Actes des 19es Journées d'Intelligence Artificielle Fondamentale et des 20es 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

19es Journées d'Intelligence Artificielle Fondamentale
et 20es Journées Francophones sur
la Planification, la Décision et l'Apprentissage
pour la conduite de systèmes

PFIA 2025

Table des matières

Éditorial

19es Journées d'Intelligence Artificielle Fondamentale et 20es 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-JFPDA est soutenue par le Collège Représentation et Raisonnement de l'AFIA.

Les thématiques de recherche de JIAF-JFPDA 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. Celui-ci était composé de 28 personnes et animé par Jean-Guy Mailly (IRIT, Université Toulouse Capitole), François Schwarzentruber (LIP, ENS Lyon) et Anaëlle Wilczynski (MICS, CentraleSupélec, Université Paris-Saclay).

Lors des 19èmes Journées d'Intelligence Artificielle Fondamentale et 20èmes Journées Francophones sur la Planification, la Décision et l'Apprentissage pour la conduite de systèmes (JIAF-JFPDA 2025), nous avons reçu 21 soumissions, dont 20 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 et raisonnement à partir de cas, argumentation, choix social, jeux, logique, planification). Nous avons également eu le plaisir d'accueillir un exposé invité donné par Vaishak Belle (University of Edinburgh), intitulé *Neuro-symbolic Systems for Responsible AI: Challenges and Opportunities*.

Jean-Guy Mailly, François Schwarzentruber, Anaëlle Wilczynski

Comité de programme

Présidence

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

Membres

— Francesco Belardinelli (Imperial College London) ;
— Aurélie Beynier (LIP6, Sorbonne Université) ;
— Élise Bonzon (LIPADE, Université Paris Cité) ;
— Nadjet Bourdache (GREYC, Université de Normandie) ;
— Olivier Buffet (INRIA / LORIA) ;
— Martin C. Cooper (IRIT - Université Paul Sabatier) ;
— Célia da Costa Pereira (Université Côte d'Azur) ;
— Tiago de Lima (Université d'Artois, CRIL CNRS) ;
— Sylvie Doutre (Université Toulouse Capitole - IRIT) ;
— Florence Dupin de Saint-Cyr (Université Paul Sabatier - IRIT) ;
— Alain Dutech (Loria - Inria) ;
— Sabine Frittella (INSA Centre Val de Loire) ;
— Hugo Gilbert (Lamsade - Université Paris Dauphine) ;
— Raïda Ktari (Aix-Marseille Université) ;
— Jérôme Lang (CNRS, LAMSADE, Université Paris-Dauphine) ;
— Jean Lieber (LORIA - INRIA Lorraine) ;
— Emiliano Lorini (CNRS, IRIT) ;
— Pierre Marquis (CRIL, U. Artois & CNRS - Institut Universitaire de France) ;
— Amedeo Napoli (LORIA Nancy, CNRS - Inria - Université de Lorraine) ;
— Wassila Ouerdane (MICS, CentraleSupélec, Université Paris-Saclay) ;
— Damien Pellier (Laboratoire d'Informatique de Grenoble) ;
— Julien Rossit (Université Paris Cité, LIPADE) ;
— Stéphanie Roussel (ONERA) ;
— Régis Sabbadin (INRA-UBIA) ;
— Vincent Thomas (LORIA) ;
— Srdjan Vesic (CRIL - CNRS) ;
— Thierry Vidal (LGP, UTTOP Tarbes) ;
— Bruno Zanuttini (GREYC, Université de Normandie).

Session 1 : Logique pour les ontologies, bases de connaissances et émotions

Sémantique catégorielle de logique de descriptions et raisonnement

Ludovic Brieulle[1], Chan Le Duc[1]

[1] Université Sorbonne Paris Nord, LIMICS, U1142, F-93000, Bobigny, France

{ludovic.brieulle}, {chan.leduc}@univ-paris13.fr

Résumé

Nous présentons une nouvelle logique de description (DL) permettant la négation ainsi qu'une procédure de raisonnement pour celle-ci. La construction de cette nouvelle logique est basée sur la réécriture de la sémantique usuelle ensembliste de la DL $\mathcal{ALC}$ en langage catégoriel où les concepts et subsomptions sont respectivement représentés par des objets et des flèches d'une catégorie. Cette réécriture nous donne une plus grande modularité sur la nouvelle sémantique des différents constructeurs comparée à la sémantique ensembliste. La modularité nous permet de définir une nouvelle DL NP*-complète avec la négation en éliminant les interactions entre les constructeurs logiques responsables de la complexité* EXPTIME *de la DL $\mathcal{ALC}$.*

Mots-clés

Logique de Description, Théorie des Catégories, Ontologies.

Abstract

We present a new description logic (DL) allowing for negation and a reasoning procedure for it. This construction is based on a rewriting of the usual set semantics of the DL $\mathcal{ALC}$ using category theory where concepts and subsumptions are respectively represented as objects and arrows of a category. This rewriting offers a categorical representation of the semantics that is more modular than the usual one based on set theory. This modularity allows us to define an NP*-complete logic without the interaction between logical constructors reponsible for* EXPTIME *complexity.*

Keywords

Description Logics, Category Theory, Reasoning.

1 Introduction

Les langages basés sur les logiques de description (DL) [1] tel que OWL [14] ou OWL 2 [8] sont largement utilisés pour représenter les ontologies sur les applications orientées sur la sémantique. La logique de description $\mathcal{ALC}$ est la plus petite logique fermée par la négation et contenant tous les constructeurs booléens. Nous pouvons alors appliquer les loi de De Morgan et utiliser l'écriture en forme normale négative (NNF) afin d'exprimer des formulations négatives comme celle mentionnées par Ceusters et al. [5], présentes dans certains termes définis dans SNOMED-CT ou dans l'ontologie OBO comme les formules lacks_of ou loss_of_consciousness.

Cependant, il est bien connu que $\mathcal{ALC}$ est ExpTime-complet avec des TBox générales [15, 2], son utilisation dans un cadre médical ou réaliste est donc exclue puisque le raisonneur pourrait prendre plusieurs minutes pour donner une réponse, voire le raisonnement pourrait ne pas aboutir du tout (*e.g.* erreur de mémoire).

La principale motivation de cet article est alors d'introduire une nouvelle logique dérivée de $\mathcal{ALC}$, qui conserve la négation et possède une complexité inférieure à ExpTime. Nous conjecturons que l'interaction entre la restriction universelle et existentielle est responsable de cette complexité, nous allons utiliser la modularité de la sémantique catégorielle de la restriction universelle pour obtenir une nouvelle logique NP-complet. Puisque nous souhaitons pouvoir toujours exprimer des formules du type « *un patient ne devrait pas être traité à la pénicilline* », qui peuvent s'exprimer de la façon suivante en NNF $\neg\exists$medicatedWith.Penicillin $\equiv$ $\forall$medicatedWith.$\neg$Penicillin il nous faudra faire attention à conserver la partie de la sémantique de la restriction universelle qui nous permet d'exprimer ce type de concept.

Dans ce but, nous définissons alors une nouvelle logique $\mathcal{ALC}_{\bar{\forall}}$ en utilisant une réécriture de la sémantique ensembliste usuelle grâce aux outils de la théorie des catégories, que nous appelons sémantique catégorielle. À cause de l'absence de l'appartenance dans cette théorie, nous avons besoin de plus de contraintes pour définir la sémantique des différents constructeurs en sémantique catégorielle. Ceci nous offre en contrepartie plus de possibilités pour définir différentes logiques. Par exemple, la subsomption $\exists R.C \sqcap \forall R.D \sqsubseteq \exists R.(C \sqcap D)$ (†) est conséquence directe de la sémantique des constructeurs $\exists$ et $\forall$ dans le cadre classique, *c'est-à-dire* que (†) découle de la définition de $(\exists R.C)^{\mathcal{I}}$ et $(\forall R.D)^{\mathcal{I}}$ pour toutes interprétations $\mathcal{I}$. Ainsi, une logique qui n'inclurait pas la relation (†) dans le cadre classique ne *peut pas* être définie. Il se trouve que dans la sémantique catégorielle, nous verrons que cette interaction est une partie indépendante de la sémantique des constructeurs $\exists$ et $\forall$ dans notre nouvelle sémantique catégorielle. Cela nous permet alors de définir notre nouvelle logique $\mathcal{ALC}_{\bar{\forall}}$ où cette interaction n'existe pas entièrement à cause d'une restriction universelle *affaiblie*, tout en conservant le reste des constructeurs qui interviennent dans $\mathcal{ALC}$. Cette nouvelle sémantique ne permettra alors de préserver qu'une partie de l'interaction entre $\exists$ et $\forall$, précisément : « *Si*

$C \sqcap D$ est insatisfiable, alors $\exists R.C \sqcap \forall R.D$ est insatisfiable également » (††) qui est *déjà* vraie dans le cadre classique grâce notamment à la relation (†). Cela entraîne forcément une perte d'expressivité qui peut être négligeable dans certains cas comme nous allons le voir ; nous préciserons plus tard l'expressivité perdue.

Exemple 1 *Considérons un patient* #X *qui souffre d'une infection et à qui de la pénicilline a été prescrite. Cependant, il est également précisé quelque part dans la base de connaissance que celui-ci est allergique à la pénicilline et à l'aspirine. La situation peut être résumée par la TBox $\mathcal{T}_1$ contenant uniquement l'axiome suivant :*

$$\mathsf{treatmentofX} \sqsubseteq \exists \mathsf{medicatedWith.Penicillin} \sqcap \forall \mathsf{medicatedWith.}\neg\mathsf{Penicillin} \sqcap \forall \mathsf{medicatedWith.}\neg\mathsf{Aspirin}$$

Sous la sémantique catégorielle de $\mathcal{ALC}_{\bar{\forall}}$, la relation (††) permet de déduire directement de la conjonction $\mathsf{Penicillin} \sqcap \neg\mathsf{Penicillin}$ *que la conjonction* $\exists \mathsf{medicatedWith.Penicillin} \sqcap \forall \mathsf{medicatedWith.}\neg\mathsf{Penicillin} \sqcap \forall \mathsf{medicatedWith.}\neg\mathsf{Aspirin}$ *est insatisfiable et par conséquent le concept* $\mathsf{treatmentofX}$ *l'est également par rapport à $\mathcal{T}_1$. Alors qu'en sémantique ensembliste, (†) aurait généré* $\mathsf{Penicillin} \sqcap \neg\mathsf{Penicillin} \sqcap \neg\mathsf{Aspirin}$ *pour trouver le même résultat.*

Dans cet exemple, nous avons montré que grâce à (††) nous pouvons découvrir un conflit*, ou* clash *en anglais, sans avoir besoin de regrouper toutes les conjonctions. Il est à noter que que (†) peut amener à créer des conjonctions de $n \geqslant 2$ conjoints alors que (††) ne fonctionne qu'avec des paires de conjoints. Ce cas de figure est notamment présent dans des ontologies médicales telles que OBO ou SNOMED.*

Il est à noter que l'absence d'individu de la logique n'est dû qu'à un soucis de simplification au départ dans le traitement de la nouvelle sémantique. Nous pensons pouvoir introduire à terme la notion d'individu dans la sémantique catégorielle en utilisant des objets « singleton » $\{a\}$ tel qu'aucun objets n'aient de flèche vers celui-ci, à l'exception de $\bot$.

Il n'y a à notre connaissance que peu de travaux sur l'utilisation de la théorie des catégories dans le cadre des logiques de description, nous en présentons brièvement certains d'entre eux. Spivak et Kent [16] utilisent la théorie des catégories pour définir un langage graphique de haut niveau comparable à OWL. Codescu, Mossakowski et Kutz [6] ont introduit un système permettant de formaliser un réseau d'ontologies alignées. Il y est montré que sur un tel réseau, le raisonnement pouvait se réduire à un raisonnement local sur une logique de description donnée appartenant à celui-ci. De ce fait, la sémantique déployée sur ces DL utilisent toujours la théorie des ensembles. Pour finir, Moss [13], Kupke et Pattinson [10] ont utilisé des F-coalgèbre pour obtenir une logique co-algébrique, où F est un foncteur sur la catégorie des ensembles. Ce foncteur est utilisé pour décrire les structures des modèles de Kripke. Ce dernier travail diffère du nôtre dans le sens où nous utilisons la théorie des catégories pour directement « encoder » la sémantique ensembliste des DL, et non pas pour décrire les modèles des DL.

Cet article est organisé de la façon suivante : nous commençons par présenter $\mathcal{ALC}$ muni de la sémantique ensembliste, ainsi que les outils de la théorie des catégories pour la logique en section 2. Puis nous établirons en détail dans la section 3 notre nouvelle sémantique catégorielle de $\mathcal{ALC}$ en utilisant les différents outils que la théorie des catégories nous fournit. En section 4, nous présentons notre nouvelle logique $\mathcal{ALC}_{\bar{\forall}}$ qui vient de l'affaiblissement du constructeur de la restriction universelle et donc de son interaction avec la restriction existentielle dans le contexte de la sémantique catégorielle, ainsi qu'un algorithme permettant de façon non-déterministe et polynomiale de répondre à la question de la satisfiabilité d'un concept par rapport avec une TBox en $\mathcal{ALC}_{\bar{\forall}}$. Pour finir, dans la section 5, nous ferons le bilan des résultats et parlerons des prochaines pistes de recherche que nous souhaitons explorer.

Toutes les preuves et détails concernant les résultats énoncés dans cet article sont disponibles en ligne dans lien fourni en référence [9].

2 Préliminaires

2.1 Sémantique ensembliste de la logique de description $\mathcal{ALC}$

Soit $\mathbf{C}, \mathbf{R}$ deux ensembles non vides disjoints de concepts atomiques et de rôles atomiques respectivement.

L'ensemble des concepts $\mathcal{ALC}$ est défini de manière inductive à partir de $\mathbf{C}$ et $\mathbf{R}$ de la façon suivante : tous les concepts atomiques sont des concepts $\mathcal{ALC}$, $\bot$ et $\top$ sont des concepts $\mathcal{ALC}$, si C et D sont des concepts $\mathcal{ALC}$ et R est un rôle atomique, alors $C \sqcap D, C \sqcup D, \neg C, \exists R.C, \forall R.D$ sont des concepts $\mathcal{ALC}$.

Soit C et D deux concepts $\mathcal{ALC}$ quelconques, alors $C \sqsubseteq D$ est appelée une *inclusion de concepts générale* (GCI). Un ensemble fini de GCI $\mathcal{O}$ est appelé une TBox. Une interprétation $\mathcal{I}$ est la donnée d'un ensemble non-vide $\Delta^{\mathcal{I}}$ appelé *domaine* et d'une application $\cdot^{\mathcal{I}}$ qui à chaque concept atomique $A \in \mathbf{C}$ associe un sous-ensemble $A^{\mathcal{I}}$ de $\Delta^{\mathcal{I}}$ et à chaque rôle atomique $R \in \mathbf{R}$ un sous-ensemble $R^{\mathcal{I}}$ de $\Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$; de sorte que : $\top^{\mathcal{I}} = \Delta^{\mathcal{I}}, \bot^{\mathcal{I}} = \varnothing, (C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}, (C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}, (\neg C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}, (\exists R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \exists y : (x,y) \in R^{\mathcal{I}} \wedge y \in C^{\mathcal{I}}\}$ et $(\forall R.C)^{\mathcal{I}} = \{x \in \Delta^{\mathcal{I}} \mid \forall y : (x,y) \in R^{\mathcal{I}} \implies y \in C^{\mathcal{I}}\}$.

Une interprétation $\mathcal{I}$ satisfait une CGI $C \sqsubseteq D$ si $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$. Si $\mathcal{I}$ satisfait toutes les GCIs de $\mathcal{O}$, alors $\mathcal{I}$ est un modèle de $\mathcal{O}$, noté $\mathcal{I} \models \mathcal{O}$. S'il existe un modèle de $\mathcal{O}$, alors $\mathcal{O}$ est dit consistent. Un concept C est *satisfiable* (en sémantique ensembliste) par rapport à une Tbox $\mathcal{O}$, s'il existe $\mathcal{I} \models \mathcal{O}$ tel que $C^{\mathcal{I}} \neq \varnothing$.

2.2 Théorie des catégories pour la logique

L'objet principal utilisé dans cet article est une catégorie. Celle-ci possède une définition moins restrictive que celle pour un ensemble, elle est suffisamment abstraite pour donner une liberté sur les objets et les flèches qui entrent en

jeux. Les propriétés que nous allons définir pour établir la nouvelle sémantique que nous utiliserons héritera de cette « liberté » ou modularité est ce qui nous permettra d'obtenir le nouveau langage présenté dans ce texte.

Une *catégorie* $\mathscr{C}$ est la donnée d'une collection de $\mathscr{C}$-objets, notée $\mathsf{Ob}(\mathscr{C})$, d'une collection $\mathscr{C}$-flèches, notée $\mathsf{Hom}(\mathscr{C})$ – de deux applications dom et cod de $\mathsf{Hom}(\mathscr{C})$ vers $\mathsf{Ob}(\mathscr{C})$ qui à une flèche f associent deux objets tels que si on écrit $A := \mathsf{dom}\, f$ et $B := \mathsf{cod}\, f$ alors on écrit $f : A \longrightarrow B$. Pour tout objet $A \in \mathsf{Ob}(\mathscr{C})$, il existe une flèche identité 1_A. De plus, à toute paire de flèches $\langle g, f\rangle$ telle que $\mathsf{cod}\, f = \mathsf{dom}\, g$, nous associons la composition de flèche $g \circ f : \mathsf{dom}\, f \longrightarrow \mathsf{cod}\, g$. L'opération de composition est associative, *i.e.* $h \circ (g \circ f) = (h \circ g) \circ f = h \circ g \circ f$ et pour $f : A \longrightarrow B$, $1_b \circ f = f \circ 1_A = f$.

Un *foncteur* F d'une catégorie $\mathscr{C}$ vers une autre catégorie $\mathscr{D}$ est une application qui à tout objet X de $\mathscr{C}$ associe un objet $F(X)$ de $\mathscr{D}$ et à toute flèche $f : X \longrightarrow Y$ de $\mathscr{C}$ associe une flèche $F(f) : F(X) \longrightarrow F(Y)$. Les propriétés suivantes sont toujours vraies : $F(1_X) = 1_{F(X)}$ et $F(f \circ g) = F(f) \circ F(g)$.

Soit A et B des objets de $\mathscr{C}$, le *produit* de A et B est un objet $A \times B$ et deux flèches $\pi_1 : A \times B \longrightarrow A$, $\pi_2 : A \times B \longrightarrow B$ tels que pour chaque paires de flèches $f_A : X \longrightarrow A, f_B : X \longrightarrow B$, il existe une unique flèche $f_{A\times B} : X \longrightarrow A \times B$ telle que $f_A = \pi_1 \circ f_{A\times B}$ et $f_B = \pi_2 \times f_{A\times B}$.

Le *co-produit* de A et B est l'objet $A + B$ et deux flèches $\iota_1 : A \longrightarrow A + B, \iota_2 : B \longrightarrow A + B$ tels pour $g_A : A \longrightarrow X, g_B : B \longrightarrow X$ il existe une unique flèche $f_{A+B} : A+B \longrightarrow X$ telle que $g_A = f_{A+B} \circ \iota_1$ et $g_B = f_{A+B} \circ \iota_2$.

Soit $F : \mathscr{D} \longrightarrow \mathscr{C}$ et $G : \mathscr{C} \longrightarrow \mathscr{D}$ deux foncteurs tels que pour tout objets c de $\mathscr{C}$ et d de $\mathscr{D}$, il y a une bijection ensembliste $\mathsf{Hom}_{\mathscr{C}}(F(D), C) \simeq \mathsf{Hom}_{\mathscr{D}}(D, G(C))$. Nous disons alors que F est l'adjoint à gauche de G et G l'adjoint à droite de F.

3 Sémantique catégorielle de $\mathcal{ALC}$

Une approche classique pour donner une sémantique ensembliste à la logique propositionnelle est d'utiliser l'algèbre booléenne $\langle \mathcal{P}(D), \subseteq\rangle$, où $\mathcal{P}(D)$ est l'ensemble des parties de D et où $\mathbf{1} := D$ et $\mathbf{0} := \varnothing$. L'algèbre est munie d'une fonction de vérité V qui à chaque phrase atomique associe alors une valeur de vérité dans $\mathcal{P}(D)$. Elle est étendue aux phrases complexes qui utilisent $\wedge, \vee$ et $\neg$ grâce à l'intersection $\cap$, l'union $\cup$ et le complément $\sim$ sur $\mathcal{P}(D)$ en tenant compte de l'appartenance $\in$.

Une extension de cette algèbre booléenne est un treillis appelé l'algèbre de Heyting $\langle H, \subseteq\rangle$. Dans ce contexte, la fonction de vérité V associe à $x \wedge y$ et $x \vee y$ une *plus grande limite inférieure* et une *plus petite limité supérieure* respectivement.

Dans notre cas, Le Duc [11] et Brieulle [4], nous avons remplacé le treillis de l'algèbre par une catégorie de sorte que les concepts/rôles soient des objets et la relation d'ordre $\subseteq$ est remplacée par les flèches de cette catégorie.

Notre but est d'introduire une nouvelle sémantique pour la logique de description $\mathcal{ALC}$ en utilisant les notions de la théorie des catégories que nous venons de présenter.

À cette fin, nous aurons besoin de deux catégories, $\mathscr{C}_c$ une catégorie pour représenter les concepts et leurs interactions par rapport aux différents constructeurs et $\mathscr{C}_r$ une catégorie pour représenter les rôles et comment ils influencent les interactions entre les différents concepts. Nous aurons également besoin de deux foncteurs $\Pi_\ell, \Pi_r : \mathscr{C}_r \longrightarrow \mathscr{C}_c$ qui joueront le rôle des projecteurs π_ℓ et π_r pour une relation quelconque. Par exemple, considérons la relation $R = \{(x_1, y_1), (x_2, y_2)\}$ alors les projections nous donnent $\pi_\ell(R) = \{x_1, x_2\}$ et $\pi_r(R) = \{y_1, y_2\}$.

Nous commençons d'abord par définir ce que nous appelons des catégories de *syntaxe* qui serviront de base sur lesquelles nous ajouterons alors les structures nécessaires à exprimer la sémantique des différents constructeurs qui entrent en jeu.

Définition 1 *Soit* $\mathbf{C}$ *et* $\mathbf{R}$ *deux ensembles non vides disjoints de concepts atomiques et de rôles atomiques respectivement. Les catégories syntaxiques de concepts* $\mathscr{C}_c$ *et de rôles* $\mathscr{C}_r$ *sont définies à partir de la signature* $\langle \mathbf{C}, \mathbf{R}\rangle$ *de la façon suivante :*

1. *Les objets de* $\mathscr{C}_c$ *sont les concepts atomiques* $C \in \mathbf{C}$*, l'objet terminal* $\top$ *et l'objet initial* $\bot$*. Les flèches de* $\mathscr{C}_c$ *sont les flèches identités* 1_C *pour tout objet* C *ainsi que les flèches initiales* $\bot \longrightarrow C$ *et terminales* $C \longrightarrow \top$ *pour tout objet* C *de* $\mathscr{C}_c$*.*
2. *Les objets de* $\mathscr{C}_r$ *sont les rôles atomiques* $R \in \mathbf{R}$*, les objets* $\mathfrak{R}_{\exists R.C}$ *pour tout objet* C *de* $\mathscr{C}_c$ *et* $R \in \mathbf{R}$ *ainsi que l'objet terminal* $R_\top$ *et initial* $R_\bot$*. Les flèches de* $\mathscr{C}_r$ *sont les flèches identités* 1_R*,* $R \longrightarrow \mathfrak{R}_\top, \mathfrak{R}_\bot \longrightarrow R$ *pour tout objet* R *de* $\mathscr{C}_r$ *ainsi que* $\mathfrak{R}_{\exists R.C} \longrightarrow R$ *pour tout objet* C *de* $\mathscr{C}_c$ *et* $R \in \mathbf{R}$*.*
3. *Nous définissons deux foncteurs* $\Pi_\ell, \Pi_r : \mathscr{C}_r \longrightarrow \mathscr{C}_c$ *qui envoient chaque objet de* $\mathscr{C}_r$ *vers un objet de* $\mathscr{C}_c$ *et chaque flèche de* $\mathscr{C}_r$ *vers une flèche de* $\mathscr{C}_c$*, en particulier pour tout objet* R *de* $\mathscr{C}_r$ *il existe des objets* $\Pi_\ell(R), \Pi_r(R)$ *dans* $\mathscr{C}_c$*.*

   De plus, si $R \longrightarrow \mathfrak{R}_\bot$ *est dans* $\mathscr{C}_r$ *alors* $\Pi_{\ell,r}(R) \longrightarrow \bot$ *est dans* $\mathscr{C}_c$ *pour* R *un objet de* $\mathscr{C}_r$*.*

Il est à noter que $\mathsf{Ob}(\mathscr{C})$ et $\mathsf{Hom}(\mathscr{C})$ ne sont pas forcément des ensembles en général, mais ce sera le cas pour nos catégories.

Il nous faut instancier $\mathscr{C}_c$ et $\mathscr{C}_r$ pour obtenir des catégories qui illustrent les contraintes imposées par la sémantique des différents constructeurs logiques et les axiomes de $\mathcal{O}$. Nous commençons par réécrire la sémantique de chaque constructeur dans la théorie des catégories.

Axiomes.

(F1) $C \sqsubseteq D \in \mathcal{O}$ alors $C \longrightarrow D \in \mathsf{Hom}(\mathscr{C}_c)$.

Conjonction. Le produit catégoriel (voir section 2) est utilisé en remplaçant $\times$ par $\sqcap$ pour illustrer la conjonction de deux concepts $C \sqcap D$, avec pour conséquences :

(F2) $C \sqcap D \in \mathsf{Ob}(\mathscr{C}_c) \implies C \sqcap D \longrightarrow C, C \sqcap D \longrightarrow D \in \mathsf{Hom}(\mathscr{C}_c)$

(F3) $\forall X \in \mathsf{Ob}(\mathscr{C}_c), X \longrightarrow C, X \longrightarrow D \in \mathsf{Hom}(\mathscr{C}_c) \implies X \longrightarrow C \sqcap D \in \mathsf{Hom}(\mathscr{C}_c)$

Le premier point découle des projections du produit et le deuxième de sa propriété universelle. Autrement dit, $C \sqcap D$ est le plus grand objet contenu à la fois dans C et D. De plus, si chaque flèche ($\longrightarrow$) est remplacée par la subsomption ($\sqsubseteq$) alors les propriétés usuelles de la sémantique de $\sqcap$ par une interprétation $\mathcal{I}$ sont toujours respectées, voir lemme B1 [9].

Disjonction. Pour la conjonction, nous utilisons le co-produit catégoriel, où $\sqcup$ remplace $+$, $C \sqcup D$ (voir section 2) qui s'illustre par les deux points suivants, conséquences des propriétés du dit co-produit :

(F4) $C \sqcup D \in \mathsf{Ob}(\mathscr{C}_c) \implies C \longrightarrow C \sqcup D, D \longrightarrow C \sqcup D \in \mathsf{Hom}(\mathscr{C}_c)$

(F5) $\forall X \in \mathsf{Ob}(\mathscr{C}_c), C \longrightarrow X, D \longrightarrow X \in \mathsf{Hom}(\mathscr{C}_c) \implies C \sqcup D \longrightarrow X \in \mathsf{Hom}(\mathscr{C}_c)$

Nous pouvons alors voir $C \sqcup D$ comme le plus petit objet contenant à la fois C et D.

De même que pour la conjonction, si nous remplaçons les flèches ($\longrightarrow$) par des subsomptions ($\sqsubseteq$) alors les propriétés usuelles de la sémantique de la disjonction $\sqcup$ par une interprétation $\mathcal{I}$ sont respectées, voir lemme B2 [9].

Négation. Nous devons utiliser la conjonction et la disjonction, donc le produit et le co-produit, pour pouvoir définir la négation. Pour tout $C \in \mathsf{Ob}(\mathscr{C}_c)$, nous définissons alors $\neg C$ comme l'objet tel que :

(F6) $C \sqcap \neg C \longrightarrow \bot, C \sqcup \neg C \longrightarrow \top \in \mathsf{Hom}(\mathscr{C}_c)$

(F7) $\forall X \in \mathsf{Ob}(\mathscr{C}_c), C \sqcap X \longrightarrow \bot \in \mathsf{Hom}(\mathscr{C}_c) \implies X \longrightarrow \neg C \in \mathsf{Hom}(\mathscr{C}_c)$

(F8) $\forall X \in \mathsf{Ob}(\mathscr{C}_c), \top \longrightarrow C \sqcup X \in \mathsf{Hom}(\mathscr{C}_c) \implies \neg C \longrightarrow X \in \mathsf{Hom}(\mathscr{C}_c)$

Il est possible de voir $\neg C$ comme le plus grand objet disjoint de C. Comme pour les constructeurs précédents, les propriétés de la sémantique de la négation $\neg$ par une interprétation $\mathcal{I}$ sont toujours conservées en remplaçant les flèches ($\longrightarrow$) par des subsomptions ($\sqsubseteq$), lemme B3 [9].

Restriction existentielle.

Pour la restriction existentielle, nous avons besoin des objets de rôles $\mathfrak{R}_{\exists R.C}$ et des foncteurs Π_ℓ, Π_r définis dans la définition 1. La sémantique de $\exists$ est alors caractérisée par les deux points suivants :

(F9) Si $\exists R.C \in \mathsf{Ob}(\mathscr{C}_c)$ alors $\mathfrak{R}_{\exists R.C} \in \mathsf{Ob}(\mathscr{C}_r)$, $\mathfrak{R}_{\exists R.C} \longrightarrow R \in \mathsf{Hom}(\mathscr{C}_r)$ et $\Pi_\ell(\mathfrak{R}_{\exists R.C}) \longleftrightarrow \exists R.C \in \mathsf{Hom}(\mathscr{C}_c), \Pi_r(\mathfrak{R}_{\exists R.C}) \longrightarrow C \in \mathsf{Hom}(\mathscr{C}_c)$

(F10) Si $R' \longrightarrow R \in \mathsf{Hom}(\mathscr{C}_r), \Pi_r(R') \longrightarrow C \in \mathsf{Hom}(\mathscr{C}_c)$ alors $\Pi_\ell(R') \longrightarrow \exists R.C \in \mathsf{Hom}(\mathscr{C}_c)$

Il est à noter que MacLane et al. [12] ont défini pour les restriction existentielle non-qualifiée $\exists R$ comme l'adjoint à gauche (voir section 2) d'un foncteur inverse $\Pi^\star$. Cependant nous avons dû utiliser des propriétés différentes pour l'adapter à nos catégories et à la restriction existentielle qualifiée.

Restriction universelle. Puisque nous avons déjà défini la sémantique catégorielle de $\neg$ et $\exists$, il est alors possible définir $\forall R.C$ comme $\neg\exists R.\neg C$.

(F11) Si $\forall R.C \in \mathsf{Ob}(\mathscr{C}_c)$ alors $\forall R.C \longleftrightarrow \neg\exists R\neg C \in \mathsf{Hom}(\mathscr{C}_c)$

Comme pour les précédents constructeurs, remplacer ($\longrightarrow$) par ($\sqsubseteq$) redonne les propriétés usuelles de tous les constructeurs présentés, voir lemme B4 et B5 [9].

Si nous prenons en compte uniquement les points F1 à F11 alors il nous manque les interactions entre les différents constructeurs pour couvrir la totalité de la sémantique ensembliste de $\mathcal{ALC}$. Plus précisément, la distributivité entre $\sqcap$ et $\sqcup$ et le regroupement entre $\exists$ et $\forall$. Pour compléter la sémantique, nous commençons par ajouter la propriété suivante :

(F12) *(Distributivité)* Si $C \sqcap (D \sqcup E) \in \mathsf{Ob}(\mathscr{C}_c)$ alors $C \sqcap (D \sqcup E) \longrightarrow (C \sqcap D) \sqcup (C \sqcap E) \in \mathsf{Hom}(\mathscr{C}_c)$.

Si nous remplaçons ($\longrightarrow$) par ($\sqsubseteq$), il est facile de voir que nous gardons les propriétés usuelles de la ditributivités de part les propriétés de $\cap$ et $\cup$, voir Lemme B6 [9].

Enfin, la dernière propriété qu'il nous reste à ajouter concerne l'interaction entre $\exists$ et $\forall$, interaction qui est déduite de la propriété suivante en combinaison avec les précédentes :

(F13) Si $R' \longrightarrow R \in \mathsf{Hom}(\mathscr{C}_r)$ et $\Pi_\ell(R') \longrightarrow \forall R.D \in \mathsf{Hom}(\mathscr{C}_c)$ alors $\Pi_r(R') \longrightarrow D \in \mathsf{Hom}(\mathscr{C}_c)$.

Maintenant que nous avons toutes les propriétés nécessaires pour donner la sémantique aux objets de $\mathscr{C}_c$ et $\mathscr{C}_r$, il va nous être possible d'effectuer des raisonnements à partir de ces propriétés. Nous souhaitons pouvoir déterminer la satisfiabilité d'un concept par rapport avec une TBox spécifique $\mathcal{O}$. Jusqu'à présent, nous n'avons à notre disposition que des catégories de syntaxe avec très peu de structure, ne prenant pas en compte les axiomes d'une TBox ou les différents constructeurs de $\mathcal{ALC}$. Cela dit, nous venons d'introduire les propriétés nécessaires permettant d'exprimer la sémantique des différents constructeurs mais également qui prennent en compte les axiomes d'une TBox $\mathcal{O}$. Nous allons formaliser la définition de catégories d'ontologies qui prennent en compte ces propriétés.

Définition 2 (Catégorie d'ontologie) *Soit $\mathcal{O}$ une TBox en $\mathcal{ALC}$ avec $\mathbf{C}, \mathbf{R}$ des ensembles non-vides disjoints de concepts atomiques et de rôles atomiques respectivement, ainsi que C_0 un concept en $\mathcal{ALC}$. Les* catégories d'ontologie de C_0 par rapport à $\mathcal{O}$ *sont les catégories de syntaxe $\mathscr{C}_c\langle C_0, \mathcal{O}\rangle$ et $\mathscr{C}_r\langle C_0, \mathcal{O}\rangle$ de signature $\langle \mathbf{C}, \mathbf{R}\rangle$ telles que les propriétés F1 à F13 soient satisfaites et telles que $C_0 \in \mathsf{Ob}(\mathscr{C}_c\langle C_0, \mathcal{O}\rangle)$.*

Jusqu'à mention du contraire, nous fixons C_0 et $\mathcal{O}$ comme un concept en $\mathcal{ALC}$ et une TBox en $\mathcal{ALC}$ respectivement. Les flèches du lemme suivants nous donnent les propriétés complètes pour la restriction existentielle, universelle leurs intéractions ainsi que celles avec la négation, la conjonction et la disjonction.

Lemme 1 (Propriétés existentielles et universelles) *Nous supposons que tous les objets qui interviennent dans les résultats sont bien la catégorie $\mathscr{C}_c\langle C_0, \mathcal{O}\rangle$. Alors si $\mathscr{C}_c\langle C_0, \mathcal{O}\rangle$ satisfait F1 à F13, les propriétés suivantes sont vraies.*

$$C \longrightarrow \bot \implies \exists R.C \longrightarrow \bot \quad (1)$$
$$C \longrightarrow D \implies \exists R.C \longrightarrow \exists R.C \quad (2)$$
$$C \longrightarrow D \implies \forall R.C \longrightarrow \forall R.D \quad (3)$$
$$\exists R.C \sqcap \forall R.D \longrightarrow \exists R.(C \sqcap D) \quad (4)$$
$$(\text{F10}) \textit{ et } (\text{F11}) \iff \quad (5)$$
$$(C \sqcap D \longrightarrow \bot \implies \exists R.C \sqcap \forall R.D \longrightarrow \bot)$$
$$(\text{F10}) \textit{ et } (\text{F11}) \iff \quad (6)$$
$$(\top \longrightarrow C \sqcup D \implies \top \longrightarrow \exists R.C \sqcup \forall R.D)$$

Pour les points (5) et (6), il est important de préciser que les équivalences sont obtenues sans l'intervention de F13 ; cela nous servira plus tard. Il est également à préciser que les constructeurs $\sqcap$ et $\sqcup$ sont tous les deux commutatifs, associatifs et idempotents, ces propriétés sont conséquences des définitions du produit et du co-produit utilisé dans les points F2 à F5.
Notons aussi que les propriétés de la négation combinées avec celles de la disjonction, la conjonction, les restrictions universelles et existentielles permettent d'obtenir les lois de De Morgan et, à terme, d'écrire tous les concepts en *forme normale négative* (NNF).

Lemme 2 (Propriétés de la négation catégorielle) *Sous l'hypothèse que tous les objets sont présents dans la catégorie d'ontologie $\mathscr{C}_c\langle C_0, \mathcal{O}\rangle$, alors les flèches suivantes sont présentes dans celle-ci.*

$$C \longleftrightarrow \neg\neg C \quad (7)$$
$$C \longrightarrow \neg D \iff D \longrightarrow \neg C \quad (8)$$
$$C \sqcap D \longrightarrow \bot \iff C \longrightarrow \neg D \textit{ (or } D \longrightarrow \neg C) \quad (9)$$
$$\top \longrightarrow \neg C \sqcup D \iff C \longrightarrow D \quad (10)$$
$$\neg(C \sqcap D) \longleftrightarrow \neg C \sqcup \neg D \quad (11)$$
$$\neg(C \sqcup D) \longleftrightarrow \neg C \sqcap \neg D \quad (12)$$
$$\neg(\forall R.C) \longleftrightarrow \exists R.\neg C \quad (13)$$
$$\neg(\exists R.C) \longleftrightarrow \forall R.\neg C \quad (14)$$

Une notion importante qui est à la base de ce que nous souhaitons obtenir est la notion d'*indépendance* d'une propriété par rapport à toutes les autres. Nous disons qu'une propriété/flèche est indépendante si elle n'est pas conséquence des autres propriétés, elles sont nécessaires pour obtenir la totalité de la sémantique des constructeurs. C'est notamment le cas pour F12 par rapport aux propriétés F2 à F4. Ceci explique pourquoi nous avons expliciter la distributivité de $\sqcap$ et $\sqcup$ alors qu'elle est la conséquence des définitions de $\cap$ et $\cup$ en sémantique ensembliste.
Cependant, la propriété indépendante qui nous intéresse est F13. Pour illustrer sont indépendance, nous allons construire une catégorie qui satisfait toutes les propriétés de F1 à F12. Nous verrons alors que la flèche du point 4 du lemme 1 ne peut être obtenue si l'intervention de la flèche F13 n'est pas autorisée.

Exemple 2 *Soit $\mathsf{C}_0 = (\exists \mathsf{R}.\mathsf{D} \sqcap \forall \mathsf{R}.\mathsf{C}) \sqcup \exists \mathsf{R}.(\mathsf{D} \sqcap \mathsf{C})$. Soit $\mathscr{C}_c$ et $\mathscr{C}_r$ les catégories de concepts et de rôles de $\langle \mathsf{C}_0, \varnothing\rangle$ respectivement. Pour faciliter les notations, nous posons $\mathsf{X} = \exists \mathsf{R}.\mathsf{D} \sqcap \forall \mathsf{R}.\mathsf{C}$, $\mathsf{Y} = \neg\exists \mathsf{R}.\neg\mathsf{C} \sqcap \exists \mathsf{R}.\neg\mathsf{C}$ et $\mathsf{Z} = \neg\exists \mathsf{R}.\neg\mathsf{C} \sqcup \exists \mathsf{R}.\neg\mathsf{C}$. Par soucis de clarté, nous ignorerons dans la suite les flèches identités, terminales et initiales ainsi que les compositions inutiles dans la recherche d'une flèche entre $\exists R.C \sqcap \forall R.D$ et $\exists R.(C \sqcap D)$. D'après la propriété F9, la catégorie $\mathscr{C}_r$ admet les flèches $\mathsf{R}_{(\exists \mathsf{R}.\mathsf{D})} \longrightarrow \mathsf{R}$, $\mathsf{R}_{(\exists \mathsf{R}.\neg\mathsf{C})} \longrightarrow \mathsf{R}$, $\mathsf{R}_{(\exists \mathsf{R}.(\mathsf{D} \sqcap \mathsf{C}))} \longrightarrow \mathsf{R}$.
En considérant uniquement que les propriétés F1 à F9, ainsi que la propriété F11 et les flèches qui découlent de celles-ci, nous ne pouvons pas prouver l'appartenance de $\mathsf{X} \longrightarrow \exists \mathsf{R}.(\mathsf{D} \sqcap \mathsf{C})$ à $\mathsf{Hom}(\mathscr{C}_c)$. Incluons à partir de maintenant la propriété F10). Pour qu'elle puisse être utiliser, nous devons commencer par ajouter arbitrairement à $\mathsf{Ob}(\mathscr{C}_r)$ un objet R' avec $\Pi_\ell(\mathsf{R}') \longrightarrow \mathsf{X}$. Pour que nous puissions obtenir $\mathsf{X} \longrightarrow \exists \mathsf{R}.(\mathsf{D} \sqcap \mathsf{C}) \in \mathsf{Hom}(\mathscr{C}_c)$ de F10, il nous faut également la flèche $\Pi_r(R') \longrightarrow X$ l'existence n'a pas pu être prouvée avec les propriétés F1-F11. Cependant, l'existence de la flèche $\mathsf{X} \longrightarrow \exists \mathsf{R}.(\mathsf{D} \sqcap \mathsf{C})$ découle de la propriété F13. Ainsi, F13 est bien indépendante des propriétés F1-F11.*

Tout comme en sémantique ensembliste, nous pouvons définir une forme normale négative (NNF) d'un concept C, que nous écrivons $\mathsf{NNF}(C)$, comme une forme où la négation n'apparaît que devant des concepts atomiques. Il est possible de convertir chaque objet vers sa forme NNF en utilisant les lois de De Morgan, présentes dans le lemme 2, et la propriété F11. Ainsi, par la suite, nous pouvons supposer que si le symbole $\neg$ est devant un concept C, alors C est un concept atomique. Sous cette hypothèse, nous pouvons réduire le nombre de propriétés nécessaires pour caractériser la sémantiques catégorielles en terme de satisfiabilité. Nous rappelons que dans une catégorie d'ontologie la présence de la flèche $C \longrightarrow D$ dans la catégorie est équivalent à la présence de $\top \longrightarrow \neg C \sqcup D$ comme le montre la propriété (10) dans le lemme 2.

Définition 3 *Soit C_0 un concept en $\mathcal{ALC}$ et $\mathcal{O}$ une TBox en $\mathcal{ALC}$. Soit $\mathscr{C}_c\langle C_0, \mathcal{O}\rangle$ une catégorie de concept d'ontologie telle que tous ses objets soient écrits en* NNF *et que les lois de conjonctions et disjonctions (commutatives, associatives et idempotentes) soient toujours valides dans celle-ci. La catégorie $\mathscr{C}_c\langle C_0, \mathcal{O}\rangle$ est dite* NNF-saturée *si les propriétés suivantes sont vérifiées.*

- *3.1* $C \sqsubseteq D \in \mathcal{O}$ *implique* $\top \longrightarrow \neg C \sqcup D \in \mathsf{Hom}(\mathscr{C}_c\langle C_0, \mathcal{O}\rangle)$.
- *3.2* $C \sqcap D \in \mathsf{Ob}(\mathscr{C}_c)$ *implique* $C \sqcap D \longrightarrow C, C \sqcap D \longrightarrow D \in \mathsf{Hom}(\mathscr{C}_c)$.
- *3.3* $X \longrightarrow C, X \longrightarrow D \sqcup E \in \mathsf{Hom}(\mathscr{C}_c)$ *impliquent* $C \sqcap D, C \sqcap E \in \mathsf{Ob}(\mathscr{C}_c)$.
- *3.4* $X \longrightarrow C, X \longrightarrow D \sqcup E, C \sqcap D \longrightarrow \bot, C \sqcap E \longrightarrow \bot \in \mathsf{Hom}(\mathscr{C}_c)$ *impliquent* $X \longrightarrow \bot \in \mathsf{Hom}(\mathscr{C}_c)$.

3.5 $X \longrightarrow A, X \longrightarrow \neg A \in \mathsf{Hom}(\mathscr{C}_c)$ *impliquent* $X \longrightarrow \bot \in \mathsf{Hom}(\mathscr{C}_c)$.

3.6 $\exists R.C \in \mathsf{Ob}(\mathscr{C}_c)$ *implique* $C \in \mathsf{Ob}(\mathscr{C}_c)$.

3.7 $\exists R.C \in \mathsf{Ob}(\mathscr{C}_c)$, $C \longrightarrow \bot \in \mathsf{Hom}(\mathscr{C}_c)$ *impliquent* $\exists R.C \longrightarrow \bot \in \mathsf{Hom}(\mathscr{C}_c)$.

3.8 $X \longrightarrow \exists R.C, X \longrightarrow \forall R.C' \in \mathsf{Hom}(\mathscr{C}_c)$ *impliquent* $X \longrightarrow \exists R.(C \sqcap C') \in \mathsf{Hom}(\mathscr{C}_c)$.

Comme les propriétés énoncées sont uniquement nécessaire pour découvrir des flèches avec pour codomaine $\bot$, cela les rend *de facto* plus faibles que les propriétés F1 à F13 qui servent à décrire l'entièreté de la sémantique engendrée par l'ontologie. Par exemple, la propriété 3.2 est une partie de la définition de la conjonction, plus précisément F2. La propriété 3.3 ne rajoute que le conjoint du codomaine de la flèche F12, 3.4 est conséquence de F12, F6 et la loi de composition. Le point 3.5 est également une conséquence de F3, F6 et qu'il s'agit d'une catégorie. L'implication 3.5 est encore une fois le résultant des propriétés F3 et F6, 3.6 quant à elle est déduite de la flèche $\Pi_r(\mathfrak{R}_{\exists R.C}) \longrightarrow C$ de la propriété F9. Enfin, 3.7 est l'illsutration de l'implication 1 du lemme 2 et 3.8 est le point 4 qui est lui-même conséquence de F13.

Définition 4 (Catégorie minimale) *Une catégorie d'ontologie $\mathscr{C}_c$ est dite* minimale *si pour toute catégorie d'ontologie $\mathscr{C}'_c$, nous avons* $\mathsf{Hom}(\mathscr{C}_c) \subseteq \mathsf{Hom}(\mathscr{C}'_c)$.

Comme chaque objet d'une catégorie est muni de sa flèche identité, alors $\mathsf{Hom}(\mathscr{C}_c) \subseteq \mathsf{Hom}(\mathscr{C}'_c)$ implique $\mathsf{Ob}(\mathscr{C}_c) \subseteq \mathsf{Ob}(\mathscr{C}'_c)$, d'où nous pouvons nous concentrer uniquement sur les flèches qui impliquent l'existence des objets dans la catégorie en général.
La minimalité est ici pour s'assurer qu'aucun objet ou flèche autre que ce qui est nécessaire pour respecter les propriétés F1 à F13 ne soient ajoutés de manière arbitraire à la catégorie, rendant alors la sémantique complètement caduque. L'existence de catégories saturées et minimales en elle-même découle du fait que ces catégories peuvent être construites de manière algorithmique en utilisant les propriétés 3.1 à 3.8, tout en tenant compte des propriétés usuelles d'une catégorie, notamment la composition de flèche et les propriétés liées aux deux objets terminale et initiale respectivement.
Il peut être remarqué que les propriétés liées exclusivement à la disjonction $\sqcup$ ne sont pas explicites, mais incluses dans les propriétés liées à la distributivité. Il est effectivement possible de montrer que (i) $C \sqcup D \longrightarrow \bot$ implique $C \longrightarrow \bot, D \longrightarrow \bot$ et (ii) $C \longrightarrow \bot, D \longrightarrow \bot$ impliquent $C \sqcup D \longrightarrow \bot$ en utilisant uniquement les propriétés liées à la distributivité dans la définition 3.
Nous sommes alors en capacité d'énoncer clairement la satisfiabilité catégorielle d'un concept par rapport à une TBox.

Définition 5 *Soit C_0 un concept en $\mathcal{ALC}$ et $\mathcal{O}$ une TBox en $\mathcal{ALC}$. Soit $\mathscr{C}_c\langle C_0, \mathcal{O}\rangle$ la catégorie d'ontologie NNF saturée minimale de $\mathcal{O}$ et C_0. Alors, le concept C_0 est* insatisfiable catégoriellement par rapport à $\mathcal{O}$ *si la flèche $C_0 \longrightarrow \bot$ est dans* $\mathsf{Hom}(\mathscr{C}_c\langle C_0, \mathcal{O}\rangle)$.

Le théorème suivant assure qu'il y a une équivalence sémantique entre la satisfiabilité ensembliste et catégorielle d'un concept par rapport à une TBox $\mathcal{O}$.

Théorème 1 (Équivalence sémantique) *Soit $\mathcal{O}$ une TBox en $\mathcal{ALC}$ et soit C_0 un concept en $\mathcal{ALC}$. Alors C_0 est insatisfiable catégoriellement par rapport avec $\mathcal{O}$ si et seulement si C_0 est insatisiafable (en ensembliste) par rapport à $\mathcal{O}$.*

Esquisse de preuve ($\Longleftarrow$). La direction de la sémantique catégorielle à la sémantique ensembliste est assez directe à prouver, nous considérons chaque propriétés de la définition 3 et montrons pour chaque flèche $X \longrightarrow Y \in \mathsf{Hom}(\mathscr{C}_c)$ ajouté, alors $\mathcal{O} \models X \sqsubseteq Y$. Cela se fait grâce au lemme B1 à B6 de [9], ainsi que le lemme 1 et 2. Autrement dit, chaque inclusion de concept qui correspond à une flèche ajoutée par la définition 3 est validée par les axiomes de $\mathcal{O}$ sous la sémantique ensembliste.
($\Longrightarrow$). La direction ensembliste à catégorielle va nous demander un peu plus de travail. Nous allons seulement énoncer les grandes lignes et invitons le lecteur ou la lectrice à consulter la preuve complète dans le rapport technique [9].
Nous commençons par générer grâce à l'algorithme du tableau et à partir d'un concept $\mathcal{ALC}$ insatisfiable par rapport à une ontologie, un ensemble d'arbre de complétion $T = \langle v_0, V, E, L\rangle$ contenant un conflit, *i.e.* $\{A, \neg A\}$ est présent dans le label d'au moins un nœud de T. Nous renvoyons la lectrice ou le lecteur vers [3] pour des détails plus formels sur les algorithmes de tableau.
Lorsqu'une règle **[$\sqcup$-rule]** est appliquée dû à la présence d'une disjonction $X_1 \sqcup X_2$ dans un label $L(v)$, alors nous générons deux nouveaux graphes T_1 et T_2. Ces deux graphes sont des copies de T à l'exception du nœud v qui est remplacé à par des nœuds v_i dans T_i tels que $L(v_i) = L(v) \cup X_i$ pour $i \in \{1, 2\}$. Les v_i sont appelés des nœuds disjoints et v est un nœud de disjonction. Nous définissons alors l'arbre $\mathbb{T}$ dont les nœuds sont les arbres de complétions T. Celui-ci est construit de sorte que si un nœud v possède un conflit, alors il appartient à un arbre feuille et si une règle **[$\sqcup$-rule]** doit être appliqué à un nœud de disjonction v, alors la règle a déjà été appliquée à tous les ancêtres v' de v.
À partir de là, nous construisons une catégorie NNF saturée minimale $\mathscr{C}_c\langle C_0, \mathcal{O}\rangle$ à partir de $\mathbb{T}$ dont tous les arbres contiennent un conflit. La construction est faite en sorte de pouvoir propager le conflit de chacun des arbres feuilles jusqu'à la racine v_0 de l'arbre racine T_0 de $\mathbb{T}$, dont le label est $L(v_0) = \{C_0\}$. Cette propagation va rencontrer deux problèmes principaux : d'un nœud à son ancêtre qui est un nœud disjoint et de ce nœud disjoint au nœud disjonction qui se trouve dans l'arbre parent dans $\mathbb{T}$. Cependant, les règles de constructions utilisées pour créer $\mathscr{C}_c\langle C_0, \mathcal{O}\rangle$ permettent de franchir ces obstacles jusqu'à atteindre la racine v_0. Grâce à la propagation, nous en déduisons donc que $C_0 \longrightarrow \bot \in \mathsf{Hom}(\mathscr{C}_c\langle C_0, \mathcal{O}\rangle)$ et donc que C_0 est bien insatisfiable catégoriellement par rapport à $\mathcal{O}$. □

Ce résultat nous fournit un algorithme optimal dans le pire des cas (*i.e* ExpTime) pour pouvoir déterminer la satifiabilité d'un concept en $\mathcal{ALC}$. En effet, en utilisant directement les propriétés des définitions 1, 2 et 3 comme règles, nous remplaçons le verbe « impliquer » par « ajouter à » tout en vérifiant que l'objet n'a pas été préalablement ajouté, et ainsi nous obtenons un procédé pour construire une catégorie saturée minimale NNF. Il suffit alors de stocker les conjonctions et les disjonctions comme des ensembles de conjoints et disjoints, cela permet d'éviter les doublons ; par exemple, si deux objets partagent le même disjoints ou conjoints. Alors une implémentation de ce type pourrait borner (exponentiellement) le nombre de conjoints et disjoints ajoutés par (3.3) et (3.8).
La question de l'*entailment* peut alors se poser, *i.e.* de savoir si une subsomption est satisfaite pour tout modèle de $\mathcal{O}$. Dans ce cas, nous pourrions répondre à cette question en construisant une catégorie saturée au sens de la définition 3 et vérifier si $C \longrightarrow D$ est une flèche de cette catégorie. Une autre façon est d'utiliser le travail que nous venons de faire. L'équivalence 9 assure que l'existence de la flèche $C \longrightarrow D$ dans la catégorie est équivalente à l'existence de la flèche $C \sqcap \neg D \longrightarrow \bot$ – et donc à l'insatisfiabilité du concept $C \sqcap \neg D$ par définition. Ainsi, pour que $\mathcal{O} \models C \sqsubseteq D$, il faut et il suffit que $\mathscr{C}_c\langle C \sqcap \neg D, \mathcal{O}\rangle$ contienne la flèche $C \sqcap \neg D \longrightarrow \bot$.

4 $\mathcal{ALC}_{\bar{\forall}}$ et raisonnement

Dans cette section, nous allons exploiter la modularité apportée par les propriétés de la théories des catégories. Plus précisément, nous allons utiliser l'indépendance de la propriété F13 par rapport aux autres propriétés, comme illustré dans l'exemple 2. Cette propriété permet de déduire l'interaction entre les constructeurs $\exists$ et $\forall$, interaction qui est en partie responsable de la complexité exponentielle de $\mathcal{ALC}$ [2]. C'est après cette observation que nous avons décidé de développer une nouvelle logique dérivée $\mathcal{ALC}_{\bar{\forall}}$ de $\mathcal{ALC}$ en « affaiblissant » la restriction universelle.
Concrètement, nous enlèverons la propriété 3.8 de la définition 3 et introduirons de nouvelles propriétés à celles déjà existantes. Avant de définir proprement $\mathcal{ALC}_{\bar{\forall}}$, il a été prouvé, voir le lemme 1, que F13, et donc la propriété 3.8, implique également

$$C \sqcap D \to \bot \implies \exists R.C \sqcap \forall R.D \longrightarrow \bot \quad (*)$$

Par conséquent, retirer F13 nous prive alors de cette propriété nécessaire pour découvrir certain *conflit* ou clash en anglais. Cependant, comme cela a été montré dans le lemme 1, combinée à F10, la propriété F11 est suffisante (et nécessaire) pour obtenir l'implication $(*)$. Ainsi, même si l'interaction est incomplète, nous en gardons une conséquence tout en obtenant une complexité inférieure comme nous allons le montrer.
Puisque nous voulons conserver F11, *i.e.* la relation $\forall R.C \longleftrightarrow \neg\exists R.\neg C$, nous pouvons toujours supposer que les objets écrits dans la catégorie sont en NNF. Par la suite, nous écrirons alors $\bar{\forall}$ en lieu et place de $\forall$ pour désigner la restriction universelle *affaiblie* dans la logique $\mathcal{ALC}_{\bar{\forall}}$.

Définition 6 (Insatisfiabilité catégorielle dans $\mathcal{ALC}_{\bar{\forall}}$)
Soit C_0 un concept en $\mathcal{ALC}_{\bar{\forall}}$ et $\mathcal{O}$ une TBox dans cette même logique. Une catégorie d'ontologie $\mathscr{C}_c$ pour $\langle C_0, \mathcal{O}\rangle$ est dite NNF-saturée *en $\mathcal{ALC}_{\bar{\forall}}$ si elle satisfait toutes les propriétés de (3.1) à (3.7), ainsi que les deux propriétés suivantes :*

6.1 $\forall X \in \mathsf{Ob}(\mathscr{C}_c), X \longrightarrow \exists R.C, X \longrightarrow \bar{\forall} R.C' \in \mathsf{Hom}(\mathscr{C}_c)$ *implique* $C \sqcap C' \in \mathsf{Ob}(\mathscr{C}_c)$

6.2 $\forall X \in \mathsf{Ob}(\mathscr{C}_c), C \sqcap C' \longrightarrow \bot, X \longrightarrow \exists R.C,$ $X \longrightarrow \bar{\forall} R.C' \in \mathsf{Hom}(\mathscr{C}_c)$ *implique* $X \longrightarrow \bot \in \mathsf{Hom}(\mathscr{C}_c)$

Dans ce cas, C_0 est dit catégoriellement insatisfiable par rapport à $\mathcal{O}$ dans $\mathcal{ALC}_{\bar{\forall}}$ *si la catégorie NNF-saturée minimale en $\mathcal{ALC}_{\bar{\forall}}$, $\mathscr{C}_c\langle C_0, \mathcal{O}\rangle$, possède la flèche $C_0 \longrightarrow \bot$,* i.e. $C_0 \longrightarrow \bot \in \mathsf{Hom}(\mathscr{C}_c\langle C_0, \mathcal{O}\rangle)$.

Pour illustrer la différence du pouvoir d'expressivité entre $\mathcal{ALC}$ et $\mathcal{ALC}_{\bar{\forall}}$, utilisons l'exemple de la TBox $\mathcal{O}_1$ contenant uniquement l'axiome :

$$Y \sqsubseteq \exists R.C \sqcap \bar{\forall} R.D \sqcap \bar{\forall} R.\neg D$$

Dans le contexte de la logique $\mathcal{ALC}$ complète, le regroupement $C \sqcap D \sqcap \neg D$ est suffisant pour rendre Y insatisfiable par rapport à $\mathcal{O}_1$ dans $\mathcal{ALC}$. Cependant, pour la logique $\mathcal{ALC}_{\bar{\forall}}$ ce regroupement n'est pas autorisé. Les seules conjonctions qui pourraient influer sur la satisfiabilité du concept $\exists R.C \sqcap \bar{\forall} R.D \sqcap \bar{\forall} R.\neg D$ sont les objets $C \sqcap D$ et $C \sqcap \neg D$ qui n'ont aucune raison d'avoir une flèche les reliant à $\bot$ dans la catégorie NNF-saturée $\mathscr{C}_c\langle Y, \mathcal{O}_1\rangle$ – le concept est alors bien satisifiable dans $\mathcal{ALC}_{\bar{\forall}}$ par rapport à $\mathcal{O}_1$. Il est à noter que si un concept est insatisfiable dans $\mathcal{ALC}_{\bar{\forall}}$, il l'est *nécessairement* dans la logique $\mathcal{ALC}$ en remplaçant $\bar{\forall}$ par $\forall$, inversement, si un concept est satisfiable dans $\mathcal{ALC}$ l'est également pour $\mathcal{ALC}_{\bar{\forall}}$ en modifiant les bons constructeurs. Pour les deux autres cas, nous ne pouvons pas conclure.

Pour le reste de la section, il nous reste alors à introduire les différentes notions et structures nécessaires à la description et définition de l'alogrithme qui nous permettra de vérifier la satisfiabilité d'un concept C_0 en $\mathcal{ALC}_{\bar{\forall}}$ par rapport à une TBox dans cette même logique. Nous pourrions tout simplement construire une catégorie en utilisant directement les propriétés de la définition 6, mais cela reviendrait à ajouter un nombre exponentielle d'objet, notamment à cause de la propriété 3.3.
Pour obtenir un algorithme NP de vérification de la satisfiabilité dans $\mathcal{ALC}_{\bar{\forall}}$, nous allons avoir besoin d'une structure intermédiaire que nous appelons *graphe d'objets*. Les nœuds de ce graphe sont labelisés par un ensemble d'objets représentant des *sous-concepts*. Nous désignons alors par $\mathsf{sub}\langle C_0, \mathcal{O}\rangle$ le plus petit ensemble des sous-concepts qui apparaissent dans C_0 et $\mathcal{O}$. De sorte que $\mathsf{sub}\langle C_0, \mathcal{O}\rangle$ contient C_0 et $\mathsf{NNF}(\neg E \sqcup F)$ pour tout $E \sqsubseteq F \in \mathcal{O}$. Il contient également les conjoints des conjonctions, les disjoints des disjonctions et les *fillers* des restrictions existentielles et universelles.

Définition 7 (Graphes d'objets) *Soit C_0 un concept en $\mathcal{ALC}_{\bar{\forall}}$ et $\mathcal{O}$ une TBox. Nous appelons graphe d'objet pour $\langle C_0, \mathcal{O}\rangle$, un graphe $G := \langle V, E, L, v_0\rangle$ tel que V est l'ensemble des nœuds et $v_0 \in V$, E est l'ensemble des arêtes, L est une fonction qui associe à chaque nœud v sont label $L(v) \subseteq \mathsf{sub}\langle C_0, \mathcal{O}\rangle$ et v_0 est un nœud tel que $C_0 \in L(v_0)$. Un nœud $v \in V$ possède un* conflit *si $\bot \in L(v)$ ou s'il existe un concept atomique A tel que $A, \neg A \in L(v)$ – dans ce cas, nous disons que G* contient un conflit*, sinon G est dit* sans conflit*.*

Pour déterminer la satisfiabilité d'un concept C_0 en $\mathcal{ALC}_{\bar{\forall}}$ par rapport à une TBox $\mathcal{O}$, l'algorithme 1 construit un graphe d'objet $G = \langle V, E, L, v_0\rangle$ avec l'entrée $\langle C_0, \mathcal{O}\rangle$. Il commence par créer le nœud racine v_0 avec $C_0, \top \in L(v_0)$ et ajoute $\mathsf{NNF}(\neg F \sqcup F')$ à $L(v_0)$ pour chaque $F \sqsubseteq F' \in \mathcal{O}$. Par la suite, tant qu'un changement est effectué sur le graphe, différentes opérations sont opérées sur les labels de chaque nœud $v \in V$. Si $C \sqcap D \in L(v)$ et $\{C, D\} \nsubseteq L(s)$ alors C, D sont ajoutés à $L(v)$; ceci vient de la propriété (3.2). Si $C \sqcup D \in L(v)$ et $L(s) \cap \{C, D\} = \varnothing$ alors l'algorithme choisit d'ajouter soit C, soit D au label de v ; cette règle non-déterministe couvre la propriété (3.3). Un nouveau nœud est créé dans seulement deux cas : le premier cas se présente s'il y a un nœud v tel que $\exists R.C \in L(v)$ et qu'il n'y a pas un autre nœud v' tel que $C \in L(v')$; cette opération est conséquence de (3.6). Le second, s'il existe un nœud v tel que $\exists R.C, \bar{\forall} R.C' \in L(v)$ et qu'il n'y a pas de nœud $v' \in V$ tel que $\{C, C'\} \subseteq L(v)$; ceci est la conséquence de (6.1). Il est à noter qu'à chaque création d'un nouveau nœud v, l'algorithme ajoute $\top$ ainsi que $\mathsf{NNF}(\neg F \sqcup F')$ à $L(v)$ pour tout $F \sqsubseteq F' \in \mathcal{O}$.
La fonction $\mathsf{search}(v, C, C')$ sert à déterminer si le concepts C ou les concepts C, C' sont présents dans le graphe. Elle commence par chercher parmi les voisins v' de v et retourne faux si $\{C, C'\} \subseteq L(v')$. Sinon elle parcourt tous les autres nœuds de G et retourne vrai si elle ne trouve aucun nœud qui contient les concepts. En conséquence, pour chaque pair $\exists R.C, \bar{\forall} R.C'$, il y a au plus deux nœuds qui sont crées par l'algorithme 1. L'algorithme se termine lorsqu'il n'y a plus aucun changement à faire sur G, *i.e.* tous les éléments de tous les labels n'entraînent plus de modifications de labels, ni de créations de nouveau nœuds.
Regardons le fonctionnement de l'algorithme 1 dans l'exemple suivant.

Exemple 3 *Soit $\mathcal{T}$ une TBox en $\mathcal{ALC}_{\bar{\forall}}$ définie comme suit :*

$$\mathcal{T} = \{A \sqsubseteq \exists R.C \sqcap \bar{\forall} R.D, D \sqsubseteq \exists R.E, D \sqsubseteq \bar{\forall} R.F, D \sqsubseteq \bar{\forall} R.H, E \sqcap (F \sqcap H) \sqsubseteq \bot\}$$

Pour que A soit insatisfiable dans $\mathcal{ALC}_{\bar{\forall}}$ par rapport à $\mathcal{T}$, $\exists R.C \sqcap \bar{\forall} R.D$ et donc D doivent être également insatisfiables. Cela est possible uniquement si E, $E \sqcap F$ ou $E \sqcap H$ sont insatisfiables par rapport $\mathcal{T}$; ce qui n'est pas le cas. Comme A est satisfiable dans $\mathcal{ALC}_{\bar{\forall}}$ par rapport à $\mathcal{T}$, alors il existe des choix (ligne 10) tels que l'algorithme 1 produise un graphe sans conflit avec l'entrée $\langle A, \mathcal{T}\rangle$. Il est alors possible que l'algorithme produise un graphe $G := \langle V, E, L, v_0\rangle$ où :

Input : $\langle C_0, \mathcal{O}\rangle$ where C_0 is an $\mathcal{ALC}_{\bar{\forall}}$ concept, $\mathcal{O}$ an ontology
Output : A graph of objects $G = \langle V, E, L, v_0\rangle$

1 Initialize $V \leftarrow E \leftarrow L \leftarrow \varnothing$;
2 Create a node v_0 and set $V \leftarrow V \cup \{v_0\}$, $L(v_0) \leftarrow L(v_0) \cup \{C_0\}$;
3 $L(v_0) \leftarrow L(v_0) \cup \{\mathsf{NNF}(\neg F \sqcup F'), \top\}$ for each $F \sqsubseteq F' \in \mathcal{O}$;
4 **while** *G is changed* **do**
5 **foreach** $v \in V$ *and* $X \in L(v)$ **do**
6 **if** $X = C \sqcap D$ *and* $\{C, D\} \nsubseteq L(v)$ **then**
7 $L(v) \leftarrow L(v) \cup \{C, D\}$;
8 **end**
9 **if** $X = C \sqcup D$ *and* $\{C, D\} \cap L(v) = \varnothing$ **then**
10 Choose some $X \in \{C, D\}$ and $L(v) \leftarrow L(v) \cup \{X\}$;
11 **end**
12 **if** $X = \exists R.C$ *and there is some* $\bar{\forall} R.C' \in L(v)$ *and* $\mathsf{search}(v, C, C')$ **then**
13 Create a new node v';
14 $V \leftarrow V \cup \{v'\}$, $L(v') \leftarrow L(v') \cup \{C, C', \top\}$, $E \leftarrow E \cup \{\langle v, v'\rangle\}$;
15 $L(v') \leftarrow L(v') \cup \{\mathsf{NNF}(\neg F \sqcup F')\}$ for each $F \sqsubseteq F' \in \mathcal{O}$;
16 **end**
17 **if** $X = \exists R.C$ *and there is no object* $\bar{\forall} R.C' \in L(v)$ *and* $\mathsf{search}(v, C, \top)$ **then**
18 Create a new node v';
19 $V \leftarrow V \cup \{v'\}$, $L(v') \leftarrow L(v') \cup \{C, \top\}$, $E \leftarrow E \cup \{\langle v, v'\rangle\}$;
20 $L(v') \leftarrow L(v') \cup \{\mathsf{NNF}(\neg F \sqcup F')\}$ for each $F \sqsubseteq F' \in \mathcal{O}$;
21 **end**
22 **if** $X = \forall R.C$ *and there is some* $\exists R.C' \in L(v)$ *and* $\mathsf{search}(v, C, C')$ **then**
23 Create a new node v';
24 $V \leftarrow V \cup \{v'\}$, $L(v') \leftarrow L(v') \cup \{C, C', \top\}$, $E \leftarrow E \cup \{\langle v, v'\rangle\}$;
25 $L(v') \leftarrow L(v') \cup \{\mathsf{NNF}(\neg F \sqcup F')\}$ for each $F \sqsubseteq F' \in \mathcal{O}$;
26 **end**
27 **end**
28 **end**
29 **return** G;

Algorithme 1 : Construction d'un graphe G pour $\langle C_0, \mathcal{O}\rangle$.

$$\begin{aligned}
\{A, \exists R.C, \bar{\forall} R.D, \neg D, \neg E\} &\subseteq L(v_0)\\
\{C, D, \neg A, \exists R.E, \bar{\forall} R.F, \bar{\forall} R.H, \neg E\} &\subseteq L(v_1)\\
\{C, \neg A, \neg D, \neg E\} &\subseteq L(v_2)\\
\{E, \neg A, \neg D, \neg F\} &\subseteq L(v_3)\\
\{E, F, \neg A, \neg D, \neg H\} &\subseteq L(v_4)\\
\{E, H, \neg A, \neg D, \neg F\} &\subseteq L(v_5)
\end{aligned}$$

qui n'a effectivement aucun conflit et donc vient appuyer le

fait que A soit satisfiable par rapport à $\mathcal{T}$ dans $\mathcal{ALC}_{\forall}$.

Une procédure de décision sur la satisfiabilité revient alors à soit trouver un graphe G sans conflit, auquel cas le concept C_0 est bien satisfiable par rapport $\mathcal{O}$, soit à montrer que tous les graphes G générés grâce à l'algorithme contiennent un conflit. Il nous reste à prouver que effectivement un concept C_0 est satisfiable par rapport à une TBox $\mathcal{O}$, dans le sens de la sémantique ensembliste, si et seulement si la catégorie saturée NNF, $\mathscr{C}_c\langle C_0, \mathcal{O}\rangle$ ne contient pas de flèche $C_0 \longrightarrow \bot$.

Lemme 3 (Correction et complétude) *Soit C_0 un concept en $\mathcal{ALC}_{\forall}$ et $\mathcal{O}$ une TBox. Si l'algorithme 1 prend $\langle C_0, \mathcal{O}\rangle$ en entrée, alors il retourne un graphe d'objets sans conflit si et seulement si C_0 est satisfiable par rapport à $\mathcal{O}$.*

Esquisse de preuve "$\Longrightarrow$". Nous commençons par définir une catégorie $\mathscr{C}$ à partir de tous les $G = \langle V_G, E_G, L_G, v^0_G\rangle$ générés par l'algorithme 1. En nous basant sur des propriétés particulières de $L_G(v)$ pour $v \in V_G$, nous définissons L'_G tel que (i) $L'_G(v)$ contient un objet minimal, que nous nommons $\mathsf{core}(v)$ de tel sorte qu'il y ait une flèche $\mathsf{core}(v) \longrightarrow C$ pour tout $C \in L'_G(v)$ et (ii) $L'_G(v)$ est « fermé » par les flèches $X \longrightarrow Y \in \mathsf{Hom}(\mathscr{C})$ dans le sens où si $X \longrightarrow Y$ est une flèche de $\mathscr{C}$ alors $X \in L'_G(v)$ implique $Y \in L'_G(v)$.
Pour construire cette catégorie, nous initialisons $\mathscr{C}$, une fonction $\mathsf{core}(v)$ qui associe à chaque v dans $\bigcup_G V_G$ une conjonction d'objets dans $L_G(v)$ et une fonction de label $L'_G(v)$ définie depuis $L_G(v)$ avec $L_G(v) \subseteq L'_G(v)$. Tous ces objets sont initialisés de sorte que chacune les propriétés (S1) à (S5), voir [9], restent invariantes à chaque changement effectué sur $\mathscr{C}$ par l'application une propriété de $\mathcal{ALC}_{\forall}$.

