \vec{c} : \vec{d}$ est vraie si et seulement si $\vec{a} - \vec{b} = \vec{c} - \vec{d}$, c'est-à-dire, $\forall i, a_i - b_i = c_i - d_i$. Remarquez que cela est conforme à la Table de vérité 1 dans le cas booléen : lorsque $a_i, b_i, c_i, d_i \in \{0, 1\}, a_i : b_i :: c_i : d_i \Leftrightarrow a_i - b_i = c_i - d_i$. Voir [39] pour plus de détails sur cette perspective. Ainsi, les plongements lexicaux des mots 'filles', 'garçons', 'femmes', et 'hommes' devraient être telles que $v_{\vec{filles}} - v_{\vec{garcons}} = v_{\vec{femmes}} - v_{\vec{hommes}}$.

De plus, cette perspective est conforme au calcul des plongements lexicaux des phrases comme la somme des plongements lexicaux des mots de la phrase. En effet, nous avons $(\vec{a} - \vec{b}) + (\vec{a'} - \vec{b'}) = (\vec{c} - \vec{d}) + (\vec{c'} - \vec{d'}) \Leftrightarrow (\vec{a} + \vec{a'}) - (\vec{b} + \vec{b'}) = (\vec{c} + \vec{c'}) - (\vec{d} + \vec{d'})$, ce qui signifie que si deux 4-uplets $(\vec{a}, \vec{b}, \vec{c}, \vec{d})$ et $(\vec{a'}, \vec{b'}, \vec{c'}, \vec{d'})$ de mots forment une PA, leur regroupement additif forme également une PA. En effet, pour confirmer cela, nous avons utilisé des plongements lexicaux de GloVe représentant chaque phrase comme la moyenne des plongements lexicaux de ses mots. Nous avons ensuite calculé $1 - \cos \frac{\vec{b}-\vec{a}}{\vec{d}-\vec{c}}$, qui représente la proximité des vecteurs dans la formation d'une analogie, obtenant une valeur de 0, 8513, montrant que les 4 vecteurs sont proches de former un parallélogramme.

Créativité guidée

Étant donné un ensemble $\mathcal{I}$ d'éléments existants, chacun représenté en termes du même ensemble d'attributs booléens $\mathcal{A}$, la créativité peut consister à produire un nouvel élément, non présent dans $\mathcal{I}$, mais décrit par le même ensemble d'attributs $\mathcal{A}$ [23]. Vu sous cet angle, la créativité semble facile : nous pouvons choisir au hasard des valeurs pour les attributs dans $\mathcal{A}$ et vérifier si le résultat n'est pas déjà dans $\mathcal{I}$. Cependant, avec un tel processus, nous n'avons aucun contrôle sur les valeurs d'attribut qui pourraient être souhaitables.

Dans la section précédente, nous avons proposé une autre option pour générer un nouvel élément à partir de trois éléments connus en utilisant des proportions analogiques. Comme souligné dans [40], la proportion analogique consiste à associer une paire $(\vec{a}, \vec{b})$ avec une paire $(\vec{c}, \vec{d})$. Ainsi, à partir de l'ensemble d'éléments $\mathcal{I}$, on peut construire un ensemble de k paires ordonnées $\mathcal{P} = \{(\vec{a}^j, \vec{b}^j) \mid \vec{a}^j \in \mathcal{I}, \vec{b}^j \in \mathcal{I}, j = 1, \cdots, k\}$. Cet ensemble de paires représente potentiellement les connaissances pouvant être obtenues par comparaison par paires des éléments de $\mathcal{I}$. En prenant un élément $\vec{c}$ comme point de départ que nous aimerions modifier, nous pouvons obtenir un nouvel élément $\vec{x}$ en résolvant, composante par composante, les équations $\vec{a}^j : \vec{b}^j :: \vec{c} : \vec{x}^j$ (lorsque la solution existe), c'est-à-dire $\vec{x^j} = \vec{c} \equiv (\vec{a}^j \equiv \vec{b}^j)$ pour toute paire j. En d'autres termes, nous recherchons l'ensemble des solutions.

$$\mathcal{S} = \{\vec{x} \mid \exists(\vec{a}^j, \vec{b}^j) \in \mathcal{P}, j \in \{1,.., k\} \text{ s.t. } \vec{a}^j : \vec{b}^j :: \vec{c} : \vec{x}\}$$

Notons que : i) si une propriété est acquise (resp. perdue) lors du passage de $\vec{a}^j$ à $\vec{b}^j$, c'est-à-dire $a_i^j = 0, b_i^j = 1$ (resp. $a_i^j = 1, b_i^j = 0$) et que $\vec{c}$ n'a pas la propriété (resp. a la propriété), alors cette propriété est également acquise (resp. perdue) par $\vec{x}^j$; ii) s'il n'y a pas de changement de $\vec{a}^j$ à $\vec{b}^j$ sur l'attribut i, c'est-à-dire $a_i^j = b_i^j$, alors $\vec{x}^j$ copiera la valeur de $\vec{c}^j$ pour la valeur de l'attribut i.

Pour mieux contrôler le processus de génération de nouveaux éléments, on peut utiliser un ensemble de paires *sélectionnées*. Plus précisément, supposons que les éléments dans $\mathcal{I}$ à partir desquels l'ensemble $\mathcal{P}$ de paires est construit représentent des objets / profils / situations appartenant à un univers réel, et ensuite, que chaque paire ordonnée $(\vec{a}^j, \vec{b}^j)$ de vecteurs représente des changements légitimes / faisables / autorisés / précieux de $\vec{a}^j$ à $\vec{b}^j$.

Lorsqu'il n'y a pas de solution dans $\mathcal{S}$ (parce que $\mathcal{P}$ est trop petit), ou lorsque les éléments trouvés ne sont pas considérés comme suffisamment satisfaisants, nous devons envisager l'option d'élargir l'ensemble potentiel $\mathcal{P}$ avec lequel nous commençons, en construisant de nouvelles paires ordonnées à partir des paires déjà présentes dans $\mathcal{P}$. Cela donne naissance à un *processus d'inférence créative* qui tente d'améliorer un élément ou une entité particulière, en tirant parti d'un ensemble de paires ordonnées d'éléments existants, en utilisant le mécanisme basé sur la proportion analogique.

Afin d'agrandir la base initiale de paires, nous calculons de nouvelles paires au moyen d'une opération, notée $\wedge\vee$, introduite dans [41, 40], qui produit une nouvelle paire ordonnée en combinant deux paires. Cette opération $\wedge\vee$ est définie de la manière suivante :

$$(\vec{a}, \vec{b}) \wedge\vee (\vec{c}, \vec{d}) = (\vec{a} \wedge \vec{c}, \vec{b} \vee \vec{d})$$

où la conjonction et la disjonction des éléments sont définies composante par composante :

$\vec{a} \wedge \vec{b} = (a_1 \wedge b_1, ..., a_n \wedge b_n)$;

$\vec{a} \vee \vec{b} = (a_1 \vee b_1, ..., a_n \vee b_n)$.

Clairement, cet opérateur $\wedge\vee$ est commutatif, associatif et idempotent par construction. On peut vérifier que si (a_i, b_i) ou $(c_i, d_i) = (0, 1)$, $(a_i \wedge c_i, b_i \vee d_i) = (0, 1)$. Ainsi, si une propriété i est acquise lors du passage du premier élément d'une paire au deuxième élément de cette paire, alors la propriété est également acquise dans la nouvelle paire résultant de la combinaison par $\wedge\vee$. En revanche, $(a_i \wedge c_i, b_i \vee d_i) = (1, 0)$ seulement si $(a_i, b_i) = (c_i, d_i) = (1, 0)$. Ainsi, si une propriété i est perdue dans la nouvelle paire, c'était déjà le cas dans chacune des paires combinées. L'opération $\wedge\vee$ a le mérite de "cumuler" l'acquisition de caractéristiques en préservant les motifs $(0, 1)$.

Étendre l'ensemble initial $\mathcal{P}$ de paires nous donne plus de chances de créer un nouvel élément d'intérêt, peut-être avec des caractéristiques plus désirables. Plus précisément, nous calculons l'ensemble $\mathcal{S}$ précédemment défini où $\mathcal{P}$ est remplacé par

$$\mathcal{P}' = \{(a^k, b^k) \mid (a^k, b^k) = (a^i, b^i) \wedge\vee (a^j, b^j)$$

tel que $((a^i, b^i), (a^j, b^j)) \in \mathcal{P}^2\}$.

Nous pouvons appliquer cette extension de $\mathcal{P}$ de manière itérative à $\mathcal{P}'$ et ainsi de suite. En raison de l'idempotence de $\wedge\vee$, $\mathcal{P} \subseteq \mathcal{P}'$.

Ensuite, étant donné un élément fixé, choisi, représenté par un vecteur $\vec{c} \in \mathcal{I}$, on peut chercher quel(s) nouvel(s) élément(s) pourrait/peuvent être obtenu(s) en appliquant un changement existant dans la base de paires ordonnées $\mathcal{P}'$.

Nous illustrons maintenant l'ensemble du processus avec un petit exemple librement inspiré de la créativité en littérature. Les éléments que nous considérons sont censés être des descriptions sommaires de romans. Nous avons 6 attributs qui indiquent si le roman i) est écrit en vers ou en prose (`ver.`) ; ii) est épistolaire ou non (`epi.`) ; iii) est un roman d'aventure ou non (`adv.`) ; iv) est un roman d'amour ou non (`rom.`) ; v) se déroule aujourd'hui ou non (`tod.`) ; vi) se déroule dans un pays exotique ou non (`exo.`).

Nous avons 2 paires constituant $\mathcal{P}$:
- $(\vec{a_1}, \vec{b_1}) = ([0,0,0,0,1,0], [0,1,0,0,1,1])$
- $(\vec{a_2}, \vec{b_2}) = ([0,0,0,0,1,0]), [1,0,0,0,1,1])$.

Ainsi, $\vec{a_1}$ fait référence à un roman écrit en prose, non épistolaire, sans aventure, sans romance, se déroulant aujourd'hui dans un pays non exotique. $\vec{b_1}$ est similaire à $\vec{a_1}$ à l'exception qu'il est épistolaire et exotique et puisque $(\vec{a_1}, \vec{b_1}) \in \mathcal{P}$, $\vec{b_1}$ est considéré comme "plus intéressant" que $\vec{a_1}$. La paire $(\vec{a_2}, \vec{b_2})$ peut être lue de manière similaire.

Lorsque nous étendons $\mathcal{P}$ avec l'opérateur $\wedge\vee$ (mais sans effectuer la fermeture complète), nous ajoutons à $\mathcal{P}$ la paire suivante : - $(\vec{a_3}, \vec{b_3}) = ([0,0,0,0,1,0], [1,1,0,0,1,1])$
puisque $(\vec{a_3}, \vec{b_3}) = (\vec{a_1}, \vec{b_1}) \wedge\vee (\vec{a_2}, \vec{b_2})$.
Parce que $\vec{a_1} = \vec{a_2}$, évidemment $\vec{a_1} = \vec{a_3}$, mais cela n'est pas du tout obligatoire, juste une question de simplicité dans l'exemple. Notez que $\vec{b_3}$ cumule les "avantages" de $\vec{b_1}$ et $\vec{b_2}$ par rapport à $\vec{a_1}$. En partant de $\vec{c} = [0,0,0,1,0,0]$, nous observons dans le Tableau 5 que les 3 équations analogiques correspondantes sont solvables. Ainsi, nous avons obtenu 3 nouveaux profils de romans possibles $\vec{x_1}, \vec{x_2}, \vec{x_3}$. Par exemple, la solution de la troisième équation est alors un nouveau type de roman, distinct des 5 vecteurs existants $\vec{a_1}, \vec{b_1}, \vec{b_2}, \vec{x_1}, \vec{x_2}$. Il s'agit d'un roman d'amour exotique écrit en vers et épistolaire, sans aventure, se déroulant dans le passé ou dans le futur. Voir le Tableau 5. Le processus que nous avons décrit garantit que i) les éléments obtenus sont nouveaux, et ii) ils sont obtenus à partir d'un $\vec{c}$ existant sur la base de changements déjà existants, tels qu'observés sur des paires ordonnées d'éléments existants. Les résultats ainsi obtenus sont-ils "intéressants" ? C'est une question complètement différente : tout ce qui est nouveau n'est pas nécessairement intéressant, mais au moins le fait d'être nouveau peut aider à penser différemment.

Dans la Table 5, nous ne donnons qu'un très petit exemple pour expliquer le mécanisme qui nous permet i) d'élargir un ensemble de paires qui présente des changements intéressants qui sont cumulés dans l'élargissement ; ii) et ensuite de produire un nouvel élément amélioré à partir de l'ensemble obtenu de paires ordonnées et d'un élément de référence ; cf. [42] pour une première mise en œuvre. Notons qu'une façon

	ver.	epi.	adv.	rom.	tod.	exo.
$\vec{a_1}$	0	0	0	0	1	0
$\vec{b_1}$	0	1	0	0	1	1
$\vec{c}$	0	0	0	1	0	0
x_1	0	1	0	1	0	1
	ver.	epi.	adv.	rom.	tod.	exo.
$\vec{a_2}$	0	0	0	0	1	0
$\vec{b_2}$	1	0	0	0	1	1
$\vec{c}$	0	0	0	1	0	0
x_2	1	0	0	1	0	1
	ver.	epi.	adv.	rom.	tod.	exo.
$\vec{a_3}$	0	0	0	0	1	0
$\vec{b_3}$	1	1	0	0	1	1
$\vec{c}$	0	0	0	1	0	0
x_3	1	1	0	1	0	1

TABLE 5 – $(\vec{a_3}, \vec{b_3}) = (\vec{a_1}, \vec{b_1}) \wedge\vee (\vec{a_2}, \vec{b_2})$.

d'étudier le pouvoir créatif de ce processus pourrait consister à comparer la distribution de probabilité des valeurs des caractéristiques dans les paires avec lesquelles nous commençons, à la distribution de probabilité des valeurs des caractéristiques dans l'ensemble des éléments produits (en faisant varier ou non l'élément de référence) à l'aide de la divergence de Kullback-Leibler [27].

L'inférence basée sur la proportion analogique fournit une approche à la créativité qui repose sur la recopie de ce qui est observé dans une paire, en termes de changement et de permanence, sur une autre paire, dont le premier élément est connu. De plus, nous avons montré qu'il est possible de combiner des paires ordonnées existantes en de nouvelles paires tout en préservant des caractéristiques souhaitables. Cependant, nous ne prétendons certainement pas que toute forme d'inférence analogique créative, prise au sens large, pourrait être capturée par le mécanisme de transfert que nous proposons.

Expériences

Nous présentons la procédure expérimentale pour le cas de représentations symboliques, ou par plongements lexicaux, avant d'expliquer l'implémentation et de présenter les résultats [2].

Procédure

Nous commençons avec une base de données contenant les descriptions des animaux en termes de caractéristiques booléennes. Nous supposons également qu'une classe est connue pour chaque animal. Ainsi, un animal A est décrit par un vecteur $\vec{a}$ avec sa classe $cl(\vec{a})$. L'idée, étant donné

2. Le code de nos expériences est accessible ici : `https://github.com/Sasufox/TER_Analogy`

un sous-ensemble S de la base de données, est de calculer les solutions des équations de la forme $\vec{a} : \vec{b} :: \vec{c} : \vec{x}$ où $\vec{a}, \vec{b}, \vec{c} \in S$, et de voir si $\vec{x}$ est, ou non, dans la base de données. De plus, nous pouvons utiliser uniquement la restriction des vecteurs $\vec{a}, \vec{b}, \vec{c}$ à un sous-ensemble de caractéristiques. Plus précisément, nous recherchons de petits sous-ensembles de caractéristiques utilisées pour la description des animaux, comme dans l'exemple de la Table 6 (où la solution existe effectivement dans le règne animal). Dans

	allaite leurs petits	*pond des oeufs*
scorpions	0	0
mammiferes	1	0
oiseaux	0	1
?	1	1

TABLE 6 – **?** = *monotrèmes*

cet exemple, la résolution d'équations analogiques "crée" une espèce animale qui pond des œufs et allaite ses petits. Il s'avère que de tels animaux existent : les ornithorynques, les échidnés.

Notez que lors de la recherche de solutions analogiques, nous devons prendre en compte deux problèmes : 1. l'équation doit être solvable pour chaque caractéristique considérée ; 2. $\vec{a} : \vec{b} :: \vec{c} : \vec{x}$ et $\vec{a} : \vec{c} :: \vec{b} : \vec{x}$ ont la même solution. De plus, d'un point de vue de la créativité, les vecteurs $\vec{a}, \vec{b}, \vec{c}$ (ou les sous-vecteurs utilisés) font référence à des animaux qui devraient être suffisamment différents. C'est pourquoi il peut être utile d'imposer les contraintes $cl(\vec{a}) \neq cl(\vec{b}), cl(\vec{a}) \neq cl(\vec{c}), cl(\vec{b}) \neq cl(\vec{c})$.

De plus, nous sommes également intéressés par la résolution d'équations analogiques lorsque les vecteurs sont des plongements de mots, c'est-à-dire en calculant $\vec{x} = \vec{c} + \vec{b} - \vec{a}$, et en recherchant les mots dont le plongement est proche de $\vec{x}$. La question est alors de voir si les résultats sont compatibles avec ceux obtenus à partir des représentations booléennes.

16 attributs	5 attributs les plus importants
22.64%	59.67%

TABLE 7 – Précision avec des vecteurs booléens

Implémentation

Approche symbolique

Nous utilisons l'ensemble de données *Zoo* (voir note de bas de page 2). Cet ensemble de données contient 101 animaux, chacun décrit par 16 caractéristiques et leur classe. Parmi ces 17 caractéristiques, 15 sont binaires tandis que deux sont nominales. La première caractéristique nominale décrit le nombre de pattes que possède l'animal, tandis que l'autre décrit sa classe (mammifère, reptile, etc.).

Les premières expériences ont pris en compte l'ensemble complet des 16 caractéristiques, en excluant la classe de l'animal. Pour la liste complète des 101 animaux, nous avons calculé tous les triplets possibles $\binom{101}{3}$ et nous n'avons conservé que ceux pour lesquels les trois animaux appartenaient à une classe différente. Pour chaque triplet $(\vec{a}, \vec{b}, \vec{c})$, nous avons prédit l'existence, ou non, d'un quatrième élément représentant un animal "potentiel" en utilisant des proportions analogiques pour chaque caractéristique, comme décrit dans les sections précédentes.

Imposer des proportions analogiques pour les 16 caractéristiques complètes est assez restrictif, car il suffit qu'une caractéristique ne soit pas dans la PA pour écarter tout le triplet. Examiner tous les sous-ensembles possibles de caractéristiques de cardinalité 2 ou plus est computationnellement prohibé, car cela entraînerait $\binom{101}{3} \times 2^{15}$ instances. Nous voulions donc identifier le sous-ensemble des caractéristiques les plus *"importantes"* afin de réaliser nos expériences. Pour ce faire, nous avons identifié les caractéristiques partagées par la plupart des animaux et nous avons sélectionné les cinq premières. Ces caractéristiques étaient `a des poils`, `pond des œufs`, `produit du lait`, `venimeux`, et `domestiqué`. Nous avons appliqué la même procédure que précédemment pour ce sous-ensemble de caractéristiques.

Expériences avec GloVe

Dans une autre série d'expériences, nous avons examiné comment les plongements représentant des mots peuvent être utilisés pour la créativité à base d'analogies. Comme expliqué ci-dessus, chaque élément dans un triplet $(\vec{a}, \vec{b}, \vec{c})$ est représenté sous une forme vectorielle avec des valeurs réelles. Nous prédisons un quatrième élément $\vec{d}$ tel que $\vec{d} = \vec{c} + \vec{b} - \vec{a}$. Nous représentons chaque animal A, B, C à l'aide de plongements statiques ou contextuels. Pour les plongements statiques, nous avons choisi les plongements GloVe [35] avec 300 dimensions ("common crawl" avec 840 milliards de jetons). Après avoir calculé $\vec{d}$, nous trouvons les mots représentés par les plongements GloVe les plus proches de ce vecteur en utilisant la distance euclidienne. Notre objectif dans cet ensemble d'expériences était d'examiner si ces mots représentent effectivement des animaux présents dans la base de données Zoo.

Expériences avec BERT

Enfin, nous voulions également tester les plongements contextuels créés à l'aide des architectures Transformer [49]. Nous avons opté pour une approche basée sur l'architecture BERT [10]. Les architectures basées sur Transformer fournissent des plongements qui ne sont pas statiques mais dépendent plutôt de leur contexte, s'appuyant fortement sur un mécanisme d'attention complexe sur un contexte de taille fixe de sous-mots. Pour cette raison, nous n'avons pas pu suivre la même procédure que les

plongements statiques de GloVe. Au lieu de cela, nous avons opté pour la création de phrases pour chaque animal dans la base de données, reflétant les caractéristiques qui décrivent chaque animal. Plus précisément, nous avons choisi les cinq caractéristiques mentionnées précédemment (`a des poils`, `pond des œufs`, `produit du lait`, `venimeux`, et `domestiqué`). Par exemple, pour l'*ornithorynque*, nous avons créé la phrase suivante, reflétant les caractéristiques qui décrivent cet animal :

> *L'ornithorynque est un animal qui a des poils, pond des œufs, produit du lait, n'est pas venimeux et n'est pas domestiqué.*

Chaque phrase ainsi créée a été encodée à l'aide de Sentence-BERT [45]. En raison de la nature non statique des plongements BERT, nous n'avons pas pu rechercher parmi les phrases potentielles dans l'espace de plongement BERT celle qui était la plus proche du vecteur $\vec{d} = \vec{c} + \vec{b} - \vec{a}$. Nous avons donc opté pour la méthode suivante. Une fois que nous avions créé de tels plongements, pour un triplet donné (A, B, C) d'animaux, nous avons calculé la distance euclidienne, pour tout $D \notin \{A, B, C\}$ dans la base de données, entre $\vec{a} - \vec{b}$ et $\vec{c} - \vec{d}$. Nous décrivons les résultats dans la section suivante.

Les descriptions d'animaux mentionnées que nous avons créées incluent l'animal lui-même. En tant que tel, les plongements de phrases sBERT reposent indéniablement fortement sur cette information. Nous voulions donc tester si les plongements de phrases sBERT qui n'utilisent pas d'informations sur l'animal sont capables de simuler l'approche symbolique que nous avons décrite ci-dessus. Nous avons donc procédé en créant des descriptions d'animaux basées sur leurs caractéristiques sans mentionner explicitement l'animal. Nous avons créé deux variantes : a) des descriptions complètes et b) des descriptions omettant les caractéristiques absentes.

Par exemple, pour l'ornithorynque, une description complète serait :

> *C'est un animal qui a des poils, pond des œufs, produit du lait, n'est pas venimeux et n'est pas domestiqué.*

tandis que la description omettant les caractéristiques absentes serait :

> *C'est un animal qui a des poils, pond des œufs et produit du lait.*

Comme dans le cas précédent, pour un triplet donné (A, B, C), nous avons créé des descriptions et avons ainsi obtenu des vecteurs sBERT $\vec{a}, \vec{b}$ et $\vec{c}$ à partir desquels nous avons calculé $\vec{d} = \vec{c} + \vec{b} - \vec{a}$. Nous avons ensuite procédé au calcul des 32 descriptions différentes pour les 5 caractéristiques mentionnées ci-dessus [3] exactement de la même manière que nous l'avons fait pour les animaux, et obtenu des vecteurs d_i sBERT avec $i \in [1 \dots 32]$ pour chacune des combinaisons possibles. Nous avons ainsi pu calculer dans un ordre décroissant la distance euclidienne entre d_i et $\vec{d} = \vec{c} + \vec{b} - \vec{a}$. De plus, pour chacun des 32 d_i, nous avons pu récupérer le sous-ensemble des animaux partageant exactement les mêmes caractéristiques. Cela nous a permis de comparer directement une approche basée sur BERT avec l'approche symbolique que nous avons décrite ci-dessus. Nous décrivons nos résultats dans la section suivante.

3. C'est-à-dire pour l'ensemble de caractéristiques suivant : `a des poils`, `pond des œufs`, `produit du lait`, `venimeux`, et `domestiqué`.

Résultats et discussion

Approche symbolique

Évaluer la créativité est un sujet de recherche ouvert. Il n'existe pas de mesures de créativité largement acceptées et la plupart des évaluations restent subjectives. Dans cet article, nous voulions avoir une estimation approximative de la fréquence à laquelle les animaux que nous avons proposés existaient effectivement dans le micro-monde de la base de données Zoo. Nous avons donc choisi de mesurer la Précision de nos résultats : le nombre de prédictions pour lesquelles au moins un animal existe dans la base de données Zoo sur l'ensemble de nos prédictions. Les résultats pour les vecteurs booléens sont présentés dans le Tableau 7.

Comme nous pouvons le voir lorsque nous utilisons la liste complète des 16 caractéristiques, 22,64% des animaux proposés existent déjà dans la base de données Zoo, tandis que pour les 5 caractéristiques les plus importantes, ce pourcentage est de 59,67%. Tenant compte de la subjectivité dans l'évaluation de la créativité, examinons quelques cas spécifiques. Prenons par exemple les trois animaux suivants : *serpent de mer, grenouille, tamanoir*. Leurs vecteurs pour le sous-ensemble des caractéristiques importantes sont les suivants : $\vec{a} = (0, 0, 0, 1, 0), \vec{b} = (0, 1, 0, 1, 0), \vec{c} = (1, 0, 1, 0, 0)$ respectivement. Lorsque nous appliquons les PA sur ces vecteurs, nous obtenons le vecteur $\vec{d} = (1, 1, 1, 0, 0)$, prédisant ainsi un animal poilu qui pond des œufs et allaite ses petits. Bien que cela puisse sembler étrange à première vue, un tel animal existe effectivement : l'*ornithorynque*. Considérons maintenant le triplet d'animaux (*ornithorynque, antilope, raie*) avec les vecteurs correspondants : $\vec{a} = (1, 1, 1, 0, 0), \vec{b} = (1, 0, 1, 0, 0), \vec{c} = (0, 1, 0, 1, 0)$. En appliquant la même procédure, nous obtenons $\vec{d} = (0, 0, 0, 1, 0)$, ce qui correspond à un être non poilu venimeux qui ne pond pas d'œufs (lors de la naissance) et n'allaite pas ses petits. De tels animaux existent effectivement : les *scorpions* et les *serpents de mer*. Notre approche les identifie effectivement tous les deux.

Expériences avec GloVe

En ce qui concerne GloVe, le coût computationnel était prohibitif pour calculer $\binom{101}{3}$ instances. Puisque notre objectif dans cet article est d'explorer le potentiel des analogies pour la créativité, nous voulions comparer les résultats entre une approche basée sur la logique prédicative et une basée sur les plongements de mots. Nous avons donc décidé d'évaluer GloVe sur les résultats obtenus avec l'approche basée sur la logique prédicative et voir si GloVe pouvait les égaler. Après avoir calculé $\vec{d} = \vec{c} + \vec{b} - \vec{a}$, nous recherchons les 10 vecteurs GloVe les plus proches de $\vec{d}$ selon la distance euclidienne. Nous éliminons les instances qui ne correspondent pas à des animaux selon les "synsets" de WordNet [31], tel que mis en œuvre par NLTK. Ensuite, nous calculons la précision à k, c'est-à-dire que pour les k premières propositions de GloVe, nous examinons si au moins une est présente comme animal dans la base de données Zoo, après avoir appliqué le "stemming" avec NLTK. Les résultats pour $k \in [1, 8]$ sont présentés dans la Table 8. Comme nous pouvons le voir, déjà pour $k = 2$, GloVe atteint une précision de 64,60, dépassant les résultats obtenus dans la méthode basée sur la logique prédicative, tandis que pour $P@8 = 83,44$.

Expériences avec BERT

Concernant les plongements contextuels utilisant les architectures Transformer, comme mentionné précédemment, nous avons utilisé Sentence-BERT [45] après avoir créé des phrases pour chaque animal reflétant leurs caractéristiques. Bien que nous puissions calculer $\vec{d} = \vec{c} + \vec{b} - \vec{a}$ et ensuite rechercher une phrase qui est plus proche de $\vec{d}$, le faire pour chaque phrase potentielle est computationnellement prohibitif. Nous avons donc opté pour l'identification de $\vec{d}$ pour les vecteurs représentant les animaux restreints à la base de données Zoo, en calculant les distances euclidiennes entre $\vec{a} - \vec{b}$ et $\vec{c} - \vec{d}$ pour tout $d \notin \{a, b, c\}$. Nous voulions ensuite évaluer les résultats de cette approche par rapport aux résultats de l'approche symbolique décrite ci-dessus. Plus précisément, pour tout triplet pour lequel l'approche symbolique fournissait une liste d'animaux non vide, nous avons calculé la $Prcision@k$ entre cette liste et les $k = 8$ premiers animaux (représentés par le vecteur $\vec{d}$) calculés avec Sentence-BERT. Les résultats sont présentés dans la Table 8.

Comme nous pouvons le voir, il y a un certain chevauchement entre l'approche basée sur BERT et l'approche symbolique, mais nous ne pouvons pas prétendre qu'il y ait un chevauchement très étroit. Cela dit, notons cependant que, comme dans le cas symbolique, dans le cas de BERT, des animaux tels que l'*ornithorynque* sont également "prédits". Par exemple, pour le triplet *(scorpion, vipère, tamanoir)*, l'approche symbolique fournit l'*ornithorynque* comme seule prédiction, tandis que l'approche basée sur BERT fournit une fois de plus l'*ornithorynque* comme l'animal "analogique" le plus probable. Il en va de même pour le triplet *(scorpion, tamanoir, physalie)*. En ce qui concerne le *scorpion*, celui-ci est donné comme l'animal "analogique" le plus probable par différents triplets par BERT, par exemple *(oryx, autruche, poney)*, ainsi que par l'approche symbolique, par exemple *(ornithorynque, opossum, physalie)*, bien qu'il n'y ait aucun triplet qui fournisse le *scorpion* à la fois pour l'approche symbolique et pour celle basée sur BERT.

	P@1	P@2	P@3	P@4	P@5	P@6	P@7	P@8
GloVe	46.33	64.60	73.94	78.75	81.33	82.60	83.21	83.44
exp.	6.87	12.48	16.65	20.66	24.04	27.17	30.15	32.89
not exp.	0.0	0.0	0.0	0.0	0.0	0.2	1.4	2.6
om.	0.0	0.01	0.08	0.12	0.19	0.2	0.4	0.6

TABLE 8 – Précision@k pour les vecteurs GloVe et sBERT en pourcentages. Pour sBERT, les résultats reflètent le pourcentage d'animaux prédits qui ont également été prédits par l'approche symbolique. Les descriptions d'animaux mentionnent explicitement l'animal (exp.), ou non (not exp.), ou ne mentionnent pas explicitement l'animal et omettent des caractéristiques dont l'animal est dépourvu (om.).

Une deuxième approche consistait à comparer directement l'approche symbolique par rapport à une approche basée sur BERT. Nous avons donc créé, comme décrit dans la section précédente, des descriptions complètes ou des descriptions omettant les caractéristiques absentes des animaux sans fournir le nom des animaux. Cela nous a permis de générer des vecteurs sBERT et de calculer $\vec{d} = \vec{c} + \vec{b} - \vec{a}$. Nous avons ensuite procédé à la création de descriptions similaires pour toutes les combinaisons possibles des 5 caractéristiques sélectionnées, donnant ainsi $2^5 = 32$ descriptions et donc 32 vecteurs BERT $\vec{d}_i, i \in [1, \ldots, 32]$. Nous avons ensuite ordonné dans un ordre décroissant chaque $\vec{d}_i$ par rapport à $\vec{d}$. Comme nous avons pu récupérer le sous-ensemble des animaux qui satisfont chaque description, nous avons ainsi pu comparer directement à l'approche symbolique. Nous avons décidé d'utiliser la $Prcision@k$ comme précédemment afin de mesurer le chevauchement entre l'approche symbolique et celle basée sur BERT. Comme le montrent les résultats de la Table 8, BERT s'appuie fortement sur la sémantique du mot lui-même pour décrire l'animal au lieu de la description de ses caractéristiques.

Un dernier exemple de créativité avec les PA

Les tests de Raven [44] sont des tests de QI bien connus. Ils se présentent sous la forme d'une série d'instances ayant le format d'une matrice $n \times n$ (où n est 2, 3 ou 4) dont les cellules contiennent diverses figures géométriques (voir

Figure 2 pour un exemple), à l'exception de la dernière cellule qui est vide et doit être complétée en sélectionnant une solution parmi 8 candidats (lorsque $n = 3$). Les séries de matrices d'un test sont de difficulté progressive (d'où la dénomination de Raven's Progressive Matrices, abrégé en RPM). Lorsqu'aucune solution n'est proposée, nous sommes confrontés à un exercice de créativité pour construire le contenu de la cellule vide.

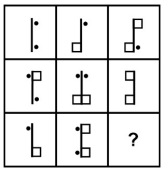


FIGURE 2 – Le problème

Décrivons notre ensemble d'images, où $im9$ doit être deviné, comme une matrice telle que :

$$\begin{pmatrix} im1 & im2 & im3 \\ im4 & im5 & im6 \\ im7 & im8 & im9 \end{pmatrix}$$

Chaque image i peut être codée par un vecteur avec 4 composantes :

- tl décrit l'élément en haut à gauche de la barre verticale ;
- tr décrit l'élément en haut à droite de la barre ;
- br décrit l'élément en bas à droite de la barre ;
- bl décrit l'élément en bas à gauche de la barre.

La valeur de chaque composante appartient à l'ensemble $X = \{point, carr, rien\} = \{p, c, r\}$ de sorte qu'une image est un vecteur dans X^4.

Cela conduit à une nouvelle représentation de la matrice initiale :

$$\begin{pmatrix} nddn & ndns & nsds \\ dsdn & ddss & snns \\ dnsn & dssd & im9 \end{pmatrix}$$

En termes de proportion analogique, le problème peut être lu comme les PA suivantes : $(im1, im2) : im3 :: (im4, im5) : im6$ et $(im4, im5) : im6 :: (im7, im8) : im9$ ou avec la représentation vectorielle : $(nddn, ndns) : nsds :: (dsdn, ddss) : snns$ et $(dsdn, ddss) : snns :: (dnsn, dssd) : im9$, où $im9$ est alors la solution à trouver. En d'autres termes, la relation R entre les 2 premières images et la 3e dans une ligne est la même quelle que soit la ligne. Cette hypothèse est également valable pour les colonnes : la relation R entre les 2 premières images et la 3e dans une colonne est la même quelle que soit la colonne. La manière naturelle de considérer une relation entre 2 images et une troisième est de supposer que la troisième est une combinaison T des deux premières, où T est un opérateur $X \times X \to X$, défini composante par composante. T doit être défini sur 9 paires. Observer les deux premières lignes complètes, ainsi que les deux premières colonnes complètes de la matrice, nous donne une définition complète de l'opérateur T : Pour chaque ligne ou colonne, nous obtenons 4 équations :

ligne 1 $T(n,n) = n, T(d,d) = s, T(d,n) = d, T(n,s) = s$;
ligne 2 $T(d,d) = s, T(s,d) = n, T(d,s) = n, T(n,s) = s$;
col. 1 $T(n,d) = d, T(d,s) = n, T(d,d) = s, T(n,n) = n$;
col. 2 $T(n,d) = d, T(d,d) = s, T(n,s) = s, T(s,s) = d$.

Cela définit entièrement T, et nous obtenons $im9 = ssdd$ puisque $T(d,d) = s, T(n,s) = s, T(s,s) = d$, et $T(n,d) = d$, conduisant ainsi à la solution unique ci-contre :

Cet exemple montre les capacités créatives de l'inférence basée sur l'AP, à partir de l'utilisation d'une représentation basée sur des caractéristiques (symboliques). Remarquablement, cette approche s'est avérée efficace sur une série de 36 tests RPM avancés [7, 8] ; dans 16 cas, l'approche donne un bon résultat même lorsqu'elle est appliquée au niveau granulaire du pixel.

Cet exemple illustre la puissance créative de l'inférence basée sur les proportions analogiques (PA). Remarquons que les PA sont ici de la forme $(x, y) : \mathcal{T}(x, y) :: (x', y') : \mathcal{T}(x', y')$ où $\mathcal{T}$ s'applique aux paires de vecteurs et renvoie un vecteur. De tels PA sont proches des PA de la forme $x : f(x) :: x' : f(x')$, étudiés en détail par [18], qui peuvent être liés aux PA booléennes [1]. Dans cet exemple, $\mathcal{T}$ est entièrement défini à travers les huit premières cellules de la matrice. Mais des exemples similaires où $\mathcal{T}$ est seulement partiellement défini conduiraient à plusieurs solutions. Cette forme de créativité est une "génération nouvelle adaptée aux contraintes d'une tâche particulière" [30], comme c'est généralement le cas avec le raisonnement analogique [17].

Les tests RPM sont un exemple de la puissance de l'inférence basée sur les proportions analogiques. Cette approche s'écarte clairement de celles basées sur la théorie de la correspondance des structures ("structure mapping theory") [14, 13], ou de ses variantes [24], y compris l'algorithme d'inférence CWSG ("copy with substitution and generation") [19], qui ont également été appliquées aux problèmes de RPM, voir par exemple [29], ou [28, 48]. Mais ces approches nécessitent de disposer de solutions potentielles, ce qui n'est pas très conforme à l'idée de créativité. Voir [8] pour une discussion comparative.

Remarques de conclusion

Cet article présente une étude préliminaire sur le pouvoir créatif des PA, montrant des résultats encourageants. Dans son travail pionnier, Margaret Boden [2] distingue trois formes de créativité : la créativité combinatoire, exploratoire et transformationnelle. La créativité combinatoire est le résultat de la combinaison d'idées familières. Les deux autres types de créativité, selon Boden, présupposent l'existence d'un espace conceptuel à travers lequel de nouvelles idées sont réalisées. Ainsi, dans le deuxième type de créativité, l'agent créatif explore les différentes parties de l'espace conceptuel à la recherche d'instances qui peuvent être

considérées comme créatives. Le dernier type de créativité présuppose que l'agent "élargit", pour ainsi dire, les limites de l'espace conceptuel, permettant de nouvelles dimensions et donc une compréhension plus profonde du monde et donc la proposition de nouvelles idées créatives.

Notre travail va au-delà de la simple combinaison de vecteurs, nous proposons un mécanisme basé sur les PA, pour changer un vecteur $\vec{c}$ en un vecteur $\vec{d}$, une fois que le *contexte approprié* d'une autre paire $(\vec{a}, \vec{b})$ est trouvé dans l'environnement, souscrivant ainsi à la définition de la créativité *exploratoire*, grâce aux nombreuses paires $(\vec{a}, \vec{b})$ et aux pivots $\vec{c}$ qui peuvent être utilisés en général.

Nous concluons donc que le processus créatif que nous présentons dans cet article est du deuxième type. En d'autres termes, nous explorons l'espace conceptuel dans le micromonde de la base de données Zoo et proposons des instances nouvelles qui n'ont jamais existé auparavant. Les résultats montrent que la méthode proposée est capable d'identifier l'existence d'animaux dont on pourrait ne pas penser qu'ils existent, tels que des animaux qui pondent des œufs et allaitent leurs petits (ornithorynque) ou ceux qui donnent naissance à leurs petits par ovoviviparité mais ne les allaitent pas (scorpions).

Par ailleurs, nous voulions examiner comment cette approche se comporte pour les plongements lexicaux obtenus par différents modèles de langage. Nous avons donc examiné à la fois les plongements lexicaux statiques, en utilisant GloVe, et les plongements lexicaux contextuels en utilisant BERT. En ce qui concerne les résultats de GloVe, nous avons constaté que notre approche est capable de proposer des animaux "nouveaux" avec jusqu'à 83,44% (pour $P@8$) des animaux proposés dans la base de données originale, alors que même les résultats pour $P@2$ dépassent l'approche booléenne. Ces résultats ne sont à prendre que comme une indication du potentiel de cette approche. Comme montré dans [15], GloVe fonctionne vraiment bien lorsqu'il s'agit d'ensembles de données d'analogies simples, comme celui proposé par [34], mais seuls 30% des analogies sont capturées dans leur nouveau corpus BATS. Des preuves récentes [51] montrent que des modèles plus avancés basés sur les technologies des transformateurs sont capables d'identifier des analogies d'une manière comparable à celle des humains. Nous avons donc choisi d'examiner comment les plongements lexicaux contextuels basés sur les architectures de Transformateur, tels que BERT, se comparent à notre approche. Nous avons donc fourni des descriptions en langage naturel des animaux et calculé des plongements lexicaux de phrases pour chacun en utilisant S-BERT. Nous avons réalisé deux séries d'expériences, l'une contenant le nom de l'animal et l'autre ne contenant pas le nom de l'animal. Bien que cette approche puisse en effet récupérer des animaux tels que les ornithorynques et les scorpions (lorsque les noms des animaux sont fournis), ils ont très peu de chevauchement avec l'approche symbolique. Il reste beaucoup à faire pour comprendre la créativité en utilisant les PA pour la génération de nouvelles instances créatives. Tout d'abord, nous avons besoin de meilleures mesures pour évaluer la créativité qui ne prennent pas uniquement en compte l'existence ou non des instances proposées, [4] mais aussi d'une certaine manière mesurent à quel point les instances proposées sont "intéressantes". À l'avenir, nous prévoyons de travailler sur de telles mesures ainsi que d'explorer d'autres approches plus sophistiquées des grands modèles de langage tels que GPT-3 et GPT-4. Des expériences initiales concernant l'évaluation de la partie "créative" des animaux proposés pourraient être réalisées en se basant sur la fréquence de la combinaison de leurs caractéristiques. Plus précisément, plus la combinaison de leurs caractéristiques est rare, tout en étant un animal "valide", plus on peut argumenter que la proposition de l'animal est créative.

De manière générale, le raisonnement génératif par le biais des PA ne garantit pas la valeur pratique des résultats, ni leur caractère imprévu. Peut-être peut-on s'inspirer des mesures utilisées dans les systèmes de recommandation [26, 22], pour évaluer l'imprévu dans la créativité. Cependant, le raisonnement basé sur les PA ne reste pas dans le voisinage de ce qui est connu, comme le montrent les exemples de cet article, et peut produire des résultats innovants. Lorsqu'il s'agit de représentations booléennes ou nominales, les résultats peuvent être expliqués, et comme le montre l'exemple de Raven, il n'est pas nécessaire de choisir entre des résultats candidats.

Certes la notion de créativité est délicate à cerner. On peut toutefois convenir que créer, c'est avant tout imaginer et concevoir quelque chose de nouveau, quelque chose qui n'a pas encore été vu ou réalisé. Ainsi, un enfant ayant observé une vache avec son veau, voyant ensuite une jument, pourrait imaginer à quoi ressemble un poulain, même s'il n'en a jamais vu auparavant. De façon similaire, un mathématicien peut concevoir qu'un certain résultat pourrait être valide, car le problème qu'il cherche à résoudre présente des similitudes avec un problème dans un domaine différent où un résultat similaire est déjà établi. Si l'idée du nouveau résultat émerge de façon analogique, sa validation formelle se fera à travers une démarche déductive. La déduction permet de révéler des faits jusque-là ignorés, mais elle ne constitue pas en soi un acte de créativité au sens évoqué précédemment. D'une manière générale, l'espace des faits formellement déductibles d'un ensemble de connaissance est très grand, voire infini. Un outil pour "traverser, parcourir" l'espace des combinaisons possibles est nécessaire. Cet outil peut être de nature probabiliste, car un changement aléatoire d'un des paramètres d'un problème peut relancer la recherche dans de nouvelles directions. Les

4. Dans l'expérience avec l'ensemble de données Zoo, nous avons bénéficié de la connaissance des animaux existants, mais de telles informations sont rarement disponibles pour juger de la créativité.

modèles de diffusion [53], essentiellement basés sur la théorie de probabilités, sont une excellente illustration de cette approche. Mais l'outil peut être aussi de nature analogique, car le mécanisme de la proportion analogique peut parfois être vu comme un processus de recopie partielle [7]. La création, au lieu d'être le résultat du hasard, peut aussi être une affaire de recombinaison d'idées/concepts ou d'attributs préalablement connus : peut-on imaginer un centaure si on ne sait pas ce qu'est un cheval et un homme [9] ? L'inférence à base de PA peut donner l'illusion d'être déductive, d n'est-il pas le résultat du calcul $c+b-a$? Mais pour faire ce calcul il faut partir d'un c et apprécier la différence entre un b et un a qu'il faut avoir choisis, et des b et des a il y en en général beaucoup qui pour un même c ne conduiront pas toujours au même d !

Remerciements

Cette recherche a été soutenue par le projet ANR "Analogies : from Theory to Tools and Applications" (AT2TA), ANR-22-CE23-0023.

Références

[1] N. Barbot, L. Miclet, H. Prade, and G. Richard. A new perspective on analogical proportions. In *ECSQARU, LNCS, 11726, 163-174*. Springer, 2019.

[2] M. Boden. *The Creative Mind : Myths and Mechanisms, 2nd edition*. Routledge, 2004.

[3] M. Bounhas, M. Pirlot, H. Prade, and O. Sobrie. Comparison of analogy-based methods for predicting preferences. In *SUM, LNCS, 11940, 339-354*. Springer, 2019.

[4] M. Bounhas and H. Prade. Analogy-based classifiers : An improved algorithm exploiting competent data pairs. *Int. J. Approx. Reason.*, 158 :108923, 2023.

[5] M. Bounhas, H. Prade, and G. Richard. Analogy-based classifiers for nominal or numerical data. *Int. J. of Approximate Reasoning*, 91 :36 – 55, 2017.

[6] S. Colton. Experiments in constraint-based automated scene generation. In P. Gervás, R. Pérez y Pérez, and T. Veale, editors, *Proc. Int. Conf. on Computational Creativity, Madrid*, pages 127–136, 2008.

[7] W. F. Correa, H. Prade, and G. Richard. When intelligence is just a matter of copying. In *ECAI*, Frontiers in Artificial Intelligence and Applications, 242, pages 276–281. IOS Press, 2012.

[8] W. Correa Beltran, H. Prade, and G. Richard. Constructive solving of Raven's IQ tests with analogical proportions. *Int. J. Intell. Syst.*, 31(11) :1072–1103, 2016.

[9] R. Descartes. *Méditations Métaphysiques*. 1641.

[10] J. Devlin, Ming-Wei Chang, K. Lee, and K. Toutanova. BERT : Pre-training of deep bidirectional transformers for language understanding. In *Proc. 2019 Conf. of the North Amer. Chapter of the Assoc. for Comput. Ling. : Human Language Technologies, Vol.1 (Long and Short Papers)*, 4171-4186, Minneapolis, 2019.

[11] T. G. Evans. A program for the solution of a class of geometric-analogy intelligence-test questions. In M. L. Minsky, editor, *Semantic Information Processing*, pages 271–353. MIT Press, Cambridge, 1968.

[12] M. A. Fahandar and E. Hüllermeier. Learning to rank based on analogical reasoning. In *AAAI*, pages 2951–2958, 2018.

[13] B. Falkenhainer, K. D. Forbus, and D. Gentner. The structure-mapping engine : Algorithm and examples. *Artif. Intell.*, 41(1) :1–63, 1989.

[14] D. Gentner. Structure-mapping : A theoretical framework for analogy. *Cognit. Sci.*, 7(2) :155–170, 1983.

[15] A. Gladkova, A. Drozd, and S. Matsuoka. Analogy-based detection of morphological and semantic relations with word embeddings : what works and what doesn't. In *Proc. NAACL Student Res. Workshop*, San Diego, 8-15, 2016. Assoc. for Comput. Ling.

[16] A. K. Goel. Design, analogy and creativity. *IEEE Expert*, 12 :62–70, 1997.

[17] A.E. Green, D.J. Kraemer, J.A. Fugelsang, J.R. Gray, and K.N. Dunbar. Neural correlates of creativity in analogical reasoning. *J. Experim. Psychology : Learning, Memory, and Cognition*, 38 :264–272, 2012.

[18] D. R. Hofstadter and Fluid Analogies Research Group. *Fluid Concepts and Creative Analogies : Computer Models of the Fundamental Mechanisms of Thought*. Basic Books, New York, 1995.

[19] K. J. Holyoak. Analogy. In K. J. Holyoak and R. G. Morrison, editors, *The Cambridge Handbook of Thinking and Reasoning*, pages 117–142. Cambridge Univ. Press, 2005. Chap. 6.

[20] K. J. Holyoak and P. Thagard. *Mental Leaps : Analogy in Creative Thought*. MIT Press, 1995.

[21] Xiaoyang Hu, S. Storks, R. Lewis, and J. Chai. In-context analogical reasoning with pre-trained language models. In *Proc. 61st Annual Meeting of the Association for Computational Linguistics (Volume 1 : Long Papers)*, Toronto, July 2023.

[22] M. Kaminskas and D. Bridge. Diversity, serendipity, novelty, and coverage : A survey and empirical analysis of beyond-accuracy objectives in recommender systems. *ACM Trans. Interact. Intell. Syst.*, 7(1), 2016.

[23] A. Kaufmann. L'imagination artificielle - (heuristique automatique). *Revue française d'automatique, d'informatique et de recherche opérationnelle (R.A.I.R.O.) - Recherche opérationnelle*, 3(3) : 5–24, 1969.

[24] M. T. Keane, T. Ledgeway, and S. Duff. Constraints on analogical mapping : A comparison of three models. *Cognitive Science*, 18(3) :387–438, 1994.

[25] S. Klein. Culture, mysticism & social structure and the calculation of behavior. In *Proc. 5th Europ. Conf. in Artif. Intel. (ECAI'82), Orsay*, 141-146, 1982.

[26] D. Kotkov, Shuaiqiang Wang, and J. Veijalainen. A survey of serendipity in recommender systems. *Knowledge-Based Systems*, 111 :180–192, 2016.

[27] S. Kullback and R. Leibler. On information and sufficiency. *Annals Math. Statistics*, 22 :79–86, 1951.

[28] M. Kunda, K. McGreggor, and A. K. Goel. A computational model for solving problems from the Raven's Progressive Matrices intelligence test using iconic visual representations. *Cognitive Systems Research*, 22 :47–66, 2013.

[29] A. Lovett, K. Forbus, and J. Usher. A structure-mapping model of Raven's progressive matrices. In *Proc. 32nd Annual Conf. of the Cognitive Science Soc., Portland, OR*, 2010.

[30] R. E. Mayer. Fifty years of creativity research. In R.J. Sternberg, editor, *Handbook of Creativity*, pages 449–460. Cambridge Univ. Press, 1998.

[31] J. P. McCrae, A. Rademaker, F. Bond, E. Rudnicka, and Ch. Fellbaum. English WordNet 2019 – an open-source WordNet for English. In *Proc. 10th Global Wordnet Conference*, pages 245–252, Wroclaw, 2019.

[32] L. Miclet, S. Bayoudh, and A. Delhay. Analogical dissimilarity : definition, algorithms and two experiments in machine learning. *JAIR, 32*, 793-824, 2008.

[33] L. Miclet and H. Prade. Handling analogical proportions in classical logic and fuzzy logics settings. In *ECSQARU*, LNCS, 5590, 638-650. Springer, 2009.

[34] T. Mikolov, I. Sutskever, K. Chen, G. S Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges et al., editor, *Advances in Neural Inform. Processing Syst. 26,*. 3111-3119, Curran Assoc. Inc., 2013.

[35] J. Pennington, R. Socher, and Ch. Manning. GloVe : Global vectors for word representation. In *Proc. 2014 Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, 2014.

[36] V. Pirrelli and F. Yvon. Analogy in the lexicon : a probe into analogy-based machine learning of language. In *Proc. 6th Int. Symp. on Human Communic.*, 1999. Santiago de Cuba, 6 p.

[37] H. Prade and G. Richard. Analogical proportions : From equality to inequality. *Int. J. of Approximate Reasoning*, 101 :234 – 254, 2018.

[38] H. Prade and G. Richard. Analogical proportions : Why they are useful in AI. In *IJCAI*, pages 4568–4576, 2021.

[39] H. Prade and G. Richard. Multiple analogical proportions. *AI Commun.*, 34(3) :211–228, 2021.

[40] H. Prade and G. Richard. First steps towards a logic of ordered pairs. In *ECSQARU*, LNCS, 14294, 198-209. Springer, 2023.

[41] H. Prade and G. Richard. Premiers pas vers une logique des paires ordonnées. In *JIAF*, pages 104–112, 2023.

[42] H. Prade and G. Richard. Analogical proportion-based induction : From classification to creativity. *J. of Applied Logics — IfCoLog J. of Logics and their Applications*, 11 :51–87, 2024.

[43] M. Ragni and S. Neubert. Analyzing Raven's intelligence test : Cognitive model, demand, and complexity. In *Computational Approaches to Analogical Reasoning*, Studies in Computational Intelligence, 548, pages 351–370. Springer, 2014.

[44] J. C. Raven. *Progressive Matrices*. The Psychological Corporation, New York, 1965.

[45] N. Reimers and I. Gurevych. Sentence-bert : Sentence embeddings using siamese bert-networks, 2019. arXiv, cs.CL, 1908.10084.

[46] D. E. Rumelhart and A. A. Abrahamson. A model for analogical reasoning. *Cognit. Psycho.*, 5 :1–28, 1973.

[47] J. Schmidhuber. Formal theory of creativity, fun, and intrinsic motivation (1990-2010). *IEEE Trans. on Autonomous Mental Development*, 2(3) :230–247, 2010.

[48] S. Shegheva. A computational model for solving Raven's Progressive Matrices intelligence test, 2018. Master of Science dissertation, School of Computer Science, Georgia Institute of Technology, Aug., 45 p.

[49] A. Vaswani, N. Shazeer, N. Parmar, J.Uszkoreit, Llion Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need, 2017. arXiv, cs.CL, 1706.03762.

[50] T. Veale. An analogy-oriented type hierarchy for linguistic creativity. *Knowledge-Based Systems*, 19(7) :471 – 479, 2006.

[51] T. Webb, K. J. Holyoak, and Hongjing Lu. Emergent analogical reasoning in large language models. *Nature Human Behaviour*, 7(9) :1526–1541, 2023.

[52] T. Wijesiriwardene, R. Wickramarachchi, B. Gajera, S. Gowaikar, C. Gupta, A. Chadha, A. N. Reganti, A. Sheth, and A. Das. ANALOGICAL - a novel benchmark for long text analogy evaluation in large language models. In *Findings of the Association for Computational Linguistics : ACL 2023*, Toronto, 2023.

[53] L. Yang, Z. Zhang, Y. Song, S. Hong, R. Xu, Y. Zhao, W. Zhang, B. Cui, and M.-H. Yang. Diffusion models : A comprehensive survey of methods and applications. *ACM Computing Surveys*, 56(4), 2023.

Session 7 : Planification hiérarchique ou temporelle

Actes JIAF 2024

Apprentissage de domaines HDDL à partir d'observations partielles et bruitées

Maxence Grand Damien Pellier Humbert Fiorino

Univ. Grenoble Alpes, LIG, 38000 Grenoble, France
maxence.grand@univ-grenoble-alpes.fr
damien.pellier@univ-grenoble-alpes.fr
humbert.fiorino@univ-grenoble-alpes.fr

Résumé

La planification HTN (Hierarchical Task Network) est largement utilisée pour décrire des domaines de planification. Contrairement à la planification STRIPS classique, qui ne requiert que la déclaration des actions, la planification HTN nécessite la déclaration des tâches et leur décomposition en sous-tâches, appelées méthodes. Par conséquent, l'encodage manuel des domaines HTN est considéré comme plus difficile, en particulier pour les experts applicatifs qui ne sont pas familiarisés avec les langages de planification. Pour relever ce défi, nous présentons une nouvelle approche appelée HierAMLSI, qui s'appuie sur l'induction grammaticale pour apprendre des domaines de planification HTN. Contrairement à d'autres méthodes d'apprentissage, HierAMLSI apprend, avec un haut degré de précision, les actions et les méthodes à partir de traces bruitées et partiellement observées.

Abstract

The Hierarchical Task Network (HTN) formalism is widely used for expressing planning domains. Unlike the classical STRIPS formalism, which only requires specifying actions, HTN demands the specification of tasks and their decompositions into subtasks, known as methods. Consequently, manually encoding HTN domains is considered more challenging and error-prone, particularly for application experts unfamiliar with planning languages. To address this challenge, we introduce a novel approach called HierAMLSI, which relies on grammar induction to learn HTN planning domain knowledge. Unlike other learning methods, HierAMLSI can learn with a high degree of accuracy both actions and methods from noisy and partially observed input data.

1 Introduction

La planification automatique repose sur des "modèles d'actions", c'est-à-dire des domaines de planification exprimés dans un langage déclaratif tel que PDDL (Planning Domain Description Language) [28] ou son équivalent hiérarchique HDDL (Hierarchical Domain Description Language) [18]. Contrairement à la planification classique, les modèles d'actions hiérarchiques intègrent des éléments supplémentaires pour représenter des "recettes" permettant de résoudre des "tâches". Par exemple, pour accomplir la tâche "préparer un repas", il peut être nécessaire de réaliser des sous-tâches telles que "préparer les entrées", "préparer le plat principal" et "préparer les desserts". En décomposant davantage la tâche "préparer les entrées", on peut avoir des sous-tâches telles que "préparer des cornichons", etc. Diverses "méthodes" peuvent être employées pour accomplir une tâche donnée, comme "préparer un repas végétalien" ou "utiliser des ingrédients sans gluten".

Cependant, cette expressivité s'accompagne de quelques inconvénients. L'encodage manuel de modèles d'actions hiérarchiques est souvent considéré comme une tâche difficile, fastidieuse et sujette aux erreurs, en particulier pour les experts applicatifs qui ne sont pas familiarisés avec les langages de planification. Pour relever ce défi, diverses techniques d'apprentissage automatique ont été proposées pour apprendre des modèles d'actions, qu'ils soient hiérarchiques ou non, à partir d'observations, c'est-à-dire des traces, des séquences de changements d'état du monde. Par exemple, certaines approches comme ARMS [38], LSONIO [29] et LOCM [3] se concentrent uniquement sur l'apprentissage de modèles d'actions non hiérarchiques. En revanche, d'autres, comme CAMEL [23], HTN-Maker [14, 12] et LHTNDT [30], fournissent des actions simples à l'algorithme d'apprentissage dans le but d'apprendre des

hiérarchies de tâches représentées sous la forme de méthodes. D'autres, comme HTN-Learner [39, 40], sont capables d'apprendre à la fois des modèles d'actions simples et des méthodes.

Malgré les progrès récents, l'apprentissage de modèles d'actions hiérarchiques présente encore plusieurs problèmes difficiles à résoudre. Tout d'abord, il nécessite une grande quantité de données d'apprentissage dont l'obtention peut s'avérer difficile et coûteuse dans le cadre d'applications réelles. De plus, les modèles appris ne sont souvent pas suffisamment précis pour être directement utilisés par des planificateurs. Une relecture par un expert humain est souvent nécessaire pour corriger les erreurs syntaxiques et garantir que le modèle appris fonctionne correctement. Enfin, peu d'algorithmes sont capables d'apprendre des modèles d'actions à partir d'observations partielles et bruitées.

Dans cet article, nous présentons HierAMLSI, un algorithme d'apprentissage spécialement conçu pour acquérir des domaines HDDL. HierAMLSI fait partie des rares algorithmes capables d'apprendre à la fois les actions et les méthodes, y compris leurs préconditions. Aussi, HierAMLSI apprend les domaines HDDL à partir de traces d'exécutions bruitées et partiellement observées.

Le reste de l'article est organisé comme suit. Dans la section 2, nous présentons le cadre formel. Dans la section 3, nous donnons une description de l'algorithme AMLSI [7, 8] sur lequel HierAMLSI est basé. La section 4 détaille l'algorithme HierAMLSI. Ensuite, la section 5 évalue les performances de HierAMLSI sur des benchmarks IPC [1]. Grâce à l'évaluation expérimentale, nous montrerons que HierAMLSI relève efficacement trois grands défis mis en évidence dans la littérature existante : (1) traiter des ensembles de données partielles et bruitées, (2) minimiser le volume de données requis pour l'apprentissage, et (3) obtenir des domaines très précis, réduisant ainsi la nécessité d'une relecture et d'une correction approfondies par des experts humains. Enfin, la section 6 présente l'état de l'art.

2 Cadre Formel

2.1 Planification STRIPS

Dans cette section, nous utilisons les définitions et les notations proposées par [17] et les adaptons au problème de l'apprentissage.

Un problème de planification STRIPS est un n-uplet $P = (L, A, S, s_0, g, \delta, \tau, \lambda)$, où L est un ensemble de propositions logiques décrivant les états du monde, S est un ensemble d'états, $s_0 \in S$ est l'état initial, et g est le but. La fonction $\lambda : S \to 2^L$ est une fonction d'observation qui attribue à chaque état l'ensemble des propositions logiques vraies dans cet état. A est un ensemble d'actions. Les préconditions, les effets positifs et négatifs des actions sont donnés par les fonctions $prec$, add et del incluses dans $\delta = (prec, add, del)$. La fonction $prec$ est définie comme $prec : A \to 2^L$. Les fonctions add et del sont définies de la même manière.

La fonction booléenne $\tau : A \times S \to \{\text{true}, \text{false}\}$ retourne vrai si une action est applicable dans un état, c'est-à-dire $\tau(a, s) \Leftrightarrow \text{prec}(a) \subseteq \lambda(s)$. Chaque fois que l'action a est applicable dans l'état s_i, la fonction de transition d'état $\gamma : S \times A \to S$ retourne l'état résultant $s_{i+1} = \gamma(s_i, a)$ tel que $\lambda(s_{i+1}) = [\lambda(s_i) \setminus \text{del}(a)] \cup \text{add}(a)$.

Une séquence d'actions $(a_0 a_1 \dots a_n)$ est applicable dans un état s_0 lorsque chaque action a_i, avec $0 \leq i \leq n$, est applicable dans l'état s_i. Étant donné une séquence applicable $(a_0 a_1 \dots a_n)$ dans l'état s_0, $\gamma(s_0, (a_0 a_1 \dots a_n)) = \gamma(\gamma(s_0, a_0), (a_1 \dots a_n)) = s_{n+1}$. Il est important de noter que cette définition récursive de γ entraîne la génération d'une séquence d'états $(s_0 s_1 \dots s_{n+1})$. Un état s est un état but si $\lambda(g) \subseteq \lambda(s)$. L'état s satisfait g, c'est-à-dire $s \models g$, si et seulement si s est un état but. Une séquence d'actions est un plan solution pour un problème de planification P si et seulement si elle est applicable dans s_0 et entraîne un état but.

En langage formel, un ensemble de règles décrit la structure des mots valides, et le langage est l'ensemble de ces mots. Pour un problème de planification $P = (L, A, S, s_0, g, \delta, \tau, \lambda)$, ce langage est défini comme suit :

$$\mathcal{L}(P) = \{\pi = (a_0 a_1 \dots a_n) \mid a_i \in A, \gamma(s_0, \pi) \models g\}$$

Nous savons que les problèmes de planification STRIPS génèrent un ensemble de langages réguliers [17]. En d'autres termes, un problème de planification STRIPS P engendre un langage $\mathcal{L}(P)$ équivalent à un automate fini déterministe (AFD) $\Sigma = (S, A, \gamma)$. S et A représentent respectivement les nœuds et les arcs de l'AFD, et γ est la fonction de transition.

Par conséquent, si un AFD Σ peut être appris à partir d'un ensemble d'observations $\Omega \subseteq \mathcal{L}(P)$ (voir la figure 1 pour un exemple d'AFD appris), alors il est possible d'apprendre le problème de planification STRIPS $P = (L, A, S, s_0, g, \tau, \delta, \lambda)$ à partir de ces observations. En particulier, il est possible d'apprendre la fonction δ définissant les actions de P et de la généraliser afin de l'exprimer dans un domaine PDDL.

2.2 Planification HTN

Un problème de planification HTN, selon les notations étendues de [15], est un n-uplet $P = (L, C, A, S, M, s_0, w_I, g, \delta, \tau, \lambda, \sigma, \zeta)$. Les éléments L, S, $s_0 \in S$, g, λ, et δ sont similaires à ceux du problème de planification STRIPS.

A est l'ensemble des actions (ou tâches primitives) et C est l'ensemble des tâches composées (ou non primitives),

1. **I**nternatinnal **P**lanning **C**ompetition

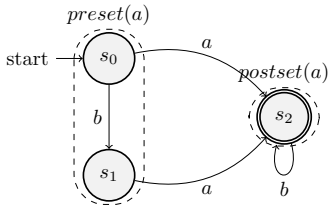


FIGURE 1 – Exemple d'AFD appris depuis un ensemble d'observations : $\Omega = \{\lambda(s_0) \xrightarrow{a} \lambda(s_2), \lambda(s_0) \xrightarrow{a} \lambda(s_2) \xrightarrow{b} \lambda(s_2), \lambda(s_0) \xrightarrow{b} \lambda(s_1) \xrightarrow{a} \lambda(s_2), \lambda(s_0) \xrightarrow{b} \lambda(s_1) \xrightarrow{a} \lambda(s_2) \xrightarrow{b} \lambda(s_2), \lambda(s_0) \xrightarrow{a} \lambda(s_2) \xrightarrow{b} \lambda(s_2) \xrightarrow{b} \lambda(s_2), \ldots$.
$preset(a)$ (resp. $postset(a)$) représente les *prédécesseur* (resp. *successeur*) de a.

avec $C \cap A = \emptyset$. Un réseau de tâches, noté ω, est une séquence de tâches [2]. Soit $T = C \cup A$, l'ensemble des tâches primitives et composées, un réseau de tâches est un élément de T^*, c'est-à-dire l'ensemble des tâches construit à partir de T. Les tâches composées sont décomposées en utilisant des méthodes. L'ensemble M contient toutes les méthodes. Les méthodes sont définies par la fonction $\sigma : M \rightarrow C \times T^*$. Ensuite, une tâche composée c est décomposable dans un état s en un réseau de tâches ω, si et seulement si il existe une méthode $m \in M$ telle que : $\sigma(m) = (c, \omega)$ et $prec(m) \subseteq s$. La fonction $\zeta : T^* \times S \rightarrow T^*$ est la fonction de décomposition. Cette fonction permet de décomposer la première tâche d'un réseau de tâches. Utilisée récursivement, elle décompose toutes les tâches d'un réseaux de tâches. Étant donné un réseau de tâches totalement ordonné $t\omega$ avec t comme première tâche, ζ est définie comme suit :

$$\zeta(t\omega, s) = \begin{cases} t\omega & \text{si } t \text{ est une tâche primitive.} \\ \omega^{'}\omega & \text{si } t \text{ est une tâche composée} \\ & \text{et est décomposable dans } s. \\ \emptyset & \text{sinon.} \end{cases}$$

Nous notons $\omega \rightarrow^* \pi$, avec $\pi \in A^*$, si ω peut être décomposé en π en utilisant récursivement ζ sur l'ensemble des tâches du réseaux ω. Enfin, ω_I est le réseau de tâches initial.