(S1) Si $X \in L'_G(v) \setminus L_G(v)$, alors soit $X = \bot$ ou X est une conjonction d'objets de $L_G(v)$;

(S2) Si $X \longrightarrow Y \in \mathsf{Hom}(\mathscr{C})$ et $X \in L'_G(v)$ pour $v \in V_G$, alors $Y \in L'_G(v)$;

(S3) Si $X \longrightarrow Y \in \mathsf{Hom}(\mathscr{C})$ alors soit $Y = \mathsf{NNF}(\neg E \sqcup F)$ que nous appelons alors *objets axiomes*, avec $E \sqsubseteq F \in \mathcal{O}$ ou Y un conjoint de X ou bien $Y = \bot$.

(S4) Pour chaque $v \in \bigcup_G V_G$, il y a une flèche $\mathsf{core}(v) \longrightarrow U \in \mathsf{Hom}(\mathscr{C})$ pour tout $U \in L'_G(v)$.

(S5) Si $A, \neg A \in L'_G(v)$ pour $v \in V_G$, où A est un concept atomique, alors $\bot \in L'_G(v)$.

Nous montrons alors qu'en appliquant toutes les propriétés de $\mathcal{ALC}_{\forall}$ à $\mathscr{C}$, nous obtenons (i) que tous les objets venant de tous les graphes G générés par l'algorithme 1 peuvent être ajoutés à $\mathscr{C}$ et (ii) que (S1)-(S5) soient invariants. Nous prouvons également que $\mathscr{C}$ est NNF-saturée et minimale.
Par hypothèse, il existe un graphe G_0 sans conflit généré par l'algorithme. Supposons alors que $C_0 \longrightarrow \bot$ est une flèche de $\mathscr{C}$, autrement dit que C_0 est insatisfiable par rapport à $\mathcal{O}$. Alors par (S2), nous avons $\bot \in L'(v^0_{G_0})$ pour un nœud v^0. Par hypothèse, $\bot \notin L(v^0_{G_0})$ et donc par (S1), $\bot$ doit être la conjonction de deux objet, nécessairement $A \sqcap \neg A$ et par conséquent $A, \neg A \in L(v^0_{G_0})$ ce qui contredit le fait que G_0 soit sans conflit.
"$\Longleftarrow$". Supposons que $\mathscr{C}$ est une catégorie minimale saturée NNF telle que $C_0 \longrightarrow \bot$ n'est *pas* une flèche de $\mathscr{C}$. Nous allons montrer que l'algorithme 1 peut générer un graphe sans conflit.
Pour cela, nous initialisons un graphe d'objet $G = \langle V, E, L, v_0\rangle$ et une fonction $\mathsf{core}(v)$ qui associe à chaque nœud $v \in V$ une conjonction d'objet dans $L(v)$. Nous avons également besoin d'une fonction L' définie à partir de L telle que $L(v) \subseteq L'(v)$ pour $v \in V$. Pendant l'exécution de l'algorithme 1, G, L' et core vont être mis à jour et étendus de sorte que les propriétés (C1) à (C4), voir [9], soient vérifiées.

(C1) Si $X \in L'(v) \setminus L(v)$ alors X est une conjonction d'objet de $L(v)$;

(C2) Pour tout $v \in V$, si $C \in L'(v)$ alors $C \in \mathsf{Ob}(\mathscr{C})$.

(C3) Pour tout $v \in V$, nous avons $\mathsf{core}(v) = \prod_{X\in\Gamma} X$, avec $\Gamma \subseteq L(v)$ et $\mathsf{core}(v) \longrightarrow X \in \mathsf{Hom}(\mathscr{C})$ pour tout $X \in L'(v)$

(C4) $\mathsf{core}(v) \longrightarrow \bot \notin \mathsf{Hom}(\mathscr{C})$ pour tout $v \in V$.

Par l'absurde, nous supposons que G possède un conflit, *i.e.* $\bot \in L(v)$ ou $A, \neg A \in L(v)$ pour un $v \in V$. Comme $L(v) \subseteq L'(v)$ alors $\bot \in L'(v)$ ou $A, \neg A \in L'(v)$. Dans les deux cas, d'après (C3) nous avons $\mathsf{core}(v) \longrightarrow \bot \in \mathsf{Hom}(\mathscr{C})$ ce qui est en contradiction avec (C4). □

Lemme 4 (Complexité) *Soit C_0 un concept en $\mathcal{ALC}_{\forall}$ et $\mathcal{O}$ une TBox. L'algorithme 1 s'exécute en un temps polynomial non-déterministe par rapport à la taille de $\langle C_0, \mathcal{O}\rangle$.*

Esquisse de preuve La complexité de l'algorithme 1 dépend principalement de la taille de $\mathsf{sub}\langle C_0, \mathcal{O}\rangle$, noté $|\mathsf{sub}\langle C_0, \mathcal{O}\rangle|$ ainsi que de $|V|$ pour le graphe $G := \langle V, E, L, v_0\rangle$ construit par l'algorithme. Comme nous pouvons voir les sous-concepts comme de sous-chaînes de caractères de C_0 et $\mathcal{O}$ vus eux-même comme des chaînes de caractères, alors nous obtenons que $|\mathsf{sub}\langle C_0, \mathcal{O}\rangle| \leqslant n(n+1)/2 = O(n^2)$ où $n = \|\langle C_0, \mathcal{O}\rangle\|$ est la taille en octets pour pouvoir stocker C_0 et $\mathcal{O}$. Notons $n_\exists$ et $n_\forall$ le nombre d'éléments de $\mathsf{sub}\langle C_0, \mathcal{O}\rangle$ de la forme $\exists R.C$ et $\forall R.C$ respectivement. Nous pouvons alors montrer que pour le nombre de nœuds $n_v = |V|$, nous avons bien $n_v \leqslant n_\exists(n_\forall + 1)$. De plus, nous avons également que $|L(v)|$, $n_\exists$ et $n_\forall$ sont tous plus petit ou égaux à $|\mathsf{sub}\langle C_0, \mathcal{O}\rangle|$.
L'algorithme s'arrête lorsqu'il n'y a plus de changements effectués, *i.e.* la boucle (ligne 4) s'arrête après avoir effectué $n_{\mathsf{iter}} \leqslant n_\exists(n_\forall + 1) \times |\mathsf{sub}\langle C_0, \mathcal{O}\rangle| \leqslant (|\mathsf{sub}\langle C_0, \mathcal{O}\rangle|)^3$ itérations de la boucle ligne 5. Pour chaque itérations, il y a au plus $m+4$ opérations avec $m := |\mathcal{O}|$. Il y a donc au plus $(|\mathsf{sub}\langle C_0, \mathcal{O}\rangle|)^3 \times (m+4) \leqslant O(n^6)$ opérations faites par l'algorithme. Ainsi, l'algorithme 1 s'exécute en un temps polynomial non-déterministe en la taille de l'entrée. □

Lemme 5 *Le problème de satisfiabilité d'un concept en $\mathcal{ALC}_{\forall}$ est* NP*-difficile.*

Ce lemme est conséquence du fait que $\mathcal{ALC}_{\bar{\forall}}$ inclue la logique propositionnelle qui est connue pour être NP-complète [7]. Le théorème suivant est lui conséquence des lemmes 4 et 5.

Théorème 2 *Le problème de satisfiabilité dans* $\mathcal{ALC}_{\bar{\forall}}$ *est* NP*-complet.*

5 Conclusion

Nous venons de présenter une réécriture la sémantique usuelle ensembliste de $\mathcal{ALC}$ en terme de théorie des catégories et un nouveau langage défini à partir de $\mathcal{ALC}$ utilisant cette réécriture.
Grâce à l'absence de l'appartenance ensembliste dans la théorie des catégories, nous avons besoin de deux propriétés indépendantes pour illustrer la restriction universelle et son interaction la restriction existentielle. En affaiblissant cette dernière, nous avons pu obtenir un nouvelle langage $\mathcal{ALC}_{\bar{\forall}}$ qui est NP-complet en terme de satisfiabilité, alors que $\mathcal{ALC}$ est EXPTIME-complet. Il est à noté que ce n'est pas cette interaction affaiblie qui est responsable de la NP-complètude de $\mathcal{ALC}_{\bar{\forall}}$, mais bien celle de la conjonction et disjonction.
Cette observation nous donne l'espoir de pouvoir obtenir de nouveau langage tractable pouvant être étendue grâce à cette version affaiblie de la restriction universelle.
Nous sommes actuellement en train de terminer l'implémentation de l'algorithme pour le comparer avec d'autres raisonneurs sur des ontologies choisies. Pour nos prochains travaux, nous explorons aussi une nouvelle logique possédant non seulement une version affaiblie de l'interaction entre la restriction universelle et existentielle, mais également une distributivité affaiblie entre la conjonction et la disjonction. Cette nouvelle logique serait alors tractable d'après nos résultats. Nous pourrions alors ajouter les deux nouveaux constructeurs affaiblies à $\mathcal{EL}$++ qui est, comme nous l'avons mentionné, la logique de description sous-jacente pour de nombreuses ontologies médicales.

Références

[1] Franz Baader, Diego Calvanese, Deborah L. McGuinness, Daniele Nardi, and Peter F. Patel-Schneider, editors. *The Description Logic Handbook : Theory, Implementation and Applications, Second Edition*. Cambridge University Press, 2010.

[2] Franz Baader, Ian Horrocks, Carsten Lutz, and Uli Sattler. *An Introduction to Description Logic*. Cambridge University Press, 2017.

[3] Franz Baader and Ulrike Sattler. Tableau algorithms for description logics. In *Proceedings of the International Conference on Automated Reasoning with Tableaux and Related Methods*, volume 1847, page 118, St Andrews, Scotland, UK, 2000. Springer-Verlag.

[4] Ludovic Brieulle, Chan Le Duc, and Pascal Vaillant. Reasoning in the description logic ALC under category semantics (extended abstract). In *Proceedings of the 35th International Workshop on Description Logics (DL 2022)*, volume 3263 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2022.

[5] Werner Ceusters, Peter Elkin, and Barry Smith. Negative findings in electronic health records and biomedical ontologies : A realist approach. *International Journal of Medical Informatics*, 76 :S326–S333, 2007.

[6] Mihai Codescu, Till Mossakowski, and Oliver Kutz. A categorical approach to networks of aligned ontologies. *J. Data Semant.*, 6(4) :155–197, 2017.

[7] Cook. The Complexity of Theorem-Proving Procedures. *STOC '71, Proceedings of the third annual ACM symposium on Theory of computing*, pages 151–158, 1971.

[8] B. Cuenca Grau, I. Horrocks, B. Motik, B. Parsia, P. Patel-Schneider, and U. Sattler. Owl 2 : The next step for owl, journal of web semantics : Science, services and agents. *World Wide Web*, 6 :309–322, 2008.

[9] Ludovic Brieulle et Chan Le Duc. Rapport technique : Sémantique catégorielle de logique de description et raisonnement. http://limics2.univ-paris13.fr/~brieulle/JIAF2025TR.pdf, 2025.

[10] Clemens Kupke and Dirk Pattinson. Coalgebraic semantics of modal logics : An overview. *Theoretical Computer Science*, 412(38) :5070–5094, 2011.

[11] Chan Le Duc. Category-theoretical semantics of the description logic alc. In Martin Homola, Vladislav Ryzhikov, and Renate A. Schmidt, editors, *Proceedings of the 34th International Workshop on Description Logics (DL 2021)*, volume 2954 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2021.

[12] Sounders Mac Lane and Ieke Moerdijk. *Sheaves in Geometry and Logic*. Springer, 1992.

[13] Lawrence S. Moss. Coalgebraic logic. *Annals of Pure and Applied Logic*, 96(1) :277–317, 1999.

[14] Peter Patel-Schneider, P. Hayes, and Ian Horrocks. Owl web ontology language semantics and abstract syntax. In *W3C Recommendation*, 2004.

[15] Klaus Schild. A Correspondence Theory for Terminological Togics : Preliminary Report. In *Proceedings of the 12th international joint conference on Artificial intelligence*, pages 466–471, 1991.

[16] David I Spivak and Robert E Kent. Ologs : a categorical framework for knowledge representation. *PloS one*, 7(1) :e24274, 2012.

What killed the cat again? Towards a logical formalization of curiosity (and suspense, and surprise) in narratives with an experimental use of LLM for capturing causality

Benjamin Callac[1], Anne-Gwenn Bosser[1], Florence Dupin de Saint-Cyr[2], Eric Maisel[1]

[1] Lab-STICC_COMMEDIA, CNRS UMR6285, Brest, France
[2] IRIT, Université de Toulouse, France

Résumé

Cet article étend légèrement celui présenté à la conférence TIME'2024[17] dans lequel nous formalisons les affects de la tension narrative – curiosité, suspense et surprise – en logique propositionnelle. Le raisonnement non-monotone est utilisé pour représenter de façon compacte le comportement par défaut du monde et raisonner sur l'état affectif d'un agent. Nous ajoutons ici une expérimentation de calcul automatique de la causalité avec un LLM.

Mots-clés

Représentation de connaissances, Narration, Cognition.

Abstract

This paper slightly extends the one presented at TIME'2024[17] in which we formalize the affects of narrative tension – curiosity, suspense and surprise – in propositional logic. Non-monotonic reasoning is used to compactly represent the default behavior of the world and to reason about the affective state of an agent. We add here an experiment in automatic causality computation with an LLM.

Keywords: *Knowledge Representation, Narration, Cognition*

1 Introduction

Humans tell stories to make sense of the world and communicate their understanding of what happens. Storytelling supposes to be able to sort out which events are worth telling, deciding on a level of detail for describing events, selecting among possible causes the ones which are deemed worth telling. It also supposes to make use of an affective machinery for capturing an audience's attention (emotional contagion, suspense elicitation...). In the act of storytelling, structural and affective phenomena are thus combined with communicative goals in mind. This combination has indeed shown its effectiveness in this respect: the phenomenon of narrative transportation (the experience of being immersed in a story) has been linked to persuasion [27]. The narrative paradigm therefore provides an appropriate framework, in which causal reasoning about the situations narrated [54] is combined with narrative devices to encourage the audience's emotional involvement [51], to study and model how opinion is formed and evolves. Building a framework for reasoning about and unveiling storytelling mechanics could pave the way for intellectual self-defense supporting tools, enabling citizens to arm themselves against hostile disinformation or influence campaigns.

Previous works in structural narratology have studied the way stories are conveyed to their audience and seminal work from (for instance) Genette [25] or Propp [46] have previously served as the backbone inspiration for computational narrative models and storytelling systems [44]. Whilst the operationalization of narrative theories is still subject to debate and caution, such works have shed light on how the story material to tell and the manner in which it is told interacts with a model of the listener[1] (which, depending on the media used for conveying the story can also be a reader, a spectator, or even a gamer): the act of storytelling can thus be understood as knowledge transfer and manipulation of her beliefs.

According to Sternberg [51] or Baroni [5], emotions more specific to narratives which are *suspense*, *curiosity* and *surprise* are critical to retain the interest of the listener. Drivers of the *narrative tension*, they are paramount in maximizing her engagement. In this paper we focus on these narrative tension's building blocks.

In the field of computational narratives, numerous studies and frameworks exist to tell interactive stories, a number of them as an application of planning technologies [12] allowing to adapt the narrative to each user's actions. However, adapting a narrative to a model of the user's emotions remains largely a challenge that needs to be addressed to favor engagement: narrative engagement depends partly on the appropriate maintenance of narrative tension, itself based on the uncertainty occurring in a narrative [8], and listener's models based on a formalization of related emotions have comparatively been less addressed so far in the literature. While suspense and surprise have been the object of previous studies [14] [23], there is — to our knowledge — still no curiosity model applicable to narratives.

In the following, we present a preliminary study

[1] For sake of homogeneity, we use the term listener in all the paper, while this kind of agent is called interpret by Baroni.

for characterizing these emotions from an epistemic standpoint, with a focus on modeling the listener's curiosity depending on her beliefs and knowledge using a propositional language. Our overarching aim is to provide a unifying framework allowing to represent emotions relevant to the characterization of narrative tension and its evolution, which would enable to discuss their relationships and ultimately help establish dramatic metrics about a narrative.

In Section 2, we describe the main emotions supporting narrative tension. We also describe the problems and solutions for formalizing reasoning about action and change, as well as the ways in which the notions of surprise and awareness have been treated in the literature. Section 3 details our proposal, which relies on a non-monotonic framework resulting from extending propositional logic with default rules. The properties of the framework are presented in Section 4, along with some preliminary ideas for developing metrics.

2 Background on reasoning about change and narrative tension

In order to reason about a story, it is useful to dispose of a way to handle the concepts involved for understanding a sequence of facts and events. We first briefly outline the background to this vast subject, before discussing the formalization of emotions in the literature.

2.1 Reasoning about action and change

The formalization of action and change is an old field of research in the domain of knowledge representation and reasoning in AI. There are many different reasoning tasks in this field (see e.g. [18]) like prediction of the new state of the world after an action (which is related to *belief update* [56, 31]), or integration of an observation (which is related to *belief revision* [2]), or *event abduction* which consists in guessing which event took place, or *scenario extrapolation* [19, 16] which consists in taking a partial description of facts and events that occurred and complete it (by prediction or event abduction) or *scenario recognition* [15].

These reasoning tasks were studied in various frameworks, the representation of actions in a compact way has given rise to some problems known as the frame, the ramification and the qualification problems [39, 24, 38]. In propositional logic, these problems were solved by a majority of approaches by introducing a special symbol for expressing a causal rule relating preconditions of an action to its effect (indeed classical implication cannot separate a cause from a consequence due to contraposition). Actions are first described by such rules, then given the set of causal rules, a set of formulas (called frame axioms) are generated stating that any fluent f is true at time $t+1$ if and only if it was (a) true at t and no causal rule concluding $\neg f$ can be fired at t or (b) false at t and a causal rule concluding f can be fired. As a proof of concept, we choose to use *propositional logic* in this article where we face a problem that can be viewed as an extension of belief extrapolation with narrative tension analysis. Moreover in order to perform non-monotonic reasoning (which allows agents to change their minds and thus accept surprises), we propose to use default rules of the form $a \rightsquigarrow b$ to encode causal relations. Note that the field of non-monotonic reasoning has been extensively studied and many powerful approaches were proposed (see e.g. [29] for an overview), here, we choose to rely on a simple formalism at first.

Logical approaches to computational narratives have been proposed in the past. In [9], (Intuitionistic) Linear logic has been argued to be a suitable representational model for narratives for its capacity to finely represent narrative actions through the production and consumption of resources. This language provides the symbol $\multimap$ which can be used in $A \multimap B$ to express the validity of transforming resource A into resource B, the flow of resources consumption through the associated sequent calculus allowing to establish causal relations. Dynamic logic [30] and its epistemic extensions [7] are formalisms with higher expressiveness. In this work, we propose to characterize narrative tension phenomenon in *propositional logic* (extended with default rules) to demonstrate the representational uniformity of these concepts and their relationships with each other. We will explore how to encode them in aforementioned logics in the future, keeping in mind the challenges raised by their operationalization in their most expressive fragments [3].

2.2 Emotions supporting narrative tension

Psychological models of emotions are often used in the field of affective computing such as models from Ekman [22] or Plutchik [45] (which include surprise). Other works [42] consider that every emotion should have a valence and, as a consequence, surprise, which is inherently neither good nor bad, is considered as a different affective phenomenon. As we position ourselves in the context of studying the emotional states of a listener, we will rely on the characterizations given by Baroni in [5]. Curiosity occurs when there is a partial omission of crucial knowledge: at a given stage when experiencing the storytelling experience, the listener knows they are missing important information. Suspense arises when an event could potentially lead to an impacting result — be it good or bad — to the storyline, and is correlated with anticipation. Surprise results from a rupture from previous expectations, which retroactively invalidates some of the predictions made by the listener: the listener has expectations about how the story will develop, based on story genre or common sense. Going against these expectations while maintaining coherence is what causes surprise. Baroni distinguishes curiosity and suspense from surprise, as the former two are tied to anticipation and an urge to know, whilst the latter arises sporadically as the narrative progresses, which will be reflected in our model.

Related to suspense, the concept of *narrative closure* also reflects the epistemic nature of storytelling (as theorized by Carroll [13]): this encompasses the phenomenological feeling of finality that is generated when all the questions saliently posed by the narrative to the listener are answered.

Previous work in the psychology of narrative understanding [54] has also tied the perception of the importance of story events to causal relationships' perception. In this paper we borrowed from this work, especially tracing a graph representing the narrative with nodes being actions, preconditions and effects and edges being causal relations. We will consider the degree of a node as reflecting its importance in the narrative, reading it as, the more an action is a consequence and has consequences the more important it is.

2.3 Logical models of emotions, surprises and awareness

The logical representation of emotions has already received some attention, see e.g. Lorini [35] or Adam [1] who formalized emotions based on the OCC theory [42]. In these works (which relies on a modal logic for BDI (Belief-Desire-Intention) agents), an agent has beliefs, including beliefs about what is *good for herself*, and expresses different emotions such as joy or sadness.

The particular case of *surprise* was studied by several authors in computer science, but the first study is due to an economist named Shackle [48] who defined the degree of surprise associated with an event as the degree of impossibility of this event given the uncertain knowledge about the situation considered. In Lorini and Castelfranchi [34], the role of surprise is investigated in the context of belief update. They associate a surprise with a difficulty to integrate the new piece of information, this occurs when there is a form of inconsistency between expectation and perception. Surprise was recently formalized in the context of the analysis of jokes by [20], indeed surprise has been considered as an important ingredient for laughter by many authors, the model of surprise of [20] is based on a revision operator and non-monotonic reasoning: to be surprised the listener of a joke should be able to jump to conclusions that can be questioned and even revised.

The characterization of *curiosity* provided by Baroni emphasizes that the listener is aware of its incomplete knowledge and that surprise is linked to a notion of disturbance which makes the agent to question his assumptions/beliefs and leads him to reconsider his understanding of the story. This reconsideration reminds the operation of *awareness raising* introduced by [55] to allow agents "to make their implicit knowledge explicit". Logical models taking into account agent's awareness have previously been defined in the literature. As Halpern [28] states, traditionally when reasoning about agents' beliefs, it is assumed they are aware of every proposition. Modica and Ristichini [40] first came up with a definition of awareness based on knowledge, stating that an agent is aware of p if he knew p or if he knew he did not know p. Halpern extends on this by introducing *implicit* knowledge, where agents are aware of all propositions and can reason with them ; and *explicit* knowledge, which captures the conclusions of which the agent is explicitly aware of. In this system, explicit knowledge is also implicit, while the reverse is not necessarily true.

Previous works have proposed models for agents in computational narratives such as [41] or [11] based on BDI. In [47], a BDI agent aiming to simulate player behavior in interactive stories takes into account the player personality. Other work has assigned personality stereotypes to users [53, 4] according to their interactions with the system. Whilst such models allow personalizing an interactive narrative, they would not enable a storytelling engine to finely drive the narrative tension. By contrast, the *Suspenser* system by [14] offers an operationalization for suspense elicitation, one of the three drivers of narrative tension. In *Suspenser*, suspense is maximized by ordering multiple story bits at the discourse level. We lay in this paper the groundwork for ultimately representing in a unified logical framework suspense, curiosity and surprise, the three drivers of narrative tension. This will build strong foundations for future generative and interactive systems able to operate both at the story and discourse levels.

We approach the modelization of curiosity, suspense and surprise as constructs at given moments of a narrative experience from the listener's beliefs and by non-monotonic reasoning about these beliefs.

Doing so, we believe our model is compatible with previous formalization while providing new insights.

3 Formalizing curiosity, surprise and suspense

We first present an example to illustrate the concepts introduced throughout this article.

Example 1 (The box)**.** *To illustrate the framework, we present a short story involving three agents, Albert, Erwin as well as a protagonist Cecilia*[2] *(respectively agents A, E and C). A short narrative : "Cecilia enters her office. She sees a box lying on her desk that was not there when she last left the room." Our hypothesis is that this event sparks curiosity in Cecilia's mind. We look at it from the point of view of Cecilia who reasons in a closed world where nothing, except three particular events (Albert putting a box on Cecilia's desk, Erwin doing it, Cecilia opening the box) can interact with the state of the world.*

We consider a set of variable symbols $\mathcal{V}$ denoted by Latin lower case letters, from this set of symbols we build the vocabulary $\mathcal{V}_T$ containing all variables of $\mathcal{V}$ indexed by all the integers taken in the set $T = \{0, 1, .., N\}$ representing time points. $\mathcal{L}$ is the propositional language based on $\mathcal{V}_T$ with the usual connectors and constants $\vee$, $\wedge$, $\neg$, $\rightarrow$, $\equiv$, $\bot$ and $\top$ denoting respectively the logical connectors "or", "and", "not", material implication and logical equivalence, contradiction, and tautology. The symbol $\models$ represent satisfiability. Let Ω denote the set of interpretations induced by $\mathcal{V}_T$, we will often use ω for naming a particular interpretation in Ω, each interpretation will be described by the list of literals satisfied by it, e.g., considering the vocabulary $\mathcal{V} = \{a, b\}$, and a set of two time points

[2] We consider a story involving Albert Einstein, Erwin Schrödinger and Cecilia Payne-Gaposchkin, hence the cat in the title.

$T = \{0, 1\}$ $\omega = (a_0,\ \neg b_0,\ \neg a_1, \neg b_1)$ is an interpretation in Ω that associates the truth value True to a and False to b at time step 0 and False to a and b at time step 1. The set $Mod(A) \subseteq \Omega$ is the set of interpretations satisfying the set of propositional formulas $A \subseteq \mathcal{L}$ ($Mod(A) = \{\omega \in \Omega \mid \omega \models \bigwedge_{\varphi \in A} \varphi\}$), the same notation is used to represent the set of models of a formula $Mod(\varphi) = \{\omega \in \Omega \mid \omega \models \varphi\}$.

Example 1 (continued): *To study this flow of events taking place in 4 time steps denoted $T = \{0, 1, 2, 3\}$ we need a vocabulary $\mathcal{V} = \{box, A, E, C, empty, vis\}$ meaning respectively there is a box on Cecilia's desk, agent A puts a closed box on the desk, agent E puts a closed box on the desk, agent C opens the box, there is nothing in the box and something inside the box is uncovered (and thus the box has been opened). In the language $\mathcal{L}$ built on $\mathcal{V}$ and T, the following expression is an example of a well-formed formula: $(A_0 \vee E_0) \wedge box_1 \wedge C_2 \wedge \neg empty_2$.*

Default rules are rules that tolerate exceptions and allow us to reason in presence of incomplete information, by assuming that the situation is not exceptional when there is no evidence for the contrary. The notation $\alpha \rightsquigarrow \beta$ (with $\alpha, \beta \in \mathcal{L}$) is used to represent a default rule interpreted as when α is true, it is more plausible that β is true than false.

Example 1 (continued): *In order to be able to encode this example we propose to use default rules to express that by default some fluents keep their value: the following rule is expressing that when there is no box at time point 0 then by default there is no box at time point 1: $\neg box_0 \rightsquigarrow \neg box_1$. This rule admits exceptions: namely, if A puts a box on the desk at time point 0 then generally there is a box at time point 1: $A_0 \wedge \neg box_0 \rightsquigarrow box_1$.*

Given a set of default rules Δ it is possible to define a ranking of these rules according to their specificity, thanks to "System Z" algorithm [43], the default base is then called stratified, its stratas are the formulas with the same rank[3]. Note that there are sets of default rules that do not admit a Z ordering, such default sets are called "inconsistent" in [26]. In this paper, we restrict ourselves to consistent default sets. From a stratified default base lexicographic-entailment [6, 33] is a non-monotonic inference relation which imposes that the more specific the rules, the more mandatory it is to comply with them:

Definition 1 (Lex-inference [6])**.** *Let $\Delta = \Delta_1 \cup \cdots \cup \Delta_n$ be a stratified default base with n strata ordered from the most specific strata Δ_1 to the least specific one Δ_n, and let A and B be two subsets of Δ, and α, β be two formulas of $\mathcal{L}$,*

- Notations: str (for "strict") is a function that translates a set of default rules into a set of formulas of $\mathcal{L}$, i.e., $str(A) = \bigcup_{\alpha \rightsquigarrow \beta \in A} \{\neg\alpha \vee \beta\}$. For all $i \in [1, n]$, and any $E \subseteq \Delta$, E_i denotes the ith strata of E: $E_i = E \cap \Delta_i$.

[3]System Z ordering method is based on the tolerance notion between rules. More precisely, a rule $r = \alpha \rightsquigarrow \beta$ is tolerated by a set of n rules $R \subseteq \Delta$ iff $\alpha \wedge \beta \wedge \bigwedge_{\alpha_i \rightsquigarrow \beta_i \in R} (\neg\alpha_i \vee \beta_i)$ is consistent. The process continues until Δ contains only rules tolerated by all the other ones, they constitute the most specific stratum called Δ_1 (Δ_n being the least specific stratum, with n being the number of iterations).

- A is Lex-preferred *to B given Δ, denoted $A \succ_\Delta B$,*

$$\textit{iff there exists } k \in [1, n] \textit{ s.t. } \begin{cases} |A_k| > |B_k| \textit{ and} \\ \forall i < k,\ |A_i| = |B_i| \end{cases}$$

- A is a Lex-preferred *α-consistent subbase of Δ if $A \subseteq \Delta$ and $str(A) \cup \{\alpha\} \not\models \bot$ and for any $B \subseteq \Delta$ s.t. $str(B) \cup \{\alpha\} \not\models \bot$, $B \not\succ_\Delta A$ holds*

- $\alpha \mid\!\sim_\Delta \beta$ iff for any Lex-preferred α-consistent subbase B of Δ, $str(B) \cup \{\alpha\} \models \beta$

Example 1 (continued): *Let us consider that the common knowledge Δ about the world consists only in the default persistence of the fluents box, empty and vis and on the default effects of the occurrences of events A, E and C when their preconditions hold.*

$\neg box_0 \rightsquigarrow \neg box_1$	$(A_0 \vee E_0) \wedge \neg box_0 \rightsquigarrow box_1$
$box_0 \rightsquigarrow box_1$	$C_0 \wedge \neg vis_0 \rightsquigarrow vis_1$
$\neg empty_0 \rightsquigarrow \neg empty_1$	$C_0 \wedge \neg vis_0 \wedge empty_0 \rightsquigarrow \neg vis_1$
$empty_0 \rightsquigarrow empty_1$	$\neg box_1 \rightsquigarrow \neg box_2$
$\neg vis_0 \rightsquigarrow \neg vis_1$	...
$vis_0 \rightsquigarrow vis_1$	$C_2 \wedge \neg vis_2 \wedge empty_2 \rightsquigarrow \neg vis_3$

System Z will give a stratification in three strata where all persistence rules (of the form $v_t \rightsquigarrow v_{t+1}$ or $\neg v_t \rightsquigarrow \neg v_{t+1}$) are in the least specific stratum Δ_3 (since they are tolerated by all the other rules). As seen before, $(A_0 \vee E_0) \wedge \neg box_0 \rightsquigarrow box_1$ describes an exception to the persistence of $\neg box$, just as $C_0 \wedge \neg vis_0 \rightsquigarrow vis_1$ describes an exception to the persistence of $\neg vis$ which leads us to place them in Δ_2, the latter itself admits an exception described by rule $C_0 \wedge \neg vis_0 \wedge empty_0 \rightsquigarrow \neg vis_1$ making it the most specific rule thus placed in Δ_1 by System Z algorithm. At the end, we get:

$$\Delta_1 = \{C_t \wedge \neg vis_t \wedge empty_t \rightsquigarrow \neg vis_{t+1}\}_{t \in \{0,1,2\}}$$

$$\Delta_2 = \left\{ \begin{matrix} C_t \wedge \neg vis_t \rightsquigarrow vis_{t+1} \\ (A_t \vee E_t) \wedge \neg box_t \rightsquigarrow box_{t+1} \end{matrix} \right\}_{t \in \{0,1,2\}}$$

$$\Delta_3 = \left\{ \begin{matrix} v_t \rightsquigarrow v_{t+1} \\ \neg v_t \rightsquigarrow \neg v_{t+1} \end{matrix} \right\}_{\substack{t \in \{0,1,2\} \\ v \in \{box, empty, vis\}}}$$

Using lexicographic inference we get: $\neg box_0 \mid\!\sim_\Delta \neg box_1$ and $\neg box_0 \wedge (A_0 \vee E_0) \mid\!\sim_\Delta box_1$, meaning that a priori if there was no box at time 0, there is no box at time 1, but knowing that either A or E has placed a box makes it more plausible that there is a box at time 1.

Note that in this example, for the sake of simplicity, we want to make a closed world assumption (CWA) in order to express that the only possible way to change the variable box (respectively vis) from false to true is the occurrence of A or E (respectively the performance of action C):

$$\begin{aligned} \text{CWA} = &\{(\neg box_t \wedge box_{t+1}) \rightarrow (A_t \vee E_t), \\ &(\neg vis_t \wedge vis_{t+1}) \rightarrow C_t\}_{t \in \{0,1,2\}} \end{aligned}$$

From the set of default rules and the closed world assumption, we can then obtain: $\{box_1\} \cup \text{CWA} \mid\!\sim_\Delta box_0$ meaning that the most plausible interpretation is that when there is a box at time point 1 it means that there was already a box at time 0. However, if we know that there were no box at time 0 then $\{box_1, \neg box_0\} \cup \text{CWA} \mid\!\sim_\Delta (A_0 \vee E_0)$

We choose to use the lexicographic entailment in this paper, because [6] have shown that it is a powerful non-monotonic inference relation that satisfies the set of rational properties called System P. The System P, introduced by Kraus, Lehmann and Magidor [32], gathers properties that should follow rationally when one wants to deduce new inferences from a set of existing inference rules. The following definition describes an agent epistemic states via the pieces of information that she believes.

Definition 2 (Agent epistemic state and inference)**.** *A user is represented by a tuple $B = (F, B_{\mathcal{L}}, B_{\Delta})$ composed of a set $F \subseteq \mathcal{L}$ of formulas representing facts, and two sets $B_{\mathcal{L}} \subseteq \mathcal{L}$ and B_{Δ} respectively representing the strict and default rules known by the agent, the default rules of B_{Δ} are expressions of the form $\alpha \rightsquigarrow \beta$ with $\alpha, \beta \in \mathcal{L}$.*
When $F \cup B_{\mathcal{L}}$ and B_{Δ} are both consistent[4]*, the user is equipped with an inference relation between formulas of $\mathcal{L}$ denoted $\mid\!\sim_B$ defined by:*

$$\alpha \mid\!\sim_B \beta \quad \textit{iff} \quad \left| \begin{array}{l} \{\alpha\} \cup F \cup B_{\mathcal{L}} \textit{ is consistent and} \\ \textit{for any Lex-preferred} \\ (\alpha \wedge \bigwedge_{\varphi \in F \cup B_{\mathcal{L}}} \varphi)\textit{-consistent} \\ \textit{subbase} A \in B_{\Delta}, \\ \qquad A \cup \{\alpha\} \cup F \cup B_{\mathcal{L}} \models \beta \end{array} \right.$$

In the following, $\mid\!\sim_B \varphi$ is a shortcut for $\top \mid\!\sim_B \varphi$.

In order to formally introduce curiosity, we need to define awareness. This will be done by simply stating that an agent is aware of a variable if this variable appears in the facts contained in its epistemic state, and we assume that when an agent is aware of a variable then it becomes also aware of every variable of the strict or default rules of its epistemic state containing this variable (mimicking a kind of introspection). We use the notation $v \in \varphi$ to express that the variable v appears in the formula φ, this notation can be applied to variables of $\mathcal{V}_T$ as well as of $\mathcal{V}$.

Definition 3 (awareness)**.** *An agent represented by $B = (F, B_{\mathcal{L}}, B_{\Delta})$ is*

- aware of a variable $v \in \mathcal{V}$ *if*
  - *there is a formula $\varphi \in F$ s.t. $v \in \varphi$ or*
  - *there is a formula $\varphi \in B_{\mathcal{L}} \cup str(B_{\Delta})$ s.t. $v \in \varphi$ and there is a variable $v' \in \varphi$ of which the agent is aware; and*
- aware of a formula $\varphi \in \mathcal{L}$ *iff for any variable $v_t \in \varphi$, the agent is aware of v.*

Example 1 (continued): *Let us consider that the epistemic state of agent C is $(\emptyset, \text{CWA}, \Delta = \Delta_1 \cup \Delta_2 \cup \Delta_3)$, in this case it does not know any fact, which means that it is not aware of anything. Assume now that at time point 1, our agent Cecilia comes to her office and sees a box on her desk, then the epistemic state of agent C is $(\{box_1\}, \text{CWA}, \Delta)$. In this state, it is aware that a box is on the desk, moreover by introspection its is aware of the possibility to open it due to rule concerning C, the possibility that Albert or Erwin are able to put it on the desk, the possibility that this box is empty or that something inside of it could be vis.*
The following definition enables us to keep only formulas that do not concern a time point later than a given time point t, i.e., keep the formulas such that all their variables are indexed by time points no later than t.

Definition 4 (epistemic state until t)**.** *Given an epistemic state $(F, B_{\mathcal{L}}, B_{\Delta})$ and a time point $t \in [0, N]$, the* epistemic state until t*, denoted $B_{\to t} = (F_{\to t}, B_{\mathcal{L} \to t}, B_{\Delta \to t})$, is defined by:*
$F_{\to t} = \{\varphi \in F \mid \textit{for all } v_{t'} \in \varphi, t' \leq t\}$,
$B_{\mathcal{L} \to t} = \{\varphi \in B_{\mathcal{L}} \mid \textit{for all } v_{t'} \in \varphi, t' \leq t\}$,
$B_{\Delta \to t} = \{\delta \in B_{\Delta} \mid \textit{for all } v_{t'} \in str(\{\delta\}), t' \leq t\}$.

In the following, we use $[\varphi]_{<t}$ (and respectively $[\varphi]_{\leq t}$, $[\varphi]_{>t}$, $[\varphi]_{\geq t}$ and $[\varphi]_t$) to denote that φ is a formula containing only variables indexed by time points earlier than t (resp. earlier than or equal to t, strictly later than t, later than or equal to t, equal to t).

Remark 1. *For any formula $\varphi \in F_{\to t} \cup B_{\mathcal{L} \to t} \cup str(B_{\Delta \to t})$, $[\varphi]_{\leq t}$ holds.*

An agent is curious about a formula at time point t if according to its epistemic state until t it is aware of this formula but it is not able to deduce its truth value at time t.

Definition 5 (curiosity)**.** *An agent with state B is* curious about $\varphi \in \mathcal{L}$ *at $t \in T$ if, according to $B_{\to t}$, it is aware of φ and $\not\mid\!\sim_{B_{\to t}} \varphi$ and $\not\mid\!\sim_{B_{\to t}} \neg\varphi$.*

Example 2. *Coming back to Example 1, the epistemic state of C being $B = (\{box_1\}, \text{CWA}, \Delta)$, its state at 0 is $B_{\to 0} = (\emptyset, \emptyset, \emptyset)$, meaning that at 0 it is aware of nothing, thus according to Definition 5 it is not curious about anything at 0. In the epistemic state B, she first thinks that $\{box_1\} \cup \text{CWA} \mid\!\sim_{\Delta} box_0$. However, she remembers that there was no box on her desk at time 0 before she left her office. Meaning that her epistemic state is now $B' = (\{\neg box_0, box_1\}, \text{CWA}, \Delta)$ which enables her to draw the inference $\{\neg box_0, box_1\} \mid\!\sim_{B'} (A_0 \vee E_0)$, however there is no way of knowing which of Albert or Bernard (or both) dropped off the box. More formally, we can say that Cecilia is curious about the possibility that Albert dropped off the box at time 0 because she is aware of this possibility and $\{\neg box_0, box_1\} \not\mid\!\sim_{B'} A_0$ and $\{\neg box_0, box_1\} \not\mid\!\sim_{B'} \neg A_0$. Now if we consider that Albert told Cecilia that he placed a box on her desk at 0 before she entered her office. In that case, the epistemic state of Cecilia is $B'' = (\{A_0, \neg box_0, box_1\}, \text{CWA}, \Delta)$, there is no more ambiguity as she knows who put it there, hence she is not curious about A_0.*

To define suspense, we propose for this formalization to use Baroni's description of *primary suspense* [5] which relies solely on temporal and belief factors. Baroni also describes other types of suspense involving different

[4] Here consistent is not used with the same meaning: for the propositional formulas it means classical logic consistency, while for the default rules base it means that the base can be stratified.

emotional components (empathy and identification with a protagonist for instance). These components affect suspense by strengthening the intensity of curiosity, and we will leave them for further study at the time being. The following definition expresses that an agent feels suspense about a formula φ when this agent is curious about it at time t, and thinks that it is not impossible for facts or events (below denoted ψ) to come to light and reveal the truth of φ (satisfying curiosity about it at last).

Definition 6 (suspense)**.** *An agent represented by an epistemic state $B = (F, B_{\mathcal{L}}, B_{\Delta})$ feels* suspense about $\varphi \in \mathcal{L}$ at time point t *if*

1. *according to B, the agent is curious about φ at time t*
2. *and there is a formula $\psi \in \mathcal{L}$ such that $[\psi]_{>t}$ and $F_{\rightarrow t} \cup B_{\mathcal{L}} \cup \{\psi\}$ consistent*
3. *and there is $t' > t$ s.t. either $\mid\!\sim_{B'} \varphi_{t'}$ or $\mid\!\sim_{B'} \neg\varphi_{t'}$ holds, with $B' = (F \cup \{\psi\}, B_{\mathcal{L}}, B_{\Delta})$.*

Example 3. *In the context of Example 1, assume now that agent C has the following epistemic state $B = (\{\neg box_0, box_1, \neg vis_1\}, \mathrm{CWA}, \Delta)$. Here, at time 1 agent C is aware of the box, she is also aware that it is either empty or not, but has no way at this time to know which is true. Formally, $\not\mid\!\sim_B empty$ and $\not\mid\!\sim_B \neg empty$. Hence she is curious about the variable $empty$ at time point 1. Still according to Definition 3, the agent is also aware of the formulas $(C_2 \wedge \neg vis_2 \rightsquigarrow vis_3)$ and $(C_2 \wedge \neg vis_2 \wedge empty_2 \rightsquigarrow \neg vis_3)$. Meaning she is aware she will know the content of the box once she opens it.*
More precisely, the formula $\psi = C_2 \wedge vis_3$ can be added to the facts of the epistemic state because $\{\neg box_0, box_1, \neg vis_1\} \cup \mathrm{CWA} \cup \{C_2 \wedge vis_3\}$ is consistent. Now, $B' = (\{\neg box_0, box_1, \neg vis_1, C_2, vis_3\}, \mathrm{CWA}, \Delta)$ yields $\mid\!\sim_{B'} \neg empty_2$. Hence Cecilia feels suspense at time 1 about the truth value of $empty$.

In order to formalize surprise, following [20], we propose to exploit our non-monotonic setting that enables agents to imagine several more or less plausible situations, i.e., enables them to incorporate new contradicting information by revising previous conclusions. This is required in order to avoid locking the agent in a state of total incomprehension. Surprise can then be defined by the occurrence of a formula that was unexpected but which is completely plausible.

Definition 7 (surprise)**.** *An agent represented by $B = (F, B_{\mathcal{L}}, B_{\Delta})$ is* surprised at time t about a formula $\varphi \in \mathcal{L}$ *if $\varphi \in F_{\rightarrow t}$ and $B_{\rightarrow t}$ is consistent (φ occurred and it was not impossible) and $B' = (F_{\rightarrow t-1}, B_{\mathcal{L}\rightarrow t}, B_{\Delta\rightarrow t})$ is such that: $\mid\!\sim_{B'} \neg\varphi$ (φ was unexpected)*

Example 2 (continued): *Cecilia is surprised to find the box at time 1. Indeed, given the epistemic state $B = (\{\neg box_0, box_1\}, \mathrm{CWA}, \Delta)$, before seeing the box at time 1, the persistence of $\neg box_0$ into $\neg box_1$ was the most plausible evolution. More formally, we can check that $box_1 \in F$ and $B' = (\{\neg box_0\}, \mathrm{CWA}_{\rightarrow 1}, \Delta_{\rightarrow 1})$ is such that $\mid\!\sim_{B'} \neg box_1$, and B is consistent (hence $B_{\rightarrow 1}$ as well).*

4 Properties

In this section we show several simple properties relating the three emotions, moreover we establish the computational complexity of their detection.
An agent who knows nothing is aware of nothing.

Proposition 1. *If the epistemic state of an agent has no fact, i.e, $B = (\emptyset, B_{\mathcal{L}}, B_{\Delta})$ then the agent is not aware of any variable.*

Proof. Even if $B_{\mathcal{L}}$ or B_{Δ} are non-empty, there is no awareness since no variable appears in F. □

This kind of agent is not curious nor able to feel suspense since curiosity requires awareness, and suspense requires curiosity.

Corollary 1 (of Proposition 1)**.** *If the epistemic state of an agent has no fact, i.e., $B = (\emptyset, B_{\mathcal{L}}, B_{\Delta})$, then the agent is not curious and does not feel suspense about any formula at any time point.*

The following proposition states that an omniscient agent (i.e., an agent with complete information about the world) is never curious nor able to feel suspense.

Proposition 2. *If the epistemic state $B = (F, B_{\mathcal{L}}, B_{\Delta})$ of an agent admits only one most plausible interpretation in Ω, then for any finite formula, there is no time point where the agent is curious or feels suspense about it.*

Proof. In order to be curious, there should exist at least one variable whose truth value is unknown. Hence there should be at least two interpretations that are equally most plausible. □

Because surprise occurs when the agent expects something and then the opposite happens, it means that it is not curious about it (because the surprise makes it know it).

Proposition 3. *Given an epistemic state B of an agent, if the agent is surprised about φ at time t then the agent is not curious about φ neither at time $t-1$ nor at time t.*

Proof. Let us assume that $B = (F, B_{\mathcal{L}}, B_{\Delta})$ is *surprised at time t about a formula $\varphi \in \mathcal{L}$*, it means that $\varphi \in F_{\rightarrow t}$ and $B_{\rightarrow t}$ is consistent and $B' = (F_{\rightarrow t-1}, B_{\mathcal{L}\rightarrow t}, B_{\Delta\rightarrow t})$ is such that: $\mid\!\sim_{B'} \neg\varphi$. It means that the agent could infer the truth value of φ at time $t-1$, hence she was not curious at $t-1$. Now since $\varphi \in F$ and $B_{\rightarrow t}$ consistent then $\mid\!\sim_{B_{\rightarrow t}} \varphi$ hence she is not curious about it at t. □

This proposition shows that surprise and curiosity are antagonists in a given epistemic state, however we can imagine stories where the same event sequence may produce curiosity (e.g. by keeping some information hidden, namely the name of the murderer) when told in a given way and surprise when told differently (e.g. revealing this same information at start). The following proposition shows the complexity class of the decision problems associated to awareness, curiosity, suspense and surprise.

Proposition 4. *Given an epistemic state B, a formula $\varphi \in \mathcal{L}$ and a time point $t \in T$,*

- *Deciding whether B is aware of φ at time point t can be done in linear time.*
- *Deciding whether B is curious or feels suspense or surprise about φ at t is Δ_2^p-complete.*

Proof. In order to check awareness about a variable, it is enough to check membership of this variable to a set of formulas, which is linear in the size of the epistemic state, this process should be repeated for all the variables of a formula to check formula awareness. Concerning curiosity, in addition to a test of awareness, it uses two lexicographic inference tests which have been shown to be in P^{NP} by [21]. Suspense requires a curiosity check and a consistency check of the strict part of the base B, which is a SAT problem hence NP-complete. It then requires several lexicographic inferences in order to find the time point where $\not\sim_{B'} \varphi_{t'}$ or $\not\sim_{B'} \neg\varphi_{t'}$ holds. Surprise requires a consistency check of the default base of B (which is a P^{NP}-complete problem according to [21]) and a lexicographic inference, hence the result. □

The complexity P^{NP} of these decision problems is due to the use of the lexicographic inference in their definition. Note that the upper bound (N) on time steps could relieve the computational complexity as obtained in traditional STRIPS planning [10] where the complexity of certain decision problems drops from PSPACE-complete to NP-complete. Note also that formulation of AI planning in answer set programming gives rise to similar complexity [50].

5 Curiosity and causality

For further characterizing narrative tension, we need to quantify the intensity of the emotions generated in an agent when listening to a story. This section is a first attempt towards this goal.

5.1 Curiosity intensity

We propose a definition of the emotional intensity of curiosity. In the following definition we propose to rely on findings from Trabasso and Sperry [54] as a heuristic in order to evaluate the intensity of the curiosity. We first define the causal graph associated with an epistemic state as the one relating variables of $\mathcal{V}_T$ with the links (called E_B) induced by the default rules (B_Δ) and strict rules ($F \cup B_\mathcal{L}$) of the epistemic state B.

Definition 8 (causal graph)**.** *The* causal graph $\mathcal{G}_B$ *induced by an epistemic state $B = (F, B_\mathcal{L}, B_\Delta)$ is a pair (V_B, E_B):*

- $V_B = \{v_t \in \mathcal{V}_T \mid v_t \in \varphi, \varphi \in F \cup B_\mathcal{L} \cup str(B_\Delta)\}$ *is the set of vertices of $\mathcal{G}_B$*
- $E_B = \{(v_t, v'_{t'}) \in \mathcal{V}_T \times \mathcal{V}_T \mid v_t \in \alpha, v_{t'} \in \beta, \alpha \rightsquigarrow \beta \in B_\Delta\}$ $\cup \{(v_t, v'_{t'}) \in \mathcal{V}_T \times \mathcal{V}_T \mid \{l_t\} \cup F \cup B_\mathcal{L} \models l'_{t'}$ *with $l_t \in \{v_t, \neg v_t\}, l'_t \in \{v'_{t'}, \neg v'_{t'}\}$*

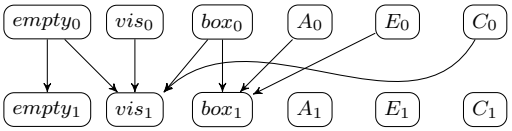


Figure 1: Causal graph induced by the epistemic state $B_{\to 1}$

We illustrate this definition on the epistemic state of Cecilia until time point 1.

Example 3 (continued): *Considering $B = (\{\neg box_0, box_1, \neg vis_1\}, \text{CWA}, \Delta)$, the causal graph induced by $B_{\to 1}$ is shown in Figure 1.*

Definition 9 (curiosity intensity)**.** *Given an agent with state $B = (F, B_\mathcal{L}, B_\Delta)$ and* curious about $\varphi \in \mathcal{L}$ *at $t \in T$, her curiosity intensity level is $c_B(\varphi, t) = \sum_{v_{t'} \in \varphi} deg(v_{t'})$ where $deg(x)$ is the degree of the node x in the causal graph induced by B.*

Example 2 (continued): *Given the epistemic state of Cecilia $B = (\{box_1\}, \text{CWA}, \Delta)$, she is the most curious about vis_1 with intensity 5, denoted $c_B(vis, 1) = 5$. Note that the degree of vis_1 is only four on Figure 1 but there is a supplementary outgoing arc from vis_1 to vis_2 when considering B instead of $B_{\to 1}$.*

Note that another way to refine the curiosity intensity analysis is to take into account the user profile and her preferences over different subjects, which is left for further studies.

5.2 LLMs for building causal graphs

As seen in the previous section the causal graph is crucial to characterize the basic ingredients of narrative tension. In particular we need this graph both for capturing the dynamic laws governing the world and the saliency of each event (hence useful to refine the intensity of curiosity, suspense, and surprise). We first assumed that this graph could be extracted from an encoding of the epistemic state of the agent. Another approach is to take advantage of LLMs in order to obtain this graph, As presented by [52]. While we focused here on a simple story, whose events were transcribed into logic by hand, they tackled the question of generating a narrative graph of salient events from texts. Such system would enable to test our framework on much more complex narratives by automating the generation of such event graphs. Moreover they provide a structured way to identify successively hierarchical, temporal and causal dependencies between events. More precisely, the approach of [52] proceeds in five steps translated in cascading prompts: 1) Summarize the text 2) Capture the events, 3) Generate a hierarchical graph of events, 4) Generate a temporal graph, 5) Generate a causal graph. We[5] experimented their approach on our simple story. It amounted to send a first prompt (corresponding to step 2):

[5]The experiment was conducted by Benjamin Callac a.k.a. "Ben".

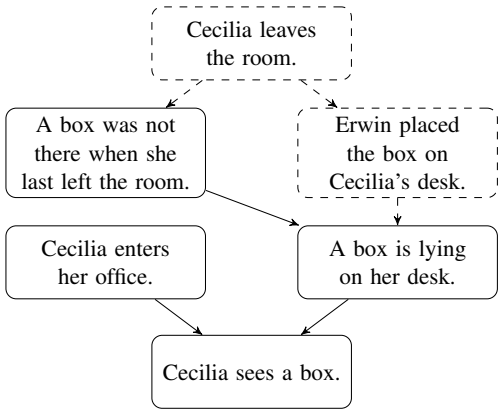


Figure 2: Causal graph generated by an LLM using Tan's [52] method. The two dotted events were added through further prompting of the LLM.

Ben> A structured event is something that happened as described in the text. A structured event is represented as a tuple, which consists of actors, a trigger, and objects. Could you list all the structured events in what follows? "We consider a story involving Albert Einstein, Erwin Schrödinger and Cecilia Payne-Gaposchkin[6] *: Cecilia enters her office. She sees a box lying on her desk that was not there when she last left the room."*
Resulting in the following list of events from ChatGPT-4o[7]:
GPT> Here are the structured events:
1. (Cecilia; enters; her office).
2. (Cecilia; sees; a box).
3. (A box; lying; on her desk).
4. (A box; was not; there when she last left the room).
Following the method used in [52] we then prompt a ChatGPT instance to complete a python program to generate hierarchical, temporal and causal graphs of events. In these prompts the different relations are described as :
Hierarchical Graph: "Each edge in the graph represents a *subevent* relation between the head and tail nodes which are events. They are closely related but on different granularity levels." [8]
Temporal Graph: "Each edge in the graph represents a *temporal* relation between the head and tail nodes which are events. An edge means the head event happens before the tail event temporally."[8]
Causal Graph: "Each edge represents a *causal* relation between the head and tail nodes which are events. An edge means the head event is *caused by* the tail event. The head event will not happen if the tail event did not happen."[8]

We can notice that this causal graph does not contain implicit events (like Albert or Erwin actions), since they were not present in the initial text. Moreover there is no genericity, i.e., the LLM underlies the actual causality in the story but do not give explicitly the causal laws. However each causal arc is labeled with a kind of generic explanation since we ask GPT to add comments for each edge created, e.g. for the edge: `(box_on_desk) -> (cecilia_sees_box)`, it is commented with "Cecilia can only see the box if it is on the desk".
Finally, we observe that the LLM was not initially able to infer events that were not described in the prompt. But, after prompting:
Ben> There are missing events in the story, complete the story with actions from either Albert or Erwin that Cecilia was not aware of.
GPT> (...) Earlier that day, Erwin Schrödinger had placed the box on Cecilia's desk.
Then after asking to complete the graph in python with the event that Erwin put the box there, GPT delivered a python program where it added an edge with the comment that ""A box is lying on her desk." is causally dependent on "Erwin placed the box on Cecilia's desk."". Moreover, we had to suggest to GPT to add the event of Cecilia leaving the room, which finally results in the graph presented in Figure 2.

6 Conclusion

This paper aims at providing a unified framework in which the three emotions at the heart of narrative tension, namely curiosity, surprise, and suspense are formalized and their relationships clarified. This framework is built on non-monotonic reasoning for representing compactly the default behavior of the world and also for simulating the reasoning of an agent in front of a story. The use of non-monotonic reasoning induces a cost in complexity: the detection problems associated with the three emotions are in P^{NP} (due to the use of lexicographic inference). We describe methods to evaluate the curiosity intensity.
While we illustrated our formalization by adopting the point of view of a single agent in a chronological story for the sake of clarity, it does not preclude its adaptability for storytelling using other points of views such as an extradiegetic narrator disclosing knowledge to the listener through a discourse that does not reflect the timeline of the story. To operationalize this model, we plan to investigate different frameworks that are equipped with solvers namely PDDL planning, Linear logic with Ceptre [36] and propositional default logic with TouIST [49]. Moreover, the inherent growing complexity of this problem for scaling to complex narratives requires further study about the granularity of story events, for instance inspired by discussions about the representation of causality [37]. The experiment based on [52] is very promising but further study is needed to establish a generic and guaranteed process for generating valuable causal graphs from any story.

Acknowledgements

We would like to thank the French Defence Innovation Agency (AID) for its support. We also extend our thanks to the reviewers for their insightful comments and ideas for enhancing this paper.

[6] Hence the cat in the title.
[7] "chatgpt-4o-latest" version as of the 15th March 2025.
[8] Prompt extracted from [52].

References

[1] Carole Adam, Andreas Herzig, and Dominique Longin. A logical formalization of the OCC theory of emotions. *Synthese*, 168(2):201–248, May 2009.

[2] Carlos E Alchourrón, Peter Gärdenfors, and David Makinson. On the logic of theory change: Partial meet contraction and revision functions. *The journal of symbolic logic*, 50(2):510–530, 1985.

[3] Guillaume Aucher and Thomas Bolander. Undecidability in epistemic planning. In F. Rossi, editor, *Proc. of Int. Joint Conf. on Artificial Intelligence (IJCAI'2013)*, pages 27—33, Beijing, China, 2013.

[4] Heather Barber and Daniel Kudenko. Generation of dilemma-based interactive narratives with a changeable story goal. In *2nd International Conference on INtelligent TEchnologies for interactive enterTAINment*. ICST, 5 2010.

[5] Raphaël Baroni. *La tension narrative: suspense, curiosité et surprise*. Poétique. Éd. du Seuil, Paris, 2007.

[6] Salem Benferhat, Claudette Cayrol, Didier Dubois, Jérôme Lang, and Henri Prade. Inconsistency management and prioritized syntax-based entailment. In *Proc. of Int. Joint. Conf. on Artificial Intelligence (IJCAI'93)*, volume 93, pages 640–645, 1993.

[7] Thomas Bolander and Mikkel Birkegaard Andersen. Epistemic planning for single-and multi-agent systems. *Journal of Applied Non-Classical Logics*, 21(1):9–34, 2011.

[8] Lorenzo Bonoli. Raphaël Baroni, La tension narrative. Suspense, curiosité, surprise, Paris, Seuil, 2007. *Cahiers de Narratologie. Analyse et théorie narratives*, 14, February 2008.

[9] Anne-Gwenn Bosser, Marc Cavazza, and Ronan Champagnat. Linear Logic for Non-Linear Storytelling. *ECAI 2010*, pages 713–718, 2010. Publisher: IOS Press.

[10] Tom Bylander. The computational complexity of propositional strips planning. *Artificial Intelligence*, 69(1-2):165–204, 1994.

[11] Rogelio E. Cardona-Rivera, Bradley A. Cassell, Stephen G. Ware, and R. Michael Young. Indexter : A Computational Model of the Event-Indexing Situation Model for Characterizing Narratives, 2012.

[12] Rogelio E. Cardona-Rivera, Arnav Jhala, Julie Porteous, and R. Michael Young. The story so far on narrative planning. *Proceedings of the International Conference on Automated Planning and Scheduling*, 34(1):489–499, May 2024.

[13] Noël Carroll. Narrative closure. *Philosophical Studies*, 135(1):1–15, August 2007.

[14] Yun-Gyung Cheong and R. Michael Young. Suspenser: A Story Generation System for Suspense. *IEEE Transactions on Computational Intelligence and AI in Games*, 7(1):39–52, March 2015.

[15] Christophe Dousson and Pierre Le Maigat. Chronicle recognition improvement using temporal focusing and hierarchization. In *IJCAI*, volume 7, pages 324–329. Citeseer, 2007.

[16] Florence Dupin de Saint-Cyr. Scenario Update Applied to Causal Reasoning. In *Principles of Knowledge Representation and Reasoning: Proceedings of the 11th International Conference, KR 2008*, pages 188–197, January 2008.

[17] Florence Dupin de Saint-Cyr, Anne-Gwenn Bosser, Benjamin Callac, and Eric Maisel. What killed the cat? Towards a logical formalization of curiosity (and suspense, and surprise) in narratives. In Michael Sioutis, Pietro Sala, and Fusheng Wang, editors, *LIPIcs*, volume 318, pages 10:1–10:16, Montpellier, France, October 2024. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[18] Florence Dupin de Saint-Cyr, Andreas Herzig, Jérôme Lang, and Pierre Marquis. Reasoning About Action and Change. In Pierre Marquis, Odile Papini, and Henri Prade, editors, *A Guided Tour of Artificial Intelligence Research*, volume 1 / 3 of *Knowledge Representation, Reasoning and Learning*, pages 487–518. Springer International Publishing, May 2020.

[19] Florence Dupin De Saint-Cyr and Jérôme Lang. Belief extrapolation (or how to reason about observations and unpredicted change). *Artificial Intelligence*, 175(2):760–790, February 2011.

[20] Florence Dupin de Saint-Cyr and Henri Prade. Belief revision and incongruity: Is it a joke? *Journal of Applied Non-Classical Logics*, 33(3-4):467–494, October 2023.

[21] Thomas Eiter and Thomas Lukasiewicz. Default reasoning from conditional knowledge bases: Complexity and tractable cases. *Artificial Intelligence*, 124(2):169–241, 2000.

[22] Paul Ekman. An argument for basic emotions. *Cognition and Emotion*, 6(3-4):169–200, May 1992.

[23] Jeffrey Ely, Alexander Frankel, and Emir Kamenica. Suspense and Surprise. *Journal of Political Economy*, 123(1):215–260, February 2015.

[24] Joseph Jeffrey Finger. *Exploiting constraints in design synthesis*. PhD thesis, Stanford University, Stanford, CA, 1987.

[25] Gerard Genette. *Nouveau Discours du Récit*. Seuil, 1983.

[26] Moisés Goldszmidt and Judea Pearl. On the consistency of defeasible databases. *Artificial Intelligence*, 52(2):121–149, 1991.

[27] Melanie Green and Timothy Brock. The Role of Transportation in the Persuasiveness of Public Narrative. *Journal of personality and social psychology*, 79:701–21, November 2000.

[28] Joseph Y. Halpern. Alternative Semantics for Unawareness. *Games and Economic Behavior*, 37(2):321–339, November 2001.

[29] Andreas Herzig and Philippe Besnard. *Knowledge Representation: Modalities, Conditionals, and Nonmonotonic Reasoning*, pages 45–68. Springer International Publishing, Cham, 2020.

[30] Kaarlo Jaakko Juhani Hintikka. *Knowledge and belief: An introduction to the logic of the two notions*. Cornell University Press, Ithaca and London, 1962.

[31] Hirofumi Katsuno and Alberto O Mendelzon. On the difference between updating a knowledge base and revising it. *KR*, 91:387–394, 1991.

[32] Sarit Kraus, Daniel Lehmann, and Menachem Magidor. Nonmonotonic reasoning, preferential models and cumulative logics. *Artificial intelligence*, 44(1-2):167–207, 1990.

[33] Daniel Lehmann. Another perspective on default reasoning. *Annals of mathematics and artificial intelligence*, 15:61–82, 1995.

[34] Emiliano Lorini and Cristiano Castelfranchi. The cognitive structure of surprise: looking for basic principles. *Topoi*, 26(1):133–149, 2007.

[35] Emiliano Lorini and François Schwarzentruber. A logic for reasoning about counterfactual emotions. *Artificial Intelligence*, 175(3):814–847, March 2011.

[36] Chris Martens. Ceptre: A language for modeling generative interactive systems. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 11, pages 51–57, 2015.

[37] Lawrence J. Mazlack. Granular causality speculations. In *IEEE Annual Meeting of the Fuzzy Information, 2004. Processing NAFIPS '04.*, volume 2, pages 690–695 Vol.2, 2004.

[38] John McCarthy. Epistemological problems of artificial intelligence. In *Proc. Int. Joint Conf on Artificial Intelligence (IJCAI'77)*, pages 1038–1044. Elsevier, 1977.

[39] John McCarthy and Patrick J Hayes. Some philosophical problems from the standpoint of artificial intelligence. *Machine Intelligence*, 4:463–502, 1969.

[40] Salvatore Modica and Aldo Rustichini. Awareness and partitional information structures. *Theory and decision*, 37:107–124, 1994.

[41] Emma Norling. Capturing the quake player: Using a BDI agent to model human behaviour. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 1080–1081, Melbourne Australia, July 2003. ACM.

[42] Andrew Ortony, Gerald L Clore, and Allan Collins. *The cognitive structure of emotions*. Cambridge university press, 2022 (original work published 1988).

[43] Judea Pearl. System Z: A natural ordering of defaults with tractable applications to nonmonotonic reasoning. In *Proc. 3rd Conf. on Theoretical Aspects of Reasoning about Knowledge*, pages 121–135, 1990.

[44] David Pizzi, Fred Charles, Jean-Luc Lugrin, and Marc Cavazza. Interactive Storytelling with Literary Feelings, April 2007.

[45] Robert Plutchik. A general psychoevolutionary theory of emotion. In *Theories of emotion*, pages 3–33. Elsevier, 1980.

[46] Vladimir Propp. *Morphology of the Folktale: Second Edition*. University of Texas Press, 1968.

[47] Jessica Rivera-Villicana, Fabio Zambetta, James Harland, and Marsha Berry. Using BDI to Model Players Behaviour in an Interactive Fiction Game. In Frank Nack and Andrew S. Gordon, editors, *Interactive Storytelling*, volume 10045, pages 209–220. Springer International Publishing, Cham, 2016.

[48] George Lennox Sharman Shackle. *Decision, Order and Time in Human Affairs*. (2nd edition), Cambridge University Press, UK, 1961.

[49] Khaled Skander Ben Slimane, Alexis Comte, Olivier Gasquet, Abdelwahab Heba, Olivier Lezaud, Frederic Maris, and Maël Valais. Twist your logic with TouIST. *CoRR*, abs/1507.03663, 2015.

[50] Tran Cao Son, Enrico Pontelli, Marcello Balduccini, and Torsten Schaub. Answer set planning: a survey. *Theory and Practice of Logic Programming*, 23(1):226–298, 2023.

[51] Meir Sternberg. How Narrativity Makes a Difference. *Narrative*, 9(2):115–122, 2001.

[52] Xingwei Tan, Yuxiang Zhou, Gabriele Pergola, and Yulan He. Cascading large language models for salient event graph generation. In Luis Chiruzzo, Alan Ritter, and Lu Wang, editors, *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 2223–2245, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics.

[53] David Thue and Vadim Bulitko. Modelling goal-directed players in digital games. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 2, pages 86–91, 2006.

[54] Tom Trabasso and Linda L Sperry. Causal relatedness and importance of story events. *Journal of Memory and Language*, 24(5):595–611, October 1985.

[55] Johan Van Benthem and Fernando R Velázquez-Quesada. The dynamics of awareness. *Synthese*, 177:5–27, 2010.

[56] Marianne Winslett. *Updating Logical Databases*. Cambridge University Press, 1990.

Un cadre paraconsistant pour l'évaluation de similarité dans les bases de connaissances

José-Luis Vilchis-Medina
ENSTA - Institut Polytechnique de Paris, Lab-STICC, Robex, Brest, 29200, France

jose.vilchis@ensta.fr

Résumé

*Cet article propose un **cadre paraconsistant pour l'évaluation de la similarité** dans les bases de connaissances. Contrairement aux approches classiques, ce cadre intègre explicitement les **contradictions**, permettant une mesure de similarité plus robuste et interprétable. Une nouvelle mesure S^* est introduite, pénalisant les incohérences tout en valorisant les propriétés partagées. Des **super-catégories paraconsistantes** Ξ_K^* sont définies pour organiser hiérarchiquement les entités de connaissances. Le modèle inclut également un **extracteur de contradictions** E et un mécanisme de réparation, garantissant la cohérence des évaluations. Les résultats théoriques assurent la réflexivité, la symétrie et les bornes de S^*. Cette approche offre une solution prometteuse pour la gestion des connaissances conflictuelles, avec des perspectives dans les systèmes multi-agents.*

Mots-clés

Mesure de similarité, Logique paraconsistante, Contradictions, Représentation des connaissances.

Abstract

*This article proposes a **paraconsistent framework for evaluating similarity** in knowledge bases. Unlike classical approaches, this framework explicitly integrates **contradictions**, enabling a more robust and interpretable similarity measure. A new measure S^* is introduced, which penalizes inconsistencies while rewarding shared properties. **Paraconsistent super-categories** Ξ_K^* are defined to hierarchically organize knowledge entities. The model also includes a **contradiction extractor** E and a repair mechanism, ensuring consistency in the evaluations. Theoretical results guarantee reflexivity, symmetry, and boundedness of S^*. This approach offers a promising solution for managing conflicting knowledge, with perspectives in multi-agent systems.*

Keywords

Similarity measure, Paraconsistent logic, Contradictions, Knowledge representation.

1 Introduction

Dans un contexte marqué par une prolifération croissante d'informations, l'interaction entre les systèmes formels et le raisonnement qualitatif s'avère essentielle [12]. La logique fournit une structure rigoureuse pour l'analyse des arguments et la résolution de problèmes complexes, tandis que le raisonnement qualitatif permet de saisir les nuances et le contexte souvent négligés par les approches purement quantitatives. Cette synergie est particulièrement cruciale dans les processus décisionnels, où les informations incomplètes ou contradictoires sont fréquentes [4].

Les bases de connaissances (BC) jouent un rôle central dans la gestion et la consultation d'informations structurées. Cependant, leur utilité est souvent entravée par des défis inhérents, tels que les contradictions, les données incomplètes et l'évolution dynamique des connaissances [17]. Les méthodes traditionnelles d'évaluation de la similarité dans les BC reposent généralement sur des approches numériques ou probabilistes, qui peinent à gérer efficacement ces problèmes [20]. En particulier, ces méthodes supposent souvent l'absence de contradiction, ce qui n'est pas réaliste dans des environnements complexes comme les systèmes juridiques ou médicaux [27]. De plus, elles présentent une rigidité hiérarchique qui limite leur adaptabilité face aux évolutions des structures de connaissance.

Pour pallier ces limitations, cet article propose un cadre fondé sur des principes de logiques paraconsistentes, un vaste ensemble de formalismes permettant de raisonner en présence de contradictions sans tomber dans la trivialité [1]. Contrairement aux logiques classiques, où une seule contradiction entraîne l'effondrement du système (principe d'explosion), les logiques paraconsistentes isolent les incohérences et permettent de continuer à raisonner sur les parties non contradictoires des connaissances [8, 21]. Bien qu'il existe de nombreuses variantes de logiques paraconsistentes, notre travail se concentre sur leur application originale au problème de la mesure de similarité dans les bases de connaissances.

Fondements de la logique paraconsistante

Parmi les différentes logiques paraconsistentes, nous adoptons un cadre symbolique basé sur des opérateurs non classiques pour modéliser les entités de connaissance. Cette approche permet de représenter explicitement les contradictions tout en conservant la cohérence globale du système. Le concept de similarité occupe une place centrale dans de nombreux domaines, notamment la médecine, le droit et l'intelligence artificielle [27]. Cependant, les mé-

thodes existantes rencontrent souvent des difficultés pour gérer les contradictions et les données incomplètes, conduisant à des évaluations erronées ou non pertinentes [22]. Notre cadre introduit la notion d'extracteur de contradictions et d'entités réparables, permettant des évaluations significatives même en présence d'incohérences. En résolvant les contradictions tout en préservant les structures hiérarchiques, cette approche garantit à la fois la cohérence et l'interprétabilité.

Cet article présente un cadre original pour surmonter les limites des mesures de similarité classiques. L'espace de propriétés de similarité Ξ_P est étendu avec des propriétés contradictoires (notées Ξ_P^*), permettant ainsi la formalisation de la similarité en présence de contradictions. La mesure de similarité paraconsistante proposée S^* quantifie la similarité en tenant compte explicitement des propriétés partagées et contradictoires, assurant une robustesse face aux données incomplètes ou conflictuelles. De plus, les super-catégories paraconsistentes Ξ_K^* fournissent une structure hiérarchique flexible pour organiser les entités de connaissances en fonction de leurs scores de similarité, améliorant ainsi l'interprétabilité et l'adaptabilité [28].

Contributions principales

Les contributions principales de ce travail incluent :

- Une définition formelle de Ξ_P^* et S^*, généralisant les mesures de similarité classiques pour intégrer les contradictions. Cette extension permet de capturer la nature nuancée des connaissances réelles, où des propriétés conflictuelles coexistent souvent.
- Des garanties théoriques, y compris la réflexivité, la symétrie de S^*, assurant des propriétés fondamentales pour les applications pratiques. Ces propriétés garantissent que la mesure de similarité reste cohérente et interprétable, même en présence de contradictions.
- Un cadre pour l'organisation hiérarchique des connaissances via Ξ_K^*, validé par des exemples issus des domaines médicaux et juridiques. Ce cadre permet de regrouper dynamiquement les entités de connaissances en fonction de leurs scores de similarité, reflétant ainsi les hiérarchies naturelles des domaines d'application.
- Un extracteur de contradictions $\mathcal{E}$ et un mécanisme de réparation, qui permettent de gérer les incohérences tout en préservant la cohérence des bases de connaissances. Ces outils sont essentiels pour garantir la robustesse du cadre dans des contextes dynamiques et complexes.