Une solution à un problème de planification HTN est un réseau de tâches π tel que :

1. $\omega_I \rightarrow^* \pi$, où π est obtenu en décomposant ω_I.
2. $\gamma(s_0, \pi) \models g$, où π est applicable dans s_0 et atteint un état but.

Il est maintenant possible de définir un problème de planification HTN $P = (L, C, A, S, M, s_0, w_I, g, \delta, \tau, \lambda, \sigma, \zeta)$ en tant que langage formel comme suit :

2. Nous considérons uniquement les domaines totalement ordonnés

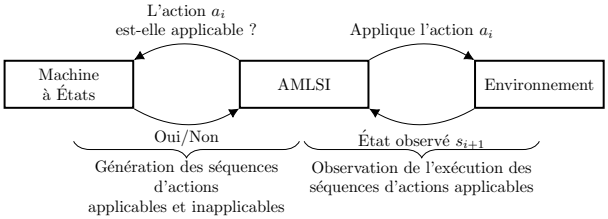


FIGURE 2 – AMLSI : Génération des observations.

$$\mathcal{L}(P) = \{\pi = (t_0 t_1 \ldots t_n) \mid t_i \in A, \gamma(s_0, \pi) \models g, \omega_I \rightarrow^* \pi\}$$

Ainsi, apprendre les actions et les méthodes d'un problème HTN $P = (L, C, A, S, M, s_0, w_I, g, \delta, \tau, \lambda, \sigma, \zeta)$ consiste à apprendre, à partir d'un ensemble donné d'observations $\Omega \subseteq \mathcal{L}(P)$, deux fonctions : δ, qui, comme pour la planification STRIPS, définit les actions, mais aussi σ, qui définit les méthodes.

Cependant, contrairement aux problèmes STRIPS, le langage $\mathcal{L}(P)$, produit par un problème HTN, n'est pas nécessairement régulier [16] et ne peut pas être représenté par un AFD. Comme mentionné par [16, 15], $\mathcal{L}(P)$ est l'intersection de deux langages :

1. $\mathcal{L}_{\mathcal{A}}(P) = \{\pi \in A^* \mid \gamma(s_0, \pi) \models g\}$, qui est défini par le système de transition d'états défini par les préconditions et les effets des tâches primitives. $\mathcal{L}_{\mathcal{A}}(P)$ est régulier.
2. $\mathcal{L}_{\mathcal{T}}(P) = \{\pi \in A^* \mid w_I \rightarrow^* \pi\}$, qui est défini par la hiérarchie de décomposition, c'est-à-dire par les tâches composées et les méthodes. $\mathcal{L}_{\mathcal{T}}(P)$ n'est pas nécessairement régulier.

Le principe de notre approche est : (i) apprendre l'AFD correspondant au langage régulier $\mathcal{L}_{\mathcal{A}}(P)$ et apprendre la fonction δ, (ii) modifier l'AFD appris en ajoutant des transitions de tâches composées afin d'encoder $\mathcal{L}_{\mathcal{T}}(P)$ et d'approximer le langage $\mathcal{L}(P)$; enfin, (iii) apprendre σ. Une fois que les fonctions δ et σ ont été apprises, il est possible de les généraliser dans un domaine HDDL.

3 Aperçu de l'algorithme AMLSI

Dans cette section, nous donnons un aperçu de l'algorithme AMLSI sur lequel HierAMLSI est basé (pour plus de détails, voir [8, 7]).

L'idée principale d'AMLSI est qu'il est possible d'apprendre une machine à états modélisant un problème de planification $P = (L, A, S, s_0, g, \tau, \delta, \lambda)$ en testant ses transitions et en observant les états résultants. Cette machine à état est une boite noire complexe que AMLSI apprend et représente sous la forme d'un domaine PDDL. AMLSI suppose que L, l'ensemble de propositions décrivant le monde,

A, l'ensemble des actions (les étiquettes des transitions de la machine à états), et s_0, l'état initial de P, sont connus. La fonction d'observation λ est considérée comme partielle et bruitée. Aucune hypothèse n'est faite sur le but g de P et sur la fonction τ qui retourne vrai lorsque une action est applicable dans un état. L'objectif d'AMLSI est d'apprendre δ qui définit les actions de P, et de généraliser δ afin d'obtenir un domaine PDDL.

L'algorithme AMLSI se compose de 5 étapes : (1) génération des observations, (2) apprentissage de l'AFD correspondant aux observations, (3) inférence des actions à partir de l'AFD appris, (4) généralisation de δ pour l'exprimer comme un ensemble d'actions PDDL, et (5) enfin, raffinement des actions PDDL pour traiter les observations partielles et bruitées des états.

Étape 1 (Génération des observations). La figure 2 donne un aperçu de cette étape. AMLSI génère l'ensemble des observations Ω avec une marche aléatoire. Ω est composé de deux sous-ensembles tels que $\Omega = I_+ \cup I_-$ et $I_+ \cap I_- = \emptyset$ construits comme suit. Si une action est applicable dans l'état courant, la séquence d'actions est ajoutée à I_+, l'ensemble des exemples positifs. Sinon, la séquence est ajoutée à I_-, l'ensemble des exemples négatifs.

Une fois les exemples positifs et négatifs générés, AMLSI exécute les exemples positifs et observe un ensemble d'états. Ce sont ces observations d'états qui sont bruitées et partielles. Plus précisément, une observation partielle, est un état contenant un sous-ensemble de propositions logiques non observées. Une observation bruitée est un état contenant un sous-ensemble de propositions logiques mal-évaluées.

Étape 2 (Apprentissage de l'AFD). Pour apprendre l'AFD $\Sigma = (S, A, \gamma)$, AMLSI utilise une variante de RPNI [32], un algorithme classique d'inférence grammaticale régulière, pour exploiter à la fois I_+ et I_-.

Étape 3 (Inférence des actions). À cette étape, AMLSI infère, pour chaque action, δ à partir de l'AFD. Pour les préconditions $prec(a)$ de l'action a, AMLSI calcule les propositions logiques qui sont dans tous les états précédant a dans Σ.

$$prec(a) = \cap_{s \in preset(a)} \lambda(s)$$

Pour les effets positifs $add(a)$ de l'action a, AMLSI calcule les propositions logiques qui ne sont jamais présentes dans les états avant l'exécution de a, et présentes après l'exécution de a.

$$add(a) = \cap_{s \in postset(a)} \lambda(s) \setminus prec(a)$$

Et inversement, pour les effets négatifs $del(a)$ de l'action a, AMLSI calcule les propositions logiques qui sont toujours

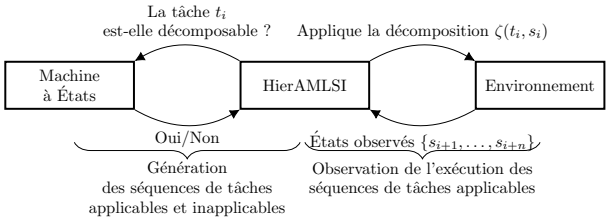


FIGURE 3 – HierAMLSI : Génération des observations.

présentes dans les états avant l'exécution de a, et absentes après l'exécution de a.

$$del(a) = prec(a) \setminus \cap_{s \in postset(a)} \lambda(s)$$

Étape 4 (Généralisation des actions). Une fois les préconditions et les effets inférés, les actions sont généralisées en actions PDDL. Les constantes dans les préconditions et les effets sont substitués par des variables. Ensuite, les préconditions et les effets des actions PDDL sont calculés comme l'intersection des préconditions et des effets de toutes les actions généralisées.

Étape 5 (Raffinement des actions). Pour traîter les observations partielles, AMLSI commence par raffiner les effets et les préconditions afin de garantir que chaque transition de l'AFD appris soit faisable. Pour ce faire, AMLSI ajoute tous les effets en veillant à ce que chaque transition dans le DFA soit faisable. Ensuite, AMLSI raffine les préconditions des actions. Comme dans [38], nous supposons que les effets négatifs d'une action doivent être présentes dans ses préconditions. Ainsi, pour chaque effet négatif, AMLSI ajoute les propositions correspondantes. Étant donné que le raffinement des effets dépend des préconditions, et que le raffinement des préconditions dépend des effets, AMLSI répète ces deux étapes jusqu'à convergence, c'est-à-dire jusqu'à ce que plus aucune précondition et plus aucun effet ne soit ajouté. Enfin, pour traîter les observations bruitées, AMLSI effectue une recherche Tabu [6]. Une fois que la recherche Tabu a atteint un optimum local, AMLSI répète les différentes étapes de raffinement jusqu'à convergence.

4 L'approche HierAMLSI

HierAMLSI est basé sur AMLSI. Il partage les mêmes hypothèses qu'AMLSI, c'est-à-dire qu'il est possible d'apprendre une machine à états modélisant un problème de planification HTN $P = (L, C, A, S, M, s_0, w_I, g, \delta, \tau, \lambda, \sigma, \zeta)$ en testant ses transitions et en observant les états résultants. Comme pour AMLSI, cette machine à état est une boite noire complexe que HierAMLSI apprend et représente sous la forme d'un domaine HDDL. De plus, L, A et s_0 sont

$I_+ = \{$ ⟨*(pick-up a), (stack a b), (unstack a b), (put-down a)*⟩, ⟨*(pick-up b), (stack b a)*⟩,
⟨*(pick-up b), (stack b a), (unstack b a), (put-down a), (pick-up b), (put-down b)*⟩, . . .}
$I_- = \{$ ⟨*(pick-up a), (stack a b), (unstack a b), (put-down a), (put-down a)*⟩,
⟨*(pick-up b), (stack b a), (stack b a)*⟩, ⟨*(stack b a)*⟩,
⟨*(pick-up b), (stack b a), (unstack b a), (put-down a), (stack b a)*⟩, . . .}

FIGURE 4 – Un exemple d'observations Ω contenant l'ensemble des séquences positives I_+ et l'ensemble des séquences négatives I_-.

connus, la fonction λ est partielle et bruitée, aucune hypothèse n'est faite sur le but de P. Dans HierAMLSI, la fonction de décomposition ζ est partiellement annotée. En d'autres termes, en ce qui concerne la décomposition, HierAMLSI observe seulement la tâche composée racine et n'a aucune connaissance sur les tâches composées intermédiaires. L'objectif de HierAMLSI est d'apprendre δ (les actions de P) et σ (les méthodes de P), et de généraliser ces fonctions dans leur représentation HDDL. L'approche HierAMLSI se compose de 7 étapes.

Étape 1 (Génération des observations). HierAMLSI produit un ensemble d'observations Ω avec une marche aléatoire en appliquant non seulement des tâches primitives comme dans AMLSI, mais aussi des tâches composées à partir de l'état initial de P. Cette différence est présentée dans 4.1.

Étape 2 (Apprentissage de l'AFD). HierAMLSI apprend l'AFD qui définit le système de transition du problème de planification P. Cette étape est similaire à l'étape 2 d'AMLSI.

Étape 3 (Inférence des actions). HierAMLSI infère le modèle d'actions δ à partir de l'AFD. Cette étape est également identique à l'étape 3 d'AMLSI.

Étape 4 (Annotation de l'AFD). HierAMLSI annote l'AFD appris en ajoutant des transitions pour les tâches composées afin d'approximer le langage $\mathcal{L}(P)$. Cette étape est détaillée dans 4.2.

Étape 5 (Inférence de méthodes). HierAMLSI infère σ qui définit les préconditions et les réseaux de tâches des méthodes observées à partir d'AFD annoté. L'inférence des préconditions des méthodes est effectuée en utilisant les techniques décrites à l'étape 3 d'AMLSI. L'inférence des réseaux de tâches des méthodes est spécifique à HierAMLSI. Nous détaillons ce point dans 4.3.

Étape 6 (Généralisation des méthodes et des actions). HierAMLSI généralise δ et σ en actions et méthodes HDDL. La méthode utilisée est celle décrite à l'étape 4 d'AMLSI.

Étape 7 (Raffinement des méthodes et actions HDDL). HierAMLSI utilise les techniques de raffinement décrites à l'étape 5 d'AMLSI pour affiner les actions et les méthodes HDDL afin de traiter les observations partielles et bruitées.

4.1 Génération des observations

La génération des observations (voir la figure 3) est similaire à celle utilisée dans AMLSI. Pour générer les observations Ω, HierAMLSI utilise des marches aléatoires en appliquant une tâche à partir de l'état initial du problème. La principale différence est que HierAMLSI peut choisir de manière aléatoire non seulement des tâches primitives (actions) mais aussi des tâches composées.

Si la tâche choisie est primitive et est applicable dans l'état courant, comme pour AMLSI, la séquence de tâches primitives de l'état initial à l'état courant est valide, et est ajoutée à I_+. Sinon, la séquence de tâches primitives est ajoutée à I_-. Si la tâche choisie est composée et est applicable dans l'état courant, la tâche composée est décomposée en une séquence de tâches primitives. La séquence de tâches primitives est alors ajouté à I_+. Sinon, la séquence dont la dernière tâche n'est pas applicable, est ajoutée à I_-. Les marches aléatoires sont répétées jusqu'à ce que I_+ et I_- atteignent une taille arbitraire.

Par exemple, supposons que nous ayons la séquence de tâches suivante générée à partir d'un problème simple de Blocksworld avec deux blocs a et b : ⟨ *(do-put-on a b), (do-clear a)*⟩, la séquence de tâches primitives générée et ajoutée à I_+ est : ⟨ *(pick-up a), (stack a b), (unstack a b), (put-down a)*⟩. Maintenant, supposons que nous ayons généré la séquence de tâches inapplicables suivante : ⟨ *(do-put-on a b), (do-clear a), (put-down a)*⟩, la séquence de tâches primitives ajoutée à I_- est : ⟨ *(pick-up a), (stack a b), (unstack a b), (put-down a), (put-down a)*⟩. La figure 4 donne un exemple d'un ensemble d'observations Ω.

Enfin, comme pour AMLSI, une fois les exemples positifs et négatifs générés, HierAMLSI exécute les exemples positifs et observe un ensemble d'états. Ce sont ces observations d'états qui sont bruitées et partielles.

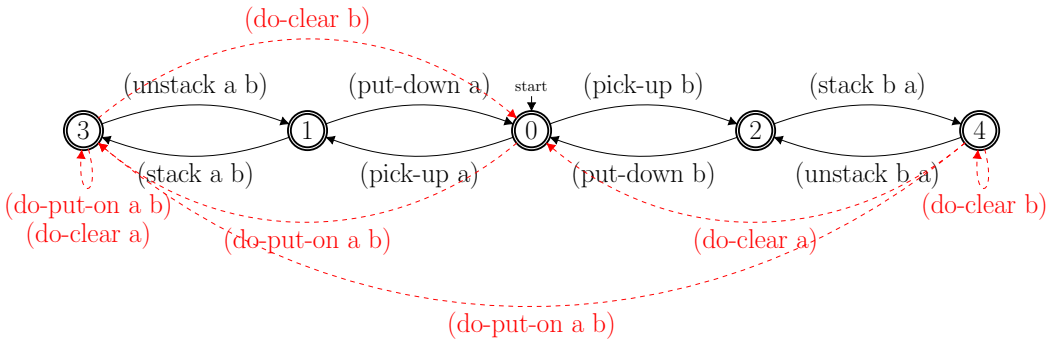


FIGURE 5 – Étapes d'apprentissage et d'annotation de l'AFD : l'AFD contenant uniquement les tâches primitives est représenté à l'aide des transitions noirs, et l'AFD comprenant les tâches composées est représenté avec les transitions rouges.

4.2 Annotation de l'AFD

Une fois que l'AFD des tâches primitives a été appris à partir des observations des tâches primitives, HierAMLSI annote l'AFD en ajoutant des transitions pour les tâches composées en insérant des transitions dont les étiquettes sont des tâches composées. Par exemple, supposons que nous ayons observé dans Ω que la tâche composée *(do-put-on a b)* a été décomposée par les tâches primitives $\langle$*(pick-up a), (stack a b)*$\rangle$ depuis le nœud 0 et a atteint le nœud 3. Alors, nous ajoutons la transition suivante à l'AFD : $\gamma(0,$ *(do-put-on a b)*$) \rightarrow 3$. La figure 5 donne un exemple d'AFD contenant à la fois les tâches primitives et composées.

4.3 Inférence des méthodes

Après avoir appris et annoté l'AFD, HierAMLSI infère la fonction σ qui définit les méthodes du problème de planification HTN. L'inférence des préconditions des méthodes est effectuée à l'aide de la méthode d'inférence utilisée pour inférer les préconditions des actions (voir l'étape 3 d'AMLSI pour plus de détails). Nous nous concentrons dans cette section sur l'inférence des méthodes.

Dans le pire cas, $|T|.|I_+|^2$ méthodes peuvent être inférées, où $|T|$ représente le nombre de tâches du problème et $|I_+|$ représente le nombre de tâches primitives dans l'ensemble I_+. Pour réduire le nombre de méthodes et avoir une représentation compacte des méthodes, il est nécessaire de minimiser l'ensemble de méthodes tout en garantissant qu'elles décomposent toutes les tâches composées observées. Trouver cet ensemble minimal peut être réduit à une variante du problème de couverture par ensembles (Set Cover Problem) [24], qui est NP-Complet. Pour faire face à cette complexité et approximer l'ensemble minimal de méthodes, nous proposons une procédure itérative.

La première étape consiste à initialiser σ en ajoutant une méthode pour chaque transition de tâche composée dans l'AFD et pour chaque décomposition possible de la tâche composée observée. Par exemple, trois méthodes sont créées pour la tâche composée *(do-put-on a b)*, une pour chacune de ses décompositions possibles de tâches primitives (voir figure 6a). Initialement, σ est composé de 7 méthodes. Ensuite, à chaque itération et pour chaque tâche composée de σ qui n'a pas déjà été choisie lors d'une itération précédente, nous calculons l'ensemble minimal de méthodes qui permet de décomposer toutes les tâches. Ce calcul est effectué avec une approximation gloutonne (GA) [2]. GA est un processus itératif polynomial qui, à chaque étape, ajoute le réseau de tâches de la méthode couvrant le plus grand nombre de décompositions. GA s'arrête une fois que toutes les décompositions sont couvertes par l'ensemble de méthodes.

Supposons que nous voulions calculer l'ensemble minimal des méthodes de la tâche composée *(do-put-on a b)* en considérant la tâche composée *(do-clear a)*. À l'itération 1, nous pouvons réduire la tâche composée *(do-put-on a b)*, initialement avec 3 méthodes $\omega_1 = \emptyset$, $\omega_2 = \langle$*(pick-up a), (stack a b)*$\rangle$, $\omega_3 = \langle$*(unstack b a), (put-down b), (pick-up a), (stack a b)*$\rangle$ à une représentation plus compacte avec seulement 2 méthodes : $\omega_1 = \emptyset, \omega_2 = \langle$*(do-clear a), (pick-up a), (stack a b)*$\rangle$ (voir figure 6b). À la fin de chaque itération, σ est défini sur l'ensemble minimal calculé. La tâche composée utilisée pour obtenir le nouvel ensemble de méthodes σ ne peut pas être réutilisée pour l'itération suivante. Les itérations s'arrêtent lorsqu'il n'y a plus de tâche composée pouvant être utilisée pour calculer un ensemble plus petit de méthodes. Par conséquent, la procédure proposée effectue au plus k itérations où $k \leq |C|$ est le nombre de tâches composées dans σ lors de la première itération. L'ensemble des méthodes inférées est présenté dans la figure 6c.

(do-clear a) : $\omega_1 = \emptyset, \omega_2 = \langle (unstack\ b\ a), (put\text{-}down\ b) \rangle$
(do-clear b) : $\omega_1 = \emptyset, \omega_2 = \langle (unstack\ a\ b), (put\text{-}down\ a) \rangle$
(do-put-on a b) : $\omega_1 = \emptyset, \omega_2 = \langle (pick\text{-}up\ a), (stack\ a\ b) \rangle, \omega_3 = \langle (unstack\ b\ a), (put\text{-}down\ b), (pick\text{-}up\ a), (stack\ a\ b) \rangle$

(a) **Itération 0 :** Ensemble initial des méthodes inférées

(do-clear a) : $\omega_1 = \emptyset, \omega_2 = \langle (unstack\ b\ a), (put\text{-}down\ b) \rangle$
(do-clear b) : $\omega_1 = \emptyset, \omega_2 = \langle (unstack\ a\ b), (put\text{-}down\ a) \rangle$
(do-put-on a b) : $\omega_1 = \emptyset, \omega_2 = \langle (do\text{-}clear\ a), (pick\text{-}up\ a), (stack\ a\ b) \rangle$

(b) **Itération 1 :** Ensemble des méthodes inférées en considérant la tâche composée *do-clear a*.

(do-clear a) : $\omega_1 = \emptyset, \omega_2 = \langle (unstack\ b\ a), (put\text{-}down\ b) \rangle$
(do-clear b) : $\omega_1 = \emptyset, \omega_2 = \langle (unstack\ a\ b), (put\text{-}down\ a) \rangle$
(do-put-on a b) : $\omega_1 = \emptyset, \omega_2 = \langle (do\text{-}clear\ a), (do\text{-}clear\ b), (pick\text{-}up\ a), (stack\ a\ b) \rangle$

(c) **Itération k :** Ensemble des méthodes inférées à la dernière itération.

FIGURE 6 – Exemple d'itérations de la procédure d'approximation de la décomposition minimale des tâches utilisée pour inférer les méthodes.

5 Évaluation expérimentale

En pratique, l'évaluation des algorithmes d'apprentissage, en particulier ceux liés à l'apprentissage de domaines hiérarchiques, peut être complexe. La récente normalisation du langage HDDL en 2020 [18] complique davantage la comparaison avec les approches plus anciennes. De plus, les entrées fournies à ces algorithmes peuvent varier considérablement, et l'expressivité des domaines appris peut également différer (voir la section 6). Dans la littérature seules quelques approches [39, 40, 36] partagent la capacité de HierAMLSI a apprendre à la fois les actions et les méthodes. Cependant une trop grande disparité dans les entrées empêche toute comparaison entre ces méthodes et la nôtre. Par conséquent, il n'est pas possible, à notre connaissance, de sélectionner une approche comme référence dans la littérature pour mener une évaluation équitable avec HierAMLSI.

Notre évaluation consiste à évaluer la capacité de HierAMLSI à : (1) apprendre uniquement des actions [3] ; (2) apprendre uniquement des méthodes et (3) apprendre à la fois des actions et des méthodes. Pour chacun de ces cas, nous considérons quatre scénarios expérimentaux distincts : (1) observations complètes (100%) et non bruitées (0%) ; (2) observations complètes (100%) et bruitées (20%) ; (3) observations partielles (20%) et non bruitées (0%) ; et (4) observations partielles (20%) et bruitées (20%).

Il est important de noter que les observations bruitées et partielles sont introduites exclusivement dans les séquences d'états. Ces perturbations sont introduites en supprimant ou en modifiant une partie des propositions, les propositions spécifiques étant sélectionnées de manière aléatoire.

Chaque scénario est testé sur un benchmark composé de 8 domaines HDDL [18, 19] (voir le tableau 1 pour les caractéristiques des domaines) de la compétition IPC 2020 : Blocksworld, Childsnack, Transport, Zenotravel, Gripper, Spanner, Elevator et Robot.

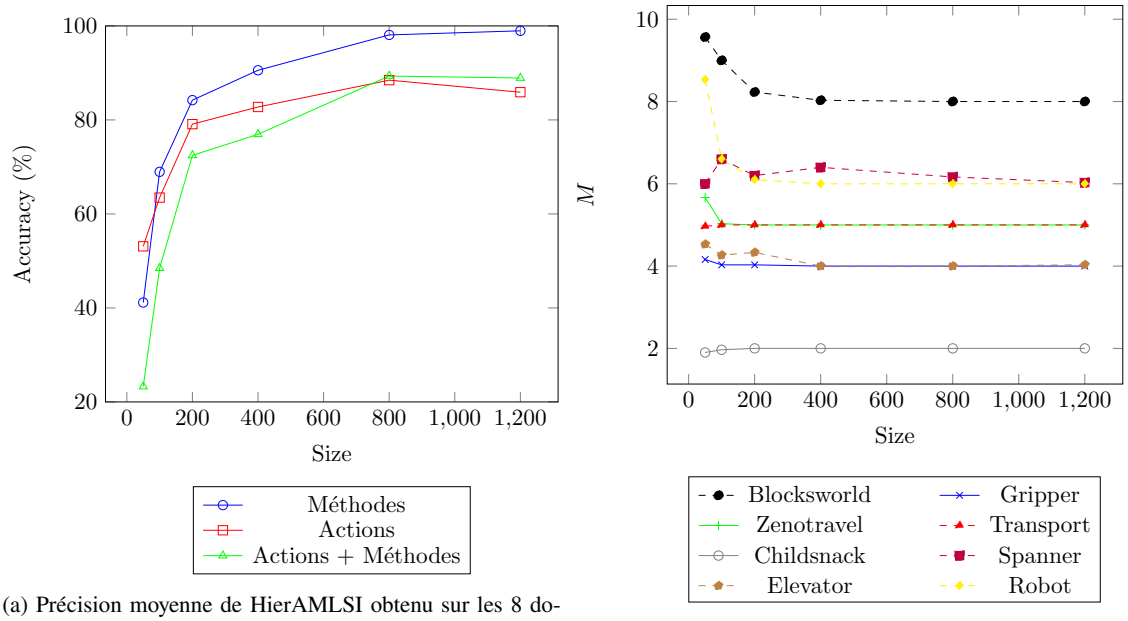
3. Ce scénario est équivalent à utiliser AMLSI seul

Domain	A	C	M	L
Blocksworld	4	4	8	5
Gripper	3	3	4	4
Zenotravel	4	2	5	7
Transport	3	4	5	5
Childsnack	6	1	2	12
Spanner	3	3	7	6
Elevator	2	4	5	7
Robot	4	4	7	8

TABLE 1 – Charactéristiques du benchmark, de gauche à droite, le nombre de tâche primitives A, le nombre de tâches composées C, le nombre de méthode M et le nombre de prédicat L pour chaque domaine IPC.

De plus, HierAMLSI apprend un domaine HDDL à partir d'un seul problème. Cependant, pour éviter tout biais potentiel découlant du choix d'un problème spécifique, HierAMLSI est évalué avec trois problèmes distincts. Pour chaque problème, l'apprentissage est répété et évalué dix fois. Tous les tests ont été effectués sur un serveur Ubuntu 14.04 équipé d'un processeur Intel Xeon CPU E5-2630 multi-cœur fonctionnant à 2,30 GHz et de 16 Go de mémoire. Les données d'apprentissage ont été générés à l'aide de la bibliothèque PDDL4J [33].

Pour conclure l'évaluation de notre approche, nous démontrons comment HierAMLSI se comporte dans le scénario le plus difficile (scénario 4) en faisant varier le nombre de traces d'observations en entrée. Ce scénario vise à montrer que HierAMLSI peut apprendre efficacement des domaines HDDL même avec des données d'entrée limitées.



(a) Précision moyenne de HierAMLSI obtenu sur les 8 domaines de notre benchmark.

(b) Nombre moyen de méthodes inférées par HierAMLSI.

FIGURE 7 – Performances moyenne de HierAMLSI sur les 8 domaines de notre benchmark lorsque la taille des données d'apprentissage varie.

5.1 Critères d'évaluation

Traditionnellement, deux critères sont couramment utilisés pour évaluer les techniques d'apprentissage de domaines de planification : l'*erreur syntaxique* [42], qui quantifie la dissimilarité entre le domaine original et le domaine appris en termes de syntaxe, et la *précision* [41], qui évalue la performance du domaine appris lorsqu'il est utilisé pour résoudre de nouveaux problèmes.

Bien que l'erreur syntaxique soit fréquemment utilisée dans la littérature, il est important de noter que HierAMLSI est uniquement évalué en termes de précision. Il existe plusieurs raisons à ce choix. Tout d'abord, la précision est probablement la mesure la plus significative pour la planification, car elle mesure la capacité d'un domaine appris à résoudre de nouveaux problèmes. En pratique, même une légère divergence entre le domaine original et le domaine appris telle qu'une précondition ou un effet manquant, qui entraînerait une petite erreur syntaxique, peut rendre un domaine incapable de résoudre de nouveaux problèmes de planification. Deuxièmement, HierAMLSI ne possède pas de connaissances sur les noms des méthodes dans les domaines IPC nécessaires pour la comparaison avec les domaines appris. Il ne dispose que d'informations sur les noms des tâches. Par conséquent, le calcul de l'erreur syntaxique, qui repose sur la comparaison de la syntaxe des méthodes, n'est pas réalisable dans ce contexte.

Formellement, la précision $Acc = N/N^*$ est calculée comme le rapport entre N, qui représente le nombre de problèmes correctement résolus en utilisant le domaine appris, et N^*, qui représente le nombre total de problèmes à résoudre. Dans cette évaluation, la précision est calculée avec 20 problèmes. Ces problèmes sont résolus en utilisant le planificateur TFD, qui fait partie de la bibliothèque PDDL4J [33]. La validation des plans est effectuée avec VAL, l'outil de validation de la compétition IPC [20].

5.2 Discussion

La tableau 2 présente la précision obtenue par HierAMLSI sur les domaines de notre benchmark pour tous les scénarios expérimentaux. Indépendamment du scénario, HierAMLSI apprend systématiquement des domaines précis avec une précision variant de 57% dans le pire des cas à 100% dans le meilleur des cas. Plus précisément, lorsque les actions sont connues, HierAMLSI atteint généralement une précision élevée dans l'apprentissage des domaines (> 95%) dans tous les scénarios expérimentaux. Il est à noter que les performances diminuent légèrement lorsque HierAMLSI doit apprendre à la fois les actions et les méthodes. Cependant, même pour les scénarios les plus difficiles, avec des niveaux élevés de bruits et d'observations partielles, les domaines appris maintiennent une bonne précision.

Il est important de noter que, même dans le scénario le plus difficile, seules 200 tâches sont nécessaires pour apprendre des domaines précis (comme indiqué dans la figure 7a). Cela démontre que HierAMLSI peut produire des

Observabilité		100%		20%	
Bruit		0%	20%	0%	20%
Blocksworld	A	100%	86.7%	100%	76.7%
	M	100%	100%	100%	100%
	A + M	100%	86.7%	76.7%	83.2%
Gripper	A	100%	100%	100%	100%
	M	100%	100%	100%	100%
	A+M	100%	100%	100%	100%
Zenotravel	A	100%	100%	100%	100%
	M	100%	100%	100%	100%
	A+M	100%	100%	100%	100%
Transport	A	100%	100%	100%	100%
	M	100%	100%	100%	100%
	A+M	100%	100%	100%	100%
Childsnack	A	100%	74.8%	96.7%	82.5%
	M	100%	100%	100%	100%
	A+M	100%	80.5%	96.7%	82.5%
Spanner	A	100%	94.7%	83%	54.7%
	M	96.7%	96.7%	96.7%	96.7%
	A+M	96.7%	91.3%	96.7%	82.7%
Elevator	A	100%	100%	100%	100%
	M	95%	95%	95%	95%
	A+M	95%	95%	94.8%	95%
Robot	A	83.5%	90%	86.8%	73.3%
	M	100%	100%	100%	100%
	A+M	83.3%	90%	86.7%	73.3%

TABLE 2 – Précision de HierAMLSI obtenue sur les 8 domaines de notre benchmark. Les données d'apprentissage sont composées de 30 séquences de 40 tâches. 3 variantes sont testées : (A) seules les actions sont apprises, (M) seules les méthodes sont apprises et (A+M) à la fois les actions et les méthodes sont apprises

domaines de bonne qualité avec un minimum de données, soulignant son efficacité dans l'apprentissage de domaine.

Enfin, la figure 7b montre le nombre de méthodes inférées par HierAMLSI. Pour la plupart des domaines, à l'exception de Spanner et Elevator, HierAMLSI a tendance à inférer un nombre similaire de méthodes à celles des domaines IPC (voir Table 1). Cependant, pour les domaines Spanner et Elevator, le nombre de méthodes inférées est inférieur à celui des domaines IPC. Dans ces cas, les réseaux de tâches inférés au sein des méthodes sont trop généraux. Cela entraîne la génération de plans solution excessivement longs ou incorrects.

6 État de l'art

De nombreuses approches ont été développées pour apprendre des domaines de planification HTN. Ces approches peuvent être catégorisées en fonction du type de données d'entrée utilisées dans le processus d'apprentissage et de la sortie qu'elles produisent. La sortie peut inclure le modèle d'actions, l'ensemble des méthodes, et peut ou non inclure les préconditions des méthodes. Les données d'entrée peuvent comprendre des plans générés en résolvant un ensemble de problèmes de planification, des plans annotés, des arbres de décomposition, ou des marches aléatoires. De plus, les données d'entrée peuvent incorporer des états observés en plus des tâches, et ces états peuvent être entièrement observables, partiellement observables, ou bruités.

Ces approches se concentrent principalement sur les aspects structurels de la décomposition des tâches. Notamment, des approches telles que [37, 13, 25, 10, 11, 1, 27, 26, 35] reposant sur des traces de plans comme données d'entrée. Certaines approches, telles que HTN-Maker [14, 12], LHTNDT [30] et l'approche proposée par [4], utilisent également des plans annotés, segmentés en différentes tâches et leurs décompositions intermédiaires, mais l'acquisition de tels exemples annotés demande un effort humain important.

Ces approches présentent plusieurs limitations. Elles se concentrent uniquement sur l'apprentissage des méthodes. De plus, les préconditions des méthodes ne sont pas toujours apprises. Seules les approches proposées par [23, 21, 22, 11, 31, 26, 35, 4] apprennent les préconditions des méthodes.

Enfin, seules quelques approches [39, 40, 36] sont capables d'apprendre à la fois des modèles d'actions et des méthodes. Cependant, aucune de ces approches n'est robuste aux observations bruitées et les domaines appris ne sont pas suffisamment précis pour être utilisées directement, c'est à dire sans la moindre correction humaine, par un planificateur.

7 Conclusion

Dans cet article, nous avons présenté l'algorithme d'apprentissage de domaines HDDL HierAMLSI. HierAMLSI est parmi les rares algorithmes capables d'apprendre à la fois des actions et des méthodes, et il détient la capacité unique d'apprendre à partir d'observations partielles et bruitées. Notre évaluation expérimentale a démontré que HierAMLSI peut apprendre de manière efficace des domaines de planification précis même dans des scénarios avec des niveaux élevés de bruit, et ce, avec un minimum de données d'entrée. De plus, HierAMLSI montre la capacité à dériver une représentation concise des méthodes, le nombre de méthodes apprises se rapprochant de celui des domaines d'origines.

Malgré ces résultats prometteurs, l'expressivité des domaines appris par HierAMLSI reste limitée. Une piste d'amélioration serait d'augmenter l'expressivité des domaines appris. Plus précisément, des travaux récents ont montré des résultats prometteurs pour l'apprentissage de domaines temporels non hiérarchiques [9, 5]. Une piste d'amélioration serait d'étendre notre approche pour apprendre des domaines temporels hiérarchiques en tirant parti des efforts en cours de la communauté pour incorporer des caractéristiques temporelles dans le langage HDDL [34].

Références

[1] Kevin Chen, Nithin Shrivatsav Srikanth, David Kent, Harish Ravichandar, and Sonia Chernova. Learning hierarchical task networks with preferences from unannotated demonstrations. In *Proc. of CoRL*, pages 1572–1581, 2021.

[2] Vasek Chvátal. A greedy heuristic for the set-covering problem. *Math. Oper. Res.*, 4(3) :233–235, 1979.

[3] Stephen Cresswell, Thomas Leo McCluskey, and Margaret Mary West. Acquiring planning domain models using *LOCM*. *Knowl. Eng. Rev.*, 28(2) :195–213, 2013.

[4] Morgan Fine-Morris, B Auslander, Michael W Floyd, Greg Pennisi, Héctor Muñoz-Avila, and EDU Kalyan Moy Gupta. Learning hierarchical task networks with landmarks and numeric fluents by combining symbolic and numeric regression. In *Proc. of the 2020 Conference on Advances in Cognitive Systems*, 2020.

[5] Antonio Garrido and Sergio Jiménez. Learning temporal action models via constraint programming. In *Proc. of ECAI*, pages 2362–2369, 2020.

[6] Fred W. Glover and Manuel Laguna. *Tabu Search*. Kluwer, 1997.

[7] M. Grand, H. Fiorino, and D. Pellier. Amlsi : A novel and accurate action model learning algorithm. In *Proc. of KEPS Workshop*, 2020.

[8] M. Grand, H. Fiorino, and D. Pellier. Retro-engineering state machines into pddl domains. In *Proc. of ICTAI*, pages 1186–1193, 2020.

[9] Maxence Grand, Damien Pellier, and Humbert Fiorino. Tempamlsi : Temporal action model learning based on STRIPS translation. In *Proc. of ICAPS*, pages 597–605, 2022.

[10] Bradley Hayes and Brian Scassellati. Autonomously constructing hierarchical task networks for planning and human-robot collaboration. In *Proc. of ICRA*, pages 5469–5476, 2016.

[11] Philippe Hérail and Arthur Bit-Monnot. Leveraging demonstrations for learning the structure and parameters of hierarchical task networks. In *Proc. of FLAIRS*, 2023.

[12] Chad Hogg, Ugur Kuter, and Héctor Munoz-Avila. Learning hierarchical task networks for nondeterministic planning domains. In *Proc. of IJCAI*, 2009.

[13] Chad Hogg, Ugur Kuter, and Hector Munoz-Avila. Learning methods to generate good plans : Integrating htn learning and reinforcement learning. In *Proc. of AAAI*, volume 24, 2010.

[14] Chad Hogg, Héctor Munoz-Avila, and Ugur Kuter. Htn-maker : Learning htns with minimal additional knowledge engineering required. In *Proc. of AAAI*, pages 950–956, 2008.

[15] Daniel Höller. Translating totally ordered htn planning problems to classical planning problems using regular approximation of context-free languages. In *Proc. of ICAPS*, volume 31, pages 159–167, 2021.

[16] Daniel Höller, Gregor Behnke, Pascal Bercher, and Susanne Biundo. Language classification of hierarchical planning problems. In *Prof. of ECAI*, pages 447–452, 2014.

[17] Daniel Höller, Gregor Behnke, Pascal Bercher, and Susanne Biundo. Assessing the expressivity of planning formalisms through the comparison to formal languages. In *Proc. of ICAPS*, page 158–165, 2016.

[18] Daniel Höller, Gregor Behnke, Pascal Bercher, Susanne Biundo, Humbert Fiorino, Damien Pellier, and Ron Alford. Hddl : An extension to pddl for expressing hierarchical planning problems. In *Proc. of AAAI*, pages 9883–9891, 2020.

[19] Daniel Höller, Gregor Behnke, Pascal Bercher, Susanne Biundo, Humbert Fiorino, Damien Pellier, and Ronald Alford. Hierarchical planning in the IPC. In *Proc. of HPlan Workshop (ICAPS)*, 2019.

[20] Richard Howey and Derek Long. Val's progress : The automatic validation tool for pddl2. 1 used in the international planning competition. In *Proc. of IPC Workshop*, pages 28–37, 2003.

[21] Okhtay Ilghami, Héctor Muñoz-Avila, Dana S. Nau, and David W. Aha. Learning approximate preconditions for methods in hierarchical plans. In *Proc. of ICML*, pages 337–344, 2005.

[22] Okhtay Ilghami, Dana S. Nau, and Héctor Muñoz-Avila. Learning to do HTN planning. In *Proc. of ICAPS*, pages 390–393, 2006.

[23] Okhtay Ilghami, Dana S. Nau, Héctor Muñoz-Avila, and David W. Aha. Camel : Learning method preconditions for HTN planning. In *Proc. of ICAPS*, pages 131–142, 2002.

[24] Richard M. Karp. Reducibility among combinatorial problems. In *Proc. of a symposium on the Complexity of Computer Computations*, pages 85–103, 1972.

[25] Nan Li, William Cushing, Subbarao Kambhampati, and Sungwook Yoon. Learning probabilistic hierarchical task networks as probabilistic context-free grammars to capture user preferences. *ACM Trans. Intell. Syst. Technol.*, 5(2) :1–32, 2014.

[26] Ruoxi Li, Mark Roberts, Morgan Fine-Morris, and Dana Nau. Teaching an htn learner. In *Proc. of HPlan*, pages 68–72, 2022.

[27] Damir Lotinac and Anders Jonsson. Constructing hierarchical task models using invariance analysis. In *Proc. of ECAI*, pages 1274–1282, 2016.

[28] Drew McDermott, Malik Ghallab, Adele Howe, Craig Knoblock, Ashwin Ram, Manuela Veloso, Daniel Weld, and David Wilkins. PDDL-the planning domain definition language, 1998.

[29] Kira Mourão, Luke S. Zettlemoyer, Ronald P. A. Petrick, and Mark Steedman. Learning STRIPS operators from noisy and incomplete observations. In *Proc. of UAI*, pages 614–623, 2012.

[30] Fatemeh Nargesian and Gholamreza Ghassem-Sani. Lhtndt : Learn htn method preconditions using decision tree. In *Proc. of ICINCO-ICSO*, pages 60–65, 2008.

[31] Conny Olz, Susanne Biundo, and Pascal Bercher. Revealing hidden preconditions and effects of compound htn planning tasks–a complexity analysis. In *Proc. of AAAI*, pages 11903–11912, 2021.

[32] Jose Oncina and Pedro García. Inferring regular languages in polynomial update time. In *Pattern Recognition and Image Analysis : Selected Papers from the IVth Spanish Symposium*, volume 1, pages 49–61. World Scientific, 1992.

[33] D. Pellier and H. Fiorino. PDDL4J : a planning domain description library for java. *J. Exp. Theor. Artif. Intell.*, 30(1) :143–176, 2018.

[34] Damien Pellier, A. Albore, Humbert Fiorino, and R. Bailon-Ruiz. HDDL 2.1 : Towards defining a formalism and a semantics for temporal htn planning. In *Proc. of the HPlan Workshop*, 2023.

[35] Greg Pennisi, Morgan Fine-Morris, Michael W Floyd, Bryan Auslander, Héctor Munoz-Avila, Jeff Heflin, and Kalyan Moy Gupta. Htn learning via transfer learning of domain landmarks. In *Proc. of the 2021 International Florida Artificial Intelligence Research Society Conference*, 2021.

[36] José A Segura-Muros, Raúl Pérez, and Juan Fernández-Olivares. Learning htn domains using process mining and data mining techniques. In *Proc. of Generalized Planning Workshop*, 2017.

[37] Zhanhao Xiaoa, Hai Wan, Hankui Hankz Zhuoa, Andreas Herzigb, Laurent Perrusselc, and Peilin Chena. Learning htn methods with preference from htn planning instances. *Proc. of HPlan Workshop*, page 31, 2019.

[38] Qiang Yang, Kangheng Wu, and Yunfei Jiang. Learning action models from plan examples using weighted MAX-SAT. *Artif. Intell.*, 171(2-3) :107–143, 2007.

[39] Hankz Hankui Zhuo, Derek Hao Hu, Chad Hogg, Qiang Yang, and Hector Muñoz-Avila. Learning HTN method preconditions and action models from partial observations. In *Proc. of IJCAI*, pages 1804–1810, 2009.

[40] Hankz Hankui Zhuo, Héctor Muñoz-Avila, and Qiang Yang. Learning hierarchical task network domains from partially observed plan traces. *Artificial intelligence*, 212 :134–157, 2014.

[41] Hankz Hankui Zhuo, Tuan Anh Nguyen, and Subbarao Kambhampati. Refining incomplete planning domain models through plan traces. In *Proc. of IJCAI*, pages 2451–2458, 2013.

[42] Hankz Hankui Zhuo, Qiang Yang, Derek Hao Hu, and Lei Li. Learning complex action models with quantifiers and logical implications. *Artif. Intell.*, 174(18) :1540–1569, 2010.

Actes JIAF 2024

Extension de la planification HTN aux problèmes temporels

Nicolas Cavrel Humber Fiorino Damien Pellier

Univ. Grenoble Alpe, LIG, France
Nom.Prenom@imag.fr

Résumé

Cet article présente TEP (Temporal Event Planning), une approche originale conçue pour les réseaux de tâches hiérarchiques (HTN) temporels. TEP se distingue par sa capacité à aborder les trois catégories de problèmes temporels, telles que catégorisées par Cushing, ce qui le différencie des approches prédominantes qui s'attaquent principalement aux deux premières catégories. TEP accomplit cela en traduisant les problèmes HTN temporels en problèmes non temporels, permettant ainsi d'utiliser des heuristiques développées dans la planification HTN non temporelle pour orienter la recherche de solutions temporelles et de surpasser les approches existantes. Cette stratégie innovante commence par affiner les tâches en actions instantanées et par relâcher les contraintes de durée, assurant ainsi une cohérence avec les heuristiques de recherche conventionnelles tout en préservant l'expressivité inhérente du problème.

1 Introduction

Le formalisme de planification des réseaux de tâches hiérarchiques (HTN) est conçu pour représenter les problèmes de planification sous forme d'ensembles de tâches complexes. Conformément à la stratégie classique de division et de conquête, ces problèmes sont résolus en décomposant de manière récursive les tâches complexes en tâches plus simples jusqu'à ce que le problème ne se compose uniquement de tâches indécomposables, appelées "actions".

Alors que la plupart des approches HTN se concentrent principalement sur la planification HTN non temporelle, e.g., [6, 12, 17], où les actions sont considérées comme instantanées, nous nous attaquons dans cet article à la résolution de problèmes de planification HTN temporelle, où les actions sont définies sur des intervalles de temps. La prise en compte de cette dimension est essentielle lors de la résolution de problèmes du monde réel, où les contraintes temporelles jouent un rôle essentiel et où la synchronisation entre agents est impérative. Les différentes approches proposées pour résoudre les problèmes de planification temporelle peuvent être catégorisées en fonction de leur capacité à aborder différentes catégories de problèmes temporels, telles que définies par Cushing [9]. La classification de Cushing distingue trois catégories de problèmes de planification temporelle. Dans la première catégorie, les problèmes temporels ont des solutions consistant en des plans séquentiels : la concurrence des actions n'est pas nécessaire (solutions non concurrentes). La deuxième catégorie englobe les problèmes temporels où les plans de solution peuvent potentiellement être concurrents, mais les solutions séquentielles sont toujours valides (solutions éventuellement concurrentes). La troisième catégorie comprend les problèmes temporels où les seuls plans de solution réalisables sont intrinsèquement concurrents, et les solutions séquentielles ne sont pas possibles (solutions concurrentes nécessaires). La plupart des approches existantes relèvent des deux premières catégories de Cushing. C'est le cas par exemple de [2, 1, 11, 22, 25], où les actions temporelles sont prétraitées en séquences de tâches non temporelles [2] ou d'actions. Un avantage important de ces approches est leur capacité à tirer parti des heuristiques de recherche et des algorithmes développés dans un cadre non temporel. D'autre part, certaines approches telles que par exemple [26, 4, 14, 7] résolvent toutes les catégories de Cushing, mais elles manquent d'efficacité en raison d'un manque d'heuristiques informatives.

Dans cet article, nous présentons TEP (Temporal Event Planning), une nouvelle approche de planification hybride regroupant des méthodes de liaison causale à ordre partiel et des méthodes HTN, adaptée aux domaines HTN temporels. Notre contribution réside dans la capacité de TEP à résoudre toutes les catégories de problèmes HTN de Cushing, tout en utilisant des heuristiques développées pour la planification HTN non temporelle. TEP commence par compiler les problèmes HTN temporels en problèmes non temporels. Il accomplit cela en affinant les tâches abstraites et les tâches duratives en actions non temporelles instantanées. Ensuite, il tente de trouver un plan de solution en relâchant les contraintes de durée, uniquement en vérifiant la cohérence des contraintes. Le problème non temporel résultant,

une fois relâché, ressemble étroitement à un problème HTN classique, ce qui le rend compatible avec les heuristiques de recherche non temporelles. En particulier, il conserve l'expressivité requise pour résoudre des problèmes dans toutes les catégories de Cushing. Si une solution au problème relâché est trouvée, TEP procède en calculant des affectations de timestamp qui satisfont les contraintes temporelles avec un solveur de CSP (Problème de Satisfaction de Contraintes).

Le document est structuré comme suit. La partie 1 introduit le formalisme de planification HTN temporelle. Dans la partie 2, l'algorithme TEP est présenté. La partie 3 démontre comment TEP résout les problèmes HTN dans la troisième catégorie de Cushing. La partie 4 examine en détail les heuristiques adoptées de la planification hybride non temporelle et utilisées dans TEP pour aborder les problèmes de planification HTN temporelle. Enfin, la partie 5 décrit l'évaluation de TEP.

2 Formalisation

Dans cette partie, nous proposons une formalisation qui intègre les caractéristiques temporelles dans la planification HTN. Les notations sont fondées sur [19] et [6].

2.1 Tâches, Réseau de Tâches, Action et Méthodes

Les concepts clés en planification HTN, et a fortiori en planification HTN temporelle, sont les *tâches* et les *réseaux de tâches*. Une *tâche* est définie par un nom et une liste de paramètres. Nous distinguons trois types de tâches : les tâches instantanées, les tâches duratives et les tâches abstraites. Contrairement aux *tâches instantanées* qui modifient l'état du monde, les *tâches duratives* et les *tâches abstraites* ne le font pas. Ce sont des noms faisant référence à d'autres tâches (soit instantanées, duratives ou abstraites) qui doivent être accomplies selon certaines contraintes. Chaque tâche a un début et une fin. Nous faisons référence respectivement au début et à la fin d'une tâche t avec des variables temporelles notées $v^s t$ et $v^e t$. Une tâche t est instantanée, i.e., que $v^s t = v^e t$, donc nous nous référerons simplement à l'instant où s'exécute la tâche instantanée t comme v_t. Un état s est défini comme un ensemble de propositions logiques.

Un *réseau de tâches* représente un ensemble partiellement ordonné de tâches. Un réseau de tâches w est un tuple $(I, \alpha, \prec)$ où I est un ensemble d'identifiants de tâches, $\alpha : I \mapsto T$ un mapping d'identifiants de tâches vers des noms de tâches [1] T, et $\prec$ un ensemble de contraintes d'ordonnancement sur les identifiants de tâches dans I. Les contraintes d'ordonnancement sont définies sur les variables de temporelles de début ou de fin des identifiants de tâches I. Les contraintes d'ordonnancement possibles sont celles de l'algèbre de points classique [8] : $<$, $\leq$, $>$, $\geq$, $=$ et $\neq$. Nous permettons également les contraintes de la forme $d = v^e t - v^s t$ pour exprimer la durée de la tâche. Par exemple, la contrainte d'ordonnancement temporel $v^s t_1 < v^e t_2$ exprime que le début de la tâche $t_1 \in I$ doit se produire strictement avant la fin de $t_2 \in I$. Notez qu'un tel ensemble de contraintes C peut être représenté sous forme d'un graphe de contraintes dont la cohérence peut être vérifiée en calculant ses composantes fortement connectées en temps polynomial $O(|C|)$. Les tâches instantanées, duratives et abstraites peuvent être réalisées respectivement en appliquant des *actions instantanées*, des *actions duratives* et des *méthodes* définies ci-dessous.

1. Les identifiants de tâches sont nécessaires car une tâche peut apparaître plusieurs fois dans un réseau de tâches.

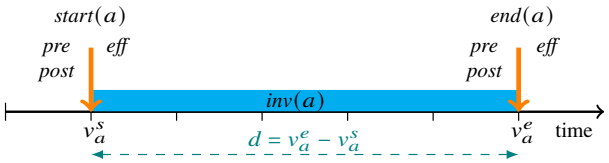


FIGURE 1 – Chronologie de l'application d'une action durative a.

Une *action instantanée* a est un tuple $(name(a), pre(a), post(a), \mathit{eff}(a))$, où $name(a)$ est le nom de la tâche accomplie par a et $pre(a)$, $post(a)$ et $\mathit{eff}(a)$ sont des ensembles de propositions. Soit v_a l'instant auquel a est exécuté. Une action instantanée a **deux** ensembles de conditions à satisfaire pour être exécutée : $pre(a)$ qui doit être vraie strictement avant v_a (cas classique en planification) et $post(a)$ qui doit être vraie strictement après v_a, i.e., dans l'état résultant de l'exécution de a. $post(a)$ **est une nouveauté pour exprimer des propriétés invariantes qui doivent être satisfaites sur un intervalle de temps**, et qui seront utilisées pour résoudre la 3ème catégorie de Cushing.

Enfin, comme en planification non temporelle, l'exécution de a produit les effets $\mathit{eff}(a)$ tels que $\mathit{eff}(a) = \mathit{eff}^+(a) \cup \mathit{eff}^-(a)$ et $\mathit{eff}^+(a) \cap \mathit{eff}^-(a) = \emptyset$ où $\mathit{eff}^+(a)$ et $\mathit{eff}^-(a)$ sont des ensembles de propositions, respectivement vraies et fausses après l'exécution de a.

Une *action durative* a est un tuple $(name(a), start(a), end(a), inv(a), d)$: $name(a)$ est la tâche accomplie par a, $start(a)$ et $end(a)$ sont des actions instantanées ; $inv(a)$ est un ensemble de propositions qui doivent être vraies après l'exécution de $start(a)$ et jusqu'au début de $end(a)$, i.e., sur l'intervalle $]v^s a, v^e a[$ et $d = v_a^e - v_a^s$ est la durée de a. Nous supposons comme en planification temporelle [10] que $v_a^s < v_a^e$ est vrai. Par conséquent, la durée de a est un nombre strictement positif. Le déroulement temporel de l'application d'une action durative est donné dans la Figure 1.

Une *méthode* m est un tuple $(name(m), w)$, où $name(m)$ est le nom de la tâche accomplie par m et w est le réseau de tâches qui définit comment $name(m)$ doit être décomposée en sous-tâches.

2.2 Plans Temporels Partiels, Défauts et Menaces

Pour traiter des caractéristiques temporelles, nous étendons la définition d'un *plan partiel* généralement utilisé en planification hybride [6] et en planification POCL [15] afin de gérer les *postconditions* des actions instantanées comme suit. Un *plan temporel partiel* π est un tuple (w, C) où $w = (I, \alpha, \prec)$ est un réseau de tâches qui définit les tâches de π et ses contraintes d'ordonnancement, et C est un ensemble de liens de causalité de la forme $\langle t_i \xrightarrow{p} t_j \rangle$ avec t_i et t_j deux identifiants de tâches instantanées dans I, et p une proposition. Le but d'un lien de causalité est de confirmer qu'une précondition ou une postcondition p d'une tâche t_j est soutenue par une autre tâche t_i qui produit p comme effet. Deux cas doivent être considérés selon que t_i soutient une précondition ou une postcondition de t_j. Si t_i soutient une précondition de t_j, c'est-à-dire que $p \in \mathit{eff}(t_i)$ et $p \in \mathit{pre}(t_j)$ alors $(v^e_{t_i} < v^s_{t_j}) \in \prec$ (cas classique en planification POCL). Si t_i soutient une postcondition de t_j, c'est-à-dire que $p \in \mathit{eff}(t_i)$ et $p \in \mathit{post}(t_j)$ alors $(v^e_{t_i} \leq v^s_{t_j}) \in \prec$: dans ce cas, **la contrainte d'ordonnancement n'est pas stricte**. Par extension, une tâche instantanée t_k dans un plan temporel partiel π est une *menace* sur un lien de causalité $\langle t_i \xrightarrow{p} t_j \rangle$ en fonction de si t_i soutient une précondition ou une postcondition p de t_j. Si t_i soutient une précondition p de t_j, alors t_k menace $\langle t_i \xrightarrow{p} t_j \rangle$ si et seulement si t_k a $\neg p$ comme effet, et $(v^e_{t_i} < v^s_{t_k})$, $(v^e_{t_k} < v^s_{t_j})$ sont compatibles avec $\prec$ (le cas habituel en planification POCL). Si t_i soutient une postcondition p de t_j, alors t_k menace $\langle t_i \xrightarrow{p} t_j \rangle$ si et seulement si t_k a $\neg p$ comme effet, et $(v^e_{t_i} < v^s_{t_k})$, $(v^e_{t_k} \leq v^s_{t_j})$ dans $\prec$. Ici, la contrainte d'ordonnancement entre t_k et t_j n'est pas stricte. Un *défaut* dans un plan partiel $\pi = (w, C)$ avec $w = (I, \alpha, \prec)$ est soit (1) une condition ouverte, c'est-à-dire une précondition ou une postcondition d'une tâche qui n'est pas soutenue par un lien de causalité, (2) une menace, c'est-à-dire une tâche qui peut interférer avec un lien de causalité, ou (3) un défaut de décomposition, c'est-à-dire une tâche abstraite ou durative qui n'est pas décomposée en tâches instantanées.

2.3 Problème HTN temporel et solution

Un problème de planification HTN (Hierarchical Task Network) temporelle P est un tuple $(L, T, A, M, s_0, w_0, g)$, où L est un ensemble fini de propositions, T est un ensemble de tâches, A est un ensemble d'actions duratives, M est l'ensemble des méthodes, s_0 est l'état initial construit sur L, w_0 est le réseau de tâches initial à réaliser, et g est un ensemble de propositions définissant l'objectif à atteindre.

La solution d'un problème de planification HTN temporelle est un plan temporel partiel π obtenu en affinant un plan partiel initial π_0 comme dans la planification POCL en tâches instantanées en appliquant des actions duratives et des méthodes. Le plan initial π_0 est construit avec w_0 comme réseau de tâches et deux tâches instantanées spéciales t_0 et t_∞ ordonnées respectivement comme la première tâche et la dernière tâche de w_0. t_0 n'a pas de précondition et l'état initial comme effets positifs. t_∞ a l'objectif comme précondition et pas d'effets. Formellement, un plan temporel partiel $\pi = (w, C)$ est une solution d'un problème de planification $P = (L, T, A, M, s_0, w_0, g)$ si et seulement si : (1) π est un affinement du plan temporel partiel initial π_0, (2) π est exécutable dans l'état initial s_0. Ainsi, (2.1) toutes les tâches dans π sont des tâches instantanées, (2.2) π n'a pas de défauts, c'est-à-dire pas de précondition ou postcondition ouverte, et pas de menaces, (2.3) pour toutes les tâches t dans π, v_t est assigné et satisfait les contraintes d'ordonnancement dans w.

Il reste à définir comment affiner un plan temporel partiel en un plan ne contenant que des tâches instantanées en utilisant des actions duratives et des méthodes. Tout d'abord, considérons le cas de l'affinement d'une tâche durative. Pour effectuer cet affinement, il est d'abord nécessaire de modifier légèrement la définition des deux actions instantanées $\mathit{start}(a)$ et $\mathit{end}(a)$ pour traduire les propriétés invariantes $\mathit{inv}(a)$ dans la logique POCL. Cette modification consiste, d'une part, à ajouter $\mathit{inv}(a)$ aux postconditions et aux effets de $\mathit{start}(a)$, et, d'autre part, aux préconditions de $\mathit{end}(a)$. Il est maintenant possible d'exprimer les propriétés invariantes d'une tâche durative par des liens de causalité classiques entre les effets de $\mathit{start}(a)$ et les préconditions de $\mathit{end}(a)$ conformément à la sémantique PDDL 2.1, ce qui contraint $\mathit{inv}(a)$ à être vérifié sur l'intervalle $]v^s_a, v^e_a[$.

Formellement, soit $a = (\mathit{name}(a), \mathit{start}(a), \mathit{end}(a), \mathit{inv}(a), d)$ une action durative qui affine un identifiant de tâche i d'un plan $\pi_1 = (w_1, C_1)$ avec $w_1 = (I_1, \alpha_1, \prec_1)$ tel que $i \in I_1$. Alors, a affine π_1 en un plan $\pi_2 = (w_2, C_2)$ avec $w_2 = (I_2, \alpha_2, \prec_2)$ et

$$
\begin{aligned}
I_2 &= (I_1 - \{i\}) \cup \{i^s, i^e\} \\
\alpha_2 &= (\alpha_1 - (i, \mathit{name}(a))) \cup (i^s, \mathit{start}(a)) \cup (i^e, \mathit{end}(a)) \\
\prec_2 &= \{v^s_i < v^e_i\} \cup \{v^e_i - v^s_i = d\} \\
&\quad \cup \{(v'_i < v^s_i) \mid \forall (v'_i < v_i) \in \prec_1\} \\
&\quad \cup \{(v'_i \leq v^s_i) \mid \forall (v'_i \leq v_i) \in \prec_1\} \\
&\quad \cup \{(v^e_i < v'_i) \mid \forall (v_i < v'_i) \in \prec_1\} \\
&\quad \cup \{(v^e_i \leq v'_i) \mid \forall (v_i \leq v'_i) \in \prec_1\}
\end{aligned}
\tag{1}
$$

Enfin, le dernier cas est celui de l'affinement de méthode. Soit $m = (t_m, w_m)$ une méthode avec un réseau de tâches $w_m = (I_m, \alpha_m, \prec_m)$ qui affine un identifiant de tâche i d'un plan $\pi_1 = (w_1, C_1)$ avec $w_1 = (I_1, \alpha_1, \prec_1)$ tel que $i \in I_1$. Alors, m affine π_1 en un plan $\pi_2 = (w_2, C_2)$ avec $w_2 = (I_2, \alpha_2, \prec_2)$ et

$$
\begin{aligned}
I_2 &= (I_1 - \{i\}) \cup I_m \\
\alpha_2 &= \alpha_1 \cup \alpha_m - \{(i, t_m)\} \\
\prec_2 &= \prec_m \ \cup \{(v'_i < v^s_i) \mid \forall (v'_i < v_i) \in \prec_1\} \\
&\quad \cup \{(v'_i \leq v^s_i) \mid \forall (v'_i \leq v_i) \in \prec_1\}
\end{aligned}
$$

$$\cup \{(v_i^e > v_i') \mid \forall (v_i > v_i') \in \prec_1\}$$
$$\cup \{(v_i^e \geq v_i') \mid \forall (v_i \geq v_i') \in \prec_1\}$$
$$\cup \{(v_i' \geq v_i^s) \mid v_i' \in I_m)\} \cup \{(v_i' \leq v_i^e) \mid v_i' \in I_m)\}$$
$$\cup \{(v_i^s \leq v_i^e)\}$$
$$C_2 = C_1 \tag{2}$$

3 Temporal Event Planner

Dans cette partie, nous définissons la procédure de recherche de TEP (*Temporal Planning Event*). Cette procédure est fondée sur la procédure de planification hybride POCL proposée par [6]. Elle traite des deux ensembles de conditions qui doivent être remplies pour exécuter une action : les préconditions et les postconditions. Nous montrons comment TEP est capable d'utiliser les heuristiques développées pour la planification hybride non temporelle et comment elle peut résoudre toutes les catégories de problèmes temporels de Cushing [9]. Il est important de souligner que les problèmes de la troisième catégorie de Cushing sont des problèmes dont les plans de solution nécessitent nécessairement la concurrence. À notre connaissance, TEP est le seul planificateur hybride capable de résoudre cette catégorie de Cushing.

3.1 Procédure de recherche

La principale idée de TEP est de mener la recherche d'un plan de solution en intercalant deux étapes. La première étape affine un plan partiel initial en un plan ne contenant que des tâches instantanées, comme dans la planification hybride, tout en vérifiant la cohérence des contraintes d'ordonnancement des tâches. Contrairement à la planification hybride, TEP vérifie à la fois la cohérence des contraintes $<$ et $\leq$. Cette première étape se résume à résoudre un problème non temporel, ce qui permet à la recherche d'être guidée par des heuristiques classiques. Nous détaillerons ce point dans la prochaine section. L'objectif de la première étape de TEP est de vérifier les critères 1, 2 (a) et 2 (b) d'un plan de solution. La deuxième étape attribue une valeur à toutes les variables temporelles associées aux tâches instantanées du plan affiné, en respectant les contraintes d'ordonnancement ainsi que les contraintes de durée $d = v_a^e - v_a^s$ [2].

La procédure de recherche TEP est donné dans l'algorithme 1. Il prend en entrée un problème $P = (L, T, A, M, s_0, w_0, g)$ et renvoie un plan de solution π ainsi qu'un ensemble V représente les valeurs affectées aux variables temporelles pour chaque tâche instantanée si la procédure réussit, et un échec sinon. TEP commence (ligne 1) par exprimer l'état initial s_0, le réseau de tâches initial w_0 et l'objectif g sous la forme d'un plan partiel à raffiner π_0.

2. Pour des raisons de simplicité, nous nous limitons ici à ce type de contraintes, en gardant à l'esprit que l'approche CSP peut être utilisée pour exprimer des contraintes plus complexes.

Algorithm 1: TEP$(L, T, A, M, s_0, w_0, g)$

```
1  π_0 ← the inital plan built from s_0, w_0 and g
2  open ← {π_0}
3  while open ≠ ∅ do
4      π ← non-deterministically select plan in open
5      flaws ← the set of flaws of π
6      if flaws = ∅ then
7          V ← search timestamp assignments for π
8          if V ≠ ∅ then return (π, V)
9          else return Failure
10     φ ← arbitrarily select a flaw in flaws
11     open ← open ∪ solveFlaw(π, φ)
12 return Failure;
```

π_0 a w_0 comme réseau de tâches et deux tâches instantanées spéciales t_0 et t_∞ ordonnées respectivement comme la première tâche et la dernière tâche de w_0. t_0 n'a pas de précondition et l'état initial comme effet positif. t_∞ a l'objectif comme précondition et aucun effet. Ensuite (ligne 2), π_0 est ajouté à la liste en attente des plans partiels à explorer et la première étape de raffinement démarre. À chaque itération, un nouveau plan partiel π est sélectionné (ligne 4) et ses défauts sont calculés (ligne 5). Si π contient des défauts, l'un d'eux est sélectionné (ligne 10). La résolution de ce défaut génère un nouvel ensemble de plans temporels partiels qui sont ajoutés à la liste en attente des plans partiels à explorer (ligne 11). Si un plan sans défaut est sélectionné (ligne 7), TEP passe à la deuxième étape et utilise un solveur CSP pour trouver l'ensemble V des affectations des variables temporelles pour toutes les tâches instantanées de π. Si le solveur CSP réussit, π et V sont renvoyés, sinon la procédure continue d'itérer, retourne à la première étape et essaie de raffiner un nouveau plan partiel.