Cet article est structuré comme suit : la Section 2 passe en revue les travaux connexes et met en évidence les limitations des approches existantes. La Section 3 introduit le cadre proposé, en détaillant les définitions formelles, les propositions et les théorèmes. Ensuite, la Section 4 analyse les contributions. Enfin, la Section 5 conclut en résumant les apports principaux et en explorant les perspectives.

2 État de l'art

Approches logiques et probabilistes

L'évaluation de la similarité dans les bases de connaissances a été largement explorée à travers diverses méthodologies. Les approches traditionnelles reposent souvent sur la logique du premier ordre (FOL) [4] ou sur des modèles probabilistes [2]. Bien que les systèmes basés sur la FOL excellent dans le raisonnement formel, ils manquent de mécanismes pour gérer les contradictions, ce qui peut entraîner une trivialisation dans des contextes incohérents [23]. Les méthodes probabilistes, telles que les réseaux bayésiens [7], modélisent efficacement l'incertitude mais nécessitent des hypothèses distributionnelles fortes et peuvent échouer dans des ensembles de données clairsemés ou biaisés [18]. Les progrès récents en apprentissage profond [9] et en traitement du langage naturel (NLP) [6] ont considérablement amélioré les mesures de similarité sémantique. Cependant, ces méthodes restent gourmandes en données et souvent opaques dans leurs processus de raisonnement, limitant leur interprétabilité dans des domaines critiques tels que le droit ou la médecine [15, 16].

Méthodes basées sur les graphes et les ontologies

La théorie des graphes [11] et l'alignement des ontologies [24] offrent des représentations structurées des connaissances, permettant une meilleure interprétabilité et la modélisation de relations complexes [3, 25]. Cependant, ces approches sont confrontées à des défis computationnels importants, notamment lorsqu'il s'agit de traiter de grandes bases de données. Par exemple, l'alignement des entités dans les graphes de connaissances [24] nécessite souvent des systèmes à haute mémoire et un réglage manuel approfondi. Les approches basées sur les ontologies [14] améliorent l'interprétabilité mais peinent à suivre les mises à jour dynamiques et l'évolution des connaissances, limitant ainsi leur applicabilité dans des domaines en évolution rapide [10].

Limitations des cadres existants

Les méthodes actuelles pour évaluer la similarité dans les bases de connaissances présentent plusieurs limitations critiques :

- **Ignorance des contradictions :** Les mesures de similarité classiques [22] considèrent les contradictions comme irréconciliables, conduisant à une surestimation de la similarité. Cette approche ne capture pas la complexité des connaissances réelles, où des propriétés conflictuelles coexistent souvent.
- **Manque d'adaptabilité hiérarchique :** Les systèmes existants imposent souvent des classifications rigides, ne reflétant pas les hiérarchies nuancées inhérentes aux domaines de connaissances. Cette rigidité limite leur capacité à s'adapter à des structures évolutives ou spécifiques à un domaine.
- **Problèmes de transparence :** Les modèles boîte noire, en particulier ceux basés sur l'apprentissage

profond [9], entravent l'interprétabilité. Ce manque de transparence est particulièrement problématique dans des domaines comme le droit ou la médecine, où la justification des décisions est essentielle.

Enfin, les métriques de similarité classiques, bien qu'efficaces dans certains contextes, souffrent de limites importantes lorsqu'il s'agit de gérer des contradictions ou des incertitudes. Des exemples notables incluent :

— La distance de Levenshtein [30], qui ne prend pas en compte les incohérences sémantiques.
— La distance de Hamming [19], qui suppose des chaînes de longueurs identiques.
— La distance de Jaro-Winkler [29], qui reste sensible aux erreurs de transcription sans gérer les conflits logiques.

La logique paraconsistante comme solution

La logique paraconsistante [5] offre une alternative prometteuse en permettant un raisonnement avec des contradictions sans s'effondrer dans la trivialité [23]. Contrairement aux logiques traditionnelles, le raisonnement paraconsistant permet des comparaisons significatives même lorsque les entités de connaissances contiennent des propriétés conflictuelles [24]. Des travaux récents ont exploré l'intégration de la logique paraconsistante avec les systèmes de représentation des connaissances, démontrant son potentiel pour améliorer la robustesse et la transparence [13, 14].

Le cadre proposé s'appuie sur cette base pour répondre aux limitations des approches existantes :

— **Intégration des contradictions :** Un extracteur de contradictions et un mécanisme de réparation sont introduits, intégrant les contradictions dans Ξ_P^* pour éviter la perte d'information et assurer la cohérence.
— **Pénalisation des contradictions :** Le cadre définit S^* pour pénaliser les contradictions, assurant une robustesse face au bruit et améliorant la précision des évaluations de similarité.
— **Regroupement hiérarchique dynamique :** En exploitant Ξ_K^*, le cadre regroupe dynamiquement les entités de connaissances via des seuils de similarité θ, reflétant les hiérarchies de domaine et permettant une adaptabilité.

Malgré ces avancées, des défis subsistent, notamment dans la gestion de l'évolution dynamique des connaissances et la nature subjective de la sélection des propriétés [26]. Le cadre proposé aborde ces problèmes en introduisant des espaces de similarité adaptatifs et des mécanismes pour une sélection cohérente des propriétés, assurant à la fois la robustesse et l'adaptabilité dans les évaluations de similarité [28].

3 Cadre proposé

Le cadre proposé repose sur une modélisation rigoureuse des entités de connaissances au sein d'un langage de logique du premier ordre, permettant l'intégration explicite des contradictions dans l'évaluation de similarité. Ce cadre inclut une mesure de similarité paraconsistante S^*, une hiérarchie organisée via des super-catégories paraconsistantes Ξ_K^*, ainsi qu'un extracteur de contradictions E et un mécanisme de réparation pour gérer les incohérences. Les propriétés fondamentales garantissent la robustesse et l'interprétabilité du modèle.

3.1 Définitions formelles des entités de connaissance

Soit $\mathcal{K} = \{K_1, K_2, \ldots, K_n\}$ l'ensemble fini des entités de connaissances traitées dans la base de connaissances. Chaque entité K_i est définie comme un ensemble fini de littéraux clos dans un langage de logique du premier ordre $\mathcal{L}$, soit :

$$K_i = \{\ell_1, \ldots, \ell_m\}, \quad \text{où } \ell_j \in \text{Lit}(\mathcal{L}), \tag{1}$$

où $\text{Lit}(\mathcal{L})$ désigne l'ensemble des littéraux atomiques et négatifs clos de $\mathcal{L}$. Cette définition fournit une base sémantique claire pour évaluer les relations entre entités, tout en autorisant la coexistence de propriétés contradictoires. Chaque littéral ℓ_j peut représenter une propriété spécifique ou son négation, ce qui permet de modéliser des connaissances complexes et ambivalentes.

3.2 Espace de propriétés paraconsistantes

L'espace de propriétés de similarité classique Ξ_P est étendu pour intégrer explicitement les propriétés contradictoires, conduisant à la définition suivante :

Définition 1 (Espace paraconsistant des propriétés)**.** On définit l'espace paraconsistant des propriétés comme :

$$\Xi_P^* = \Xi_P \cup P_{\text{contradictory}}, \tag{2}$$

où :

— Ξ_P : ensemble des propriétés partagées par deux entités K_1 et K_2,
— $P_{\text{contradictory}}$: ensemble des couples de propriétés $(p, \neg p)$ entre K_1 et K_2.

Cette définition permet de formaliser les interactions entre entités de connaissances en présence de conflits, sans invalider l'ensemble du raisonnement. Elle introduit explicitement les contradictions, ce qui est crucial pour une évaluation de similarité robuste en contextes incertains ou conflictuels.

3.3 Mesure de similarité paraconsistante S^*

La mesure S^* est reformulée pour refléter à la fois la similarité positive et la divergence induite par les contradictions :

Définition 2 (Mesure de similarité paraconsistante)**.** Soient K_1 et K_2 deux entités de connaissances. On définit la mesure de similarité paraconsistante S^* comme :

$$S^*(K_1, K_2) = S^+(K_1, K_2) - D^\pm(K_1, K_2), \tag{3}$$

où :

$$S^+(K_1, K_2) = \frac{|P_{\text{shared}}|}{|P_{\text{total}}|}, \quad D^\pm(K_1, K_2) = \frac{|P_{\text{contradictory}}|}{|P_{\text{total}}|}. \tag{4}$$

Ainsi, $S^* \in [-1, 1]$, avec :

— $S^* > 0$: mesures nettement similaires,
— $S^* = 0$: équilibre entre similitude et divergence,
— $S^* < 0$: divergence dominante.

Cette formulation permet de pénaliser les contradictions tout en valorisant les propriétés partagées. Par exemple, si K_1 et K_2 partagent une propriété p mais contiennent également une contradiction $(p, \neg p)$, la mesure S^* sera négative, reflétant une divergence nette. Cela garantit une interprétation claire et intuitive de la similarité en présence de conflits.

3.4 Propriétés fondamentales de S^*

Les propriétés essentielles garantissant la cohérence du cadre sont les suivantes :

Proposition 1 (Réflexivité)**.** Pour toute entité de connaissance K, on a :

$$S^*(K, K) = 1. \tag{5}$$

Démonstration. Par définition, $P_{\text{shared}} = P_{\text{total}}$ et $P_{\text{contradictory}} = \emptyset$. Donc :

$$S^*(K, K) = \frac{|P_{\text{total}}|}{|P_{\text{total}}|} - \frac{0}{|P_{\text{total}}|} = 1 - 0 = 1. \tag{6}$$

□

Proposition 2 (Symétrie)**.** Pour tous $K_1, K_2 \in \mathcal{K}$, on a :

$$S^*(K_1, K_2) = S^*(K_2, K_1). \tag{7}$$

Proposition 3 (Bornes extrêmes)**.** Pour toutes entités K_1, K_2, la mesure S^* satisfait :

$$-1 \leq S^*(K_1, K_2) \leq 1. \tag{8}$$

Ces propriétés garantissent que S^* est une mesure cohérente et interprétable, même en présence de contradictions. La réflexivité et la symétrie sont des propriétés standard pour les mesures de similarité, tandis que les bornes garantissent que S^* reste dans un intervalle raisonnable.

3.5 Super-catégories paraconsistantes Ξ_K^*

Les entités de connaissances peuvent être organisées hiérarchiquement selon leur niveau de similarité via le seuil $\theta \in [-1, 1]$.

Définition 3 (Super-catégories paraconsistantes)**.** Étant donné un seuil θ, les super-catégories paraconsistantes forment un ensemble défini par :

$$\Xi_K^* = \bigcup_{i=1}^{n} \{K_i \mid S^*(K_i, K_j) > \theta, \forall j \neq i\}. \tag{9}$$

Théorème 1 (Disjonction des super-catégories)**.** Si $K_1 \in \Xi_K^*$ et $K_2 \in \Xi_K^*$ appartiennent à différentes super-catégories, alors :

$$S^*(K_1, K_2) \leq \theta. \tag{10}$$

Les super-catégories Ξ_K^* permettent de regrouper dynamiquement les entités de connaissances en fonction de leur score de similarité, tout en respectant une hiérarchie naturelle. Cela améliore l'interprétabilité et l'adaptabilité du cadre, en particulier dans des domaines de connaissances dynamiques.

3.6 Extracteur de contradictions et mécanisme de réparation

L'un des défis principaux dans la gestion des bases de connaissances est la présence de contradictions, qui peuvent compromettre la cohérence et la fiabilité des analyses. Pour surmonter ce problème, le cadre proposé inclut un mécanisme de gestion des contradictions, composé d'un extracteur de contradictions E et d'un mécanisme de réparation. Ces outils permettent de détecter, résoudre et gérer les incohérences tout en préservant la structure hiérarchique des connaissances.

3.6.1 Définition de l'extracteur de contradictions

L'extracteur de contradictions E est un opérateur logique qui identifie les propriétés contradictoires présentes dans une entité de connaissance K. Formellement, on définit :

Définition 4 (Extracteur de contradictions)**.** Un extracteur de contradictions est un opérateur logique E tel que :

$$E(K) = \{p_i \in K \mid \exists q_j \in K \text{ tels que } (p_i \wedge \neg q_j) \vee (\neg p_i \wedge q_j)\}. \tag{11}$$

L'extracteur de contradictions E identifie les propriétés p_i dans K qui sont en contradiction avec d'autres propriétés q_j de la même entité. Cela permet de localiser les sources de conflits dans les bases de connaissances. Par exemple, si une entité K contient à la fois p et $\neg p$, alors p sera identifié comme contradictoire par E.

3.6.2 Entités réparables

Une entité de connaissance peut être réparée si certaines contradictions peuvent être résolues sans affecter les autres propriétés. Cette notion est formalisée comme suit :

Définition 5 (Entité réparable)**.** Une entité K est réparable si $\exists R \subseteq E(K)$ tel que :

$$\neg\text{Contradictoire}(K \setminus R). \tag{12}$$

Une entité K est réparable si elle contient un sous-ensemble R de contradictions qui, une fois supprimé ou résolu, permet de rendre K cohérente. Cela garantit que les bases de connaissances peuvent être maintenues cohérentes, même en présence de conflits. Par exemple, si une entité K contient une contradiction $(p, \neg p)$ mais que la suppression de $\neg p$ suffit à rendre K cohérente, alors K est réparable.

3.6.3 Mécanisme de réparation minimale

Le mécanisme de réparation vise à identifier le sous-ensemble minimal de contradictions à résoudre pour rendre une entité cohérente. Cela est formalisé par le corollaire suivant :

Corollaire 1.1 (Réparation minimale)**.** La réparation minimale $R_{\min}$ est définie comme :

$$R_{\min} = \arg \min_{R \subseteq E(K)} (\|R\| \mid \neg\text{Contradictoire}(K \setminus R)). \tag{13}$$

La réparation minimale $R_{\min}$ est le sous-ensemble de contradictions le plus petit qui, une fois résolu, permet de rendre K cohérente. Cela garantit que le nombre de modifications apportées à K est minimisé, ce qui est essentiel pour préserver l'intégrité des connaissances. Par exemple, si une entité K contient plusieurs contradictions, la réparation minimale $R_{\min}$ consiste à résoudre uniquement celles qui sont nécessaires pour rendre K cohérente.

3.6.4 Impact sur la mesure de similarité paraconsistante

L'introduction de l'extracteur de contradictions et du mécanisme de réparation permet de définir une version améliorée de la mesure de similarité paraconsistante, qui tient compte des réparations effectuées :

Définition 6 (Similarité paraconsistante avec réparations)**.** Soient K_1 et K_2 deux entités de connaissances. La similarité paraconsistante avec réparations $\Xi_{RP}(K_1, K_2)$ est définie comme :

$$\Xi_{RP}(K_1, K_2) = \Xi_P(K_1', K_2'), \tag{14}$$

où K_1' et K_2' sont les versions réparées de K_1 et K_2, respectivement, obtenues en résolvant les contradictions à l'aide de E.

La similarité paraconsistante avec réparations Ξ_{RP} permet de comparer les entités de connaissances après que les contradictions ont été résolues. Cela garantit une évaluation plus précise et cohérente de la similarité, même en présence de conflits. Par exemple, si K_1 et K_2 contiennent des contradictions, la similarité $\Xi_{RP}(K_1, K_2)$ est calculée sur les versions réparées K_1' et K_2', ce qui réduit l'impact des contradictions sur la mesure de similarité.

3.6.5 Propriétés théoriques

Le cadre proposé garantit plusieurs propriétés théoriques essentielles pour assurer la cohérence et la robustesse des évaluations de similarité, même en présence de contradictions.

Proposition 4 (Existence de l'extracteur de contradictions)**.** Pour toute entité de connaissance K, il existe un extracteur de contradictions $E(K)$.

Démonstration. D'après la définition 4, $E(K)$ est construit en identifiant des paires de propriétés p_i et q_j telles que $p_i \wedge \neg q_j$ ou $\neg p_i \wedge q_j$. Puisque la logique du premier ordre (FOL) garantit que les propriétés peuvent être exprimées logiquement, $E(K)$ est bien défini. □

Proposition 5 (Réparabilité d'une entité)**.** Une entité de connaissance K est réparable si et seulement si $E(K) \neq K$.

Démonstration. Si $E(K) = K$, alors chaque propriété de K est impliquée dans une contradiction, rendant K irréparable. Inversement, si $E(K) \neq K$, il existe au moins une propriété non impliquée dans une contradiction, permettant l'existence d'un sous-ensemble cohérent $K \setminus R$. □

Théorème 2 (Préservation des catégories de similarité)**.** La similarité paraconsistante avec réparations $\Xi_{RP}(K_1, K_2)$ préserve les catégories de similarité originales $K' \approx, K' \neq$ après réparation de K_1 et K_2.

Démonstration. D'après la définition 6, $\Xi_{RP}(K_1, K_2)$ est calculée sur les versions réparées K_1' et K_2'. Puisque les réparations résolvent les contradictions sans altérer les propriétés partagées, les catégories de similarité restent cohérentes avec l'original $\Xi_P(K_1, K_2)$. □

Théorème 3 (Cohérence après réparation)**.** Si S est une super-catégorie dans Ξ_K, alors la réparation de S assure la cohérence dans toutes les sous-catégories.

Démonstration. D'après la définition 6, la réparation de S implique la résolution des contradictions dans ses propriétés constitutives. Puisque les sous-catégories héritent des propriétés de S, la réparation de S assure la cohérence à tous les niveaux de la hiérarchie. □

Corollaire 3.1 (Réparation minimale)**.** La réparation minimale $R_{\min}$ satisfait :

$$R_{\min} = \arg \min_{R \subseteq E(K)} \left(\|R\| \mid \neg \text{Contradictoire}(K \setminus R) \right). \tag{15}$$

Démonstration. D'après la proposition 5, $R_{\min}$ est le plus petit sous-ensemble de $E(K)$ qui résout toutes les contradictions, assurant la réparabilité. □

Corollaire 3.2 (Preservation de la structure hiérarchique)**.** La réparation d'une super-catégorie S n'affecte pas la structure hiérarchique de Ξ_K.

Démonstration. D'après le théorème 3, la réparation de S assure la cohérence dans les sous-catégories, préservant l'organisation hiérarchique de Ξ_K. □

3.7 Exemples

Nous illustrons ici le fonctionnement du cadre avec deux exemples concrets :

Exemple 1 (Contradictions simples)**.** Soient :

$$K_1 = \{p_1, p_2, \neg p_3\}, \quad K_2 = \{p_2, p_3, \neg p_1\}. \tag{16}$$

Alors :

$$\begin{aligned} P_{\text{shared}} &= \{p_2\}, \\ P_{\text{contradictory}} &= \{p_1, p_3\}, \\ P_{\text{total}} &= \{p_1, p_2, p_3, \neg p_1, \neg p_3\}. \end{aligned}$$

Donc :

$$S^*(K_1, K_2) = \frac{1}{5} - \frac{2}{5} = -\frac{1}{5}.$$

Cela montre que malgré une propriété partagée, les contradictions entraînent une valeur de similarité négative.

Dans cet exemple, les entités K_1 et K_2 partagent une seule propriété p_2, mais présentent deux contradictions $(p_1, \neg p_1)$ et $(p_3, \neg p_3)$. La mesure S^* prend en compte ces contradictions, ce qui entraîne une valeur négative. Cela illustre comment les contradictions peuvent réduire significativement la similarité entre deux entités, reflétant ainsi la complexité des connaissances réelles.

Exemple 2 (Regroupement hiérarchique)**.** Considérons un ensemble de cinq entités de connaissances $\mathcal{K} = \{K_1, K_2, K_3, K_4, K_5\}$, chacune représentant une base de connaissances concernant des diagnostics médicaux. Les entités sont définies comme suit :

$K_1 = \{\text{fièvre}, \text{toux}, \neg\text{maux de tête}\},$
$K_2 = \{\text{fièvre}, \neg\text{toux}, \text{maux de tête}\},$
$K_3 = \{\text{fièvre}, \text{toux}, \text{fatigue}\},$
$K_4 = \{\text{essoufflement}, \text{vomissements}, \neg\text{fièvre}\},$
$K_5 = \{\text{essoufflement}, \neg\text{vomissements}, \text{douleur abdominale}\}.$

Étape 1 : Calcul des mesures de similarité paraconsistante S^*

Nous appliquons la mesure de similarité paraconsistante S^* entre chaque paire d'entités, définie par l'Équation 3. Voici les résultats calculés :

$$
\begin{aligned}
S^*(K_1, K_2) &= \frac{1-2}{5} = -0.2, \\
S^*(K_1, K_3) &= \frac{2-0}{4} = 0.5, \\
S^*(K_1, K_4) &= \frac{0-1}{6} = -0.17, \\
S^*(K_1, K_5) &= \frac{0-0}{5} = 0, \\
S^*(K_2, K_3) &= \frac{1-1}{5} = 0, \\
S^*(K_2, K_4) &= \frac{0-1}{6} = -0.17, \\
S^*(K_2, K_5) &= \frac{0-0}{5} = 0, \\
S^*(K_3, K_4) &= \frac{0-1}{6} = -0.17, \\
S^*(K_3, K_5) &= \frac{0-0}{5} = 0, \\
S^*(K_4, K_5) &= \frac{1-1}{5} = 0.
\end{aligned}
$$

Étape 2 : Définition du seuil de similarité θ

Fixons un seuil $\theta = 0.4$. Selon la définition des super-catégories paraconsistantes (cf. section 3.5), deux entités appartiennent à la même super-catégorie si leur similarité dépasse ce seuil.

Étape 3 : Formation des super-catégories paraconsistantes Ξ_K^*

À partir des valeurs ci-dessus, nous identifions les paires satisfaisant $S^*(K_i, K_j) > \theta$:

— $S^*(K_1, K_3) = 0.5 > 0.4$, donc K_1 et K_3 appartiennent à la même super-catégorie.
— Toutes les autres paires ont $S^* \leq 0.4$, donc restent isolées.

Ainsi, les super-catégories paraconsistantes sont :

$$\Xi_K^* = \{\{K_1, K_3\}, \{K_2\}, \{K_4\}, \{K_5\}\}. \tag{17}$$

Étape 4 : Gestion des contradictions via l'extracteur E

Examinons K_1 et K_2, dont la similarité est négative ($S^* = -0.2$). Appliquons l'extracteur de contradictions E sur ces deux entités :

$$E(K_1) = \emptyset, \quad E(K_2) = \{\text{toux}, \neg\text{toux}\}. \tag{18}$$

Supprimons $\neg$toux de K_2 pour obtenir une version réparée $K_2' = \{\text{fièvre}, \text{maux de tête}\}$, puis recalculons la similarité :

$$S^*(K_1, K_2') = \frac{1-0}{4} = 0.25 < \theta. \tag{19}$$

Même après réparation, K_1 et K_2' ne dépassent pas le seuil de regroupement.

Étape 5 : Hiérarchie finale et interprétation

La structure hiérarchique finale est donc :

$$\Xi_K^* = \{\{K_1, K_3\}, \{K_2\}, \{K_4\}, \{K_5\}\}. \tag{20}$$

L'interprétation clinique est claire :

— $\{K_1, K_3\}$ représente des patients présentant des symptômes similaires liés aux voies respiratoires supérieures.
— K_2, malgré sa proximité sémantique, reste isolé en raison des contradictions persistantes.
— $\{K_4, K_5\}$, bien que partageant une propriété commune (essoufflement), sont séparés car leurs différences symptomatiques dominent.

Cet exemple illustre comment le cadre paraconsistant permet de structurer des bases de connaissances complexes en tenant compte non seulement des similitudes, mais aussi des contradictions. La mesure S^*, couplée à l'extracteur de contradictions E, garantit une évaluation robuste et interprétable, même en présence de conflits logiques.

4 Discussion

Le cadre proposé pour l'évaluation de la similarité dans les bases de connaissances représente une avancée significative par rapport aux méthodes classiques. En intégrant explicitement les contradictions via la logique paraconsistante, il permet une évaluation plus précise, interprétable et robuste de la similarité entre entités de connaissances. Cette section analyse les contributions principales du cadre, compare sa mesure S^* aux approches traditionnelles, souligne ses avantages théoriques et pratiques, et identifie des perspectives d'amélioration future.

4.1 Contributions principales

Le cadre introduit une mesure de similarité paraconsistante S^*, définie par l'Équation 3 :

$$S^*(K_1, K_2) = S^+(K_1, K_2) - D^{\pm}(K_1, K_2),$$

Cette mesure étend les mesures classiques en intégrant les contradictions de manière explicite, ce qui permet de pénaliser les divergences tout en valorisant les similitudes.
En outre, le cadre propose des **super-catégories paraconsistantes** Ξ_K^*, définies par l'Équation 9 :

$$\Xi_K^* = \bigcup_{i=1}^{n} \{K_i \mid S^*(K_i, K_j) > \theta, \forall j \neq i\}.$$

Ces super-catégories organisent hiérarchiquement les entités de connaissances selon leur niveau de similarité, améliorant ainsi l'interprétabilité et l'adaptabilité du modèle.
Un mécanisme central du cadre est également l'extracteur de contradictions E, défini par l'Équation 11 :

$$E(K) = \{p_i \in K \mid \exists q_j \in K \text{ tels que } (p_i \wedge \neg q_j) \vee (\neg p_i \wedge q_j)\}.$$

L'extracteur E permet d'identifier les incohérences locales, facilitant ainsi la gestion des conflits sans compromettre la cohérence globale du système.

4.2 Avantages par rapport aux approches existantes

Le cadre proposé surmonte plusieurs limitations des méthodes classiques :

- **Prise en compte des contradictions** : Contrairement aux méthodes classiques qui ignorent ou neutralisent les contradictions, le cadre paraconsistant les intègre dans l'évaluation, ce qui permet une mesure de similarité plus fidèle à la réalité des données conflictuelles.
- **Flexibilité hiérarchique** : Les super-catégories paraconsistantes offrent une organisation dynamique basée sur un seuil variable θ, contrairement aux classifications rigides des approches traditionnelles.
- **Transparence accrue** : Le cadre repose sur des principes logiques explicites, ce qui le distingue des modèles boîte noire comme ceux basés sur l'apprentissage profond. Cela est crucial dans des domaines critiques comme la médecine ou le droit.

Comparaison avec des mesures classiques

Pour illustrer la valeur ajoutée de la mesure de similarité paraconsistante S^* par rapport aux approches traditionnelles, nous présentons ici une comparaison explicite avec la mesure de Jaccard, souvent utilisée pour quantifier la similarité entre ensembles.

Définition 7 (Similarité de Jaccard)**.** Étant donnés deux ensembles finis A et B, la similarité de Jaccard est donnée par :

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}.$$

Bien que cette mesure soit robuste et intuitive dans les cas non contradictoires, elle ne prend pas en compte les propriétés conflictuelles, ce qui peut mener à des évaluations erronées dans des contextes où les contradictions sont présentes.

Exemple : S^* vs. Jaccard

Considérons deux entités de connaissances K_1 et K_2, définies comme suit :

$$K_1 = \{p_1, p_2, \neg p_3\}, \quad K_2 = \{p_2, p_3, \neg p_1\}.$$

Les ensembles de propriétés associés sont :

- $P_{\text{shared}} = \{p_2\}$,
- $P_{\text{contradictory}} = \{p_1, p_3\}$,
- $P_{\text{total}} = \{p_1, p_2, p_3, \neg p_1, \neg p_3\}$.

Calculons à la fois la mesure paraconsistante S^* et la mesure de Jaccard :

- Mesure paraconsistante S^* :

$$S^*(K_1, K_2) = \frac{|P_{\text{shared}}| - |P_{\text{contradictory}}|}{|P_{\text{total}}|} = \frac{1-2}{5} = -\frac{1}{5} = -0.2.$$

- Mesure de Jaccard : En considérant uniquement les littéraux positifs (approche standard), on a :

$$A = \{p_1, p_2\}, \quad B = \{p_2, p_3\},$$

donc :

$$J(A, B) = \frac{|\{p_2\}|}{|\{p_1, p_2, p_3\}|} = \frac{1}{3} \approx 0.33.$$

Dans cet exemple, la mesure de Jaccard attribue une valeur positive à la similarité (~ 0.33), suggérant une certaine proximité entre les deux entités. Cependant, cette évaluation ignore complètement les contradictions présentes entre p_1 et $\neg p_1$, ainsi que p_3 et $\neg p_3$, ce qui peut conduire à une surestimation de la similarité dans des contextes conflictuels. En revanche, la mesure S^* intègre explicitement ces contradictions, réduisant la similarité globale et produisant une valeur négative (-0.2). Ce résultat reflète plus fidèlement la réalité : malgré une propriété partagée, les incohérences dominent, rendant la similarité nettement problématique.
L'exemple présenté précédemment illustre clairement l'avantage théorique et pratique de la mesure S^* par rapport aux mesures classiques comme celle de Jaccard. En intégrant les contradictions dans l'évaluation, S^* offre une vision nuancée, interprétable et robuste de la similarité entre entités de connaissances, particulièrement utile dans des domaines tels que le droit, la médecine ou les systèmes multi-agents où la gestion des conflits est cruciale.
Cette capacité à distinguer les similitudes réelles des faux positifs dus à des incohérences constitue une contribution essentielle du cadre paraconsistant proposé.

4.3 Perspectives futures

Malgré les avantages de ce cadre, il présente des défis qu'il conviendrait d'explorer dans le cadre de recherches futures :

- **Complexité computationnelle** : L'identification des contradictions et le calcul de S^* peuvent devenir coûteux dans de grandes bases de connaissances. Des algorithmes optimisés ou des techniques d'approximation pourraient être développés.
- **Adaptation aux bases de connaissances évolutives** : Une extension du cadre pourrait inclure des mécanismes pour gérer l'évolution dynamique des connaissances, par exemple en intégrant des mises à jour incrémentielles ou des techniques d'apprentissage en ligne.
- **Automatisation de la sélection des propriétés** : Actuellement, la sélection des propriétés est manuelle. Des méthodes automatisées basées sur l'apprentissage supervisé ou non supervisé pourraient améliorer l'objectivité et la pertinence des comparaisons.
- **Validation empirique** : Bien que des exemples illustratifs aient été fournis, une évaluation sur des ensembles de données réels restent à réaliser pour confirmer l'efficacité du cadre dans des applications pratiques.

Applications dans les systèmes multi-agents

Le cadre paraconsistant proposé peut être appliqué dans les systèmes multi-agents (SMA), notamment pour la résolution de conflits entre agents ayant des représentations de connaissances hétérogènes ou conflictuelles. Par exemple, les agents pourraient utiliser S^* pour détecter et résoudre les incohérences dans leurs bases de connaissances partagées, facilitant ainsi une coopération plus robuste et cohérente.

De plus, les super-catégories paraconsistantes Ξ_K^* pourraient servir à organiser dynamiquement les relations entre agents, en fonction de seuils de similarité θ ajustables. Cette flexibilité est particulièrement précieuse dans des environnements dynamiques comme les réseaux de capteurs distribués ou les systèmes autonomes.

Intégration potentielle dans des formalismes logiques étendus

Une voie prometteuse serait l'intégration du cadre paraconsistant avec d'autres formalismes logiques, tels que :

- **Logique modale** : Pour modéliser la croyance, la possibilité ou la nécessité dans les évaluations de similarité.
- **Logique temporelle** : Pour prendre en compte l'évolution des connaissances dans le temps.
- **Ontologies et graphes de connaissances** : Pour renforcer l'interopérabilité avec les standards actuels de représentation des connaissances.

Ces extensions pourraient enrichir les capacités du cadre tout en conservant son fondement paraconsistant.

5 Conclusion

Le présent travail a proposé un **cadre paraconsistant pour l'évaluation de la similarité dans les bases de connaissances**, visant à surmonter les limitations des approches classiques. Ce cadre s'appuie sur une **logique paraconsistante**, permettant de gérer efficacement les contradictions tout en préservant la cohérence et l'interprétabilité des évaluations de similarité.

Une **mesure de similarité paraconsistante** S^* a été introduite, qui intègre explicitement les propriétés contradictoires, offrant ainsi une évaluation plus robuste et nuancée que les mesures traditionnelles. Cette mesure a été formalisée en tenant compte des propriétés partagées et contradictoires, assurant une réflexivité et une symétrie, des propriétés essentielles pour une application pratique.

Un **ensemble de super-catégories paraconsistantes** Ξ_K^* a également été défini, permettant d'organiser hiérarchiquement les entités de connaissances selon leurs scores de similarité. Cette structure a été validée par des exemples illustratifs dans des domaines tels que la médecine et le droit, montrant sa pertinence pratique.

Un **extracteur de contradictions** E a été intégré au cadre, permettant d'identifier et de résoudre les incohérences locales, tout en préservant la cohérence globale des bases de connaissances. Des garanties théoriques ont été fournies, notamment sur la réparabilité des entités et la préservation de la structure hiérarchique après réparation.

L'analyse comparative avec des méthodes classiques, telles que la mesure de Jaccard, a permis de mettre en évidence les avantages du cadre paraconsistant : une gestion explicite des contradictions, une flexibilité accrue dans l'organisation des connaissances, et une meilleure interprétabilité des résultats.

Cependant, plusieurs défis subsistaient, notamment la **complexité computationnelle** lors de l'évaluation à grande échelle, l'**évolution dynamique** des bases de connaissances, et la **sélection des propriétés** pour la comparaison. Ces limites ont été identifiées comme des axes de recherche futurs, ouvrant la voie à des améliorations potentielles, comme l'intégration d'algorithmes optimisés, de mécanismes d'adaptation dynamique, ou de techniques d'apprentissage automatique pour la sélection des propriétés.

Enfin, une **perspective prometteuse** a été soulignée concernant l'intégration de ce cadre dans les **systèmes multi-agents (SMA)**, où la gestion des conflits et la coordination entre agents sont cruciales. Le cadre proposé a démontré son potentiel pour améliorer la robustesse, la transparence et l'adaptabilité des systèmes basés sur la logique paraconsistante.

Remerciements

L'auteur tient à remercier les personnes qui ont soutenu ce travail, ainsi que les relecteurs anonymes pour leurs commentaires constructifs qui ont contribué à l'amélioration de cet article.

Références

[1] Jean-Yves Béziau. Are paraconsistent negations negations ? In *Paraconsistency*, pages 465–486. CRC Press, 2002.

[2] Veronica Biazzo, Angelo Gilio, Thomas Lukasiewicz, and Giuseppe Sanfilippo. Probabilistic logic under coherence, model-theoretic probabilistic logic, and default reasoning in system p. *Journal of Applied Non-Classical Logics*, 12(2) :189–213, 2002.

[3] Vincent D Blondel, Anahí Gajardo, Maureen Heymans, Pierre Senellart, and Paul Van Dooren. A measure of similarity between graph vertices : Applications to synonym extraction and web searching. *SIAM review*, 46(4) :647–666, 2004.

[4] Ronald Brachman and Hector Levesque. *Knowledge representation and reasoning*. Elsevier, 2004.

[5] Ronald J Brachman and Hector J Levesque. The tractability of subsumption in frame-based description languages. In *AAAI*, volume 84, pages 34–37, 1984.

[6] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. Universal sentence encoder. *arXiv preprint arXiv :1803.11175*, 2018.

[7] Huanhuan Cheng and Runsheng Wang. Semantic modeling of natural scenes based on contextual bayesian networks. *Pattern Recognition*, 43(12) :4042–4054, 2010.

[8] Newton CA Da Costa. On the theory of inconsistent formal systems. *Notre dame journal of formal logic*, 15(4) :497–510, 1974.

[9] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert : Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics : human language technologies, volume 1 (long and short papers)*, pages 4171–4186, 2019.

[10] AnHai Doan, Jayant Madhavan, Pedro Domingos, and Alon Halevy. Ontology matching : A machine learning approach. *Handbook on ontologies*, pages 385–403, 2004.

[11] Xinbo Gao, Bing Xiao, Dacheng Tao, and Xuelong Li. A survey of graph edit distance. *Pattern Analysis and applications*, 13 :113–129, 2010.

[12] R.L. Goldstone and J.Y. Son. Similarity. *Oxford University Press*, 2012.

[13] Benjamin N Grosof, Ian Horrocks, Raphael Volz, and Stefan Decker. Description logic programs : Combining logic programs with description logic. In *Proceedings of the 12th international conference on World Wide Web*, pages 48–57, 2003.

[14] Nicola Guarino and Christopher A Welty. An overview of ontoclean. *Handbook on ontologies*, pages 201–220, 2009.

[15] Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57, 1999.

[16] Jaz Kandola, Nello Cristianini, and John Shawe-taylor. Learning semantic similarity. *Advances in neural information processing systems*, 15, 2002.

[17] Leonid Libkin. Incomplete data : what went wrong, and how to fix it. In *Proceedings of the 33rd ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 1–13, 2014.

[18] Eliana Minicozzi. Some natural properties of strong-identification in inductive inference. *Theoretical Computer Science*, 2(3) :345–360, 1976.

[19] Mohammad Norouzi, David J Fleet, and Russ R Salakhutdinov. Hamming distance metric learning. *Advances in neural information processing systems*, 25, 2012.

[20] Judea Pearl. *Probabilistic reasoning in intelligent systems : networks of plausible inference*. Elsevier, 2014.

[21] Graham Priest. *Beyond the limits of thought*. Oxford University Press, 2002.

[22] Philip Resnik. Using information content to evaluate semantic similarity in a taxonomy. *arXiv preprint cmp-lg/9511007*, 1995.

[23] Mikhail Sheremet, Dmitry Tishkovsky, Frank Wolter, and Michael Zakharyaschev. A logic for concepts and similarity. *Journal of Logic and Computation*, 17(3) :415–452, 2007.

[24] Pavel Shvaiko and Jérôme Euzenat. Ontology matching : state of the art and future challenges. *IEEE Transactions on knowledge and data engineering*, 25(1) :158–176, 2011.

[25] John F Sowa. Conceptual graphs. *Foundations of artificial intelligence*, 3 :213–237, 2008.

[26] Peter D Turney and Patrick Pantel. From frequency to meaning : Vector space models of semantics. *Journal of artificial intelligence research*, 37 :141–188, 2010.

[27] Amos Tversky. Features of similarity. *Psychological review*, 84(4) :327, 1977.

[28] José-Luis Vilchis-Medina. Building intelligent databases through similarity : Interaction of logical and qualitative reasoning. In *2024 7th International Conference on Algorithms, Computing and Artificial Intelligence (ACAI)*, pages 1–5, 2024.

[29] Yaoshu Wang, Jianbin Qin, and Wei Wang. Efficient approximate entity matching using jaro-winkler distance. In *International conference on web information systems engineering*, pages 231–239. Springer, 2017.

[30] Li Yujian and Liu Bo. A normalized levenshtein distance metric. *IEEE transactions on pattern analysis and machine intelligence*, 29(6) :1091–1095, 2007.

Session 2 : Logique possibiliste

40 ans de recherche en logique possibiliste - Une vue d'ensemble

Didier Dubois[1], Henri Prade[1]

[1] IRIT, CNRS & Université Toulouse III - Paul Sabatier, 118 route de Narbonne, 31062 Toulouse cedex 9, France

{didier.dubois, henri.prade}@irit.fr

Résumé

Les premiers éléments de logique possibiliste datent de 40 ans. Cette logique manipule des formules logiques classiques associées à des pondérations prenant des valeurs dans un ensemble linéairement ordonné ou plus généralement dans un treillis. Au cours des décennies, la logique possibiliste a connu de nombreux développements tant au niveau théorique qu'au niveau appliqué. L'ambition de cet article est de passer en revue ces développements tout en exposant les idées principales qui les sous-tendent.

Mots-clés

théorie des possibilités, logique, incertitude, préférence.

Abstract

The first elements of possibilistic logic date back 40 years. This logic handles classical logic formulas associated with weights taking values in a linearly ordered set or more generally in a lattice. Over the decades, possibilistic logic has undergone numerous developments at both theoretical and applied levels. The goal of this article is to review all these developments while exposing the main ideas behind them.

Keywords

possibility theory, classical logic, uncertainty, preference.

1 Introduction

La logique possibiliste est issue de la théorie des possibilités. Cette théorie offre un cadre pour la représentation de l'incertitude épistémique due à une information incomplète. Cette théorie a été d'abord proposée par un économiste, G. L. S. Shackle [80], qui a introduit un calcul de degrés de surprise potentielle (qui sont des degrés d'impossibilité) ; elle a été redécouverte indépendamment par L. A. Zadeh [84] qui s'est concentré sur l'idée de possibilité graduelle en relation avec la modélisation de l'information linguistique, et finalement développée dans [45] en utilisant conjointement la double paire de mesures de possibilité et de nécessité associées à une distribution de possibilité.

La logique possibiliste [41] (dans sa forme de base) manipule des formules logiques classiques associées à des bornes inférieures de mesures de nécessité entendues comme des niveaux de certitude. La règle du modus ponens prend alors, sémantiquement, la forme :

$$N(p) \geq \alpha, N(p \rightarrow q) \geq \beta \Rightarrow N(q) \geq \min(\alpha, \beta),$$

où N est une mesure de nécessité, p et q sont des formules logiques, et $\alpha, \beta \in [0, 1]$. Cela correspond à l'intuition (remontant à Théophraste [77]) selon laquelle la force d'une conclusion reflète la force de la prémisse la plus faible. Cette règle d'inférence pondérée apparaît pour la première fois en 1982. [1] Cependant, ce n'est qu'au milieu des années 1980 que les premiers éléments d'une logique possibiliste à part entière ont commencé à être développés [59, 39].

Les informations incomplètes sont présentes partout et il est important de gérer correctement l'incertitude épistémique. Comme nous le verrons, la logique possibiliste, en stratifiant la connaissance en niveaux de certitude, offre un cadre simple, proche de la logique classique, pour traiter l'incertitude et l'incohérence, mais la logique possibiliste peut aussi prendre d'autres formes, comme les réseaux ou les matrices possibilistes. De plus, la logique possibiliste hérite sa polyvalence de la puissance de représentation de la théorie des possibilités.

Cet article propose une vue d'ensemble - aussi complète que possible en 10 pages - des travaux sur la logique possibiliste depuis 40 ans. Il y a déjà eu plusieurs synthèses sur le sujet depuis [47] qui sont maintenant en partie dépassées. Certaines se concentrent sur les relations avec la logique modale [51], d'autres offrent une perspective plus appliquée [52]. En outre, il existe aussi des introductions plus longues et plus détaillées (mais désormais incomplètes) [41, 48]. Le présent article, avec une structure renouvelée, offre un nouveau regard sur la logique possibiliste.

L'exposé est organisé en deux parties principales. La première partie présente les principaux aspects théoriques et insiste sur les questions de représentation. La seconde partie passe en revue une série de domaines de recherche en IA auxquels la logique possibiliste a été appliquée et peut encore contribuer. Plus précisément, la première partie, après un rappel sur les mesures de possibilité et de nécessité, présente la syntaxe, la sémantique et la théorie de la preuve de la logique possibiliste de base où seules des contraintes de la forme $N(p) \geq \alpha$ sont traitées. Puis, les principales caractéristiques du calcul matriciel possibiliste et des réseaux possibilistes (de type bayésien) sont présentées. Ensuite, divers types d'extensions de la logique possi-

1. [Prade, Thèse d'Etat, 1982] ; [Prade, IJCAI'1983,130-136] (éq. 56).

biliste sont passés en revue : i) pour traiter l'incohérence ; ii) pour prendre en compte des niveaux de certitude symboliques (dont la valeur précise reste inconnue) ; iii) pour introduire de nouveaux types de poids afin de traiter le temps, le multi-sources, les agents, les raisons argumentatives, ou les niveaux de certitude mal connus, grâce à l'utilisation de fonctions de possibilité et de nécessité généralisées prenant leurs valeurs sur un treillis booléen ou sur un treillis distributif pseudo-complémenté plutôt que sur une échelle linéaire ; iv) pour faire face à l'information bipolaire (c'est-à-dire ayant des composantes positives et négatives) grâce à la notion de possibilité garantie, une autre fonction d'ensemble de la théorie des possibilités ; v) pour gérer non seulement les conjonctions, mais aussi les négations et les disjonctions des contraintes présentes en logique possibiliste de base. La première partie se termine par une brève discussion sur le lien avec deux calculs proches : les fonctions de classement de Spohn et la logique de Markov. La deuxième partie passe en revue l'utilisation de la logique possibiliste dans le raisonnement par défaut, la révision des croyances, la fusion d'informations, les logiques de description, la programmation logique, la modélisation des préférences et la décision, l'argumentation et l'apprentissage automatique. Une sous-section est également consacrée à des applications en base de données, en raison de leurs liens étroits avec la représentation des connaissances.

2 Questions théoriques et représentationnelles

Cette première partie traite des bases de la logique possibiliste et des cadres de représentation associés, avant de présenter diverses extensions de la logique possibiliste, et enfin de discuter des relations avec d'autres cadres.

2.1 Théorie des possibilités

En théorie des possibilités, les informations disponibles sont représentées par des distributions de possibilité. Une distribution de possibilité est une application π d'un ensemble U, compris comme un ensemble d'états, de valeurs ou d'alternatives, mutuellement exclusifs (dont l'un est le monde réel, si U est exhaustif), dans une échelle totalement ordonnée $\mathcal{S}$ bornée, dont le plus grand élément est noté 1 et le plus petit 0. Différents types d'échelles peuvent être utilisés depuis une échelle finie $\mathcal{S} = \{1 = \lambda_1 > \dots \lambda_n > \lambda_{n+1} = 0\}$ dans le cas qualitatif, jusqu'à l'intervalle unitaire $\mathcal{S} = [0, 1]$ dans le cas quantitatif, voir [50] pour d'autres options. $\pi(u) = 0$ signifie que u est rejeté comme impossible ; $\pi(u) = 1$ signifie que l'état u est entièrement possible. Plus $\pi(u)$ est grand, plus u est possible. La cohérence de l'état épistémique décrit par π est exprimée par la condition de normalisation $\exists u, \pi(u) = 1$ qui garantit qu'au moins un u est entièrement possible. Si l'information est sans incertitude, mais peut-être imprécise, π est la fonction caractéristique d'un sous-ensemble E de U et $\pi(u) \in \{0, 1\}$. L'information complète correspond aux situations où E est un singleton. L'échelle $\mathcal{S}$ est supposée équipée d'une bijection inversant l'ordre $\lambda \in \mathcal{S} \mapsto 1 - \lambda \in \mathcal{S}$.

Une mesure de possibilité Π et une mesure de nécessité duale N sont associées à une distribution de possibilité π : $\forall A \subseteq U$,
$\Pi(A) = \sup_{u \in A} \pi(u)$; $N(A) = 1 - \Pi(A^c) = \inf_{u \notin A} 1 - \pi(u)$ avec $A^c = U \setminus A$. Lorsque la distribution de possibilité se réduit à un sous-ensemble classique $E \subseteq U$, on a : i) $\Pi(A) = 1$ si $A \cap E \neq \emptyset$, et 0 sinon ; ii) $N(A) = 1$ si $E \subseteq A$, et 0 sinon. $\Pi(A)$ (resp. $N(A)$) évalue à quel point l'événement A est cohérent avec π (resp. est impliqué par π). Par normalisation, $\Pi(U) = N(U) = 1$ et $\Pi(\emptyset) = N(\emptyset) = 0$.
Les mesures de possibilité sont caractérisées par la propriété de « maxitivité » $\Pi(A \cup B) = \max(\Pi(A), \Pi(B))$, et les mesures de nécessité sont « minitives » : $N(A \cap B) = \min(N(A), N(B))$. En raison de la normalisation de π, $\min(N(A), N(A^c)) = 0$ et $\max(\Pi(A), \Pi(A^c)) = 1$, ou de manière équivalente $\Pi(A) = 1$ lorsque $N(A) > 0$, à savoir que quelque chose de quelque peu certain doit être entièrement possible, c'est-à-dire cohérent avec l'information disponible. De plus, on ne peut pas être quelque peu certain à la fois de A et de A^c, sans être incohérent. Nous n'avons que $N(A \cup B) \geq \max(N(A), N(B))$, ce qui va bien avec l'idée qu'on peut être certain de l'événement $A \cup B$, sans être vraiment certain d'événements plus spécifiques comme A ou comme B. La possibilité et la nécessité se différencient d'une probabilité P, qui est auto-duale, et telle que $P(A^c) = 0 \Rightarrow P(A) = 1$, tandis que $N(A^c) = 0 \not\Rightarrow N(A) = 1$ (mais $\Pi(A^c) = 0 \Rightarrow \Pi(A) = 1$).
Les énoncés qualifiés en termes de certitude de la forme « A est certain au degré α » sont représentés par la contrainte $N(A) \geq \alpha$. La plus grande distribution de possibilité π, donc la moins restrictive, qui obéit à cette contrainte est donnée par [45] : $\pi_{(A,\alpha)}(u) = 1$ si $u \in A, \pi_{(A,\alpha)}(u) = 1 - \alpha$ sinon. Si $\alpha = 1$ on obtient la fonction caractéristique de A. Si $\alpha = 0$, on obtient l'ignorance totale. C'est un élément clé de la sémantique de la logique possibiliste.

2.2 Logique possibiliste de base

Une formule de logique possibiliste de base (LPB en abrégé) est une paire (p, α) où p est une formule en logique classique et α un niveau de certitude dans $\mathcal{S} \setminus \{0\}$, considéré comme une borne inférieure d'une mesure de nécessité : (p, α) signifie sémantiquement $N(p) \geq \alpha$. En raison de la minitivité des mesures de nécessité, une base en LPB, c'est-à-dire un ensemble de formules de LPB, peut être mise sous une forme clausale équivalente.
Aspects syntaxiques Nous nous intéressons ici au cas où p dans (p, α) est une proposition ; pour la logique possibiliste du *premier ordre* (de base), voir [41].
Axiomes et règles d'inférence. Les axiomes de LPB [41] sont ceux de la logique propositionnelle, où chaque schéma d'axiomes a une certitude 1. Ses règles d'inférence sont :
- si $\beta \leq \alpha$ alors $(p, \alpha) \vdash (p, \beta)$ (diminution de la certitude)
- $(\neg p \vee q, \alpha), (p, \alpha) \vdash (q, \alpha), \forall \alpha \in (0, 1]$ (modus ponens).
De plus, la règle d'inférence suivante est valide :
- $(\neg p \vee q, \alpha), (p \vee r, \beta) \vdash (q \vee r, \min(\alpha, \beta))$ (résolution)
La règle d'inférence suivante d'*affaiblissement de la formule*, est aussi valide, en conséquence de la α-β-résolution :
- si $p \vdash q$ alors $(p, \alpha) \vdash (q, \alpha), \ \forall \alpha \in (0, 1]$.

Inférence et cohérence. Soit $K = \{(p_i, \alpha_i), i = 1, ..., m\}$ un ensemble de formules en LPB. Prouver $K \vdash (p, \alpha)$ revient alors à prouver $K, (\neg p, 1) \vdash (\bot, \alpha)$ par application répétée de la règle de résolution. De plus, notons que $K \vdash (p, \alpha)$ ssi $K_\alpha \vdash (p, \alpha)$ ssi $(K_\alpha)^* \vdash p$, où $K_\alpha = \{(p_i, \alpha_i) \in K, \alpha_i \geq \alpha\}$ et $K^* = \{p_i \mid (p_i, \alpha_i) \in K\}$. Les niveaux de certitude stratifient la base de connaissances K en coupes de niveaux imbriquées K_α, c'est-à-dire $K_\alpha \subseteq K_\beta$ si $\beta \leq \alpha$. Une conséquence (p, α) de K ne peut être obtenue qu'à partir de formules dans K_α.
Le *niveau d'incohérence* de K est défini par $inc(K) = \max\{\alpha | K \vdash (\bot, \alpha)\}$. Les formules (p_i, α_i) dans K telles que $\alpha_i > inc(K)$ sont à l'abri de l'incohérence. En effet, si $\alpha > inc(K)$, $(K_\alpha)^*$ est cohérent, et K^* cohérent $\Leftrightarrow$ $inc(K) = 0$.
La complexité de l'inférence en LPB reste similaire à celle de la logique classique [68].
Aspects sémantiques. La sémantique du LPB [41] s'exprime en termes de distributions de possibilité, et de mesures de nécessité sur l'ensemble Ω des interprétations ω du langage. La base K est sémantiquement associée à la distribution de possibilité, qui est un ensemble *flou* d'interprétations :

$$\pi_K(\omega) = \min_{i=1}^{m} \max([p_i](\omega), 1 - \alpha_i)$$

où $[p_i]$ est la fonction caractéristique des modèles de p_i, à savoir $[p_i](\omega) = 1$ si $\omega \vDash p_i$ et $[p_i](\omega) = 0$ sinon. Ceci est en accord avec la qualification en termes de certitude : Intuitivement, cela signifie que toute interprétation qui est un contre-modèle de p_i, est d'autant moins possible que p_i est plus certain ; π_K est obtenu comme la conjonction basée sur min des distributions de possibilité représentant chaque formule. Comme attendu, $N_K(p_i) \geq \alpha_i$ pour i=1,...,m, où N_K est défini à partir de π_K. L'implication sémantique est définie par $K \vDash (p, \alpha)$ ssi $\forall \omega, \pi_K(\omega) \leq \pi_{\{(p,\alpha)\}}(\omega)$. LPB est sain et complet [41] par rapport à cette sémantique :

$$K \vdash (p, \alpha) \text{ ssi } K \vDash (p, \alpha).$$

De plus, nous avons $inc(K) = 1 - \max_{\omega \in \Omega} \pi_K(\omega)$, qui reconnaît le fait que la normalisation de π_K est équivalente à la cohérence classique de K^*.
En probabilités, la seule utilisation de la règle de résolution (localement optimale) $Prob(\neg p \vee q) \geq \alpha, Prob(p \vee r) \geq \beta \vdash Prob(q \vee r) \geq \max(0, \alpha + \beta - 1)$, ne peut pas assurer la complétude d'une contrepartie probabiliste de LPB.

2.3 Forme matricielle

Une règle « si p alors q » est représentée plus naturellement en termes de conditionnement plutôt qu'en utilisant l'implication matérielle de la logique qui permet la contraposition. Le conditionnement en théorie des possibilités obéit à :

$$\Pi(p \wedge q) = \Pi(q \mid p) \star \Pi(p)$$

où $\star$ est min ou le produit, selon que l'on choisit d'être dans un cadre qualitatif ou quantitatif.[2] Pour $\star = \min$, la solution la plus grande et la moins restrictive de l'équation ci-dessus est $\Pi(q \mid p) = \Pi(p \wedge q)$ si $\Pi(p \wedge q) < \Pi(p)$, $\Pi(q \mid p) = 1$ sinon. Pour $\star =$ produit, le conditionnement (quantitatif) ressemble à un conditionnement probabiliste : $\Pi(q \mid p) = \frac{\Pi(p \wedge q)}{\Pi(p)}$ pour $\Pi(p) \neq 0$ et correspond à la règle de conditionnement de Dempster dans la théorie de Shafer [81]. La nécessité conditionnelle est définie par dualité : $N(q|p) = 1 - \Pi(\neg q|p)$.
En utilisant la possibilité conditionnelle qualitative, un calcul matriciel (voir [44][53] pour des études approfondies) peut être développé en utilisant le produit matriciel max-min $\otimes$ (en notant que $\Pi(q) = \max(\Pi(p \wedge q), \Pi(\neg p \wedge q))$:

$$\begin{bmatrix} \Pi(q) \\ \Pi(\neg q) \end{bmatrix} = \begin{bmatrix} \Pi(q|p) & \Pi(q|\neg p) \\ \Pi(\neg q|p) & \Pi(\neg q|\neg p) \end{bmatrix} \otimes \begin{bmatrix} \Pi(p) \\ \Pi(\neg p) \end{bmatrix}.$$

$\otimes$ préserve la normalisation : $\forall r, \max(\Pi(r), \Pi(\neg r)) = 1$.
Un tel produit matriciel peut être appliqué à un ensemble de m règles incertaines en parallèle de la forme "si $a_i^1(x)$ est P_i^1 et $\cdots$ et $a_i^k(x)$ est P_i^k alors $b_i(x)$ est Q_i" ($i = 1, \cdots, m$) qui relie des variables appartenant aux valeurs d'attributs d'un élément x, et où les P_i^j et Q_i sont des sous-ensembles classiques dans les domaines d'attributs correspondants. Il a été montré que le résultat de leur application conjointe (incluant la fusion des résultats obtenus à partir de chaque règle) peut être mis sous la forme d'un produit matriciel min-max [53] ; voir [5] pour le cas général. Le résultat de ce produit min-max est une distribution de possibilité sur une collection d'alternatives mutuellement exclusives (induite par des conclusions pondérées sur les Q_i). De plus, la vision conditionnelle peut être étroitement liée à la LPB, puisque $N(q|p) = N(\neg p \vee q)$ si $N(q|p) > 0$.

2.4 Réseaux possibilistes

Comme pour les distributions de probabilité conjointes, une distribution de possibilité conjointe associée à des variables ordonnées $X_1, \ldots, X_n$ peut être décomposée en termes de distributions de possibilité conditionnelle à l'aide d'une règle de chaînage, en utilisant $\star = \min$, ou $\star =$ produit :

$$\pi(X_1,..., X_n) = \pi(X_n | X_1,..., X_{n-1}) \star ... \star \pi(X_2 | X_1) \star \pi(X_1)$$

De la même manière que pour les réseaux bayésiens, une forme d'indépendance permet de simplifier la décomposition. Cependant, il existe plusieurs définitions de l'indépendance possibiliste conditionnelle entre variables en théorie des possibilités qualitatives, l'une étant symétrique : $\Pi(x, y|z) = \min(\Pi(x|z), \Pi(y|z))$ et une autre, plus forte, étant asymétrique : $\Pi(x|z) = \Pi(x|z, y)$. Dans le cadre quantitatif, l'indépendance basée sur le produit entre variables ($\forall x, y, z$, $\Pi(x|y, z) = \Pi(x|z)$ où $\Pi(y, z) > 0$) est symétrique car elle est équivalente à $\forall x, y, z$, $\Pi(x, y|z) = \Pi(x|z) \cdot \Pi(y|z)$. Il existe des algorithmes efficaces pour l'inférence dans les réseaux possibilistes. [9], [69].
Les réseaux possibilistes et les bases LPB sont des représentations compactes des distributions de possibilité. Une caractéristique remarquable de ce cadre est que les réseaux possibilistes peuvent être directement traduits en bases LPB et vice-versa, que le conditionnement soit basé sur le minimum ou sur le produit [14].
Des formats de représentation hybrides ont été introduits où des bases en LPB sont associées localement aux nœuds d'une structure graphique plutôt que des tables de possibilités conditionnelles [31].
Ainsi, le cadre de la LPB offre de multiples formats de représentation équivalents : ensemble de formules logiques

2. Dans ce dernier cas, la possibilité et la nécessité peuvent être interprétées comme une probabilité supérieure et inférieure, voir, e.g., [53].

priorisées, réseaux possibilistes, mais aussi ensemble de conditionnels de la forme $\Pi(p \wedge q) > \Pi(p \wedge \neg q)$ ($\Leftrightarrow N(q|p) > 0$), tous sémantiquement équivalents à des préordres sur les interprétations (c'est-à-dire à des distributions de possibilité). Il existe des algorithmes permettant de traduire un format dans un autre [14].

Par ailleurs, les réseaux possibilistes ont été étudiés du point de vue du raisonnement causal, en utilisant le concept d'*intervention*, qui revient à imposer les valeurs de certaines variables afin de révéler leur influence sur d'autres [11].

2.5 Gestion des incohérences

Le niveau d'incohérence $inc(K)$ d'une base K en LPB fournit un outil pour gérer les incohérences. Cependant, la LPB souffre d'un « effet de noyade » puisque toutes les formules en dessous de $inc(K)$ sont perdues même si elles ne participent pas à une sous-base incohérente minimale. Il existe différentes manières d'élargir l'ensemble des conséquences qui peuvent être déduites de K [21].

Une façon de le faire tout en préservant un ensemble cohérent de conséquences est la suivante. Étant donnée une base K en LPB, on construit sa complétion paraconsistante K^o constituée de formules bi-pondérées : pour chaque formule (p, α) de K, on calcule un triplet (p, β, γ) où β (resp. γ) est le degré le plus élevé avec lequel p (resp. $\neg p$) est soutenu dans K (p est dit *soutenu* dans K *au moins au degré* β s'il existe une sous-base *cohérente* de $(K_\beta)^*$ qui prouve p).

Le sous-ensemble des formules de la forme $(p, \beta, 0)$ dans K^o ne sont pas paraconsistantes et conduisent à des conclusions sûres. Nous pouvons toujours obtenir un ensemble plus large de conclusions cohérentes à partir de K^o comme suit. Nous avons besoin de deux évaluations : i) le niveau d'*infaillibilité* d'un ensemble cohérent S de formules : $\mathrm{UD}(S) = \min\{\beta \mid (p, \beta, \gamma) \in K^o \text{ et } p \in S\}$; ii) le niveau d'*insécurité* d'un ensemble cohérent S de formules : $\mathrm{US}(S) = \max\{\gamma | (p, \beta, \gamma) \in K^o \text{ et } p \in S\}$. Alors une inférence $\vdash_{SS}$, nommée relation de conséquence *soutenue de manière sûre*, est définie par $K^o \vdash_{SS} q$ si et seulement $\exists$ un sous-ensemble minimal cohérent S qui implique classiquement q tel que $\mathrm{UD}(S) > \mathrm{US}(S)$. On peut montrer que l'ensemble $\{q \mid K^o \vdash_{SS} q\}$ est classiquement cohérent. Voir [49] pour les détails, les discussions et d'autres approches de la gestion de l'incohérence dans le cadre de la LPB, y compris la logique quasi-possibiliste où l'utilisation de la résolution après l'introduction d'une disjonction est interdite (pour éviter le *sequitur ex falso quodlibet*).

2.6 Logique possibiliste symbolique

Il peut y avoir plusieurs raisons pour gérer les niveaux de certitude des formules en LPB de manière *symbolique* : notamment pour garder une trace de l'impact de certains niveaux dans le calcul, ou parce que leur valeur est inconnue. Dans ce dernier cas, les valeurs des niveaux de certitude associés aux formules (toujours supposées appartenir à une échelle totalement ordonnée) sont inconnues, mais l'ordre relatif entre certaines d'entre elles peut être partiellement connu. Dans [29], cela est codé au moyen d'une logique propositionnelle typée, où les formules possibilistes sont des clauses avec des littéraux spéciaux qui font référence aux niveaux. Les contraintes sur l'ordre de certains des niveaux se traduisent en formules logiques du type correspondant et sont rassemblées dans une base de connaissances auxiliaire distincte. Le processus d'inférence est caractérisé par l'utilisation de « variables d'oubli » pour gérer les niveaux symboliques, et ainsi un processus d'inférence est obtenu au moyen d'une compilation en DNF des deux bases de connaissance [29].

Lorsque l'ordre des poids est complètement connu, ce codage offre un moyen de compiler une base de connaissance possibiliste afin de pouvoir en traiter l'inférence en temps polynomial [30].

Dans une approche [35] qui rejoint la précédente pour traiter les connaissances partielles sur la valeur relative des niveaux de certitude, deux méthodes d'inférence syntaxique sont proposées : l'une calcule le degré de nécessité d'une formule possibiliste en utilisant la notion de sous-base minimale incohérente, tandis que l'autre s'inspire des ATMS, en utilisant les nogoods et les labels.

2.7 Extensions de la logique possibiliste basées sur des treillis

Il existe plusieurs extensions de la logique possibiliste où les poids sont des niveaux de certitude combinés avec des ensembles tels que des périodes de temps [40], des ensembles de sources, ou des groupes d'agents [8, 54] qui conduisent à utiliser des structures de treillis distributif pseudo-complémenté. Lorsque les ensembles sont remplacés par un singleton unique (c'est-à-dire que nous considérons un instant, une source ou un agent), la logique possibiliste de base est retrouvée.

Nous prenons l'exemple de la logique possibiliste multi-agents pour expliquer l'idée. Les formules (propositionnelles) sont désormais associées à un sous-ensemble d'agents : chaque formule (p, A) signifie que *au moins tous* les agents de A croient que p est vrai. Une telle pondération booléenne introduit une différence notable : le supremum de deux sous-ensembles propres peut être l'univers entier (tandis que le supremum de deux niveaux non maximum n'est jamais le niveau maximum dans une échelle totalement ordonnée). C'est pourquoi la règle explicite de renforcement $(p, A), (p, B) \vdash (p, A \cup B)$ est nécessaire, au niveau syntaxique, à côté de règles d'inférence sur l'affaiblissement du sous-ensemble, le modus ponens et la résolution. Les théorèmes de correction et de complétude sont valides par rapport à une sémantique en termes de fonctions de possibilité et de nécessité à valeurs ensemblistes : $\boldsymbol{\Pi}(p) = \bigcup_{w \models p} \boldsymbol{\pi}(w)$ où $\boldsymbol{\pi}(w)$ est le sous-ensemble *maximal* des agents qui trouvent l'interprétation ω possible, et $\boldsymbol{N}(p) = [\boldsymbol{\Pi}(\neg p)]^c = \bigcap_{w \models \neg p} [\boldsymbol{\pi}(w)]^c$ (où ^{c} désigne la complémentation d'ensemble).

Nous avons maintenant deux types de normalisation conduisant à une vision plus riche de la / (l'in)cohérence : l'une qui signifie que chaque agent trouve au moins un ω possible ($\forall a, \exists \omega, a \in \boldsymbol{\pi}(\omega)$, c'est-à-dire qu'aucun agent n'est incohérent. Cette condition est plus faible que la condition $\exists \omega, \boldsymbol{\pi}(\omega) = \mathit{All}$ (All est l'ensemble de tous

les agents), ce qui signifie qu'il existe une interprétation que tous les agents croient possible, exprimant une condition de cohérence collective. Par exemple, la base $K = \{(p, A), (\neg p, A^c)\}$ viole la dernière condition, mais pas la première.

Cela s'étend au cas général où les propositions sont à la fois associées à un niveau de certitude et à un ensemble d'agents. Les formules sont alors de la forme $(p, \alpha/A)$ où A est un sous-ensemble d'agents et $\alpha \in (0, 1]$, qui se lit « au moins tous les agents de A sont certains de p au moins au niveau α ». La sémantique est alors en termes de fonctions de possibilité et de nécessité floues : Le poids symbolique α/A représente un ensemble flou d'agents a avec des degrés d'appartenance α si l'agent $a \in A$, et 0 sinon.

Une logique de raisonnement sur les raisons [54] gère des paires de la forme (p, x) où p et x sont deux formules logiques propositionnelles exprimées dans deux langages distincts, p est appelé une affirmation et x une raison. La formule (p, x) se lit donc "x est une raison pour p". (p, x) est plus faible que $(\neg x \vee p, 1)$ (le premier n'entraîne pas le second). La vérité de (p, x) signifie que toutes les situations où x est vrai sont des raisons de croire p. La sémantique de cette logique est isomorphe à celle de la logique multi-agent précédente ; une extension peut prendre en compte la force des raisons [54]. Cette logique des raisons s'apparente à la logique des "supporters" [67], mais est un peu plus simple.

La logique possibiliste basée sur les intervalles [26, 28] est une autre extension basée sur un treillis de la logique possibiliste, où les valeurs possibles de niveaux de certitude mal connus sont restreintes par des intervalles.

Mentionnons enfin une manière de conserver une structure ordonnée linéairement tout en enrichissant l'échelle. En LPB, seul le plus petit poids des formules utilisées dans une preuve est conservé ; aucune différence n'est faite par exemple entre une preuve avec une seule prémisse faible et une preuve avec plusieurs prémisses faibles de même force. Cela peut être capturé en utilisant une nouvelle règle de résolution $(\neg p \vee q, \widetilde{\alpha})$; $(p \vee r, \widetilde{\beta}) \vdash (q \vee r, \widetilde{\alpha\beta})$ où $\widetilde{\alpha}$ et $\widetilde{\beta}$ sont des listes de poids, et $\widetilde{\alpha\beta}$ est la liste obtenue par concaténation. Nous pouvons ensuite classer les preuves en fonction de leur force en utilisant un classement lexicographique des listes (une fois qu'elles ont été complétées par des 1 pour les rendre de longueur égale) ; ceci est décrit dans [52].

2.8 Logique possibiliste bipolaire

En théorie des possibilités, il existe deux autres fonctions d'ensemble : i) une mesure de *possibilité garantie* ou de possibilité *forte* (voir, par exemple, [48]) : $\Delta(A) = \inf_{u \in A} \pi(u)$ qui estime dans quelle mesure *tous* les états dans A sont possibles selon les observations. $\Delta(A)$ peut être utilisé comme un degré de support garanti pour A, et son dual $\nabla(A) = 1 - \Delta(A^c) = \sup_{u \notin A} 1 - \pi(u)$ évalue le degré de nécessité potentielle ou *faible* de A, car il vaut 1 dès qu'un état u hors de A est impossible. Les fonctions Δ et ∇ sont *décroissantes* par rapport à l'inclusion (en complet contraste avec Π et N qui sont croissantes). Elles satisfont les propriétés caractéristiques $\Delta(A \cup B) = \min(\Delta(A), \Delta(B))$ et $\nabla(A \cap B) = \max(\nabla(A), \nabla(B))$.

Ainsi la contrainte $\Delta(p) \geq \gamma$, notée syntaxiquement $[p, \gamma]$, exprime que tout modèle de p est au moins possible au degré γ. Ceci peut être représenté par l'ensemble flou $\delta_{[p,\gamma]}(\omega) = 0$ if $\omega \models \neg p$, and $\delta_{[p,\gamma]}(\omega) = \gamma$ if $\omega \models p$. Un ensemble de contraintes $P = \{[q_j, \gamma_j] | j = 1, k\}$ est alors représenté par la distribution de possibilité $\delta_P(\omega) = \max_{j=1,k} \delta_{[q_j,\gamma_j]}(\omega)$ en cumulant les possibilités garanties. Notons que δ_P est obtenu comme la combinaison *disjonctive* basée sur le max de la représentation de chaque formule dans P. Ceci contraste avec π_K (dans la section 2.2) obtenu comme une combinaison *conjonctive* basée sur le min. Ainsi, une distribution de possibilité peut être représentée "par en haut" au moyen de contraintes basées sur la nécessité, et "par en bas" au moyen de contraintes basées sur la possibilité garantie. Au niveau syntaxique, les contraintes basées sur la nécessité sont naturellement associées à une décomposition en CNF pondérée, tandis que les contraintes basées sur Δ conduisent à une décomposition en DNF pondérée. Cette dernière est régie au niveau de l'inférence par le pendant suivant de la règle de résolution

$$[\neg p \wedge q, \gamma], [p \wedge r, \gamma']) \vdash [q \wedge r, \min(\gamma, \gamma')].$$

Une contrainte $[p, \gamma]$ basée sur Δ correspond naturellement à l'expression d'une information positive, c'est-à-dire que les interprétations qui sont des modèles de p sont possibles au moins au degré γ, tandis qu'une contrainte (p, α) basée sur N correspond à une expression négative indiquant que les contre-modèles de p sont quelque peu impossibles (leur possibilité est au plus $1 - \alpha$). Ainsi, plus d'informations positives augmentent δ_P en rendant plus d'interprétations réellement possibles, tandis que plus d'informations négatives diminuent π_K en restreignant davantage les mondes possibles [17].

On peut utiliser soit des formules basées sur Δ, soit des formules basées sur N pour représenter les informations disponibles, selon ce qui semble le plus pratique. Quand il est judicieux de faire la distinction entre les informations positives et les informations négatives (par exemple, des exemples réels de prix et des prix non-interdits par la réglementation), nous devons conserver séparément la base de connaissance K et la base de connaissance P, chacune sémantiquement associée à leurs distributions respectives (censées satisfaire la condition de cohérence $\delta_P \leq \pi_K$) ; voir [38], [18] pour des cadres de gestion de ce dernier cas.

2.9 Logique possibiliste généralisée

En LPB, seules les conjonctions de formules de logique possibiliste sont autorisées. Mais comme (p, α) est sémantiquement interprété comme $N(p) \geq \alpha$, une formule possibiliste peut être manipulée comme une formule propositionnelle qui est vraie (si $N(p) \geq \alpha$) ou fausse (si $N(p) < \alpha$). Les formules possibilistes peuvent alors être combinées avec tous les connecteurs propositionnels, y compris la disjonction et la négation. Il s'agit de la *logique possibiliste généralisée* (LPG) [57, 51]. La LPG est une logique propositionnelle à deux niveaux, dans laquelle les formules propositionnelles sont encapsulées par des opérateurs modaux pondérés interprétés en termes de mesures de nécessité et de possibilité.