3.2 Réparation des défauts

Les défauts dans les plans partiels sont très similaires aux défauts dans les réseaux de tâches non temporels. Nous avons 3 types différents de défauts : (1) des conditions ouvertes, i.e., des préconditions ou des postconditions de tâches qui ne sont pas supportées par un lien de causalité, (2) des menaces, i.e., des tâches qui peuvent interférer avec un lien de causalité, ou (3) des défauts de décomposition, i.e., des tâches abstraites ou durables qui ne sont pas encore décomposées en tâches instantanées. Les défauts sont réparés comme suit.

Conditions ouvertes. Nous distinguons deux mécanismes de réparation pour les conditions ouvertes en fonction de savoir si une condition ouverte p d'une tâche $t_j \in \pi$ est une précondition ou une postcondition (voir Figure 2). Si p est une *précondition*, alors p est réparé en ajoutant à π un *lien de causalité* $\langle t_i \xrightarrow{p} t_j \rangle$ et une contrainte d'ordonnancement $(t_i < t_j)$. Si p est une *postcondition*, alors p

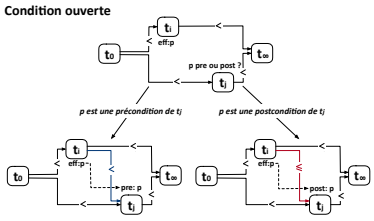


FIGURE 2 – Réparation d'une condition ouverte : Un lien de causalité $\langle t_i \xrightarrow{p} t_j \rangle$ est ajouté au plan pour soutenir la condition ouverte, et une contrainte de précédence ($t_i < t_j$) ou ($t_i \leq t_j$) selon que la condition ouverte est une *précondition* ou une *postcondition*.

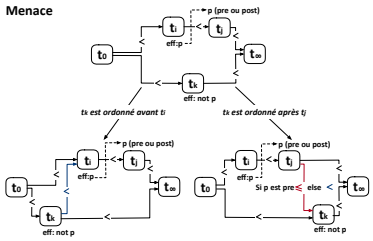


FIGURE 3 – Il existe deux façons de résoudre une *menace* t_k sur un lien de causalité $\langle t_i \xrightarrow{p} t_j \rangle$, on peut soit ordonner la tâche menaçante strictement avant t_i, soit en même temps (ou après) la tâche t_j selon que la condition ouverte soutenue est une précondition ou une postcondition.

est réparé en ajoutant à π un *lien de causalité* $\langle t_i \xrightarrow{p} t_j \rangle$ et une contrainte d'ordonnancement ($t_i \leq t_j$). Dans les deux cas, les contraintes ajoutées doivent être cohérentes avec les contraintes d'ordonnancement actuelles dans π. Notez que la réparation des préconditions ouvertes est effectuée comme dans la planification hybride non temporelle, tandis que la réparation des postconditions ouvertes introduit des contraintes $\leq$ au lieu de $<$.

Menaces. Supposons qu'une tâche t_k menace un lien de causalité $\langle t_i \xrightarrow{p} t_j \rangle$. Cette menace peut être résolue de deux manières différentes (voir Figure 3). La première consiste à contraindre t_k strictement avant la tâche t_i qui produit p en ajoutant ($t_k < t_i$) dans les contraintes d'ordonnancement. La deuxième consiste à contraindre t_k après ou en même temps que la tâche t_j en ajoutant la contrainte ($t_j \leq t_k$) si p est une précondition de t_j, ou strictement après si p est une postcondition de t_j. Comme toujours, l'ajout de contraintes d'ordonnancement à π doit garantir la cohérence des contraintes. La réparation d'une menace est très similaire à la réparation de menace dans la planification hybride, sauf que nous autorisons des contraintes d'ordonnancement non strictes.

Défauts de décomposition. Nous distinguons deux mécanismes de réparation différents selon que $\pi = (w, C)$

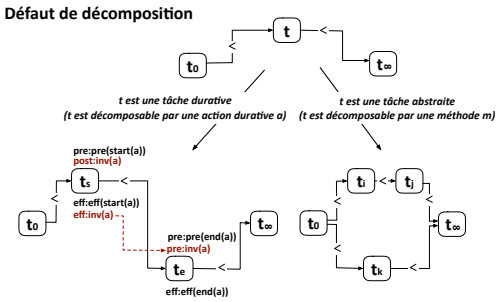


FIGURE 4 – Les deux façons possibles de décomposer une tâche dépendent de savoir si la tâche est durative (à droite) ou abstraite (à gauche).

contient une tâche durative ou abstraite t à décomposer en appliquant respectivement les Équations 1 et 2. La Figure 4 montre un exemple de décomposition de tâche durative (à droite) et de décomposition de tâche abstraite (à gauche). Pour la décomposition de tâche durative, nous supposons que t est décomposée par une action durative a telle que $t_s = start(a)$ et $t_e = end(a)$. Nous montrons en rouge les modifications des deux tâches instantanées t_s et t_e nécessaires pour traduire les propriétés invariantes $inv(a)$ dans la logique POCL. Pour la décomposition de tâche abstraite, nous supposons que t est décomposée par une méthode m en trois sous-tâches t_i, t_j, t_k avec $t_i < t_k$. Aucune autre contrainte ne lie t_i, t_j et t_k.

3.3 Un exemple de la troisième catégorie de Cushing

Dans cette partie, nous illustrons comment TEP gère la troisième catégorie de la classification de Cushing où les plans de solution nécessitent une concurrence nécessaire.

Supposons que deux tâches duratives t_a et t_b peuvent être décomposées par deux actions duratives a et b. La chronologie de l'exécution de a et b est donnée par la Figure 5 : a et b ont la même durée d. L'action a produit l'effet *(doing a)* et a *(doing b)* comme invariant, et symétriquement, b produit l'effet *(doing b)* et a *(doing a)* comme invariant. Le problème n'a pas de méthodes, et l'état initial et l'état final du problème sont vides. Le réseau de tâches initial du problème contient les tâches t_a et t_b sans contraintes d'ordonnancement. Le plan partiel obtenu par TEP après la décomposition de t_a et t_b en tâches instantanées en appliquant les actions duratives a et b, est représenté dans la Figure 6. La seule solution à ce problème est un plan concurrent où t_a et t_b sont exécutées en même temps.

La Figure 6 représente le plan partiel obtenu après la décomposition de t_a et t_b avec a et b. Il présente quatre défauts. Deux défauts sont des conditions ouvertes qui doivent être soutenues par des liens causaux : la postcondition *(doing a)* de la tâche t_a^s et la postcondition *(doing b)* de la tâche t_b^s. Les deux autres défauts sont des menaces : la tâche t_a^e menace le lien causal $\langle t_b^s \xrightarrow{(doing\ a)} t_b^e \rangle$ et la tâche t_b^e menace le

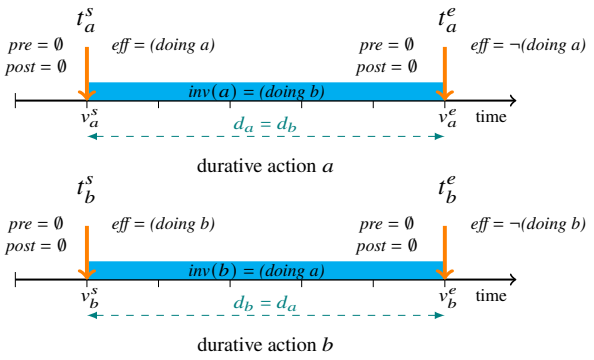


FIGURE 5 – Les deux actions duratives du problème de Cushing

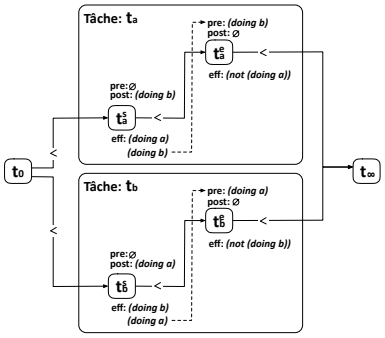


FIGURE 6 – Plan partiel obtenu dans le problème de Cushing après avoir décomposé les deux tâches duratives t_a et t_b en appliquant les actions duratives a et b. Les flèches en pointillés représentent les liens causaux et les flèches pleines en noir représentent les contraintes d'ordonnancement.

lien causal $\langle t_a^s \xrightarrow{(doing\ b)} t_a^e \rangle$.

Supposons que les conditions ouvertes soient sélectionnées et résolues en premier. Pour corriger ces deux défauts, TEP doit ajouter au plan partiel deux liens causaux, le premier $\langle t_b^s \xrightarrow{(doing\ b)} t_a^s \rangle$ pour soutenir la postcondition *(doing b)* de t_a^s et le deuxième $\langle t_a^s \xrightarrow{(doing\ a)} t_b^s \rangle$ pour soutenir la postcondition *(doing a)* de t_b^s. En plus de ces deux liens causaux, TEP doit ajouter deux contraintes d'ordonnancement indiquant que t_a^s doit être exécuté avant ou en même temps que t_b^s puisque t_a^s supporte maintenant une postcondition de t_b^s, et inversement, que t_b^s doit être exécuté avant ou en même temps que t_a^s puisque t_b^s supporte maintenant une postcondition de t_a^s. Le plan partiel résultant est représenté dans la Figure 7. Les liens causaux et les contraintes d'ordonnancement ajoutés sont indiqués en rouge.

Finalement, les menaces doivent être corrigées. En général, une menace est résolue en contraignant la tâche instantanée menaçante soit *strictement* avant le lien causal menacé, soit *strictement ou en même temps* après le lien causal, selon

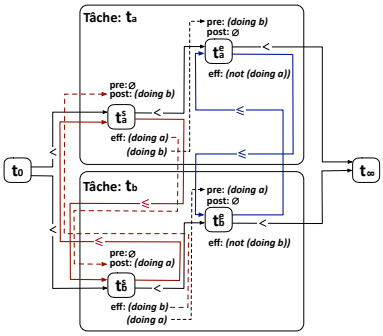


FIGURE 7 – Le plan partiel correspondant au problème de Cushing après la résolution des deux conditions ouvertes et des deux menaces. Les liens causaux et les contraintes d'ordonnancement en rouge représentent la modification du plan partiel nécessaire pour corriger les conditions ouvertes, et en bleu pour corriger les menaces.

que le lien causal supporte une précondition ou une postcondition. Dans notre exemple, la seule résolution valide est de contraindre t_b^e (resp. t_a^e) après ou en même temps que t_a^e (resp. t_b^e). Le plan partiel résultant est affiché sur la figure 7. Les contraintes d'ordonnancement ajoutées sont indiquées en bleu. TEP a maintenant résolu tous les défauts. TEP renvoie le plan partiel de la figure 7 comme solution.

4 Les heuristiques de TEP

Les heuristiques de TEP ont pour principal avantage d'exploiter (avec certaines adaptations présentées ci-dessous) les heuristiques développées pour la planification hybride non temporelle afin de résoudre des problèmes temporels. La procédure de recherche de TEP (voir Algo. 1) repose sur deux choix non déterministes, qui sont en pratique réalisés en utilisant deux fonctions heuristiques. Le premier choix effectue une sélection non déterministe parmi l'ensemble des plans partiels temporels en attente à explorer, et décide lequel explorer en premier (ligne 4). Les fonctions heuristiques guidant ce choix sont appelées *heuristiques de sélection de plan* et impactent considérablement à la fois les performances de la recherche (le temps nécessaire pour trouver un plan de solution) et la qualité du plan retourné (le nombre d'actions dans le plan de solution).

Le deuxième choix (ligne 9) effectue une sélection non déterministe parmi les défauts à résoudre dans le plan partiel actuel. Les fonctions heuristiques guidant ce choix sont appelées *heuristiques de sélection de défaut*. Notez que chaque défaut dans le plan partiel doit éventuellement être résolu afin de trouver un plan de solution. Ainsi, la *faisabilité* de la procédure TEP ne dépend que des heuristiques de sélection de plan. Cependant, l'*ordre* dans lequel les défauts sont résolus par rapport aux heuristiques de sélection de défaut impacte considérablement les performances de recherche de la procédure. Dans ce qui suit, nous présen-

tons les heuristiques de sélection de plan et de défaut telles qu'implémentées dans TEP.

4.1 Les heuristiques de sélection de plan

Parmi les heuristiques de sélection de plan pour la planification hybride non temporelle, e.g., [24, 23, 6], les plus efficaces sont celles développées par [6]. Leur principe est d'estimer le nombre de changements nécessaires pour atteindre une solution de plan en comptant le nombre de défauts dans le plan partiel actuel. L'idée de Bercher est d'étendre ce principe à la planification hybride en estimant plus précisément le nombre de défauts dans un plan en tenant compte des défauts qui seront introduits par les tâches qui n'ont pas encore été décomposées. Cette estimation est basée sur une structure appelée Graphes de Décomposition de Tâches (TDG) qui encode la décomposition du problème hiérarchique. Dans la section suivante, nous présentons (1) comment TEP étend le concept de TDG pour encoder non seulement les décompositions de tâches abstraites mais aussi les décompositions de tâches duratives dans une nouvelle structure appelée Graphes de Décomposition de Tâches Temporelles (TTDG), et (2) comment les heuristiques classiques pour la planification hybride développées dans [6, 5] peuvent être dérivées à partir d'un TTDG pour résoudre les problèmes de planification hybride temporelle.

Graphes de Décomposition de Tâches Temporelles. Un TTDG d'un problème de planification $P = (L, T, A, M, s_0, w_0, g)$ est un graphe orienté AND/OR $G = (V_T, V_M, V_A, E)$, où V_T est un ensemble de sommets comprenant des tâches instantanées, duratives et abstraites qui peuvent être obtenues en décomposant le plan partiel initial π_0 construit à partir de s_0, w_0, et g. V_M et V_A sont respectivement des ensembles de sommets de méthode ou de durative qui décomposent une tâche abstraite ou une tâche durative dans V_T. Enfin, E est un ensemble d'arêtes qui relient les nœuds de V_T à V_M ou V_A. Pour des raisons de brièveté, les nœuds enfants d'un nœud v sont notés $child(v) = v_i | (v, v_i) \in E$. Un TTDG est illustré dans la Figure 8. Dans le cas général, un TTDG comme un TDG peut être un graphe cyclique.

Le calcul des heuristiques. Pour généraliser les différentes heuristiques de sélection de plan développées en planification hybride à la planification temporelle, nous devons d'abord définir deux estimations cruciales qui peuvent être extraites d'un TTDG : (1) une estimation du nombre de décompositions nécessaires pour décomposer le plan en tâches instantanées, et (2) une estimation des conditions ouvertes à prendre en charge.

Ces deux estimations reposent sur les concepts de tâches obligatoires et de cardinalité des tâches. Les tâches obligatoires, notées $M(t)$, sont des tâches qui apparaissent dans toutes les méthodes de décomposition ou actions duratives

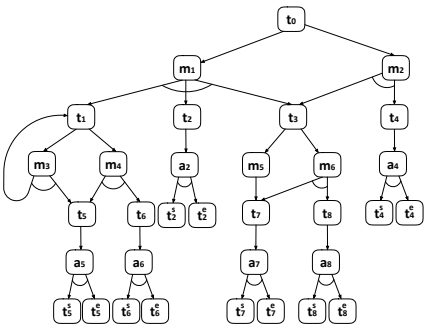


FIGURE 8 – Un exemple simple de TTDG représenté sous forme de graphe AND/OR. Les symboles t_0, t_1, t_3 représentent des tâches abstraites. $t_4, t_5 \ldots, t_8$ représentent des tâches duratives et $t_4^s, t_4^e, \ldots, t_8^s, t_8^e$ représentent des tâches instantanées. $m_1, \ldots m_6$ décrivent des sommets de méthode et $a_5, \ldots a_8$ des sommets d'actions duratives.

associées à la même tâche t. En termes plus simples, l'ensemble des tâches obligatoires $M(t)$ inclut les tâches qui seront obligatoirement incluses dans un plan partiel lorsque la tâche t est décomposée. Par exemple (voir Figure 8 pour le contexte), nous avons $M(t_o) = t_3, t_7, t_7^s, t_7^e$, $M(t_1) = t_5, t_5^s, t_5^e$, $M(t_3) = t_7, t_7^s, t_7^e$, $M(t_4) = t_4^s, t_4^e$, $M(t_5) = t_5^s, t_5^e$, $M(t_6) = t_6^s, t_6^e$, $M(t_7) = t_7^s, t_7^e$ et $M(t_8) = t_8^s, t_8^e$. La cardinalité d'une tâche t noté TC peut être définie de manière simple comme le nombre de tâches obligatoires, i.e., $TC = |M(t)|$. La cardinalité d'une tâche t sert de borne inférieure pour estimer l'effort requis pour la décomposer.

Le calcul de la première estimation revient à calculer TC. En pratique, cela est déterminé à partir d'un TTDG de la manière suivante :

$$TC(v) = \begin{cases} 1 & \text{si } v \in V_T \text{ et } v \text{ est instantanée} \\ \min\limits_{v_i \in child(v)} TC(v_i) & \text{si } v \in V_T \text{ et } v \text{ est abstraite} \\ \sum\limits_{v_i \in child(v)} TC(v_i) & \text{sinon} \end{cases}$$

La deuxième estimation, connue sous le nom de *MME* (Effort de Modification Minimal), représente le nombre de conditions ouvertes, comprenant à la fois les préconditions et les postconditions, qui doivent être satisfaites pour chaque nœud. Elle peut être déterminée à partir d'un TTDG de la manière suivante :

$$MME(v) = \begin{cases} 1 + |pre(v) \cup post(v)| & \text{si } v \in V_T \text{ et } v \text{ est inst.} \\ \sum\limits_{d \in Child} TC(d) & \text{si } v \in V_M. \\ 1 + \min\limits_{d \in Child} TC(d) & \text{si } v \in V_T \text{ et } v \text{ est abst.} \end{cases}$$

où $pre(v)$ désigne les préconditions d'un nœud primitif et $post(v)$ ses postconditions.

Il est finalement possible d'étendre le calcul des heuristiques à un plan temporel $\pi = (w, C)$ aux quatre heuristiques classiques proposées par [6, 5] comme suit. Ici,

$w = (I, \alpha, \prec)$ définit les tâches et leurs contraintes associées, tandis que C représente les liens de causalité entre les tâches au sein de π, et F désigne ses défauts.

$$h_{TC}(\pi) = \Sigma_{i \in I | \alpha(i) \text{ est abtraite}} TC(\alpha(i))$$
$$h_{MME}(\pi) = \Sigma_{i \in I} MME(\alpha(i))$$
$$h_{TDGm}(\pi) = h_{MME}(\pi) - |C|$$
$$h_{TC+\#F}(\pi) = h_{TC}(\pi) + |F|$$

nombre de modifications qui seront introduites dans le plan, comme mentionné dans [5]. Cependant, elles ne garantissent pas un plan de solution optimal en ce qui concerne le makespan. La preuve est omise par manque de place.

4.2 Heuristiques de sélection des défauts

La stratégie la plus courante pour sélectionner l'ordre des défauts est de choisir en premier les défauts ayant le moins de moyens de résolution possibles, ce qui signifie que les défauts les plus contraignants sont résolus en premier. Cette stratégie est connue sous le nom de LCFR (Least Cost Flaw Repair) [13]. Bien que cette heuristique de sélection de défauts ait été initialement développée pour des problèmes non hiérarchiques, elle a été utilisée en planification hybride par le planificateur hybride de pointe PANDA [6]. Lors de son adaptation à la planification hiérarchique, la priorité est donnée à la décomposition des tâches abstraites par rapport aux autres défauts pour maintenir la complétude de la procédure. Cela est dû au fait que le nombre de moyens de résolution pour un défaut dépend du choix de la décomposition.

TEP met en œuvre la même stratégie de sélection des défauts que PANDA, en donnant la priorité aux défauts de décomposition, puis en priorisant les autres défauts à partir de ceux ayant le moins de moyens de résolution. La seule différence ici est que les défauts de décomposition comprennent non seulement les tâches abstraites, mais aussi les tâches duratives qui ne sont pas décomposées en tâches instantanées.

5 Expérimentations

Dans cette partie, nous comparerons TEP et ses heuristiques avec l'approche *timeline* proposée par FAPE, retenue comme la meilleure configuration [7]. L'approche *timeline* est actuellement la seule approche capable de résoudre des problèmes temporels hiérarchiques dans la troisième catégorie de Cushing. Ainsi, à notre connaissance, seule l'approche *timeline* et TEP partagent la même expressivité.

5.1 Configuration Expérimentale

Les deux planificateurs, TEP et FAPE, ont été testés sur un seul cœur d'un processeur Intel Core i7-9850H, avec une limite de 8 Go de RAM et une limite de temps de 600 secondes. Pour garantir une comparaison équitable, les deux planificateurs ont été mis en œuvre en utilisant la même bibliothèque PDDL4J [20], et ont utilisé le même solveur CSP, celui fourni par OR-Tools [21]. Les critères d'évaluation sont (1) le *temps de résolution*, qui représente le temps nécessaire pour résoudre un problème, (2) le *makespan* d'un plan de solution, qui représente la longueur totale du plan, i.e., le temps entre le début et la fin du plan de solution, et enfin, (3) la *couverture* de chaque planificateur, i.e., le nombre de problèmes résolus de chaque domaine. Notez que le temps de résolution et le makespan sont présentés sous forme de scores IPC [3] (International Planning Competition). La meilleure configuration atteint un score IPC de 1, tandis que les autres configurations reçoivent un pourcentage basé sur leurs performances par rapport à la meilleure. Pour calculer le score d'un domaine, nous additionnons les scores IPC obtenus pour chaque problème. La couverture fait référence au nombre de problèmes résolus dans un domaine.

5.2 Benchmarks de planification

Il n'existe actuellement aucun benchmark standard disponible pour la planification hiérarchique et temporelle. Par conséquent, nous proposons un ensemble de benchmarks basés sur le langage HDDL2.1 [19], qui vise à faciliter la comparaison des planificateurs hiérarchiques et temporels. Parmi ces benchmarks, certains sont des adaptations de domaines hiérarchiques non temporels des compétitions de planification IPC. Ils ont été modifiés en ajoutant des durées aux actions : *Gripper*, *Satellite* et *Rover*. Dans le domaine *Gripper*, toutes les solutions sont séquentielles et aucune concurrence temporelle n'est requise (première catégorie de Cushing). Dans les domaines *Satellite* et *Rover*, des problèmes avec plusieurs satellites ou rovers permettent la concurrence, mais elle n'est pas obligatoire. Le planificateur peut choisir de trouver soit un plan simple et non concurrent, soit un plan concurrent utilisant plusieurs rovers. Ces problèmes appartiennent à la deuxième catégorie de Cushing. De plus, certains domaines ont été spécialement conçus pour correspondre à la troisième catégorie de Cushing :

— *Area Scan* modélise un ensemble de dispositifs hétérogènes qui coopèrent pour scanner des zones. Area Scan a deux versions : une version totalement ordonnée, où les tâches abstraites des problèmes sont totalement ordonnées, et une version désordonnée où la concurrence peut également être réalisée grâce à des tâches abstraites concurrentes.
— *Table Carriers* est inspiré du domaine *Gripper*. Dans ce domaine, les agents doivent coopérer pour transporter des tables, les tables nécessitant une, deux ou trois personnes pour les transporter simultanément avec succès.

3. https://ipc2023-htn.github.io/

	Gripper			**Satellite**			**Rover**			**AreaScan PO**			**AreaScan TO**			**TableCarriers**			**Total**		
	Temps	Span	Couv.	Temps	Span	Couv.	Temps	Span	Couv.	S. Temps	Span	Couv.	Temps	Span	Couv.	Temps	Span	Couv.	S. Temps	Span	Couv.
TEP(TC)	6.72	**7.00**	**7/10**	6.21	6.50	**8/9**	4.96	5.43	**10/10**	19.99	**21.83**	**22/22**	19.93	**21.97**	**22/22**	7.13	8.33	9/10	64.94	71.05	**78/83**
TEP(MME)	**6.81**	**7.00**	**7/10**	**7.09**	6.71	**8/9**	6.62	7.41	9/10	17.64	21.62	**22/22**	20.25	21.74	**22/22**	**8.57**	9.34	**10/10**	**66.99**	73.82	**78/83**
TEP(TDGm)	4.09	6.00	6/10	5.24	7.38	**8/9**	**8.56**	7.63	9/10	18.67	21.62	**22/22**	**21.71**	21.74	**22/22**	4.64	**9.77**	**10/10**	62.90	**74.14**	77/83
TEP(TC + #F)	4.79	6.00	6/10	6.71	**7.63**	**8/9**	6.67	7.26	**10/10**	**20.39**	21.62	**22/22**	21.48	21.77	**22/22**	3.67	8.61	9/10	63.70	72.88	77/83
FAPE	1.52	6.00	6/10	2.74	5.88	6/9	3.64	**7.99**	8/10	10.00	18.92	19/22	13.69	21.72	**22/22**	2.73	7.00	7/10	34.31	67.51	68/83

TABLE 1 – Résultats expérimentaux pour les différentes configurations sous forme de scores IPC (International Planning Competition) pour le temps, le makespan et la couverture.

5.3 Résultats

Les résultats sont présentés dans le Tableau 1. En termes de critère de *temps de résolution*, TEP a constamment surpassé FAPE dans tous les domaines. Cependant, le choix des heuristiques les plus performantes variait selon le domaine. Dans les domaines Gripper et Satellite, TEP combiné avec les heuristiques MME, appelé $TEP(MME)$, a obtenu les meilleurs scores. En revanche, dans le domaine Rover, la configuration $TEP(TDGm)$ a atteint le meilleur score pour la métrique du temps de résolution. Cette disparité peut être attribuée à la nature distinctive de MME et $TDGm$. Contrairement à MME, qui se concentre uniquement sur l'accomplissement de propositions au sein d'un plan partiel, les heuristiques $TDGm$ tiennent compte du nombre de liens causaux déjà intégrés dans le plan partiel. Par conséquent, tandis que $TEP(MME)$ affine continuellement un plan partiel particulier en ajoutant des liens causaux jusqu'à ce qu'il forme une solution ou soit élagué, $TDGm$ a tendance à présenter un comportement de type exploration en profondeur. L'approche contrastée en profondeur de $TDGm$ et la nature de type exploration en largeur de MME éclairent les différences dans l'efficacité de leurs heuristiques. Des différences similaires peuvent être observées entre TC et $TC+F$. Alors que la configuration $TEP(TC)$ obtient des scores supérieurs dans le domaine *AreaScan PO*, $TEP(TC + F)$ se distingue comme la configuration la plus performante dans le domaine *AreaScan TO*. Une fois de plus, l'incorporation du nombre de défauts restants dans $TC + F$ incite une stratégie de recherche en profondeur, ce qui s'avère avantageux pour la résolution de problème ou les plans solution sont assez longs. En revanche, le comportement de type exploration en largeur manifesté par les heuristiques TC conduit à de meilleures performances dans d'autres domaines.

Les différentes configurations de TEP donnent pratiquement toujours les meilleurs scores en termes de makespane. Dans les domaines *Gripper* et *AreaScan*, $TEP(TC)$ se distingue comme la configuration la plus performante. En revanche, dans le domaine *Satellite*, $TEP(TC + F)$ présente la meilleure performance, tandis que dans le domaine *TableCarriers*, $TEP(TDGm)$ surpasse les autres. De façon notable, FAPE obtient le meilleur score dans le domaine Rover. Dans l'ensemble, les configurations affichent des résultats comparables à travers les domaines testés. Les scores IPC semblent davantage influencés par la couverture des instances de domaine que par la qualité du plan, indiquant que les configurations comparées fournissent au moins des résultats comparables en termes de durée totale.

Finalement, en termes de couverture, TEP surpasse FAPE sur tous les domaines.

6 État de l'art

Les approches pour la résolution de problèmes de planification hiérarchique et temporelle peuvent être classées en fonction de leur efficacité à traiter les catégories de problèmes définies par Cushing [9].

La plus part des approches proposées peuvent résoudre des problèmes relevant des deux premières catégories grâce à différentes techniques. Une approche populaire consiste à adapter les planificateurs HTN non temporels pour gérer des formalismes temporels. Par exemple, SHOP (Simple Hierarchical Ordered Planner) [18] a été étendu pour devenir SHOP2 [2, 11] afin d'incorporer des actions avec des durées. D'autres avancées, comme GSCCB-SHOP2 [22], gèrent des contraintes temporelles et de ressources complexes. Un autre exemple est le planificateur SIADEX [1], qui applique une méthode de chaînage avant similaire à la planification classique, adaptée au contexte temporel. Il établit des méta-actions temporelles qui se lient séquentiellement pour élaborer un plan de solution. De plus, parmi les planificateurs s'attaquant aux deux premières catégories, on trouve le planificateur CHIMP [25]. CHIMP traduit les problèmes temporels et hiérarchiques en problèmes de satisfaction de contraintes pour dériver des plans de solution temporelle. Bien que ces planificateurs bénéficient de l'efficacité de la planification non temporelle, leurs méthodologies ont du mal à traiter pleinement la complexité de la planification temporelle, les rendant inadéquats pour les problèmes impliquant de la concurrence.

Les planificateurs capables de traiter la troisième catégorie de Cushing utilisent généralement des méthodes basées sur les timelines, surveillant le statut de chaque proposition au fil du temps. Parmi les exemples clés, on peut citer HSTS [16], plus tard affiné en EUROPA [3], un planificateur basé sur CSP. De plus, IxTeT [14] est indépendante du domaine et efficace pour gérer des timelines complexes. Sa dernière itération, FAPE [7], gère habilement l'ensemble des expressions de concurrence nécessaires.

7 Conclusion

Dans cet article, nous avons présenté TEP, une approche pour résoudre à la fois des problèmes hiérarchiques et temporels. Elle consiste à relaxer les problèmes temporels en problèmes non temporels afin d'appliquer des heuristiques de recherche développées pour résoudre des problèmes non-temporels. Nous avons comparé notre approche avec une approche fondée sur les timelines partageant la même expressivité en termes de catégories de Cushing. Nous avons montré que notre approche surpassait cette dernière en termes de temps d'exécution pour trouver une solution, et que les deux approches étaient comparables en ce qui concerne la qualité des plans solution (makespan).

Références

[1] Marc Asunción, Luis Castillo, Juan Fdez-Olivares, Óscar García-Pérez, Antonio González-Muñoz, and Francisco Palao. Siadex : An interactive knowledge-based planner for decision support in forest fire fighting. *AI Commun.*, 18 :257–268, 01 2005.

[2] Tsz-Chiu Au, Okhtay Ilghami, Ugur Kuter, J. William Murdock, Dana S. Nau, Dan Wu, and Fusun Yaman. Shop2 : An htn planning system. *J. Artif. Intell. Res.*, 20 :379–404, 2003.

[3] Javier Barreiro, Matthew Boyce, Minh Do, Jeremy Frank, Michael Iatauro, Tatiana Kichkaylo, Paul Morris, James Ong, Emilio Remolina, Tristan Smith, et al. Europa : A platform for ai planning, scheduling, constraint programming, and optimization, 2012.

[4] Patrick Bechon, Magali Barbier, Guillaume Infantes, Charles Lesire, and Vincent Vidal. Hipop : Hierarchical partial-order planning. In *STAIRS*, pages 51–60, 2014.

[5] Pascal Bercher, Gregor Behnke, Daniel Höller, and Susanne Biundo. An Admissible HTN Planning Heuristic. In *IJCAI*, pages 480–488, 2017.

[6] Pascal Bercher, Shawn Keen, and Susanne Biundo. Hybrid Planning Heuristics Based on Task Decomposition Graphs. In *SOCS*, pages 35–43, 2014.

[7] Arthur Bit-Monnot, Malik Ghallab, Félix Ingrand, and David E. Smith. FAPE : a Constraint-based Planner for Generative and Hierarchical Temporal Planning. *ArXiv, abs/2010.13121.*, 2020.

[8] M. Broxvall and P. Jonsson. Point algebras for temporal reasoning : Algorithms and complexity. *Artif. Intell.*, 149(2) :179–220, 2003.

[9] William Cushing. Evaluating Temporal Planning Domains. In *ICAPS*, pages 105–112, 2007.

[10] M. Fox and D. Long. PDDL2.1 : an extension to PDDL for expressing temporal planning domains. *J. Artif. Intell. Res.*, 20 :61–124, 2003.

[11] R. Goldman. Durative planning in htns. In *ICAPS*, pages 382–385, 2006.

[12] Daniel Höller and Pascal Bercher. Landmark generation in htn planning. In *AAAI*, pages 11826–11834, 2021.

[13] David Joslin and Martha E. Pollack. Least-cost flaw repair : A plan refinement strategy for partial-order planning. In *AAAI*, pages 1004–1009, 1994.

[14] Solange Lemai. *IXTET-EXEC : planning, plan repair and execution control with time and resource management*. PhD thesis, Institut National Polytechnique de Toulouse, 2004.

[15] David A. McAllester and David Rosenblitt. Systematic nonlinear planning. In *AAAI*, pages 634–639, 1991.

[16] Nicola Muscettola. HSTS : Integrating Planning and Scheduling. Technical report, Robotics Institute, Camegie Mellon University, 2013.

[17] Dana Nau, Yash Bansod, Sunandita Patra, Mark Roberts, and Ruoxi Li. Gtpyhop : A hierarchical goal+task planner implemented in python. In *HPlan Workshop (ICAPS)*, 2021.

[18] Dana S. Nau, Tsz-Chiu Au, Okhtay Ilghami, Ugur Kuter, J. William Murdock, Dan Wu, and Fusun Yaman. SHOP2 : an HTN planning system. *J. Artif. Intell. Res.*, 20 :379–404, 2003.

[19] Damien Pellier, Alexandre Albore, H. Fiorino, and R Bailon-Ruiz. HDDL 2.1 : Towards Defining an HTN Formalism and Semantics with Time. 2023. HPLlan Workshop (ICAPS).

[20] Damien Pellier and Humbert Fiorino. PDDL4J : a planning domain description library for Java. *J. Exp. Theor. Artif. Intell.*, 30(1) :143–176, 2018.

[21] Laurent Perron, Frédéric Didier, and Steven Gay. The cp-sat-lp solver. In *CP*, pages 3 :1–3 :2, 2023.

[22] Chao Qi, Dan Wang, Héctor Muñoz-Avila, and Peng Zhao. Hierarchical task network planning with resources and temporal constraints. *Knowledge-Based Systems*, 133 :17–32, 2017.

[23] Bernd Schattenberg. *Hybrid planning & scheduling*. PhD thesis, University of Ulm, Germany, 2009.

[24] Bernd Schattenberg, Julien Bidot, and Susanne Biundo. On the construction and evaluation of flexible plan-refinement strategies. In *German Conference on AI*, pages 367–381, 2007.

[25] Sebastian Stock, Masoumeh Mansouri, Federico Pecora, and Joachim Hertzberg. Online task merging with a hierarchical hybrid task planner for mobile service robots. In *IROS*, pages 6459–6464, 2015.

[26] Håkan L. S. Younes and Reid G. Simmons. VHPOP : versatile heuristic partial order planner. *J. Artif. Intell. Res.*, 20 :405–430, 2003.

Actes JIAF 2024

A more efficient and informed algorithm to check Weak Controllability of Simple Temporal Networks with Uncertainty

Ajdin Sumic[1] **Thierry Vidal**[1]

[1] Laboratoire Génie de Production - Université Technologique de Tarbes

asumic@enit.fr
tvidal@enit.fr

Résumé

Les réseaux temporels simples avec incertitude (STNU) sont un modèle bien connu basé sur les contraintes qui exprime des plans d'activités liées par des contraintes temporelles, chacune ayant des durées possibles sous la forme d'intervalles convexes. L'incertitude provient du fait que certaines de ces durées sont "contingentes", c'est-à-dire que l'agent qui exécute le plan ne peut pas décider de la durée réelle au moment de l'exécution. Pour vérifier que l'exécution satisfera toutes les contraintes, il existe trois niveaux de *contrôlabilité*, selon que l'agent observe ou non la durée réelle et à quel moment ; deux d'entre eux, la contrôlabilité forte et dynamique (SC/DC), se sont révélés à la fois utiles dans la pratique et prouvables en un temps polynomial. La troisième, la contrôlabilité faible (WC), suppose que les durées incertaines seront fixées par un "oracle" juste avant le début de l'exécution, et on suppose qu'elle est co-NP-complète. En outre, les algorithmes de vérification de la contrôlabilité sont des stratégies de propagation, qui présentent l'inconvénient habituel, en cas d'échec, de s'avérer incapables de localiser clairement les contingents qui expliquent la non-contrôlabilité. Cet article a trois contributions : (1) il justifie l'utilité de la WC dans les systèmes multi-agents (SMA) où un autre agent contrôle un contingent, et les agents se mettent d'accord juste avant l'exécution sur les durées ; il fournit un nouvel algorithme de vérification de la WC (2) dont la performance en pratique dépend de la structure du réseau, et semble être pseudo-polynomiale dans les réseaux faiblement connectés correspondant à des cas réalistes, et (3) qui fournit les cycles défaillants dans le réseau qui expliquent la non-WC : cela rendra donc possible dans les SMA de réparer les contingents identifiés par des négociations avec d'autres agents.

Abstract

Simple Temporal Networks with Uncertainty (STNU) are a well-known constraint-based model expressing plans of activities related by temporal constraints, each having possible durations in the form of convex intervals. Uncertainty comes from some of these durations being *contingent*, i.e., the agent executing the plan cannot decide the actual duration at execution time. To check that execution will satisfy all the constraints, three levels of *controllability* exist, depending on if and when the agent observes the actual duration; two of them, the Strong and Dynamic Controllability (SC/DC), have proven to be both useful in practice and provable in polynomial time. The third one, the Weak Controllabiilty (WC) assumes that uncertain durations will be set through some 'oracle' just before execution starts, and is conjectured to be co-NP-complete. Moreover, controllability checking algorithms are propagation strategies, which have the usual drawback, in case of failure, to prove unable to clearly locate the contingents that explain the non-controllability. This paper has three contributions: (1) it substantiates the usefulness of WC in multi-agent systems (MAS) where another agent controls a contingent, and agents agree just before execution on the durations; it provides a new WC-checking algorithm (2) whose performance in practice depends on the network structure, and seems to be pseudo-polynomial in loosely connected ones corresponding to realistic cases, and (3) which provides the failing cycles in the network that explain non-WC: that shall hence make it possible in MAS to repair the identified contingents through negotiations with other agents.

Keywords: Temporal constraints, Temporal uncertainty, Multi-agent planning, Graph-based algorithms, Explainable inconsistency

1 Introduction

Temporal Constraint Satisfaction Problems (TCSP) are constraint-based problem formulations that allow to represent and reason on temporal constraints. They are used

in a lot of domains, such as planning and scheduling (on which we will focus), supervision of dynamic systems, or workflow design. They are based on a graphical model, the reason why they are usually called Temporal Constraint Networks (TCN)[6] : variables/nodes are time-points for which one shall assign a timestamp. Constraints/edges express sets of possible durations relating them. A key issue is the ability to check the consistency of the whole network. The simplest class, called the Simple Temporal Network (STN), arises when they have only binary constraints with only convex intervals of values (no disjunctions). One of the main strengths of this restricted, but often sufficient in practice, model is that consistency checking is made through a polynomial propagation algorithm (the Floyd-Warhsall reduction) and provides a complete *minimal* network in which all inconsistent values are removed. This minimal network can be passed on to some execution manager that can take any value on the domain of the first activity to schedule, and repropagate, and so on iteratively.

A well-known extension of STNs that handles uncertainties, called STNU (Simple Temporal Network with Uncertainty), has been proposed by [13]. An STNU contains uncertain (*contingent*) durations between time-points which means the effective duration is not under the control of the agent executing the plan, which is useful for addressing realistic dynamic and stochastic domains.

In STNUs, the notion of temporal consistency has been redefined in the form of *controllability* : an STNU is controllable if there exists a strategy for executing the schedule whatever the values taken by the contingent durations. In [13] the authors introduce three levels of controllability that express how and when the uncertainties are resolved : the Weak Controllability (WC) proves a solution exists for any possible combination of contingent values. Which requires that some 'oracle' provides those values before the timing of controllable time-points is decided ; the Dynamic Controllability (DC) is more demanding as it assumes that at execution time a strategy can be built based on past observations only, thus whatever the contingent durations still to be observed ; last, the Strong Controllability (SC) is even more demanding as it enforces that there is one unique assignment of controllable timepoints values, which defines a static control strategy that works whatever the contingent durations will be at execution time. WC has often appeared to be unrealistic in dynamic applications that assume full progressive observability at execution time. DC answers to that and looks more relevant, and has received much attention in previous works. Comparatively, SC is usually too demanding unless under partial or non-observability, or when some strict commitment must be made on the execution schedule timing for some client.

Previous works prove that SC and DC can be resolved with specially designed propagation-based algorithms that run in polynomial time [10, 3, 13]. WC is more complex because it is supposed to be a co-NP-complete problem, and only exponential algorithms exist to check WC [5, 13]. This is another reason why WC has not received as much attention as DC and has been disregarded [2, 13].

Anyway, nowadays, STNUs appear to be useful in even more new application domains in which WC would actually be relevant, especially when it comes to multi-agent task management. This paper aims to address it, discussing more precisely how and when one could need it. As we will argue some of those applications definitely need more efficient algorithms : this paper aims at providing a new algorithm for checking WC in STNUs that is much more efficient in practice and even experimentally pseudo-polynomial under some conditions,i.e., behaves like a polynomial time algorithm. Contrary to the complete propagation algorithms proposed for SC and DC, our algorithm maintains and reasons only on the input constraints, which form paths in the networks. As in any graph, such paths join and form cycles. We prove that it is possible to check the global weak controllability by locally checking the elementary cycles of an STNU.

Moreover, the algorithm is also capable of diagnosing the source of uncontrollability of a non-WC STNU, i.e., to detect the set of constraints that makes the STNU not weakly controllable. This explainable issue is a problem recently addressed in STNU for DC in [9]. This is important for the executor manager in order to repair the schedule [2, 1, 12]. The paper is organized as follows : Section 2 first recalls the necessary background on STNU. Section 3 then discusses the usefulness in practical applications of WC. Then, we prove in Section 4 how local controllability on cycles is equivalent to global WC. Next, Section 5 will present how to locally check WC, and Section 6 will present the new algorithm for globally checking WC. Some experimental evaluation will be displayed in Section 7 before concluding our contribution with some prospects.

2 Background

2.1 STNU

A Simple Temporal Network (*STN*)[6] is a pair, (V, E), where V is a set of time-points v_i representing event occurrence times, and E a set of temporal constraints between these time-points, in the form of convex intervals of possible durations. A reference time-point v_0 is usually added to V which is the 'origin of time', depending on the application (might be e.g. the current day at 0 :00). The goal is to assign values to time-points such that all constraints are satisfied, which is equivalent to assigning a value to each constraint in its interval domain.

An STN with Uncertainty (*STNU*) is an extension in which one distinguishes a subset of constraints whose values are parameters that cannot be assigned but will be

observed. Before defining the model and the controllability levels, we introduce the usual basic notations :

— minimal bounds $l_{ij} \in \mathbb{R} \cup \{-\infty\}$,

— maximal bounds $u_{ij} \in \mathbb{R} \cup \{+\infty\}$,

— $\prec$ is the usual qualitative precedence relation between time-points : $v_i \prec v_j$, i.e., v_i happens before v_j. A numerical constraint in the STNU from v_i to v_j has its lower bound $l_{ij} \geq 0$ iff $v_i \preceq v_j$.

Definition 1 *(**STNU**) An STNU is a tuple (V, E, C) with :*

— *V a set of time-points $\{v_0, v_1, \ldots, v_n\}$, partitioned into controllable (V_c) and uncontrollable (V_u) ;*

— *v_0 the reference time-point : $\forall i, v_0 \preceq v_i$*

— *E a set of requirement constraints $\{e_1, \ldots, e_{|E|}\}$, where each e_k is of the form $[l_{ij}, u_{ij}]$ with, $v_i, v_j \in V$.*

— *C a set of contingent constraints $\{c_1, \ldots, c_{|C|}\}$, where each c_k is of the form $[l_{ij}, u_{ij}]$ with, $v_i \in V_c$, $v_j \in V_u$, and necessarily $v_i \preceq v_j : 0 \leq l_{ij} \leq u_{ij}$.*

Intuitively, controllable time points ($\mathcal{V}_c$) are moments in time to be decided by the scheduling agent, which is trying to satisfy all the requirement constraints (E) under any possible instantiation of the contingent constraints (C). Moreover, it's semantically impossible to have a contingent duration between two unordered time-points. Figure 1a is the graphical representation of an STNU.

In addition, an STN (and hence an STNU too) has an equivalent *distance graph* representation [6, 7]. Each constraint of the form $[l, u]$ between v_i and v_j would be represented as $v_i \xrightarrow{[l,u]} v_j$ in the STN, or equivalently through two corresponding edges in its distance graph : $v_i \xrightarrow{u} v_j$ and $v_j \xrightarrow{-l} v_i$.

2.2 Three levels of controllability

In a STN, the notion of consistency is applied to define that there exists a solution that satisfies the constraints. In a STNU, the consistency has been redefined in three levels of controllability that we are going to recall before focusing on the problem of Weak Controllability.

Definition 2 *(**Schedule**) A schedule δ of an STNU $\mathcal{X}$ is an assignment of the controllable time-points $\delta = \{\delta(v) \mid v \in V_c\}$*

Definition 3 *(**Situation and Projection**) Given an STNU $\mathcal{X}$, for all $k = 1 \ldots |C|, c_k = [L_k, U_k], \Omega = [L_1, U_1] \bigtimes \ldots \bigtimes [L_{|C|}, U_{|C|}]$ is the domain of all possible **situations** of $\mathcal{X}$.*
*A tuple $\omega = \langle \omega_1 \in [L_1, U_1], \ldots, \omega_C \in [L_{|C|}, U_{|C|}] \rangle \in \Omega$ is called a complete situation of $\mathcal{X}$ and $\mathcal{X}_\omega$ the **projection** of $\mathcal{X}$, is an STN where $\mathcal{X}_\omega = (V, E \cup C')$ with $C' = \{[\omega_k, \omega_k]\} \mid c_k \in C$*
*Last, a schedule δ_ω which satisfies all the constraints in $\mathcal{X}_\omega$ is called a **solution** of $\mathcal{X}_\omega$.*

Intuitively, the projection X_ω is a problem without uncertainty in which each contingent duration has been fixed to a given value. Hence X_ω is an STN.

Definition 4 *(**Weak Controllability (WC)**) An STNU $\mathcal{X}$ is **weakly controllable** iff $\forall \omega \in \Omega, \exists \delta$ such that δ is a solution of $\mathcal{X}_\omega$.*

This definition implies that the scheduler is clairvoyant, i.e. there is an 'oracle' that predicts the future and communicates the durations of the contingents to the scheduler before execution time. In other words WC requires all projections to be consistent independently from one another.

The two other controllablity levels impose more requirements, first (DC) removing the clairvoyance assumption and demanding that each assignment of a controllable time-point only depends on the past observations and not the future ones, and even (SC) demanding that the (unique) schedule is totally independent from any observation [13].

As said before, propagation-based checking algorithms exist for SC and DC [13][10][3]. But not for WC checking, which is conjectured to be co-NP-complete ; anyway, considering only the bounds is enough to verify any level of controllability in STNU : *it is enough to consider projections on the bounds to be sure that all intermediate projections will be consistent as well*[13]. The original algorithm to check WC checks the consistency of all $2^{|C|}$ STNs obtained by replacing the contingents with one of their bounds (upper or lower), which is an exponential algorithm.

3 Weak Controllability

In this section, we will argue that WC may be more relevant than DC and SC for some applications, while checking WC has been somehow left apart in the literature so far. This is mainly because WC assumes that the duration of the contingent activities is revealed when execution starts. Somehow, it requires an entity capable of predicting and sharing these durations with the scheduler agent before the actual occurrence of such activities.

However, in classical planning and scheduling applications, uncertainties come from external causes ; they are somehow 'controlled by Nature', and can only be observed at their time of occurrence. Which is why DC/SC are more relevant in practice. For instance, when one starts cooking a steak, they do not know how long it will take and will observe it once the steak is actually well done.

However, in many domains (logistics, transport, services), they have a first strategic phase that builds a plan without assigning all real resources ; a more precise tactical version will do that later. For instance, in a health service

or in a construction site, one needs a weekly plan for visiting patient rooms, or for the construction tasks, but the assigned teams (number of people, skills) are not known, which results in flexible and large enough intervals of possible durations. The precise assignment is only known each day for the next day, which allows to get a more precise plan just before execution, which is exactly the definition of WC.

Moreover, uncontrollable durations also appear in multi-agent systems, when some event might be owned by another agent instead of Nature. Still, in such contexts, collaboration may rely on the timely communication of effective durations at execution time. But in some application contexts, it might be the case that the interval of activity's duration represents the degree of freedom, and hence the *flexibility*, that some agent wishes to keep as long as possible to be more robust. But at the same time, actual durations must be set and communicated just before execution to the other agents that depend on it for better coordination. Thus, some tasks might be controllable (requirement) for one agent but uncontrollable for another one (contingent). That may arise, for instance, in collaborating hospital services that share common resources : they plan in advance their weekly operations with maximum flexibility but must set and confirm to others their own schedules each day for the next one. At this point, all shared events that are not controlled by Nature become requirement constraints at execution time. In a setting where no events are controlled by Nature, checking WC instead of DC/SC enables the agents to be more robust through least-commitment strategies, retaining flexibility as long as possible and hence computing more optimal solutions for their schedule.

One should notice that even though some approaches exist to deal with multi-agent STNUs [4], those actually consider a global plan shared among agents in which contingent constraints are still controlled by Nature. Considering inter-dependent STNUs, with shared activities that are controllable by one agent and only observed by others, still need to be investigated. That is the topic of future research for which this paper somehow is a cornerstone.

4 From local controllability to global controlability

4.1 Updated STNU graphical model

A starting point for resolving the issue of Weak Controllability is to add some features to STNU's graphical representation and adapt the model accordingly. Nodes in an STNU will not only be divided between controllable and uncontrollable time-points (the ending time-points of some contingent constraint), but we also need to identify among them *divergent* time-points and *convergent* ones.

Definition 5 ***(Convergent and Divergent time-points)*** *In a STNU* $\mathcal{X} = (V, v_0, E, C)$ *:*

- $v_i \in V$ *is called a* ***divergent*** *time-point iff* $\exists j, k, i \neq j \neq k$ *with* $v_i \rightarrow v_j \in E \cup C$ *and* $v_i \rightarrow v_k \in E \cup C$ *;*
- $v_i \in V$ *is called a* ***convergent*** *time-point iff* $\exists j, k, i \neq j \neq k$ *with* $v_j \rightarrow v_i \in E \cup C$ *and* $v_k \rightarrow v_i \in E \cup C$ *;*
- V_{dv} *is the set of divergent time-points with* $V_{dv} \subseteq V$ *;*
- V_{cv} *is the set of convergent time-points with* $V_{cv} \subset V$ *;*

Intuitively, a divergent node has at least two outgoing edges in the input graph modeling the STNU, and a convergent one has at least two incoming edges.

Please note that if a contingent link is necessarily a directed edge (implicit precedence), a requirement link may be a non directed edge : e.g. $v_i \xrightarrow{[-5,10]} v_j$, imposing some constraint on the temporal distance between the time-points but allowing any order between them at execution time. Hence, v_i or v_j in this example may be considered as a divergent time-point, depending on the order between them in the input link defined at the design level (here the link will be an outgoing edge from v_i). As it will be shown in the next sub-section, it will only change the begin and end points of the two paths that form a cycle, but the cycle will still be the same.

In addition, $V_{dv} \cap V_{cv}$ may not be void, i.e. any $v \in V$ may be convergent, divergent, convergent *and* divergent, or neither convergent nor divergent (we shall call the latter *serial* nodes). In fact, these definitions are orthogonal to the distinction between controllable and contingent time-points, meaning that a controllable time-point might be convergent or divergent, etc., and a contingent one alike (since a contingent and a controllable constraint can converge on the same point, but not two or more contingent : an assumption we make as it's obvious such STNU is not controllable).

Of course, by definition, v_0 cannot be a convergent time-point, but usually, a divergent one, even though the model does not enforce it, as v_0 is used to define the absolute time of any time-point v_i as a constraint between v_0 and v_i.

One can see that such a characterization is very similar to what is done in flow networks [8], but there the problem is to check that the sum of labels (capacities) that converge on a point equals the sum of the labels that exit that node, while here we will instead use this distinction to look for cycles, i.e. identify that two *paths* diverging from one node and reunite in a convergent node have compatible overall durations whatever values the contingent in those paths will take, which is a local WC condition.

In figure 1(a), we present an STNU as defined in definition 1 augmented by definition 5. Figure 1(b) exhibits an alternative way to represent the STNU that will be explained in the next sections.

4.2 Weak controllability on cycles

First, checking WC of $\mathcal{X}$ is straightforward when there is no convergent time-point. Thus $\mathcal{X}$ is a multi-linear graph,

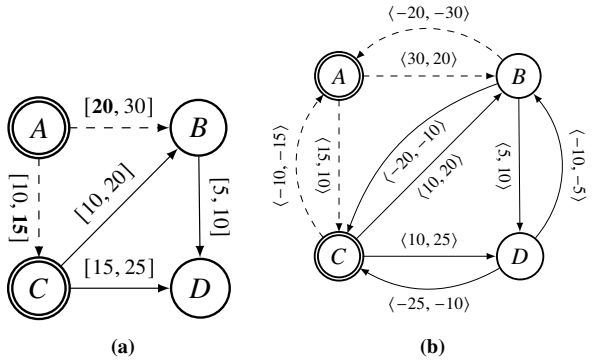


Figure 1 – An STNU is presented in (a) where time-point A plays here the role of the reference point v_0, $V_{dv} = \{A, C\}$ (doubly circled nodes) and $V_{cv} = \{D, B\}$. The links represented by the dotted arrows are contingent constraints. Hence, C and B are uncontrollable time points, while A and D are controllable time points. The proposed STNU is not weakly controllable due to the projection highlighted in bold on the contingent constraints $A \overset{[10,15]}{\dashrightarrow} C$ and $A \overset{[20,30]}{\dashrightarrow} B$ that violates the synchronization on B. We show in (1b the controllable bounds graph of the STNU.

i.e. only divergent simple timelines on which it is always possible to just wait for the observation of an uncontrollable time-point to then decide the activation of the next controllable point, which builds up both the projection and the schedule overtime. In that case, DC and WC are actually equivalent and are always satisfied. When there is at least one convergent point (which entails that there is at least one divergent point), that means there are **paths** that diverge at some point and merge at another point.

Definition 6 ***(Path)*** *A path ρ in $\mathcal{X}$ is a sequence of time-points $v_1, \ldots, v_p$ such that $\forall i = 1 \ldots p-1, v_i \rightarrow v_{i+1} \in E \cup C$ or $v_{i+1} \rightarrow v_i \in E \cup C$, $v_1 \in V_{dv}$ and $v_p \in V_{cv}$.*

One can see that in that definition, we allow a path to follow edges in the graph in any direction, thus ensuring that all possible cycles in the STNU will not be forgotten. For example, in Figure 1(a), considering divergent node C and convergent node D, there is obviously a path C-B-D, but C-A-B-D should also be considered, which is equivalent to stating that there is a path in the corresponding distance graph. Somehow, Figure 1(b), if one disregard, for now, the labels, can be viewed as such a distance graph : the sequence of directed links C-A-B-D is apparent.

Anyway, it is easy to see that any cycle of initial constraints in the input STNU can be defined as a pair of distinct paths as defined above with the same starting $v_1 \in V_{dv}$ and ending $v_p \in V_{cv}$ time-points. It is a peculiar way of defining those cycles that will be useful for our algorithm.

Definition 7 ***(WC Divergent Cycle)*** *A **divergent cycle** $\mathcal{M}$ is a pair (ρ_1, ρ_2) such that ρ_1 and ρ_2 are two **paths** starting at the same divergent time point $v_d \in V_{dv}$ and ending at the same converging time point $v_c \in V_{cv}$, where v_d, v_c are the only common time points in ρ_1, ρ_2, i.e. $\forall v_i \in \rho_1, v_c \neq v_i \neq v_d : v_i \notin \rho_2$.*
A cycle $\mathcal{M}$ is said to be weakly controllable if the sub-STNU restricted to the set of time-points and constraints involved in both paths is WC.

For example, Figure 3b shows a cycle with a first path represented by C-D and a second one by C-B-D.

Then an STNU is WC if and only if all divergent cycles are WC. We will present this result in two steps, first defining a local property that might be checked for a divergent node, and then generalizing to all divergent nodes ; which will mainly be useful for better explaining our algorithm.

Definition 8 ***(Local divergent-WC)*** *Let $\mu(v_d) = \{\mathcal{M}_1, \ldots, \mathcal{M}_n\}$ the set of all cycles starting from $v_d \in V_{dv}$, converging on a set of convergent nodes of V_{cv} that are necessarily ordered (topological ordering) after v_d in the STNU $\mathcal{X}$. We say that $\mathcal{X}$ is **locally divergent-WC on** v_d iff $\forall\, \mathcal{M}_i \in \mu(v_d)$, $\mathcal{M}_i$ is weakly controllable*

For example, Figure 3a shows the cycles starting from the divergent time-point A.

The local divergent-WC does not imply WC, as the corresponding sub-STNU might contain other divergent nodes.

Theorem 1 ***(Global controllability)*** *$\mathcal{X}$ is Weakly controllable (WC) iff $\forall v_d \in V_{dv}$, $\mathcal{X}$ is locally divergent-WC on v_d*

Theorem 1 implies that checking the local divergent-WC property of all the divergent nodes of an STNU is enough to verify the WC.

Proof : The forward implication is straightforward to prove : if there is a divergent node for which at least one divergent cycle, which is a sub-STNU, is not WC, that means there is at least one projection for which there is no consistent local schedule, then the global STNU will not be WC.

For the reverse implication, suppose the global STNU is not WC. Then there is at least one projection for which the corresponding STN is inconsistent ; that is equivalent to having a negative cycle somewhere in that STN ; and that negative cycle necessarily relates time-points that form a divergent cycle in the STNU, which in turn is not WC following Definition 7 [13].

5 Local Weak Controllability

In this section, we show how to check the WC of a cycle by exploiting the convexity of the problem : as proved in [13], considering the lower and upper bounds of the contingents is enough to check WC for STNU.

Definition 9 ***(Controllable Bounds)*** *Given an STNU $\mathcal{X} = (V, v_0, E, C)$, and $t_{ij} \in E \cup C$. The* ***controllable bounds*** *of t_{ij}, denoted Π_{ij}^{ctl}, is the tuple of discrete values such that :*

$$\Pi_{ij}^{ctl} = \langle min_{ij}^{ctl}, max_{ij}^{ctl} \rangle$$

where, min_{ij}^{ctl} and max_{ij}^{ctl} respectively represent the minimal and maximal duration that can be guaranteed for t_{ij}.

It is easy to see that any requirement constraint $e_i = [l_{ij}, u_{ij}]$, has a minimal and maximal duration that can be guaranteed with $min_{ij}^{ctl} = l_{ij}$ and $max_{ij}^{ctl} = u_{ij}$.

A contingent constraint is different due to its uncontrollable duration. Still, some guarantee can be extracted on its execution. Indeed, the duration of any $c_i \in C$ is guaranteed, for the minimum, to be at most equal to U_{ij} and, for the maximum, at least equal to l_{ij}. Thus, we have : $min_{ij}^{ctl} = u_{ij}$ and $max_{ij}^{ctl} = l_{ij}$. Intuitively, min_{ij}^{ctl} and max_{ij}^{ctl} represent the two worst-case scenarios of a contingent duration. In the following, we generalize Π_{ij}^{ctl} as follows :

$$\Pi_{ij}^{ctl} = \begin{cases} \langle u_{ij}, l_{ij} \rangle \text{ iff } t_{ij} \in C \\ \langle l_{ij}, u_{ij} \rangle \text{ iff } t_{ij} \in E \end{cases} \quad (1)$$

Then, from Equation 1, it is actually possible to represent an STNU $\mathcal{X}$ in terms of its controllable bounds graph denoted $\Pi_{\mathcal{X}}^{ctl}$, which is shown in Figure 1 (b). This graph considers each original constraint and its inverse. A requirement constraint $e_i = [l_{ij}, u_{ij}]$, equivalently $(v_j - v_i) \geq l_{ij}$ and $(v_j - v_i) \leq u_{ij}$, has an inverse constraint $e_i' : (v_i - v_j) \leq -l_{ij}$ and $(v_i - v_j) \geq -u_{ij}$ equivalently represented as $e_i' = [-u_{ij}, -l_{ij}]$. The same transformation is applied to contingent constraints. Please note that we introduce this new graph representation instead of reusing the well-known *distance graph* of an STNU as it's simpler. We also believe this new representation could pave the way for more efficient algorithms for DC.

From this transformation, it's possible to compute the controllable bounds of a path ρ composed of constraints in $E \cup C$ by controllable bounds propagation from v_1 to v_p.

Definition 10 ***(Controllable Path Bounds)*** *Let ρ be a path in $\Pi_{\mathcal{X}}^{ctl}$, with $v_1, \ldots, v_p$ the sequence of time-points of ρ. The* ***controllable path bounds*** *denoted Π_{ρ}^{ctl} is defined as follows :*

$$\Pi_{\rho}^{ctl} = \langle \sum min_{ij}^{ctl}, \sum max_{ij}^{ctl} \rangle$$

From this point, it's possible to check the WC controllability of a cycle $\mathcal{M} = (\rho_1, \rho_2)$ through the controllable paths bounds $\Pi_{\rho_1}^{ctl}$ and $\Pi_{\rho_2}^{ctl}$. Indeed, we need to guarantee that the minimum controllable duration of ρ_1 is lesser or equal to the maximum controllable duration of ρ_2 and vice-versa. Intuitively, if the condition is not satisfied, then there exists a projection of $\mathcal{M}$ such that ρ_1 and ρ_2 cannot synchronize on v_p as Π_{ρ}^{ct} represent the worst-case scenarios of ρ. the worst-case scenarios for synchronizing two paths are those where, for one path, its contingents take the minimal bound l_{ij} and the maximal bounds u_{ij} for the second path.

Theorem 2 ***(Cycle WC property)***
Given a cycle $\mathcal{M} = (\rho_1, \rho_2)$ and the controllable paths bounds $\Pi_{\rho_1}^{ctl} = \langle min_{\rho_1}^{ctl}, max_{\rho_1}^{ctl} \rangle$ and $\Pi_{\rho_2}^{ctl} = \langle min_{\rho_2}^{ctl}, max_{\rho_2}^{ctl} \rangle$. $\mathcal{M}$ is weakly controllable iff :

$$(min_{\rho_1}^{ctl} \leq max_{\rho_2}^{ctl}) \wedge (min_{\rho_2}^{ctl} \leq max_{\rho_1}^{ctl}) \quad (2)$$

Proof : The forward implication is straightforward to prove : if $\mathcal{M}$ is WC, then whatever the bounds of the contingents in $\mathcal{M}$, there always exists a schedule that satisfies the constraints of $\mathcal{M}$. Let's suppose Equation 2 is false. It means there exists a projection of ρ_1 and ρ_2 such that the synchronization on v_p is impossible and forms a negative cycle. Thus, such a projection is inconsistent, and $\mathcal{M}$ is not WC. This is impossible as all projections of $\mathcal{M}$ are consistent.

For the reverse implication, let's suppose $\mathcal{M}$ is not WC, but Equation 2 is satisfied. Then, it means that the projections of the two worst-case scenarios of $\mathcal{M}$ are consistent as there exists at least one schedule that guarantees the synchronization on v_p. Thus, any projection satisfies the synchronization on v_p. This is not possible as $\mathcal{M}$ is not WC, which implies the sub-STNU M has a negative cycle [13].

For simplicity, we denote M^{ctl} a worst-case scenario of $\mathcal{M}$. A running example would be considering the network on the left of Figure **??** as an STNU. The controllable bounds are : $\{30, 20\}$ for $A \xrightarrow{[20,30]} B$, $\{15, 10\}$ for $A \xrightarrow{[10,15]} C$, and $\{10, 20\}$ for $C \xrightarrow{[10,20]} B$. Only one cycle exists that is formed by the two paths (ρ_1, ρ_2) from A to B, where $\Pi_{\rho_1}^{ctl} = \{30, 20\}$ and $\Pi_{\rho_2}^{ctl} = \{25, 30\}$, which does not satisfy Equation 2 as $min_{\rho_2}^{ctl} > max_{\rho_1}^{ctl}$.

6 The WC-Checking algorithm

6.1 Description of the algorithm

In this section, we present the new WC-checking algorithm for STNU by finding and checking its cycles. The algorithm comprises two parts : the first one finds the cycles of a divergent time-point, and the second one checks those cycles. In the following, we formalize the basic structures for the WC checking algorithm given an STNU $\mathcal{X}$:

— a **projection path** p, is a path $\langle \alpha_p, C_p, V_p \rangle$ in $\Pi_{\mathcal{X}}^{ctl}$ from v_d to v_m, where α is the minimal or maximal controllable bound of Π_p^{ctl} obtained by propagation. C_ρ is the set of contingent constraints of p ($C_p \subseteq C$),

and V_p the set of time-points of p ($V_p \subseteq V$). Please note that p is the definition of a path from an algorithm point of view, while in reality, p and ρ are equivalent.

— $P(v_d)$ is a set of **projection paths** $P(v_d) = \{p_i, \ldots, p_m\}$ that start at a divergent time-point v_d and still have to converge on a common v_c.

— the **minimal divergent cycles** $D_{min}(v_d)$ is a mapping of convergent time points v_c to a set of projection paths ($P^{min}_{v_c}$) that converge from v_d to v_c such that $\forall p \in P_{v_c}, \alpha_p = min^{ctl}_p$.