La LPG utilise une échelle finie de degrés de certitude $\Lambda_k = \{0, \frac{1}{k}, \frac{2}{k}, ..., 1\}$ ($k \in \mathbb{N} \setminus \{0\}$) ; $\Lambda_k^+ = \Lambda_k \setminus \{0\}$. Le langage de la LPG, $\mathcal{L}_{\mathbf{N}}^k$, est construit sur un langage propositionnel $\mathcal{L}$ comme suit : i) Si $p \in \mathcal{L}$, $\alpha \in \Lambda_k^+$, alors $\mathbf{N}_\alpha(p) \in \mathcal{L}_{\mathbf{N}}^k$; ii) si $\varphi \in \mathcal{L}_{\mathbf{N}}^k$,$\psi \in \mathcal{L}_{\mathbf{N}}^k$, alors $\neg\varphi$ et $\varphi \wedge \psi$ sont aussi dans $\mathcal{L}_{\mathbf{N}}^k$. Ici, $\mathbf{N}_\alpha(p)$ représente (p, α), la notation soulignant la proximité avec la logique modale. Ainsi, un agent affirmant $\mathbf{N}_\alpha(p)$ a un état épistémique tel que $N(p) \geq \alpha > 0$. Ainsi, $\neg\mathbf{N}_\alpha(p)$ signifie $N(p) < \alpha$, ce qui est équivalent à $N(p) \leq \alpha - \frac{1}{k}$ et donc $\Pi(\neg p) \geq 1 - \alpha + \frac{1}{k}$. En particulier, $\mathbf{\Pi}_1(p) \equiv \neg\mathbf{N}_{\frac{1}{k}}(\neg p)$ si $k > 1$. Ainsi, en LPG, on peut faire la distinction entre l'absence de certitude suffisante que p est vrai ($\neg\mathbf{N}_\alpha(p)$) et l'affirmation plus forte que p est quelque peu certainement faux ($\mathbf{N}_\alpha(\neg p)$).

La sémantique de la LPG est comme en LPB définie en termes de distributions de possibilité normalisées sur des interprétations propositionnelles, où les degrés de possibilité sont dans Λ_k. Toute distribution de possibilité à valeurs sur Λ_k telle que $N(p) \geq \alpha$ est un modèle d'une formule $\mathbf{N}_\alpha(p)$ de la LPG. Mais, l'ensemble des distributions de possibilité satisfaisant une formule de LPG n'a pas toujours un plus grand élément, comme c'était le cas en LPB.

La LPG peut être considérée comme un fragment de la logique modale KD, sans modalités imbriquées, mais les modalités y sont graduées. Voir [57] pour son axiomatique, les résultats de correction et complétude, et l'étude de sa complexité. La LPG est un puissant cadre unificateur pour divers formalismes de représentation des connaissances, y compris la logique possibiliste avec des formules partiellement ordonnées, ou une logique d'assertions conditionnelles. Le raisonnement sur l'ignorance explicite, ou certaines tâches de raisonnement sur plusieurs agents, telles que le problème des "enfants couverts de boue" ("muddy children"), peuvent également être traités en LPG [52].

De même, une logique possibiliste multi-agents généralisée qui permet la disjonction et la négation de ses formules de base a été récemment étudiée [54]. Une construction similaire s'applique également à la logique des raisons.

2.10 Relations avec d'autres cadres

Les fonctions de rang ("ranking functions") de Spohn [82] sont similaires aux mesures de possibilité mais elles sont évaluées sur des entiers positifs. Elles utilisent donc des échelles différentes pour évaluer la (im)plausibilité, ce qui rend leurs pouvoirs expressifs quelque peu différents. En effet, il n'y a pas de côté logique pour les fonctions de classement puisqu'il n'y a pas de contrepartie au modus ponens pondéré, et le conditionnement de Spohn, basé sur l'addition, s'inspire des probabilités infinitésimales, tandis que la logique possibiliste n'utilise que des opérations idempotentes telles que max et min [50].

La logique de Markov [78] utilise des formules pondérées pour encoder de manière compacte une distribution de probabilité, mais les pondérations ne sont pas faciles à interpréter. Cependant, on peut toujours construire une base logique possibiliste qui capture exactement un réseau logique de Markov ; voir [65], [57].

3 Applications

La logique possibiliste a trouvé des applications dans de nombreux domaines de recherche en IA. En raison de l'espace limité, nous n'avons pu sélectionner qu'un petit échantillon de références pour chaque application.

3.1 De la gestion de l'incertitude au raisonnement par défaut

La LPB a été conçue à l'origine pour propager l'incertitude dans des moteurs d'inférence pour les systèmes experts, en tirant parti du format matriciel [60].

La capacité de LPB à gérer l'incohérence, en utilisant le niveau d'incohérence d'une base de connaissances, est exploitée dans le raisonnement par défaut, une fois les règles par défaut traduites en formules possibilistes. Une règle par défaut "généralement, si p alors q"est représentée par la condition $\Pi(p \wedge q) > \Pi(p \wedge \neg q) \iff N(q|p) > 0$ Ainsi, $N(q|p) > 0$ exprime que dans le contexte où p est vrai, avoir q vrai est strictement plus possible que q faux. Comme pour les probabilités, ce conditionnement n'est pas monotone. On peut avoir que $N(q|p) > 0$, alors que la conclusion opposée $N(\neg q|p \wedge p') > 0$ est vraie dans le contexte plus restreint $p \wedge p'$.

Ensuite, à partir de la plus grande distribution de possibilité sous-jacente à un ensemble cohérent de défauts $\Pi(p_i \wedge q_i) > \Pi(p_i \wedge \neg q_i)$ pour $i = 1, n$, il est possible de stratifier l'ensemble des défauts selon leur spécificité (les défauts les plus spécifiques reçoivent les niveaux les plus élevés), puis de les coder par des formules de logique possibiliste [20] : chaque défaut est transformé en une clause possibiliste $(\neg a_i \vee b_i, N(\neg a_i \vee b_i))$, où N est calculé à partir de la plus grande distribution de possibilité induite par l'ensemble des contraintes modélisant la base de règles par défaut. Ce codage tire parti du fait que lorsque de nouvelles informations sûres sont reçues, le niveau d'incohérence de la base ne peut pas diminuer, et s'il augmente strictement, certaines inférences qui étaient sûres auparavant sont maintenant noyées dans le nouveau niveau d'incohérence de la base et ne sont donc plus possibles, d'où un mécanisme de conséquence non monotone. Il a été prouvé que cette approche est en plein accord avec une approche basée sur les postulats du raisonnement non monotone [19]. Cela est également équivalent à une modélisation probabiliste des conditionnelles en termes d'un type spécial de distributions de probabilités appelées probabilités à grandes marches [22].

3.2 Révision de croyances

Le raisonnement non monotone et la révision des croyances sont étroitement liés, de sorte que la LPB trouve également une application en révision des croyances. En effet, les relations de nécessité comparative (qui peuvent être codées par des mesures de nécessité) ne sont rien d'autre que les relations d'enracinement épistémique [46] qui sous-tendent les processus de révision des croyances bien conduits [61]. Cela permet au cadre de la LPB de fournir des opérateurs de révision syntaxique qui s'appliquent aux bases de connais-

sances possibilistes, y compris dans le cas d'entrées incertaines [24, 76]. En LPB, l'enracinement épistémique est rendu explicite par les niveaux de certitude des formules. En outre, dans un processus de révision, on s'attend à ce que toutes les formules indépendantes de la validité des informations d'entrée restent dans l'état révisé de croyance ; cette idée peut recevoir une signification précise en utilisant une définition de l'indépendance causale possibiliste entre les événements [37].

3.3 Fusion d'informations

La combinaison des distributions de possibilité peut être effectuée de manière équivalente en termes de bases de LPB : La contrepartie syntaxique de la combinaison point par point de deux distributions de possibilité π_1 et π_2 en une distribution $\pi_1 \circledast \pi_2$ par tout opérateur de combinaison monotone $\circledast$ tel que $1 \circledast 1 = 1$, peut être calculée, suivant une idée proposée pour la première fois dans [33]. A savoir, si la base LPB K_1 est associée à π_1 et la base K_2 à π_2, une base LPB $K_{1\circledast 2}$ sémantiquement équivalente à $\pi_1 \circledast \pi_2$ est donnée par : $\{(p_i, 1-(1-\alpha_i) \circledast 1)$ s.t. $(p_i,\alpha_i) \in K_1\} \cup \{(q_j, 1-1\circledast(1-\beta_j))$ s.t. $(q_j,\beta_j) \in K_2\} \cup \{p_i \vee q_j, 1-(1-\alpha_i) \circledast (1-\beta_j))$ s.t. $(p_i,\alpha_i) \in K_1, (q_j,\beta_j) \in K_2\}$. Pour $\circledast = \min$, on obtient $K_{1\oplus 2} = K_1 \cup K_2$ avec $\pi_{K_1\cup K_2} = \min(\pi_1,\pi_2)$ comme attendu (combinaison conjonctive). Pour $\circledast = \max$ (combinaison disjonctive), on a $\Gamma_{1\oplus 2} = \{(p_i \vee q_j, \min(\alpha_i,\beta_j))$ s.t. $(p_i,\alpha_i) \in K_1$, et $(q_j,\beta_j) \in K_2\}$. Avec des opérateurs $\oplus$ non idempotents, certains effets de renforcement peuvent être obtenus. Voir, par exemple [63], pour une étude sur les opérateurs de fusion en LPB. En outre, cette approche peut également être appliquée au codage syntaxique de la fusion de bases logiques *classiques* basées sur la distance de Hamming (où les distances sont calculées entre chaque interprétation et les différentes bases en logique classique, donnant ainsi naissance à des équivalents de distributions de possibilité) [15]. Dans [27] une représentation basée sur Δ est utilisée ; voir [32] pour un exemple illustratif.

3.4 Logique de description

La gestion possibiliste de l'incertitude en logique de description a été proposée pour la première fois dans [75]. Elle présente des avantages informatiques, en particulier dans le cas de la famille *possibilistic DL-Lite* où l'extension de la puissance expressive de DL-Lite se fait sans coûts de calcul supplémentaires [12] ; il est alors facile d'utiliser l'opération min pour la fusion de bases DL-Lite possibilistes.
Une méthode polynomiale pour calculer une réparation possibiliste unique pour une ABox pondérée partiellement pré-ordonnée qui peut être incohérente par rapport à la TBox a été proposée dans [7].

3.5 Programmation logique

Différentes propositions ont été faites pour une gestion possibiliste de l'incertitude en programmation logique et en programmation par ensembles-réponses [2, 71, 72, 62, 6].
En outre, une application remarquable de la LPG est sa capacité à coder des programmes par ensembles-réponses (ASP), en utilisant une échelle à 3 valeurs $\Lambda_2 = \{0, 1/2, 1\}$. Ce qui permet de faire la distinction entre les propositions dont on est totalement certain et les propositions qu'on considère seulement comme plausibles, et d'expliciter en LPG la sémantique épistémique de règles avec négation par échec. Par exemple, la règle ASP $r \leftarrow p \wedge \text{not } q$ est codée par $\mathbf{N}_1(p) \wedge \mathbf{\Pi}_1(\neg q) \rightarrow \mathbf{N}_1(r)$ en LPG. Voir [57].

3.6 Bases de données

Le calcul de *provenance*, basé sur deux opérations formant un demi-anneau, combine et propage des annotations associées à des données. Ce calcul, lorsqu'il est basé sur les opérations max et min, correspond exactement à l'évaluation des requêtes lorsque les données sont étiquetées avec des niveaux de certitude, comme en LPB [55].
La LPB présente un intérêt pour la conception de bases de données où la présence de certains tuples dans la base de données peut n'être possible que dans une certaine mesure, et où les dépendances fonctionnelles sont incertaines [70]. Le même genre d'idée a été appliqué au nettoyage possibiliste de données [64].

3.7 Préférences et décision qualitative

Une formule de LPB (p,α) peut représenter un objectif p avec un niveau de priorité α. Des préférences telles que "Je préfère p à q et q à r" (où p, q, r peuvent ne pas être mutuellement exclusifs) peuvent être représentées par la base possibiliste $P = \{(p \vee q \vee r, 1), (p \vee q, 1-\gamma), (p, 1-\beta)\}$ avec $\gamma < \beta < 1$, comme un ensemble d'objectifs plus ou moins impératifs. D'autres formats tels que les conditionnels, les réseaux possibilistes, la représentation basée sur Δ sont également intéressants pour représenter les préférences [23]. De plus, l'expression des préférences peut être bipolaire : énoncé de situations qui sont plus ou moins fortement rejetées, et de situations qui sont garanties satisfaisantes à un certain degré [16]. Mentionnons l'équivalence représentationnelle [13] entre la logique de choix qualitatif QCL [34] et la logique des possibilités garanties. De telles préférences sont également intéressantes pour l'expression de requêtes flexibles à une base de données [58].
Les réseaux possibilistes (basés sur le produit) ont été utilisés pour représenter des préférences conditionnelles possibilistes avec des pondérations symboliques [10]. Les interprétations (correspondant aux différentes alternatives) sont alors comparées en termes de vecteurs symboliques exprimant la satisfaction ou la violation des formules associées aux différentes préférences, en utilisant des relations d'ordre appropriées. La représentation obtenue est compatible avec celle des CP-nets, dont elle fournit une bonne approximation [83].
La théorie des possibilités fournit un cadre pour la décision qualitative sous incertitude où des critères de décision pessimistes et optimistes ont été axiomatisés [56]. La contrepartie de ces critères, lorsque la connaissance et les préférences sont sous la forme de *deux bases en LPB distinctes*, est donnée par les définitions [42] :
- l'utilité pessimiste $u_*(d)$ de la décision d est le maximum $\alpha \in \mathcal{S}$ s.t. $K_\alpha \wedge d \vdash_{PL} P_{\underline{\nu(\alpha)}}$,

- l'utilité optimiste $u^*(d)$ de d est le maximum $\nu(\alpha) \in \mathcal{S}$ s.t. $K_{\underline{\alpha}} \wedge d \wedge P_{\underline{\alpha}} \not\equiv \bot$,
où $\mathcal{S}$ est une échelle totalement ordonnée finie bornée, ν une bijection inversant l'ordre de cette échelle ; K_α est un ensemble de formules logiques classiques rassemblant les connaissances certaines à un niveau au moins α, et où $P_{\underline{\beta}}$ est un ensemble de formules logiques classiques constitué d'un ensemble d'objectifs dont le niveau de priorité est *strictement* supérieur à β. Une décision pessimiste optimale assure la satisfaction de tous les objectifs de $P_{\underline{\nu(\alpha)}}$ avec une priorité aussi basse que possible, en utilisant seulement une partie K_α de la connaissance qui a une grande certitude. Une décision optimiste optimale maximise la cohérence de tous les objectifs plus ou moins importants avec tous les éléments de connaissance plus ou moins certains.

3.8 Argumentation

Un langage de programmation logique possibiliste et réfutable qui combine des caractéristiques de la théorie de l'argumentation et de la programmation logique, intégrant également le traitement possibiliste de l'incertitude, a été proposé dans [1].
La logique possibiliste peut être utilisée pour représenter les états mentaux des agents (croyances éventuellement imprégnées d'incertitude et objectifs prioritaires), pour réviser les bases de croyances et pour décrire la procédure de décision pour sélectionner une nouvelle offre dans une négociation basée sur l'argumentation [3].
La logique des raisons [54] qui traite les formules (p, x) exprimant que "x est une raison pour p", où la négation peut être appliquée à p, x et (p, x), offre un cadre riche pour le raisonnement argumentatif.

3.9 Apprentissage automatique

L'étude de l'apprentissage des théories logiques possibilistes [73], a montré que de nombreux résultats d'apprentissage en temps polynomial pour la logique classique peuvent être transférés à l'extension possibiliste respective. La logique possibiliste bipolaire offre un cadre gradué pour étendre le cadre d'apprentissage de l'espace des versions [74].
La LPB peut également être appliquée à la programmation logique inductive (ILP). En effet, avoir un ensemble stratifié de règles logiques du premier ordre comme hypothèses en ILP est intéressant pour l'apprentissage à la fois de règles couvrant les cas normaux et de règles plus spécifiques pour les cas exceptionnels [79], [66].
Une cascade de produits min-max de matrices représentant des règles possibilistes de type si-alors présente une ressemblance structurelle avec un réseau neuronal min-max. Une telle cascade peut être montrée comme étant équivalente à un réseau neuronal min-max, chaque produit matriciel correspondant à une couche et la fonction d'activation utilisée étant l'identité ;voir [5] pour plus de détails.[4] offre une approche neuro-symbolique possibiliste très complète.

3.10 Autres applications

D'autres applications peuvent être trouvées, comme la modélisation des désirs à l'aide de fonctions Δ [43], ou l'expression des buts des agents en logique possibiliste dans des jeux booléens lorsque les agents peuvent avoir une connaissance incomplète des préférences des autres [36].
Une autre application est le codage des politiques d'accès de contrôle [25]. Une description formelle des politiques de sécurité est nécessaire pour vérifier si les propriétés de sécurité sont satisfaites ou non. Les règles de contrôle d'accès, garantissant les propriétés de confidentialité et d'intégrité, sont codées en termes de bases de connaissances stratifiées. La stratification reflète la hiérarchie entre les rôles et est utile pour gérer les conflits.

4 Conclusion

Cet article a passé en revue un grand nombre de travaux sur le développement de la logique possibiliste et ses applications. La logique possibiliste est bien adaptée à la représentation d'informations incomplètes et de croyances acceptées plus ou moins ancrées. Elle reste proche de la logique classique et offre un cadre riche, simple et polyvalent pour la représentation de l'information et le raisonnement qualitatif sur l'incertitude. La théorie des possibilité, comme celle des probabilités, mérite qu'on s'y intéresse !

Références

[1] T. Alsinet, C. I. Chesñevar, L. Godo, and G. R. Simari. A logic programming framework for possibilistic argumentation : Formalization and logical properties. *Fuzzy Sets Syst.*, 159 :1208–1228, 2008.

[2] T.Alsinet,L.Godo, andS.SandriTwo formalisms of extended possibilistic logic programming with context-dependent fuzzy unification : A comparative description.*Elec. Notes in Theo. Comput. Sci.*, 66 (5), 2002.

[3] L. Amgoud and H. Prade. Reaching agreement through argumentation : A possibilistic approach. In *KR*, pages 175–182, 2004.

[4] I. Baaj and P. Marquis. Π-NeSy: A possibilistic neuro-symbolic approach. cs.AI arXiv 2504.07055, 2025.

[5] I. Baaj, J. P. Poli, W. Ouerdane, and N. Maudet. Min-max inference for possibilistic rule-based system. In *FUZZ-IEEE*, pages 1–6. IEEE, 2021.

[6] K. Bauters, S. Schockaert, M. De Cock, and D. Vermeir. Characterizing and extending answer set semantics using possibility theory. *Theory Pract. Log. Program.*, 15(1) :79–116, 2015.

[7] S. Belabbes and S. Benferhat. Computing a possibility theory repair for partially preordered inconsistent ontologies. *IEEETrans. Fuz. Syst.*, 30:3237–3246, 2022.

[8] A. Belhadi, D. Dubois, F. Khellaf-Haned, and H. Prade. Reasoning with multiple-agent possibilistic logic. In*SUM*, LNCS 9858,67-80. Springer, 2016.

[9] N. Ben Amor, S. Benferhat, and K. Mellouli. Anytime propagation algorithm for min-based possibilistic graphs. *Soft Comput.*, 8 :150–161, 2003.

[10] N. Ben Amor, D. Dubois, H. Gouider, and H. Prade. Possibilistic preference networks. *Inf. Sci.*, 460-461 :401–415, 2018.

[11] S. Benferhat. Interventions and belief change in possibilistic graphical models. *Artif. Intell.*, 174(2) :177–189, 2010.

[12] S. Benferhat, Z. Bouraoui, and Z. Loukil. Min-based fusion of possibilistic DL-Lite knowledge bases. *Proc. IEEE/WIC/ACM Int. Conf. on Web Intelligence (WI'13)*, Atlanta, 23-28. 2013.

[13] S. Benferhat, G. Brewka, and D. Le Berre. On the relation between qualitative choice logic and possibilistic logic. In *IPMU vol.2*, pages 951–957, 2004.

[14] S. Benferhat, D. Dubois, L. Garcia, and H. Prade. On the transformation between possibilistic logic bases and possibilistic causal networks. *Int. J. Approx. Reas.*, 29 :135–173, 2002.

[15] S. Benferhat, D. Dubois, S. Kaci, and H. Prade. Possibilistic merging and distance-based fusion of propositional information*Ann. Math. A I*, 34 :217–252, 2002.

[16] S. Benferhat, D. Dubois, S. Kaci, and H. Prade. Bipolar possibility theory in preference modeling : Representation, fusion and optimal solutions. *Inf. Fusion*, 7 :135–150, 2006.

[17] S. Benferhat, D. Dubois, S. Kaci, and H. Prade. Modeling positive and negative information in possibility theory.*Int. J. Intel. Syst.*,23 :1094–1118, 2008.

[18] S. Benferhat, D. Dubois, S. Kaci, and H. Prade. Bipolar possibilistic representationsProc.UAI,45-52,2002.

[19] S. Benferhat, D. Dubois, and H. Prade. Nonmonotonic reasoning, conditional objects and possibility theory. *Artif. Intell.*, 92 :259–276, 1997.

[20] S. Benferhat, D. Dubois, and H. Prade. Practical handling of exception-tainted rules and independence information in possibilistic logic. *Appl. Intell.*, 9(2) :101–127, 1998.

[21] S. Benferhat, D. Dubois, and H. Prade. An overview of inconsistency-tolerant inferences in prioritized knowledge bases. In *Fuzzy Sets, Logic and Reasoning about Knowledge*. 395-417, Kluwer, 1999.

[22] S. Benferhat, D. Dubois, and H. Prade. Possibilistic and standard probabilistic semantics of conditional knowledge bases. *J. Log. Comput.*, 9 :873–895, 1999.

[23] S. Benferhat, D. Dubois, and H. Prade. Towards a possibilistic logic handling of preferences. *Appl. Intell.*, 14(3) :303–317, 2001.

[24] S. Benferhat, D. Dubois, H. Prade, and M.-A. Williams. A framework for iterated belief revision using possibilistic counterparts to Jeffrey's rule. *Fundam. Inform.*, 99(2) :147–168, 2010.

[25] S. Benferhat, R. El Baida, and F. Cuppens. A possibilistic logic encoding of access control. In *FLAIRS Conf.*, pages 481–485, 2003.

[26] S. Benferhat, J. Hué, Sylvain Lagrue, and J. Rossit. Interval-based possibilistic logic. In *IJCAI*, pages 750–755, 2011.

[27] S. Benferhat and S. Kaci. Logical representation and fusion of prioritized information based on guaranteed possibility measures : Application to the distance-based merging of classical bases. *Artif. Intell.*, 148(1-2) :291–333, 2003.

[28] S. Benferhat, A. Levray, K. Tabia, and V. Kreinovich. Compatible-based conditioning in interval-based possibilistic logic. In *IJCAI, 2777-2783*, 2015.

[29] S. Benferhat and H. Prade. Encoding formulas with partially constrained weights in a possibilistic-like many-sorted propositional logic. *Proc. 9th IJCAI*, Edinburgh, 1281-1286. 2005.

[30] S. Benferhat and H. Prade. Compiling possibilistic knowledge bases. *ECAI*, 337-341, IOS Press, 2006.

[31] S. Benferhat and S. Smaoui. Hybrid possibilistic networks. *Int. J. Approx. Reason.*, 44(3) :224–243, 2007.

[32] S. Benferhat and C. Sossai. Reasoning with multiple-source information in a possibilistic logic framework. *Inf. Fusion*, 7(1) :80–96, 2006.

[33] L. Boldrin and C. Sossai.Local possibilistic logic. *J. of Applied Non-Classical Logics*, 7(3) :309–333, 1997.

[34] G. Brewka, S. Benferhat, and D. Le Berre. Qualitative choice logic. *Artif. Intel.*, 157 :203–237, 2004.

[35] C. Cayrol, D. Dubois, and F. Touazi. Symbolic possibilistic logic : completeness and inference methods. *J. Log. Comput.*, 28(1) :219–244, 2018.

[36] S. De Clercq, S. Schockaert, A. Nowé, and M. De Cock. Modelling incomplete information in Boolean games using possibilistic logic. *Int. J. Approx. Reason.*, 93 :1–23, 2018.

[37] D. Dubois, L. Fariñas del Cerro, A. Herzig, and H. Prade. A roadmap of qualitative independence. In *Fuzzy Sets, Logics and Reasoning about Knowledge*, pages 325–350. Kluwer, 1999.

[38] D. Dubois, P. Hajek, and H. Prade. Knowledge-driven versus data-driven logics. *J. Logic, Language, and Information*, 9 :65–89, 2000.

[39] D. Dubois, J. Lang, and H. Prade. Theorem proving under uncertainty - A possibility theory-based approach. In *IJCAI*, pages 984–986, 1987.

[40] D. Dubois, J. Lang, and H. Prade. Timed possibilistic logic. *Fund. Infor.*, 15 :211–234, 1991.

[41] D. Dubois, J. Lang, and H. Prade. Possibilistic logic. In *Handbook of Logic in Artificial Intelligence and Logic Programming 3*. 439-513, OUP, 1994.

[42] D. Dubois, D. Le Berre, H. Prade, and R. Sabbadin. Using possibilistic logic for modeling qualitative decision : ATMS-based algorithms. *Fundamenta Informaticae*, 37 :1–30, 1999.

[43] D.Dubois, E.Lorini, and H.Prade.The strength of desires:A logical approach*MindsMach.*,27:199–231, 2017.

[44] D. Dubois and H. Prade. Possibilistic logic under matrix form. In *Fuzzy Logic in Knowledge Engineering*. Verlag TÜV Rheinland, 112-126, 1986.

[45] D. Dubois and H. Prade. *Possibility Theory : An Approach to Computerized Processing of Uncertainty*. Plenum Press, 1988.

[46] D. Dubois and H. Prade. Epistemic entrenchment and possibilistic logic. *Artif. Intell.*, 50 :223–239, 1991.

[47] D. Dubois and H. Prade. Possibilistic logic : a retrospective and prospective view. *Fuzzy Sets Syst.*, 144(1) :3–23, 2004.

[48] D. Dubois and H. Prade. Possibilistic logic - an overview. In *Computational Logic*, Handbook of the History of Logic,9. 283-342, Elsevier, 2014.

[49] D. Dubois and H. Prade. Inconsistency management from the standpoint of possibilistic logic. *Int. J. of Uncertainty, Fuzziness and Knowledge-Based Systems*, 23(Supp.-1) :15–30, 2015.

[50] D. Dubois and H. Prade. Qualitative and semi-quantitative modeling of uncertain knowledge - A discussion. In *Computational Models of Rationality*, pages 280–296. College Publications, 2016.

[51] D. Dubois and H. Prade. A crash course on generalized possibilistic logic. In *SUM*, volume 11142 of *LNCS*, pages 3–17. Springer, 2018.

[52] D. Dubois and H. Prade. Possibilistic logic : From certainty-qualified statements to two-tiered logics - A prospective survey. In *JELIA*, volume 11468 of *LNCS*, pages 3–20. Springer, 2019.

[53] D. Dubois and H. Prade. From possibilistic rule-based systems to machine learning - A discussion paper. *SUM*, LNCS 12322, 35-51, Springer. 2020.

[54] D. Dubois and H. Prade. Boolean weighting in possibilistic logic. In *SUM*, volume 15350 of *LNCS*, pages 130–146. Springer, 2024.

[55] D. Dubois and H. Prade. Possibilistic provenance. In *SUM*, LNCS 15350, 147-153. Springer, 2024.

[56] D. Dubois, H. Prade, and R. Sabbadin. Decision-theoretic foundations of qualitative possibility theory. *Europ. J. of Operat. Research*, 128 :459–478, 2001.

[57] D. Dubois, H. Prade, and S. Schockaert. Generalized possibilistic logic : Foundations and applications to qualitative reasoning about uncertainty. *Artif. Intell.*, 252 :139–174, 2017.

[58] D. Dubois, H. Prade, and F. Touazi. A possibilistic logic approach to conditional preference queries. In *Proc. FQAS*. LNCS 8132, 376-388, Springer, 2013.

[59] H. Farreny and H. Prade. Default and inexact reasoning with possibility degrees. *IEEE Trans. Syst. Man Cybern.*, 16(2) :270–276, 1986.

[60] H. Farreny, H. Prade, and E. Wyss. Approximate reasoning in a rule-based expert system using possibility theory :A case study. *IFIP Cong.*, 407-414. 1986.

[61] P. Gärdenfors. *Knowledge in Flux*. MIT Press, 1988.

[62] J. Hué, M. Westphal, and S. Wölfl. Towards a new semantics for possibilistic answer sets. In *KI*, LNCS, 8736, pages 159–170. Springer, 2014.

[63] S. Kaci, S. Benferhat, D. Dubois, and H. Prade. A principled analysis of merging operations in possibilistic logic. In *UAI'00*, pages 24–31, 2000.

[64] H. Koehler and S. Link. Possibilistic data cleaning. *IEEE Trans. Knowl. Data Eng.*, 34 :5939–5950, 2022.

[65] O. Kuzelka, J. Davis, and S. Schockaert. Encoding Markov logic networks in possibilistic logic. In *UAI Conf.*, pages 454–463, 2015.

[66] O. Kuzelka, J. Davis, and S. Schockaert. Induction of interpretable possibilistic logic theories from relational data. In *IJCAI*, pages 1153–1159, 2017.

[67] C. Lafage, J. Lang, and R. Sabbadin. A logic of supporters. In *Information, Uncertainty and Fusion*, pages 381–392. Kluwer, 1999.

[68] J. Lang. Possibilistic logic : complexity and algorithms. In *Algorithms for Uncertainty and Defeasible Reasoning*, pages 179–220. Kluwer, 2001.

[69] A. Levray, S. Benferhat, and K. Tabia. Possibilistic networks : Computational analysis of MAP and MPE inference.*Int. J. Artif. Intel. Tools*, 29 :1–28, 2020.

[70] S. Link and H. Prade. Relational database schema design for uncertain data. *Inf. Syst.*, 84 :88–110, 2019.

[71] P. Nicolas, L. Garcia, I. Stéphan, and C. Lefèvre. Possibilistic uncertainty handling for answer set programming.*A. Math. Artif. Intell.*, 47 :139–181, 2006.

[72] J. C. Nieves, M. Osorio, and U. Cortés. Semantics for possibilistic disjunctive programs. In *LPNMR*, LNCS 4483. 315-320, Springer, 2007.

[73] C. Persia and A. Ozaki. On the learnability of possibilistic theories. In *IJCAI,1870-1876*, 2020.

[74] H. Prade and M. Serrurier. Bipolar version space learning. *Int. J. Intel. Syst.*, 23(10) :1135–1152, 2008.

[75] G.l. Qi, J. Z. Pan, and Q. Ji. Extending description logics with uncertainty reasoning in possibilistic logic. In *ECSQARU*, LNCS 4724 828-839. Springer, 2007.

[76] G.l. Qi and K.w. Wang. Conflict-based belief revision operators in possibilistic logic. AAAI, 800-806, 2012.

[77] N. Rescher. *Plausible Reasoning*. Van Gorcum, 1976.

[78] M. Richardson and P. M. Domingos. Markov logic networks. *Machine Learning*, 62 :107–136, 2006.

[79] M. Serrurier and H. Prade. Introducing possibilistic logic in ILP for dealing with exceptions. *Artificial Intelligence*, 171 :939–950, 2007.

[80] G. L. S. Shackle. *Expectation in Economics*. Cambridge University Press, 1949.

[81] G. Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, 1976.

[82] W. Spohn. *The Laws of Belief : Ranking Theory and Its Philosophical Applications*. Oxford Univ. P., 2012.

[83] N. Wilson, D. Dubois, and H. Prade. CP-nets, π-pref nets, and Pareto dominance. In *SUM*, volume LNCS 11940, pages 169–183. Springer, 2019.

[84] L. A. Zadeh. Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Syst.*, 1(1) :3–28, 1978.

Théorie des possibilités et modèles numériques linéaires pour l'inférence sur une base de croyances

Armand Gaudillier[1], Khaled Belahcène[1], Wassila Ouerdane[1], Sébastien Destercke[2]

[1] CentraleSupelec, Université Paris-Saclay, MICS
[2] UMR CNRS 7253 Heudiasyc, Universite de Technologie de Compiegne, Sorbonne Universite

{prenom.nom} at centralesupelec.fr, sebastien.destercke at hds.utc.fr

Résumé

En combinant logique possibiliste et modèles numériques linéaires, nous proposons un nouveau moyen d'inférer sur une base de croyances. Via l'introduction de degrés d'incertitudes, nous pouvons inférer malgré la présence d'inconsistence dans les informations fournies, et éventuellement les remettre en cause. Nous montrons que mêler modèles numériques et symboliques permet des inférences syntaxiques efficaces. Le tout est illustré à travers un exemple de l'aide à la décision multi-critères.

Mots-clés

Logique possibiliste, Raisonnement sur base de croyances, Aide à la décision multi-critères, complexité, sémantique et syntaxe

1 Introduction

In this paper, we propose a new framework for reasoning on a belief base. We use the term belief as the statements forming our base are possibly uncertain. In multiple fields, as in Multi-Criteria Decision Aiding (MCDA), it is common to reason over statements from the Decision Maker (DM), that are often uncertain. The purpose of the process is to help her [1] choose an alternative among several, each defined on a set of criteria. Her preferences are formalized as statements, from which it is possible to infer. With the hypotheses that her preferences can be modelled by a numerical model, each statement is a constraint that reduces the set of possible models. For a review of the numerical value models used to represent preferences, see [Bourdache, 2020]. Many frameworks for MCDA have been developed over the years. Classical DM preferential information consists of pairwise comparisons of alternatives, the pairs are usually chosen through some heuristics with the idea of improving the model fitness [Ciomek et al., 2017]. Translating this statement into constraints is usually done by excluding all models that do not satisfy the new constraint. An example of this kind of process can be found in [Jacquet-Lagreze and Siskos, 1982], where the UTA framework is introduced. In [Greco et al., 2008], it is extended to a robust framework. While *pointwise* models aim at finding a single value of the parameter of the model, which is usually done by minimizing some loss function, *robust* MCDA frameworks can make an inference only if all of the totally possible models support the inference. This differs from non-robust models, where procedures are applied to choose one of the models and use it to infer. Unfortunately, if the statements received are contradictory, inconsistency can appear during the process. In other words, there are no totally possible models. It may be caused by an error from the DM, or an error when the model was selected by the analyst. If all statements are considered certain, the situation is tricky, as a choice has to be made to continue the process. Previous work tends to solve the problem by removing formulas that cause inconsistency, as in [Mousseau et al., 2003, Greco et al., 2008]. However, if all statements are certain, removing one seems counter-intuitive.

To address this issue, [Greco et al., 2008] proposes as an extension to associate a degree of credibility with each piece of information contained in the base. This idea has been developed in [Adam and Destercke, 2024], where belief bases are combined with reasoning under inconsistency. The paper proposes to represent beliefs within the scope of possibility theory. Possibility theory has been developed thanks to the preliminary work of Zadeh in [Zadeh, 1978]. It is built upon fuzzy sets, sets where each element has an ordinal priority degree associated. Within possibility theory, the degree of priority is the degree of credibility of each statement. For a complete overview of the possibility theory, we invite the reader to refer to [Dubois and Prade, 2015, Dubois and Prade, 2004].

As well as providing a representation of uncertainty, possibility theory is of interest when it comes to reasoning as it introduces logic tools to infer over a set of beliefs stratified by credibility, as in [Dubois et al., 1994]. As each statement is uncertain at a given degree, it is possible to attach a certainty degree to pieces of information deducted from the base, depending on which statement allow the deduction. This reinforces the trust the analyst has in the presented deductions. Furthermore, tools for reasoning

1. We use the neutral feminine for the DM

under inconsistency in possibilistic logic are developed in [Benferhat et al., 1999]. These tools have not been used in [Adam and Destercke, 2024] where they introduce numerical ones for reasoning. We believe that the semantics for reasoning under inconsistency from possibilistic logic are interesting, as they permit the definition of computationally good syntaxes providing certificates for the analyst. Hence, the aim of this article is to use the possibilistic reasoning framework with linear numerical models. The benefits of this combination are the appearance of the degree of credibility and the tools for reasoning provided by the logical framework.

A research gap also exists within possibility theory, where little attention was given to computational results. [Lang, 2000] proposes algorithms and computational results when the beliefs are expressed using propositional logic. There is a need to define syntaxes for possibility theory applied to beliefs expressed as numerical formulae and give computational results. These syntaxes should be created while keeping in mind the explanation, as in [Belahcene et al., 2017, Amoussou et al., 2023], process that plays a significant part in the trust that our analyst will place in our decisions. Even though this is not the current scope of our work, this is a natural extension.

Section 2 will present possibility theory and the semantics of possibilistic logic, as seen in the literature. In Section 3, we restrict our universe to beliefs expressed as linear inequalities. We develop new syntaxes for inferring and give computational results. In Section 4, we give a possible application of our logic of linear comparison, MCDA. Our proposal is discussed regarding the literature in Section 5. We end the paper with some conclusions in Section 6.

2 Possibilistic logic : semantics

Let us have Ω, the set of states of the world, with $\omega \in \Omega$, a state. A possibility distribution π is a mapping from Ω to $[0, 1]$. It represents the plausibility of each state of the world, the higher, the more plausible. $\pi(\omega) = 1$ means that the state is totally possible, $\pi(\omega) = 0$ that it is impossible.

A belief base is built on a set of formulae, each associated with a level of belief. We assume formulae belong to a set $\mathcal{F}$ called the *hypotheses class*, e.g. propositional formulae over a given language, or linear comparisons between numerical values. For $\phi \in \mathcal{F}$, we note $[\![\phi]\!] \subseteq \Omega$ the subset of ω that satisfies ϕ. We associate a value α with ϕ, taken in an ordinal set. It represents the priority of its associated formula, its credence, compared to the other formulae present in the possibilistic base Γ.

The degree of belief possess a clear semantics. It can be linked to imprecise probabilities as defined by [Walley, 1991, Dubois et al., 1993]. Given a formula ϕ, consider the lottery in which a gambler receives a unit gain if ϕ is satisfied and zero otherwise. α is defined as the maximum purchase price one would pay to enter this game. In particular, α is one iff the gambler is certain of a positive outcome, otherwise she would forfeit a sure gain or incur a sure loss.

The statement (ϕ, α) claims the worlds where ϕ is satisfied are fully possible and the world where ϕ is not are partially possible, at level $1 - \alpha$, as captured by Equation (1).

$$\pi_{(\phi_j,\alpha_j)}(\omega) = \begin{cases} 1 & \text{if } \omega \in [\![\phi_j]\!], \\ 1 - \alpha_j & \text{otherwise.} \end{cases} \tag{1}$$

When dealing with a conjunction of statements, we opt for the *minimal specificity* criterion where the possibility level is aggregated using the minimum operator. This allows to define the possibility distribution induced by a belief base Γ according to Equation (2).

$$\pi_\Gamma(\omega) = \min_{(\phi_j,\alpha_j)\in\Gamma} \pi_{(\phi_j,\alpha_j)}(\omega) \tag{2}$$

This definition extends the case where information is perfect : when levels are restricted to $\{0, 1\}$, a new statement ϕ prunes the set of possible world by asserting all worlds that do not satisfy ϕ are completely impossible. Observe that, when $\alpha_1 > \alpha_2$, the statement (ϕ, α_1) subsumes the statement (ϕ, α_2).

Example 1. *We have a bag. Inside the bag there could be nothing, a banana, an apple, or both. We have* $\Omega = \{ba, \bar{b}a, b\bar{a}, \bar{b}\bar{a}\}$. *We receive the information that* $(a \oplus b)$ *with a certainty of* 0.9. *Hence,* $\Gamma^1 = \{(a \oplus b, 0.9)\}$. *In other words, the necessity that there is exactly a fruit in the bag is superior or equal to* 0.9. Γ^1 *induces a new possibility distribution for all states of the world, as shown in Table 1.*

ω	π_{Γ^1}
ba	0.1
$\bar{b}a$	1
$b\bar{a}$	1
$\bar{b}\bar{a}$	0.1

TABLE 1 – Possibility distribution induced by Γ^1

The states of the world not satisfying the formula in Γ^1 *become less possible, as in Equation (1). However, states of the world satisfying the piece of information are still totally possible. If we consider* $\Gamma^2 = \{(a \oplus b, 0.9), (b, 0.7)\}$, *we have the possibility distribution of Table 2.*

ω	π_{Γ^2}
ba	0.1
$\bar{b}a$	0.3
$b\bar{a}$	1
$\bar{b}\bar{a}$	0.1

TABLE 2 – Possibility distribution induced by Γ^2

For $\bar{b}\bar{a}$ *not satisfying both formulas, we take the minimum possibility value between* $1 - 0.9$ *and* $1 - 0.3$, *as shown in Equation (2).*

If an ω does not satisfy ϕ_j, we give it a possibility value equal to $1 - \alpha_j$, as in Equation (1). Hence, if ϕ_j has a

strong credence, an ω not satisfying it will be deemed less possible. To compute the possibility distribution for Γ, we simply take the minimum value returned by each pair, as in Equation (2). Therefore, if an ω belongs to all the $[\![\phi_j]\!]$, its possibility value will be 1, it is totally possible. However, if such an ω does not exist, we say that the base is inconsistent. The level of inconsistency is evaluated as in Equation (3).

$$Inc(\Gamma) = 1 - \max_{\omega \in \Omega} \pi_\Gamma(\omega) \tag{3}$$

If the base is consistent, $Inc(\Gamma)$ is equal to 0. Otherwise, it is at least of value $1 - \alpha_j$ where α_j is the lowest value among the pairs (ϕ_j, α_j).

The handling of uncertainty in possibility theory sees three advantages. Firstly, it captures standard set theory, as a possibility is coherent whenever its maximum reaches one (therefore, multiple elements can have possibility one). This contrasts with probabilities, where the summation constraint means that making one element more plausible decreases the plausibility of some others, making it unable to capture set theory. Secondly, the coherence of the final result is not considered an axiom. This differs, e.g., from standard probabilities or even from lower previsions [Walley, 1991]. Thirdly, the user does not require to know how new numbers are computed, as the computations are done with an ordinal setting and the min operator.

Finally, we introduce the notion of α-cuts. These are subbases $\Gamma_{>\alpha}$ and $\Gamma_{\geq\alpha}$, defined in Equation (4).

$$\begin{aligned} \Gamma_{>\alpha} &= \{(\phi_j, \alpha_j) \in \Gamma \text{ s.t } \alpha_j > \alpha\} \\ \Gamma_{\geq\alpha} &= \{(\phi_j, \alpha_j) \in \Gamma \text{ s.t } \alpha_j \geq \alpha\} \end{aligned} \tag{4}$$

We simply do not consider pairs with an α_j (strictly or not) below the given threshold. This will prove useful when dealing with inconsistent bases, and allows for a concise characterization of the inconsistency level :

$$Inc(\Gamma) = \alpha \iff \Gamma_{>\alpha} \text{ is consistent but not } \Gamma_{\geq\alpha} \tag{5}$$

Example 2 (Example 1 continued)**.** *We now have* $\Gamma^3 = \{(a \oplus b, 0.9), (b, 0.7), (\bar{a} \wedge \bar{b}, 0.6)\}$. *Our analyst tells us that the necessity that there are no fruits inside the bag is at least* 0.6. *This makes the base inconsistent, and we have the possibility distribution of Table 3.*

ω	π_{Γ^3}
ba	0.1
$\bar{b}a$	0.3
$b\bar{a}$	0.4
$\bar{b}\bar{a}$	0.1

TABLE 3 – Possibility distribution induced by Γ^3

This base is inconsistent. If we solely consider the base without the degree of certainty, the analyst is affirming that there are fruits and there are no fruits in the bag. It is totally contradictory. Without degrees, to repair such an inconsistency, one would have to choose between either the first two pieces of information or the last. However, we have access to additional information with the degrees of certainty. A basic way to repair the belief base is to consider the formulae on the degree of inconsistency, as defined in Equation (3). As the most plausible world is $b\bar{a}$ *with degree 0.4, we have* $Inc(\Gamma^3) = 0.6$. *In other words, the strict* 0.6*-cut of* Γ, $(\Gamma^3)_{>0.6}$, *is consistent. It amounts to ignoring formulae with level 0.6 or lower, which actually yields* Γ^2.

With the basics of possibility theory clearly defined, we can now focus on the possibilistic inference of a formula ϕ from a possibility distribution induced by Γ, π_Γ. When information is certain (i.e., all αs are one), inferring on such a base is equivalent to *robust* (or *skeptical*) inference : checking that all the totally possible states of the world satisfy the formula we want to infer. The possibilistic inference semantics extends this process to imperfect information, with level in $[0, 1]$ and beliefs that are not necessarily consistent.

Definition 1. ([Dubois et al., 1994], Possibilistic inference) $: \Gamma \models_{pi} (\phi, \alpha)$ iff $\forall \omega, \pi_\Gamma(\omega) \leq \pi_{\{(\phi,\alpha)\}}(\omega)$

Example 3 (Example 1 continued)**.** *We consider* Γ^2. *It is possible to infer* $\bar{a}$, *i.e. the absence of an apple, from the base. The two possibilities distribution are in Table 4.*

ω	π_{Γ^2}	$\pi_{(\bar{a},\alpha)}$
ba	0.1	$1-\alpha$
$\bar{b}a$	0.3	$1-\alpha$
$b\bar{a}$	1	1
$\bar{b}\bar{a}$	0.1	1

TABLE 4 – Possibility distribution induced by Γ^2 and $\bar{a}$

With $\alpha = 0.7$, *for all possible states of the world, the possibility of having* $\bar{a}$ *is greater than the possibility of* Γ^2. *Hence, we can infer* $(\bar{a}, 0.7)$ *from* Γ^2. *In other words, from the facts we believe it is 0.9-certain there is one fruit in the bag and it is 0.7-certain there is a banana, we can deduce it is 0.7-certain there is no apple.*

Possibilistic inference is similar to what is done in other kinds of logic. For example, Definition 1 can be rewritten as a refutation. Instead of checking the inclusion of all totally possible ω in $[\![\phi]\!]$, it is equivalent to verify that the intersection between the totally possible ω and $[\![\neg\phi]\!]$ is empty. However, when the base is inconsistent ($Inc(\Gamma) > 0$), we can infer any formula at $\alpha = 1 - Inc(\Gamma)$ because the empty set is included in every set. At this point, it is necessary to remember the difference between inferring and deciding.

Inferring ϕ and $\neg\phi$ on an inconsistent base without degrees of certainty is problematic, as it does not come with tools helping to solve the conundrum. By contrast, the higher-order information provided by the valuation of beliefs with degrees of certainty can help to address the problem, as the

decision has to be made between (ϕ, α) and $(\neg\phi, \beta)$, a richer information. As decision is not the focus of this work, we leave this question open. Two new types of inference will be introduced, specially built to deal with the inconsistent case. We introduce the first semantics, the *non-trivial inference*, in Definition 2.

Definition 2. ([Dubois et al., 1994], Non-trivial inference) $\Gamma \models_{nt} (\phi, \alpha)$ iff $\pi_{\Gamma_{>Inc(\Gamma)}}(\omega) \leq \pi_{\{(\phi,\alpha)\}}(\omega)$

Equivalently, this can be written :

$$\Gamma \models_{nt} (\phi, \alpha) \text{ iff } \Gamma_{\geq\alpha} \text{is consistent and } \Gamma_{\geq\alpha} \models_{pi} (\phi, \alpha) \tag{6}$$

The semantics is based on the possibility degrees. We simply ignore the formulas under the inconsistency level. As $\Gamma_{>Inc(\Gamma)}$ is always consistent, inferring is safe. Here, inference and decision are equivalent, as it is not possible to infer ϕ and $\neg\phi$ on a consistent base. However, the disadvantage of this semantics is the drowning effect : all information below the inconsistency level is lost, even though some of it might be consistent with the curated base. Hence, the accepted inferences will be high in terms of certainty, but less pieces of information will be inferred. For a specific discussion of this issue, refer to [Benferhat et al., 1999, Section 3.2]. To avoid this, we introduce a last semantics to tackle inconsistency, the safe possibilistic inference in Definition 3.

Definition 3. ([Benferhat et al., 1999], Safe inference) $\Gamma \models_s (\phi, \alpha)$ iff $\exists\Gamma^* \subseteq \Gamma$ s.t $\max_\omega \pi_{\Gamma^*}(\omega) = 1$ and $\forall\omega, \pi_{\Gamma^*}(\omega) \leq \pi_{\{(\phi,\alpha)\}}(\omega)$

If a consistent subbase of Γ supports the inference of (ϕ, α), then this inference is deemed safe. In this case, inferring and deciding are different. Two different coherent subbases could allow to respectively infer ϕ and $\neg\phi$. In the literature, the quality of a subbase is measured by the minimum degree of certainty of one of its formulas. If we can infer ϕ and $\neg\phi$, deciding ϕ would require the best subbase for ϕ to have a strictly better rank than the best for $\neg\phi$. However, before turning to the decision mechanism, it is necessary to construct syntaxes for our inference process.

Now that a very brief definition of possibility theory and semantics for inference have been given, let us restrain our Ω so that we can give computational results and define a syntax for our semantics given in Definitions 1, 2 and 3.

3 The logic of linear comparison

3.1 Restricting to linear inequalities

Our aim is to use the expressivity and good computational properties of linear models while keeping the logical tools for reasoning over a base. To our knowledge, very few works in the field of possibility theory have covered this point. [Adam and Destercke, 2024] proposes to use possibility theory to address inconsistency in a MCDA setup, but does not use possibilistic tools to perform the inference.

From now on, we restrict Ω to the Cartesian product of domains defined over a continuous space. We have $\Omega = (\mathbb{R}_+)^n \setminus \{0\}$. Therefore, each $\omega \in \Omega$ is a non-null vector of n dimensions in $\mathbb{R}_+$. Without loss of generality, we consider that each belief base is composed of a number of statements and the n necessary statements with $\omega_i \geq 0$ with credence 1. We also restrict our class of hypotheses $\mathcal{F}$ to the *dual space* $\Omega^*_{\geq}$, so that each formula $\phi^{(j)}$ is associated to a n-dimensional vector $(\phi_1^{(j)}, \ldots, \phi_n^{(j)}) \in \mathbb{R}^n$ and to a linear inequality of the form given in Equation (7).

$$\sum_{i=1}^{n} \phi_i^{(j)} \omega_i \geq 0 \tag{7}$$

With $\mathcal{F}$ and Ω specified, it is possible to give computational results for the various forms of possibilistic inference semantics.

3.2 Computational results

According to Definition 1, inferring ϕ on a consistent possibilistic base is made by checking that the set of totally possible worlds is included in $[\![\phi]\!]$. This is equivalent to proving that no totally possible ω is in $\Omega \setminus [\![\phi]\!]$.
When Ω is a set of propositional variables and $\mathcal{F}$ the propositional formulae over them, checking whether a possibilistic base is consistent or not is DP-complete, and inference is computationally difficult [Lang, 2000]. With $\Omega = \mathbb{R}^n \setminus \{0\}$ and $\mathcal{F} = \Omega^*_{\geq}$, this check can be performed in polynomial time with linear programming, yielding low complexity results for the various inference problems.

Proposition 1. When $\Omega = {\mathbb{R}_+}^n \setminus \{0\}$ and $\mathcal{F} = \Omega^*_{\geq}$, checking whether a given possibilistic belief base is consistent is polytime.

Proof. Consider the linear program with decision variables in Ω consisting in maximizing $\phi^N := \sum_{i=1}^n \omega_i$ subject to all formulae in the base and the fundamental constraints $\omega_i \geq 0$. This linear program is always feasible (because the null vector satisfies all constraints), can be solved in polynomial time [Khachiyan, 1979, Karmarkar, 1984], and the deduction is valid iff the optimum is strictly positive. □

As a corollary, finding the inconsistency level of a belief base is polytime.

Corollary 1. When $\Omega = {\mathbb{R}_+}^n \setminus \{0\}$ and $\mathcal{F} = \Omega^*_{\geq}$, given a possibilistic belief base Γ, computing $Inc(\Gamma)$ is polytime.

Proof. Finding the inconsistency level can be performed with a binary search on α, by checking whether $\Gamma_{\geq\alpha}$ is consistent or not, as proposed in [Lang, 2000]. It requires $O(\log_2 |\Gamma|)$ calls to a polynomial algorithm, hence it is still polytime.

□

For our three semantics of interest, inference over a consistent belief base is polytime.

Corollary 2. When $\Omega = {\mathbb{R}_+}^n \setminus \{0\}$ and $\mathcal{F} = \Omega^*_{\geq}$, given a possibilistic belief base Γ, a formula ϕ and a level $\alpha \in [0,1]$, if $\Gamma_{\geq\alpha}$ is consistent, then deciding whether $\Gamma \models_X (\phi, \alpha)$ is polytime whatever $X \in \{pi, nt, s\}$.

Proof. For all three semantics, inferring (ϕ, α) amounts to checking whether $\Gamma_{\geq\alpha} \cup (-\phi, \alpha)$ is inconsistent. For $\models_{pi}$, it is direct via Corollary 1. As the base is consistent, it is also the case for $\models_{nt}$ and $\models_s$ because all subbases are consistent. □

Corollary 3. When $\Omega = {\mathbb{R}_+}^n \setminus \{0\}$ and $\mathcal{F} = \Omega^*_{\geq}$, given a possibilistic belief base Γ, a formula ϕ and a level $\alpha \in [0,1]$, deciding whether $\Gamma \models_{nt} (\phi, \alpha)$ is polytime.

Proof 1. *Consistent case in Corollary 2. For the inconsistent case, computing $Inc(\Gamma)$ is polytime. As $\Gamma_{>Inc(\Gamma)}$ is always consistent, inferring with $\models_{nt}$ on an inconsistent base is polytime.*

When considering *safe* possibilistic inference, the requirement to find a consistent subbase makes the problem more computationally demanding. Fortunately, Proposition 2 ensures that it remains in NP.

Proposition 2. When $\Omega = \mathbb{R}^n \setminus \{0\}$ and $\mathcal{F} = \Omega^*_{\geq}$, given a possibilistic belief base Γ, a formula ϕ and a level $\alpha \in [0,1]$, deciding whether $\Gamma \models_s (\phi, \alpha)$ is NP-complete.

Proof. Membership : given a subset Γ^* of the belief base, checking whether Γ^* is consistent is polytime. If it is, then checking whether $\Gamma^* \models_{pi} (\phi, \alpha)$ is polytime.
Hardness : reduction from VERTEX COVER [Karp, 1972]. From an instance $\langle V, E, K\rangle$ of VERTEX COVER, we build an instance $\langle \Gamma, (\phi, \alpha)\rangle$ of SAFE POSSIBILISTIC INFERENCE as follows. The parameter set is $\Omega := (R_+)^{E \cup V \cup \{\alpha,\beta\}}$. For each edge e and vertex $u \in e$, we denote ϕ^u_e the formula $\omega_e + \omega_\alpha \leq \omega_u$, and Γ the possibilistic belief base containing all these formulae with credence 1. Let $\phi^\#$ the formula $\sum_{v\in V} \omega_v + \omega_\beta \leq K\omega_\alpha$, and $\phi : \sum_{e\in E} \omega_e + \omega_\beta \leq K\omega_\alpha$. We claim $\Gamma, (\phi^\#, 1) \models_t (\phi, 1)$ iff there is a vertex cover of (V, E) of cardinality $\leq K$.
Indeed, suppose $U \subseteq V$ covers E with $|U| \leq K$, and consider the sub-base Γ_U containing all formulae ϕ^u_e for $e \in E$ and $u \in e \cap U$, with credence 1. First, the subbase $\Gamma_U \cup \{(\phi^\#, 1)\}$ is consistent, because the interpretation where $\omega_e = 0$ for all edges in E, $\omega_u = \omega_\alpha$ for all vertices in U, $\omega_v = 0$ for all vertices in $V \setminus U$ and $\omega_\beta = 0$ satisfies all its formulae. Second, it allows to deduce ϕ with credence 1, because, for every interpretation of $\omega \in \Omega$ satisfying both Γ_U and $\phi^\#$, we have, by summation of all comparisons in Γ_U that $\sum_{e\in E} \omega_e \leq \sum_{u\in U}(\omega_u - \omega_\alpha) = \sum_{u\in U} \omega_u - |U|\omega_\alpha$. Moreover $\sum_{u\in U} \omega_u \leq \sum_{v\in V} \omega_v$ because all ω_u are non-negative and $U \subseteq V$. Thus, $\phi^\#$ yields $\sum_{e\in E} \omega_e + \omega_\beta \leq \sum_{v\in V} \omega_v + \omega_\beta \leq K\omega_\alpha$ and ϕ holds in every possible world.
Reciprocally, suppose there is a sub-base $\Gamma^* \subset \Gamma \cup \{(\phi^\#, 1)\}$ which is consistent and allows to derive ϕ. Suppose $\phi^\# \notin \Gamma^*$: there is no constraint bounding ω_β from above, and starting from any feasible model and letting $\omega_\beta \to +\infty$ yields a feasible model that does not satisfy ϕ at some point : a contradiction. For each edge $e \in E$, if Γ^* contained no formula ϕ^u_e for some $u_e \in e$, there would be no constraint bounding ω_e from above, leading to a similar contradiction. Thus, the set $U := \bigcup_{e\in E}\{u_e\}$ covers E, and because $\phi^\#$ holds, it cannot have cardinality above K. □

At this point, we have not given any general results concerning the baseline semantics of possibilistic inference. Observe this semantics still holds if the base is inconsistent, allowing to infer *any* formula (and also its negation) at some level $\alpha > 0$. Computationally, this is a much more difficult problem.

Proposition 3. When $\Omega = {\mathbb{R}_+}^n \setminus \{0\}$ and $\mathcal{F} = \Omega^*_{\geq}$, given a possibilistic belief base Γ, a formula ϕ and a level $\alpha \in [0,1]$, deciding whether $\Gamma \models_{pi} (\phi, \alpha)$ is NP-hard.

Sketch of proof. Adaptation of the reduction from VERTEX COVER put forward in the proof of Proposition 2. Let $e^* = \{u^*, v^*\}$ an arbitrary edge in E, and consider a slightly modified belief base Γ', where the formulae $\phi^{u^*}_{e^*}$ and $\phi^{v^*}_{e^*}$ now have a slightly lower credence 0.9 instead of 1. The formula ϕ can be inferred with $\models_{pi}$ at level 0.9 iff there is a vertex cover, and at level 1 iff there is a vertex cover without e^*. □

The question of membership in NP is left open.

3.3 Certified inference

The last section proposed a calculus for possibilistic logic based on linear programming. This is a satisfying solution from the computational point of view, allowing tractable inference, even though NP-completeness is hardly scalable, it is sufficient to process small instances. Nevertheless, linear programming is certainly not human-friendly, and relying on a solver to check the validity of inference does not seem to fulfil the requirement of transparency for trustworthy AI. Thus, we propose to support inference with evidence allowing to check its adequacy. The main tool in this endeavour is Farkas'lemma : a system of linear inequalities over $\mathbb{R}^n$ $\phi^1 \geq 0, \ldots, \phi^m \geq 0$ entails $\phi \geq 0$ if, and only if, ϕ is a convex combination of the ϕ^k. Thus, deduction made under the assumptions of Corollary 2 can be supported with a certificate proving its soundness.

Reasoning w.r.t. safe inference. We propose a certificate for safe inference.

Definition 4. When $\Omega = \mathbb{R}^n \setminus \{0\}$, a primal/dual (or p/d)-certificate is an ordered pair (ω^*, λ^*) where $\omega^* \in \Omega$ and λ^* is a tuple of non-negative numbers. Given a possibilistic belief base $\Gamma = \{(\phi^{(1)}, \alpha_1), \ldots, (\phi^{(m)}, \alpha_m)\}$, a formula ϕ and a level $\alpha \in [0,1]$, we write $\Gamma \vdash_s^{(\omega^*,\lambda^*)} (\phi, \alpha)$ when all following conditions are satisfied :

i) the length of λ^* is equal to m, the cardinality of Γ ;
ii) for all $1 \leq k \leq m$, if $\lambda^*_k > 0$ then $\phi^{(k)}(\omega^*) \geq 0$;
iii) for all $1 \leq k \leq m$, if $\lambda^*_k > 0$ then $\alpha_k \geq \alpha$; and

iv) $\phi \geq \sum_{k=1}^{m} \lambda_k^* \phi^{(k)}$.

We write $\Gamma \vdash_s (\phi, \alpha)$ when there is a p/d-certificate (ω^*, λ^*) such that $\Gamma \vdash_s^{(\omega^*, \lambda^*)} (\phi, \alpha)$.

Proposition 4. Syntactic deduction $\vdash_s$ is sound and complete w.r.t. $\models_s$.

Proof. Define $\Gamma^* := \{(\phi^{(k)}, \alpha_k) : \lambda_k^* > 0\}$. Condition ii) ensures the consistency of Γ^*, as witnessed by the totally possible ω^*. Condition iii) ensures the credence level of Γ^* is at least α, warranting inference at this level. Condition iv) ensures the conclusion ϕ is in the convex span of the formulae in Γ^*. All these conditions are necessary for safe possibilistic inference, and together they are sufficient. □

Reasoning w.r.t. non-trivial inference. There are two ways to perform non-trivial inference :

— either to compute the inconsistency level $Inc(\Gamma)$ beforehand; maybe certify it with a primal/dual certificate (ω^*, λ^*) such that $\sum_k \lambda^* \phi^k = -\sum_k \omega_k$ (thus $\Gamma_{>Inc(\Gamma)}$ is consistent) and for all $1 \leq k \leq m$, if $\alpha_k > Inc(\Gamma)$ then $\phi^{(k)}(\omega^*) \geq 0$, and if $\alpha_k \leq Inc(\Gamma)$ then $\lambda_k = 0$ (thus $\Gamma_{\geq Inc(\Gamma)}$ is inconsistent); then perform inference with the consistent base (maybe supporting it with a dual certificate).

— or to perform safe inference restricted to a stratified consistent subbase $\Gamma^* \subseteq \Gamma_{>Inc(\Gamma)}$.

Definition 5. Under the same assumption as Definition 4, we write $\Gamma \vdash_{nt}^{(\omega^*, \lambda^*)} (\phi, \alpha)$ when conditions i), ii), iii) and iv) are satisfied, as well as :

v) for all $1 \leq k \leq m$, if $\alpha^k \geq \alpha > Inc(\Gamma)$ then $\phi^{(k)}(\omega^*) \geq 0$.

We write $\Gamma \vdash_{nt} (\phi, \alpha)$ when there is a p/d-certificate (ω^*, λ^*) such that $\Gamma \vdash_{nt}^{(\omega^*, \lambda^*)} (\phi, \alpha)$.

Proposition 5. Syntactic deduction $\vdash_{nt}$ is sound and complete w.r.t. $\models_{nt}$.

Proof. This is corollary of the fact non-trivial inference is simply safe inference restricted to the case where the subbase $\Gamma^* \subseteq \Gamma_{>Inc(\Gamma)}$. Condition v) enforces this. □

As a direct consequence from Propositions 4 and 5, $\vdash_s$ and $\vdash_{nt}$ are not harder than their respective semantics, allowing NP-complete and polynomial-time inference.

Reasoning with possibilistic inference . Defining a concise certificate for possibilistic inference remains an open question, conjectured to be equivalent to asserting whether deciding $\models_{pi}$ belongs to NP.

3.4 Argued consequence and syntaxes

$\models_s$ originates from [Benferhat et al., 1999]. In the paper, a syntax for safe inference is defined. It differs from Definition 4 as it was not specifically built for linear numerical bases. However, even though the syntaxes are different, their properties are similar.

The generic definition of an argued consequence is given in Definition 6. A subbase Γ^* is an argued consequence for a formula ϕ if it respects the following properties. We note $\vdash_{ac}$ this syntax.

Definition 6. ([Benferhat et al., 1999], Argued consequence)
$$\begin{cases} 1. & Inc(\Gamma^*) = 0, \\ 2. & \Gamma^* \vdash_{pi} \phi \text{ (relevance)}, \\ 3. & \forall \phi_j \in \Gamma^*, \Gamma^* - \{\phi_j\} \nvdash \phi \quad \text{(economy)}, \end{cases}$$

The relevance and consistency properties are also contained in $\vdash_s$, as in Definition 4. Relevance as Farkas' criterion of infeasibility (through the decomposition), and consistency as a consistent subbase is searched. However, the minimality property is implicit in $\vdash_s$. We have no guarantee that our certificate will be minimal. Therefore, it is not possible to enounce that the two syntaxes are exactly equivalent, but they share two strong properties, the relevance and the consistency of the certificate.

Hence, we have defined three different semantics and syntaxes. One general for the consistent case and two allowing to make non-trivial and safe decision from an inconsistent base. They all have interesting computational results and were linked with Farkas' Lemma, in order to be able to provide a certificate to our user. This point will be developed in the next section, accompanied by illustrations of notions that have been defined in the two last sections. It will be presented through the MCDA example.

4 A possible application : MCDA

4.1 MCDA in a nutshell

In Multi-Criteria Decision Aiding, our aim is to aid our user (DM) to choose between several alternatives evaluated on several criteria. For example, she might have to choose a hotel in Dijon and her criteria are the price, the number of stars, the distance to the conference center, and the presence/absence of breakfast. We want to learn her preferences. A DM might prefer a cheap hotel while another DM might want an expensive one close to the commute. The common way to model preferences is to consider a weight vector ω defined over n continuous finite domains, n being the number of criteria. Numerous different models exist, from the weighted sum, to Choquet's integral and the OWA/WOWA. For an extensive review, refer to [Bourdache, 2020]. To conduct the elicitation process, we ask her questions in the form of linear inequalities.

Example 4. *Let us take an example where we have to help the DM choose between several alternatives, each described by three criteria. For each criterion, the higher the score the better. The alternatives are presented in Table 5. For illustration purpose, it will be considered that the DM's preferences can be modelled through a weighted sum, one of the simplest numerical models. It is parameterized by a vector* $\omega \in \Omega = (\mathbb{R}_+)^3$. *It represents the preferences of the DM as follows : given two alternatives* x, y, x *is preferred to* y, *denoted* $x \succsim_\omega y$ *iff*

Alternative	Criterion 1	Criterion 2	Criterion 3
a	10	2	6
b	8	3	7
c	6	6	4
d	4	9	5

TABLE 5 – Alternatives comparison

$$\sum_{i=1}^{3} x_i\omega_i \geq \sum_{i=1}^{3} y_i\omega_i \tag{8}$$

Given some preference information under the form of a consistent belief base Γ, it is customary to write $x \succsim_\Gamma y$ if Equation (8) holds for all ω compatible with all beliefs. We shall keep this notation, but we assume each comparative statement of Γ comes with an assessment of its credence, and in turn we provide an indication of credence for inferred beliefs.
The DM states she prefers c to a, a belief represented by $6\omega_1 + 6\omega_2 + 4\omega_3 \geq 10\omega_1 + 2\omega_2 + 6\omega_3$ or, more succinctly, the formula $-4\omega_1 + 4\omega_2 - 2\omega_3 \geq 0 \in \Omega^*_{\geq}$. Besides comparative statements, more general beliefs can be expressed with linear comparisons. For instance, given criteria are expressed on the same scale, "criterion 2 is more important than criterion 1" is represented by $\omega_2 \geq \omega_1$.

We perform an active and incremental elicitation process. Examples from the literature can be found in [White et al., 1984, Greco et al., 2008, Jacquet-Lagreze and Siskos, 1982, Adam and Destercke, 2024]. Our aim is to fit the preferences of the DM by restricting the parameters of our numerical model. When the space of totally possible ω is small enough, we are able to infer on the base by using the syntaxes we defined in the last section.

4.2 MCDA and logic of linear comparison

One can see that the MCDA framework we just defined is very close to our logic of linear inequalities. However, several points need to be clarified.

Firstly, very few works in the field of MCDA have tried to associate a credence degree with each piece of information we use to infer. This idea is evoked at the end of [Greco et al., 2008] but, to our knowledge, the only paper that has tried to develop this idea is [Adam and Destercke, 2024]. This is due to a central question around this work ; is it relevant to ask our DM for α_j ? This question has already been addressed in Section 2. In the MCDA setup, it is also a maximum purchase price for a ticket for an uncertain lottery. The limitation of current robust elicitation frameworks, as defined in [Greco et al., 2008], is that they assume certain knowledge. We believe that this representation is too naive. This is why we advocate that a rational agent would rather choose a maximum purchase price below the winning prize. We extend the robust approach by allowing the agent to express her doubts on the information she gives us, instead of blindly believing all that she says.

The rest of this section will consist of an example using the tools we have defined in the 2 last sections.

4.3 An example to illustrate the developed tools

We continue Example 4. Figure 1 represents Ω [2].

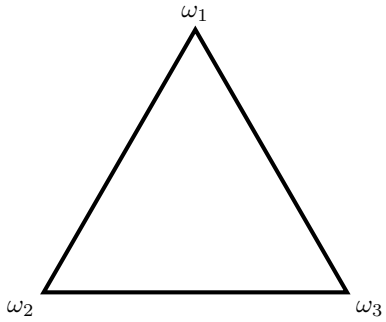


FIGURE 1 – Barycentric representation of the set of worlds Ω.

In this representation, $\Gamma^0 = \{(\omega_1 \geq 0, 1), (\omega_2 \geq 0, 1), (\omega_3 \geq 0, 1)\}$. For a reminder, these are the necessary statements on each ω_i. She gives us a preference :

$$(\phi^{(1)} := \omega_1 \geq \omega_2, \alpha_1 := 0.9)$$

In other words, she thinks that criterion 1 is more important than criterion 2 with a certainty of 0.9. We have $\Gamma^1 = \{(\omega_1 \geq 0, 1), (\omega_2 \geq 0, 1), (\omega_3 \geq 0, 1), (\omega_1 \geq \omega_2, 0.9)\}$. According to our interpretation of such a pair and Equation (1), Figure 2 represents the updated space of parameters.

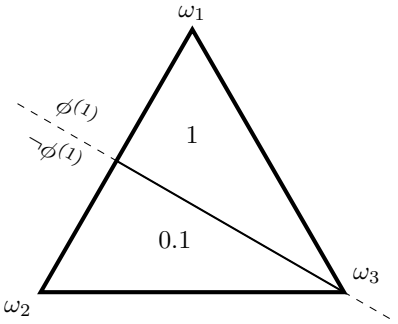


FIGURE 2 – The possibility distribution π_{Γ^1}.

The upper half-space ($[\![\phi^{(1)}]\!]$) is constituted of all the ω such that $\omega_1 \geq \omega_2$, therefore with a $\pi_\Gamma(\omega) = 1$. The lower one is constituted of all the ω not satisfying $\phi^{(1)}$, hence with a possibility of $1 - 0.9$. She adds a second statement.

$$(\phi^{(2)} := \omega_3 \geq \omega_1, \alpha_2 := 0.8)$$

2. Without loss of generality, a normalization constraint ($\omega_1 + \omega_2 + \omega_3 = 1$) is added to permit the use of barycentric coordinates for an easier representation

Hence, $\Gamma^2 = \{(\omega_1 \geq 0, 1), (\omega_2 \geq 0, 1), (\omega_3 \geq 0, 1), (\omega_1 \geq \omega_2, 0.9), (\omega_3 \geq \omega_1, 0.8)\}$, we obtain the possibility distribution over Ω shown in Figure 3. Due to the minimum in Equation (2), for the left part of the triangle (i.e. ω not satisfying any of the two constraints), we take the minimum value of possibility.

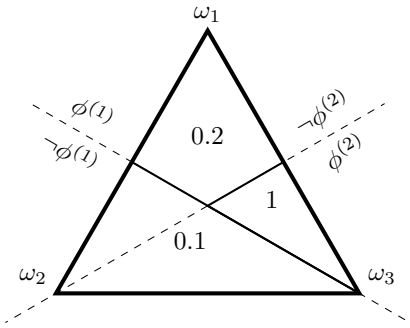


FIGURE 3 – The possibility distribution π_{Γ^2}.

Γ^2 is consistent, as witnessed e.g. by the primal certificate $\omega = (\frac{1}{3}, \frac{1}{6}, \frac{1}{2})$. To verify whether $a \succeq_{\Gamma^2} d$ can be inferred, $\phi^{a \succ d} := (6\omega_1 - 7\omega_2 + \omega_3 \geq 0)$ must hold for all totally possible ω. First, the proof by refutation is drawn in Figure 4. The half-space corresponding to the negation of $\phi^{a \succ d}$, with a certainty of 1, is added to Γ^2.

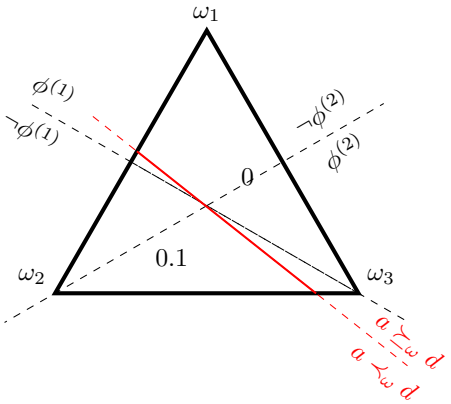


FIGURE 4 – The possibility distribution induced by $\Gamma^2 \cup (\neg\phi^{a \succ d}, 1)$.

The distribution in Figure 4 is inconsistent as no ω belongs to all half-spaces. This semantically proves that our DM prefers alternative a over alternative d, with a certainty of 0.8, because the maximum of π_{Γ^2} over the worlds where $d \succ a$ is 0.2. We establish this preference syntactically by providing a dual certificate. The inequality system is presented in Equation (9).

$$\Gamma^2 = \begin{cases} \omega_1 \geq 0 & (d^{(1)}1, 1) \\ \omega_2 \geq 0 & (d^{(2)}, 1) \\ \omega_3 \geq 0 & (d^{(3)}, 1) \\ \omega_1 - \omega_2 \geq 0 & (\phi^{(1)}, 0.9) \\ \omega_3 - \omega_1 \geq 0 & (\phi^{(2)}, 0.8) \end{cases} \tag{9}$$

Observe $\phi = 7\phi^{(1)} + \phi^{(2)}$, which allows to deduce $\phi^{a \succ d}$ from Γ^2 with certainty $min_{i \text{ s.t} \lambda_i \neq 0} \alpha_j = 0.8$.
Let us add a new $\phi^{(3)}$ to our base that breaks consistency.

$$(\phi^{(3)} = c \succeq_\omega b, \alpha_3 := 0.7)$$
$$\text{with } \phi^{(3)} \Leftrightarrow -2\omega_1 + 3\omega_2 - 3\omega_3 \geq 0$$

We now consider $\Gamma_3 := \Gamma_2 \cup \{(\phi^{(3)}, 0.7\}$.
The possibility distribution induced by Γ^3 is displayed on Figure 5.

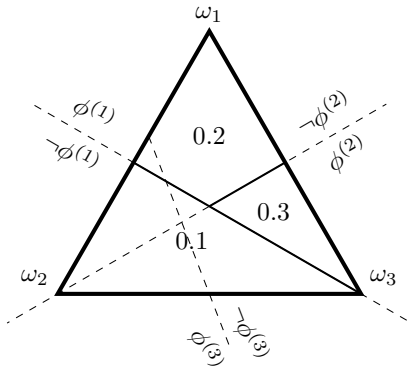


FIGURE 5 – The possibility distribution π_{Γ^3}.

Γ^3 is inconsistent, no $\omega \neq 0$ is a solution of its system. Observe that the convex combination $7\phi^{(1)} + 4\phi^{(2)} + 2\phi^{(3)}$ ensures $-\omega_1 - \omega_2 - 2\omega_3 \geq 0$, while the non-negativity of $\omega_1, \omega_2, \omega_3$ entails $\omega_1 + \omega_2 + 2\omega_3 \geq 0$. Both cannot be satisfied if ω is non null. We illustrate how inference based on different semantics produces different outcomes.
Possibilistic inference $\models_{pi}$ allows to infer *any* formula ϕ from Γ_3, but with a level $\alpha = 1 - \max_{\neg\phi} \pi_{\Gamma^3} \in \{0.1, 0.2, 0.3\}$. This is a much more nuanced version of the principle of explosion of classical logic which puts all formulae on the same level.
Non-trivial inference $\models_{nt}$ prescribes to ignore the less certain stratum at level 0.7 which provokes inconsistency. Thus, from this viewpoint, Γ^3 is equivalent to its safe strata Γ^2. Observe that, if beliefs compatible with Γ^2, but no more certain than 0.7 were present in the base, they would have been "drowned".
Unsafe inference could be performed by leveraging dual certificates but ignoring the requirement of a primal certificate ensuring local consistency. Indeed, define $\phi := (a \succeq_\omega d) \Leftrightarrow (6\omega_1 - 7\omega_2 + \omega_3 \geq 0)$ and observe $\phi = 10\phi^{(1)} + 2\phi^{(2)} + \phi^{(3)} + 2\omega_3$, allowing to *unsafely* derive ϕ at level $\alpha = 0.7$. However, the basis of this certificate is $\Gamma^* = \Gamma^3$ and is inconsistent, undercutting the ar-

gument supporting ϕ : in every world where $\phi^{(1)}$, $\phi^{(1)}$ and $\phi^{(3)}$ hold, ϕ holds, but there is none.
Safe inference allows to make deductions based on any (maximally) consistent subbase of Γ^3 : either Γ^2 by ignoring $\phi^{(3)}$, or the bases obtained by ignoring respectively $\phi(1)$ or $\phi^{(2)}$. This is much more versatile than non-trivial inference, at the cost of solving a NP-complete problem.
In this section, the logic of linear comparison has been illustrated through MCDA. We applied our syntaxes to infer on a belief base in the context of the MCDA problem. However, other problems solved by numerical models could also be used within this framework, such as scheduling. This idea is not recent, such as [Dubois et al., 2003] gave an overview on how possibility theory can be applied to scheduling. This is left for further research, such as the decision procedure.

5 Related works

5.1 Possibility theory

Quantitative Possibility Theory, as defined in [Dubois and Prade, 1998], is different from what we want to achieve in this work. QPT defines the semantics of another possibility theory, where α is not defined over an ordinal set but over a continuous domain, as in probability theory.

To our knowledge, the only example of work combining possibility theory and numerical preferences (except [Adam and Destercke, 2024]), is [Kaci and Prade, 2008]. The fundamental difference between our approaches is that, instead of considering the priority as a degree of certainty, they choose to consider it as a degree of intensity.

Computational results have not been the main focus in possibilistic logic, except for [Lang, 2000] and papers on combinatorial logic. In this paper, results have been given on numerical linear logic, results absent from the literature.

5.2 Numerical models

The proposed extension in [Greco et al., 2008] extends the robust approach. In some papers as [Greco et al., 2008], the hypothesis is that the DM never commits any mistake. Even if we drop this hypothesis, no other work has tried to associate a degree of certainty to each piece of information.

6 Conclusion

In this paper, a theoretical framework combining logical frameworks and numerical models has been presented. Our main objective is to benefit from their respective qualities. Numerical models, in our case linear inequalities, are interesting from a computational point of view. A logical framework provides clear semantics and tools to infer, even in case of inconsistency. We chose to use Possibility Theory, as it is a formalism to represent uncertainty that comes with several good properties, as the conservation of the bounds value during the elicitation process thanks to the minimum specificity principle and its non-necessity to normalize the distribution of possibility at every step.

Semantics and syntaxes for inferring over a base built out of formulas and their associated degree of certainty have been defined : a polytime semantics and syntax to infer on a consistent base, another polytime couple to infer on an inconsistent base but with the downside that several pieces of information are not considered, and a last one to infer on an inconsistent base, which is at least NP-hard. All of these syntaxes have been linked to Farkas' criterion of contradiction, as used in the explanation literature. We believe that an analyst would be more confident in the decision returned by the elicitation if each of them was associated with a degree of certainty. Furthermore, Farkas' provides a certificate of infeasibility, which can be seen as an early form of explanation.

We finally gave a possible application to our logic of linear combination, in the world of MCDA. It was necessary to give a justification for the existence of the degree of uncertainty associated with each formula. We believe that they can be seen as a maximum purchase price in a game when the agent wins if the information she gives is proved to be correct. Other examples could have been investigated, such as fuzzy scheduling.

Future works include searching for new syntaxes to infer despite the inconsistency. Another peculiar point of interest is the decision process in case of inconsistency. It may be linkable to bipolar argumentation, as defined in [Amgoud et al., 2008, Al Anaissy, 2024]. Finally, exploring the explanation process for a recommendation coming from a belief base is an important subject.

Remerciements

Armand Gaudiller and Khaled Belahcène are supported by the chair "Explainable artificial intelligence for the future of Industry" funded by the ANR.

Références

[Adam and Destercke, 2024] Adam, L. and Destercke, S. (2024). Handling inconsistency in (numerical) preferences using possibility theory. *Information Fusion*, (103) :102089.

[Al Anaissy, 2024] Al Anaissy, C. (2024). *Principles and Practices for Bipolar Semantics and Impact Measures in Computational Argumentation*. PhD thesis, Université d'Artois.

[Amgoud et al., 2008] Amgoud, L., Cayrol, C., Lagasquie-Schiex, M.-C., and Livet, P. (2008). On bipolarity in argumentation frameworks. *International Journal of Intelligent Systems*, 23(10) :1062–1093.

[Amoussou et al., 2023] Amoussou, M., Belahcène, K., Labreuche, C., Maudet, N., Mousseau, V., and Ouerdane, W. (2023). Questionable stepwise explanations for a robust additive preference model. *International Journal of Approximate Reasoning*, page 108982.

[Belahcene et al., 2017] Belahcene, K., Labreuche, C., Maudet, N., Mousseau, V., and Ouerdane, W. (2017). Explaining robust additive utility models by sequences of preference swaps. *Theory and Decision*, (82) :151–183.

[Benferhat et al., 1999] Benferhat, S., Dubois, D., and Prade, H. (1999). An overview of inconsistency-tolerant inferences in prioritized knowledge bases. *Fuzzy Sets, Logic and Reasoning about Knowledge*, (15) :395–417.

[Bourdache, 2020] Bourdache, N. (2020). *Élicitation incrémentale des préférences pour l'optimisation multi-objectifs : modèles non-linéaires, domaines combinatoires et approches tolérantes aux erreurs*. PhD thesis, Sorbonne Université.

[Ciomek et al., 2017] Ciomek, K., Kadziński, M., and Tervonen, T. (2017). Heuristics for selecting pair-wise elicitation questions in multiple criteria choice problems. *European Journal of Operational Research*, 262(2) :693–707.

[Dubois et al., 2003] Dubois, D., Fargier, H., and Fortemps, P. (2003). Fuzzy scheduling : Modelling flexible constraints vs. coping with incomplete knowledge. *European Journal of Operational Research*, 147(2) :231–252.

[Dubois et al., 1993] Dubois, D., Godo, L., Mántaras, R., and Prade, H. (1993). Qualitative reasoning with imprecise probabilities. *J. Intell. Inf. Syst.*, 2 :319–363.

[Dubois et al., 1994] Dubois, D., Lang, J., and Prade, H. (1994). Possibilistic logic. In Gabbay, D. M., Hogger, C. J., Robinson, J. A., and Nute, D., editors, *Handbook of Logic in Artificial Intelligence and Logic Programming*, volume 3, pages 439–513. Oxford Univ. Press.

[Dubois and Prade, 1998] Dubois, D. and Prade, H. (1998). Possibility Theory : Qualitative and Quantitative Aspects. In Smets, P., editor, *Quantified Representation of Uncertainty and Imprecision*, volume 1 of *Handbook of Defeasible Reasoning and Uncertainty Management Systems book series (HAND)*, pages 169–226. Kluwer Academic Publishers.

[Dubois and Prade, 2004] Dubois, D. and Prade, H. (2004). Possibilistic logic : a retrospective and prospective view. *Fuzzy Sets and Systems*, (144) :3–23.

[Dubois and Prade, 2015] Dubois, D. and Prade, H. (2015). Possibility theory and its applications : Where do we stand ? *Springer handbook of computational intelligence*, page 31–60.

[Greco et al., 2008] Greco, S., Mousseau, V., and Słowinski, R. (2008). Ordinal regression revisited : Multiple criteria ranking using a set of additive value functions. *European Journal of Operational Research*, (191) :416–436.

[Jacquet-Lagreze and Siskos, 1982] Jacquet-Lagreze, E. and Siskos, J. (1982). Assessing a set of additive utility functions for multicriteria decision-making, the uta method. *European Journal of Operational Research*, 10(2) :151–164.

[Kaci and Prade, 2008] Kaci, S. and Prade, H. (2008). Mastering the processing of preferences by using symbolic priorities in possibilistic logic. In *Proc. ECAI 2008*, pages 376–380.

[Karmarkar, 1984] Karmarkar, N. (1984). A new polynomial-time algorithm for linear programming. *Combinatorica*, 4(4) :373–396.

[Karp, 1972] Karp, R. (1972). Reducibility among combinatorial problems. In Miller, R. and Thatcher, J., editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press.

[Khachiyan, 1979] Khachiyan, L. (1979). A polynomial algorithm for linear programming. *Doklady Akademii Nauk SSSR*, 224(5) :1093–1096.

[Lang, 2000] Lang, J. (2000). Possibilistic logic : complexity and algorithms. In Kohlas, J. and Moral, S., editors, *Handbook of Defeasible Reasoning and Uncertainty Management Systems*, pages 179–220. Springer-Science + Buisness Media, B.V.

[Mousseau et al., 2003] Mousseau, V., Figueira, J., Diás, L., Gomes da Silva, C., and ao Clímaco, J. (2003). Resolving inconsistencies among constraints on the parameters of an mcda model. *European Journal of Operational Research*, 147(1) :72–93.

[Walley, 1991] Walley, P. (1991). *Statistical Reasoning with Imprecise Probabilities*. Chapman & Hall.

[White et al., 1984] White, C. C., Sage, A. P., and Dozono, S. (1984). A model of multiattribute decisionmaking and trade-off weight determination under uncertainty. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-14 :223–229.

[Zadeh, 1978] Zadeh, L. (1978). Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems*, 1 :3–28.

Session 3 : MDP et décision séquentielle

Complexité des stratégies maxmin pures dans les jeux sous forme extensive à deux joueurs*

Junkang Li[1,2], Bruno Zanuttini[2], Véronique Ventos[1]

[1] NukkAI, Paris, France
[2] Université Caen Normandie, ENSICAEN, CNRS, Normandie Univ, GREYC UMR6072, F-14000 Caen, France

junkang.li@nukk.ai, bruno.zanuttini@unicaen.fr, vventos@nukk.ai

Résumé

Nous nous intéressons au calcul des stratégies pures et robustes pour les jeux sous forme extensive. Pour ce problème, nous avons effectué une analyse complète de sa complexité en fonction du degré d'information imparfaite des joueurs, et la même analyse sous trois cadres orthogonaux : les jeux représentés explicitement par leur arbre de jeu ; les jeux représentés de façon compacte par leurs règles ; les jeux dans lesquels le choix de stratégie des adversaires est restreint à un ensemble connu de modèles d'opposants.

Mots-clés

Théorie des jeux, complexité algorithmique, système multi-agents.

1 Jeux sous forme extensive

Les jeux sous forme extensive (*extensive-form games*, ou EFGs ci-après) représentent une interaction séquentielle entre plusieurs agents. Un EFG est défini par un arbre muni d'informations supplémentaires, comme le propriétaire des nœuds internes ou la récompense pour chaque joueur aux feuilles. Un tel jeu commence à la racine de son arbre, et se poursuit ainsi : pour chaque nœud interne, son propriétaire choisit un enfant de ce nœud, ce jusqu'à ce qu'une feuille soit atteinte ; chaque joueur reçoit alors une récompense en fonction de la feuille atteinte. Dans un EFG, une *stratégie pure* d'un joueur est une application qui associe chacun de ses nœuds à un enfant de ce nœud.

Il se peut aussi qu'un joueur dans un jeu ne possède pas d'information parfaite. Par exemple, un joueur peut ne pas observer tous les choix effectués par les autres joueurs dans le passé. Ce joueur peut alors ignorer où il est dans l'arbre lorsqu'il doit prendre une décision. Une telle situation se modélise en partitionnant les nœuds d'un joueur en des *ensembles d'information*, et en imposant que ce joueur doive prendre la même action à tous les nœuds dans le même ensemble d'information. Intuitivement, pendant le déroulement d'un tel jeu, lorsqu'un joueur doit choisir une action, il est seulement informé de l'ensemble d'information auquel le nœud courant appartient, mais pas du nœud lui-même. Si les ensembles d'information d'un joueur sont tous des singletons, on dit que ce joueur a une *information parfaite* ; sinon, il a une *information imparfaite*. Strictement entre ces deux notions se situe la notion nommée *mémoire parfaite* (*perfect recall*), qui signifie intuitivement qu'un joueur se souvient toujours des informations qu'il a reçues.

Il est également possible de modéliser explicitement les facteurs de hasard dans un EFG, en attribuant certains nœuds internes à un joueur spécial dénommé *Nature*. À chacun de ses nœuds (nommés *nœuds de hasard*), Nature choisira un successeur de façon probabiliste, selon une distribution de probabilités connue de tous les joueurs du jeu. Selon l'existence ou non de nœuds de hasard, un jeu est appelé un *jeu de hasard* ou un *jeu sans hasard*.

2 Cadre du travail

Nous nous focalisons sur les EFGs à deux joueurs, dénommés respectivement MAX et MIN. Cette étude concerne la valeur *maxmin pure*, c'est-à-dire la plus grande récompense dont MAX peut s'assurer en jouant l'une de ses stratégies pures, peu importe ce que fera MIN. Maxmin est donc une notion de robustesse. Notre travail consiste à établir la complexité de trouver une borne inférieure de la valeur maxmin pure pour les EFGs ; on nomme le problème de décision associé PURE MAXMIN.

Nos contributions principales sont les suivantes. (i) Nous avons établi la complexité de PURE MAXMIN sous trois cadres orthogonaux : les jeux représentés explicitement par leur arbre de jeu ; les jeux représentés de façon compacte par leurs règles de jeu ; les jeux dans lesquels le choix de stratégie des adversaires est restreint à un ensemble connu de modèles d'opposants. (ii) Nous avons rassemblé des résultats éparpillés à travers la littérature et complété les trous parmi eux. (iii) Nous présentons tous les résultats d'une manière cohérente, avec des preuves rigoureuses et des références appropriées. (iv) Dès que possible, nous avons renforcé les résultats existants ou simplifié leur preuve.

3 Complexité des EFGs explicites

Nous commençons par les EFGs dont l'arbre de jeu est explicitement donné comme entrée. Autrement dit, la com-

*Cet article est un résumé étendu en français de [ ].

plexité se mesure par rapport à la taille de cet arbre. La complexité varie en fonction des caractéristiques du jeu, comme l'existence de hasard ou le degré d'information des joueurs. Pour ce dernier, nous considérons trois cas : information parfaite (IP), mémoire parfaite (MP) et mémoire parfaite multi-agents (MP-MA). MP-MA veut dire qu'un joueur est une équipe de plusieurs agents partageant le même gain et ayant chacun une mémoire parfaite ; les agents n'observent pas nécessairement les choix de leurs coéquipiers.
La complexité de PURE MAXMIN est présentée dans la Table 1 ; ceux en caractère gras sont nouveaux. Dans ce tableau, la complexité est croissante en les deux dimensions ; on constate aussi que conformément à notre intuition, le fait d'être multi-agents rend un jeu plus difficile à résoudre.
Pour prouver ces résultats, nous avons utilisé des réductions minimales, en ce sens qu'elles impliquent au plus deux agents de chaque équipe (MAX et MIN), le moins de tours possible et seulement des récompenses booléennes.

MAX \ MIN	Sans hasard	Avec hasard		
	IP/MP/MP-MA	IP	MP	MP-MA
IP	P	P	**NP-c**	**Σ_2^P-c**
MP	P	NP-c	NP-c	**Σ_2^P-c**
MP-MA	NP-c	NP-c	NP-c	Σ_2^P-c

TABLE 1 – Complexité de PURE MAXMIN.

4 Complexité des jeux représentés de façon compacte

Nous étudions ensuite la complexité des jeux représentés de façon compacte. La motivation est naturelle : les jeux de table auxquels on joue sont rarement définis explicitement par leur arbre de jeu, mais le sont plutôt par leurs *règles* et par quelques paramètres décrivant la taille du jeu. Par exemple, le jeu de go est défini par ses règles et par la taille du goban (traditionnellement de taille 19×19, mais possiblement d'autres dimensions) ; le jeu du bridge est défini par ses règles et par le nombre de couleurs et de rangs (normalement 4 couleurs, chacune avec 13 rangs). Ainsi, pour ces jeux, ce sont ces paramètres, plutôt que la taille de leur arbre de jeu, qui sont pertinents pour analyser la complexité.
Pour modéliser les jeux représentés de façon compacte, nous proposons deux formalismes diamétralement opposés.

— Le premier, que nous nommons *jeu booléen compact* (CBG ci-après), est une généralisation des formules booléennes quantifiées (avec ou sans dépendances ; QBF ci-après). Les QBF sont bien connues dans la littérature ; elles encodent de façon compacte des EFGs à information parfaite et sans hasard. Nous proposons le formalisme CBG pour généraliser QBF de façon minimale, tout en autorisant des facteurs de hasard et des équipes multi-agents.
— En revanche, le second, que nous nommons *jeu avec oracles* (OG ci-après), est très générique et permet de capturer les jeux qui se jouent en espace polynomial et avec un horizon polynomial, ce qui correspond à la plupart des jeux de table.

Il est facilement vérifiable qu'OG est effectivement plus expressif que CBG. Pour chaque classe de complexité, nous montrons le résultat de difficulté pour CBG et le résultat d'appartenance pour OG, ce qui nous permet de confirmer :

— que ces deux formalismes (et donc également tous les formalismes intermédiaires) sont équivalents en termes de complexité (à des transformations polynomiales près) ;
— que la notion de complexité utilisée est robuste au formalisme utilisé pour les jeux représentés de façon compacte, et donc qu'elle représente bien la difficulté intrinsèque de ces jeux, indépendamment de la représentation choisie.

Sans surprise, PURE MAXMIN est exponentiellement plus difficile lorsqu'un jeu est défini de façon compacte plutôt que par son arbre de jeu. Concrètement, les problèmes inclus dans P dans la Table 1 deviennent PSPACE-complets ; ceux qui sont NP-complets deviennent NEXP-complets ; ceux qui sont Σ_2^P-complets deviennent $\mathsf{NEXP}^{\mathsf{NP}}$-complets.

5 Complexité contre des OMs

Nous considérons aussi le cas où MIN choisit sa stratégie dans un ensemble restreint et connu par MAX. Ces stratégies de MIN sont appelées *modèles d'opposants* (OM ci-après) dans la littérature ; cette notion d'OM permet de simuler les situations dans lesquelles MAX connaît (parfaitement ou partiellement) la procédure de raisonnement de MIN. Les résultats sont présentés dans la Table 2, où les colonnes correspondent au nombre d'OM connus par MAX.

MAX \ #OM	1	Constante (≥ 2)	Non majoré
Sans hasard			
IP	P	P	P
MP	P	P	**P**
MP-MA	P	**P**	NP-c
Avec hasard			
IP	P	**NP-c**	**NP-c**
MP	P	**NP-c**	NP-c
MP-MA	NP-c	NP-c	NP-c

TABLE 2 – Complexité de PURE OM-MAXMIN.

6 Autres résultats

Nous terminons le travail en étudiant la complexité des problèmes liés au calcul d'une borne supérieure ou de la valeur exacte de la valeur maxmin pure. Il s'avère que la complexité de ces problèmes est intimement liée aux résultats dans la Table 1.

Références

[1] Junkang Li, Bruno Zanuttini, and Véronique Ventos. The complexity of pure maxmin strategies in two-player extensive-form games. *J. Artif. Intell. Res.*, 82 :241–284, 2025.

Compromis entre sécurité et efficacité : Garanties formelles pour les MDP orientés but

Matisse Roche[1], Yoko Watanabe[1], Caroline P.C. Chanel[2]

Fédération ENAC ISAE-SUPAERO ONERA,
Université de Toulouse, Toulouse, France
Email: {prénom}.{nom}@onera.fr [1], caroline.chanel@isae-supaero.fr [2]

Résumé

Dans les processus de décision markoviens orientés but avec états catastrophiques, nous travaillons dans un cadre de maximisation d'utilité, une fonction qui prend en compte à la fois le coût cumulé et l'atteinte ou non du but. Ce cadre intègre un paramètre K_g représentant le bonus que l'agent reçoit lorsqu'il atteint un état but. Notre contribution principale est une formule analytique qui, pour un seuil de probabilité d'attendre le but P_{th} spécifié par l'utilisateur, détermine la valeur minimale de K_g garantissant que la politique optimale respecte ce seuil de sécurité. Cette transformation du MDP permet ainsi d'utiliser toutes les méthodes classiques et efficaces de résolution tout en bénéficiant automatiquement de la garantie de sécurité, sans recourir à la programmation par contraintes. Les évaluations expérimentales conduites sur le "river problem" démontrent l'avantage de notre approche par rapport aux méthodes traditionnelles qui privilégient exclusivement la sécurité maximale, souvent au détriment d'une efficacité raisonnable.

Mots-clés

MDP orientés but ; Garantie de sécurité ; Plus court chemin stochastique

Abstract

In goal-oriented Markov decision processes with dead-ends, we work within a utility maximization framework. Here, utility is a function that takes into account both the cumulative cost and whether the goal is reached. This framework uses a parameter K_g representing the bonus that the agent receives when it reaches a goal. Our main contribution is to propose an analytical formula that, for a user-specified threshold for a probability of reaching the goal P_{th}, determines the minimum value of K_g guaranteeing that the solution policy respects this safety threshold. This MDP model transformation thus allows the use of classical and efficient resolution algorithms for finding a policy with the safety guarantee, without the need of constraint programming. Experimental evaluations conducted on the "river problem" demonstrate the significant advantage of our approach compared to traditional methods that always prioritize maximum safety, often at the expense of reasonable efficiency.

Keywords

Goal-oriented MDPs ; Safety Guarantee ; Stochastic Shortest Path

1 Introduction

Les problèmes de plus court chemin stochastique (SSP, *Stochastic Shortest Path*) ont été formellement introduits par Bertsekas et Tsitsiklis [1] comme une classe spécifique des Processus de Décision Markoviens (MDP) orientés but. Les SSP-MDP constituent un formalisme puissant pour modéliser et résoudre des problèmes de planification en environnement incertain [6].

Dans de nombreuses applications réelles comme la robotique autonome, la planification de mission ou la gestion de systèmes critiques, ces modèles doivent prendre en compte l'existence d'états catastrophiques – des états à partir desquels il devient impossible d'atteindre le but quelle que soit la politique suivie. Face à ces états catastrophiques, les approches traditionnelles de résolution des problèmes SSP-MDP se révèlent inadaptées [7]. En effet, ces approches reposent sur deux hypothèses fondamentales : l'existence d'une politique propre, politique atteignant le but avec probabilité 1, et l'accumulation d'un coût infini pour toute politique impropre. Ces hypothèses sont rarement satisfaites dans des environnements complexes où les états catastrophiques sont inévitables.

Diverses extensions ont été proposées pour contourner ces limitations. Le cadre SSPUDE (*SSP with unavoidable dead-ends*) de Kolobov (voir [7]) traite les états catastrophiques en leur attribuant soit une pénalité finie (fSSPUDE), soit une pénalité infinie (iSSPUDE). Le cadre S3P de Teichteil-Königsbuch [9] adopte quant à lui une approche lexicographique qui maximise d'abord la probabilité d'atteindre le but, puis minimise le coût espéré parmi les politiques plus sûres. Ces approches partagent cependant un inconvénient majeur : en priorisant systématiquement la sécurité maximale, elles peuvent conduire à des comportements excessivement conservateurs et inefficaces. Dans de nombreux contextes industriels et applications réelles, il est plus pertinent de rechercher un compromis entre sécurité et efficacité.

Une autre approche est la recherche de politique assurant un minimum de sécurité, notamment [8], qui introduit le problème *AtLeastProb*. Ce problème consiste à trouver une po-

litique garantissant un seuil de probabilité d'atteinte du but défini par l'utilisateur. Dans cette approche, aucun critère lié au coût ou à l'efficacité n'est pris en compte dans la recherche de la politique optimale.
Alternativement, un nouveau model, basé sur la maximisation de l'utilité a été proposé par [5]. Ce nouveau cadre, appelé le modèle GUBS, permet un contrôle du compromis entre sécurité et efficacité : on peut préférer une politique moins sûre mais la perte de sécurité est limitée. à l'aide d'un paramètre K_g représentant un bonus en cas d'atteinte du but.
Dans cet article on aborde le problème de maximisation de l'efficacité sous la contrainte de sécurité minimale. Inspirés par le modèle GUBS, nous proposons une formule analytique pour déterminer la valeur de K_g qui garantisse que la politique optimale respecte ce seuil de sécurité. Notre approche à l'avantage de ne pas considérer la contrainte lors de l'optimisation. On peut ainsi utiliser les algorithmes classiques de résolution de MDP. Les évaluations expérimentales conduites sur un problème de référence démontrent que notre méthode respecte systématiquement les contraintes de sécurité imposées. De plus, notre méthode est capable de fournir différentes politiques en fonction de la contrainte de sécurité choisie.
Nous présentons d'abord les fondements théoriques des processus de décision markoviens orientés but et les approches existantes pour traiter les états catastrophiques (section 2). Nous développons ensuite notre contribution principale sur le compromis efficacité-sécurité, en établissant une formule analytique qui garantit un seuil minimal de sécurité (section 3). Nous détaillons notre méthode de résolution numérique (section 4) avant de présenter les résultats de nos expérimentations sur le problème de la rivière, qui confirment la validité de notre approche (section 5).

2 Définitions et travaux connexes

2.1 Définitions préliminaires

Les processus de décision markoviens (MDP) [2, 6] constituent un cadre fondamental pour la modélisation et la résolution de problèmes de décision séquentielle en environnement stochastique. Parmi leurs nombreuses variantes, les MDP orientés but représentent une classe particulièrement pertinente pour diverses applications comme la planification de chemin dans le cadre de la robotique autonome.
Formellement, un MDP orienté but est défini comme un tuple $\langle S, A, T, c, \mathcal{G}\rangle$ où :

- S est l'ensemble fini des états ;
- $\mathcal{G} \subseteq S$ est l'ensemble fini des états but ;
- A est l'ensemble fini des actions ;
- $T : S \times A \times S \to [0, 1]$ est la fonction de transition ;
- $c : S \times A \times S \to \mathbb{R}$ est la fonction de coût.

Dans ce cadre, les états but $g \in \mathcal{G}$ possèdent deux propriétés fondamentales :

- Ils sont absorbants : $T(g, a, g) = 1$ pour toute action $a \in A$;
- Ils n'engendrent aucun coût supplémentaire : $c(g, a, g) = 0$ pour toute action $a \in A$.

Une *politique* pour un MDP est une règle de décision qui spécifie quelle action choisir en fonction de l'historique du processus. Formellement, une politique générale peut être définie comme une fonction $\pi = (\pi_0, \pi_1, \ldots)$ où $\pi_t : H_t \to \Delta(A)$ associe à chaque historique $h_t \in H_t$ jusqu'à l'instant t une distribution de probabilité sur les actions. Ici, H_t représente l'ensemble des historiques possibles jusqu'à l'instant t, c'est-à-dire les séquences $h_t = (s_0, a_0, s_1, a_1, \ldots, s_t)$.
Une *politique stationnaire* est une politique qui ne dépend que de l'état courant et non de l'historique complet ou du temps. Elle est définie comme une fonction $\pi : S \to \Delta(A)$ où $\pi(s)$ représente une distribution de probabilité sur les actions pour l'état s. Dans le cas déterministe, une politique stationnaire est simplement une fonction $\pi : S \to A$ qui associe à chaque état une unique action.
Une *trajectoire* θ correspond à un historique $h = \{s_0, a_0, s_1, a_1, \ldots\}$. Nous notons $\Theta_{\mathcal{G}}^{\pi,s,t}$ l'ensemble des trajectoires jusqu'à l'instant t qui atteignent un état but $g \in \mathcal{G}$ lorsque la politique π est appliquée à partir de l'état initial s. t est omis lorsque l'horizon est infini : $\Theta_{\mathcal{G}}^{\pi,s} = \Theta_{\mathcal{G}}^{\pi,s,\infty}$.

2.2 Problèmes de plus court chemin stochastique (SSP)

Les problèmes de plus court chemin stochastique (SSP, *Stochastic Shortest Path*) ont été formellement introduits par Bertsekas et Tsitsiklis [1] comme une classe spécifique de MDP orientés but. Les SSP reposent sur deux hypothèses essentielles :

1. Il existe au moins une politique stationnaire (dite *propre*) qui atteint un état but avec probabilité 1 depuis tout état initial.
2. Toute politique impropre (n'atteignant pas un but avec probabilité 1) accumule un coût infini.

La seconde hypothèse implique généralement que tous les coûts des transitions entre états non-buts sont strictement positifs, ce qui garantit que les cycles infinis sont pénalisés par un coût infini.
Sous ces conditions, résoudre un SSP consiste à trouver une politique π qui minimise l'espérance du coût cumulé jusqu'à l'atteinte d'un état but. Ce critère d'optimisation définit la fonction valeur telle que :

$$V^*(s) = \min_{\pi} \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} c(s_t, \pi(s_t), s_{t+1}) | s_0 = s\right]$$

.
Bertsekas et Tsitsiklis [2] ont montré que, sous les hypothèses SSP, cette fonction valeur est l'unique solution de l'équation de Bellman :

$$V^*(s) = \min_{a \in A} \sum_{s' \in S} T(s, a, s') \left[c(s, a, s') + V^*(s')\right]$$

avec $V^*(g) = 0$ pour tout $g \in \mathcal{G}$.

Des algorithmes classiques de résolution de MDP, tels que *Value Iteration* (VI) ou *Policy Iteration* (PI), peuvent être utilisés pour résoudre des SSP. D'autres algorithmes, plus efficaces que VI ou PI, ont été développés pour résoudre les SSP. Citons notamment *Labelled Real-Time Dynamic Programming* (LRTDP) [3], une variante de VI qui explore l'espace d'états de manière plus efficace en se concentrant sur les trajectoires les plus pertinentes.

2.3 Extensions pour les problèmes avec états catastrophiques

Dans de nombreux contextes pratiques, l'hypothèse selon laquelle un état but est toujours atteignable avec probabilité 1 n'est pas réaliste. Certains environnements comportent des états catastrophiques (ou "dead-ends") – des états à partir desquels il est impossible d'atteindre un état but, quelle que soit la politique suivie.

2.3.1 Approches par facteur d'actualisation

Une première approche pour traiter les états catastrophiques consiste à introduire un facteur d'actualisation $0 < \gamma < 1$ dans la fonction valeur :

$$V_\gamma^*(s) = \min_\pi \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t c(s_t, a_t, s_{t+1}) | s_0 = s \right]$$

.

Ce facteur garantit la convergence de la somme, même en présence de trajectoires infinies. Cependant, cette approche présente plusieurs inconvénients :

- Elle modifie la structure du problème original en dévalorisant les coûts futurs.
- Elle ne distingue pas les trajectoires qui atteignent le but de celles qui aboutissent à des états catastrophiques.
- Le choix de γ influence la politique obtenue sans critère clair pour sa détermination.

2.3.2 Cadre SSPUDE

Kolobov et al. [7] ont proposé une extension plus formelle appelée SSPUDE (*Stochastic Shortest Path with Unavoidable Dead-Ends*). Ce cadre distingue deux variantes :

- **fSSPUDE** (Finite-penalty SSPUDE) : Attribue une pénalité finie D à chaque état catastrophique, ce qui permet d'utiliser des algorithmes standards comme VI ou LRTDP.
- **iSSPUDE** (Infinite-penalty SSPUDE) : Attribue une pénalité infinie aux états catastrophiques, formalisant l'idée qu'ils doivent être évités à tout prix. Cette variante est plus complexe algorithmiquement car elle nécessite une approche en deux phases : d'abord identifier les états à éviter, puis optimiser les coûts sur les états restants.

L'avantage principal de fSSPUDE est sa facilité d'implémentation, tandis qu'iSSPUDE offre des garanties théoriques plus fortes sur l'évitement des états catastrophiques. fSSPUDE est computationnellement beaucoup plus efficace qu'iSSPUDE. Pour une valeur suffisamment grande de la pénalité D, fSSPUDE produit des politiques équivalentes à celles d'iSSPUDE. Malheureusement, il n'existe pas de méthode analytique pour déterminer cette valeur critique de pénalité, ce qui rend le paramétrage de fSSPUDE délicat en pratique.

2.3.3 Le cadre S3P

Teichteil-Königsbuch [9] a introduit le cadre S3P (*Stochastic Safest and Shortest Path*) qui adopte explicitement une approche bicritère pour les problèmes avec états catastrophiques. Ce cadre définit un ordre lexicographique sur les politiques :

1. Maximiser d'abord la probabilité d'atteindre un état but : $P^\pi(s) = Pr(\theta \in \Theta_\mathcal{G}^{\pi,s})$
2. Parmi les politiques maximisant cette probabilité, minimiser l'espérance du coût conditionnel des trajectoires réussies :

$$C^\pi(s) = \mathbb{E}[C(\theta) | \theta \in \Theta_\mathcal{G}^{\pi,s}]$$

.

Formellement, une politique optimale π_{S3P}^* pour S3P est définie par :

$$\pi_{S3P}^*(s) \in \arg \min_{\pi \in \arg\max_{\pi'} P^{\pi'}(s)} C^\pi(s)$$

.

Cette approche présente plusieurs avantages :

- Elle est bien définie même en présence d'états catastrophiques inévitables ;
- Elle se concentre uniquement sur les coûts des trajectoires qui atteignent effectivement le but ;
- Elle généralise le cadre SSP standard (lorsque des politiques propres existent, les solutions S3P sont identiques aux solutions SSP).

Dans tous ces cadres, on fait généralement l'hypothèse que les coûts des transitions entre états non-buts sont strictement positifs. Cette hypothèse permet d'éviter les problèmes de cycles de coût nul ou négatif, qui compliqueraient considérablement l'analyse théorique et les algorithmes de résolution.

2.3.4 Le cadre GUBS

Le cadre GUBS, proposé par Freire et Delgado [5], évalue une trajectoire θ en fonction à la fois de son coût cumulé et de l'atteinte ou non d'un état but. Plus précisément, l'utilité d'une trajectoire est définie par :

$$U(\theta) = u(C(\theta)) + K_g \cdot \mathbf{1}_{\theta \in \Theta_\mathcal{G}} \quad (1)$$

où :

- u est une fonction d'utilité sur les coûts satisfaisant les deux propriétés suivants :
  1. $u : \mathbb{R} \cup \{\infty\} \to [U_{min}, U_{max}]$, et
  2. u est strictement décroissante ;
- $C(\theta)$ est le coût cumulé de la trajectoire θ ;

- K_g est un paramètre représentant le bonus d'utilité accordé pour l'atteinte d'un état but ;
- $\mathbf{1}_{\theta \in \Theta_\mathcal{G}}$ est l'indicateur d'atteinte du but.

Dans le cadre GUBS, l'évaluation d'une politique π se fait par la fonction de valeur définie comme l'espérance de l'utilité sur toutes les trajectoires possibles :

$$V^{\pi}_{GUBS}(s) = \mathbb{E}\left[U\left(C(\theta), \mathbf{1}_{\theta \in \Theta_\mathcal{G}}\right) \mid \theta \in \Theta^{\pi,s}\right] \quad (2)$$

Cette formulation peut être développée comme suit :

$$V^{\pi}(s) = \left(U^{\pi}(s) + K_g\right) P^{\pi}(s) + \left(1 - P^{\pi}(s)\right) U_{\min} \quad (3)$$

où

$$U^{\pi}(s) = \mathbb{E}\left[u(C(\theta)) \mid \theta \in \Theta^{\pi,s}_{\mathcal{G}}\right] \quad (4)$$

La politique optimale π^*_{GUBS} pour GUBS est celle qui maximise la fonction de valeur (2.3.4). Pour garantir que le cadre GUBS priorise l'atteinte du but, il faut que $K_g > U_{max} - U_{min}$. Cette condition assure qu'une trajectoire atteignant le but aura toujours une utilité supérieure à une trajectoire qui n'y parvient pas, indépendamment des coûts.

2.3.5 Propriétés fondamentales du cadre GUBS

Pour calculer l'utilité d'un chemin dans le cadre GUBS, il est nécessaire de savoir si ce chemin atteint ou non un état but. Cette particularité entraîne une conséquence importante : la politique optimale pour le critère GUBS n'est généralement pas stationnaire. En effet, l'action optimale à exécuter dans un état dépend non seulement de l'état courant, mais aussi du coût cumulé jusqu'à ce point : $a^*_t = \pi^*_{GUBS}(s_t, C(\theta_t))$. Cela présente des défis computationnels significatifs, car cette dépendance au coût cumulé revient à créer un nouveau MDP en augmentant l'espace d'états avec la variable C, ce qui peut engendrer un nombre potentiellement infini d'états, tant en termes de calcul que de mémoire nécessaire pour stocker la politique. La résolution n'est d'ailleurs possible que si les coûts sont des rationnels. Néanmoins, Crispino et al. [4] ont démontré un résultat important : il existe une valeur C_{max} calculable à partir des données du problème, telle que pour tout état s et tout coût cumulé $C \geq C_{max}$, la politique optimale sous GUBS devient stationnaire. De plus, cette politique stationnaire correspond exactement à la politique optimale selon le critère lexicographique risque-sensible, comme la politique optimale S3P. Formellement, pour deux politiques π et π', on a $\pi \succ \pi'$ (c'est-à-dire que π est préférée à π') si :

- $P^{\pi}(s) > P^{\pi'}(s)$, ou
- $P^{\pi}(s) = P^{\pi'}(s)$ et $U^{\pi}(s) > U^{\pi'}(s)$

Lorsque la fonction d'utilité choisie est de type exponentielle, on parle alors de eGUBS (exponential GUBS). Pour résoudre ce problème particulier, Crispino et al. [4] ont proposé l'algorithme eGUBS-VI une approche qui adapte les méthodes de programmation dynamique classiques pour résoudre un MDP avec leurs critères.

2.4 Discussion sur ces approches

L'introduction d'états catastrophiques inévitables dans le modèle soulève une question fondamentale : comment évaluer et comparer les différentes politiques ? En effet, dans ce contexte, deux critères entrent en compétition :

- La sécurité : maximiser la probabilité d'atteindre un état but
- L'efficacité : minimiser le coût moyen des trajectoires atteignant un état but

Plusieurs approches ont été proposées dans la littérature pour traiter ce dilemme. La plus courante, utilisée dans S3P [9] ou iSSPUDE [7], adopte une préférence lexicographique qui priorise la sécurité maximale, puis minimise le coût parmi les politiques maximalement sûres. Cette approche présente cependant un inconvénient majeur : elle peut conduire à des politiques extrêmement inefficaces en termes de coût pour obtenir un gain marginal en probabilité. Par exemple, pour augmenter la probabilité d'atteindre sa destination de 95% à 96%, un voyageur pourrait devoir tripler son temps de trajet, ce qui est rarement acceptable en pratique.
De plus, dans de nombreux contextes industriels et applications réelles, il n'est pas toujours nécessaire ou même souhaitable de maximiser absolument la probabilité de succès. On recherche plutôt des politiques qui garantissent un niveau de sécurité suffisant (au-dessus d'un certain seuil) tout en optimisant l'efficacité. D'autres travaux ont exploré cette direction, notamment [8], qui introduit le problème AtLeast-Prob. Ce problème consiste à trouver une politique garantissant un seuil de probabilité d'atteinte du but défini par l'utilisateur $P_{th} \in [0, 1]$, soit formellement une politique π telle que $P^{\pi}(s) \geq P_{th}$ (ou prouver qu'une telle politique n'existe pas). Bien que cette formulation permette de fixer un niveau de sécurité minimal acceptable, elle ne traite pas de l'aspect efficacité. En effet, aucun critère lié au coût ou à l'efficacité n'est pris en compte dans la recherche de la politique optimale.

3 Le compromis Efficacité-Sécurité

3.1 MDP orienté but sous la contrainte de sécurité

Dans ce contexte, nous proposons une approche alternative qui permet de spécifier un niveau minimal de sécurité souhaité tout en optimisant l'efficacité. Formellement, nous cherchons à résoudre le problème d'optimisation suivant :

$$\min_{\pi \in \Pi} \mathbb{E}[C(\theta) | \theta \in \Theta^{\pi,s}_{\mathcal{G}}]$$
$$\text{t.q.} \quad P^{\pi}(s) \geq P_{th}$$

où P_{th} est un seuil de probabilité spécifié par l'utilisateur. Nous appelons ce seuil, le seuil de sécurité.

3.2 Garantie de sécurité pour GUBS

Pour obtenir une politique efficace, qui respecte le seuil de sécurité, nous nous appuyons sur le cadre théorique GUBS [5], présenté dans la section 2.3.4.

Nous proposons une méthode permettant de garantir qu'une politique optimale sous GUBS, π^*_{GUBS}, atteigne une probabilité de succès minimale P_{th} dans un MDP orienté but, en ajustant le bonus K_g. Pour cela, nous considérons les deux politiques extrêmes suivantes :

- La politique optimale pour l'ordre lexicographique π^*_L, qui atteint la probabilité maximale $P^*(s) = \max_\pi P^\pi(s)$ d'attendre le but ;
- La politique la plus efficace π^*_E, qui atteint l'espérance conditionnelle maximale $U^{\pi^*_E}(s) = \max_\pi U^\pi(s)$.

Théorème 3.1 (Garantie de sécurité pour GUBS)**.** *Soit un MDP orienté but $M = \langle S, A, T, c, \mathcal{G}\rangle$ et soit $P^*(s_0)$ la probabilité maximale d'atteindre $\mathcal{G}$ depuis l'état initial s_0. Pour tout seuil de sécurité $P_{th} \in]0, P^*(s_0)[$, définissons*

$$K^*_{P_{th}} = \frac{P_{th}(U^{\pi^*_E}(s_0) - U_{min}) - P^*(s_0)(U^{\pi^*_L}(s_0) - U_{min})}{P^*(s_0) - P_{th}}, \tag{5}$$

avec π^*_L la politique optimale pour l'ordre lexicographique et π^*_E pour la politique la plus efficace. $U^\pi(s_0)$ est l'espérance conditionnelle de l'utilité du coût cumulé pour les trajectoires atteignant un état but, défini par (2.3.4).
Alors, pour tout $K_g > K^*_{P_{th}}$, toute politique optimale selon le critère GUBS avec paramètre K_g possède une probabilité d'atteindre un état but supérieure ou égale à P_{th}.

Démonstration. Nous voulons démontrer que, pour $K_g \geq K^*_{P_{th}}$, aucune politique de proba de succès inférieure à P_{th} ne peut avoir une utilité moyenne supérieure à celle de la politique optimale pour l'ordre lexicographique.
La formule suivante définie $U^{\pi^*_E}(s)$:

$$U^{\pi^*_E}(s) = \max_\pi \lim_{T\to\infty} \mathbb{E}[u(C(\theta)) \mid \theta \in \Theta^{\pi,s,T}_{\mathcal{G}}]$$

Soit une politique π telle que $P^\pi(s_0) < P_{th}$, on majore l'utilité moyenne d'une telle politique :

$$\begin{aligned} V^\pi(s_0) &= \mathbb{E}\left[u(C(\theta)) + K_g \mathbf{1}_{\theta\in\Theta^{\pi,s_0}_{\mathcal{G}}} \mid \theta \in \Theta^{\pi,s_0}\right] \\ &= P^\pi(s_0)\left(\mathbb{E}[u(C(\theta)) \mid \theta \in \Theta^{\pi,s_0}_{\mathcal{G}}] + K_g\right) \\ &\quad + (1 - P^\pi(s_0))U_{min} \\ &= P^\pi(s_0)\left(\mathbb{E}[u(C(\theta)) \mid \theta \in \Theta^{\pi,s_0}_{\mathcal{G}}]\right. \\ &\quad \left. + K_g - U_{min}\right) + U_{min} \\ &< P_{th}(U^{\pi^*_E}(s_0) + K_g - U_{min}) + U_{min} \end{aligned}$$

D'autre part, pour π_L la politique optimale pour l'ordre lexicographique, on a :

$$V^{\pi^*_L}(s_0) = P^*(s_0)(U^{\pi^*_L}(s_0) + K_g - U_{min}) + U_{min}$$

Si l'on suppose que $V^{\pi^*_L}(s_0) \leq V^\pi(s_0)$, alors :

$$\begin{aligned} &P^*(s_0)(U^{\pi^*_L}(s_0) + K_g - U_{min}) + U_{min} \\ &\quad \leq P_{th}(U^{\pi^*_E}(s_0) + K_g - U_{min}) + U_{min} \end{aligned}$$

Ce qui implique :

$$\begin{aligned} K_g(P^*(s_0) - P_{th}) \leq P_{th}(U^{\pi^*_E}(s_0) - U_{min}) \\ - P^*(s_0)(U^{\pi^*_L}(s_0) - U_{min}) \end{aligned}$$

Finalement :

$$K_g \leq \frac{P_{th}(U^{\pi^*_E}(s_0) - U_{min}) - P^*_G(s_0)(U^{\pi^*_L} - U_{min})}{P^*_G(s_0) - P_{th}}$$

Par contraposée, si $K_g > K^*_{P_{th}}$, avec $K^*_{P_{th}}$ définit par l'équation (5), alors la politique π^*_L domine toute politique de probabilité de succès strictement inférieure à P_{th}. La politique optimale a donc une probabilité de succès supérieure à P_{th}, ce qui constitue la garantie de sécurité recherchée. □

3.3 Garantie de sécurité pour GUBS sans la politique la plus efficace

Dans l'expression précédente, le terme $U^{\pi^*_E}(s_0)$ n'est pas accessible sans résoudre le MDP en maximisant $U^\pi(s_0)$. Si on a accès à un majorant de $U^{\pi^*_E}(s_0)$ on peut l'utiliser à la place du terme $U^{\pi^*_E}(s_0)$ dans la formule précédente et la contrainte de sécurité sera toujours garantie. On peut obtenir un tel majorant en considérant le graphe de transition du MDP.

Théorème 3.2 (Garantie de sécurité pour GUBS sans π^*_E)**.** *Soit un MDP orienté but $M = \langle S, A, T, c, \mathcal{G}\rangle$ et soit $P^*(s_0)$ la probabilité maximale d'atteindre $\mathcal{G}$ depuis l'état initial s_0. Pour tout seuil de sécurité $P_{th} \in]0, P^*(s_0)[$, définissons*

$$K^+_{P_{th}} = \frac{P_{th}(u(d_G(s_0, \mathcal{G}) - U_{min}) - P^*(s_0)(U^{\pi^*_L}(s_0) - U_{min})}{P^*(s_0) - P_{th}}, \tag{6}$$

avec π^*_L la politique optimale pour l'ordre lexicographique et $d_G(s_0, \mathcal{G}) = \min_{\theta\in\Theta_{\mathcal{G}}} C(\theta)$.
Alors, pour tout $K > K^+_{P_{th}}$, toute politique optimale selon le critère GUBS avec paramètre K possède une probabilité d'atteindre un état but supérieure ou égale à P_{th}.

Démonstration. D'après le théorème 3.1, il suffit de montrer que $K^*_{P_{th}} \leq K^+_{P_{th}}$.
Par hypothèse u est décroissante, donc pour tout π et tout $\theta \in \Theta^\pi_{\mathcal{G}}$, $u(\theta) \leq u(d_G(s_0, \mathcal{G}))$, donc pour tout π, $U^\pi(s) \leq u(d_G(s_0, \mathcal{G}))$, d'où $U^{\pi^*_E}(s_0) \leq u(d_G(s_0, \mathcal{G}))$. On a donc bien $K^*_{P_{th}} \leq K^+_{P_{th}}$. □

4 Résolution numérique

Dans cette section, nous présentons l'algorithme eGUBS-VI, proposé par Freire et Delgado [4] qui calcule de manière efficace la fonction de valeur optimale et la politique optimale associée pour un paramètre K_g donné. Notre approche consiste à choisir une valeur du paramètre K_g de sorte à ce que la contrainte de sécurité soit garantie. Nous proposons également une méthode pour calculer $u(d_G(s_0, \mathcal{G}))$. On peut donc résoudre le MDP, avec la valeur de K_g fournit au théorème 3.2, et obtenir une politique qui respecte la contrainte de sécurité.

Comme Crispino et al [4], nous faisons l'hypothèse que la fonction d'utilité est une exponentielle décroissante :

$$u(C) = e^{\lambda C} \tag{7}$$

avec $\lambda < 0$ un paramètre. Cette hypothèse nous permet d'utiliser l'algorithme de résolution proposé par Crispino et al. Toutefois, notre méthode et notre garantie théorique restent valides quel que soit le choix de la fonction d'utilité.

4.1 Algorithme eGUBS-VI

L'algorithme eGUBS-VI [4] suit les étapes suivantes :

1. **Calcul de la politique optimale lexicographique sensible au risque :** D'abord, calculer la politique optimale pour le critère lexicographique sensible au risque en utilisant l'algorithme RiskSensitiveLexicographicVI, une variante de VI qui procède en deux phases à chaque itération. Pour chaque état s, il calcule d'abord la probabilité maximale d'atteindre l'objectif :

$$P_{\mathcal{G}}(s) \leftarrow \max_{a \in A} \sum_{s' \in S} T(s,a,s') P'_{\mathcal{G}}(s')$$

Ensuite, il définit l'ensemble des actions qui maximisent cette probabilité :

$$A_s^{P_{\mathcal{G}}} = \text{Argmax}_{a \in A} \sum_{s' \in S} T(s,a,s') P_{\mathcal{G}}(s')$$

Enfin, il met à jour la fonction d'utilité en utilisant uniquement ces actions maximisant la probabilité :

$$U_\lambda(s) = \max_{a \in A_s^{P_{\mathcal{G}}}} \left\{ \sum_{s' \in S} T(s,a,s') e^{\lambda c(s,a,s')} U_\lambda(s') \right\}$$

La fonction $U_\lambda(s)$ converge vers $U^{\pi_L^*}(s)$, qui représente l'utilité conditionnelle espérée pour la politique lexicographique sensible au risque π_L^*. De même, $P_{\mathcal{G}}(s)$ converge vers $P^*(s)$, la probabilité maximale d'atteindre un état objectif à partir de s.

2. **Calcul de C_{max} :** Ensuite, déterminer la valeur C_{max} à partir de laquelle la politique optimale devient stationnaire. L'algorithme pour calculer C_{max} est présenté dans [4] page 26.
3. **Calcul de la politique optimale GUBS :** L'algorithme calcule itérativement en remontant de C_{max} à 0 :
   i. **La politique optimale** pour chaque état s et coût C :

$$\pi^*(s,C) = \arg\max_{a \in A} \{Q(s,a,C) + K_g \cdot P'_{\mathcal{G}}(s,a,C)\}$$

   ii. **Les composantes pour évaluer les actions :**

$$Q(s,a,C) = \sum_{s'} T(s,a,s') \cdot e^{\lambda C'} \cdot V^*(s',C')$$

$$P'_{\mathcal{G}}(s,a,C) = \sum_{s'} T(s,a,s') \cdot P_{\mathcal{G}}(s',C')$$

   où $C' = C + c(s,a,s')$ et $s' \in S$.

   iii. **Mise à jour des valeurs optimales :** Après avoir déterminé $a^* = \pi^*(s,C)$, on met à jour :

$$V^*(s,C) = Q(s,a^*,C)$$
$$P_{\mathcal{G}}(s,C) = P'_{\mathcal{G}}(s,a^*,C)$$

   Ces valeurs sont utilisées pour calculer la politique aux étapes précédentes.

4.2 Calcul de $d_G(s_0, \mathcal{G})$

La méthode proposé dans cet article nécessite de calculer la valeur de $K^+_{P_{th}}$ en utilisant l'équation (6) dans le théorème 3.2. L'algorithme e-GUBS VI fournit déjà les termes $P^*(s_0)$ et $U^{\pi_L^*}(s_0)$. Nous expliquons ici comment calculer rapidement le dernier terme manquant, $d_G(s_0, \mathcal{G})$.

Définition 4.1 (Graphe déterministe d'un MDP)**.** Soit $M = \langle S, A, T, c \rangle$ un processus décisionnel de Markov. Le graphe déterministe associé à M est un graphe orienté pondéré $G = (V, E, w)$ où :

- $V = S$ est l'ensemble des sommets correspondant aux états du MDP ;
- $E \subseteq S \times S$ est l'ensemble des arêtes tel que $(s,s') \in E$ si et seulement s'il existe une action $a \in A$ telle que $T(s,a,s') > 0$;
- $w : E \to \mathbb{R}^+$ est la fonction de pondération des arêtes définie par :

$$w(s,s') = \min_{a \in A : T(s,a,s') > 0} c(s,a,s')$$

Dans ce graphe, le poids d'une arête représente le coût minimal pour transiter d'un état à un autre, à condition qu'une telle transition soit possible avec une probabilité strictement positive dans le MDP orienté but original.
On note que $d_G(s_0, \mathcal{G})$ est égal au coût du plus court chemin de s_0 vers un état but dans le graphe déterministe G.
Pour calculer $d_G(s_0, \mathcal{G})$, nous utilisons l'algorithme de Dijkstra, qui détermine efficacement le plus court chemin dans un graphe à pondérations positives.

5 Évaluations expérimentales

Pour évaluer notre approche, nous avons utilisé l'environnement "River Problem" introduit par Freire et Delgado [5] dans le cadre de leurs travaux sur GUBS. Ce banc d'essai constitue un cas d'étude permettant de tester notre méthode et de visualiser les différents compromis entre sécurité et efficacité.

5.1 L'environnement *River Problem*

Le problème de la rivière (*River Problem*), illustré dans la figure 1), considère une carte sous forme de grille de dimensions $N_x \times N_y$, où les extrémités en coordonnée x (soit $x = 1$ et $x = N_x$) représentent les berges de la rivière. L'agent doit traverser la rivière, ce qui peut être réalisé de deux manières :

- en nageant à partir de n'importe quel point de la berge, ou

- en longeant la berge jusqu'à atteindre un pont situé à $y = N_y$.

Cependant, la rivière s'écoule vers une chute d'eau (située à $y = 1$), où l'agent peut se retrouver piégé. Ces états correspondent à des états catastrophiques dans notre modèle.
L'état initial se trouve d'un côté de la rivière et loin du pont : $x_0 = 1$ et $y_0 = 2$. L'état but se situe de l'autre côté de la rivière, loin du pont : $x_g = N_x$ et $y_g = 1$.
Les actions peuvent être prises dans chacune des directions cardinales : Nord (N), Sud (S), Est (E) et Ouest (W). Si les actions sont effectuées sur la berge ou sur le pont, les transitions sont déterministes vers les directions cardinales correspondantes. Chaque action engendre un coût de 1. En revanche, si les actions sont effectuées dans la rivière, les transitions deviennent probabilistes : l'agent suit la direction cardinale choisie avec une probabilité $(1 - P)$, ou est emporté vers le bas (Sud) de la rivière avec une probabilité P. La chute d'eau est modélisée comme un ensemble d'états catastrophiques.

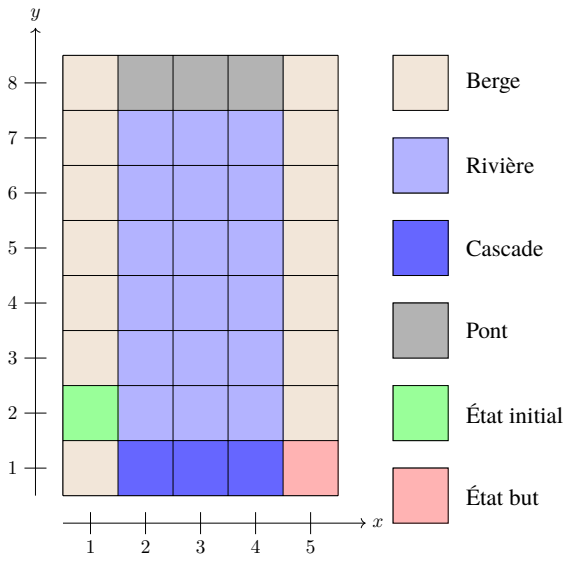


FIGURE 1 – Représentation de l'environnement "river problem" pour les paramètres $N_x = 5$ et $N_y = 8$.

5.2 Configuration expérimentale

Pour nos expériences, nous avons fixé les paramètres suivants : $N_x = 5$, $N_y = 15$ et $P = 0.4$. Le coût de chaque action est constant et égal à 1. On choisit $\lambda = -0.1$ comme constante pour l'utilité exponentielle 4.
Afin de simplifier l'analyse et la visualisation des politiques, nous avons légèrement modifié l'environnement original en n'autorisant que l'action d'aller vers l'Est (E) lorsque l'agent se trouve dans la rivière. Cette modification présente plusieurs avantages :

- Une politique peut être résumée essentiellement par le moment où l'agent décide d'entrer dans l'eau (la coordonnée y d'entrée dans la rivière).
- Cette configuration permet de dériver une formule analytique exacte pour calculer la probabilité d'atteindre le but en fonction de P et de la coordonnée y d'entrée dans la rivière, ce qui nous permet de comparer notre méthode numérique avec un oracle. Ici l'oracle est donc, pour un niveau de sécurité P_{th} donné, la politique la plus efficace, de proba de succès supérieure à P_{th}.

Cette simplification permet de représenter visuellement les différentes politiques par un seul paramètre : la hauteur à laquelle l'agent décide de traverser. Elle met également en évidence le compromis fondamental de ce problème : plus l'agent monte au Nord avant de traverser, plus il augmente sa sécurité (probabilité d'atteindre le but), mais en contrepartie, le coût du trajet augmente également.

5.3 Protocole expérimental

Notre protocole expérimental a consisté à :

1. Choisir plusieurs valeurs de seuil de sécurité P_{th}.
2. Pour chaque valeur de P_{th}, calculer la valeur minimale de $K^+_{P_{th}}$ en utilisant notre formule analytique présentée dans le théorème 3.2.
3. Résoudre le problème GUBS avec cette valeur de $K_g = K^+_{P_{th}}$ à l'aide de l'algorithme eGUBS-VI [4].
4. Calculer la probabilité exacte d'atteindre le but pour chaque politique, en exploitant les caractéristiques de cet environnement qui permettent de déterminer cette probabilité en fonction de la coordonnée d'entrée dans la rivière.
5. Déterminer la politique oracle pour comparer et évaluer les performances de notre approche par rapport à cette référence optimale.

5.4 Résultats et analyse

Les résultats de notre méthode sont présentés dans plusieurs graphiques qui illustrent différents aspects du compromis entre sécurité et efficacité.
La figure 2 illustre l'évolution de la politique optimale en fonction du seuil de sécurité P_{th}. L'axe des ordonnées représente la coordonnée y à laquelle l'agent décide d'entrer dans la rivière selon la politique calculée.
Comme prévu, plus le seuil de sécurité P_{th} augmente, plus la politique optimale choisit de traverser la rivière à une altitude élevée, renforçant ainsi la sécurité du parcours.
La figure 3 compare la probabilité de succès de notre méthode avec celle de la politique oracle. Notez que ces probabilités sont des valeurs exactes calculées analytiquement, et non des résultats empiriques d'exécutions répétées. Nous constatons que notre méthode respecte systématiquement la contrainte de sécurité, tous les points se situant au-dessus de la droite d'équation $y = x$. Cela confirme que la probabilité d'atteindre le but pour un P_{th} donné est toujours supérieure ou égale à P_{th} avec notre approche.
Néanmoins, comme le montre la Figure 4, qui représente l'utilité du coût cumulé en fonction de P_{th}, notre méthode produit des politiques plus conservatrices que l'oracle. La

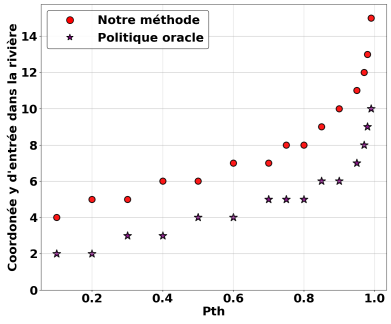


FIGURE 2 – Coordonnée d'entrée dans la rivière en fonction du seuil de sécurité P_{th}

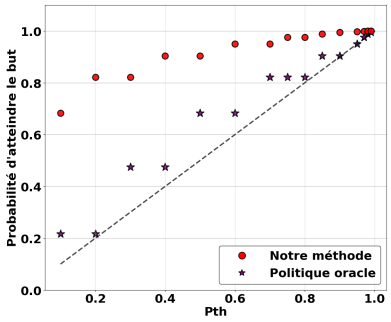


FIGURE 3 – Probabilité de succès observée en fonction du seuil de sécurité P_{th}.

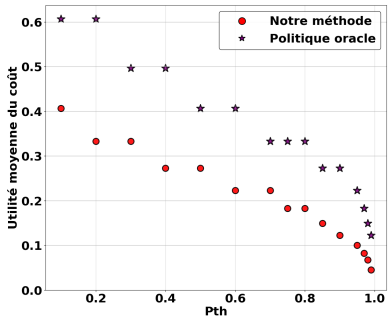


FIGURE 4 – Utilité du coût cumulé total en fonction du seuil de sécurité P_{th}.

politique oracle présente systématiquement une meilleure utilité pour un même seuil de sécurité. Cette observation suggère que notre approche, bien que garantissant formellement le respect du seuil de sécurité, tend à être un peu trop prudente dans certains cas. Des raffinements de notre formule analytique pourraient permettre de réduire cet écart dans de futurs travaux.

La Figure 2 révèle une discontinuité significative dans la politique optimale : l'agent ne sélectionne jamais l'altitude $y = 14$ comme point d'entrée dans la rivière. Cette discontinuité dans l'espace des politiques optimales s'explique avec la théorie de GUBS.

Pour comprendre ce phénomène, la Figure 5 présente l'évolution de C_{max} en fonction de P_{th}. Dans la théorie GUBS,

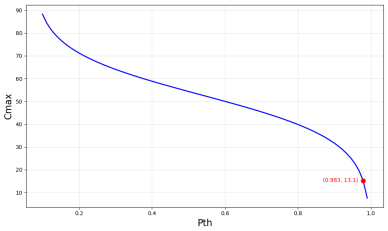


FIGURE 5 – Évolution de C_{max} en fonction du seuil de sécurité P_{th}.

pour un coût cumulé supérieur à C_{max}, la politique optimale correspond à celle de l'ordre lexicographique. On observe que pour $P_{th} \approx 0.98$, C_{max} chute en dessous de 12. Cette valeur est cruciale car elle représente exactement le coût accumulé par l'agent lorsqu'il se trouve encore sur la berge à la position $y = 14$. Ainsi, lorsque l'agent atteint cette position avec un P_{th} supérieur à 0.98, il bascule automatiquement vers la politique la plus sûre (la politique lexicographique) et continue donc son chemin jusqu'au pont situé à $y = 15$, sans jamais traverser la rivière à $y = 14$. Ce phénomène explique l'absence de politique d'entrée à $y = 14$ dans la Figure 2 et illustre comment la théorie GUBS influence directement le comportement de l'agent en fonction du seuil de sécurité imposé.

6 Conclusion et perspectives

Dans cet article, nous avons abordé la problématique du compromis entre sécurité et efficacité dans les processus de décision markoviens orientés but avec états catastrophiques. Nous avons constaté qu'il n'existe pas de critère d'optimalité canonique dans ce cadre, car nous sommes confrontés à une optimisation multi-objectif où sécurité et efficacité sont souvent en opposition.

Notre contribution principale réside dans une formule analytique qui détermine une valeur du bonus d'utilité K_g garantissant qu'une politique optimale dans le cadre GUBS respecte un seuil de probabilité d'atteindre le but spécifié par l'utilisateur. Cette approche, à notre connaissance, constitue la première tentative formelle d'obtenir des garanties de sécurité tout en optimisant l'efficacité pour les MDP orientés but avec états catastrophiques.

Les évaluations expérimentales conduites sur le problème de la rivière démontrent que notre méthode respecte systématiquement les contraintes de sécurité imposées. Cependant, comme l'illustrent nos résultats, notre approche actuelle tend à être conservatrice, produisant des politiques dont la probabilité de succès dépasse souvent le seuil requis, parfois au détriment de l'efficacité optimale.

Dans nos travaux futurs, nous envisageons principalement d'affiner notre méthode en développant des majorations plus précises pour l'estimation du paramètre K_g. Cette perspective d'amélioration est essentielle car elle permettrait de réduire le caractère conservateur de notre approche actuelle tout en maintenant les garanties formelles de sécurité qui constituent l'avantage principal de notre méthode. En paral-

lèle, nous prévoyons d'étendre nos expérimentations à des MDP orientés but de plus grande dimension afin d'évaluer la scalabilité de notre approche. Nous envisageons également de réaliser une analyse comparative entre notre méthode et les approches existantes de la littérature.
En conclusion, bien que notre approche ne fournisse pas encore la politique optimale exacte pour le critère de sécurité minimale que nous avons défini, elle offre néanmoins un cadre formel solide et une méthode pratique pour obtenir des politiques respectant des garanties de sécurité tout en maintenant une efficacité raisonnable, représentant ainsi une avancée significative par rapport aux approches existantes.

Références

[1] Dimitri P. Bertsekas and John N. Tsitsiklis. *Analysis of Stochastic Shortest Path Problems*, volume 16. INFORMS, 1991.

[2] Dimitri P. Bertsekas and John N. Tsitsiklis. An analysis of stochastic shortest path problems. *Mathematics of Operations Research*, 16(3) :580–595, 1991.

[3] Blai Bonet and Hector Geffner. Labeled RTDP : Improving the convergence of real-time dynamic programming. In *Proceedings of the 13th International Conference on Automated Planning and Scheduling (ICAPS)*, pages 12–21. AAAI Press, 2003.

[4] G. N. Crispino, V. Freire, and K. V. Delgado. Gubs criterion : Arbitrary trade-offs between cost and probability-to-goal in stochastic planning based on expected utility theory. *Artificial Intelligence*, 316 :103848, 2023.

[5] V. Freire and K. V. Delgado. GUBS : A utility-based semantic for goal-directed Markov decision processes. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, pages 1260–1268. IFAAMAS, 2017.

[6] Andrey Kolobov et al. *Planning with Markov decision processes : An AI perspective*, volume 17. Morgan & Claypool Publishers, 2012.

[7] Andrey Kolobov, Mausam, and Daniel S. Weld. A theory of goal-oriented mdps with dead ends. In *Proceedings of the Twenty-Eighth Conference on Uncertainty in Artificial Intelligence*, pages 438–447, 2012.

[8] Marcel Steinmetz, Jörg Hoffmann, and Olivier Buffet. Goal probability analysis in probabilistic planning : Exploring and enhancing the state of the art. *Journal of Artificial Intelligence Research*, 57 :229–271, 2016.

[9] Florent Teichteil-Königsbuch. Stochastic safest and shortest path problems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 26, pages 1825–1831, 2012.

Post-Hoc Interpretation of POMDP Policies

Geoffrey Laforest[1], Olivier Buffet[2], Alexandre Niveau[1], Bruno Zanuttini[1]

[1] Université Caen Normandie, ENSICAEN, CNRS, Normandie Univ
GREYC UMR6072, F-14000 Caen, France
[2] Université de Lorraine, CNRS, Inria
LORIA, F-54000 Nancy, France
{prenom.nom}@unicaen.fr, {prenom.nom}@loria.fr

May 16, 2025

Résumé

Les politiques pour des processus de décision markoviens partiellement observables sont des objets riches, prescrivant des actions en fonction de l'historique entier des observations et actions. Avec l'interpretabilité comme objectif, nous proposons de redécrire des représentations peu interprétables comme des fonctions définies sur les valeurs de descripteurs de l'état de croyance courant, construits de manière systématique à partir de descripteurs de l'état. Ces fonctions peuvent à leur tour être représentées par des objets intelligibles, comme des matrices de poids ou des arbres de décision. Nous étudions les problèmes de calcul afférents, et comparons empiriquement notre approche avec une redescription des politiques par des contrôleurs à états finis.

Mots-clés

POMDP, politique, interprétabilité

Abstract

Policies for partially observable Markov decision processes are rich objects, prescribing actions to take depending on the whole history of observations and actions. Towards interpretability, we propose to redescribe poorly interpretable representations as mappings defined on the value of features of the current belief state, built in a systematic manner from state features. Those mappings can in turn be represented by intelligible objects, like matrices of weights or decision trees. We investigate related computational problems and experimentally compare our approach with redescription of policies as finite-state controllers.

Keywords

POMDP, policy, interpretability

1 Introduction

We are interested in the representation of policies of actions for partially observable Markov decision processes (POMDPs), which are stochastic decision processes with partially observable states. At execution time, a policy prescribes actions to take depending on the whole history of actions and observations. Standard representations of such policies are by finite-state controllers (with states/nodes associated to actions and transitions via observations), and by sets of α-*vectors*, which are hyperplanes in the belief space, that is, the space of probability distributions over the state space (which represent possible beliefs of the agent over the current state).