— the **maximal divergent cycles** $D_{max}(v_d)$ is a mapping of convergent time points v_c to a set of projection paths ($P^{max}_{v_c}$) that converge to v_c from v_d such that $\forall p \in P_{v_c}, \alpha_p = max^{ctl}_p$.

We introduce in Algorithm 1 the *findDivergentCycles* algorithm in charge of finding the cycles of a divergent time-point v_d. To avoid going through all possible paths in the controllable bounds graph $\Pi^{ctl}_{\mathcal{X}}$, we prune the number of paths in two ways :

— We first add the notion of *rank*, which was not formally defined in our model as it was not needed but is common in qualitative temporal networks : it is possible to define a partial order of all time-points with regard to the precedence relation ; $rank(v_0) = 0$, then for all v_i such that $v_0 \preceq v_i \in E \cup C$ and there is no v_j such that $v_0 \preceq v_j \in E \cup C$ and $v_j \preceq v_i \in E \cup C$, $rank(v_i) = 1$, and so on and so forth.

— Using that rank, a forward search is then applied by ordering the time-points through a topological ordering algorithm from v_0 (rank 0). This enables us to avoid any time-point v_i with a lower rank than the current divergent time-point v_d.

— A distinction between the minimal and maximal controllable bounds of a path. We apply two forward searches : one that computes the paths with only the maximal controllable bound and one with the minimal controllable bound. This allows us to prune the paths that converge to any convergent time-point to keep only stricter ones. For example, it is easy to see that for two paths p_i and p_j such that $C_{p_i} = C_{p_j} = \{\emptyset\}$ (only requirement constraints) p_i is stricter than p_j if $min^{ctl}_{p_i} > min^{ctl}_{p_j}$ (respectively, $max^{ctl}_{p_i} < max^{ctl}_{p_j}$). Hence, it's useless to consider further p_j as p_i is a stricter path, and only p_i is kept in $D_{min}(v_d)$ or $D_{max}(v_d)$ depending on the computed controllable bound. This also holds for a path p_j such that $C_{p_j} \neq \{\emptyset\}$ (with contingent constraints). However, when C_{p_i} and C_{p_j} are not empty, it's impossible to apply these rules as it might result in removing an inconsistent cycle in the graph. Suppose we have the minimal controllable bounds of p_i and p_j and the maximal controllable bounds of a path p_k such that the tuple (p_j, p_k) is the only one to form a cycle M^{ctl}. Then, if ρ_i is stricter than p_j and p_j is not kept, M^{ctl} will never be checked likewise for the WC of $\mathcal{X}$. Therefore, both p_i and p_j must be kept in $D_{min}(v_d)$. This is actually the reason why full reduction of intervals through the intersection of different edges is not possible, and hence, a polynomial time algorithm cannot be found, unlike DC and SC.

From lines 1 to 3, we initialize the maps $D_{max}(v_d)$ and $D_{min}(v_d)$, and the set of paths P. Then, the core of the algorithm (lines 5-16) propagates the paths in P to find and keep all stricter paths of v_d in $D_{max}(v_d)$ until P = $\{\emptyset\}$. In fact, in line 14, we also update P(v_d) and P_{v_j} by removing the paths that are not stricter anymore. A second forward search is done for $D_{min}(v_d)$ where P(v_d) is reset. Once the forward searches are over, the maps $D_{max}(v_d)$ and $D_{min}(v_d)$ contain all the restrictive paths from v_d to a convergent time-point v_c. Then, we execute the *checkCycles* algorithm (see Algorithm 2) in charge of checking the WC of the cycles of v_d. This algorithm is trivial as it simply searches and checks for each v_c in $D_{max}(v_d)$ and $D_{min}(v_d)$ all the pairs of paths (p_{min}, p_{max}) that converge on v_c and form a cycle $\mathcal{M}^{ctl}$ where $V_{p_{min}} \cap V_{p_{max}} = \{v_d, v_c\}$.

Algorithm 1: findDivergentCycles algorithm

Input: v_d :(time-point), $\Pi^{ctl}_{\mathcal{X}}$ **: (graph), rank : map**
Output: Boolean

1 $D_{min}(v_d) = \{\}$
2 $D_{max}(v_d) = \{\}$
3 $P(v_d) = [\langle 0, [], [v_d]\rangle]$
4 `A first forward search for` D_{max}
5 **while** *P not empty* **do**
6 p = $P(v_d)[0]$ `p is removed in` $P(v_d)$
7 **for** *each child* v_j *of* $v_m \in V_p$ *with rank*(v_j) $\geq$ *rank*(v_d) *and* $v_j \notin V_p$ **do**
8 p = propagateMaxPath($\Pi^{ctl}_{\mathcal{X}}$, p, max^{ctl}_{mj})
9 **if** v_j *is a convergent time point* ($v_j \in V_c$) **then**
10 **if** v_j *not in* $D_{max}(v_d)$ **then**
11 add $v_j \rightarrow [p]$ in $D_{max}(v_d)$
12 **else**
13 **if** *p is a restrictive path in* P_{v_j} **then**
14 add p to P_{v_j} and to $P(v_d)$
15 **else**
16 add p to P
17 `A second forward search for` $D_{min}(v_d)$
18 **return** checkCycles($D_{max}(v_d)$, $D_{min}(v_d)$)

Finally, Algorithm 3 presents the *WC-checking* algorithm that, for a given STNU $\mathcal{X}$, computes its controllable bounds graph $\Pi^{ctl}_{\mathcal{X}}$ (line 1), determines the topological ordering of

Algorithm 2: checkCycles algorithm

Input: $D_{max}(v_d)$, $D_{min}(v_d)$
Output: Boolean

```
1 for each v_c → P^min_{v_c} in D_min(v_d) do
2     for each p_min in P^min_{v_c} do
3         for each p_max in P^max_{v_c} in D_max(v_d) do
4             if (p_min, p_max) is of the form M^ctl then
5                 if α_{p_min} > α_{p_max} then
6                     return False   Or the cycle
7 return True
```

the time-points (line 2), and find and check the cycles of each divergent time-point in V_d. We inform the readers that the algorithm does not need to check the pseudo-controllability of Morris [11] as long as the constraint bounds l_{ij}, u_{ij} are as follows : $l_{ij} \neq -\infty$ and $u_{ij} \neq +\infty$.

We show, in a simplified manner, the execution of our WC-Checking algorithm in Figure 3. The order from the ranking is $A \rightarrow C \rightarrow B \rightarrow D$ and is used in the forward searches for finding and checking the cycles from A and C. We show in Figure 2 a running example of our algorithm with the divergent time-point A. Please note that for the sake of this paper, we simplified the example.

Algorithm 3: WC-Checking algorithm

Input: $\mathcal{X}$: STNU(V,v_0, V_c,V_d,E,C)
Output: Boolean

```
1 Π^ctl_X = getDistanceGraph(X)
2 rank = orderFromRank(X)
3 for each v_d in V_d do
4     if findDivergentCycles(v_d, Π^ctl_X, rank ) ==
        False then
5         return False  Or non-WC cycles of v_d
6 return True   Or all non-WC cycles
```

6.2 Features and Complexity

The previous section presented the new WC-checking algorithm for STNU that checks WC by checking the internal cycles of an STNU. The particularity of this approach easily allows the algorithm to return the set of negative cycles of a non-weakly controllable STNU (see Algorithms 2, 3). This feature is important for explainability, as explained in Section 1. Moreover, the divergent time-points are independent, which makes parallelization over the divergent time-points possible. In addition, the pseudo-controllability step is not required for constraint bounds with finite values (no constraints with a $+\infty$ or $-\infty$ for bounds). Thus, the algorithm can be executed incrementally by checking only divergent time points of the same rank or lesser (topological ordering) than the starting time point of the newly added constraint is enough due to divergent time-points being independent. However, it's not optimal as it might check cycles that are not linked to the newly added constraint. The drawback of the algorithm is that the minimal network of weakly controllable STNUs is not computed.

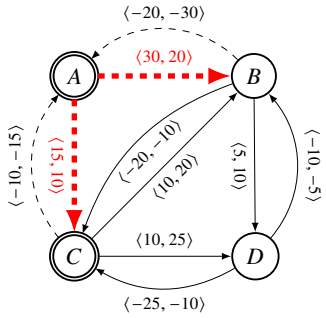
The temporal complexity of the algorithm depends on the number of cycles to check, which is related to multiple parameters such as the number of contingents, the number of divergent time-points, and the number of successors per divergent time-point. For a complete graph, the algorithm is exponential, and its complexity is not better than the original algorithm ($2^{|C|}$). However, our interest lies in realistic graphs where the density of the graph is low. Thus, in the next section, we will compare our algorithm (new_WC) with the original WC-checking algorithm (old_WC) (see Section 2) by restricting the parameters. We will also compare it with the Floyd-Warshall algorithm (APSP) to compare the behavior of our algorithm to see if a polynomial behavior is possible when parameters are restricted enough. Please note that the APSP cannot check the WC of STNU.

7 Experiments

In order to empirically test the effectiveness of the proposed algorithm, we consider the time from the start of the algorithm execution until it has finished all computation (not simply after finding an STNU to be weakly controllable or not). The benchmark comes from a random generator we implemented that can randomly generate sparse STNUs. It creates an STNU in the form of a complete directed acyclic graph (DAG), from which we randomly and remove a number of edges depending on parameters : the number of time points n, the rate of divergent time points r_d and the number of successors n_c, and the rate of contingency r_c.

All the experiments have been performed on a machine equipped with an Intel Core processor : 11th Gen Intel(R) Core(TM) i7-11850H @ 2.50GHz 2.50 GHz. We used a time/memory limit of 10 minutes/4GB and sequential, single-core computation for the sake of the experiments.

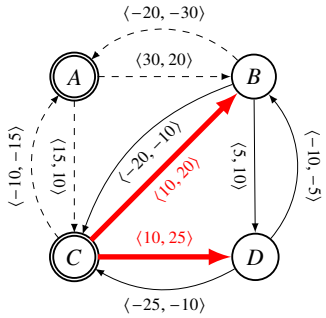
We experiment under different settings : n = $\{20, 50, 100, 200, 500, 1000\}$, $r_d = \{0.1, 0.2, 0.3\}$ meaning 10 to 30% of divergent time-points, $r_c = \{0.2, 0.3\}$, and $n_c = 3$. For each combination of parameters, we generate 20 STNUs and compute the average execution time. We show in Figure 4a that, in general, our algorithm clearly outperforms the *old-WC* algorithm and is slightly worse than the APSP algorithm up to 20% of contingent constraints. This shows that the parameters were bounded enough to have a polynomial behavior. However, beyond this threshold, our algorithm starts to show its limit. This shows the sensitivity of our algorithm to the parameters (see Figure 4b).



Step 1

$D_{min} = \{C : \langle 15, AC\rangle, \text{B} : \langle 30, AB\rangle\}$

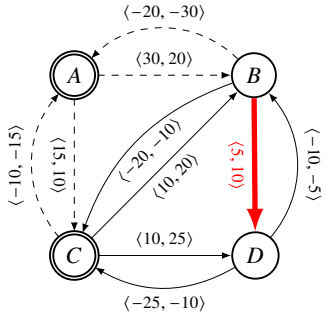
$D_{max} = \{C : \langle 10, AC\rangle, \text{B} : \langle 20, AB\rangle\}$



Step 2

$D_{min} = \{C : \langle 15, AC\rangle, \text{B} : [\langle 20, AB\rangle, \langle 25, ACB\rangle], \text{D} : \langle 15, ACD\rangle\}$

$D_{max} = \{C : \langle 10, AC\rangle, \text{B} : [\langle 20, AB\rangle, \langle 30, ACB\rangle], \text{D} : \langle 25, ACD\rangle\}$



Step 3

$D_{min} = \{C : \langle 15, AC\rangle, B : [\langle 20, AB\rangle, \langle \mathbf{25}, \mathbf{ACB}\rangle],$
$D : [\langle 15, ACD\rangle, \langle 35, ABD\rangle, \langle 30, ACBD\rangle]\}$

$D_{max} = \{C : \langle 10, AC\rangle, B : [\langle \mathbf{20}, \mathbf{AB}\rangle, \langle 30, ACB\rangle],$
$D : [\langle 25, ACD\rangle, \langle 30, ABD\rangle, \langle 40, ACBD\rangle]\}$

Figure 2 – This Figure shows, in a simplified manner, a running example of Algorithm 1 with divergent time-point A by showing only the value and the time-points of a path p. In addition, we highlight the edges taken at each step in the forward search (that respect the conditions in line 7 of Algorithm 1). After step 3, D_{min} and D_{max} contain all the restrictive paths (only those that need to be kept). Then, with Algorithm 2, we find the couples of paths that form a cycle and check if they are WC. It is easy to see that the paths that form the cycles for A (see Figure 3a) are present in D_{min} and D_{max} (we highlight the one that is not WC).

8 Conclusion

This paper introduced a novel approach for checking the WC of an STNU by checking the consistency of its elementary cycles. Interesting features of our algorithm to consider further are as follows : it can identify the constraints causing the uncontrollability, and it can be executed in an incremental way (not optimal) and in a parallelized way. However, it is not capable of computing the minimal network of an STNU. Moreover, we exhibited that the algorithm's complexity depends on the sparsity of the STNU, which makes it exponential in the worst cases. However, experiments show that in loosely connected STNU, the algorithm tends to behave in a polynomial-like way. Finally, the paper argues the relevance of the problem of WC in a multi-agent setting, where uncontrollable events are not controlled by nature but by other agents in the system. Further work will tackle the problem of repairing negative cycles by negotiating the duration of the uncontrollable events, whose duration depends on the other agents' decisions.

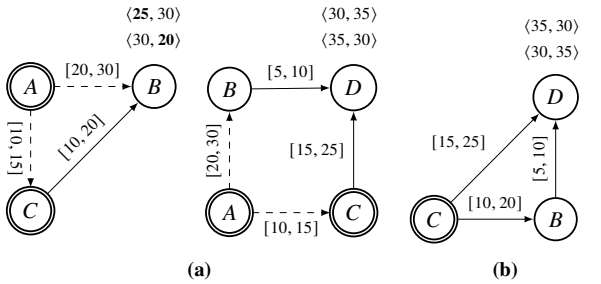
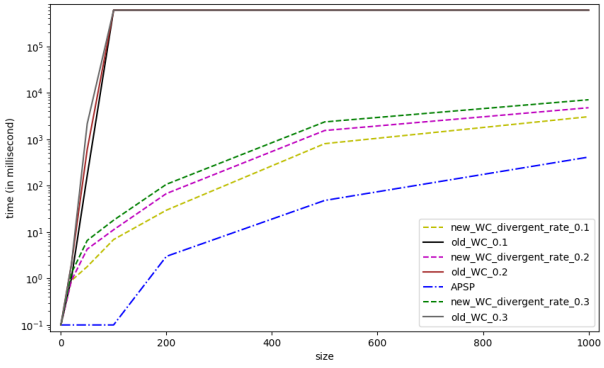
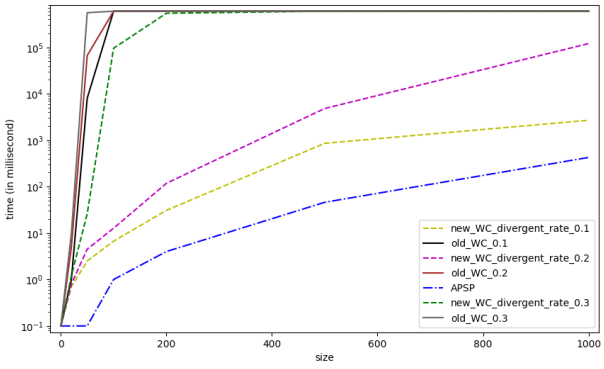


Figure 3 – The figure represents, in a simplified way, the steps of the algorithm for the example of Figure 1. We show in Figure 3a the cycles of the divergent time-point A that are found and checked by the algorithm. The same is done for the divergent time-point C in Figure 3b. The STNU in Figure 1 is not weakly controllable as the cycle ABC is not weakly controllable due to equation (15 + 10) ≤ 20, highlighted in bold, being false.



(a)



(b)

Figure 4 – Experimentation with 20% (a), and 30% (b) of contingent constraints

Références

[1] Shyan Akmal, Savana Ammons, Hemeng Li, and James C Boerkoel Jr. Quantifying degrees of controllability in temporal networks with uncertainty. In *Proceedings of the International Conference on Automated Planning and Scheduling*, 2019.

[2] Shyan Akmal, Savana Ammons, Hemeng Li, Michael Gao, Lindsay Popowski, and James C. Boerkoel. Quantifying controllability in temporal networks with uncertainty. *Artificial Intelligence*, 2020.

[3] Arthur Bit-Monnot and Paul Morris. Dynamic controllability of temporal plans in uncertain and partially observable environments. *J. Artif. Intell. Res.*, 2023.

[4] Guillaume Casanova, Cédric Pralet, Charles Lesire, and Thierry Vidal. Solving dynamic controllability problem of multi-agent plans with uncertainty using mixed integer linear programming. In *ECAI 2016 - 22nd European Conference on Artificial Intelligence, 29 August-2 September 2016, The Hague, The Netherlands - Including Prestigious Applications of Artificial Intelligence (PAIS 2016)*, 2016.

[5] Alessandro Cimatti, Andrea Micheli, and Marco Roveri. Solving temporal problems using smt : weak controllability. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2012.

[6] Rina Dechter, Itay Meiri, and Judea Pearl. Temporal constraint networks. *Artificial intelligence*, 1991.

[7] Luke Hunsberger and Roberto Posenato. Speeding up the rul dynamic-controllability-checking algorithm for simple temporal networks with uncertainty. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

[8] Jsen-Shung Lin, Chin-Chia Jane, and John Yuan. On reliability evaluation of a capacitated-flow network in terms of minimal pathsets. *Networks*, 1995.

[9] Josef Lubas, Marco Franceschetti, and Johann Eder. Resolving conflicts in process models with temporal constraints. In *Proceedings of the ER Forum and PhD Symposium*, 2022.

[10] Paul Morris. Dynamic controllability and dispatchability relationships. In *Integration of AI and OR Techniques in Constraint Programming - 11th International Conference, CPAIOR 2014, Cork, Ireland, May 19-23, 2014. Proceedings*. Springer, 2014.

[11] Paul H. Morris and Nicola Muscettola. Temporal dynamic controllability revisited. In *Proceedings, The Twentieth National Conference on Artificial Intelligence and the Seventeenth Innovative Applications of Artificial Intelligence Conference, July 9-13, 2005, Pittsburgh, Pennsylvania, USA*, 2005.

[12] Ajdin Sumic, Andrea Micheli, Alessandro Cimatti, and Thierry Vidal. Smt-based repair of disjunctive temporal networks with uncertainty : Strong and weak controllability. In *Integration of AI and OR Techniques in Constraint Programming - 11th International Conference, CPAIOR 2024, Uppsala, Sweden, May 28-31 , 2024. Proceedings*, 2024.

[13] Thierry Vidal and Hélène Fargier. Handling contingency in temporal constraint networks : from consistency to controllabilities. *J. Exp. Theor. Artif. Intell.*, 11(1) :23–45, 1999.

Session 8 : Jeux et IA

Actes JIAF 2024

Combinatorial Games with Incomplete Information*

Junkang Li[1,2] **Bruno Zanuttini**[2] **Véronique Ventos**[1]

[1]NukkAI, Paris, France

[2]Normandie Univ.; UNICAEN, ENSICAEN, CNRS, GREYC, 14 000 Caen, France

junkang.li@nukk.ai bruno.zanuttini@unicaen.fr vventos@nukk.ai

Résumé

Les jeux à information incomplète modélisent des interactions multi-agents dans lesquelles les joueurs n'ont pas connaissance commune du jeu auquel ils jouent. Nous proposons une généralisation minimale du formalisme des jeux combinatoires afin d'incorporer l'information incomplète : les jeux combinatoires à information incomplète (CGII). La caractéristique la plus importante des CGII est que toutes les actions sont publiques, ce qui permet de mieux visualiser la connaissance et l'information incomplète de chaque joueur. Pour motiver davantage l'étude de ce nouveau formalisme, nous montrons que le calcul des stratégies optimales pour les CGII a exactement la même complexité algorithmique que pour les jeux généraux sous forme extensive.

Abstract

Games with incomplete information model multi-agent interaction in which players do not have common knowledge of the game they play. We propose a minimal generalisation of combinatorial games to incorporate incomplete information, called combinatorial game with incomplete information (CGII). The most important feature of CGIIs is that all actions are public, which allows better visualisation of each player's knowledge and incomplete information. To further motivate the study of this new formalism, we show that computing optimal strategies for CGIIs has the same computational complexity as for general extensive-form games.

1 Introduction

Game theory is a mathematical framework for studying multi-agent interactions. We focus on *extensive-form games* (EFG), in which the interaction between agents takes place sequentially, i.e. every agent takes turns to make a move. Prominent examples of such games are Chess and Go.

Of particular interest to us is the notion of games with *incomplete information*, which are games in which agents do not have common knowledge of the game they play. For instance, an agent does not know the number of participants in an auction, or how much these participants value the object to be sold; a Poker player does not see the cards in their opponent's hidden hands, hence cannot know for sure the exact consequence (i.e. payoff) of calling and raising bets; a Bridge or Hearts player does not know the cards that their opponent can play during a trick since this depends on their hidden hand; etc.

The notion of (in)complete information is frequently confused with the one of *(im)perfect information*. Complete information describes situations in which the whole structure of a game (the number of players, the game tree, the information sets of each player, the owner of each node, the payoff for each player at each leaf node, etc.) is common knowledge among all the players of the game. On the other hand, perfect information is a more stringent requirement than complete information. Not only the structure of the game is common knowledge, but all players have full observability and perfect recall of the history (which is essentially a record of every decision made by every player so far). In other words, players always know their exact position in the game tree when asked to make the next decision. To summarise, incomplete information is an example of imperfect information; see Faliszewski *et al.* [15, Sec. 2.4.2].

We propose a new and minimal formalism for EFGs with incomplete information that we call combinatorial games with incomplete information (CGIIs). In such a game, Nature picks a world from a universe according to some common prior; each player may have different observability of this world. Then, the game proceeds sequentially, during which there is no chance factor and all moves by the players are publicly observable. This formalism is designed to be a minimal generalisation of the notion of combinatorial

*This article is to be published in the proceedings of the 33rd International Joint Conference on Artificial Intelligence (IJCAI 2024). A long version with proofs of all claims is available at https://hal.science/hal-04568854.

games (which are Boolean games of no chance and with perfect information; see Siegel [39]) and to closely capture the epistemic aspect of games with incomplete information.

For such games, we are interested in knowing how much reward an agent or a team of agents can guarantee for themselves; this corresponds to the notion of maxmin value, well known in optimisation under uncertainty, in which we aim to ensure that the worst possible outcome is not too bad.

By design, our new formalism seems particularly restrictive when compared to general EFGs, where hidden actions and arbitrary chance nodes are allowed. However, we show that the computational complexity of computing optimal strategies (with respect to the maxmin value) for CGIIs is as hard as for EFGs, which allows concluding that the difficulty of playing games comes from incomplete information/knowledge alone, not from hidden actions or mid-game chance factors. This also justifies that restricting algorithmic studies to CGIIs is without loss of generality. We also give a construction to enforce coordination between players in CGIIs under the constraint of public actions, which allows modelling situations similar to concurrent actions.

2 Related Work

Game theory. The study of games with incomplete information was pioneered by Harsanyi [22, 23, 24], who proposes a formalism to model games of incomplete information as EFGs with imperfect information. This formalism, called the Harsanyi model of incomplete information, introduces *types* of players, or equivalently, a universe of worlds for which each player has a potentially different partial observability (also called the Aumann model of incomplete information). For detailed and formal definitions, see textbooks on game theory, e.g. Maschler *et al.* [31, Chapter 9].

Combinatorial games, the inspiration of our formalism of CGII, are studied in the field of combinatorial game theory, established in the 70s by two books by Conway [13] and Berlekamp *et al.* [3, 4, 5, 6]. For recent advances in this field, see Nowakowski [33, 34, 35], Albert and Nowakowski [1], and Larsson [29].

Our formalism is also inspired by Frank and Basin [16], who, in order to model the card play phase of the card game Bridge, propose a game with public actions and one-sided incomplete information in which the opponent has complete information. Frank and Basin [17] show that finding optimal pure strategies for these games is NP-complete. Ginsberg [20] proposes the first exact algorithm for these games, and implements it for Bridge robots. Parallelly, Chu and Halpern [11] study a model of games with incomplete information with common payoffs, and only one round of concurrent interaction after Nature picks the world; they show that it is NP-complete to play such games optimally.

Like us, Kovarík *et al.* [27] highlight the distinction between public and private actions. They also argue that this distinction, essential for recent search algorithms, is partially lost when we model sequential multi-agent interaction with EFGs, which do not *explicitly* tell whether an action is public or not. They propose an alternative model for stochastic games that makes this distinction prominent, and show how to transform such models to augmented EFGs and vice versa.

Complexity of games. Most work in the literature on the computational complexity of games concerns the complexity of finding Nash equilibria, especially for normal-form games [18, 14]. For more references, see Conitzer and Sandholm [12], who also show that it is NP-complete to decide whether Nash Equilibria with certain natural properties exist.

Koller and Megiddo [25], Koller *et al.* [26], and von Stengel [41] make seminal contributions to understanding the complexity of two-player zero-sum EFGs. They also give polynomial-time algorithms for computing behaviour maxmin strategies of EFGs with perfect recall, based on linear programming.

Maxmin for a team of players with common payoffs is called team maxmin equilibrium (TME) in the literature, and was first proposed by von Stengel and Koller [42]. Basilico *et al.* [2] and Celli and Gatti [8] propose another notion called TMECor ("Cor" stands for "correlation"), which allows agents in the same team to access a correlation device in order to coordinate their mixed strategies. Building on these works, Gimbert *et al.* [19] and Zhang *et al.* [43] study the complexity of TME and TMECor, thereby yielding a relatively complete picture of the complexity of behaviour and mixed maxmin for two-team EFGs.

The complexity of other models of decision making have also been extensively studied, e.g. Markov decision process [32, 7, 21], propositional planning [37], graph games [10, 9]. Similar to these works, we confirm the intuition that partial observability and multi-agent coordination increases the difficulty of optimal decision making.

3 Combinatorial Games with Incomplete Information

3.1 Definitions

Combinatorial games are EFGs of no chance and with perfect information. To generalise this formalism minimally to allow incomplete information, we propose the following definition.

Definition 3.1 (CGII)**.** *A* combinatorial game with incomplete information *(CGII) is a tuple of the following elements:*

- *An Aumann model* $\langle U, A, (\mathcal{R}_i)_{i\in A}, \rho\rangle$*, where* U *is a finite set of* worlds *called* universe*,* A *is a set of* agents*,* $\mathcal{R}_i$ *is an equivalence relation over* U *for each agent* $i \in A$*, and* $\rho \in \Delta(U)$ *is a probability distribution over the universe called* common prior*;*
- *A tree* T *called* public tree*, the nodes of which (*$\mathrm{N}(T)$*) are partitioned into* $\{\mathrm{N}_i(T)\}_{i\in A} \cup \mathrm{L}(T)$*, where* $\mathrm{L}(T)$ *is the set of all leaves;*
- *A reward function* $u_i : \mathrm{L}(T) \times U \to \mathbb{R}$ *for each* $i \in A$*.*

Note that the children of a node (available actions at that node) do not depend on the real (and partially observable) world ω; only the rewards depend on ω.

A CGII is said to be *Boolean* if all its reward functions have values in $\mathbb{B}$.[1] The Aumann model of a CGII defines each agent's observability over the universe, which characterises their incomplete information.

Pure strategies in a CGII. A CGII as an EFG with incomplete information proceeds as follows. First, Nature picks the *real world* $\omega \in U$ according to ρ. Then the *state game* in ω proceeds from the root of the public tree T; agents take turns to pick a child of the current node, depending on their equivalence class of the real world. This continues until a leaf l is reached, and agent i receives a payoff $u_i(l, \omega)$.

Definition 3.2 (Pure strategy)**.** *A* pure strategy *of an agent* $i \in A$ *is a mapping* $s_i : \mathrm{N}_i(T) \times U \to \mathrm{N}(T)$ *such that for all* $v \in \mathrm{N}_i(T)$*:*

- *For all* $\omega \in U$*,* $s_i(v, \omega)$ *is a child of* v*;*
- $\forall \omega, \omega' \in U, \omega \mathcal{R}_i \omega' \implies s_i(v, \omega) = s_i(v, \omega')$.[2]

The set of all pure strategies of agent i *is denoted by* Σ_i^{P}*.*

From the definition of a strategy, one can see that the actions of every agent are indeed public: when making a decision at a node, an agent knows perfectly where the node is in the public tree, which in particular means they observe and remember the actions picked by every agent in the past, starting from the root of the public tree. In addition, compared to general games with incomplete information, the state games of a CGII have the particularity that they share the same game tree T, which does not have chance nodes. These three features (public actions, unique game tree, and no chance) are the defining features of our formalism CGII.

Each CGII describes an EFG in which Nature picks the real world at the root, and information sets are determined by the players' observability of the world.

[1] In Boolean games, the rewards 0 and 1 are interpreted as a loss and a win, respectively.

[2] This means agent i must pick the same child for a node in any two worlds indistinguishable by them.

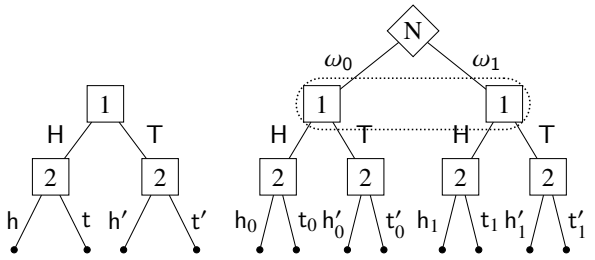


Figure 1: The public tree of a CGII and the game tree of its corresponding EFG, both with rewards omitted.

Example. *Consider the following CGII: the public tree is shown in Figure 1 (on the left); the universe reads* $\{\omega_0, \omega_1\}$*; agent 2 can distinguish these two worlds while agent 1 cannot. This CGII models a variant of Matching Pennies with incomplete information. The game tree of its corresponding EFG is also shown in Figure 1 (on the right).*

On the right, since agent 1 cannot observe the real world, they must play H *in both state games, or* T *in both. This constraint is respected by the notion of strategies in a CGII: on the left, agent 1 only has two pure strategies* H *and* T *since* ω_0 *and* ω_1 *are indistinguishable by agent 1.*

Similarly, on the right, agent 2 can pick between heads or tails, depending on the choices of Nature and agent 1. On the left, agent 2 can again pick between heads or tails, depending on agent 1's choice and the real world, since agent 2 can distinguish between ω_0 *and* ω_1*; note that the latter point is not reflected by the public tree, but by the Aumann model.*

Teams and Information in a CGII. In a CGII, agents i and j are said to be in the same *team* if $u_i = u_j$.

Definition 3.3 (Team)**.** *A* team *is an inclusion-wise maximal group of agents with the same reward function.*

We now define a team's *degree of incomplete information*.

- *Multi-agent incomplete information* (MA-II): an arbitrary team.
- *Single-agent incomplete information* (SA-II): a team of agents with the same equivalence relation (i.e. $\mathcal{R}_i = \mathcal{R}_j$ for all agents i and j in the team).
- *Complete information* (CI): a team whose agents all have the finest equivalence relation (i.e. $\mathcal{R}_i = \{(\omega, \omega) \mid \omega \in U\}$ for all agents i in the team).

In particular, CI implies SA-II, which implies MA-II. Intuitively, a team is a group of decentralised agents with shared interests working cooperatively. In a CGII, a team with SA-II can be regarded as one single agent since every

agent in this team has the same information and all actions are public.

Example. *In the CGII in Figure 1, if the two agents have the same reward function, then they are in a team with MA-II; otherwise, each is a (single-agent) team with SA-II.*[3]

Due to the public actions property, there is a close link between the degree of incomplete information of a team in a CGII and the degree of imperfect information of the corresponding team in the EFG defined by the CGII:

- a team with CI in the CGII is a player with perfect information in the EFG;
- a team with SA-II in the CGII can be seen as a single player with perfect recall in the EFG;
- a team with MA-II in the CGII is a team of players who all have perfect recall in the EFG.

This correspondence will allow us to establish upper bounds on the complexity of solving CGIIs.

Team maxmin in a CGII. Let $(s_1, \dots, s_n) \in \Sigma_1^{\mathrm{P}} \times \cdots \Sigma_n^{\mathrm{P}}$, where $n = |A|$, be a pure *strategy profile*. We write $(s_1, \dots, s_n)(\omega)$ for the unique leaf reached under this profile when the real world is ω.

Definition 3.4 (Expected utility)**.** *The* expected utility *for an agent $i \in A$ under a pure strategy profile $(s_1, \dots, s_n)$ is defined to be:*

$$\mathcal{U}_i(s_1, \dots, s_n) = \sum_{\omega \in U} \rho(\omega) u_i\big((s_1, \dots, s_n)(\omega), \omega\big).$$

Let $\mathcal{T} \subseteq A$ be a team. Notice that all agents in a team share the same expected utility function, which we denote by $\mathcal{U}_{\mathcal{T}}$. A pure strategy of the team is uniquely defined by the pure strategy of each of its players. In particular, the set of pure strategies of a team $\mathcal{T}$, denoted by $\Sigma_{\mathcal{T}}^{\mathrm{P}}$, is in bijection with $\prod_{i \in A} \Sigma_i^{\mathrm{P}}$. In the following, we also write $\Sigma_{-\mathcal{T}}^{\mathrm{P}} = \prod_{i \notin \mathcal{T}} \Sigma_i^{\mathrm{P}}$, the set of pure strategy profiles of the players not in $\mathcal{T}$.