Arguably, such representations lack interpretability: states in finite-state controllers are simply atoms, and α-vectors are high-dimensional vectors of real values. Hence we propose a new representation of policies for POMDPs, in which a set of *epistemic features* is built in a systematic manner from a given set of state features, and policies are represented by interpretable objects (e.g., matrices of weights, decision trees) defined on the embedding of the belief space onto these features.

In a nutshell, an epistemic feature is a propositional formula over the state features (a small disjunction or conjunction), and its value in a belief state is the probability that it is satisfied by a state drawn from it. This allows us to propose representations of policies over features like "the probability that the goal is at $(0, 0)$" which, we think, makes them more interpretable.

We define these representations and algorithms for computing them. Our approach is *post-hoc*, that is, starts from a (non-interpretable) policy and redescribes it into a more interpretable one. We finally compare our representations to those based on finite-state controllers in terms of size and computational time.

2 Related Work

Solving POMDPs exactly is intractable in general except for the smallest problems, due to the rapid growth in complexity. Point-based algorithms have been a breakthrough for scaling and solving POMDPs, by avoiding the curse of dimensionality and the curse of history. Point-Based Value Iteration (PBVI) [31] samples a small set of representative belief points and locally updates a lower bound of the optimal value function represented by α-vectors, which ensures that it is piece-wise linear and

convex. Heuristic Search Value Iteration (HSVI) [35, 36] draws on the same idea but, contrary to PBVI, it also maintains an upper bound on the optimal value function, and samples beliefs by generating trajectories while acting optimistically. One of the most efficient point-based algorithms is SARSOP [21]. It builds on HSVI, but tries to approximate the reachable belief states under an optimal policy. It also benefits from a pruning strategy based on the α-vector representation of the value function.

Another line of work leverages recurrent neural networks for their ability to encode the history [2]. They are often integrated in another architecture to make predictions, such as Deep Q-Networks [29]. Carefully designed and tuned, they can be a strong baseline for many POMDPs [30]. Recently, transformers have also been considered instead of recurrent networks, with mixed results [24]. A somewhat compact representation of a policy can be obtained by embedding it into an RKHS and truncating the embedding basis up to a certain number [25]. This results in a more compact but approximated policy.

2.1 Interpretability

There is a growing interest and a surge of work in explainable and interpretable machine learning [11]. This is motivated by ethical and legal concerns [7], and the need to understand algorithmic predictions to ensure decisions based on machine learning systems are safe and reliable [23]. In this work, we focus on interpretability, which refers to the degree to which a human can understand the model's results and the causes of its decisions. It often implies that a user can grasp how inputs are processed and mapped to the outputs in a human-comprehensible format [10]. Closely related notions often involve deriving agent behaviors that are more easily interpretable by external observers. For instance, behaviors may be considered *legible* if they facilitate guessing the agent's true objective, *explainable* if the agent's actions clearly align with some objective, or *predictable* if future actions can be easily anticipated [5]. Observer-aware MDPs (OAMDPs) provide a formal framework to address these interpretability challenges within stochastic environments [28].

Interpretability usually takes two roads [8]. Interpretability *by design* means that the models learnt or computed are taken from a language deemed interpretable *per se*. In planning, examples are representations by easy-to-understand trees [19, 37] or expressed with symbolic features [17, 12], sometimes both [6, 20]. In [17], a continuous state-action space is discretized, and the policy is represented with if-else fuzzy rules.

Now *post-hoc* interpretability, which is the setting of our work, means a model is redescribed, after learning or computation, into an interpretable language. In the setting of MDPs and RL, examples build on measuring the importance of reachable states [18, 9] or actions taken under the given policy. For instance, in [34], the importance of actions is assessed regarding their role in satisfying a predicate of interest at each time-step until the end of the episode, starting at a given horizon k. Another approach consists in targeting a simpler language [3].

In the POMDP setting, one can use symbolic features based on a logical language. For example, Meli et al. [26] use logical features to describe interesting traces of a given policy, then use these descriptions as heuristics for other solvers. Hence, though they are many similarities with our work, the focus there is on computing interpretable heuristics. In addition, in their experiments, they use features which require more than the transition model to be known.

2.2 Compression of Policies

Policies computed by search, as with SARSOP, are in practice represented by high-dimensional vectors with many nonzero coefficients. For this reason, symbolic solvers, which use a factored representation of α-vectors, like Symbolic Perseus [32], have been developed. Another representation of policies is by a tree branching on observations and prescribing an action at each node, and the natural generalization of this to an automaton, namely, a finite-state controller (FSC). In [13], the authors propose algorithms for computing such compact representations by FSCs, of policies given by oracles. Their focus is on (post-hoc) compression of policies, which can be seen as a manner of enhancing interpretability. We experimentally compare our approach with theirs in Section 6.

3 Background

Given a finite set S, we write $\mathcal{P}(S)$ for the set of all subsets of S, and $\Delta(S)$ for the probability simplex over S. Given a distribution δ, we write $\mathbf{x} \sim \delta$ to denote the fact that random variable $\mathbf{x}$ is sampled from δ, and $x \in \delta$ to mean that value x is in the support of δ.

A *Partially Observable Markov Decision Process* (POMDP) is a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{O}, T, Z, R, b_0, \gamma \rangle$, where $\mathcal{S}$, $\mathcal{A}$, $\mathcal{O}$ finite spaces of states, actions, and observations, resp.; for all $s \in \mathcal{S}, a \in \mathcal{A}$, $T_{s,a} \in \Delta(\mathcal{S})$ is the distribution of next states when a is taken in s; for all $s \in \mathcal{S}, a \in \mathcal{A}, s' \in T_{s,a}$, $Z_{s,a,s'} \in \Delta(\mathcal{O})$ is the distribution of observations upon transition $\langle s, a, s' \rangle$ and $R_{s,a,s'} \in \mathbb{R}$ the reward received for transition $\langle s, a, s' \rangle$; $b_0 \in \Delta(\mathcal{S})$ is an initial belief state; and $\gamma \in [0, 1)$ a discount factor.

Given a POMDP, a *history* is a finite sequence of actions and observations of the form $h := a_0 o_0 \dots a_k o_k$; we write $\langle\rangle$ for the empty history, and $\mathcal{H}$ for the set of all histories. A *belief state* is a probability distribution over $\mathcal{S}$, and we write $\mathcal{B}$ for the set of all belief states ($\mathcal{B} := \Delta(\mathcal{S})$). A history h induces a belief state $\mathrm{bs}(h)$ through Bayesian update:

$$\mathrm{bs}(\langle\rangle) := b_0,$$
$$\mathrm{bs}(h \cdot ao) := s' \mapsto \eta \sum_{s \in \mathcal{S}} Z_{s,a,s'}(o) T_{s,a}(s') \,\mathrm{bs}(h)(s),$$

where η is a normalizing constant. A *non-deterministic policy* $pol : \mathcal{H} \mapsto \mathcal{A}$ maps histories to actions. A *non-deterministic belief-based policy* $p \colon B \to \mathcal{P}(\mathcal{A})$ maps a set $B \subseteq \mathcal{B}$ to sets of actions, and induces a policy

$\mathrm{pol}(p)\colon \mathcal{H} \to \mathcal{P}(\mathcal{A})$ via $\mathrm{pol}(p)(h) := p(\mathrm{bs}(h))$, with $\mathrm{bs}(h) := b$ being the belief-state induced by h.
The *optimal value* at history h and infinite horizon is denoted by $\mathrm{V}^*(h)$ and defined by $\mathrm{V}^*(h) := \max_{a\in\mathcal{A}} \mathrm{Q}^*(h,a)$, with

$$\mathrm{Q}^*(h,a) := \mathop{\mathbb{E}}_{\substack{\mathbf{s}\sim\mathrm{bs}(h)\\ \mathbf{s}'\sim T_{\mathbf{s},a}}} (R_{\mathbf{s},a,\mathbf{s}'} + \gamma \mathop{\mathbb{E}}_{\mathbf{o}\sim Z_{\mathbf{s},a,\mathbf{s}'}} \mathrm{V}^*(h\cdot a\mathbf{o})).$$

We call *optimal nondeterministic policy* of the POMDP, the function $pol^* : \mathcal{H} \to \mathcal{P}(\mathcal{A})$ defined for all histories h by $pol^*(h) := \mathrm{argmax}_{a\in\mathcal{A}} \mathrm{Q}^*(h,a)$. This function maps any history to the set of all optimal actions to take at this history and the initial belief state, given an infinite horizon.
It is well-known that a POMDP can be cast into a fully observed MDP, called the *belief MDP*, whose abstract states correspond to the above belief states. The transition function between belief states is derived using Bayesian update as above. The belief state is a sufficient statistic for optimal decision-making, so that solving the belief MDP at infinite horizon provides a policy $p^* : \Delta(\mathcal{S}) \to \mathcal{P}(\mathcal{A})$ that is optimal for the POMDP at an infinite horizon.

Example 1. As a running example, we use a POMDP containing 4 states, written $\mathsf{s}_{00}, \mathsf{s}_{01}, \mathsf{s}_{10}, \mathsf{s}_{11}$ (read intuitively as s_{xy}). There are three actions: $\mathsf{check}_=$ does not change the state but (deterministically) produces observation true (resp. false) when taken in state s_{xy} with $x = y$ (resp. $x \neq y$); $\mathsf{switch}_\mathsf{x}$ deterministically maps each s_{xy} to $\mathsf{s}_{(1-x)y}$, yielding observation void; noop does not change the state and yields observation void. Finally, b_0 is the uniform distribution $\mathsf{b}^{\mathsf{unif}}$ over $\mathcal{S}$, and $R_{\cdot,\cdot,s'}$ is 1 for $s' \in \{\mathsf{s}_{00}, \mathsf{s}_{11}\}$, 0 otherwise.
Then $\mathsf{h}^= := \langle \mathsf{check}_=, \mathsf{false}, \mathsf{switch}_\mathsf{x}, \mathsf{void}\rangle$ is a history, inducing the belief state $\mathsf{b}^= := \mathrm{bs}(\mathsf{h}^=)$, which assigns probability $\frac{1}{2}$ to $\mathsf{s}_{00}, \mathsf{s}_{11}$. Let moreover $\mathsf{b}^{\neq}$ be the belief state assigning $\frac{1}{2}$ to $\mathsf{s}_{01}, \mathsf{s}_{10}$, and p be defined by $\mathsf{p}(\mathsf{b}^{\mathsf{unif}}) := \{\mathsf{check}_=, \mathsf{noop}\}$, $\mathsf{p}(\mathsf{b}^{\neq}) := \{\mathsf{switch}_\mathsf{x}\}$, and $\mathsf{p}(\mathsf{b}^=) = \{\mathsf{check}_=, \mathsf{noop}\}$.[1] Then p is a (non optimal) belief-based policy, inducing, for example, $\mathrm{pol}(\mathsf{p})(\mathsf{h}^=) := \mathsf{p}(\mathsf{b}^=) = \{\mathsf{check}_=, \mathsf{noop}\}$.
It is easy to see that the optimal policy is given by $\mathsf{p}^*(\mathsf{b}^{\mathsf{unif}}) := \{\mathsf{check}_=\}$, $\mathsf{p}^*(\mathsf{b}^{\neq}) := \{\mathsf{switch}_\mathsf{x}\}$, and $\mathsf{p}^*(\mathsf{b}^=) = \{\mathsf{check}_=, \mathsf{noop}\}$ □

Policies can be naturally represented as trees, a node being a history or a belief state, branching at each time-step on actions that can be taken at this node and subsequent possible observations. The problem of this representation is that it grows exponentially with the horizon.
Manipulating graphs rather than trees yields *finite state controllers* (FSCs) [27, 14, 13]. A non-deterministic FSC consists of a set of nodes N among which a set N_0 of distinguished initial nodes, an action mapping $act\colon N \to \mathcal{A}$ assigning an action to each node, and a partial transition function $\delta\colon N \times \mathcal{O} \to \mathcal{P}(N)$.[2]

[1] The definition over other belief states is irrelevant to our examples.
[2] The transition function is partial because not all observations may be received at each node during execution.

Intuitively, an FSC defines a nondeterministic policy *pol* as follows. The set of nodes *reachable* by the FSC through a history is defined by $\mathrm{N}(\langle\rangle) := N_0$ and $\mathrm{N}(h\cdot ao) := \bigcup_{n\in\mathrm{N}(h), act(n)=a} \delta(a,o)$. Now for any history h, $pol(h)$ is defined to be $\bigcup_{n\in\mathrm{N}(h)} act(n)$. In words, executing the FSC means taking any action associated with one of the current nodes, and progressing via a and o means restricting to those current nodes where a could indeed be taken, then transiting to their successors via o.
An ϵ-optimal (infinite-horizon) belief-based policy can also be obtained by approximating the optimal value function for a long horizon t. Such approximation is typically expressed as a set of sets Γ_a of *α-vectors*, one set per action a, with $V_{\{\Gamma_a\}}(b) := \max_{a\in\mathcal{A},\alpha\in\Gamma_a} b\cdot\alpha$ for all belief states b; the function $V_{\{\Gamma_a\}}$ has the nice property of being piecewise linear and convex. The induced nondeterministic policy is defined on belief states, by $p(b) := \mathrm{argmax}_{a\in\mathcal{A}}(\max_{\alpha\in\Gamma_a} b\cdot\alpha)$. However, as the dimension of the vectors grows linearly with the number of states and as the sets of alpha vectors Γ_a can become very large to represent the policy up to the desired precision, in practice this representation is neither more compact nor more readable than tree policies.
Optimal value functions (and policies) can also be obtained by relying on factored POMDPs [15], where the state and observation spaces are modeled using multiple random variables, and the transition and observation functions are modeled as 2-layer dynamic Bayesian networks. These functions, as well as the reward function, are often expressed as *algebraic decision diagrams* (ADDs) [1, 16, 32]. Dedicated operators allow computing the optimal value function also as an ADD, as in Symbolic Perseus [32], and typically lead to compact representations. We however leave this aspect for future work.

4 Epistemic Representations

Our proposal is based on the idea of describing policies over what we call *epistemic features*, which are features defined on the belief space. Formally, given a POMDP, we define an *epistemic feature* to be a function $\varphi : \mathcal{B} \to \mathbb{R}$. Epistemic features may take many different forms, for instance, entropy, but we choose to focus on epistemic features built in a systematic manner from a set of *state features*, that is, Boolean features of the form $f : \mathcal{S} \to \mathbb{B}$.[3] The reason is that such state features are typically readily available (from a factored representation of the POMDP, for instance) or easily defined by an expert.

Example 2 (continued)**.** The state space of our running example can be described with two state features, x and y, with $\mathsf{x}(\mathsf{s}_{xy}) := x$ and $\mathsf{y}(\mathsf{s}_{xy}) := y$; for instance, $\mathsf{x}(\mathsf{s}_{00}) = \mathsf{x}(\mathsf{s}_{01}) = 0$. □

To define epistemic features from state features, we use simple logical combinations. A (propositional) *clause* over variables F is a disjunction c of the form $(f_1 \vee \cdots \vee f_p \vee \neg f_{p+1} \vee \cdots \vee \neg f_w)$, where all f_i's are elements of F, all

[3] By $\mathbb{B}$ we mean $\{0,1\}$, with 0 standing for "false" and 1 for "true".

different from each other; dually, a *term* is a conjunction t of the form $(f_1 \wedge \cdots \wedge f_p \wedge \neg f_{p+1} \wedge \cdots \wedge \neg f_w)$. We call w the *width* of the clause or term. A clause or term of the above form is said to be *positive* if $p = w$ holds, that is, no negated literal occurs in it. Given that $\vee$ and $\wedge$ are commutative and idempotent, we consider $(f \vee f')$ and $(f' \vee f)$, for instance, to be the same clause (and similarly for terms).

Definition 1 (epistemic features of width w)**.** *Let $F \subseteq \mathbb{B}^{\mathcal{S}}$ be a set of Boolean state features, and let w be a positive integer. An* epistemic clause of width w *over F is an atom κ of the form $\mathrm{B}\, c$, where c is a clause of width w over F, and an* epistemic term of width w *is an atom τ of the form $\mathrm{B}\, t$, with t a term of width w over F. An* epistemic feature (of width w) *is an epistemic clause or term (of width w).*[4]

If c (resp. t) is positive, then $\mathrm{B}\, c$ (resp. $\mathrm{B}\, t$) is said to be positive as well. The set of all epistemic clauses (resp. terms, positive clauses, positive terms) of width w over F is written $\Phi_F^{\vee,w}$ (resp. $\Phi_F^{\wedge,w}$, $\Phi_F^{\vee,+,w}$, $\Phi_F^{\wedge,+,w}$).
Given a set F of state features over a state space $\mathcal{S}$, a state $s \in \mathcal{S}$ induces an assignment s_F to the features in F: $\forall f \in F : s_F(f) = f(s)$. Given a propositional formula g over F, we define $\mathrm{sat}(s, g)$ to be 1 (resp. 0) if s_F satisfies g under the standard semantics of propostional logic.

Definition 2 (semantics of epistemic features)**.** *The function induced by an epistemic clause $\kappa := \mathrm{B}\, c$ (resp. term $\tau := \mathrm{B}\, t$) is defined for all belief states b by $\kappa(b) := \mathbb{E}_{s\sim b}\, \mathrm{sat}(s, c)$ (resp. $\tau(b) := \mathbb{E}_{s\sim b}\, \mathrm{sat}(s, t)$).*

$\mathrm{B}\, c$ can literally be read as "the probability that c is true".

Example 3 (continued)**.** The epistemic clauses of width $w = 1$ are $\mathrm{B}\,\mathsf{x}$, $\mathrm{B}\,\mathsf{y}$, $\mathrm{B}\,\neg\mathsf{x}$, $\mathrm{B}\,\neg\mathsf{y}$; the first two are positive, the last two are not. For $w = 2$, the epistemic clauses are $\mathrm{B}(\mathsf{x} \vee \mathsf{y})$, $\mathrm{B}(\neg\mathsf{x} \vee \neg\mathsf{y})$, $\mathrm{B}(\mathsf{x} \vee \neg\mathsf{y})$, and $\mathrm{B}(\mathsf{y} \vee \neg\mathsf{x})$; only the first one is positive.
Consider b defined by $b(\mathsf{s}_{00}) := \frac{1}{2}$, $b(\mathsf{s}_{01}) := \frac{1}{4}$, and $b(\mathsf{s}_{10}) := \frac{1}{4}$. For $\kappa := \mathrm{B}(\mathsf{x} \vee \mathsf{y})$, we have $\kappa(b) = \frac{1}{2} \times 0 + \frac{1}{4} \times 1 + \frac{1}{4} \times 1 = \frac{1}{2}$, and for $\kappa := \mathrm{B}\,\neg\mathsf{y}$, we have $\kappa(b) = \frac{1}{2} \times 1 + \frac{1}{4} \times 0 + \frac{1}{4} \times 1 = \frac{3}{4}$. □

4.1 Projections

Given a set of epistemic features, our aim is to represent policies as mappings defined over *projections* of belief states onto these features. Importantly, we only consider policies defined over (or restricted to) a finite set of belief states (e.g., all those reachable by the policy up to a certain horizon), and leave the more general case to future work.

Definition 3 (projection)**.** *Let Φ be an ordered tuple of N epistemic features. The* projection *of belief state b (onto Φ) is defined to be the vector $\Phi(b) := (\varphi(b) \mid \varphi \in \Phi) \in \mathbb{R}^N$.*

In general, the projection of a belief state onto a set of epistemic features is lossy. However, as our experiments show (Section 6), features of small width will in general be enough for unambiguously representing optimal policies over a finite subset of the belief space, for natural sets of state features on natural problems.

Definition 4 (projectable)**.** *Let B be a finite set of belief states, $p\colon B \to \mathcal{P}(\mathcal{A})$ be a nondeterministic policy defined on B, and Φ be a set of epistemic features. Then p is said to be* projectable *onto Φ if there is a function $\pi\colon \{\Phi(b) \mid b \in B\} \to \mathcal{P}(\mathcal{A})$ satisfying $\forall b \in B : \emptyset \neq \pi(\Phi(b)) \subseteq p(b)$.*

In words, a policy is projectable if it can be faithfully represented by a function of the epistemic projection of the belief states, possibly at the price of breaking ties between actions arbitrarily. Note that if we are representing the optimal policy p^*, breaking ties between actions does not hinder optimality, since all actions are optimal at all belief states/histories.
Projectability can be straightforwardly decided in time polynomial in the numbers of belief states, states, epistemic features, and actions.

Example 4 (continued)**.** For $w = 1$, we have $(\mathrm{B}\,\mathsf{x})(b) = (\mathrm{B}\,\mathsf{y})(b) = (\mathrm{B}\,\neg\mathsf{x})(b) = (\mathrm{B}\,\neg\mathsf{y})(b) = \frac{1}{2}$ for $b \in \{\mathsf{b}^{\mathsf{unif}}, \mathsf{b}^{=}, \mathsf{b}^{\neq}\}$, hence

$$\Phi_F^{\vee,1}(\mathsf{b}^{\mathsf{unif}}) = \Phi_F^{\vee,1}(\mathsf{b}^{=}) = \Phi_F^{\vee,1}(\mathsf{b}^{\neq}) = (\frac{1}{2}, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}).$$

Hence p cannot be projected onto $\Phi_F^{\vee,1}$, since it prescribes different sets of actions for $\mathsf{b}^{\mathsf{unif}}$ and $\mathsf{b}^{\neq}$, which have the same projection onto $\Phi_F^{\vee,1}$.
On the other hand, p is projectable onto $\Phi_F^{\wedge,+,1} \cup \Phi_F^{\wedge,+,2} = \{\mathrm{B}\,\mathsf{x}, \mathrm{B}\,\mathsf{y}, \mathrm{B}(\mathsf{x} \wedge \mathsf{y})\}$, since the value v_3 of the third feature is already different on all three belief states (it is $\frac{1}{4}$ on $\mathsf{b}^{\mathsf{unif}}$, $\frac{1}{2}$ on $\mathsf{b}^{=}$, and 0 on $\mathsf{b}^{\neq}$). Hence we can define (for instance) $\pi((v_1, v_2, v_3)) = \{\mathsf{switch}_{\mathsf{x}}\}$ for $v_3 < \frac{1}{8}$, and $\pi((v_1, v_2, v_3)) = \{\mathsf{check}_{=}, \mathsf{noop}\}$ for $\frac{1}{8} \leq v_3$. Another projection (which breaks ties) is given by $\pi'((v_1, v_2, v_3)) = \{\mathsf{switch}_{\mathsf{x}}\}$ for $v_3 < \frac{1}{8}$, $\pi'((v_1, v_2, v_3)) = \{\mathsf{check}_{=}\}$ for $\frac{1}{8} \leq v_3 < \frac{3}{8}$, and $\pi'((v_1, v_2, v_3)) = \{\mathsf{noop}\}$ for $\frac{3}{8} < v_3$.
Observe that these definitions use only the value of the third feature, and hence p is also projectable onto $\Phi_F^{\wedge,+,2}$ alone. □

Obviously, projectability depends on the set of epistemic features. The following results explore this dependency.

Proposition 1. *Let B be a finite set of belief states, $p\colon B \to \mathcal{P}(\mathcal{A})$ be a nondeterministic policy defined on B, and F be a set of state features. Let $w < |F|$ be a nonnegative integer. If p is projectable onto $\Phi_F^{\vee,w}$ (resp. $\Phi_F^{\wedge,w}$), then it is projectable onto $\Phi_F^{\vee,w+1}$ (resp. $\Phi_F^{\wedge,w+1}$).*

Proof. This comes from the fact that the value of an epistemic clause (resp. term) of width w can be retrieved from the values of epistemic clauses (resp. terms) of width w. Indeed, for a propositional term t and a belief state b, we have $(\mathrm{B}\, t)(b) = (\mathrm{B}(t \wedge f))(b) + (\mathrm{B}(t \wedge \neg f))(b)$, for an arbitrary state feature $f \in F$. For a propositional clause c, writing t for a term equivalent to the negation of c, we have $(\mathrm{B}\, c)(b) = 1 - (\mathrm{B}\, t)(b) = 1 - \big((\mathrm{B}(t \wedge f))(b) + (\mathrm{B}(t \wedge$

[4] We use Latin letters for state features and formulas, and Greek letters for epistemic features and formulas.

$\neg f))(b)\big)$ for an arbitrary state feature $f \in F$, and hence $(\mathrm{B}\, c)(b) = -1 + (\mathrm{B}(c \vee \neg f))(b) + (\mathrm{B}(c \vee f))(b)$. □

Proposition 2. *Let F be an ordered set of n state features, and assume that the function $F : \mathcal{S} \to \mathbb{R}^n$ defined by $F(s) := (f(s) \mid f \in F)$ is injective. Then any policy over a finite set B of belief states is projectable onto $\Phi_F^{\vee,n}$ and onto $\Phi_F^{\wedge,n}$.*

Proof. For epistemic terms, this follows directly from the fact that the function $b \mapsto \Phi_F^{\wedge,n}(b)$ is also injective. Indeed, the probability of each state s in b can be retrieved from the projection as the value of the unique epistemic term of width n satisfied by s. The reasoning is similar for clauses: the probability of s can be retrieved as $1 - (\mathrm{B}\, c)(b)$, with c the unique epistemic clause of width n *not* satisfied by s. □

4.2 Representations

Given a finite set of belief states B, once a set of epistemic features Φ is fixed, over which a policy p^* is projectable, our aim is to compute an interpretable representation of p^*, using the epistemic features in Φ.

We view this problem as the problem of computing a classifier for a set of labelled examples, where for each belief state $b \in B$, the vector $\Phi(b)$ is an example, and the set of actions $p^*(b)$ is a set of possible labels for this vector. To keep the approach simple, we use only single-label classifiers, with the semantics that predicting any single action in p^* is correct.

Example 5 (continued)**.** Consider again $B = \{\mathsf{b}^{\mathsf{unif}}, \mathsf{b}^{=}, \mathsf{b}^{\neq}\}$, $\Phi = \Phi_F^{\wedge,+,1} \cup \Phi_F^{\wedge,+,2}$ ordered as $(\mathrm{B}\,\mathsf{x}, \mathrm{B}\,\mathsf{y}, \mathrm{B}\,\mathsf{x} \wedge \mathsf{y})$, and p. Then the examples are $(\frac{1}{2}, \frac{1}{2}, \frac{1}{4})$, $(\frac{1}{2}, \frac{1}{2}, \frac{1}{2})$, and $(\frac{1}{2}, \frac{1}{2}, 0)$, with respective sets of labels $\{\mathsf{check}_{=}, \mathsf{noop}\}$, $\{\mathsf{check}_{=}, \mathsf{noop}\}$, and $\{\mathsf{switch}_{\mathsf{x}}\}$. □

Observe that the set of examples does not cover the whole example space, so that the learned classifier has freedom to generalize them. We however require it to be correct on the given set of examples, that is, to provide an exact representation (modulo ties between optimal actions) of the given policy over the given set of belief states.

In principle, any classifier can be used. However, with our goal of building interpretable representations in mind, we will consider two classes of classifiers.

First, we call *linear representation* of p^* (over B and Φ) a matrix of real numbers $\{\theta^a_\varphi \mid \varphi \in \Phi, a \in \mathcal{A}\}$; the number θ^a_φ is called the *weight* of feature φ for action a. The policy π induced by such a representation is defined for all belief states $b \in \mathcal{B}$ by $\pi(b) := \mathrm{argmax}_{a \in \mathcal{A}} \sum_{\varphi \in \Phi} \theta^a_\varphi \cdot \varphi(b)$. Hence such representations can be seen as akin to α-vectors, but with only one vector per action, and compactly represented belief states (α-vectors can be seen as operating on projections with one feature per state).

The reason why we are interested in this representation is that it provides easy-to-interpret weights: the higher θ^a_φ, the more feature φ "promotes" action a, in the sense that the higher the value of φ on b, the more likely a is prescribed by the policy (all other things, namely, the value of other epistemic features on b, being equal).

Second, we call *decision tree representation* of p^* a decision tree branching on the values of epistemic features, and with actions at leaves. The policy π induced by such a representation is defined to map each belief state $b \in \mathcal{B}$, to the action at the leaf of the (unique) branch activated by b in the tree.

It is important to note that depending on the family of classifiers, an exact representation of a policy needs not always exist. For instance, a linear representation of a policy does not exist in general (though our experiments show that it often does), even if the policy is projectable onto the set of epistemic features. Contrastingly, if we do not bound the depth of trees nor the number of occurrences of an epistemic feature along a branch, a decision tree representation always exists, provided the policy is projectable.

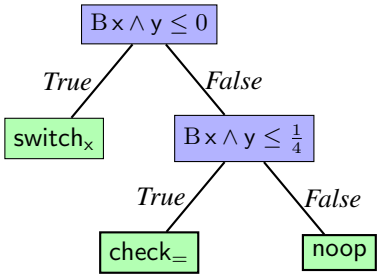


Figure 1: Decision tree representing an optimal policy in our POMDP example, branching on epistemic features.

Example 6 (continued)**.** The matrix of weights defined by $\theta^{\mathsf{check}_{=}}_{\mathrm{B}\,\mathsf{x}\wedge\mathsf{y}} = 1$ and $\theta^{\mathsf{switch}_{\mathsf{x}}}_{\mathrm{B}\,\mathsf{x}} = 0.1$ (all other weights being 0) is a linear representation of p over B and $\Phi_F^{\wedge,+,1} \cup \Phi_F^{\wedge,+,2}$. The policy π induced by this matrix indeed prescribes $\{\mathsf{switch}_{\mathsf{x}}\}$ at $\mathsf{b}^{\neq}$ (with value 0.05 against 0 for noop and $\mathsf{check}_{=}$) and $\{\mathsf{check}_{=}\}$ at $\mathsf{b}^{\mathsf{unif}}$ and $\mathsf{b}^{=}$ (with value 0.25 and 0.5, respectively, against 0.05 for $\mathsf{switch}_{\mathsf{x}}$ and 0 for noop). This can be interpreted as follows: the more the agent thinks it possible that the current state satisfies x and y, the more it should take action $\mathsf{check}_{=}$. On the other hand, the more it thinks it possible that the current state satisfies x, the more it should take action $\mathsf{switch}_{\mathsf{x}}$; however, the weight associated to the latter rule is much smaller than the one associated to the former, so that knowing $\mathsf{x} \wedge \mathsf{y}$ promotes more $\mathsf{check}_{=}$ than knowing x promotes $\mathsf{switch}_{\mathsf{x}}$.

On the other hand, there is no linear representation of p over B and $\Phi_F^{\wedge,+,2} = \mathrm{B}\,(\mathsf{x} \wedge \mathsf{y})$; indeed, such representation would necessarily give a 0 weight to all actions at $\mathsf{b}^{\neq}$ and, hence, could only represent a policy π with $\pi(\mathsf{b}^{\neq}) = \{\mathsf{check}_{=}, \mathsf{switch}_{\mathsf{x}}, \mathsf{noop}\}$.

Finally, Figure 1 depicts a decision tree representation of p (breaking the ties in favour of $\mathsf{check}_{=}$ at $\mathsf{b}^{\mathsf{unif}}$ and of noop at $\mathsf{b}^{=}$). □

Finally, executing a representation of a policy over Φ proceeds as follows. Given a POMDP, the agent first sets

$t := 0$ and computes $v_0 := \Phi(b_0)$. Then at each timestep t, it executes $a_t := \pi(v_t)$ and, upon receiving observation o_t, computes $v_{t+1} := \Phi(b_t \cdot a_t o_t)$.
Hence, like with α-vector representations, execution requires the POMDP model (transition and observation function) to be known. However, observe that the full belief state needs not be computed in general. Computing $\Phi(b_t \cdot a_t o_t)$ is a *belief tracking* task, which can be performed implicitly, for instance using operations over dynamic Bayesian networks representing the POMDP action and observation models [4]. Still, this is of course more costly than, for instance, following an arc in a finite-state controller. Hence there is a tradeoff between interpretability and efficiency of execution. Note the link with the work on knowledge-based programs [22, 38], which are representations of policies as programs branching on epistemic features (similar to ours) can yield very compact representations. Our work can indeed be seen as one using specific forms of KBPs as a target language for redescribing policies.

5 Computing Representations

We now present algorithms for computing epistemic representations starting from (non epistemic) policies. We start from a given policy p defined on the belief space, and restrict it to a finite set B of belief states (e.g., those reachable in at most t steps from b_0). In our experiments, we use the (near-optimal) policy computed from the POMDP by the off-the-shelf solver SARSOP, which comes with a representation by α-vectors, and restrict it to a large horizon.
We also fix a set of epistemic features. For this, we start from a given set F of state features (it is straightforward to define a meaningful one for practical problems). We then fix a width w, and choose whether to use clauses and/or terms, and whether to include negative features. These choices induce a set of epistemic features (for instance, $\Phi = \Phi_F^{\wedge,+,2}$).
We then proceed in two steps. First, we check whether p is projectable onto Φ (for B). As already mentioned, this can be done by a simple algorithm which goes through each belief state $b \in B$ in turn, and computes its projection $v := \Phi(b)$. If v is not mapped to a set of actions yet, then it gets mapped to $p(b)$; otherwise, if v is already mapped to a set of actions A, then it gets mapped to $A \cap p(b)$. Finally, if at any point a projection gets mapped to the empty set of actions, then the instance is not projectable onto Φ. In that case, we may, for instance, increase the width w, or include negative features if they were excluded.
Second, if the instance is projectable, then as a by-product of the first phase, we get a set of ordered pairs $\langle v_j, A_j \rangle_{j=1,\dots,m}$ that we are going to use as examples for a classifier.

5.1 Decision Tree Representations

For computing a representation as a decision tree, we view the set $\langle v_j, A_j \rangle_{j=1,\dots,m}$ as a training set for a learning algorithm. For this, for each $j \in \{1, \dots, m\}$, we first sample an action a uniformly at random from A, so as to build a set of ordered pairs $\langle v_j, a_j \rangle_{j=1,\dots,m}$, mapping a unique action to each vector. Such sampling is imposed by the fact that we focus on single-label algorithms, and of course, which action is sampled for each example impacts the final results (in terms of size, for instance).[5]
It is then a standard problem to learn a decision tree from such training set [33]. So as to get an exact representation of the policy, we impose no maximal depth on the learnt decision tree.

Linear constraints: (j ranges over $1, \dots, m$)

1. $\boldsymbol{m}_j \geq \sum_{\phi \in \Phi} \phi(v_j) \cdot \boldsymbol{\theta}_\phi^a + \boldsymbol{\eta}$ $\quad (\forall j, \forall a \in \mathcal{A} \setminus A_j)$
2. $\boldsymbol{m}_j \geq \sum_{\phi \in \Phi} \phi(v_j) \cdot \boldsymbol{\theta}_\phi^{a^*}$ $\quad (\forall j, \forall a^* \in A_j)$
3. $M \cdot \boldsymbol{k}_{j,a^*} + \sum_{\phi \in \Phi} \phi(v_j) \cdot \boldsymbol{\theta}_\phi^{a^*} \geq \boldsymbol{m}_j$ $(\forall j, \forall a^* \in A_j)$
4. $\sum_{a^* \in A_j} \boldsymbol{k}_{j,a^*} = |A_j| - 1$ $\quad (\forall j)$
5. $M \geq \boldsymbol{\eta} \geq 0$

Integrality constraints: $\boldsymbol{k}_{j,a^*} \in \{0, 1\}$ $\quad (\forall j, \forall a^* \in A_j)$

Objective: maximize $\boldsymbol{\eta}$

Figure 2: MILP for computing a linear representation.

5.2 Linear Representations

Now for computing a linear representation, we propose a formulation as a mixed-integer linear program (MILP; indeed a 0/1 linear program). This program takes advantage of the nondeterminism of the given policy, by choosing one or several actions in each set A_j so that the resulting policy is linear, if possible.
The MILP is given on Figure 2. Variables $\boldsymbol{\theta}_\varphi^a$ represent the weight θ_φ^a of feature φ for action a; variables $\boldsymbol{m}_j$ represent the highest value of an action of A_j in v_j (i.e., $\max_{a^*}(\sum \varphi(v_j) \cdot \theta_\varphi^{a^*})$); and $\boldsymbol{\eta}$ is a nonnegative margin variable (to be maximized).
Constraint 1 ensures that, for all j, the highest value $\boldsymbol{m}_j$ is greater than the value of all actions which are *not* prescribed by the policy. Constraints 2–4 together ensure that $\boldsymbol{m}_j$ receives the greatest value of all actions which *are* prescribed; for this, we use the standard trick with a "big M" constant M and 0/1 selector variables $\boldsymbol{k}_{j,a^*}$; since, as it turns out, the whole program is insensitive to multiplying by a constant, M can be chosen to be any positive value. Finally, Constraint 5 ensures that $\boldsymbol{\eta}$ is upper-bounded.
It is easy to see that this program is correct, in the following sense.

Proposition 3. *Let $\langle v_j, A_j \rangle_j$ be a set of instances, and M be any positive value. Then the optimal value $\boldsymbol{\eta}$ of the MILP of Figure 2 is strictly positive if and only if $\langle v_j, A_j \rangle_j$ has a linear representation; moreover, in this case, the values of variables $\boldsymbol{\theta}_\varphi^a$ define such a linear representation.*

[5] We plan to investigate multi-label algorithms in the future.

Solving a MILP is not known to be feasible in polynomial time. As a matter of fact, we conjecture that deciding whether a policy (given by an oracle) has a linear representation over given sets of belief states and epistemic features is an NP-complete problem.

6 Experiments

We now present experiments run on several standard problems from the POMDP literature. Our goal is to investigate to what extent our approach yields succinct and "interpretable" representations in reasonable time; to compare them with representations by FSCs; and to investigate the impact of the epistemic features chosen. In addition, we aim to investigate what epistemic width is enough for natural benchmarks from the POMDP literature.

6.1 Experimental Protocol

Each experiment consists first in choosing a POMDP and computing a near-optimal policy for it, using SARSOP.[6] Then several methods for redescribing this policy (output by SARSOP in an α-vector representation) are investigated: linear, decision tree, and FSC representations. For each method, we record the time to compute the redescription (without including the time for solving the POMDP) and its size: we define the size of a linear representation to be its number of nonzero weights, and that of a decision tree or an FSC to be its number of nodes.

Apart from SARSOP, all methods are run with our own implementation in Python, which itself uses `scipy` for solving MILPs and `sklearn` for computing decision trees. The experiments were run on servers with Intel Xeon processors, between 2.0 and 2.8 GHz and between 128 and 512 GB of RAM. For a given benchmark, they were all run on the same server, with one core dedicated to each computation. All computations were given a timeout of 1 h CPU time; SARSOP was run with target precision 10^{-3}, with $\gamma = 0.9$. For all methods, reachable belief states were gathered up to an horizon of 100.[7] For epistemic methods, we consider the following sets of epistemic features: $\Phi_i := \bigcup_{j=1}^{i}(\Phi_F^{\vee,j} \cup \Phi_F^{\wedge,j})$ and $\Phi_i^+ := \bigcup_{j=1}^{i}(\Phi_F^{\vee,+,j} \cup \Phi_F^{\wedge,+,j})$, for $i = 1, 2, 3$. So, the richest set of epistemic features includes all epistemic clauses and terms of width 1, 2, or 3, and the poorest includes only features of the form $\mathrm{B}\, f$ (for all $f \in F$).

For computing compact FSC representations, we generalized the algorithms proposed in [13]. Indeed, their approach, like ours, starts from a policy given by an oracle and from a finite set of belief states, but it requires the policy to be deterministic, and all observations to occur with a nonzero probability after any action is taken in any belief state. For lack of space, we do not give the details here, but simply note that it is a rather direct generalization of algorithm "policy2fsc" in [13]. All in all, it comes in two variants: starting from a policy pol, it computes an FSC representing a policy pol' such that, for all histories h, $\emptyset \neq pol'(h) \subseteq pol(h)$ (non-strict variant), or $pol'(h) = pol(h)$ (strict variant) holds.

[6] https://github.com/AdaCompNUS/sarsop

[7] Here it was enough for always fully representing the policy

6.2 Benchmarks

We tested our approach on four different families of POMDPs, with different features (in particular, reliable or noisy observations, fixed or combinatorial set of actions).

Wumpus. An agent must find a treasure as fast as possible on a 2D grid, while avoiding the monsters (Wumpuses). The agent always knows its own location and has deterministic actions for moving by one square in the four cardinal directions. No location can be occupied by a treasure and a Wumpus. The treasures and Wumpuses do not move and are at locations initially unknown, but the agent can "smell", which reveals whether there is a Wumpus on one of the 4 neighboring cells. Therefore, the agent may infer the locations of the Wumpuses by moving around and smelling, and eventually reach a treasure.

A Wumpus POMDP is further specified by the size of the grid, the number of Wumpuses and treasures (both known to the agent), and whether there might be a Wumpus around the agent in the initial state ("unsafe" instances; the initial belief state is otherwise uniform).

Rock-Sample. A rover navigates a 2D grid containing rocks (at known locations), each either good or bad, with equal probability. The rover can sample a rock at its current position, with a positive (resp. negative) reward if it was good (resp. bad); then the rock becomes bad. The rover can also check the status of a rock, with sensor accuracy decreasing exponentially with its distance to the rock. Reaching an exit (at one of known locations) yields a positive reward. Finally, the robot can move deterministically by one square in each cardinal direction, and always knows its position.

A Rock-Sample POMDP is further specified by the size of the grid and the positions of the rocks and exits. An interesting feature of this problem is noisy observations, which imply a large number of belief states can be reached even on small grids.

Mastermind. Our third benchmark is Mastermind, the famous code-breaking game. A Mastermind POMDP is specified by the number of positions, colors, possible repetitions in the secret code, and the number of attempts allowed. An interesting feature of this benchmark is the combinatorial set of actions, which grows with the size of the instance.

Minesweeper. Finally, we used the famous Minesweeper game. A Minesweeper POMDP is specified by the size of the grid and the number of mines.

6.3 Results

We start with the size of the representations. Figure 3 presents the results for each family of benchmarks and four representative approaches: a linear representation over Φ_1 ("epistemic-1-1-linear-withneg"), a decision tree representation over Φ_3^+ ("epistemic-1-2-3-1-2-3-tree-noneg"), and an FSC computed with the strict ("fsc-

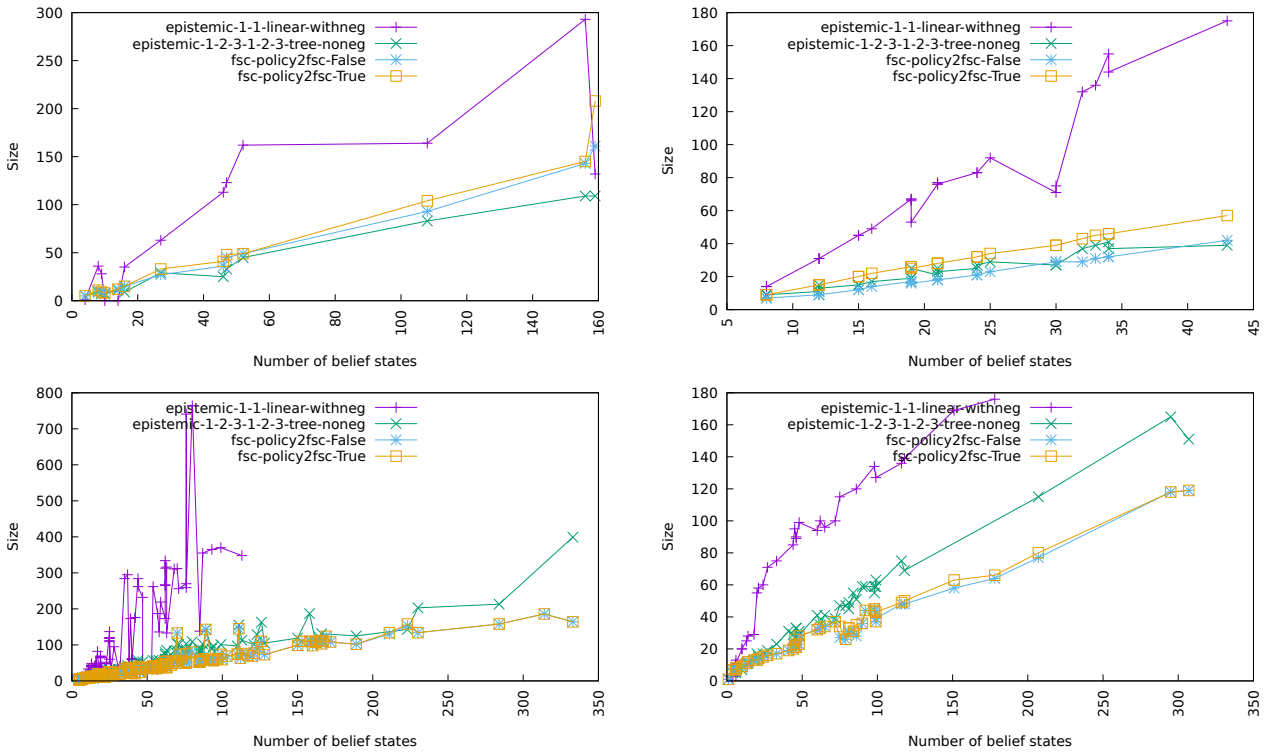


Figure 3: Size of representations for several methods. From top-left to bottom-right: Minesweeper, Rocksample, Mastermind, Wumpus

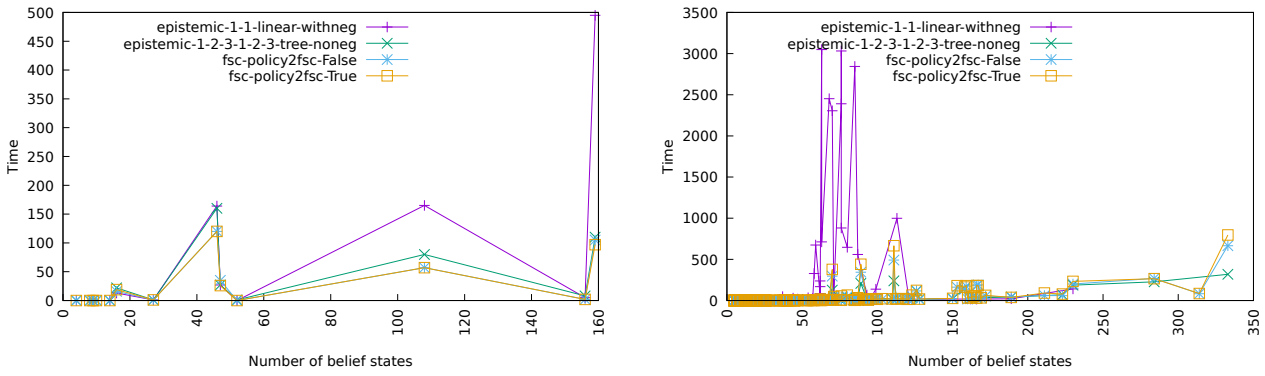


Figure 4: Time (seconds) for computing representations. Left: Minesweeper. Right: Mastermind

policy2fsc-True") or non-strict ("fsc-policy2fsc-False") variant. The x-axis is ordered by the number of belief states in the considered set B.

We observe that the representations computed by all approaches grow linearly in size with the number of belief states. Though the size is not defined in the same manner for all approaches, FSCs seem to yield the most compact models, but decision trees with positive features of size up to 3 seem to be on par, and even better on Minesweeper.

We also investigated the impact of the width of epistemic features (not displayed for lack of space). As the width grows, so does the number of features, and the size of the linear representations grows substantially as well. Moreover, eventually the MILP becomes too large and computation fails, while decision trees and FSCs can still be computed in reasonable time. Moreover, the decision tree representations tend to become smaller with larger width. This is consistent with the fact that they offer a greater expressive power (Proposition 1) and hence, allow for more decisive splits to be found. Finally, whether negative features are included or not does not seem to have much impact on the size (while restricting to positive features obviously makes the representations easier to understand).

As concerns computational time, Figure 4 presents results for two representative families of POMDPs. For both linear and decision tree models, the time needed to compute our

	Width = 1				Width = 1-2				Width = 1-2-3			
	Noneg		Withneg		Noneg		Withneg		Noneg		Withneg	
	np	nr	np	nr	np	nr	np	nr	np	nr	np	nr
M	0.72%	0.72%	0.72%	0.72%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%	0.0%
W	0.0%	2.31%	0.0%	2.31%	0.0%	0.53%	0.0%	0.53%	0.0%	0.53%	0.0%	0.0%

Table 1: Projectability and representability across epistemic settings for Mastermind and Wumpus

epistemic representations is in general of the same order of magnitude as that required for FSCs, though the small size of the problems do not allow to infer a definite trend.
Again, for lack of space, we do not depict results about our methods with different epistemic widths. However, for all families of POMDPs, the computational time required for our representations increases with the epistemic width and thus the number of features. This increase is much steeper for linear representations than for decision trees, as already evoked when analyzing the size. As concerns decision trees, even when the set of epistemic features grows, the trees select only a small subset of them, each used only once. This suggests that while the tree has access to a richer feature space, it does not necessarily require more splits to reach an effective decision boundary. In contrast, when fewer features are available but are used or reused more frequently, the tree requires additional splits to refine its decision boundaries. This, in turn, increases the number of computational rounds needed to build a classifier that fits the data perfectly.

Finally, we have analyzed the proportion of POMDPs that are either not projectable (np) or projectable but not linearly representable (nr). As Table 1 shows, there are very few instances of np/nr, and they appear in only two benchmarks. This indicates that, in most cases, a low epistemic width suffices to project a policy. Moreover, a representation can usually be computed.

7 Conclusion

We proposed an approach for redescribing POMDP policies onto sets of epistemic features, with the goal of improving interpretability. Our experiments show that our approach has the potential to yield representations on par with finite-state controllers in terms of size and computation time.
In the near future, we intend to investigate how it scales to larger problems, in particular by using factored representations of POMDPs. We also plan to investigate how to enrich FSCs with epistemic features, trying to get the best of both. Finally, our long-term objective is to perform reinforcement learning directly on the epistemic descriptions.

Acknowledgements

This work has been funded by the French National Agency for Reasearch (ANR) through grant ANR-22-CE23-0029.

References

[1] R. I. Bahar, E. A. Frohm, C. M. Gaona, G. D. Hachtel, E. Macii, A. Pardo, and F. Somenzi. Algebraic decision diagrams and their applications. *Formal Methods Syst. Des.*, 10(2/3):171–206, 1997.

[2] B. Bakker. Reinforcement learning with long short-term memory. In *Advances in Neural Information Processing Systems*, volume 14. MIT Press, 2001.

[3] T. Bewley, J. Lawry, and A. Richards. Modelling agent policies with interpretable imitation learning, 2020.

[4] B. Bonet and H. Geffner. Factored probabilistic belief tracking. In S. Kambhampati, editor, *Proc. IJCAI 2016*, pages 3045–3052. IJCAI/AAAI Press, 2016.

[5] T. Chakraborti, A. Kulkarni, S. Sreedharan, D. E. Smith, and S. Kambhampati. Explicability? Legibility? Predictability? Transparency? Privacy? Security? The Emerging Landscape of Interpretable Agent Behavior. In *Proc. ICAPS 2019*, pages 86–96. AAAI Press, 2019.

[6] V. G. Costa, J. Pérez-Aracil, S. Salcedo-Sanz, and C. E. Pedreira. Evolving interpretable decision trees for reinforcement learning. *Artif. Intell.*, 327:104057, 2024.

[7] F. Doshi-Velez and B. Kim. Towards a rigorous science of interpretable machine learning, 2017.

[8] M. Du, N. Liu, and X. Hu. Techniques for interpretable machine learning, 2019.

[9] J. Gajcin, R. Nair, T. Pedapati, R. Marinescu, E. Daly, and I. Dusparic. Contrastive explanations for comparing preferences of reinforcement learning agents, 2021.

[10] L. Gao and L. Guan. Interpretability of machine learning: Recent advances and future prospects, 2023.

[11] L. H. Gilpin, D. Bau, B. Z. Yuan, A. Bajwa, M. A. Specter, and L. Kagal. Explaining explanations: An approach to evaluating interpretability of machine learning. *CoRR*, abs/1806.00069, 2018.

[12] N. Grandien, Q. Delfosse, and K. Kersting. Interpretable end-to-end neurosymbolic reinforcement learning agents, 2024.

[13] M. Grześ, P. Poupart, X. Yang, and J. Hoey. Energy efficient execution of POMDP policies. *IEEE Transactions on Cybernetics*, 45(11):2484–2497, 2015.

[14] E. Hansen. An improved policy iteration algorithm for partially observable MDPs. In *NIPS'97*, pages 1015–1021. MIT Press, 1998.

[15] E. A. Hansen and Z. Feng. Dynamic programming for POMDPs using a factored state representation. In *Proc. AIPS, 2000*, pages 130–139. AAAI, 2000.

[16] J. Hoey, R. St-Aubin, A. J. Hu, and C. Boutilier. SPUDD: stochastic planning using decision diagrams. In K. B. Laskey and H. Prade, editors, *Proc. UAI 1999*, pages 279–288. Morgan Kaufmann, 1999.

[17] J. Huang, P. P. Angelov, and C. Yin. Interpretable policies for reinforcement learning by empirical fuzzy sets. *Engineering Applications of Artificial Intelligence*, 91:103559, 2020.

[18] T. Huber, K. Weitz, E. André, and O. Amir. Local and global explanations of agent behavior: Integrating strategy summaries with saliency maps. *Artificial Intelligence*, 301:103571, 2021.

[19] H. Kohler, R. Akrour, and P. Preux. Interpretable decision tree search as a Markov decision process, 2024.

[20] H. Kohler, Q. Delfosse, R. Akrour, K. Kersting, and P. Preux. Interpretable and editable programmatic tree policies for reinforcement learning. *CoRR*, abs/2405.14956, 2024.

[21] H. Kurniawati, D. Hsu, and W. S. Lee. SARSOP: efficient point-based POMDP planning by approximating optimally reachable belief spaces. In *Robotics: Science and Systems IV 2008*. The MIT Press, 2008.

[22] J. Lang and B. Zanuttini. Probabilistic knowledge-based programs. In *Proc. IJCAI 2015*, IJCAI'15, page 1594–1600. AAAI Press, 2015.

[23] P. Linardatos, V. Papastefanopoulos, and S. Kotsiantis. Explainable AI: A review of machine learning interpretability methods. *Entropy*, 23(1), 2021.

[24] C. Lu, R. Shi, Y. Liu, K. Hu, S. S. Du, and H. Xu. Rethinking transformers in solving POMDPs. In *Proc. ICML 2024*. OpenReview.net, 2024.

[25] B. Mazoure, T. Doan, T. Li, V. Makarenkov, J. Pineau, D. Precup, and G. Rabusseau. Representation of reinforcement learning policies in reproducing kernel Hilbert spaces, 2020.

[26] D. Meli, A. Castellini, and A. Farinelli. Learning logic specifications for policy guidance in POMDPs: An inductive logic programming approach. *Journal of Artificial Intelligence Research*, 79:725–776, 2024.

[27] N. Meuleau, L. Peshkin, K.-E. Kim, and L. Kaelbling. Learning finite-state controllers for partially observable environments. In *Proc. UAI 1999*, pages 427–436, 1999.

[28] S. Miura and S. Zilberstein. A unifying framework for observer-aware planning and its complexity. In *Proc. UAI 2021*, volume 161 of *Proceedings of Machine Learning Research*, pages 610–620. AUAI Press, 2021.

[29] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. A. Riedmiller. Playing Atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013.

[30] T. Ni, B. Eysenbach, and R. Salakhutdinov. Recurrent model-free RL can be a strong baseline for many POMDPs. In *Proc. ICML 2022*, volume 162 of *Proceedings of Machine Learning Research*, pages 16691–16723. PMLR, 2022.

[31] J. Pineau, G. Gordon, and S. Thrun. Point-based value iteration: an anytime algorithm for POMDPs. In *Proc. IJCAI 2003*, page 1025–1030. Morgan Kaufmann Publishers Inc., 2003.

[32] P. Poupart. *Exploiting Structure to Efficiently Solve Large Scale Partially Observable Markov Decision Processes*. PhD thesis, University of Toronto, 2005.

[33] J. R. Quinlan. *C4.5: programs for machine learning*. Elsevier, 1993.

[34] L. Saulières, M. C. Cooper, and F. D. de Saint Cyr. Backward explanations via redefinition of predicates, 2024.

[35] T. Smith and R. Simmons. Heuristic search value iteration for POMDPs. In *Proc. UAI 200'*, UAI '04, page 520–527, Arlington, Virginia, USA, 2004. AUAI Press.

[36] T. Smith and R. G. Simmons. Point-based POMDP algorithms: Improved analysis and implementation. *CoRR*, abs/1207.1412, 2012.

[37] N. Topin, S. Milani, F. Fang, and M. Veloso. Iterative bounding MDPs: Learning interpretable policies via non-interpretable methods, 2021.

[38] B. Zanuttini, J. Lang, A. Saffidine, and F. Schwarzentruber. Knowledge-based programs as succinct policies for partially observable domains. *Artificial Intelligence*, 288, 2020.

Session 4 : Planification hiérarchique et temporelle

Optiplan : Encodage CSP pour planification HTN-POCL optimale

Oleksandr Firsov, Humbert Fiorino, Damien Pellier

Université Grenoble Alpes - LIG

1er mars 2025

Résumé

La planification HTN est une approche efficace pour résoudre des problèmes complexes du monde réel. Un défi croissant est d'assurer la flexibilité des plans, car une certaine liberté d'exécution les rend plus résistants aux changements. Cette flexibilité est obtenue en combinant HTN avec POCL. Dans cet article, nous présentons Optiplan, une méthode d'encodage novatrice qui modélise ces problèmes sous forme de CSP et, pour la première fois, aborde l'optimisation des solutions.

Mots-clés

Planification, IA, partiellement ordonné, HTN, encodage, CSP

Abstract

HTN planning is an effective approach for solving complex real-world problems. A growing need in this context is for plan flexibility, as allowing some degree of freedom in task execution makes plans more resilient to changes. This flexibility is achieved by combining HTN with POCL planning. In this paper, we introduce Optiplan, a novel encoding method that models these planning problems as CSPs and also, for the first time, tackles the question of solution optimization.

Keywords

Planning, AI, partially ordered, HTN, encoding, CSP

1 Introduction

Au cours des dernières années, les approches de planification hiérarchique ont suscité un intérêt de recherche considérable, les réseaux de tâches hiérarchiques (Hierarchical Task Networks, HTNs) étant largement adoptés dans divers domaines pratiques[5]. La popularité des HTNs repose sur leur structure hiérarchique, qui leur permet de capturer des informations riches, spécifiques à un domaine, et d'utiliser ces connaissances pour guider la planification de manière efficace.

Au coeur de la planification HTN est l'idée de décomposer les problèmes complexes en une hiérarchie de tâches. Ce processus commence par des tâches abstraites de haut niveau, qui sont progressivement affinées en sous-tâches plus concrètes à l'aide de "recettes" préétablies, jusqu'à atteindre des actions primitives exécutables. Cette approche présente deux avantages majeurs : d'une part, elle améliore la scalabilité des problèmes, et d'autre part, elle offre aux utilisateurs un meilleur contrôle sur le processus de planification.

La majorité des planificateurs HTN se concentrent sur la génération de plans de solution totalement ordonnés en utilisant diverses techniques, telles que l'encodage SAT [3, 20, 19], l'encodage CSP [22, 21], la recherche heuristique [13, 6] ou encore la traduction du problème en planification classique STRIPS [1, 4, 9].

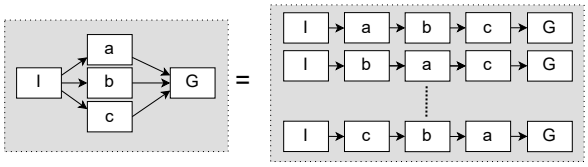


FIGURE 1 – À gauche, un plan partiellement ordonné. À droite, 3 des 6 plans totalement ordonnés déductibles.

Bien que ces approches se soient révélées efficaces [19], les plans obtenus suivent un ordre d'actions fixe, ne laissant aucune marge d'ajustement pendant l'exécution. En d'autres termes, ils manquent de flexibilité, c'est-à-dire de la capacité à s'adapter à des circonstances changeantes (voir Fig. 1). Cela peut constituer une limitation majeure dans des applications où un certain degré de liberté est essentiel pour gérer l'incertitude environnementale et les risques potentiels, comme par exemple dans la planification de missions pour robots [16] ou la planification industrielle [23].

Plusieurs approches ont été proposées pour introduire de la flexibilité dans la planification hiérarchique, comme le documentent [15, 6]. Ces approches combinent la structure hiérarchique avec le concept de liens causaux à ordre partiel (Partial Order Causal Links, POCL) [17, 18], qui suit le principe du "moindre engagement", c'est-à-dire que le planificateur ne spécifie l'ordre des tâches que lorsque cela est nécessaire. Pour faciliter la lecture, nous ferons référence à la planification hiérarchique partiellement ordonnée sous le nom de *planification HTN-POCL*.

Dans cet article, nous présentons Optiplan - une nouvelle méthode d'encodage capable de produire des plans partiellement ordonnés avec un chemin critique optimal (c'est-à-dire des plans où la plus longue chaîne d'activités dépendantes est minimisée). Il exploite des techniques CSP et s'appuie sur les avancées récentes en encodages pour la pla-

nification totalement ordonnée. À notre connaissance, Optiplan est la première approche capable de générer des plans HTN-POCL optimaux et la première à utiliser un CSP, ou toute autre forme d'encodage, dans ce contexte.

L'article est structuré comme suit. Tout d'abord, nous introduisons les concepts associés à la planification HTN-POCL. Ensuite, nous présentons en détail notre approche, en abordant la structure de données utilisée pour représenter l'espace des plans, la méthode d'encodage employée ainsi que la méthodologie d'extraction d'un plan de solution. Par la suite, nous réalisons une analyse comparative d'Optiplan par rapport à d'autres planificateurs utilisant la planification HTN partiellement ordonnée, en évaluant la performance en termes de qualité des plans (i.e., la longueur du chemin critique du plan généré) et d'agilité du planificateur (i.e., le temps nécessaire pour produire un plan). Enfin, nous discutons des travaux connexes.

2 Planification HTN partiellement ordonnée

Cette section introduit les fondements de la planification HTN-POCL.

2.1 Actions, Méthodes et Tâches

Une *proposition* logique est un objet atomique, et un *état* s est un ensemble cohérent de propositions. Une action est un triplet *a = (name(a), pre(a), eff(a))* où *name(a)* est l'identifiant de a, *pre(a)* définit les *préconditions* qui doivent être satisfaites pour appliquer l'action, et *eff(a)* définit les *effets* qui s'appliquent à l'état après l'exécution de a : $\mathit{eff}(a) = (\mathit{eff}^+(a), \mathit{eff}^-(a))$ avec $\mathit{eff}^+(a) \cap \mathit{eff}^-(a) = \emptyset$.

Une action a est *applicable* à un état s si $pre(a) \subseteq s$. L'état résultant s' de l'application de a à un état s est défini comme suit : $s' = \gamma(s, a) = (s \setminus \mathit{eff}^-(a)) \cup \mathit{eff}^+(a)$.

L'application d'une séquence d'actions est définie récursivement par $\gamma(s, []) = s$ et $\gamma(s, [a_0, a_1, \ldots, a_n]) = \gamma(\gamma(s, a_0), [a_1, \ldots, a_n])$.

Un *réseau de tâches* w est un couple $(T, \prec)$ où T est une séquence d'identifiants de tâches $[\ldots, t_i, \ldots]$, et $\prec$ est un ensemble de contraintes d'ordre sur T : t_i *précède* t_j si et seulement si $(t_i < t_j) \in \prec$.

Il est important de noter que la séquence de tâches T est totalement arbitraire, n'a aucun lien direct avec $\prec$, et est simplement utilisée comme une commodité pour construire la structure de données d'encodage d'Optiplan, à savoir l'*arbre de décomposition HTN-POCL* (voir section suivante).

Une *méthode* $m = (name(m), (T, \prec))$ est un couple où *name(m)* est l'identifiant de m, et $(T, \prec)$ est un réseau de tâches décrivant comment une méthode est accomplie.

Une action a *peut accomplir* une tâche t si *name(a)* $= t$, et une méthode m *peut accomplir* une tâche t si *name(m)* $= t$. Il est important de noter qu'une même tâche t peut être accomplie par différentes méthodes ou actions.

Nous notons $R(t)$ l'ensemble de toutes les actions et/ou méthodes qui peuvent accomplir t. Nous appelons ces actions et/ou méthodes les *accomplisseurs* de t. [1]

La manière dont ces accomplisseurs sont sélectionnés pour effectivement accomplir une tâche donnée sera expliquée plus tard dans cette section. Nous noterons la *taille* d'une méthode m (c'est-à-dire le nombre de tâches dans son réseau de tâches) par $\sharp m$. La taille de toute action est égale à 1.

2.2 Liens causaux, Plans partiels, Menaces et Défauts

Un *lien causal* est un triplet (a, p, a'), où p est une proposition, a est une action produisant p (i.e., $p \subseteq \mathit{eff}^+(a)$), et a' est une action ayant p parmi ses préconditions. Les liens causaux sont notés $a \xrightarrow{p} a'$, et ils indiquent les dépendances producteur/consommateur entre les actions d'un plan. Les liens causaux imposent des contraintes sur l'ordre des actions, puisque $a \xrightarrow{p} a'$ requiert que a précède a'.

Un *plan partiel* π est un triplet (w, α, C) où :

— w est un réseau de tâches $(T, \prec)$ qui définit les tâches et les contraintes d'ordre de π,

— α est une correspondance **partielle** de $t \in T$ vers une action ou une méthode accomplissant t. Si t est accompli par une action, alors t est considéré comme *primitif*, sinon il est *non primitif* (i.e., que soit $\alpha(t) = m$, soit $\alpha(t)$ est indéfini),

— C est un ensemble de liens causaux $a \xrightarrow{p} a'$, où a et a' sont deux actions accomplissant respectivement t et t' dans T.

Une *menace* sur un lien causal $a \xrightarrow{p} a'$ est une action a'' dans π telle que $p \in \mathit{eff}^-(a'')$ (i.e., que a'' détruit p), et où $\prec$ n'inclut pas les contraintes d'ordre $(a'' < a)$ ou $(a' < a'')$. En d'autres termes, a'' constitue une menace s'il est possible qu'il se produise entre a et a', rendant ainsi l'exécution de a' impossible.

Un *objectif ouvert* (open goal) dans π est une précondition d'une action a qui n'est soutenue par aucun lien causal. Plus formellement, $p \in pre(a)$ est un objectif ouvert s'il n'existe pas d'action a' dans le plan partiel telle que $a' \xrightarrow{p} a \in C$.

Un *défaut* dans π est soit une tâche non primitive, soit une menace, soit un objectif ouvert.

Une solution à un problème de planification est un plan partiel *sans défaut*. Ce plan peut également être représenté sous forme d'un graphe orienté acyclique (DAG) avec l'action initiale I comme source et l'action objectif G comme sommet terminal (voir Figure 1). Dans ce DAG, le chemin critique correspond au plus court chemin de I à G.

2.3 Problème de planification

Un *problème de planification HTN-POCL* P est défini comme un tuple $(\mathcal{L}, \mathcal{T}, \mathcal{A}, \mathcal{M}, \pi_0, s_0, g)$ où :

— $\mathcal{L}$ est un ensemble de propositions,

1. Sans perte de généralité, nous ne considérons pas ici les méthodes m ayant des préconditions $pre(m)$, car elles sont équivalentes à des méthodes telles qu'il existe une tâche $t \in T_m$ qui précède toutes les autres tâches, où une action a accomplit t, et où $\mathit{pre}(m) = \mathit{pre}(a)$, $\mathit{eff}^+(a) \cup \mathit{eff}^-(a) = \emptyset$.

Algorithm 1 SOLVE(P)

Soit f un défaut dans π
if f est une tâche non primitive t **then**
  $\pi' =$ DECO(π, t)
  return SOLVE(P)
else if f est un objectif ouvert p dans l'action a_j de π **then**
  $a_i =$ *choisirProducteur*(a_j, p) `// arbitraire`
  if $a_i = \emptyset$ **then**
    return FAILURE
  else
    $\pi' = (w, \alpha, C \cup (a_i \xrightarrow{p} a_j))$
    $P' = (\mathcal{L}, \mathcal{T}, \mathcal{A}, \mathcal{M}, \pi', s_0, g)$
    return SOLVE(P')
  end if
else
  return π
end if

— $\mathcal{T}$ est un ensemble de noms de tâches,
— $\mathcal{A}$ est un ensemble d'actions,
— $\mathcal{M}$ est l'ensemble des méthodes,
— π_0 est le plan initial à décomposer,
— s_0 et g sont deux sous-ensembles de $\mathcal{L}$ représentant respectivement l'état initial du problème et l'objectif à atteindre, s'il existe.

Afin de rester cohérent avec la littérature POCL, le plan initial π_0 est défini comme $(w_0, \emptyset, \emptyset)$, où w_0 contient les tâches abstraites initiales ainsi que deux tâches spéciales, t_0 et t_∞, qui représentent respectivement l'état initial et l'état objectif.
La tâche t_0 est accomplie par une action a_0, où *pre*$(a_0) = \emptyset$ et *eff*$(a_0) = s_0$, et la tâche t_∞ est accomplie par une action a_∞, où *pre*$(a_\infty) = g$ et *eff*$(a_\infty) = \emptyset$. Par conséquent, t_0 et t_∞ sont respectivement ordonnées comme la première et la dernière tâche de w_0.

2.4 Procédure de résolution et raffinement du plan

Un problème de planification P est résolu par une procédure récursive SOLVE(P) (voir Algo. 1), qui corrige les défauts sans introduire de menaces. Alors qu'un objectif ouvert est accompli en établissant un lien causal, un défaut introduit par une tâche abstraite t_i est résolu par un raffinement ou une concrétisation du plan partiel par rapport à t_i. Cette procédure de raffinement est appelée DECO(π, t_i) (voir Algo. 2), car elle applique une décomposition de la méthode associée à t_i sur π.
Lorsque DECO(P) applique une méthode, elle introduit des sous-tâches dans le réseau de tâches. La fonction *integrerOrdonnancement* indique que les contraintes d'ordre des sous-tâches doivent être incorporées dans les contraintes du réseau de tâches. De plus, si la tâche parente possède des relations d'ordre, celles-ci sont héritées par les sous-tâches. Ainsi, DECO(P) affine les plans partiels en leurs versions plus concrètes. Cependant, cette procédure ne modifie pas

Algorithm 2 DECO(π, t_i)

Soit $r \in R(t_i)$
if r est une action a **then**
  if r est une **menace then**
    return FAILURE
  else
    return $\pi' = (w, \alpha \cup \alpha(t_i) = a, C)$
  end if
else if r est une méthode m **then**
  $\alpha' = \alpha \cup \alpha(t_i) = m$
  $T' = T \cup T^m$
  $\prec' =$ *integrerOrdonnancement*$(\prec, \prec^m)$
  $w' = (T', \prec')$
  return $\pi' = (w', \alpha', C)$
end if

les liens causaux, ce qui signifie que bien que les plans obtenus soient exempts de menaces, ils ne sont pas encore parfaits. C'est SOLVE(P) qui introduit les liens causaux, aboutissant ainsi à une solution. Cette procédure est définie récursivement comme suit :

3 L'approche Optiplan

Dans cette section, nous décrivons les principes d'encodage de SOLVE(P) en un problème de satisfaction de contraintes (Constraint Satisfaction Problem, CSP).
L'espace de recherche d'Optiplan est défini par un *arbre de décomposition HTN-POCL* (arbre HiPOD), qui représente la structure hiérarchique des décompositions possibles d'un plan. L'arbre HiPOD est constitué de couches qui organisent les actions et les méthodes sous une structure arborescente, chaque couche représentant un niveau de décomposition. Cette approche est inspirée de [20], bien que dans notre cas, les éléments de chaque couche ne soient pas ordonnés, et nous explicitons les préconditions ainsi que les liens causaux.
La procédure d'encodage suit une recherche *en largeur* sur l'ensemble des décompositions possibles, en partant des noeuds racines correspondant aux tâches initiales. Elle est conçue comme un processus *incrémental*, ce qui signifie que HiPOD est construit jusqu'à une certaine profondeur k à chaque étape de décomposition. L'arbre est ensuite soumis à un solveur CSP. Si le solveur ne retourne pas de solution à cette profondeur, le niveau de HiPOD est incrémenté et la procédure est répétée jusqu'à ce qu'une solution soit trouvée ou qu'une limite arbitraire $k_{\max}$ soit atteinte, suivant [3].
Ainsi, Optiplan a la propriété suivante : étant donné un problème de planification HTN-POCL P et une profondeur k, il existe une correspondance biunivoque entre l'ensemble des solutions de P de profondeur $\leq k$ et l'ensemble des solutions du CSP. En d'autres termes, SOLVE(P) et notre encodage sont *congruents*. À partir d'une solution du CSP, si elle existe, la correspondance permet d'obtenir une solution à P. Si aucune solution n'existe, alors aucun plan de profondeur $\leq k$ n'existe.

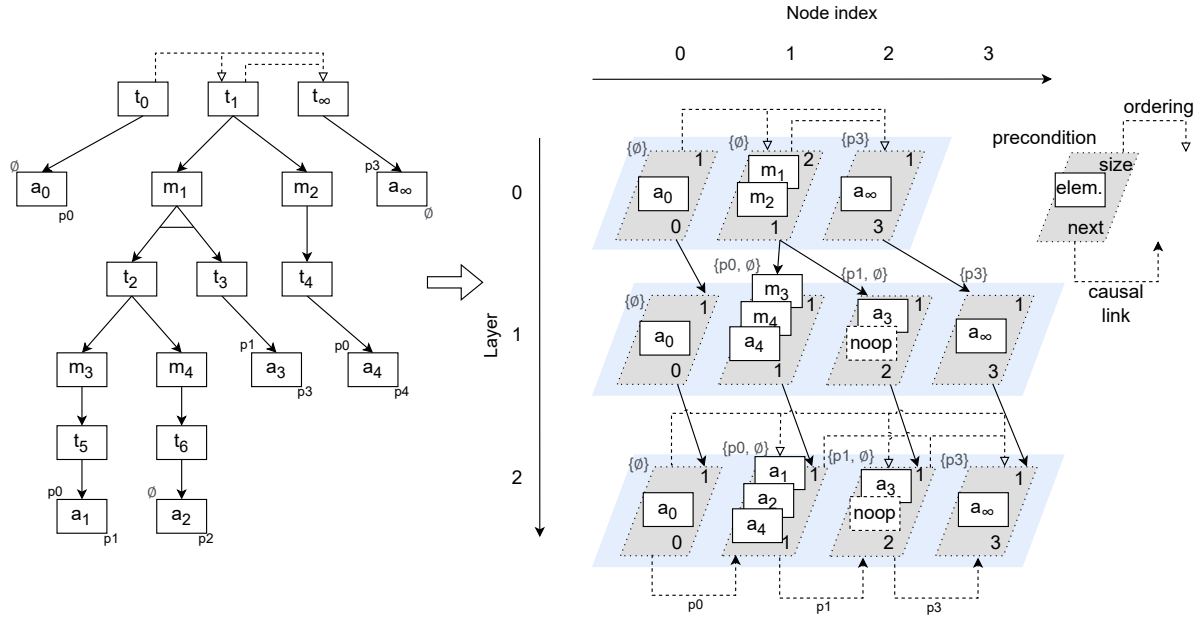


FIGURE 2 – À gauche, une décomposition représentée sous forme d'arbre ET/OU, où t représente une tâche, a une action et m une méthode. p représente une proposition : si elle est placée au-dessus d'un noeud, elle correspond à une précondition ; sinon (en dessous d'un noeud), elle représente un effet. Les flèches sortantes des tâches représentent les relations "est accompli par", et les flèches sortantes des méthodes indiquent les décompositions en sous-tâches. À droite, l'arbre HiPOD correspondant. Les éléments empilés forment $cell(i, j)$ où i correspond à la couche L_i, et j correspond à l'indice de la cellule dans le tableau L_i.

Cela garantit qu'Optiplan est à la fois *correct* et *complet*. La procédure est correcte, car chaque solution retournée par le solveur CSP est un plan valide qui satisfait toutes les contraintes du problème de planification HTN-POCL. Elle est complète, car si une solution de profondeur $\leq k$ existe, le solveur CSP est assuré de la trouver, garantissant qu'aucune solution valide n'est omise.

Dans la suite, nous détaillons le processus de construction de HiPOD et fournissons les règles d'encodage du CSP.

3.1 Arbre de décomposition HTN-POCL

L'arbre HiPOD organise la décomposition hiérarchique d'un problème de planification en plusieurs niveaux, représentés comme une séquence de couches hiérarchiques $[L_0, \ldots, L_i, \ldots, L_n]$. Chaque *couche hiérarchique* est un tableau de cellules, où chaque cellule contient un ensemble d'actions et de méthodes, désigné comme le domaine de la cellule. Nous notons la j-ième cellule de la couche L_i sous la forme $cell(i, j)$. Les cellules peuvent également contenir *noop*, une action sans effets ni préconditions (voir Fig. 2).

Chaque cellule possède une liste de préconditions associées. Comme les actions au sein d'une cellule peuvent avoir des préconditions différentes, les préconditions des cellules sont variables et leur domaine est défini par $pre_i(a), a \in cell$, où i est un indice dans la liste des préconditions. Un exemple de préconditions de cellules est illustré en Fig. 3.

Chaque précondition de cellule doit être soutenue par un lien causal. Par conséquent, chaque couche hiérarchique contient une liste de *liens causaux*, qui capture les dépendances causales potentielles entre les cellules. De même, chaque couche dispose d'une liste de *contraintes d'ordre*, qui définit l'ordre partiel des actions dans la couche. Le niveau i d'une couche hiérarchique L_i indique le nombre de décompositions qui doivent être appliquées depuis le réseau de tâches initial w_0 pour atteindre une action dans les cellules de L_i.

La première couche L_0 est construite directement à partir de w_0 comme suit : pour chaque tâche de w_0, nous créons une cellule correspondante dans L_0, où l'ordre des cellules est arbitraire. Le domaine de chaque cellule est rempli avec les actions et les méthodes capables d'accomplir la tâche correspondante. Par exemple, dans la Figure 2, la première cellule de L_0 est $(cell(0, 0) \in a_0)$ et elle correspond à t_0. Pour la tâche t_1 de w_0, nous avons $cell(0, 1) \in m_1, m_2$. Enfin, $cell(0, 2) \in a_\infty$ est obtenu à partir de t_∞.

Le réseau de tâches initial peut imposer certaines contraintes d'ordre entre les tâches. Ces contraintes sont enregistrées dans la liste des contraintes d'ordre pour L_0.

Chaque couche suivante L_{i+1} est construite de manière incrémentale à partir des cellules de L_i. Chaque $cell(i, j)$ dans L_i est un parent d'un certain nombre de cellules dans L_{i+1}. Le nombre de cellules enfants est déterminé par le plus grand nombre de sous-tâches dans toute méthode pré-

sente dans la cellule. Nous définissons ce nombre comme $size(i,j)$, qui est calculé comme suit :

$$size(i,j) = max_{\sharp} e | \forall e \in cell(i,j) \quad (1)$$

Les enfants de $cell(i,j)$ sont indexés dans L_{i+1} en calculant leur position relative au parent. Nous notons la position du premier enfant de $cell(i,j)$ sous la forme $next(i,j)$, qui est calculée récursivement comme suit :

$$next(i,j) = \begin{cases} 0 & \text{si } j = 0 \\ next(i,j-1) + size(i,j-1) & \text{sinon} \end{cases} \quad (2)$$

Ainsi, les enfants de $cell(i,j)$ sont les cellules positionnées dans l'intervalle $[next(i,j), next(i,j) + size(i,j) - 1]$.
Par exemple, dans Fig. 2, comme $size(0,1) = 2$, $cell(0,1)$ génère deux enfants dans L_1, correspondant à $cell(1,1)$ et $cell(1,2)$.
Le domaine de chaque cellule enfant est dérivé récursivement du domaine de sa cellule parente dans L_i :

— si e est une action a, alors a est ajoutée à $cell(i+1, next(i,j))$,

— si e est une méthode m avec des sous-tâches $[t_0, \ldots, t_k, \ldots, t_n]$, alors pour chaque k, les accomplisseurs $R(t_k)$ sont placés dans $cell(i+1, next(i,j)+k)$.

3.1.1 Analyse de l'accessibilité

Avant d'encoder un problème en CSP, Optiplan effectue d'abord une *analyse de l'accessibilité* sur l'arbre à la profondeur actuelle k. Cette analyse permet d'éviter des tentatives de résolution inutiles lorsque HiPOD n'a pas été suffisamment développé pour contenir une solution.
D'après la définition de SOLVE(P), nous savons qu'une couche entraînera un FAILURE si l'une des conditions suivantes est remplie :

— Il existe une cellule $cell(i,j) \in L_i$ qui ne contient aucune action dans son domaine. Cela implique que tous les plans auront un défaut de tâche non primitive.

— Pour chaque précondition de chaque action dans $cell(i,j)$, il n'existe aucun lien causal dans la liste des liens causaux de L_i.

La première condition est triviale à vérifier. La seconde condition peut être vérifiée en temps polynomial : pour chaque précondition p de chaque action a dans $cell(i,j)$, nous vérifions s'il existe une autre cellule $cell(i,k) \in L_i$ (avec $j \neq k$) contenant une action b telle que $p \in eff^+(b)$. En d'autres termes, nous vérifions si les préconditions de chaque action dans $cell(i,j)$ peuvent être satisfaites par les effets d'une action présente dans une autre cellule au même niveau de profondeur. Si aucune cellule ne permet de satisfaire ces préconditions pour toutes les valeurs du domaine de $cell(i,j)$, alors quelle que soit l'action choisie, la condition d'échec sera remplie.
Nous pouvons maintenant passer à l'explication de la synthèse d'un CSP à partir de l'arbre HiPOD.

3.2 Encodage CSP

La satisfaction de contraintes est un paradigme général et puissant pour la résolution de problèmes [10], qui peut être formalisé comme un triplet (X, D, C), où :

— $X = \langle x_1, ..., x_n \rangle$ est un ensemble de variables,

— $D = \langle D_1, ..., D_n \rangle$ est un ensemble de domaines associés aux ces variables $x_i \in D_i$,

— $C = \langle c_1, ..., c_m \rangle$ est un ensemble de contraintes sur les variables. Une contrainte d'arité k restreint les valeurs d'un tuple $(x_1, ..., x_k) \in X$. Elle peut être notée soit sous la forme $((x_1, ..., x_k), \{(v_1, ..., v_k) \in D_1, ..., D_k\})$, soit $\{(x_1 = v_1, ..., x_k = v_k) | (v_1, ..., v_k) \in D_1, ..., D_k\}$. À noter que dans l'article, nous remplaçons parfois des valeurs par _, par exemple $(x_1 = a, x_2 = \_)$, ce qui signifie que toute valeur du domaine de cette variable est valide.

Une *solution* à un CSP est un tuple de valeurs $T = (v_1, ..., v_n)$ tel que $v_i \in D_i$, et que l'affectation des variables $x_i = v_i$ satisfait la conjonction de toutes les contraintes de C.

3.2.1 Variables CSP

Dans l'approche Optiplan, nous avons besoin de trois types de variables de décision correspondant 1) au choix d'un élément de cellule, 2) à l'établissement des liens causaux et 3) à la sélection des contraintes d'ordre :

— $v_{i,j}$. Pour chaque $cell(i,j)$, il existe une variable $v_{i,j}$ dont le domaine correspond au domaine de la cellule. Par conséquent, la valeur de $v_{i,j}$ représente l'élément choisi dans la cellule.

— $o_{i,j,k}$. Cette variable indique une relation d'ordre entre $cell(i,j)$ et $cell(i,k)$. Son domaine est $\{<, >, \emptyset\}$.

— $\lambda^p_{i,j}$. Cette variable indique quelle cellule de L_i soutient la précondition p de $cell(i,j)$. Étant donné que les préconditions exactes de $cell(i,j)$ ne sont pas connues avant la sélection d'un élément dans la cellule, p représente un indice de précondition plutôt qu'un fait spécifique. Par exemple, dans la Fig. 3, la précondition 0 de $cell(0,0)$ est $x1$ si $cell(0,0) = a_1$, $\emptyset$ si $cell(0,0) = a_2$, et $x3$ si $cell(0,0) = a_3$.

Sur ces variables, nous appliquons les contraintes suivantes :

C1. Les valeurs des cellules parentes et enfants doivent être cohérentes avec les décompositions. Soit $cell(i+1, l)$, $l \in [next(i,j), next(i,j) + size(i,j)]$ un enfant de $cell(i,j)$. Chaque élément e dans la cellule introduit une contrainte définie comme suit :

— Si $e \in \mathcal{M}$ avec des sous-tâches $[t_0, ..., t_k, ..., t_n]$, alors ajouter $(v_{i,j} = e, v_{i+1,l} = e'), |, e' \in R(t_k)$.

— Si $e \in \mathcal{M}$ mais $k > |w|$, alors $(v_{i,j} = e, v_{i+1,l} = noop)$.

— Si $e \in \mathcal{A}$ et $k = 0$, alors $(v_{i,j} = e, v_{i+1,l} = e)$.

— Si $e \in \mathcal{A}$ et $k > 0$, alors $(v_{i,j} = e, v_{i+1,l} = noop)$.

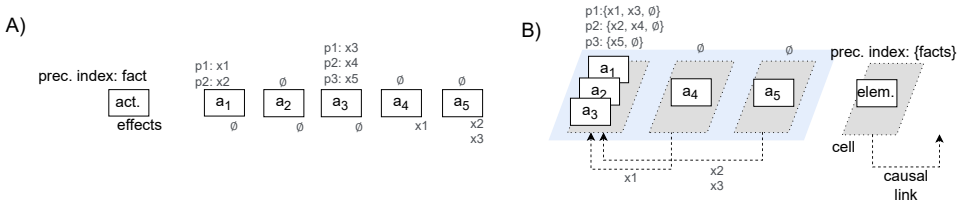


FIGURE 3 – À gauche, quelques actions avec leurs effets et préconditions. À droite, une couche hiérarchique.

Dans Fig. 2, $cell(0,1)$ génère deux contraintes :

— $(m1, m3), (m1, m4), (m2, a4)$ sur $(v_{0,1}, v_{1,1})$

— $(m1, a3), (m2, noop)$ sur $(v_{0,1}, v_{1,2})$

C2. Chaque précondition de $cell(i,j)$ est supportée par une autre cellule. Cette contrainte binaire s'applique aux couples $(v_{i,j}, \lambda^p_{i,j})$ avec $j \in [0, |L_i| - 1]$, où p désigne la p-ième précondition de $cell(i,j)$. Pour chaque e dans la cellule, les contraintes suivantes sont ajoutées :

— Si $e \in \mathcal{M}$, alors ajouter $(v_{i,j} = e, \lambda^k_{i,j} = \emptyset)$.

— Si $e \in \mathcal{A}$ dont la p-ième précondition est x, alors ajouter $(v_{i,j} = e, \lambda^p_{i,j} = l), |, \exists a \in D(v_{i,l}), x \in \mathit{eff}^+(a))$.

— Si $e \in \mathcal{A}$ et qu'il a moins de p préconditions, alors ajouter $(v_{i,j} = e, \lambda^p_{i,j} = \emptyset)$.

Dans Fig. 3, $cell(0,0)$ génère une contrainte par precondition :

— $(a, q), (b, q), (c, f)$ sur $(v_{i,j}, \lambda^0_{i,j})$

— $(a, f), (b, \emptyset)$ sur $(v_{i,j}, \lambda^1_{i,j})$

— $(a, \emptyset), (b, \emptyset)$ sur $(v_{i,j}, \lambda^2_{i,j})$

On remarque que dans cet exemple, $v_{i,j} = c$ est impossible, car la deuxième contrainte restreint le domaine de $v_{i,j}$ à a, b.

C3. Lorsqu'une cellule $cell(i,l)$ supporte une précondition de $cell(i,j)$, l'action sélectionnée doit la produire. Il s'agit d'une contrainte ternaire sur les triplets $(v_{i,j}, \lambda^p_{i,j}, v_{i,l})$. Ses valeurs possibles sont :

— $(v_{i,j} = e, \lambda^k_{i,j} = l, v_{i,l} = e')$, tel que $pre_p(e) \in \mathit{eff}^+(e')$

— $(v_{i,j} = \_, \lambda^k_{i,j} \neq l, v_{i,l} = \_)$

Dans Fig. 3, par exemple, nous avons deux contraintes pour la première précondition de $cell(i,j)$:

— $(a, q, d), (b, q, d), (\_, f, \_)$ sur $(v_{i,j}, \lambda^0_{i,j}, v_{i,q})$

— $(c, f, e), (\_, q, \_)$ sur $(v_{i,j}, \lambda^0_{i,j}, v_{i,f})$

C4. w_0 impose des relations d'ordre entre les cellules de L_0. Cette contrainte unaire est introduite pour chaque relation d'ordre $t_j < t_k$ dans $\prec_0$: $(o_{0,j,k}, <)$.

C5. Les relations d'ordre sont transitives. Cette contrainte capture le comportement suivant : si $cell(i,j) < cell(i,k)$ et $cell(i,k) < cell(i,q)$ alors $cell(i,j) < cell(i,q)$. Cette transitivité est assurée par une contrainte ternaire sur chaque triplet $(o_{i,j,k}, o_{i,k,q}, o_{i,j,q})$, avec les valeurs autorisées étant $(x, y, z), |, x, y, z \in <, >, \emptyset, x = \emptyset \vee x \neq y \vee y = z$.

C6-7. La cellule d'état initial est toujours la première, et la cellule d'état objectif - la dernière. Pour chaque $j \in [1, |L_i|]$, nous définissons une contrainte unaire $(o_{i,0,j} = ; <)$. Et pour chaque $j \in [0, |L_i| - 2]$, nous définissons $(o_{i,j,|L_i|-1} =; <)$.

C8. Les enfants héritent des relations d'ordre de leurs parents. Cette contrainte binaire s'applique aux couples $(o_{i,j,k}, o_{i+1,g,q})$ tels que $g \in [next(i,j), next(i,j) + size(i,j)]$ et $q \in [next(i,k), next(i,k) + size(i,k)]$. Ses valeurs autorisées sont $(o_{i,j,k} = x, o_{i+1,g,q} = y), |, x, y \in <, >, \emptyset, x = \emptyset \vee x = y$.

C9. Les méthodes de L_i peuvent introduire des ordres dans L_{i+1}. Cette contrainte s'applique aux couples $(v_{i,j}, o_{i+1,next(i,j)+g,next(i,j)+k})$ tels que $g, k \in [0, size(i,j)]$. Chaque $m \in v_i(j)$ introduit des valeurs comme suit :

— $(v_{i,j} = m, o_{i+1,\ldots} = \circ), |, (t_g \circ t_k) \in \prec_m$

— si m ne définit pas d'ordre sur les positions g, k, alors $(v_{i,j} = m, o_{i+1,\ldots} = \_)$, c'est-à-dire que tout ordre est autorisé.

C10. Un lien causal implique une relation d'ordre. Cette contrainte binaire s'applique aux couples $(\lambda^p_{i,j}, o_{i,l,j})$. Les valeurs autorisées sont $(\lambda^p_{i,j} = l, o_{i,l,j} =; <)$, ainsi que $(\lambda^p_{i,j} \neq l, o_{i,l,j} = \_)$.

C11. Les liens causaux ne doivent pas être menacés. Cette contrainte s'applique aux tuples $(\lambda^p_{i,j}, v_{i,j}, v_{i,q}, o_{i,j,q}, o_{i,l,q})$ pour éviter que a' ne détruise la précondition de a.

C12. Seules les actions peuvent être sélectionnées dans la dernière couche L_n. Chaque cellule $j \in [0, |L_n| - 1]$ produit une contrainte unaire $v_{n,j} = a, |, a \in \mathcal{A}$.

3.3 Extraction du plan

Une fois que le problème P avec une borne k a été encodé en CSP et résolu, l'extraction d'un plan partiellement ordonné implique deux étapes : 1) la construction du réseau de tâches w, et 2) l'identification de l'ensemble des liens causaux C. Ces deux éléments peuvent être déduits à partir de l'affectation des variables dans la couche finale L_n. Les tâches de w sont directement obtenues à partir des valeurs attribuées aux variables v. Les contraintes d'ordre entre ces tâches sont ensuite déduites en analysant les variables o correspondantes. De la même manière, l'ensemble des liens causaux C est obtenu en examinant chaque variable λ, qui indique comment les préconditions sont satisfaites dans le plan.

3.4 Optimisation du plan

Solveurs CSP souvent intègrent des fonctionnalités d'optimisation, ce qui permet d'optimiser un CSP en spécifiant simplement une fonction objectif.
L'objectif dans Optiplan est de minimiser le *chemin critique* du plan, c'est-à-dire la séquence la plus longue d'actions dépendantes. Pour cela, nous introduisons un ensemble supplémentaire de variables et de contraintes, définies comme suit :

1. Chaque $cell(i,j)$ possède une variable entière de *temps* $s_{i,j}$.
2. Nous définissons une contrainte ternaire sur $(o_{i,j,k}, s_{i,j}, s_{i,k})$, de sorte que si un ordre de tâches $t_j < t_k$ ou $t_j > t_k$ existe, alors une inégalité similaire est imposée entre $s_{i,j}$ et $s_{i,k}$.
3. Nous minimisons la valeur du temps associé à la dernière cellule de L_n (c'est-à-dire l'objectif) : $\min s_{n,|L_n|}$.

Il est important de noter que cette approche garantit l'optimalité dans le problème borné (i.e., avec une profondeur maximale k). Cependant, la véritable solution optimale du problème peut se situer à une profondeur plus grande dans l'arbre de décomposition, au-delà de la borne actuelle. Ainsi, bien que nous trouvions la meilleure solution dans les limites de k, il est possible qu'un plan globalement optimal nécessite une exploration plus profonde.

3.5 Complexité spatiale de l'encodage

Nous évaluons ici la complexité de l'encodage de HiPOD dans le CSP proposé. Supposons que l'arbre contienne N cellules. L'encodage nécessite le nombre suivant de variables :

— **Variables de cellule**. N variables pour représenter les N cellules.
— **Variables d'ordre**. $N * (N-1) \div 2$ variables pour représenter les relations d'ordre entre les cellules, de manière paire.
— **Variables de lien causal**. Le nombre de variables de lien causal est donné par $\sum_{v \in V} \max |\operatorname{pre}(e)|, e \in D(v)$, ce qui prend en compte le nombre maximal de préconditions par cellule.

La plupart des contraintes de notre encodage sont générées par cellule. Considérons une cellule quelconque $cell(i,j)$ dans le HDT :
Contraintes C1 et C5 définissent les relations entre cellules parentes et enfants. Dans le pire des cas, pour chaque $cell(i,j)$, elles génèrent $|D(v_{i,j})| * size(i,j)$ contraintes.
Contraintes C2, C3 et C11 définissent les préconditions d'une cellule et identifient quelles autres cellules les soutiennent. Dans le pire des cas, elles génèrent $size(i,j) * \max |\operatorname{pre}(e)|, e \in D(v_{i,j}) * (N-1)$ contraintes par cellule.
Contrainte C8 gère l'héritage des contraintes d'ordre des cellules parentes vers leurs enfants. Dans le pire des cas, cela peut générer jusqu'à $N - size(i,j)$ contraintes.
Contrainte C9 traite l'ordre des sous-tâches. Dans le pire des cas, où $cell(i,j)$ contient uniquement des méthodes totalement ordonnées, on génère $|D(v_{i,j})| * \sum_{i=1}^{size(i,j)} i$ contraintes par cellule.
Contraintes C4, C6, C7, C12 sont plus simples et ne nécessitent d'être encodées qu'une seule fois.
Maintenant, si l'on considère que

— N est le nombre total de cellules,
— M est la taille de la plus grande décomposition dans tout le problème,
— D est la plus grande taille de domaine rencontrée,
— P est le plus grand nombre de préconditions trouvé pour une action,

alors la complexité globale de l'encodage des contraintes peut être exprimée comme : $(D*M*N + M*P*(N-1)*N + (N-D)*N + D*M + 1)$. Cependant, comme M, D et P augmentent généralement beaucoup plus lentement que N, la complexité peut être généralisée à $\mathcal{O}(N^2)$.

4 Expérimentation

Nous avons développé un planificateur utilisant notre encodage, Optiplan et nous l'avons comparé aux deux planificateurs HTN-POCL : PANDA 3 [6], qui effectue une recherche heuristique basée sur les graphes de décomposition des tâches, [2] et pyHiPOP [15], une adaptation du planificateur HiPOP [2] sans insertion de tâches. Optiplan-N, qui est Optiplan sans fonction d'optimisation, est utilisé pour évaluer l'impact de l'optimisation sur les performances. Les expériences ont été réalisées sur une machine équipée d'un processeur Intel Core i5-1145G7 2.60GHz et d'une mémoire limitée à 10 Go. Chaque instance de problème disposait d'un temps maximal de 10 minutes de calcul CPU.
Nos évaluations couvrent 11 domaines de planification, et les planificateurs ont été évalués selon trois critères principaux : 1) le temps nécessaire pour trouver une solution, 2) la longueur du chemin critique (plus longue séquence d'actions dépendantes dans le plan généré), 3) le nombre de problèmes résolus (couverture).

4.1 Métriques d'évaluation

L'approche générale pour le calcul des scores est inspirée du système de notation de l'International Planning Competition (IPC) pour la catégorie satisficing. Dans ce système, le score de chaque planificateur est évalué relativement au meilleur planificateur et est calculé comme suit : *valeur évaluée*/ *meilleure valeur*, et si un planificateur ne trouve pas de solution, son score est 0.
Le score de benchmark correspond à la somme des scores individuels, un score plus élevé indiquant de meilleures performances relatives. Nous introduisons également un score de benchmark normalisé, calculé comme suit : *score du planificateur*$/max($*scores*$)$. Contrairement

2. Pour les évaluations, nous avons utilisé la configuration qui a montré les meilleures performances sur les problèmes HTN-POCL (heuristique TDG_c tenant compte des coûts, avec recomputation du TDG activée à chaque décomposition. Nous avons utilisé cette heuristique avec la stratégie de recherche A_2^*).

au scores classiques, dont le maximum théorique correspond à la taille de l'ensemble des problèmes (ce qui peut être irrégulier dans IPC), le score normalisé est toujours compris entre 0 et 1. Cela empêche les planificateurs de tirer avantage d'une bonne performance sur un seul benchmark volumineux.

Notre expérimentation mesure deux scores principaux : 1) *Solve time* (score d'agilité dans IPC), qui est basé sur le temps nécessaire pour trouver une solution. 2) *Qualité du plan*, mesurée en fonction de la longueur du chemin critique du plan généré. De plus, nous indiquons la couverture, c'est-à-dire le nombre de problèmes résolus par chaque planificateur.

4.2 Ensemble de problèmes de benchmark

Parmi les 11 domaines utilisés dans l'évaluation, 8 proviennent de l'ensemble de benchmarks de planification partiellement ordonnée d'IPC 2020 et 2023. Les 3 autres sont de nouveaux domaines de planification que nous avons introduits.

Nous avons défini ces nouveaux domaines car les benchmarks d'IPC sont conçus pour des planificateurs générant des plans séquentiels, laissant peu de possibilités d'exploiter pleinement la planification partiellement ordonnée.

Par exemple, dans *Woodworking*et *UM-Translog*, il est impossible d'obtenir un ordre partiel quelconque. Dans *PCP*, bien que le réseau de tâches initial puisse ne pas contenir de contraintes d'ordre, les plans générés sont toujours séquentiels. Dans *Satellite, Rover, Transport*, les plans partiellement ordonnés sont possibles, mais la concurrence ne vient que de la présence de plusieurs agents, chacun exécutant une séquence stricte d'actions. Nos nouveaux benchmarks visent à mieux tirer parti de la planification partiellement ordonnée. Nous y parvenons en définissant des décompositions avec ordre partiel ou en introduisant plusieurs agents favorisant la concurrence des actions.

Voici un aperçu des nouveaux domaines que nous avons conçus :

— **Oven :** Inspiré d'un scénario industriel réel, ce domaine modélise la cuisson de blocs de ciment dans des fours à températures différentes. Plusieurs blocs peuvent être cuits simultanément dans un four, à condition qu'ils aient la même configuration de température.

— **Medical :** Modélise un scénario collaboratif dans lequel un médecin doit effectuer une chirurgie sur un patient. Avant l'opération, il est nécessaire de préparer les outils et le patient, ce qui nécessite respectivement un et deux infirmiers.

— **Postman :** Similaire au domaine Transport d'IPC 2023, ce domaine implique la livraison de lettres à différentes destinations par un facteur. Tous les lieux sont interconnectés et le facteur n'a aucune limite de capacité pour transporter les lettres.

4.3 Résultats

Un aperçu des résultats est présenté dans Table 1. Nous constatons qu'Optiplan affiche la meilleure couverture avec 9.5/11, démontrant ainsi sa robustesse sur une grande variété de domaines.

Les deux configurations d'Optiplan (avec et sans optimisation) affichent une couverture identique, car lorsque la version optimale atteint la limite de temps, elle retourne toujours la meilleure solution trouvée jusqu'à ce moment, même si elle n'est pas théoriquement optimale.

En revanche, la baisse significative de couverture pour pyHiPOP (3.4) suggère qu'il rencontre des difficultés avec les benchmarks plus flexibles. Cela confirme les observations de [15], selon lesquelles ce planificateur échoue lorsque de nombreux plans ont des valeurs heuristiques très proches – ce qui est précisément le cas des nouveaux benchmarks que nous introduisons.

PANDA 3 affiche une couverture correcte (7.8), mais reste inférieur à Optiplan.

En ce qui concerne le temps de résolution, Optiplan-N est le plus rapide avec un score de 7.7. Cette différence est attendue, car la configuration optimale peut prendre plus de temps à explorer de meilleures solutions.

PANDA 3 est relativement performant avec un score de 6.6, tandis que pyHiPOP est en retrait avec un score de 3.8, confirmant ses difficultés sur les problèmes plus flexibles.

Concernant la qualité des plans (longueur du chemin critique), Optiplan affiche la meilleure performance avec un score de 9.5, démontrant sa capacité à produire des plans de meilleure qualité lorsque le problème le permet. Optiplan-N, bien que toujours compétitif, obtient un score plus faible de 7.9.

PANDA 3 suit avec un score de 6.7, tandis que pyHiPOP reste limité avec un score de 3.4.

En résumé, Optiplan surpasse les autres planificateurs sur tous les critères clés (couverture, temps de résolution et qualité des plans), mettant en avant sa supériorité pour résoudre les problèmes HTN-POCL.

5 Travaux connexes

La combinaison de la planification POCL avec les HTNs permet de générer des plans flexibles, capables de s'adapter aux connaissances expertes de l'utilisateur. Pour cette raison, le rapprochement entre la planification POCL [17, 18] et la planification HTN [12] est un sujet d'intérêt depuis un certain temps.

Le premier système à aborder cette question était DPOCL (Decompositional POCL) [24], mais les deux travaux qui ont réellement formalisé HTN-POCL sont [14, 7]. Tous les travaux suivants se sont développés selon deux approches principales : 1) Enrichir les planificateurs POCL avec HTN, car certaines procédures doivent être respectées strictement. 2) Étendre les planificateurs HTN avec la causalité de POCL, afin d'obtenir des plans plus adaptatifs.

Un exemple de la première approche est le cadre proposé par [14], où les auteurs affirment que les HTN traditionnels réduisent la flexibilité d'un agent face à des situations non

	Couverture			Solve time				Qualité			
Domaine (# de pbs)	*Optiplan* *Optiplan-N*	*pyHiPOP*	*PANDA 3*	*Optiplan*	*Optiplan-N*	*pyHiPOP*	*PANDA 3*	*Optiplan*	*Optiplan-N*	*pyHiPOP*	*PANDA 3*
Satellite (22)	**22/22**	3/22	18/22	15.7	**17.2**	2.5	15.4	**22.0**	20.5	3.0	18.0
Woodworking (30)	5/30	5/30	**10/30**	3.0	2.9	4.7	**6.6**	5.0	5.0	5.0	**10.0**
Smartphone (7)	**5/7**	4/7	**5/7**	1.5	1.8	**4.0**	1.7	**5.0**	**5.0**	4.0	**5.0**
PCP (17)	**11/17**	0/17	8/17	7.6	**10.3**	0.0	5.5	**11.0**	**11.0**	0.0	8.0
UM-Translog (22)	**22/22**	21/22	**22/22**	9.3	9.3	**16.6**	13.9	**22.0**	**22.0**	21.0	**22.0**
Barman-BDI (20)	0/20	**1/20**	0/20	0.0	0.0	**1.0**	0.0	0.0	0.0	**1.0**	0.0
Rover (20)	**4/20**	0/20	**4/20**	1.2	1.2	0.0	**3.6**	**4.0**	0.9	0.0	0.9
Transport (40)	**4/40**	0/40	2/40	**3.6**	**3.6**	0.0	1.6	**4.0**	**4.0**	0.0	2.0
Oven (20)	**8/20**	0/20	4/20	**7.0**	6.6	0.0	3.2	**8.0**	**8.0**	0.0	1.6
Medical (20)	**4/20**	0/20	1/20	**4.0**	**4.0**	0.0	0.2	**4.0**	1.8	0.0	2.0
Postman (20)	**2/20**	0/20	**2/20**	**1.5**	**1.5**	0.0	**1.5**	**2.0**	1.5	0.0	1.5
Total (de 238)	**87.0**	34.0	76.0	54.4	**58.2**	28.8	53.2	**87.0**	79.7	34.0	71.0
Total normalisé (de 11)	**9.5**	3.4	7.8	7.3	**7.7**	3.8	6.6	**9.5**	7.9	3.4	6.7

TABLE 1 – Performances de chaque planificateur sur les 11 benchmarks.

anticipées par l'auteur du domaine. Des travaux similaires ont été menés avec HYBIS [8] et HiPOP [2]. Cependant, ces systèmes permettent une certaine liberté dans la décomposition des tâches abstraites. En revanche, PANDA 3 [6] et pyHiPOP [15] respectent strictement la formalisation HTN, où les tâches abstraites sont uniquement décomposées via des schémas définis par l'utilisateur.

Il convient de noter qu'aucun système n'a tenté d'encoder les problèmes HTN à ordre partiel (PANDA 3 et pyHiPOP s'appuient sur des heuristiques), bien que de nombreuses recherches aient été menées sur ces problématiques dans les contextes POCL et HTN séparément.

Par exemple, parmi les encodages SAT, les contributions notables incluent les travaux de [3, 20, 19]. Ensuite, [22] et [21] ont proposé d'encoder les HTN sous forme de CSP, bien qu'aucun planificateur n'ait été implémenté sur cette base.

Pour être complet, il convient également de mentionner que [11] a proposé un encodage basé sur Answer Set Programming (ASP). Enfin, plusieurs travaux ont cherché à traduire les problèmes HTN en planification classique, comme ceux de [9, 1, 4]. Mais malgré toutes ces avancées, aucun encodage spécifique aux HTN partiellement ordonnés n'avait encore été exploré avant notre travail.

6 Conclusion

Dans ce travail, nous avons introduit une nouvelle technique d'encodage qui fusionne les principes de la planification POCL et HTN afin de résoudre des problèmes de planification hiérarchique partiellement ordonnée.

Notre approche est la première à intégrer une optimisation des plans, constituant ainsi une avancée significative dans le domaine. À travers une évaluation expérimentale sur 11 domaines de planification, nous avons démontré qu'Optiplan offre des performances compétitives en termes de temps de résolution et obtient la meilleure couverture de domaines par rapport aux planificateurs les plus performants.

Nos résultats suggèrent que de futurs travaux pourraient se concentrer sur une réduction supplémentaire de l'espace de recherche, afin d'améliorer l'efficacité du processus de planification. L'exploration de stratégies de recherche alternatives pourrait également conduire à des gains de performance significatifs, étant donné l'impact majeur de la stratégie de recherche sur le comportement global d'un solveur CSP.

De plus, notre encodage introduit la possibilité d'optimiser les plans HTN-POCL en utilisant différentes métriques. Par exemple, il serait possible de générer des plans contenant moins d'actions en maximisant le nombre de cellules vides (assignées à *noop*). Cependant, cette approche pourrait allonger le chemin critique, illustrant ainsi les compromis inhérents à l'optimisation des plans.

En résumé, notre approche représente une avancée majeure dans la planification hiérarchique partiellement ordonnée et ouvre la voie à de nouvelles recherches sur les techniques d'encodage.

Références

[1] Ron Alford, Gregor Behnke, Daniel Höller, Pascal Bercher, Susanne Biundo-Stephan, and David W. Aha. Bound to plan : Exploiting classical heuristics via automatic translations of tail-recursive htn problems. In *Proceedings of the International Conference on Automated Planning and Scheduling*, pages 20–28, 2016.

[2] Patrick Bechon, Magali Barbier, Guillaume Infantes, Charles Lesire, and Vincent Vidal. Hipop : Hierarchical partial-order planning. In *Proceedings of the*

European Starting AI Researcher Symposium, pages 51–60, 2014.

[3] Gregor Behnke, Daniel Höller, and Susanne Biundo. totsat - totally-ordered hierarchical planning through sat. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 6110–6118, 2018.

[4] Gregor Behnke, Florian Pollitt, Daniel Höller, Pascal Bercher, and Ron Alford. Making translations to classical planning competitive with other htn planners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 9687–9697, 2022.

[5] P. Bercher, R. Alford, and D. Höller. A survey on hierarchical planning – one abstract idea, many concrete realizations. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 6267–6275, 2019.

[6] Pascal Bercher, Gregor Behnke, Daniel Höller, and Susanne Biundo. An admissible htn planning heuristic. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 480–488, 2017.

[7] S. Biundo and B. Schattenberg. From abstract crisis to concrete relief – a preliminary report on combining stateabstraction and HTN planning. In *Proceedings of the European Conference on Planning*, page 157–168, 2001.

[8] Luis Castillo, Juan Fdez-Olivares, and Antonio González. On the adequacy of hierarchical planning characteristics for real-world problem solving. In *European conference on planning (ECP-2001)*, pages 169–180, 2001.

[9] N. Cavrel, D. Pellier, and H. Fiorino. Efficient htn to strips encoding for concurrent plans. In *Proceedings of International Conference on Tools with Artificial Intelligence"*, pages 962–969, 2023.

[10] Rina Dechter. *Constraint processing*. Morgan Kaufmann, 2003.

[11] Jürgen Dix, Ugur Kuter, and Dana Nau. Planning in answer set programming using ordered task decomposition. In *Proceedings of the Annual German Conference on AI*, pages 490–504, 2003.

[12] Kutluhan Erol, James A Hendler, and Dana S Nau. Umcp : A sound and complete procedure for hierarchical task-network planning. In *Proceedings of International Conference on AI Planning Systems*, pages 249–254, 1994.

[13] Daniel Höller, Pascal Bercher, Gregor Behnke, and Susanne Biundo. Htn planning as heuristic progression search. *Journal of Artificial Intelligence Research*, 67 :835–880, 2020.

[14] Subbarao Kambhampati, Amol Dattatraya Mali, and Biplav Srivastava. Hybrid planning for partially hierarchical domains. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 882–888, 1998.

[15] Charles Lesire and Alexandre Albore. pyhipop - hierarchical partial-order planner. In *Proceedings of the International Planning Competition*, pages 13–16, 2021.

[16] Charles Lesire, Guillaume Infantes, Thibault Gateau, and Magali Barbier. A distributed architecture for supervision of autonomous multi-robot missions. *Autonomous Robots*, 40 :1343–1362, 2016.

[17] David A. McAllester and David Rosenblitt. Systematic nonlinear planning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 634–639, 1991.

[18] J. Scott Penberthy and Daniel S. Weld. UCPOP : A sound, complete, partial order planner for ADL. In *Proceedings of the International Conference on Principles of Knowledge Representation and Reasoning*, pages 103–114, 1992.

[19] Dominik Schreiber. Lilotane : A lifted sat-based approach to hierarchical planning. *Journal of Artificial Intelligence Research*, 70 :1117–1181, 2021.

[20] Dominik Schreiber, Damien Pellier, Humbert Fiorino, et al. Tree-rex : Sat-based tree exploration for efficient and high-quality htn planning. In *Proceedings of the International Conference on Automated Planning and Scheduling*, pages 382–390, 2019.

[21] Tobias Schwartz, Michael Sioutis, and Diedrich Wolter. Towards hierarchical task network planning as constraint satisfaction problem. In *Proceedings of the Workshop on Hierarchical Planning*, pages 78–82, 2022.

[22] Pavel Surynek and Roman Barták. Encoding htn planning as a dynamic csp. In *Proceedings of the International Conference on Principles and Practice of Constraint Programming*, pages 868–868, 2005.

[23] Kevin Tierney, Amanda Jane Coles, Andrew Coles, Christian Kroer, Adam M. Britt, and Rune Møller Jensen. Automated planning for liner shipping fleet repositioning. In *Proceedings of the International Conference on Automated Planning and Scheduling*, pages 279–287, 2012.

[24] R. M. Young and J. D. Moore. Dpocl : A principled approach to discourse planning. *Proceedings of the International Workshop on Natural Langauge Generation*, pages 13–20, 1994.

SibylSat : Utiliser SAT comme oracle pour effectuer une recherche gloutonne en planification TOHTN

Gaspard Quenard[1], Damien Pellier[1], Humbert Fioriono[1]

[1] University Grenoble Alpes, LIG

12 mai 2025

Résumé

Cet article présente SibylSat, une nouvelle méthode basée sur SAT conçue pour résoudre efficacement les problèmes HTN Totalement Ordonnés (TOHTN). Contrairement aux planificateurs HTN basés sur SAT qui emploient une stratégie de recherche en largeur d'abord, SibylSat adopte une approche de recherche gloutonne, lui permettant d'identifier les décompositions prometteuses à développer. Le processus de sélection est facilité par une heuristique indépendante du domaine, dérivée de la résolution d'un problème relaxé, qui est également exprimé comme un problème SAT. Nos évaluations expérimentales démontrent que SibylSat surpasse les approches TOHTN basées sur SAT existantes en termes de couverture, de temps d'exécution et de qualité de plan sur la plupart des benchmarks de la Compétition International de Planification (IPC), tout en résolvant un plus grand nombre de problèmes.

Mots-clés

TOHTN, SAT, planification automatique.

1 Introduction

La planification par réseaux de tâches hiérarchiques (HTN) [10] est une technique de planification spécifique qui se concentre sur la décomposition des tâches complexes en sous-tâches plus simples. Contrairement à la planification classique, elle introduit le concept de tâches abstraites (actions de haut niveau qui ne peuvent pas être exécutées directement) et des méthodes de décomposition, lesquelles spécifient un ensemble partiellement ordonné de sous-tâches primitives et de sous-tâches abstraites supplémentaires à exécuter pour réaliser la tâche abstraite. L'objectif d'un planificateur HTN est de transformer une tâche abstraite initiale en un plan, c'est-à-dire une séquence exécutable d'actions. La planification HTN constitue un domaine de recherche majeur et fait désormais partie de la Compétition Internationale de Planification (IPC) [5, 24].

Dans cet article, nous concentrons notre étude sur la planification HTN totalement ordonnée (TOHTN) : une sous-classe particulièrement répandue des problèmes HTN, où les méthodes de décomposition spécifient une liste totalement ordonnée d'actions primitives et de tâches abstraites à exécuter afin de réaliser une tâche abstraite donnée. Les approches dominantes en planification TOHTN consistent soit à utiliser la recherche heuristique, par exemple [13, 14, 15], soit à encoder les problèmes de planification sous forme de problèmes STRIPS afin de tirer parti des améliorations constantes des planificateurs classiques, par exemple [2, 1, 9], ou encore à les encoder en logique propositionnelle, par exemple [6, 23, 4, 21], puis à employer des solveurs hautement optimisés. Cette dernière approche a suscité un regain d'intérêt ces dernières années, en particulier grâce aux planificateurs ayant obtenu d'excellentes performances lors des récentes compétitions internationales de planification. Le travail présenté dans cet article s'inscrit précisément dans cette tendance.

Dans le contexte de la planification TOHTN, la stratégie prédominante parmi les planificateurs SAT actuels consiste à employer une approche de type « recherche en largeur d'abord », limitée par le nombre maximal de décompositions de la tâche abstraite initiale. Concrètement, pour une borne donnée k, l'encodage logique considère uniquement les plans où la tâche initiale peut être décomposée au plus k fois. Si aucune solution n'est trouvée pour cette borne, la valeur de k est augmentée, élargissant ainsi la recherche à des décompositions plus complexes. Ce processus itératif se poursuit jusqu'à la découverte d'une solution satisfaisante ou jusqu'à ce qu'une limite maximale prédéfinie pour k soit atteinte.

Bien que cette approche par recherche en largeur d'abord garantisse la complétude (une solution, si elle existe, sera trouvée par le planificateur), elle s'apparente à une recherche aveugle, potentiellement très inefficace face à des espaces de recherche importants ou à des instances de problèmes complexes. Contrairement à d'autres techniques de planification qui exploitent des informations heuristiques pour orienter la recherche vers des régions prometteuses, les approches SAT actuelles pour la planification TOHTN n'intègrent pas de recherche heuristique.

Dans cet article, nous proposons SibylSat, une nouvelle approche SAT qui met en œuvre une stratégie de recherche gloutonne de type « best-first ». Cette stratégie implique d'étendre sélectivement certaines zones de recherche en fonction d'une heuristique issue de la résolution d'une version relaxée du problème de planification, ce qui permet d'identifier les décompositions prometteuses. Contrairement aux planificateurs TOHTN SAT existants, SibylSat ne développe pas automatiquement toutes les décomposi-

tions en attente lorsqu'aucune solution n'est trouvée. Au contraire, il détermine intelligemment quelles décompositions doivent être approfondies grâce à cette heuristique. Notre approche réduit ainsi considérablement l'espace de recherche et ouvre la voie à l'élaboration de nouvelles heuristiques et techniques pour résoudre les problèmes HTN par encodage SAT. Expérimentalement, nous démontrons que notre stratégie de recherche gloutonne couplée à une fonction heuristique améliore les performances sur la majorité des benchmarks IPC comparativement aux approches SAT TOHTN actuelles

L'article est organisé comme suit : nous présentons d'abord le concept de la planification TOHTN, puis nous décrivons notre planificateur, SibylSat, et enfin, nous comparons SibylSat aux autres planificateurs TOHTN basés sur SAT.

2 Problème de planification HTN

Cet article adopte une description de la planification TOHTN inspirée de la formulation présentée dans [6, 4], qui est une adaptation de la description de la planification HTN proposée dans [12] et adaptée à la planification totalement ordonnée.

2.1 Tâches, actions, méthodes et réseaux de tâches

Le concept central de la planification HTN est la notion de *tâches*. Une *tâche* est caractérisée par un nom et une liste de paramètres. Il existe deux types de tâches : les *tâches primitives* et les *tâches abstraites*. Alors que les tâches primitives affectent directement l'état du monde, les tâches abstraites ne le font pas ; elles doivent être décomposées en tâches primitives via des *méthodes de décomposition* avant de pouvoir être exécutées. Le problème de planification HTN spécifie comment une méthode peut décomposer une tâche abstraite en un *réseau de tâches*, qui est une séquence de tâches (abstraites ou primitives). Un *réseau de tâches primitives* désigne un réseau de tâches ne contenant que des tâches primitives.

Une tâche primitive a est analogue à une action en planification classique et est définie par un triplet $(\mathit{name}(a), \mathit{precond}(a), \mathit{effect}(a))$. Ici, $\mathit{name}(a)$ représente le nom de a, tandis que $\mathit{precond}(a)$ et $\mathit{effect}(a)$ désignent respectivement les ensembles de propositions définissant les préconditions et les effets. Une action a est considérée comme exécutable dans un état s, défini comme un ensemble de propositions décrivant le monde, si et seulement si $\mathit{precond}(a)$ est un sous-ensemble de s. Si O représente l'ensemble des actions et S l'ensemble des états, alors la fonction de transition d'état $\gamma : S \times O \to S$ est définie comme suit : si a est exécutable dans s, alors $\gamma(s, a) = (s \setminus \mathit{effect}^-(a)) \cup \mathit{effect}^+(a))$; sinon, $\gamma(s, a)$ est indéfinie. L'extension de γ aux séquences d'actions, notée $\gamma : S \times O^* \to S$, est définie de manière triviale.

Une méthode m définit comment une tâche abstraite peut être raffinée en un réseau de tâches. Elle est définie par un triplet $(\mathit{name}(m), c, w_m)$, où c est une tâche abstraite et w_m est un réseau de tâches. On appelle c la tâche de m et w_m les sous-tâches de m. Étant donné un réseau de tâches $w = w_1 c w_2$ où c est une tâche abstraite, l'application d'une méthode $m = (\mathit{name}(m), c, w_m)$ à w aboutira au réseau de tâches $w' = w_1 w_m w_2$ (noté $w \to^m w'$). Nous écrivons $w \to^* w'$ s'il existe une séquence de méthodes de décomposition qui transforme w en w'. Par convention, nous définissons $M(c) = \{m = (name(m), c, w_m) \mid m \in M\}$ comme l'ensemble de toutes les méthodes pouvant être appliquées à la tâche abstraite c.

2.2 Problème de planification et solution

Nous définissons un problème de planification TOHTN comme suit :

Definition 1 (Problème de planification TOHTN) *Un problème de planification HTN totalement ordonné P est un tuple $(L, C, O, M, c_I, s_I, g)$ où : L est un ensemble fini de propositions ; C est un ensemble fini de tâches abstraites (ou composées) ; O est un ensemble fini de tâches primitives (ou actions) ; M est un ensemble fini de méthodes de décomposition (ou méthodes) ; $c_I \in C$ est la tâche abstraite initiale ; $s_I \subseteq L$ est l'état initial ; g est l'état objectif (éventuellement vide).*

Résoudre un problème de planification P consiste à trouver un réseau de tâches primitives (ou *plan*) π pouvant être décomposé à partir de la tâche abstraite initiale ($c_I \to^* \pi$) de sorte que π soit exécutable dans l'état initial s_I et atteigne l'objectif g après exécution, c'est-à-dire $g \subseteq \gamma(s_I, \pi)$. Contrairement à la planification classique, une solution à un problème de planification TOHTN n'est pas simplement une séquence de tâches primitives exécutables dans l'état initial et atteignant un état objectif ; elle doit également indiquer les méthodes de décomposition utilisées pour aboutir à cette séquence de tâches primitives. Une telle solution peut être représentée par un arbre appelé *arbre de décomposition* [12], qui montre l'ensemble des étapes ayant conduit au raffinement d'une tâche abstraite en un réseau de tâches.

Definition 2 (Arbre de décomposition) *Un arbre de décomposition (DT) pour un réseau de tâches w dans un problème $P = (L, C, O, M, c_I, s_I, g)$ est un arbre $T = (N, E)$ avec :*

- *N - Un ensemble de nœuds étiquetés par une tâche primitive, une tâche abstraite ou une méthode.*
- *$E : N \to N^*$ - La fonction d'arêtes qui associe à chaque nœud une liste ordonnée d'enfants $\langle e_1, e_2, ..., e_k \rangle$.*
- *Pour tout nœud interne n étiqueté par une tâche abstraite c, $|E(n)| = 1$ et $E(n) = \langle n' \rangle$ où n' est étiqueté par une méthode $m \in M(c)$.*
- *Pour tout nœud interne n étiqueté par une méthode m, considérons $\langle t_1, t_2, \ldots, t_k \rangle$ les sous-tâches de m. Alors $|E(n)| = k$ et chaque enfant e_i de $E(n)$ est étiqueté par la tâche t_i pour tout $e_i \in E(n) = \langle e_1, e_2, \ldots, e_k \rangle$*
- *Pour la séquence de feuilles $L = \langle n^l_1, n^l_2, \ldots, n^l_k \rangle$, chaque feuille est étiquetée par une tâche primitive*

ou abstraite et, si l'on considère la séquence correspondante de tâches $\langle t_1, t_2, \ldots, t_k \rangle$, alors on a $w = \langle t_1, t_2, \ldots, t_k \rangle$.

Nous pouvons maintenant définir formellement une solution sous forme d'arbre de décomposition :

Definition 3 (Solution par arbre de décomposition) *Soit $P = (L, C, O, M, c_I, s_I, g)$ un problème de planification. Considérons T_{sol} l'arbre de décomposition pour le réseau de tâches π du problème P. L'arbre de décomposition T_{sol} est une solution pour P si et seulement si il satisfait les conditions suivantes :*

1. *La racine de T_{sol} est la tâche abstraite initiale c_I.*
2. *π ne contient que des tâches primitives.*
3. *π est exécutable dans l'état initial s_I.*
4. *π atteint l'objectif g après exécution.*

3 Arbre de décomposition de chemin et planificateurs SAT

Puisque résoudre un problème de planification HTN équivaut à trouver un arbre de décomposition (DT) qui satisfait certaines caractéristiques (cf. Definition 3), une approche consiste à explorer tous les DT possibles enracinés par la tâche abstraite initiale c_I pour un problème donné P. Cependant, l'espace de recherche représentant tous les DT possibles est très vaste pour la plupart des domaines et infini pour les domaines récursifs (domaines où une tâche abstraite peut être obtenue par des décompositions issues de la même tâche abstraite). Pour résoudre ce problème, [6, 23] ont proposé de créer des structures représentant un sous-ensemble de tous les DT possibles afin de vérifier si elles contiennent des solutions valides. Si aucune solution n'est trouvée, ces structures peuvent être étendues pour inclure d'autres DT. Ces structures sont conçues de manière à garantir qu'elles puissent être développées pour inclure tout DT possible enracinés en par la tâche abstraite initiale pour un problème donné. Ici, nous introduisons un équivalent isomorphe de leurs structures :

Definition 4 (Arbre de Décomposition de Chemin) *Un arbre de décomposition de chemin (PDT) pour un problème $P = (L, C, O, M, c_I, s_I, g)$ est un arbre $\Gamma = (N, E)$ avec :*

- *N - Un ensemble de nœuds étiquetés par une tâche primitive, une tâche abstraite ou une méthode.*
- *$E : N \rightarrow N^*$ - La fonction d'arêtes qui associe à chaque nœud une liste ordonnée d'enfants $\langle e_1, e_2, ..., e_k \rangle$.*

On note le nœud racine du PDT par r_Γ. On dit qu'un PDT est bien formé si et seulement si :

- *Le nœud racine du PDT r_Γ est étiqueté par la tâche abstraite initiale du problème c_I.*
- *Pour tout nœud interne n étiqueté par une tâche abstraite c, $|E(n)| = |M(c)|$ et $\forall m_i \in M(c), \exists e_i \in E(n)$ tel que e_i est étiqueté par la méthode m_i.*
- *Pour tout nœud interne n étiqueté par une méthode m, considérons $\langle t_1, t_2, \ldots, t_k \rangle$ les sous-tâches de m, alors $|E(n)| = k$ et e_i est étiqueté par la tâche t_i pour tout $e_i \in E(n) = \langle e_1, e_2, \ldots, e_k \rangle$.*
- *Pour toute feuille n^l, n^l est étiquetée par une tâche primitive ou une tâche abstraite.*

La figure 1 illustre un PDT pour un problème $P = (L, C, O, M, c_I, s_I, g)$ à un certain niveau de décomposition, et met en évidence un DT pour le réseau de tâches $\langle A_2, A_1, A_3, A_4 \rangle$ comme un sous-arbre de ce PDT. Ce DT est une solution pour P si et seulement si $\pi = \langle A_2, A_1, A_3, A_4 \rangle$ est exécutable dans s_I et $g \subseteq \gamma(s_I, \pi)$.

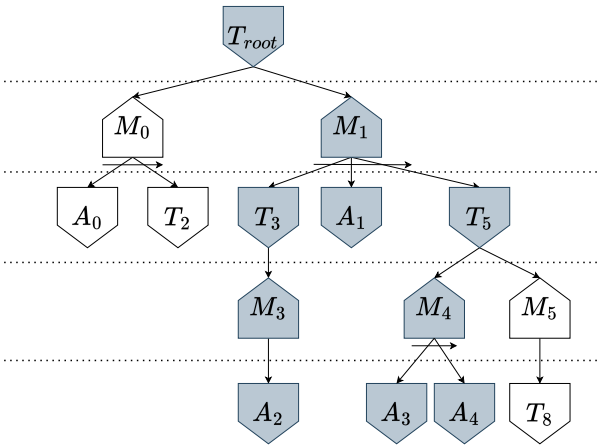


FIGURE 1 – Exemple illustrant un PDT à un certain niveau de décomposition pour un problème $P = (L, C, O, M, c_I, s_I, g)$, où $T_i \in C, M_i \in M$ et $A_i \in O$. On remarque que ce PDT ne contient pas tous les DT possibles, car les tâches abstraites T_2 et T_8 ne sont pas développées. Un DT solution potentiel est mis en évidence en gris.

La recherche d'une solution au sein d'un PDT donné est l'approche employée par tous les planificateurs TOHTN basés sur SAT actuels [23, 21, 6, 4]. L'idée clé consiste à encoder un PDT (ou une structure équivalente isomorphe) dans une formule qui est satisfaisable si et seulement s'il existe une solution dans le PDT. Si la formule est satisfaisable, la solution peut être extraite de l'assignation satisfaisante. Si aucune solution n'est trouvée dans le PDT, celui-ci peut être étendu en développant un nombre quelconque de ses *nœuds non développés* (feuilles étiquetées par des tâches abstraites). Pour chaque nœud sélectionné étiqueté par une tâche abstraite c, son processus d'expansion suit deux étapes : d'abord, on ajoute toutes les méthodes applicables $M(c)$ comme enfants ; ensuite, pour chaque méthode, on ajoute sa séquence ordonnée de sous-tâches en tant qu'enfants. Les encodages existants étendent le PDT en développant tous ses nœuds non développés. Si le PDT initial est initialisé avec un seul nœud représentant la tâche abstraite initiale, [6] ont montré que, en suivant cette ap-

proche d'expansion, le $j^{\text{ème}}$ PDT capturera tous les DT possibles avec une profondeur de décomposition maximale de j. Ainsi, cette approche correspond à une recherche en "largeur d'abord" selon la profondeur de décomposition et est, par conséquent, complète.

4 SibylSat planner

4.1 Approche de planification

Comme les autres planificateurs TOHTN basés sur SAT, SibylSat utilise un PDT comme espace de recherche et suit une procédure standard alternant entre l'expansion de l'espace de recherche, l'encodage et l'invocation d'un solveur SAT pour trouver un plan solution. Cependant, une différence majeure réside dans la stratégie d'expansion, qui ne suit pas une approche en largeur d'abord. Au lieu de cela, SibylSat sélectionne des tâches abstraites prometteuses à développer en priorité.

La procédure est illustrée sous forme de pseudo-code dans l'algorithme 1. L'algorithme commence par initialiser le PDT avec un seul nœud représentant la tâche abstraite initiale du problème. Ensuite, le processus alterne entre deux phases jusqu'à ce qu'une solution soit trouvée : l'une où SibylSat recherche un DT solution dans le PDT (cette étape servant de condition d'arrêt), et l'autre où il étend le PDT en explorant en priorité les tâches abstraites les plus prometteuses. Pour ce faire, il cherche à identifier et à développer un *DT prometteur* dans le PDT, où "prometteur" désigne un DT dont l'expansion (c'est-à-dire le développement de tous ses nœuds en attente) pourrait conduire à un DT solution. Le choix du DT prometteur à développer est effectué en recherchant un DT solution dans une relaxation du PDT où les feuilles représentant des tâches abstraites sont considérées comme des actions. Le DT solution trouvé dans ce PDT relaxé sert alors d'heuristique pour identifier les nœuds en attente à développer dans le PDT afin d'atteindre une solution. Plus précisément, toute tâche abstraite faisant partie du plan du DT solution relaxé verra sa feuille correspondante dans le PDT développée. La recherche d'un DT solution dans le PDT, ainsi que celle d'un DT prometteur dans sa version relaxée, est effectuée à l'aide d'un solveur SAT.

Une condition nécessaire (mais non suffisante) pour garantir que notre planificateur soit complet est que tout sous-arbre d'un DT solution doit également être reconnu comme un DT prometteur dans un PDT relaxé. Ne pas respecter cette condition pourrait conduire à ignorer des solutions valides. Par conséquent, lors de la relaxation du PDT, les préconditions et effets attribués à une tâche abstraite doivent être soigneusement inférés pour satisfaire cette exigence. L'approche utilisée pour les déduire sera détaillée dans une section ultérieure. Il est important de noter que cette implication n'est pas bidirectionnelle : un DT prometteur ne conduit pas nécessairement à un DT solution lorsqu'il est davantage développé. Plus nous sommes capables de représenter fidèlement les tâches abstraites sous forme d'actions (c'est-à-dire d'inférer correctement leurs préconditions et effets), plus nous nous rapprochons de cette équivalence.

L'algorithme 1 est qualifié de "glouton" car il étend le PDT en utilisant le premier DT solution relaxé trouvé lors de la phase d'expansion, même si plusieurs DT solutions relaxés existent dans le PDT. Cette méthode est complète pour les domaines de planification non récursifs car le nombre de DT est fini, et notre algorithme finit par tous les explorer à mesure qu'il étend le PDT. Cependant, cette approche est généralement non terminante pour les domaines récursifs, car l'algorithme peut continuer à développer des DT dans le PDT sans jamais aboutir à un DT solution. Nous expliquerons dans une section ultérieure comment modifier l'algorithme 1 pour garantir sa terminaison.

Algorithm 1 SibylSat Planner

```
1:  procedure SibylSat(P = (L, C, O, M, c_I, s_I, g))
2:      PDT ← InitializePDT(P)
3:      return Greedy(PDT)
4:  end procedure
5:  procedure Greedy(PDT)
6:      DT_sol ← FindSolution(PDT)
7:      if DT_sol ≠ ∅ then
8:          return DT_sol
9:      end if
10:     DT_relaxed ← FindPromisingDT(PDT)
11:     PDT ← ExpandPDT(PDT, DT_relaxed)
12:     return Greedy(PDT)
13: end procedure
```

4.2 Exemple

Pour illustrer notre algorithme, nous détaillons comment le PDT d'un problème P est développé par notre planificateur dans la Figure 1. SibylSat est initialisé en créant un PDT avec un seul nœud racine, T_{root}, représentant la tâche abstraite initiale du problème. Il alterne ensuite entre une phase de recherche et une phase d'expansion jusqu'à ce qu'un DT solution soit trouvé.

Dans la phase de recherche initiale, comme T_{root} est une tâche abstraite, le PDT ne contient pas encore de DT solution, ce qui déclenche la première phase d'expansion. Lors de cette phase, SibylSat applique la technique de relaxation pour transformer toutes les feuilles correspondant à des tâches abstraites, dans ce cas T_{root}, en actions. Une recherche est ensuite effectuée dans le PDT relaxé, identifiant le DT relaxé correspondant au plan $\langle T_{root} \rangle$ comme solution. Ce DT est alors développé dans le PDT, aboutissant à un PDT étendu encapsulant les trois premières couches de la Figure 1, avant de revenir à la phase de recherche.

La deuxième phase de recherche échoue de nouveau à trouver un DT solution dans le PDT, ce qui entraîne une nouvelle phase d'expansion. Au cours de cette phase, le PDT est à nouveau relaxé en considérant les tâches abstraites feuilles T_2, T_3 et T_5 comme des actions et en cherchant un DT solution dans ce PDT relaxé. Il est intéressant de noter que, dans ce PDT relaxé, deux DT solutions potentiels émergent : l'un impliquant le réseau de tâches $\langle A_0, T_2 \rangle$ et un autre comprenant $\langle T_3, A_1, T_5 \rangle$. En supposant que ces réseaux de tâches soient tous deux exécutables dans l'état ini-

tial du problème et permettent d'atteindre l'objectif après exécution, le solveur SAT sélectionne l'un d'eux pour un développement ultérieur. Dans cet exemple, le solveur SAT choisit le DT du réseau de tâches $\langle T_3, A_1, T_5 \rangle$, incitant SibylSat à développer les nœuds correspondants T_3 et T_5, ce qui conduit à l'ensemble du PDT présenté dans la Figure 1, avant de revenir à la phase de recherche.

Lors de cette phase de recherche dans le PDT, SibylSat identifie le DT solution associé au plan $\langle A_2, A_1, A_3, A_4 \rangle$, conduisant à la terminaison réussie du planificateur. Cependant, si cette solution s'avérait non exécutable dans l'état initial, l'absence d'autres DT solutions possibles aurait déclenché une nouvelle phase d'expansion du PDT. Dans ce cas, si les DT des réseaux de tâches $\langle A_0, T_2 \rangle$ et $\langle A_2, A_1, T_8 \rangle$ étaient des DT solutions relaxés, alors le solveur SAT aurait retourné l'un des deux, ce qui aurait conduit au développement soit du nœud T_2, soit du nœud T_8.

5 Recherche d'un DT solution dans un PDT

Lors de la phase de recherche, notre planificateur doit déterminer si un PDT contient un DT solution en tant que sous-arbre. Pour ce faire, l'approche des planificateurs TOHTN basés sur SAT consiste à créer une formule SAT qui est satisfiable si et seulement si un DT solution existe dans le PDT. Étant donné que notre espace de recherche est structurellement congruent avec ceux proposés par [21, 23, 6, 4], nous pouvons utiliser n'importe lequel des encodages introduits dans ces travaux pour rechercher un DT solution. Nous avons choisi d'utiliser l'encodage du planificateur Lilotane, en particulier parce qu'il est compatible avec un solveur SAT incrémental. Cela permet à notre solveur de conserver les connaissances acquises lors des itérations précédentes lorsqu'il recherche des DT solutions et des DT solutions relaxés. Comme l'encodage utilisé pour rechercher un DT solution dans un PDT est identique à celui proposé par Lilotane, nous ne détaillons pas dans cet article les règles SAT spécifiques employées pour rechercher un DT solution.

6 Expansion du PDT

Dans cette section, nous présentons la phase d'expansion du PDT. Cette phase intervient lorsqu'aucun DT solution n'existe dans le PDT actuel (c'est-à-dire lorsque la procédure de recherche a échoué à en trouver un). Dans ce cas, certaines feuilles en attente doivent être développées pour permettre au PDT de capturer un ensemble plus large de DT. L'idée principale de notre phase d'expansion est de trouver un DT qui soit un sous-arbre d'un DT solution. Si un tel DT peut être identifié, son développement permettra d'étendre le PDT jusqu'à inclure un DT solution en tant que sous-arbre. Cependant, identifier un DT menant à un plan primitif exécutable n'est pas faisable, car cela nécessiterait de connaître à l'avance un DT solution pour le problème. Ainsi, nous nous intéressons à une approximation. Cette approximation consiste à identifier un DT prometteur : un DT qui pourrait conduire à un DT solution une fois développé. Nous définissons un DT comme "prometteur" si son réseau de tâches relaxé est exécutable dans l'état initial et atteint l'objectif après exécution. Pour relaxer un réseau de tâches, nous convertissons toutes ses tâches abstraites en actions avec des préconditions et des effets basés sur les décompositions possibles de la tâche abstraite correspondante. Trouver un DT prometteur revient ainsi à rechercher un DT solution dans un PDT relaxé où toutes les feuilles représentant des tâches abstraites sont converties en actions.

Nous nous intéressons maintenant au processus d'inférence des préconditions et des effets d'une tâche abstraite. Comme détaillé dans l'approche de planification, une condition nécessaire pour que notre planificateur soit complet est que tous les sous-arbres d'un DT solution soient reconnus comme des DT prometteurs. Par conséquent, les préconditions et effets inférés pour une tâche abstraite doivent être valides pour toutes les raffinements possibles de cette tâche en réseaux de tâches primitifs.

Nous commençons par expliquer l'inférence des préconditions d'une tâche abstraite. Ces préconditions doivent englober les faits nécessaires à tout raffinement possible de la tâche abstraite. [20] ont qualifié ces préconditions de "préconditions obligatoires", un concept utilisé dans plusieurs planificateurs [21, 16] pour éliminer de l'espace de recherche les DT qui ne mènent pas à une séquence de tâches primitives exécutable. Dans notre planificateur, nous employons l'algorithme d'inférence des préconditions obligatoires proposé par [21].

Considérons maintenant les effets d'une tâche abstraite. Contrairement aux actions, dont les effets sont explicitement définis, les tâches abstraites introduisent une incertitude quant à l'état résultant. En effet, pour une action a, la transition $\gamma(s, a) = (s \setminus \mathit{effect}^-(a)) \cup \mathit{effect}^+(a)$ définit précisément l'état après exécution. En revanche, il est difficile de définir l'état post-exécution d'une tâche abstraite, car elle peut être décomposée en plusieurs réseaux de tâches primitives ayant des effets post-exécution différents. Notre solution consiste à calculer l'ensemble $\mathit{poss\text{-}effect}^+(t)$ (respectivement $\mathit{poss\text{-}effect}^-(t)$) d'une tâche abstraite, correspondant à tous les faits positifs (respectivement négatifs) pouvant être causés par un raffinement de cette tâche abstraite.

Si nous parvenons à calculer ces deux ensembles, nous pouvons alors définir une surestimation de l'état post-exécution d'une tâche abstraite t. Cela est réalisé en considérant l'exécution de t comme menant à un état non déterministe, défini par l'ajout d'un sous-ensemble quelconque de $\mathit{poss\text{-}effect}^+(t)$ et la suppression d'un sous-ensemble quelconque de $\mathit{poss\text{-}effect}^-(t)$ à l'état pré-exécution.

[4] ont montré que le calcul exact des effets possibles d'une tâche abstraite est un problème PSPACE- ou EXPTIME-complet, selon la structure hiérarchique de décomposition de la tâche concernée. Toutefois, il est possible d'obtenir une surestimation en temps polynomial. Pour notre algorithme, nous calculons une surestimation des effets possibles d'une tâche abstraite (notée $\mathit{poss\text{-}effect}_*$) avec la formule suivante :

$$\textit{poss-effect}^+_*(t) = \bigcup_{t' \in subtasks(m), m \in M(t)} \textit{poss-effect}^+_*(t')$$

$$\textit{poss-effect}^-_*(t) = \bigcup_{t' \in subtasks(m), m \in M(t)} \textit{poss-effect}^-_*(t')$$

Si t est une action, alors :

$$\textit{poss-effect}^+_*(t) = \textit{effect}^+(t)$$
$$\textit{poss-effect}^-_*(t) = \textit{effect}^-(t)$$

Si t est récursive, nous évitons une récursion infinie en supprimant t de l'ensemble des sous-tâches possibles de toutes les méthodes.

Il convient de noter que l'état post-exécution non déterministe d'une tâche abstraite, tel que défini ci-dessus, constitue souvent une surestimation de l'état post-exécution qu'un raffinement concret spécifique de la tâche abstraite pourrait réellement produire, car il englobe les effets de divers raffinements. Bien que [19] aient proposé une méthode permettant de généraliser les effets possibles d'une tâche abstraite en les différenciant selon les résultats de chaque raffinement, cette méthode est trop coûteuse en termes de calcul. Néanmoins, il reste possible d'affiner l'état post-exécution non déterministe d'une tâche abstraite en éliminant les incohérences résultant de la combinaison des effets de différents raffinements.

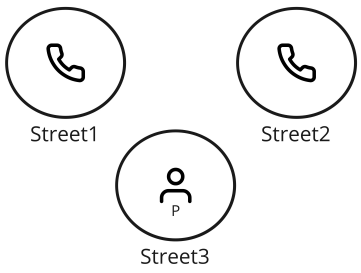


FIGURE 2 – État du monde simplifié.

Considérons l'exemple illustré dans la Figure 2. Cette figure représente un état du monde avec trois emplacements : Street1, Street2 et Street3. Une personne, notée 'P', est située à Street3 et doit appeler un taxi. Considérons une tâche abstraite $call\_taxi()$, dont l'objectif est d'appeler un taxi après s'être rendu à un emplacement où se trouve une cabine téléphonique. Dans cet exemple, il existe une cabine téléphonique à la fois à Street1 et Street2, de sorte que les effets possibles de la tâche abstraite $call\_taxi()$ incluront les prédicats $at(P, Street1)$ et $at(P, Street2)$. Si cette tâche est traitée comme une action directe, l'application de notre formule pour calculer son état post-exécution pourrait aboutir à une incohérence où la personne se retrouve simultanément à Street1 et Street2. Un tel état incohérent pourrait conduire à la découverte de DT solutions relaxés qui sont impraticables, c'est-à-dire qui ne peuvent aboutir à un DT solution valide une fois développés. Cela entraîne souvent le développement de nœuds inutiles à la recherche d'un DT solution valide.

Notre approche pour réduire ces incohérences repose sur l'utilisation de mutex. Les mutex définissent les ensembles de faits qui ne peuvent pas coexister dans le même état atteignable. Dans notre exemple, les prédicats $at(P, Street1)$ et $at(P, Street2)$ sont en mutex, car une personne ne peut pas être à deux endroits en même temps. Pour inférer ces mutex, nous utilisons l'algorithme proposé par [11], qui détecte des groupes de mutex, c'est-à-dire des ensembles de prédicats où chaque prédicat est en mutex avec tous les autres du même groupe. En utilisant ces mutex, nous améliorons la qualité de l'état post-exécution des tâches abstraites en empêchant la cooccurrence d'effets incompatibles.