Definition 3.5 (Pure maxmin for a team)**.** *The* pure maxmin *value for a team $\mathcal{T} \subseteq A$ is defined to be*

$$\underline{v}_{\mathcal{T}} \coloneqq \max_{s_{\mathcal{T}} \in \Sigma_{\mathcal{T}}^{\mathrm{P}}} \min_{s_{-\mathcal{T}} \in \Sigma_{-\mathcal{T}}^{\mathrm{P}}} \mathcal{U}_{\mathcal{T}}(s_{\mathcal{T}}, s_{-\mathcal{T}}).$$

Intuitively, this value is the largest expected reward that a team can guarantee to get by playing a pure strategy.

The notion of behaviour/mixed strategy can be defined similarly to the one for EFGs: a mixed strategy of an agent i is a probability mixture of pure strategies of i; a behaviour strategy of i picks, at each node and for each equivalence class of $\mathcal{R}_i$, a probability mixture of children (instead of just a child as for pure strategies). Hence, expected utility with respect to behaviour/mixed strategy profiles and behaviour/mixed maxmin for a team can be similarly defined.[4]

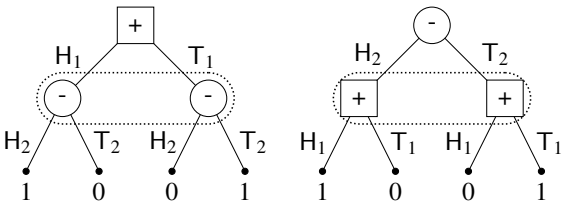


Figure 2: Two EFGs for Matching Pennies.

In the following, we focus on zero-sum two-team CGIIs. We call the two teams *player MAX* and *player MIN*, and denote them by + and −, respectively.

3.2 Motivation for CGIIs

Our motivations for introducing CGII as a subclass of games with incomplete information are multiple. First and foremost, the formalism of CGII aims to be a minimal generalisation of combinatorial games to allow incomplete information. Indeed, it is clear that a CGII with a singleton universe is a combinatorial game. This new formalism allows modelling a number of card games, notably Bridge.[5]

But more importantly, the formalism of CGII also aims to minimally capture the notion of knowledge and incomplete information. Due to the public actions property, the only source of the imperfect (in particular, incomplete) information of every agent comes from their partial observability of the real world, drawn at the beginning of a game.

In contrast, we argue that the distinction between perfect and imperfect information does not completely capture the essence of players' knowledge. For example, in the game Matching Pennies, MAX and MIN pick a side of a coin concurrently; this can be modelled by two different EFGs, shown in Figure 2. In the EFG on the left, MAX has perfect information while MIN has imperfect information but perfect recall; and in the EFG on the right, the situation is reversed. However, the roles of MAX and MIN are symmetric in Matching Pennies; MAX also has exactly the same information/knowledge in both EFGs when they need to choose an action. Hence, considering CGIIs allows one to focus on an unambiguous notion of knowledge of the players, as captured by the Aumann model and the initial drawing of a world.

[3] The team of agent 2 even has CI.

[4] Behaviour maxmin and mixed maxmin are commonly known as TME and TMECor in the literature [8].

[5] The card play of Bridge can be described as a CGII in which MAX has SA-II and MIN has MA-II.

Expressiveness. At first sight, the requirements of public actions and no chance seem particularly restrictive: many popular tabletop games with incomplete information allow private actions (e.g. concealed Kong in Mahjong, pass in Hearts) or have randomness and chance factors besides the initial drawing (e.g. dice rolls during a game). One may worry that, due to these restrictions, CGII is not expressive enough to be conceptually or algorithmically interesting. However, we argue that this impression is not correct.

First, an initial drawing over the universe is actually quite expressive. For example, for the dice rolls we evoke above, if their number and occasions are fixed in advance, then their results can be encoded into the initial drawing of worlds.[6] Another example is given by video games, which typically use a random seed as the sole source of randomness for all procedurally generated levels and random events during a playthrough. Similar ideas have been investigated in automated planning [36, Sec. 10].

Second, even with only public actions, we show in subsection 4.1 that we can still design a game to force a team of players to coordinate their actions. This means that we can essentially encode concurrent actions (as in standard Matching Pennies) using only public actions (and no chance except the initial drawing).

All in all, we suggest that at least as far as computation of optimal strategies is concerned, CGII, rather than EFG, be the right model for studying sequential multi-agent interactions depending on each player's knowledge. Moreover, as we will show, CGIIs are as hard to solve as EFGs, which confirms our intuition that the difficulty of a game actually comes from the incomplete information of a player, and not from their inability to observe the moves made by the other players.

4 Complexity of Pure Maxmin for CGIIs

The decision problem Pure Maxmin is defined as follows.

Definition 4.1 (Pure Maxmin)**.** *Let $\mathcal{G}$ be a class of zero-sum CGIIs. Then* Pure Maxmin($\mathcal{G}$) *is the following decision problem.*

Input *A CGII $G \in \mathcal{G}$ and a rational number m.*

Output *Decide whether the pure maxmin value for team MAX in G satisfies $\underline{v}_+(\Sigma_+^{\mathrm{P}}, \Sigma_-^{\mathrm{P}}) \geq m$.*

We study the complexity of Pure Maxmin for CGIIs depending on the degrees of incomplete information for MAX and MIN: complete information (CI), single-agent incomplete information (SA-II), multi-agent incomplete information (MA-II). For complexity analyses, we consider the parameters $|T|$ (number of nodes in the public tree), $|U|$ (number of worlds), and possibly the number of bits to encode the utilities, the common prior, and the threshold m.

[6] This will only enlarge the game tree by a polynomial factor.

MAX \ MIN	CI	SA-II	MA-II
CI	P	**NP-c**	**Σ_2^{P}-c**
SA-II	NP-c	NP-c	**Σ_2^{P}-c**
MA-II	NP-c	NP-c	**Σ_2^{P}-c**

Table 1: Complexity of Pure Maxmin for CGIIs.

The complexity of Pure Maxmin is summarised in Table 1. By definition, the complexity of each case is increasingly monotone in both MAX's and MIN's degree of incomplete information (CI/SA-II/MA-II). Hence, only a few hardness results have to be proved to establish the table. The results written in bold font are new from this work; the others can be directly deduced from the literature.

The membership results in Table 1 follow from results by Koller and Megiddo [25, Sec. 3.3]; in particular, memberships in NP and in Σ_2^{P} follow from the fact that given a strategy of MAX, computing MIN's best response is a problem in coNP, and even linear time when MIN has perfect recall.

Hence, we focus on hardness results. The following result is by Frank and Basin [17, Sec. 6].[7]

Proposition 4.2. Pure Maxmin *is* NP*-hard for Boolean CGIIs in which MAX has SA-II and MIN has CI.*

The symmetric case does not trivially follow from this result (since the minimax theorem does not hold for pure strategies) and necessitates a proof:

Proposition 4.3. Pure Maxmin *is* NP*-hard for Boolean CGIIs in which MAX has CI and MIN has SA-II.*

Proof sketch. By a reduction from Vertex Cover. Given a graph (V, E), consider the universe $U = \{\omega_e \mid e \in E\}$; the worlds are observable by MAX but not by MIN. During the game, MAX picks a vertex $v \in V$, then MIN picks an edge $e' \in E$. In a world $\omega_e \in U$, MAX gets a payoff of 1 if v covers e and MIN does not correctly guess this edge (i.e. $e' \neq e$); otherwise, MAX gets 0. One can verify that the pure maxmin value of this game is at least $1 - k/|E|$ if and only if the graph has a vertex cover of size at most k. □

4.1 Multi-Agent Coordination in CGIIs

Coordination game. Now we turn our attention to CGIIs with multi-agent teams. We first show how to construct CGIIs to impose a perfect coordination between agents from the same team (à la Matching Pennies).

[7] In their setting, there is no prior over the worlds; they are interested in the strategies that win in the greatest number of worlds. This is equivalent to finding maxmin strategies with respect to the uniform prior in our setting.

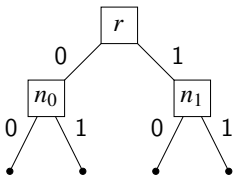


Figure 3: The public tree of the coordination game.

Consider the following Boolean CGII, which we call *coordination game*. This game has two agents of MAX, referred to as MAX 1 and MAX 2, and no agent of MIN; its universe has 4 worlds and reads $U = \{(b_1, b_2) \mid b_1, b_2 \in \mathbb{B}\}$; its Aumann model has the uniform common prior and is such that for $i = 1, 2$, MAX i only observes b_i; its public tree is shown in Figure 3; the reward for MAX is 1 if and only if $a_1 \oplus b_1 = a_2 \oplus b_2$, where $\oplus$ is the exclusive or of two bits and a_i is the action chosen by MAX i.

We refer to b_i as the *hidden bit* of MAX i since it is only observable by MAX i. The coordination game is designed in such a way that MAX 1 and MAX 2 must perfectly coordinate their answer in order to win. Intuitively, MAX 1 and MAX 2 need to agree on the same answer $A \in \mathbb{B}$, then stick to it during the game by playing $a_i = A \oplus b_i$. Indeed, if they employ this strategy, then they guarantee a win since

$$a_1 \oplus b_1 = (A \oplus b_1) \oplus b_1 = A = (A \oplus b_2) \oplus b_2 = a_2 \oplus b_2.$$

Remark. *Under these two winning strategies (one for each value of A), both MAX 1 and 2 pick the actions* 0 *and* 1 *with equal probability. Indeed, once the common answer A is fixed, which action to play by MAX i is dictated by their hidden bit b_i. Hence, the bits b_1 and b_2 act as the keys of a one-time pad to* encrypt/mask *the intended answer (i.e. A) of MAX 1 and 2. This is the key element to ensure that MAX 1 and 2 must cooperate without cheating.*

Proposition 4.4. *In a coordination game, the only winning pure strategies of team MAX are of the following form: for some $A \in \mathbb{B}$, MAX* 1 *plays $A \oplus b_1$ and MAX* 2 *plays $A \oplus b_2$.*

Proof. Notice that the pure strategies of MAX 1 can be written in the form (a_1^0, a_1^1), which means they choose a_1^0 if $b_1 = 0$ and a_1^1 if $b_1 = 1$. As for MAX 2, they have the right to pick a_2 as a function of a_1 and b_2. If MAX 1 plays (A, A) (i.e. they play A regardless of b_1) for some $A \in \mathbb{B}$, then MAX 2 has no winning strategy, since the winning condition $A \oplus b_1 = a_2 \oplus b_2$ cannot be satisfied for both values of b_1. Now if MAX 1 plays $(A, A \oplus 1)$ for some $A \in \mathbb{B}$, then to satisfy the winning condition, MAX 2 is forced to play $a_2 = A \oplus b_2$; hence $a_i = A \oplus b_i$. □

The same reasoning also shows that these pure strategies are also the only winning behaviour strategies, and that the winning mixed strategies are exactly the mixtures of them.

Remark. *From this proof, one can see that if MAX 1 cheats by using their hidden bit b_1 incorrectly (i.e. does not use b_1 to encrypt their intended answer and always picks the same action), then MAX 2 cannot cooperate perfectly since they cannot observe the value of b_1.*

In addition, when MAX 1 plays correctly (i.e. chooses a strategy of the form $(A, A \oplus 1)$), then MAX 2 must also pick A as their intended answer and mask it with their own bit b_2 in order to win. Notice that in this case, the action picked by these two agents are uniformly and independently distributed. This is an important feature since agents (of MAX or MIN) in the following part of the game tree cannot deduce any information about the intended answer of these two agents by observing only their actions.

Interrogation game. We now generalise the coordination game to the following situation: we have a finite set of questions Q, and MAX has a Boolean answer for each question $\{A_q \in \mathbb{B}\}_{q \in Q}$, or equivalently a mapping from Q to $\mathbb{B}$. We wish to verify whether MAX's mapping satisfies some given binary constraints $\{C_{qq'} \subseteq \mathbb{B}^2 \mid q, q' \in Q, q \neq q'\}$: MAX's mapping is said to be *valid* if it satisfies all the constraints, that is, $(A_q, A'_q) \in C_{qq'}$ for all $C_{qq'}$.

Example. *For cliques of a given graph, the questions are the vertices of this graph; MAX's mapping induces a subgraph (MAX's answer to a vertex corresponds to whether to include this vertex in their intended subgraph); the binary constraints impose the requirement that all vertices in this subgraph be connected. Then MAX's mapping is valid if and only if it describes a clique of the graph.*

To model this situation, consider the following Boolean CGII, which we call *interrogation game*: two agents of MAX (MAX 1 and MAX 2), and no agent of MIN; the universe reads $U = \{(q_1, b_1, q_2, b_2) \mid q_1, q_2 \in Q, b_1, b_2 \in \mathbb{B}\}$; the Aumann model is such that for $i = 1, 2$, MAX i only observes q_i and b_i; the common prior is uniform; the public tree is the same one as for the coordination game (i.e. the one in Figure 3); MAX loses if and only if either (1) $q_1 = q_2$ but $a_1 \oplus b_1 \neq a_2 \oplus b_2$ or (2) $q_1 \neq q_2$ but $(a_1 \oplus b_1, a_2 \oplus b_2) \notin C_{q_1 q_2}$.

This CGII has size $O(|Q|^2)$: the universe has size $O(|Q|^2)$, while the public tree has size $O(1)$. Notice that a coordination game is just an interrogation game with only one question (hence no binary constraint). We refer to (q_i, b_i) as the *hidden information* of MAX i. Inspired by the coordination game, we propose the following definition.

Definition 4.5 (Perfect coordination)**.** *In an interrogation game, a* perfect coordination *of team MAX is a pure strategy of MAX of this form: there is a set $\{A_q \in \mathbb{B}\}_{q \in Q}$ such that for all i, MAX i will play the action $a_i = A_{q_i} \oplus b_i$ in all worlds in which their hidden information is (q_i, b_i). For*

such a strategy, the set $\{A_q\}_{q\in Q}$ *is called the* intended mapping *or* intended answer *of the perfect coordination.*

By a similar argument to the one for the coordination game, the reward condition (1) ensures that MAX 1 and 2 have an incentive to implement a perfect coordination, which is a dominant strategy. In other words, (1) imposes non-adaptivity of MAX's answers. As for the reward condition (2), it ensures that all binary constraints are satisfied by the intended mapping of a perfect coordination, since by (1) we have $a_i \oplus b_i = A_{q_i}$ for all i. In summary, we have established the following result.

Proposition 4.6. *In an interrogation game, a pure strategy of team MAX is winning if and only if it is a perfect coordination with a valid intended mapping.*

It is straightforward to construct interrogation games involving team MIN such that if MIN does not cooperate, MAX receives a large reward. Similarly, we can also extend the construction above to allow k-ary constraints with $k \geq 2$. The interrogation game will then involve k agents of MAX, each with their hidden information (q_i, b_i), and has size $O(2^k|Q|^k)$. Such an interrogation game can be used to encode problems such as k-SAT.[8]

4.2 Hardness for Two-Team CGIIs

With the gadgets of interrogation game, it is straightforward to show that Pure Maxmin is Σ_2^{P}-hard for CGIIs in which both MAX and MIN are multi-agent teams, for instance by a reduction from the canonical problem $\exists\forall$3SAT. However, we provide a stronger result: Σ_2^{P}-hardness holds even when MAX has complete information.

Proposition 4.7. Pure Maxmin *is* Σ_2^{P}*-hard for CGIIs in which MAX has CI and MIN has MA-II.*

Proof sketch. By a reduction from the Σ_2^{P}-complete problem Succinct Set Cover [40]: given a collection of 3-DNF formulae and an integer k, decide whether there is a subset S of size at most k the disjunction of which is a tautology.

We design a game in which Nature draws a DNF formula from the collection, 3 variables, and 4 hidden bits, according to the uniform common prior. The DNF is known to MAX, who plays 1 or 0 according to whether it should be in S. This answer is masked (to MIN) by the hidden bit of MAX, as in a coordination game. Then MIN chooses either to verify the size of S or to verify that the disjunction of S is a tautology. The other 3 hidden bits are used in the latter verification: MIN plays an interrogation game over the 3 variables, with the constraint to falsify the disjunction of S.

Finally, since MAX is designed to have CI, they know the variables and the hidden bits of MIN. To ensure that MAX does not play a strategy that depends on MIN's information, we introduce one additional agent of MIN whose role is to punish MAX whenever MAX plays such a strategy. □

[8] Contrastingly, we leave open the problem of constructing an interrogation game in which MAX's answers are not binary.

MAX \ MIN	CI	SA-II	MA-II
CI	P	P	**coNP-c**
SA-II	P	P	**coNP-c**
MA-II	**NP-c**	**NP-c**	$\mathbf{\Sigma_2^P}$**-c/**$\mathbf{\Delta_2^P}$**-c**

Table 2: Complexity of Behaviour Maxmin and of Mixed Maxmin for CGIIs.

Remark. *The construction shows something stronger:* Σ_2^{P}*-hardness holds even when MIN has* joint *complete information (i.e. if the agents of MIN could pool their information, then they would have complete information).*

5 Complexity of Behaviour Maxmin and Mixed Maxmin

The decision problems Behaviour Maxmin and Mixed Maxmin can be defined similarly to Pure Maxmin, the only difference being that MAX can use behaviour/mixed strategies instead of just pure strategies.[9]

The complexity of Behaviour Maxmin and Mixed Maxmin is summarised in Table 2. Again, the complexity is increasingly monotone in both dimensions, and results written in bold font are new. The only case where the complexity differs between behaviour and mixed strategies is the case in which both MAX and MIN have MA-II; in this case, Behaviour Maxmin and Mixed Maxmin are respectively Σ_2^{P}- and Δ_2^{P}-complete.

The membership results follow from those for EFGs, which are superclasses of CGIIs: the results for P are by Koller and Megiddo [25, Sec. 3.5], and the others by Zhang *et al.* [43, Appx. C].

Therefore, we only have to establish the hardness results when MAX and/or MIN have MA-II. We first adapt a reduction from 3-SAT by Chu and Halpern [11].

Proposition 5.1. *Both* Behaviour Maxmin *and* Mixed Maxmin *are* NP*-hard for Boolean CGIIs in which MAX has MA-II and MIN has CI.*

Proof. For a 3-CNF with N clauses, consider the following CGII. The universe consists of the clauses, which are observable by MAX 1 but not by MAX 2, and with the

[9] In our definition for all these decision problems, MIN only uses pure strategies, which is without loss of generality. Indeed, MIN is a team of agents with perfect recall, hence every MIN's behaviour strategy has an equivalent mixed strategy [31, Theorem 6.11]. In addition, the best responses in mixed strategies are no better than the best responses in pure strategies due to the linearity of expected utility with respect to mixtures of strategies.

uniform prior. During the game, MAX 1 picks a variable, then MAX 2 observes this variable and picks a truth value. They win if and only if the variable picked by MAX 1 is in the clause picked by Nature, and the truth value picked by MAX 2 for this variable renders this clause true.

Since there is no agent of MIN in this game, playing behaviour or mixed strategies is no better than pure ones. It is also straightforward to verify that MAX can guarantee an expected payoff of 1 if the 3-CNF is satisfiable; otherwise, the maxmin value for MAX is at most $1 - 1/N$. □

Now, coNP-hardness for the symmetric case (when MAX has CI and MIN has MA-II) essentially follows from this result. For mixed strategies, the minimax theorem ensures that when switching the roles of MIN and MAX, and negating the utilities in the game from the proof of Proposition 5.1, the maxmin value for MAX is at least $-(1 - 1/N)$ if the 3-CNF is unsatisfiable, and -1 otherwise. The hardness for Behaviour Maxmin follows from the fact that mixed maxmin and behaviour maxmin have the same value due to MAX's perfect recall.

Proposition 5.2. Behaviour Maxmin *is* Σ_2^{P}*-hard for CGIIs in which both MAX and MIN have MA-II.*

Proof sketch. By a reduction from ∃∀3SAT (for a 3-DNF formula $\varphi(x, y)$, decides whether $\exists x \forall y\, \varphi(x, y)$ holds) which is known to be Σ_2^{P}-hard [38]. Given such a formula, we construct a CGII with 3 agents of MAX and 3 agents of MIN. The worlds consist of one existential (resp. universal) variable and one hidden bit for each agent of MAX (resp. of MIN); the common prior is uniform; each agent only observes their variable and hidden bit. During the game, the agents of MAX take turns to choose between 0 and 1, then so do the agents of MIN. The total payoff for MAX is computed as follows: (1) an inconsistency among the agents of MAX (in the sense of an interrogation game) yields $-N$ for MAX, where N is a large real number; (2) an inconsistency among the agents of MIN yields $+N$ for MAX; (3) if at least one term in $\varphi(x, y)$ is satisfied by the assignment picked by the agents of MAX and MIN, then MAX receives +1.

By choosing N large enough, agents of MAX have an incentive to perform a perfect coordination, and the same goes for agents of MIN. In particular, MAX has no incentive to play non-pure behaviour strategies, which would cause inconsistency to happen with a non-zero probability. It is then straightforward to verify that $\exists x \forall y\, \varphi(x, y)$ holds if and only if MAX can guarantee an expected utility of at least $+1/n^3$, where n is the maximum between the number of existential variables and the number of universal ones. □

Proposition 5.3. Mixed Maxmin *is* Δ_2^{P}*-hard for CGIIs in which both MAX and MIN have MA-II.*

Proof sketch. By a reduction from Last SAT (for a 3-CNF, decide whether the lexicographically maximum satisfying assignment has value 1 for the last variable), which is Δ_2^{P}-hard [28]. The construction is very similar to the last proof. Given a 3-CNF, we write the variables as $x_1, \ldots, x_n$, and we construct a CGII with 3 agents of MAX and 3 agents of MIN. The worlds consist of one variable and one hidden bit for each agent of MAX or MIN; the common prior is uniform; each agent only observes their variable and hidden bit. During the game, the agents of MAX take turns to choose between 0 and 1, then so do the agents of MIN. The total payoff for MAX is computed as follows: (1) an inconsistency among the agents of MAX or a clause violated by their assignment yields $-2N$ for MAX, where N is a large real number; (2) an inconsistency among the agents of MIN or a clause violated by their assignment yields $+N$ for MAX; (3) for the first agent of MAX (resp. of MIN), if their hidden variable and bit are x_k and b, and they pick $1 \oplus b$, then MAX receives $+2^{n-k}$ (resp. -2^{n-k}); (4) MAX receives a bonus +1 if the variable x_n is assigned $1 \oplus b_1^+$ by the first agent of MAX.

By choosing N large enough, both MAX and MIN have an incentive to perform a perfect coordination (which can be pure or mixed for MAX) with a satisfying assignment. Let $x = (x_1, \ldots x_n)$ be the lexicographically maximum satisfying assignment (if there is no such assignment, then MAX is bound to get a large negative expected utility). If $x_n = 1$, then MAX can guarantee an expected utility of $+1/n$ by choosing this assignment for their perfect coordination; the best MIN can do is to choose this assignment. If $x_n = 0$, MAX has an expected utility of at most 0 when MIN plays this assignment: MAX gets 0 by playing the same assignment, and possibly less when playing other satisfying (hence lexicographically smaller) assignments with a non-zero probability. □

6 Conclusion

We have proposed a new formalism for extensive-form games with incomplete information that we name combinatorial games with incomplete information. Compared to EFGs, CGIIs only have public actions and one chance node at the beginning of the game, thereby putting better emphasis on the aspect of incomplete information/knowledge of the players.

Apart from the conceptual simplicity, the interests in this new formalism are also justified by the complexity results. Indeed, all the upper bounds for CGIIs are provided by membership results that also hold for EFGs, while all the lower bounds, proven by hardness results, coincide with the upper bounds. In particular, for every degree of observability, CGIIs have the same complexity as EFGs.

We have also shown how to model binary concurrent actions to enforce multi-agent coordination in CGIIs. We leave to future work how to model other types of hidden actions, in particular non-binary concurrent actions. Fu-

ture work also includes tightening the complexity results to show that hardness holds even for Boolean CGIIs with a minimum number of agents and distributed knowledge of the real world for each team; designing a generic polynomial transformation from an arbitrary two-team EFG into a CGII; and extending the study to general-sum multi-team CGIIs with respect to solution concepts that generalise maxmin (e.g. strategies to commit to [30]). Algorithmic studies adapted to CGIIs will also be of interest, with the long-term goal to implement better AIs for games such as Bridge.

References

[1] Michael H. Albert and Richard J. Nowakowski, editors. *Games of No Chance 3*, volume 56 of *Mathematical Sciences Research Institute Publications*. Cambridge University Press, 2009.

[2] Nicola Basilico, Andrea Celli, Giuseppe De Nittis, and Nicola Gatti. Team-maxmin equilibrium: Efficiency bounds and algorithms. In Satinder Singh and Shaul Markovitch, editors, *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI 2017)*, pages 356–362. AAAI Press, 2017.

[3] Elwyn R Berlekamp, John H Conway, and Richard K Guy. *Winning Ways for Your Mathematical Plays, Volume 1*. CRC Press, 2nd edition, 2001.

[4] Elwyn R Berlekamp, John H Conway, and Richard K Guy. *Winning Ways for Your Mathematical Plays, Volume 2*. CRC Press, 2nd edition, 2003.

[5] Elwyn R Berlekamp, John H Conway, and Richard K Guy. *Winning Ways for Your Mathematical Plays, Volume 3*. CRC Press, 2nd edition, 2003.

[6] Elwyn R Berlekamp, John H Conway, and Richard K Guy. *Winning Ways for Your Mathematical Plays, Volume 4*. CRC Press, 2nd edition, 2004.

[7] Daniel S. Bernstein, Robert Givan, Neil Immerman, and Shlomo Zilberstein. The complexity of decentralized control of markov decision processes. *Math. Oper. Res.*, 27(4):819–840, 2002.

[8] Andrea Celli and Nicola Gatti. Computational results for extensive-form adversarial team games. In Sheila A. McIlraith and Kilian Q. Weinberger, editors, *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI 2018)*, pages 965–972. AAAI Press, 2018.

[9] Krishnendu Chatterjee, Laurent Doyen, and Thomas A. Henzinger. A survey of partial-observation stochastic parity games. *Formal Methods Syst. Des.*, 43(2):268–284, 2013.

[10] Krishnendu Chatterjee and Thomas A. Henzinger. A survey of stochastic ω-regular games. *J. Comput. Syst. Sci.*, 78(2):394–413, 2012.

[11] Francis C. Chu and Joseph Y. Halpern. On the np-completeness of finding an optimal strategy in games with common payoffs. *Int. J. Game Theory*, 30(1):99–106, 2001.

[12] Vincent Conitzer and Tuomas Sandholm. New complexity results about Nash equilibria. *Games Econ. Behav.*, 63(2):621–641, 2008.

[13] John H. Conway. *On Numbers and Games*. A K Peters/CRC Press, 2nd edition, 2000.

[14] Constantinos Daskalakis, Paul W. Goldberg, and Christos H. Papadimitriou. The complexity of computing a Nash equilibrium. *SIAM J. Comput.*, 39(1):195–259, 2009.

[15] Piotr Faliszewski, Irene Rothe, and Jörg Rothe. Non-cooperative game theory. In Jörg Rothe, editor, *Economics and Computation, An Introduction to Algorithmic Game Theory, Computational Social Choice, and Fair Division*, Springer texts in business and economics, pages 41–134. Springer, 2016.

[16] Ian Frank and David A. Basin. Search in games with incomplete information: A case study using bridge card play. *Artif. Intell.*, 100(1-2):87–123, 1998.

[17] Ian Frank and David A. Basin. A theoretical and empirical investigation of search in imperfect information games. *Theor. Comput. Sci.*, 252(1-2):217–256, 2001.

[18] Itzhak Gilboa and Eitan Zemel. Nash and correlated equilibria: Some complexity considerations. *Games and Economic Behavior*, 1(1):80–93, 1989.

[19] Hugo Gimbert, Soumyajit Paul, and B. Srivathsan. A bridge between polynomial optimization and games with imperfect recall. In Amal El Fallah Seghrouchni, Gita Sukthankar, Bo An, and Neil Yorke-Smith, editors, *Proceedings of the Nineteenth International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020)*, pages 456–464. International Foundation for Autonomous Agents and Multiagent Systems, 2020.

[20] Matthew L. Ginsberg. GIB: imperfect information in a computationally challenging game. *J. Artif. Intell. Res.*, 14:303–358, 2001.

[21] Judy Goldsmith and Martin Mundhenk. Competition adds complexity. In John C. Platt, Daphne Koller, Yoram Singer, and Sam T. Roweis, editors, *Proceedings of the Twenty-First Annual Conference on Neural*

Information Processing Systems (NIPS 2007), pages 561–568. Curran Associates, Inc., 2007.

[22] John C. Harsanyi. Games with incomplete information played by "Bayesian" players, Part I. The Basic Model. *Management Science*, 14(3):159–182, 1967.

[23] John C. Harsanyi. Games with incomplete information played by "Bayesian" players, Part II. Bayesian Equilibrium Points. *Management Science*, 14(5):320–334, 1968.

[24] John C. Harsanyi. Games with incomplete information played by "Bayesian" players, Part III. The Basic Probability Distribution of the Game. *Management Science*, 14(7):486–502, 1968.

[25] Daphne Koller and Nimrod Megiddo. The complexity of two-person zero-sum games in extensive form. *Games and Economic Behavior*, 4(4):528–552, 1992.

[26] Daphne Koller, Nimrod Megiddo, and Bernhard von Stengel. Efficient computation of equilibria for extensive two-person games. *Games and Economic Behavior*, 14(2):247–259, 1996.

[27] Vojtech Kovarík, Martin Schmid, Neil Burch, Michael Bowling, and Viliam Lisý. Rethinking formal models of partially observable multiagent decision making. *Artif. Intell.*, 303:103645, 2022.

[28] Mark W. Krentel. The complexity of optimization problems. *J. Comput. Syst. Sci.*, 36(3):490–509, 1988.

[29] Urban Larsson, editor. *Games of No Chance 5*, volume 70 of *Mathematical Sciences Research Institute Publications*. Cambridge University Press, 2019.

[30] Joshua Letchford and Vincent Conitzer. Computing optimal strategies to commit to in extensive-form games. In David C. Parkes, Chrysanthos Dellarocas, and Moshe Tennenholtz, editors, *Proceedings of the Eleventh ACM Conference on Electronic Commerce (EC-2010)*, pages 83–92. ACM, 2010.

[31] Michael Maschler, Eilon Solan, and Shmuel Zamir. *Game Theory*. Cambridge University Press, 2nd edition, 2020.

[32] Martin Mundhenk, Judy Goldsmith, Christopher Lusena, and Eric Allender. Complexity of finite-horizon markov decision process problems. *J. ACM*, 47(4):681–720, 2000.

[33] Richard J. Nowakowski, editor. *Games of No Chance*, volume 29 of *Mathematical Sciences Research Institute Publications*. Cambridge University Press, 1996.

[34] Richard J. Nowakowski, editor. *More Games of No Chance*, volume 42 of *Mathematical Sciences Research Institute Publications*. Cambridge University Press, 2002.

[35] Richard J. Nowakowski, editor. *Games of No Chance 4*, volume 63 of *Mathematical Sciences Research Institute Publications*. Cambridge University Press, 2015.

[36] Héctor Palacios and Hector Geffner. Compiling uncertainty away in conformant planning problems with bounded width. *J. Artif. Intell. Res.*, 35:623–675, 2009.

[37] Jussi Rintanen. Complexity of planning with partial observability. In Shlomo Zilberstein, Jana Koehler, and Sven Koenig, editors, *Proceedings of the Fourteenth International Conference on Automated Planning and Scheduling (ICAPS 2004)*, pages 345–354. AAAI, 2004.

[38] Marcus Schaefer and Christopher Umans. Completeness in the polynomial-time hierarchy: A compendium. *SIGACT news*, 33(3):32–49, 2002.

[39] Aaron N Siegel. *Combinatorial game theory*. American Mathematical Society, 2013.

[40] Christopher Umans. Hardness of approximating Σ_2^p minimization problems. In Paul Beame, editor, *Proceedings of the Fourtieth Annual Symposium on Foundations of Computer Science (FOCS 1999)*, pages 465–474. IEEE Computer Society, 1999.

[41] Bernhard von Stengel. Efficient computation of behavior strategies. *Games and Economic Behavior*, 14(2):220–246, 1996.

[42] Bernhard von Stengel and Daphne Koller. Team-maxmin equilibria. *Games and Economic Behavior*, 21(1):309–321, 1997.

[43] Brian Hu Zhang, Gabriele Farina, and Tuomas Sandholm. Team belief DAG: generalizing the sequence form to team games for fast computation of correlated team max-min equilibria via regret minimization. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the Fortieth International Conference on Machine Learning (ICML 2023)*, volume 202 of *Proceedings of Machine Learning Research*, pages 814–839. PMLR, 2023.