En pratique, nous appliquons une contrainte "au plus un" dans notre encodage SAT pour empêcher les effets mutuellement exclusifs d'une tâche abstraite de se produire simultanément. Il existe plusieurs méthodes pour encoder ces contraintes dans SAT, comme discuté par [18]. Nos expériences montrent que, avec le solveur Glucose [3], l'encodage par paires classiques semble légèrement plus performant que les autres encodages testés, bien qu'il consomme plus de mémoire.

Nos résultats expérimentaux indiquent que, dans de nombreux domaines, l'utilisation des mutex permet de trouver des DT solutions abstraits de meilleure qualité et, par conséquent, de développer moins de nœuds dans le PDT avant d'aboutir à un DT solution.

6.1 Recherche d'un DT solution dans un PDT relaxé

Une fois que le PDT a été relaxé afin de remplacer toutes les tâches abstraites en feuilles par leurs actions correspondantes, le processus de recherche devient similaire à celui du PDT non relaxé, à une différence majeure près : les effets des tâches abstraites sont non déterministes. Cette incertitude signifie qu'un effet listé dans $\textit{poss-effect}_*(c)$ peut ne pas nécessairement se produire, même si c est exécutée. Par conséquent, lorsqu'on utilise un encodage SAT pour rechercher un DT solution dans un PDT relaxé, les effets d'une tâche abstraite en tant que feuille ne peuvent pas être encodés de la même manière que les effets d'une action, qui sont représentés par $action \implies \textit{eff}$. Au lieu de cela, pour chaque effet possible d'une tâche abstraite en tant que feuille, nous devons encoder le fait que si cette tâche est incluse dans un DT solution, alors chacun de ses effets *peut* se produire dans son état post-exécution.

Lilotane intègre déjà des règles permettant d'encoder cette incertitude dans l'état post-exécution d'une tâche abstraite, appelées *frame axioms*. Ces axiomes sont des règles qui garantissent que si la valeur d'un prédicat change, alors une tâche abstraite ou une action spécifique ayant cet effet a nécessairement été exécutée. En utilisant ces frame axioms pour encoder les effets possibles d'une tâche abstraite, nous prenons correctement en compte leurs effets non déterministes.

Par exemple, supposons qu'une tâche t puisse potentiellement modifier un prédicat p, c'est-à-dire que p est un effet possible de t. Si le prédicat p change de valeur entre l'état avant et l'état après l'exécution de t, alors la tâche t doit nécessairement être incluse dans le DT solution.
Cela peut être formalisé par le frame axiom suivant :

$$\neg p\_avant\_t \wedge p\_apres\_t \implies t$$

Cependant, l'implication n'est pas réciproque. Si la tâche t est incluse dans le DT solution, cela ne garantit pas que le prédicat p devienne vrai. Ainsi, le non-déterminisme des effets des tâches abstraites est préservé.

6.2 Développement des feuilles du PDT

Enfin, nous expliquons comment nous développons les nœuds en attente d'un PDT en fonction d'un DT solution trouvé dans un PDT relaxé. Dans le PDT relaxé, toutes les feuilles représentant des tâches abstraites sont considérées comme des actions. Par conséquent, le plan π dans le DT solution peut inclure des tâches abstraites. Pour chacune de ces tâches abstraites, nous développons le nœud correspondant dans le PDT ainsi que tous ses nœuds enfants immédiats correspondant aux méthodes applicables. Cette approche garantit que toutes les feuilles du PDT contiennent soit des tâches primitives, soit des tâches abstraites. Par exemple, comme illustré dans la Figure 1, si le plan du DT solution pour le PDT relaxé est $\langle A_0, T_2 \rangle$, alors le nœud contenant T_2 ainsi que ses méthodes enfants immédiates seront développés.
Il est à noter que, dans notre implémentation, nous utilisons l'espace de recherche proposé par le planificateur Lilotane, où chaque nœud contient toutes les tâches abstraites pouvant apparaître simultanément à un certain niveau de raffinement. Afin d'adhérer à l'encodage incrémental fourni par Lilotane, où certaines règles dépendent de l'ensemble complet des opérations présentes dans un nœud, nous développons toutes les tâches abstraites du même nœud lorsqu'un tel nœud est étendu. Une future amélioration possible pourrait consister à modifier l'encodage afin de permettre le développement sélectif des seules tâches abstraites identifiées dans le DT solution du PDT relaxé.

7 Assurer la complétude pour les domaines récursifs

Comme la description actuelle de notre planificateur suit une approche gloutonne en "best-first" plutôt qu'une recherche en largeur d'abord, utilisée par les autres planificateurs TOHTN basés sur SAT, cette approche peut souffrir d'une non-termination dans les domaines récursifs.
Une solution possible pour atténuer ce problème consisterait à limiter la profondeur maximale de décomposition du PDT en utilisant une approche d'approfondissement itératif, garantissant que tous les DT avec une profondeur de décomposition allant jusqu'à k soient explorés avant d'augmenter progressivement cette limite. En pratique, deviner la profondeur initiale de décomposition et la manière de l'incrémenter peut être complexe, car cela dépend de la structure hiérarchique du domaine ainsi que des caractéristiques du problème (objets, tâche abstraite initiale et état initial).
Notre approche adopte une technique plus ciblée qui s'appuie directement sur la structure du domaine. Nous observons que, pour chaque tâche abstraite non récursive, le processus de décomposition répétée est fini. Par conséquent, il suffit uniquement de limiter la décomposition des tâches abstraites récursives pour assurer la complétude. Nous appliquons l'approche suivante : pour chaque tâche abstraite récursive t dans le PDT, nous empêchons l'un de ses enfants ou enfants transitifs d'être identique à t. Plus précisément, si un nœud étiqueté par une méthode m est un enfant ou un descendant d'un nœud étiqueté par une tâche abstraite t, et que l'une des sous-tâches de la méthode m est t, alors nous excluons m du PDT. Cette technique garantit que l'expansion exhaustive de notre PDT reste finie. Si aucun DT solution n'est trouvé lorsque le PDT est complètement développé avec ces restrictions, nous intégrons alors toutes les méthodes m exclues ainsi que leurs sous-tâches dans le PDT et nous répétons le processus. Cette approche permet à notre algorithme d'assurer la complétude. [16] ont proposé une technique très similaire pour éviter les boucles infinies dans leur planificateur TOHTN HyperTensionN.
Empiriquement, cette méthodologie s'est avérée efficace. Notre planificateur a été capable de résoudre des problèmes issus des benchmarks de la Compétition Internationale de Planification (IPC) de 2020 et 2023 sans jamais avoir besoin d'intégrer une seule fois les méthodes exclues dans le PDT.

8 Evaluation

8.1 Planificateurs

Nous avons comparé SibylSat à d'autres planificateurs TOHTN basés sur SAT à l'état de l'art afin d'évaluer ses performances. L'évaluation a porté sur trois planificateurs : SibylSat [1], notre planificateur utilisant l'approche garantissant la complétude avec le solveur Glucose [3] ; Lilotane [21], un planificateur TOHTN SAT Lifted, arrivé deuxième lors de la Compétition Internationale de Planification (IPC) 2020 ; et pandaPIsatt-1iB [4], une version améliorée du planificateur TOHTN SAT original totSAT [6]. Les expériences ont été réalisées sur un système équipé d'un processeur Intel Core i7-12700H et de 32 Go de RAM. Chaque instance de problème disposait d'un temps d'exécution maximal de 10 minutes. L'évaluation a été effectuée sur tous les benchmarks TOHTN proposés lors des IPC 2020 et IPC 2023.

8.2 Métriques d'évaluation

Les performances des planificateurs sont évaluées à l'aide de deux métriques différentes :

8.2.1 Score IPC

Le score IPC mesure la rapidité avec laquelle un planificateur résout un problème. Ce score varie de 0 à 1 pour chaque problème, avec 1 indiquant que le planificateur a trouvé une

1. Le code source est disponible à l'adresse : https ://github.com/gaspard-quenard/sibylsat

solution en moins d'une seconde et 0 correspondant à un dépassement du temps limite. Il est calculé comme suit :

$$\text{Score IPC} = \begin{cases} 0 & \text{Si pas de plan trouvé} \\ \min\left(1, 1 - \frac{\log(t)}{\log(T)}\right) & \text{Sinon} \end{cases}$$

où T est le temps d'exécution maximal autorisé pour trouver un plan et t est le temps (en secondes) mis par le planificateur pour trouver un plan. L'utilisation des logarithmes dans le calcul permet de mieux discriminer les performances lorsque les temps de résolution sont faibles ; ainsi, un planificateur trouvant une solution significativement plus vite qu'un autre recevra un score proportionnellement bien meilleur, encourageant la recherche de solutions rapides.

8.2.2 Score de qualité

Le score de qualité mesure le makespan (c'est-à-dire le nombre total d'actions dans un plan). Il varie de 0 à 1, avec 1 correspondant au plan le plus court et 0 signifiant qu'aucune solution n'a été trouvée. Il est défini comme suit :

$$\text{Quality Score} = \begin{cases} 0 & \text{Si pas de plan trouvé} \\ \frac{C^{ref}}{C} & \text{Sinon} \end{cases}$$

où C est le makespan du plan trouvé par le planificateur et C^{ref} est le meilleur makespan parmi tous les planificateurs évalués.

8.3 Résultats

Un aperçu des résultats pour la couverture, le score IPC et le score de qualité est présenté dans le tableau 1. Nous observons que, parmi les 26 domaines des benchmarks, notre planificateur obtient un meilleur score IPC dans 19 d'entre eux et dépasse également les autres planificateurs en termes de couverture et de score de qualité. Notons que le score de qualité est calculé sur la base du premier DT solution trouvé et n'inclut donc pas les procédures d'amélioration de plan, comme celle proposée par Lilotane, qui permet de trouver le DT solution avec le plan le plus court dans un PDT au prix d'un temps d'exécution plus élevé.

Pour vérifier si la stratégie d'expansion proposée par SibylSat permet effectivement d'explorer une plus petite partie de l'espace de recherche avant de trouver un DT solution, nous comparons dans la Figure 3 le nombre de méthodes développées avant de trouver un DT solution par Lilotane et SibylSat, étant donné que les deux planificateurs utilisent la même structure pour représenter l'espace de recherche. La figure montre que notre approche réduit généralement le nombre de méthodes développées avant de trouver un DT solution dans la plupart des domaines. Dans certains domaines, tels que Towers et Childsnack, le nombre de méthodes développées par notre planificateur est similaire à celui de Lilotane, car Lilotane développe déjà le nombre minimal de nœuds dans sa structure pour trouver un DT solution. Le seul benchmark où notre planificateur développe plus de méthodes que Lilotane est le domaine AssemblyHierarchical (indiqué par les marqueurs en étoile violette dans la Figure 3). Cela se reflète dans les résultats, car il s'agit du seul benchmark où le score IPC de SibylSat est inférieur à celui de Lilotane et de PandaPIsatt-1iB. Une analyse du domaine AssemblyHierarchical révèle la cause sous-jacente de ce résultat : la plupart des tâches abstraites de ce domaine sont récursives et fortement interconnectées, ce qui entraîne un ensemble d'effets possibles plus vaste pour chaque tâche. En raison de cette complexité, les effets possibles de chaque tâche abstraite sont étendus et conduisent souvent notre planificateur à identifier des DT solutions relaxés peu pertinents. Une amélioration de notre algorithme d'inférence des effets possibles des tâches abstraites pourrait probablement aider à résoudre ce type de problème.

Domain	# instances	SibylSat	Lilotane	PandaPIsatt-1iB
AssemblyHierarchical	30	2.61	3.89	**4.05**
Barman-BDI	20	**16.42**	15.40	15.05
Blocksworld-GTOHP	30	**25.66**	22.56	21.11
Blocksworld-HPDDL	30	**6.74**	0.80	2.91
Childsnack	30	**27.09**	26.78	21.56
Depots	30	**25.49**	22.05	24.80
Elevator-Learned	147	**146.91**	114.23	137.45
Entertainment	12	8.49	2.46	**11.53**
Factories-simple	20	**6.12**	3.77	5.99
Freecell-Learned	60	**7.31**	5.26	6.25
Hiking	30	**24.87**	22.07	18.89
Lamps	30	15.40	0	**17.31**
Logistics-Learned	80	**60.94**	28.48	49.00
Minecraft-Player	20	**2.66**	2.49	1.43
Minecraft-Regular	59	**32.78**	29.55	29.82
Monroe-FO	20	18.38	**19.09**	11.56
Monroe-PO	20	17.07	**18.21**	9.47
Multiarm-Blocksworld	74	**11.66**	1.87	9.06
Robot	20	**10.92**	10.55	10.75
Rover-GTOHP	30	**21.74**	18.05	18.54
Satellite-GTOHP	20	**14.70**	12.39	14.64
SharpSAT	21	**10.21**	8.35	8.61
Snake	20	**19.79**	19.41	16.57
Towers	20	**9.51**	8.11	5.98
Transport	40	35.05	31.18	**35.49**
Woodworking	30	26.41	**30**	22.01
Coverage	948	**706**	583	664
Normalized coverage	26	**18.97**	15.45	17.61
IPC score		**604.93**	477.0	529.83
Normalized IPC score		**16.06**	13.22	13.93
Quality score		**689.53**	520.83	606.49
Normalized quality score		**18.37**	14.17	15.91

TABLE 1 – Performance de chaque planificateur sur les benchmarks IPC 2020 et IPC 2023. Chaque cellule contient le score IPC pour un domaine spécifique. La couverture totale, le score de qualité total et le score IPC total, ainsi que leurs valeurs normalisées par domaine, sont affichés dans les six dernières lignes.

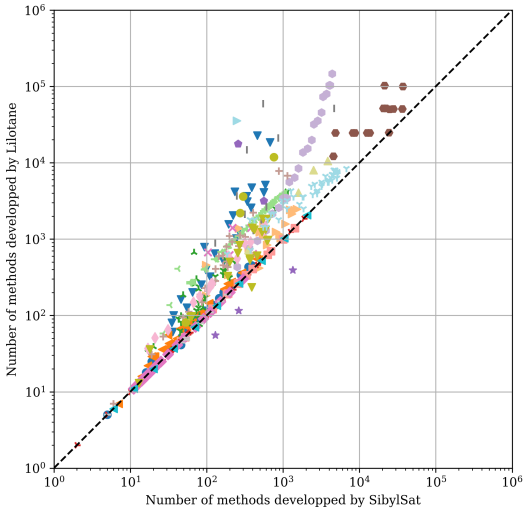


FIGURE 3 – Comparaison du nombre de méthodes développées par SibylSat et Lilotane. Un marqueur différent est utilisé pour chaque benchmark.

9 Travaux connexes

L'utilisation de planificateurs HTN basés sur SAT a été introduite pour la première fois par Mali et Kambhampati en 1998 [17], bien que leur représentation des problèmes HTN différait considérablement des formalismes actuels et que leur encodage ne permettait pas de gérer les domaines récursifs. Malgré cette première exploration, il s'est écoulé près de deux décennies sans recherches spécifiques sur la traduction des problèmes de planification HTN en logique propositionnelle. Ce n'est qu'en 2018 que totSAT [6] a été introduit, proposant une traduction SAT pour la planification TOHTN qui surpassait les autres planificateurs HTN à l'état de l'art, relançant ainsi l'intérêt pour cette approche. Contrairement à la planification classique, où les encodages sont étendus itérativement en fonction de la longueur du plan final, totSAT utilise une recherche en largeur d'abord pour étendre les encodages en fonction de la profondeur de la hiérarchie. À la suite de ce travail, Behnke et al. ont affiné leur approche pour gérer la planification HTN partiellement ordonnée [7], la recherche de plans optimaux [8], et ont proposé des techniques pour réduire l'espace de recherche et améliorer leur encodage [4].

Parallèlement à totSAT, Schreiber et al. ont proposé un nouvel encodage qui fut le premier à exploiter la résolution SAT incrémentale pour les problèmes HTN [22]. Cette approche a ensuite été améliorée avec le développement du planificateur Tree-REX [23], qui explore l'espace de recherche de manière similaire à totSAT, mais dont la traduction en logique propositionnelle est spécifiquement conçue pour la résolution SAT incrémentale, ce qui améliore les performances et réduit la taille des encodages. À la suite de ces travaux, Schreiber et al. ont introduit Lilotane [21], un successeur de Tree-REX et le premier planificateur TOHTN basé sur SAT utilisant une approche lifted. Lilotane évite le processus de mise à plat coûteux en utilisant une instanciation différée des tâches et méthodes, permettant des arguments libres lorsque nécessaire. Cela conduit à des formules SAT considérablement plus petites que les encodages précédents.

10 Conclusion

Dans cet article, nous avons présenté une nouvelle approche basée sur SAT pour résoudre les problèmes de planification TOHTN. Notre approche utilise un solveur SAT à la fois pour rechercher des solutions et pour guider l'exploration de l'espace de recherche. Plus précisément, nous avons montré comment l'encodage d'un problème relaxé peut permettre de dégager une heuristique grâce à un solveur SAT, qui peut ensuite être utilisée pour identifier des zones prometteuses dans l'espace de recherche. Nous avons démontré que notre planificateur, qui utilise cette heuristique pour explorer l'espace de manière gloutonne, surpasse les autres planificateurs TOHTN basés sur SAT à l'état de l'art en termes de temps d'exécution et de qualité des plans. Ce travail ouvre la voie à l'intégration d'informations heuristiques dans la planification TOHTN basée sur SAT.

Dans les travaux futurs, nous prévoyons d'explorer des méthodes allant au-delà de notre stratégie de recherche actuelle, qui repose sur une approche gloutonne. Plus précisément, nous souhaitons étudier si l'identification de plusieurs zones prometteuses dans l'espace de recherche à l'aide de notre heuristique, suivie d'un classement et d'une priorisation à l'aide d'heuristiques HTN classiques [13, 14, 15], pourrait améliorer encore davantage les performances.

11 Remerciements

Nous remercions les relecteurs pour leurs commentaires et suggestions. Ce travail a été partiellement soutenu par MIAI@Grenoble Alpes, (ANR-19-P3IA-0003).

Références

[1] Ron Alford, Gregor Behnke, Daniel Höller, Pascal Bercher, Susanne Biundo, and David Aha. Bound to plan : Exploiting classical heuristics via automatic translations of tail-recursive HTN problems. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 26, pages 20–28, 2016.

[2] Ronald Alford, Ugur Kuter, and Dana S Nau. Translating HTNs to PDDL : A Small Amount of Domain Knowledge Can Go a Long Way. In *IJCAI*, volume 9, pages 1629–1634, 2009.

[3] Gilles Audemard and Laurent Simon. On the glucose SAT solver. *International Journal on Artificial Intelligence Tools*, 27(01) :1840001, 2018.

[4] Gregor Behnke. Block compression and invariant pruning for SAT-based totally-ordered HTN planning. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 31, pages 25–35, 2021.

[5] Gregor Behnke, Daniel Höller, and Pascal Bercher, editors. *Proceedings of the 10th International Planning Competition : Planner and Domain Abstracts – Hierarchical Task Network (HTN) Planning Track (IPC 2020)*, 2021.

[6] Gregor Behnke, Daniel Höller, and Susanne Biundo. totSAT-Totally-ordered hierarchical planning through SAT. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[7] Gregor Behnke, Daniel Höller, and Susanne Biundo. Bringing order to chaos–A compact representation of partial order in SAT-based HTN planning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 7520–7529, 2019.

[8] Gregor Behnke, Daniel Höller, and Susanne Biundo. Finding Optimal Solutions in HTN Planning-A SAT-based Approach. In *IJCAI*, pages 5500–5508, 2019.

[9] Gregor Behnke, Florian Pollitt, Daniel Höller, Pascal Bercher, and Ron Alford. Making translations to classical planning competitive with other HTN planners. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 9687–9697, 2022.

[10] Kutluhan Erol, James A Hendler, and Dana S Nau. UMCP : A Sound and Complete Procedure for Hierarchical Task-network Planning. In *Aips*, volume 94, pages 249–254, 1994.

[11] Daniel Fišer. Lifted fact-alternating mutex groups and pruned grounding of classical planning problems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9835–9842, 2020.

[12] Thomas Geier and Pascal Bercher. On the decidability of HTN planning with task insertion. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, volume 22, page 1955, 2011.

[13] Daniel Höller, Pascal Bercher, Gregor Behnke, and Susanne Biundo. A generic method to guide HTN progression search with classical heuristics. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 28, pages 114–122, 2018.

[14] Daniel Höller, Pascal Bercher, Gregor Behnke, and Susanne Biundo. On Guiding Search in HTN Planning with Classical Planning Heuristics. In *IJCAI*, pages 6171–6175, 2019.

[15] Daniel Höller, Pascal Bercher, Gregor Behnke, and Susanne Biundo. HTN planning as heuristic progression search. *Journal of Artificial Intelligence Research*, 67 :835–880, 2020.

[16] Maurício C Magnaguagno, Felipe Rech Meneguzzi, and Lavindra De Silva. HyperTensioN : A three-stage compiler for planning. In *Proceedings of the 30th International Conference on Automated Planning and Scheduling (ICAPS), 2020, França.*, 2020.

[17] Amol Dattatraya Mali and Subbarao Kambhampati. Encoding HTN Planning in Propositional Logic. In *AIPS*, pages 190–198, 1998.

[18] Van-Hau Nguyen, Van-Quyet Nguyen, Kyungbaek Kim, and Pedro Barahona. Empirical Study on SAT-Encodings of the At-Most-One Constraint. In *The 9th International Conference on Smart Media and Applications*, pages 470–475, 2020.

[19] Conny Olz and Pascal Bercher. Can They Come Together ? A Computational Complexity Analysis of Conjunctive Possible Effects of Compound HTN Planning Tasks. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 33, pages 314–323, 2023.

[20] Conny Olz, Susanne Biundo, and Pascal Bercher. Revealing hidden preconditions and effects of compound HTN planning tasks–a complexity analysis. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 11903–11912, 2021.

[21] Dominik Schreiber. Lilotane : A lifted SAT-based approach to hierarchical planning. *Journal of artificial intelligence research*, 70 :1117–1181, 2021.

[22] Dominik Schreiber, Damien Pellier, Humbert Fiorino, and Tomáš Balyo. Efficient SAT encodings for hierarchical planning. In *11th International Conference on Agents and Artificial Intelligence (ICAART 2019)*, 2019.

[23] Dominik Schreiber, Damien Pellier, Humbert Fiorino, et al. Tree-REX : SAT-based tree exploration for efficient and high-quality HTN planning. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 29, pages 382–390, 2019.

[24] Ayal Taitler, Ron Alford, Joan Espasa, Gregor Behnke, Daniel Fišer, Michael Gimelfarb, Florian Pommerening, Scott Sanner, Enrico Scala, Dominik Schreiber, et al. The 2023 International Planning Competition, 2024.

Study of the Computational Complexity of the Repair Problem on Simple Temporal Networks with Uncertainty

Frédéric Maris[1,2], Aïdin Sumic[3], Thierry Vidal[4], Bruno Zanuttini[1]

[1] Université Caen Normandie, ENSICAEN, CNRS, Normandie Univ
GREYC UMR6072, F-14000 Caen, France
[2] Université de Toulouse, IRIT
[3] ONERA de Toulouse, DTIS
[4] Université de Technologie Tarbes Occitanie Pyrénées, PICS

frederic.maris@irit.fr, aidin.sumic@onera.fr, thierry.vidal@uttop.fr, bruno.zanuttini@unicaen.fr

Résumé

Les réseaux temporels simples avec incertitude (STNU) sont un modèle basé sur les contraintes conçu pour vérifier la contrôlabilité temporelle d'un plan sous incertitude (lorsque certaines durées contingentes ne peuvent être déterminées par l'agent de planification). Quoi qu'il en soit, dans les cas de planification multi-agents ou dépendante des ressources, il peut être possible de négocier ou de reconsidérer la flexibilité exogène et d'ajuster les limites des durées incontrôlables. Plusieurs travaux proposent une solution optimale à ce problème de réparation selon les trois niveaux de contrôlabilité (faible, dynamique, forte). Cependant, aucun d'entre eux ne caractérise la complexité théorique de ce problème. Cet article propose une évaluation approfondie de la complexité du problème de réparation pour les STNU.

Mots-clés

Complexité, satisfaction de contraintes, incertitude temporelle, réseaux temporels.

Abstract

Simple Temporal Networks with Uncertainty (STNU) are a constraint-based model designed to check the temporal 'controllability' of a plan under uncertainty (when some contingent durations cannot be decided by the planning agent). Anyway, in multi-agent or resource-dependent planning cases, it may be possible to negotiate or reconsider exogenous flexibility and adjust the bounds of the uncontrollable durations. Several works exist that provide the optimal solution to this repair problem depending on the three controllability levels (Weak, Dynamic, Strong). However, none of them characterize the theoretical complexity of this problem. This paper will provide a thorough complexity evaluation of the repair problem for STNU.

Keywords

Complexity, constraint satisfaction, temporal uncertainty, temporal networks.

1 Introduction

In many domains one needs to reason on activities that may or must not overlap in time, last over some duration, and synchronize with timestamped expected events. That is particularly true in planning and scheduling, where existing systems often use some explicit constraint-based representation [6, 18, 17, 4].

An efficient model for managing temporal durations is the Simple Temporal Network (STN) [8]: nodes are timepoints, and edges are constraints expressing convex intervals of possible durations between them. STNs are widely used to check a plan's temporal *consistency*. Consistency checking is made through polynomial-time propagation algorithms (e.g., the Floyd-Warshall reduction) and provides a complete *minimal* network in which all inconsistent values are removed [8]. This minimal network can be passed on to the plan execution manager, which can take any value on the domain of the first activity to schedule, 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 [19]. An STNU contains uncertain (*contingent*) durations between time-points, which means the effective duration is not controlled by the agent executing the plan. This is useful for addressing realistic dynamic and stochastic domains.

Temporal consistency is then redefined as *controllability*: an STNU is controllable if a strategy exists for executing the plan, whatever the values taken by the contingent durations. Three levels of controllability exist that depend on when the contingents' duration will be known: just before execution (*Weak*), during execution (*Dynamic*), or never (*Strong*).

Checking is one thing, but what to do when a plan is temporally uncontrollable has received less attention. There are many ways to respond to that at different levels. The highest is replanning, i.e., reconsidering the course of actions leading to the goal. Then, one may only reason at the scheduling level: switching the order of mutually exclusive actions, swapping a resource for another faster

one, etc; such changes are lighter and could be sufficient to change some durations for the better. Last, the lightest thing to do is only to alter the temporal durations themselves, if possible. But the durations that the executing agent controls have already been shrunk as much as possible by propagation algorithms (as in any constraint-based model), while altering a contingent duration is impossible and even forbidden as it is controlled by some other entity, often *Nature* itself.

In [15], the authors argue that contingents may be reduced in some applications in which their durations depend on the plan quality measured through some *utility* of chosen actions. Hence, they proposed some algorithms that reduce contingents to recover controllability (of STNUs) even if it results in solutions of lesser quality.

For instance, the uncertain duration of a data downloading for an observation satellite may depend on the (exogenous) imprecise size of such data: forcing a lower upper bound is possible, meaning sending incomplete or less detailed data in extreme cases, resulting in a lower quality of the plan. Another example is in manufacturing cells in which machines are set up (speed, tools) to provide the best quality, and production plans are generated in such a framework; but it is still possible in case of temporally uncontrollable plans to change those settings, e.g., by authorizing higher speed, at the price of a lower quality: in section 4 we will provide a more precise example under such assumptions.

In other words, *Nature* there appears to be some external entity, not involved into the planning and executing processes, but over which some preprocessing control is still possible.

More interesting is the case of multi-agent planning, as replanning is a last-resort solution there, as it will impinge on the coordinated execution of the whole group; the most local and least committed repair will be sought. Moreover, uncertainty often comes from the decisions of other agents: an activity duration being decided by some other agent, but shared just before execution, or during execution, or not communicated at all. In that regard, the authors in [16] proposed a new multi-agent model for STNUs that define such a particular constraint called *contract*, and proposed some algorithms to repair any non-controllable STNU by reducing these *contracts*.

These scenarios show the relevance of the STNU Repair problem (but also in higher classes of Temporal Networks under Uncertainty). However, there exists no formal proof of its computational complexity: this paper aims to fill the gap.

In the remainder we first expose the necessary background on STNUs and some related work in Sections 2 and 3. Then, in Sections 4 and 5, we characterize the STNU Repair problems in a new way and based on that we present our study of their complexities. Next, we will discuss the case of repairing for multiple agents in Section 6. Finally, we will conclude our study with some prospects.

2 Background

2.1 Simple Temporal Network

A Simple Temporal Network (STN) [8] is a tuple $\langle \mathcal{V}, E \rangle$, where $\mathcal{V}$ is a set of real-value variables called timepoints $\{v_0, \dots, v_n\}$, and E is a set of temporal constraints between these timepoints called *requirements* [8]. Specifically, each requirement constraint $e_k \in E$ is of the form $v_j - v_i \in [l, u]$, with $v_i, v_j \in \mathcal{V}$, $l \in \mathbb{R} \cup \{-\infty\}$, and $u \in \mathbb{R} \cup \{+\infty\}$. l and u represent the minimal and maximal possible temporal distance between v_i and v_j. A requirement constraint can be represented also as an edge $v_i \xrightarrow{[l,u]} v_j$.
A reference timepoint v_0 is generally included in $\mathcal{V}$ as a fixed temporal anchor for all other timepoints. v_0 is assumed to be the first executed timepoint of the network, i.e., there is an implicit constraint $v_0 \leqslant v_i$ for each $v_i \in \mathcal{V}$.
An STN is *consistent* (i.e., satisfiable) if an assignment (schedule) to timepoints exists that satisfies all the constraints. We say that a controller executes an STN when it schedules its timepoints. Consistency checking is done in polynomial time and achieved by transforming an STN into a *distance graph*. A distance graph is a directed graph where each requirement constraint ($v_i \xrightarrow{[l,u]} v_j$) is split into two edges: $v_i \xrightarrow{u} v_j$ and $v_j \xrightarrow{-l} v_i$ allowing positive and negative weights. An STN is inconsistent if its distance graph contains a negative cycle, which can be detected using shortest-path algorithms.

2.2 STNs with Uncertainty

An STN with Uncertainty (STNU) is an extension of STNs in which one distinguishes a subset of constraints whose values (duration) are decided by external entities (i.e., *Nature*) and the controller can only observe [19].

Definition 1 (STNU). *An STNU is a tuple $\langle \mathcal{V}, E, C \rangle$ with:*

- *$\mathcal{V} = \mathcal{V}_c \cup \mathcal{V}_u$ is a set of timepoints: $\mathcal{V}_c$, the set of controllable timepoints, and $\mathcal{V}_u$, the set of contingent (uncontrollable) ones.*
- *E is a set of requirement constraints as in an STN.*
- *C is a set of contingent links. A contingent link c_k is of the form $v_j - v_i \in [l, u]$, where $v_i \in \mathcal{V}_c$, $v_j \in \mathcal{V}_u$, and $0 \leqslant l \leqslant u$. The value $\omega_k = v_j - v_i$ is called* duration*, and an external entity decides it before or during execution. Once v_i is executed, the value of v_j is $v_i + \omega_k$. Any c_k is also depicted as $v_i \overset{[l,u]}{\dashrightarrow} v_j$.*

Definition 2 (Schedule). *A **schedule** for an STNU $\mathcal{X}$ is a mapping $\delta \colon \mathcal{V} \to \mathbb{R}$ from timepoints to real values.*

Definition 3 (Situation, Projection and Solution). *Given an STNU $\mathcal{X} = \langle \mathcal{V}, E, C \rangle$, the **situations** of $\mathcal{X}$ is a set of tuples Ω defined as the Cartesian product of contingent domains: $\Omega = \times_{c_k \in C}[l, u]$. Each situation $\omega = \{\omega_1, \dots, \omega_n\} \in \Omega$ represents one possible complete set of values for the duration of the contingent links of $\mathcal{X}$. A **projection** $\mathcal{X}^\omega = (\mathcal{V}, E \cup C^\omega)$ of $\mathcal{X}$ is an STN where $C^\omega = \{[\omega_k, \omega_k] \mid c_k \in$*

$C\}$. *A* ***solution*** *of* $\mathcal{X}^{\omega}$ *is a schedule* δ_{ω} *satisfying all the constraints.*

Intuitively, a projection replaces each contingent link with a rigid requirement constraint, i.e., a constraint reduced to a one-value range associated with each contingent link in ω.
In STNUs, consistency needs to be redefined: a network is now said to be *controllable* if there exists a schedule that satisfies all requirement constraints in any possible projection.

Definition 4 (Decision and Observation)**.** $\forall v_i \in \mathcal{V}_c$, $dec(v_i)$ *is the instant at which* $\delta(v_i)$ *is* ***decided*** *by the controller.*
$\forall \omega_k \in C$, $obs(\omega_k)$ *is the instant at which* ω_k *is* ***observed*** *by the controller.*

The controllability properties were defined in [19], and their semantics refined in [16]: *Strong Controllability (SC)* assumes one common schedule δ satisfies all constraints for all situations, which is relevant when external events durations cannot be observed (conformant planning). On the contrary, *Weak Controllability* (WC) assumes there is one consistent schedule for each situation, which is relevant when all contingent durations will be known just before execution (oracle). In-between, the *Dynamic Controllability (DC)* assumes the schedule values assignment depends on past observations only, regardless of the contingent durations still to be observed.

Definition 5 (Weak Controllability (WC))**.** *An STNU* $\mathcal{X}$ *is* ***weakly controllable*** *iff* $\forall \omega \in \Omega, \exists \delta$ *s.t.* δ_{ω} *is a solution of* $\mathcal{X}_{\omega}$.
Execution semantics: $\forall \omega_k \in \omega$, $obs(\omega_k) = v_0$, *and the decision policy is free:* $\forall v_i \in \mathcal{V}_c$, $dec(v_i) \leqslant v_i$.

Definition 6. ***(Strong Controllability (SC) with Execution)*** *An STNU* $\mathcal{X}$ *is* ***strongly controllable*** *iff* $\exists\ \delta$ *such that* $\forall \omega \in \Omega$, δ *is a solution of* $\mathcal{X}_{\omega}$.
Execution semantics: $\forall v_i \in \mathcal{V}_c$, $dec(v_i) = v_0$, *and the observations are free: possibly no observation (*$\forall \omega_k \in \omega$, $obs(\omega_k) = \varnothing$*) or observations during execution that will just update the bounds of the constraints in the network.*

Definition 7. ***(Dynamic Controllability (DC) with Execution)*** *An STNU* $\mathcal{X}$ *is* ***Dynamically controllable*** *iff it is Weakly controllable and* $\forall v_i \in \mathcal{V}_c, \forall \omega, \omega' \in \Omega$, $\omega^{\leq v_i} = \omega'^{\leq v_i} \implies \delta_{\omega}(v_i) = \delta'_{\omega}(v_i)$
where $\omega^{\preceq v} = \{\omega_k \in \omega$ *s.t.* $obs(\omega_k) \preceq dec(v)\}$ *is the part of the situation* ω *in which contingent constraints values are observed before executing* v.
Execution semantics: $\forall \omega_k \in \omega$, $obs(\omega_k)$=$end(c_k)$, *and* $\forall v_i \in V_c$, $dec(v_i) = v_i$

Dynamic and Strong Controllability have proven to be retractable and solvable in polynomial time [19, 10], while WC has been proven to be a co-NP complete problem [12]. In fact, DC and WC checking rely on algorithms that look for negative cycles in the *distance graph* of STNUs [10, 15].

3 Related Work

The notion of repair arises in [5]. They introduced a Multi-agent STNU (MaSTNU) model in which agents have their own plans, and hence own temporal networks, but all face common exogenous contingent constraints. They proposed a new algorithm for checking the Dynamic Controllability (DC) of MaSTNU using a Mixed Integer Linear Programming (MILP) approach. The authors claim that it should be possible to modify the encoding to reduce the bounds of the contingents so that all networks are DC.
Then, in [2], the authors compute the volume space of an STNU to assess just how far from being controllable an uncontrollable STNU is by defining some metrics for Strong and Dynamic Controllability. Later, they propose an incomplete Linear Programming approach to repair a non-DC STNU by repairing the negative cycles [3]. However, it is incomplete because it assumes that negative cycles are independent, which is not always the case.
Recently, the repair problem for STNU was formally defined in [15]. The authors tackle the case of WC and SC through Satisfiability Modulo Theory (SMT) and propose an optimization function that finds the minimal reduction of the contingent bounds to repair the network.
Later on, they proposed a distributed extension of STNUs called Multi-Agent Interdependent STNUs (MISTNU) that introduced the notion of *contract*, a shared constraint being controllable for one agent but contingent for others [16]. Hence, a contingent, being now actually controlled by another agent of the system (that is not *Nature*), becomes negotiable, making the repair problem of STNU clearly more relevant than in MaSTNU. In addition, they extend the SMT encodings to repair non-controllable MISTNU.
However, such encodings are sensitive to the number of contingents/contracts, hence, poorly scalable. Therefore, they argue that using propagation-based algorithms to identify and repair sources of uncontrollability should be more efficient and scalable.
In that regard, the closest related works that focus on STNUs and diagnosis, i.e., pinpointing reasons for non-controllability, address DC [11] and WC [14] by providing new checking algorithms that are informed, i.e., that return the sources of uncontrollability in the form of negative cycles in the distance graph of STNU. Please note that the former approach [11] for DC returns only one negative cycle at a time. Hence, one must use an iterative process to repair all of them. Yet, no repair algorithm currently exists that is based on these informed algorithms.
In this paper, we are interested in evaluating the theoretical complexity of the repair problem. Therefore, the following section will generalize the previous studies by introducing different types of repair problems relevant enough for STNUs to determine their complexity.

4 Repair Problems

In this section, we formally define the repair problems. In the following definitions, τ stands for any controllability level in $\{\mathsf{S}, \mathsf{D}, \mathsf{W}\}$ (standing for "Strong", "Dynamic",

and "Weak", respectively). We start with some new fundamental notions.

Definition 8 (Tightening and τ-repair)**.** *Let $\mathcal{X} = \langle \mathcal{V}, E, C \rangle$ be an STNU. A **tightening** of $\mathcal{X}$ is a set of ordered pairs $\rho = \langle\langle c_1, [l'_1, u'_1]\rangle, \ldots, \langle c_k, [l'_k, u'_k]\rangle\rangle$ such that $c_1, \ldots, c_k \in C$ are pairwise distinct contingent constraints of $\mathcal{X}$ and for $i = 1, \ldots, k$, if c_i is $v \xrightarrow{[l_i, u_i]} v'$, then $l_i \leqslant l'_i \leqslant u'_i \leqslant u_i$ holds. We write $\mathcal{X} \oplus \rho$ for the STNU $\langle \mathcal{V}, E, C' \rangle$, where C' is obtained from C by replacing each c_i by $v \xrightarrow{[l'_i, u'_i]} v'$.*
A τ-repair is a tightening such that $\mathcal{X} \oplus \rho$ is τ-Controllable.

Definition 9 (Repair Problem)**.** τ-REPAIR *is the decision problem:*

- Input: *An STNU $\mathcal{X}$*
- Question: *Is there a τ-repair ρ of $\mathcal{X}$?*

Definition 9 introduces a basic formulation of the repair problem. However, this minimal definition may not be sufficient in more realistic settings. Depending on the environment, such as the influence of *Nature* or other agents, or on specific optimization criteria and performance metrics relevant to the agent's objectives, a more refined definition may be required.
In order to define such refined definitions, we first introduce additional notation; let $\mathcal{X}$ be an STNU, $\rho = \langle\langle c_1, [l'_1, u'_1]\rangle, \ldots, \langle c_k, [l'_k, u'_k]\rangle\rangle$ a tightening of $\mathcal{X}$, and l_j, u_j the lower and upper bounds of c_j in $\mathcal{X}$; then we write

- $\mathrm{Supp}(\rho)$ for the set $\{j \in \{1, \ldots, k\} \mid l'_j \neq l_j \vee u'_j \neq u_j\}$, i.e., the set of constraints actually (strictly) repaired by ρ;
- $\mathrm{Cost}(\rho)$ for the quantity $\sum_{j=1,\ldots,k} l'_j - l_j + u_j - u'_j$, i.e., the sum of all interval reductions over all contingent constraints.

Definition 10 (Partial Repair)**.** PARTIAL-τ-REPAIR *is the following decision problem:*

- Input: *An STNU $\mathcal{X}$ and a subset R of its contingent constraints*
- Question: *Is there a τ-repair ρ of $\mathcal{X}$ such that $\mathrm{Supp}(\rho) \subseteq R$?*

The partial repair problem is relevant when only a subset of contingent durations can be reduced, such as in multi-agent settings where some durations are negotiable (e.g, belong to another agent) while others, controlled by *Nature*, must remain fixed.

Definition 11 (k-Budget Repair)**.** k-BUDGET-τ-REPAIR *is the following decision problem:*

- Input: *An STNU $\mathcal{X}$ and a rational number k*
- Question: *Is there a τ-repair ρ of $\mathcal{X}$ with $\mathrm{Cost}(\rho) \leqslant k$?*

The *k-budget* repair problem is particularly relevant in scenarios where optimizing a specific parameter is important, such as minimizing cost [7, 20], or maximizing flexibility or fairness among agents in multi-agent settings [16].

Definition 12 (k-constraint repair)**.** k-CONSTRAINT-τ-REPAIR *is the following decision problem:*

- Input: *An STNU $\mathcal{X}$ and an integer k*
- Question: *Is there a τ-repair ρ of $\mathcal{X}$ with $|\mathrm{Supp}(\rho)| \leqslant k$?*

The *k-constraint* repair problem is also relevant for optimization in multi-agent settings, where minimizing communication costs amounts to looking for the smallest set of contingents to negotiate.
To better grasp the STNU model and the different repair problems, we provide a realistic example, inspired by the application framework discussed in the Introduction, in Example 1.

Example 1. *Bob operates on a production line involving two machines, M_1 and M_2. Each day, a manager determines the production requirements following predefined regulations. The process begins with M_1 transforming raw materials into components. Bob then performs a quality check before passing the components to M_2, which completes the final product. The durations of tasks executed by M_1 and M_2 are uncertain, ranging from 5–12 minutes and 10–15 minutes, respectively, while Bob's task is controllable and takes between 5–15 minutes. The overall process must be completed within a time window of 25 to 30 minutes. To guarantee feasibility, the manager may adjust the machines' speed before execution, but at a cost.*

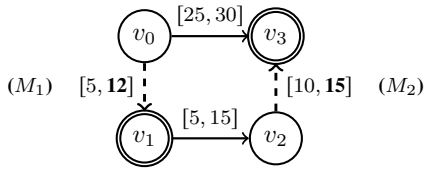
Figure 1 illustrates Bob's scheduling problem. The STNU is not weakly controllable, and a repair solution is shown with new bounds on M_1 and M_2. WC semantics is chosen here for its simplicity.

5 Complexity of Repair Problems

This section provides complexity results for the different types of repair problem (see Definitions 9, 10, 11 and 12) according to the semantics of the three controllability levels. The results of our study are summarized in Table 1.

Problem	Strong	Dynamic	Weak
Controllability	P [19]	P [10]	coNP-comp [12]
Repair	P	P	P
Partial repair	P	in NP	coNP-hard and in Σ_2^P
k-budget repair	P	in NP	coNP-hard and in Σ_2^P
k-constraint repair	NP-comp	NP-comp	coNP-hard and in Σ_2^P

Table 1: Complexity of controllability and repair problems ("comp" is short for "complete").



c_k	Repair	Partial R. (M_1)	2-Budget R.	1-Cons. R.
M_1	**[10, 10]**	**[7, 7]**	**[5, 11]**	[5, 12]
M_2	**[10, 10]**	[10, 15]	**[10, 14]**	**[10, 10]**

Figure 1: STNU of Example 1 where nodes represent timepoints (doubly circled ones are uncontrollable), edges are requirements and contingent constraints (the later relating to the tasks of machines M_1 and M_2). The STNU is not WC due to the highlighted projection $\omega = \{\omega_1 = 12, \omega_2 = 15\}$. The table highlights a solution to each type of repair problem.

5.1 Complexity of the Repair Problem

We study here the repair problem from definition 9, i.e., without an optimization criterion. We first introduce a straightforward lemma, which shows that when repairs are unconstrained, we can consider only repairs to singleton intervals without loss of generality, and additionally that we can restrict to rational numbers, of size polynomial in the size of the STNU. In fact, this is the same as considering a discretization of the intervals of the contingents with a polynomial number of possible values.

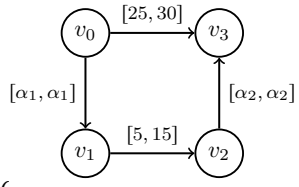
Lemma 1. *Let $\tau \in \{\mathsf{S}, \mathsf{D}, \mathsf{W}\}$ be a level of controllability. If an STNU $\mathcal{X}$ has a τ-repair $\langle\langle c_1, [l'_1, u'_1]\rangle, \ldots, \langle c_k, [l'_k, u'_k]\rangle\rangle$, then there exist $b_1, \ldots, b_k$ such that $\langle\langle c_1, [b_1, b_1]\rangle, \ldots, \langle c_k, [b_k, b_k]\rangle\rangle$ is also a τ-repair of $\mathcal{X}$.*

Proof. All three levels of controllability require that for *all* situations generated by the contingent constraints, the STNU is controllable. Hence, tighter bounds on the contingents can only make an STNU more controllable. □

Proposition 1. *Problems* S-REPAIR, W-REPAIR *and* D-REPAIR *are solvable in polynomial time.*

Proof. Lemma 1 shows that for all three levels of controllability, the repair problem has a solution if and only if it has one in which all contingent constraints are repaired to singleton intervals. Moreover, for a tightening ρ of $\mathcal{X} = \langle V, E, C\rangle$ in which all the contingent constraints of $\mathcal{X}$ are repaired to singleton intervals, the STNU $\mathcal{X} \oplus \rho$ is now an STN, since contingent durations are known (i.e., fixed).

It follows that we only have to search a value $\alpha_c \in [l, u]$ for each contingent $c = v_i \xrightarrow{[l,u]} v_j \in C$, such that the STN obtained from $\mathcal{X}$ by replacing each such c by the *requirement* $v_i \xrightarrow{[\alpha_c, \alpha_c]} v_j$, is satisfiable. This can clearly be done by solving a system of linear equations, as illustrated on Figure 2. □



$$\begin{cases} v_1 - v_0 \leqslant \alpha_1; v_1 - v_0 \geqslant \alpha_1 \\ v_3 - v_2 \leqslant \alpha_2; v_3 - v_2 \geqslant \alpha_2 \\ v_3 - v_0 \geqslant 25; v_3 - v_0 \leqslant 30 \\ v_2 - v_1 \geqslant 5; v_2 - v_2 \leqslant 15 \\ 5 \leqslant \alpha_1 \leqslant 12; 10 \leqslant \alpha_1 \leqslant 15 \end{cases}$$

Figure 2: Searching for a valid tightening of contingent constraints $\rho = \langle\langle c_1, [\alpha_1, \alpha_1]\rangle, \ldots, \langle c_2, [\alpha_2, \alpha_2]\rangle\rangle$ of the STNU of Figure 1 is equivalent to solving consistency of a polynomial size linear system where each α_i are variables.

5.2 Optimization Repairs for Strong Controllability

In this section, we will provide a new and polynomial algorithm for repairing a non-SC STNU that can be adapted for the partial and k-budget repair problems. For the k-constraint repair problem, we will reduce it to the SUBSET-SUM problem to prove that it is NP complete.

In [19], the authors prove that checking Strong Controllability can be reduced to solving a system of linear equations over the requirement constraints. In fact, any STNU can be encoded the same way as for STNs (see Figure 2) by replacing in the equations each uncontrollable timepoint $v_j \in \mathcal{V}_u$ by the relation between a controllable timepoint it relates to and the uncontrollable duration of the contingent constraint between them. The most obvious case is the one of a requirement constraint $v_i \xrightarrow{[l,u]} v_j$ where v_j is uncontrollable and relates to $v_k \xrightarrow{[l',u']} v_j$: in that case we write $v_j - v_i \in [l, u]$ as follows: $\begin{cases} (v_k + \omega_j) - v_i \geqslant l \\ (v_k + \omega_j) - v_i \leqslant u \end{cases}$

There exist three cases where an uncontrollable timepoint is involved: the one we showed where $v_j \in \mathcal{V}_u$, the one where $v_i \in \mathcal{V}_u$, and where $v_i \in \mathcal{V}_u$ and $v_j \in \mathcal{V}_u$. Nonetheless, the methodology remains the same, leading to a system of linear equations of polynomial size.

One can see that a schedule that satisfies such a system, whatever the duration of $\omega = \{\omega_1, \ldots, \omega_n\}$, is a 'strong' schedule that satisfies the SC Definition 6. Hence, if a schedule exists where for each linear equation ω takes the worst possible value, then such a schedule is a strong schedule. The worst possible value for ω is when the contingents take their lower/upper bound value. Thus, finding a solution is done in polynomial time. Figure 3 shows the encoding of checking SC with the STNU of Figure 1. First, we show with ω_1 and ω_2, then with the worst possible value for ω_1 and ω_2 in each linear equation. Formally, we have the following.

Proposition 2 ([19])**.** *Given an STNU $\mathcal{X}$, one can build*

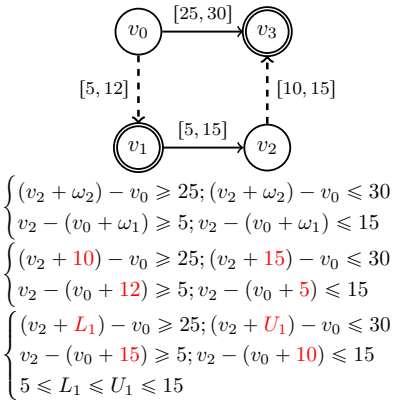


Figure 3: Reduction resulting from checking Strong Controllability. The two requirements to encode illustrate the first two cases of uncontrollable timepoints: $v_i \in \mathcal{V}_u$ with v_1, and $v_j \in \mathcal{V}_u$ with v_3. We highlight the worst-case scenario for each linear equation with the bounds' value. Then, the encoding to the partial-Strong-repair problem by replacing M_1's values (bounds) with proper variables L_1 and U_1.

in polynomial time a linear program over one variable per requirement constraint of $\mathcal{X}$, such that the solutions of this program are in one-to-one correspondence with the schedules which ensure that $\mathcal{X}$ is strongly controllable.

Proposition 3. *Problems* PARTIAL-S-REPAIR *and* k-BUDGET-S-REPAIR *are solvable in polynomial time.*

Proof. Given Proposition 2, it is easy to build a linear program of polynomial size which decides partial repairability, and k-budget repairability:

- for partial repairability, we replace in the linear program of Proposition 2, each numeric value corresponding to a lower (resp. upper) bound of a repairable contingent constraint $c = v \xrightarrow{[l,u]} v'$, by a new variable L (resp. U), and we add linear constraints $l \leqslant L \leqslant U \leqslant u$; it is then easy to see that the solutions of this linear program are in one-to-one correspondence with the repairs (of each c to $[L, U]$) and the corresponding strong schedules of the STNU. Figure 3 illustrates the encoding for the partial-Strong-repair of Figure 1 assuming only M_1 can be reduced;
- for k-budget repairability, we similarly replace the numeric values of all contingent constraints, again enforce $l \leqslant L \leqslant U \leqslant u$ for all of them, and additionally enforce the linear constraint:

$$\sum_{v \xrightarrow{[l,u]} v' \in C} (L - l + u - U) \leqslant k$$

□

Proposition 4. *Problem* k-CONSTRAINT-S-REPAIR *is* NP*-complete.*

Proof. For membership in NP, one simply need to guess a tightening ρ, consisting of the new bounds l', u' for at most k contingent links, then check that $\mathcal{X} \oplus \rho$ is strongly controllable (in polynomial time using Proposition 2). Since STNUs consist of linear constraints only, it is easy to see that one can restrict to guessing polynomial-size numbers (in the size of the numbers appearing in the STNU).

For hardness, we give a reduction from SUBSET-SUM, which is defined as follows and known to be NP-complete [9, Sec. A3.2].

- *Input:* A multiset of (strictly) positive integers $\{\{n_1, \ldots, n_\ell\}\}$ and an integer N
- *Question:* Is there $I \subseteq \{1, \ldots, \ell\}$ satisfying $N = \sum_{i \in I} n_i$?

Given an instance $S = \langle\{\{n_1, \ldots, n_\ell\}\}, N\rangle$, we write $M = \sum_{i=1}^{\ell} n_i$, and we define an STNU $\mathcal{X}^S$ derived from the multiset S as follows (see Figure 4):

- the set of controllable timepoints is $V_c^S = \{v_0, v_1, \ldots, v_\ell\}$, with v_0 acting as the reference timepoint, and the set of uncontrollable timepoints is $V_u^S = \{v_1^{\mathsf{y}}, \ldots, v_\ell^{\mathsf{y}}\} \cup \{v_1^{\mathsf{n}}, \ldots, v_\ell^{\mathsf{n}}\}$;
- the set of requirement constraints is $E^S = \{e_1^{\mathsf{y}}, \ldots, e_\ell^{\mathsf{y}}\} \cup \{e_1^{\mathsf{n}}, \ldots, e_\ell^{\mathsf{n}}\} \cup \{e^N\}$, with $e_i^{\mathsf{y}} = v_i^{\mathsf{y}} \xrightarrow{[0,n_i]} v_i$, $e_i^{\mathsf{n}} = v_i^{\mathsf{n}} \xrightarrow{[0,+\infty]} v_i$ $(i = 1, \ldots, \ell)$, and $e^N = v_0 \xrightarrow{[M+N, M+N]} v_\ell$;
- the set of contingent constraints is $C^S = \{c_1^{\mathsf{y}}, \ldots, c_\ell^{\mathsf{y}}\} \cup \{c_1^{\mathsf{n}}, \ldots, c_\ell^{\mathsf{n}}\}$, with $c_i^{\mathsf{y}} = v_{i-1} \xrightarrow{[0,n_i]} v_i^{\mathsf{y}}$ and $c_i^{\mathsf{n}} = v_{i-1} \xrightarrow{[n_i,2n_i]} v_i^{\mathsf{n}}$ $(i = 1, \ldots, \ell)$.

Moreover, we define the number of repairable constraints k to be ℓ.

Clearly, $\mathcal{X}^S$ can be constructed in polynomial time given S. We now claim that S is a positive instance of SUBSET-SUM if and only if $\mathcal{X}^S$ is k-constraint strongly repairable.

First assume that I is a solution of S, that is, $N = \sum_{i \in I} n_i$ holds. We define the k-constraint tightening ρ^I to be $\langle\langle c_i^{\mathsf{y}}, [n_i, n_i]\rangle \mid i \in I\rangle \cdot \langle\langle c_j^{\mathsf{n}}, [n_j, n_j]\rangle \mid j \notin I\rangle$ ($\cdot$ denotes concatenation of tuples). Since exactly one constraint per index $i \in \{1, \ldots, \ell\}$ is repaired, we indeed have $|\mathrm{Supp}(\rho^I)| \leqslant k$. Finally, we define a set of decisions dec by $dec(v_i) = \sum_{j \leqslant i} n_j + \sum_{j \leqslant i, j \in I} n_j$ for $i = 0, \ldots, \ell$ (in particular, $dec(v_0) = 0$). Then it is easy to prove that the schedule δ induced by dec and ω satisfies all the constraints in E^S, so that ρ^I is a k-constraint strong repair of $\mathcal{X}^S$.

Conversely, assume that ρ is a k-constraint strong repair of $\mathcal{X}^S$. We first show that for $i = 1, \ldots, \ell$, ρ repairs either c_i^{y} or c_i^{n} to $[n_i, n_i]$. For contradiction, assume first that it repairs neither, and let ω be a situation with $\omega(c_i^{\mathsf{y}}) = 0$ and $\omega(c_i^{\mathsf{n}}) = 2n_i$; then in order to satisfy constraints e_i^{y} and e_i^{n},

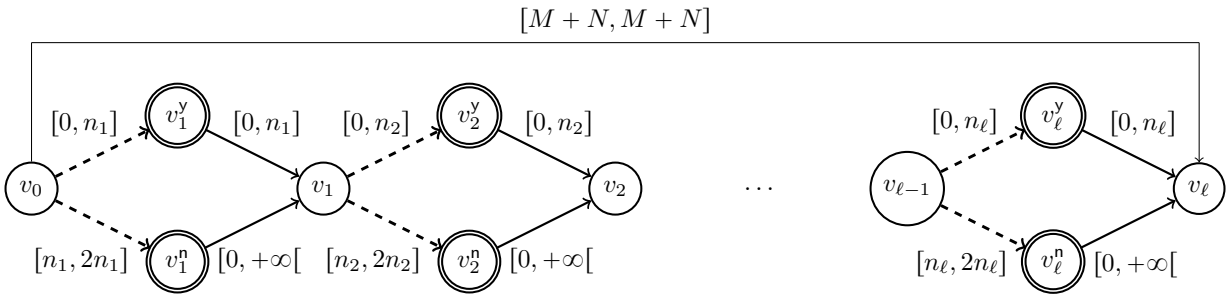


Figure 4: The STNU $\mathcal{X}_S$ built from an instance S of SUBSET-SUM.

a schedule δ must satisfy

$$\delta(v_{i-1}) + 2n_i = \delta(v_{i-1}) + \omega(c_i^{\mathsf{n}}) \leqslant \delta(v_i) \leqslant \delta(v_{i-1}) + \omega(c_i^{\mathsf{y}}) + n_i = \delta(v_{i-1}) + n_i,$$

contradicting $n_i > 0$. Hence ρ repairs at least one of c_i^{y} and c_i^{n} for each $i = 1, \ldots, \ell$, and given the budget of $k = \ell$ constraints, it must repair exactly one for each i.
First consider the case when ρ repairs c_i^{y}, and let δ be a strong schedule in the repaired STNU. Then δ is valid in particular for a situation ω with $\omega(c_i^{\mathsf{n}}) = 2n_i$, so that we have

$$\delta(v_{i-1}) + 2n_i = \delta(v_{i-1}) + \omega(c_i^{\mathsf{n}}) \leqslant \delta(v_i) \leqslant \delta(v_{i-1}) + \omega(c_i^{\mathsf{y}}) + n_i,$$

which entails $\omega(c_i^{\mathsf{y}}) = n_i$ and (hence) $\delta(v_i) = \delta(v_{i-1}) + 2n_i$. Dually, if ρ repairs c_i^{n} and δ is a strong schedule in the repaired STNU, then considering a situation ω with $\omega(c_i^{\mathsf{y}}) = 0$, we have

$$\delta(v_{i-1}) + n_i = \delta(v_{i-1}) + \omega(c_i^{\mathsf{y}}) + n_i \geqslant \delta(v_i) \geqslant \delta(v_{i-1}) + \omega(c_i^{\mathsf{n}}),$$

which entails $\omega(c_i^{\mathsf{n}}) = n_i$ and $\delta(v_i) = \delta(v_{i-1}) + n_i$.
In the end, a strong schedule δ in the repaired STNU satisfies for each $i = 1, \ldots, \ell$ either $\delta(v_i) = \delta(v_{i-1}) + n_i$ or $\delta(v_i) = \delta(v_{i-1}) + 2n_i$. Hence it satisfies $\delta(v_\ell) - \delta(v_0) = M + \sum_{i \in I} n_i$ for some subset I of $\{1, \ldots, \ell\}$; moreover, it satisfies $\delta(v_\ell) - \delta(v_0) = M + N$ due to constraint e^N, so $\sum_{i \in I} n_i = N$ holds and S is a positive instance of SUBSET-SUM. □

5.3 Optimization Repairs for Dynamic Controllability

This section will show that partial-Dynamic-repair and k-budget Dynamic repair problems are in NP, while the k-constraint Dynamic repair is NP-complete.

Proposition 5. PARTIAL-D-REPAIR *and* k-BUDGET-D-REPAIR *are in* NP.

Proof. The proof is similar to that for Strong repair (Proposition 4): we can guess a repair and then check that it is indeed a repair in polynomial time [13]. □

Proposition 6. *Problem* k-CONSTRAINT-D-REPAIR *is* NP*-complete.*

Proof. Membership in NP follows exactly from the same reasoning as in the proof of Proposition 5. For hardness, we use exactly the same reduction as in the proof of Proposition 4, and we show that $\mathcal{X}^S$ is k-constraint strongly repairable if and only if it is k-constraint dynamically repairable. One direction is obvious, since by definition Strong controllability implies Dynamic controllability.
Conversely, the proof that at least one contingent link per i (and hence exactly one) must be repaired applies verbatim from Proposition 4. Now for $i \in \{1, \ldots, \ell\}$, let δ_i be a dynamic schedule in the repaired STNU, for the case when ω satisfies $\omega(c_i^{\mathsf{n}}) = 2n_i$ (resp. $\omega(c_i^{\mathsf{y}}) = 0$) if c_i^{y} (resp. c_i^{n}) has been repaired. Then δ_i is fixed, and the same reasoning as in the proof of Proposition 4 applies, concluding that $\mathcal{X}$ is (k-constraint) strongly repairable. □

5.4 Optimization Repairs for Weak Controllability

This section will show that partial-Weak repair, k-budget Weak repair, and k-constraint Weak repair problems are at least coNP-hard and at most in Σ_2^{P}.

Proposition 7. PARTIAL-W-REPAIR, k-BUDGET-W-REPAIR *and* k-CONSTRAINT-W-REPAIR *are* coNP*-hard.*

Proof. We consider particular cases for which the problem comes down to checking Weak Controllability which is known to be coNP-complete [12]. By setting $R = \varnothing$, and $k = 0$, then no contingent can be repaired and the STNU is repairable if and only if it is weakly controllable. Therefore, all of them are as hard as the problem of checking WC. □

Proposition 8. PARTIAL-W-REPAIR, k-BUDGET-W-REPAIR *and* k-CONSTRAINT-W-REPAIR [1] *are in* Σ_2^{P}.

Proof. It is possible to guess in polynomial time a tightening ρ such that $\mathrm{Supp}(\rho) \subseteq R$ for the partial-Weak repair problem, $\mathrm{Cost}(\rho) = k$ for the k-budget Weak repair

[1] We recently found a proof that k-constraints Weak repair problem is Σ_2^{P}-complete. Indeed, Morris et al. proposed a reduction of the famous 3-color problem to WC checking of STNUs [12]. We can extend their proof to reduce the problem 2-round 3-colorability, which is known to be Σ_2^{P}-complete [1, Theorem 11.4], to k-constraints Weak repair problem. The completeness remains open for the other two repair problems.

problem, and $|\mathrm{Supp}(\rho)| = k$ for the k-constraint Weak repair problem. Checking that $\mathcal{X} \oplus \rho$ is weakly controllable can be verified by a coNP oracle. Hence, we get the membership to Σ_2^{P}. □

6 The Multi-agent Case

In this section, we discuss the complexity of the repair problem for more than one agent. For instance, the repair problem for MISTNU is more than relevant and determining its complexity is more challenging [16].

There exist two ways of repairing STNUs in a multi-agent system. One can try to centralize the repair problem using some global function that merges the local repair problem of each agent into a big problem. This is possible when formulating the problem using a first-order formula or a system of equations to solve as done in [5, 16]. In that particular case, where the repair problem is solved in a centralized way, its complexity remains the same as repairing only one STNU.

The real challenge arises when solving the repair problem distributedly is required. More (external) criteria exist that may influence the hardness of the repair problem, related to the information that is exchanged and the size of the messages, which can grow exponentially, or to communication failure or possible delays (of uncertain duration), zone of communication (dependent on the application) that can impact when information will be known; related to the nature of the agent (cooperative or selfish); or related to the global architecture and policy of the agents: pre-existing hierarchy, or on the contrary fairness issues; etc. Therefore, depending on the scenario, one repair may be harder than another and, thus, does not belong to the same class. Nonetheless, this would require a thorough evaluation of the distributed repair problem.

7 Conclusion

In this paper, we studied the complexity of the repair problem for STNU by introducing four definitions of this problem. We evaluated each of them regarding the semantics of each controllability level. At the same time, we proved their completeness for most of them. The problem remains open for some of them.

Future works will study the complexity of the repair problem for the higher class of Temporal Network with Uncertainty not only for the single agent case, but also for the multi-agent case.

References

[1] Miklós Ajtai, Ronald Fagin, and Larry J. Stockmeyer. The closure of monadic NP. *J. Comput. Syst. Sci.*, 60(3):660–716, 2000.

[2] 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*, volume 29, pages 22–30, 2019.

[3] Shyan Akmal, Savana Ammons, Hemeng Li, Michael Gao, Lindsay Popowski, and James C. Boerkoel. Quantifying controllability in temporal networks with uncertainty. *Artificial Intelligence*, 289:103384, 2020.

[4] Arthur Bit-Monnot, Malik Ghallab, Félix Ingrand, and David E. Smith. FAPE: a constraint-based planner for generative and hierarchical temporal planning. *CoRR*, abs/2010.13121, 2020.

[5] 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)*, volume 285 of *Frontiers in Artificial Intelligence and Applications*, pages 930–938. IOS Press, 2016.

[6] Amedeo Cesta, Angelo Oddi, and Stephen F. Smith. A constraint-based method for project scheduling with time windows. *J. Heuristics*, 8(1):109–136, 2002.

[7] Jing Cui, Peng Yu, Cheng Fang, Patrik Haslum, and Brian Charles Williams. Optimising bounds in simple temporal networks with uncertainty under dynamic controllability constraints. In *Proceedings of the Twenty-Fifth International Conference on Automated Planning and Scheduling, ICAPS 2015, Jerusalem, Israel, June 7-11, 2015*, pages 52–60. AAAI Press, 2015.

[8] Rina Dechter, Itay Meiri, and Judea Pearl. Temporal constraint networks. *Artificial intelligence*, 49(1-3):61–95, 1991.

[9] M. R. Garey and David S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

[10] Luke Hunsberger and Roberto Posenato. Speeding Up the RUL$^-$ Dynamic-Controllability-Checking Algorithm for Simple Temporal Networks with Uncertainty. In *36th AAAI Conf. on Artificial Intelligence 2022*, pages 9776–9785, 2022. DOI: 10.1609/aaai.v36i9.21213.

[11] 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.

[12] Paul H. Morris and Nicola Muscettola. Managing temporal uncertainty through waypoint controllability. In *Proceedings of the Sixteenth International Joint Conference on Artificial*

Intelligence, IJCAI 99, Stockholm, Sweden, July 31 - August 6, 1999. 2 Volumes, 1450 pages, pages 1253–1258. Morgan Kaufmann, 1999.

[13] Paul H. Morris, Nicola Muscettola, and Thierry Vidal. Dynamic control of plans with temporal uncertainty. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence, IJCAI 2001, Seattle, Washington, USA, August 4-10, 2001*, pages 494–502. Morgan Kaufmann, 2001.

[14] Aïdin Sumic and Thierry Vidal. A more efficient and informed algorithm to check weak controllability of simple temporal networks with uncertainty. In *18èmes Journées d'Intelligence Artificielle Fondamentale et 19èmes Journées Francophones sur la Planification, la Décision et l'Apprentissage pour la conduite de systèmes, JIAF-JFPDA 2024, La Rochelle, France, July 1-3, 2024*, pages 159–169, 2024.

[15] Ajdin Sumic, Alessandro Cimatti, Andrea Micheli, and Thierry Vidal. Smt-based repair of disjunctive temporal networks with uncertainty: Strong and weak controllability. In *Integration of Constraint Programming, Artificial Intelligence, and Operations Research - 21st International Conference, CPAIOR 2024, Uppsala, Sweden, May 28-31, 2024, Proceedings, Part II*, volume 14743 of *Lecture Notes in Computer Science*, pages 176–192. Springer, 2024.

[16] Ajdin Sumic, Thierry Vidal, Andrea Micheli, and Alessandro Cimatti. Introducing interdependent simple temporal networks with uncertainty for multi-agent temporal planning. In *31st International Symposium on Temporal Representation and Reasoning, TIME 2024, October 28-30, 2024, Montpellier, France*, volume 318 of *LIPIcs*, pages 13:1–13:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2024.

[17] Alessandro Valentini, Andrea Micheli, and Alessandro Cimatti. Temporal planning with intermediate conditions and effects. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 9975–9982. AAAI Press, 2020.

[18] Gérard Verfaillie, Cédric Pralet, and Michel Lemaître. How to model planning and scheduling problems using constraint networks on timelines. *Knowl. Eng. Rev.*, 25(3):319–336, 2010.

[19] Thierry Vidal and Hélène Fargier. Handling contingency in temporal constraint networks: from consistency to controllabilities. *Journal of Experimental & Theoretical Artificial Intelligence*, 11(1):23–45, 1999.

[20] Benjamin W. Wah and Dong Xin. Optimization of bounds in temporal flexible planning with dynamic controllability. In *16th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2004), 15-17 November 2004, Boca Raton, FL, USA*, pages 40–48. IEEE Computer Society, 2004.

Session 5 : Elicitation et agrégation de préférences

Identification de plusieurs fonctions de valeurs additives à partir de préférences exprimées anonymement

Vincent Auriau[1,2], Khaled Belahcène[1], Emmanuel Malherbe[2], Vincent Mousseau[1], Marc Pirlot[3]

[1] MICS, CentraleSupélec, Université Paris Saclay, France
[2] Artefact Research Center, France
[3] MATHRO, Université de Mons, Belgique

vincent.auriau@artefact.com, khaled.belahcene@centralesupelec.fr, emmanuel.malherbe@artefact.com, vincent.mousseau@centralesupelec.fr, marc.pirlot@umons.ac.be

Résumé

Eliciter un modèle de préférences additif consiste à poser une suite de questions à un décideur pour identifier une fonction de valeur additive représentant ses préférences. Dans ce travail, nous considérons un groupe de décideurs avec pour objectif d'éliciter une fonction de valeur pour représenter les préférences de chacun. Nous considérons le cas où l'on questionne deux décideurs, mais sans savoir quelle réponse correspond à quel décideur. Nous proposons une procédure d'élicitation identifiant les deux modèles de préférences lorsque les fonctions marginales représentées par des fonctions additives linéaires par morceaux.

Mots-clés

Modèle de préférence additif linéaire par morceaux, élicitation de modèles multiples, identification.

1 Présentation du problème et procédure d'élicitation

Le modèle de valeur additive est un choix standard pour spécifier des préférences sur un ensemble d'alternatives évaluées selon plusieurs critères, voir [3]. Plus précisément, étant donnée une alternative x définie par son évaluation sur n critères $x = (x_1, \ldots, x_n)$, où x_i est l'évaluation de x sur le critère i, la valeur de x est définie par $u(x) = \sum_i u_i(x_i)$, où u_i est la fonction de valeur marginale sur le critère i. Dans cet article, nous considérons que ces fonctions marginales u_i sont strictement croissantes et linéaires par morceaux, avec une décomposition linéaire définie a priori. Dans la suite, nous appellerons un tel modèle de préférence un modèle UTA. Afin d'éliciter un modèle UTA, il est courant d'utiliser des requêtes standards correspondant à des questions de type "matching" [4], impliquant deux alternatives x et y dont les évaluations ne diffèrent que sur deux critères i et j. Trois des évaluations parmi x_i, x_j, y_i, et y_j sont fixées, et le décideur répondant fournit la valeur manquante de manière à ce que x et y soient indifférents (noté $x \sim y$). Cela implique que $u_i(x_i) + u_j(x_j) = u_i(y_i) + u_j(y_j)$. L'ensemble des points (x_i, x_j) dans le plan (i, j) pour lesquels $u_i(x_i) + u_j(x_j)$ est constant forme une courbe d'indifférence dans ce plan. De telles questions de matching, permettent identifier les points appartenant à une courbe d'indifférence [2], ce qui sera utile dans la suite.

Nous considérons un framework dans lequel l'objectif est d'éliciter simultanément deux modèles UTA. Dans ce contexte, la réponse à une requête de matching consiste en deux évaluations a et a' pour la composante non fixée. Ces réponses sont fournies de manière anonyme, sans possibilité de savoir à quel décideur appartient chaque réponse. Le but de cet article est d'étudier ce problème sous l'angle de l'identifiabilité. Autrement dit, nous cherchons à savoir s'il est toujours possible de définir une suite finie de requêtes dont les réponses permettent d'identifier les deux modèles de valeur additive linéaire par morceaux. Ce problème d'identifiabilité peut aussi être formulé comme un jeu dans lequel un premier joueur spécifie les requêtes et un second joueur fournit des réponses (véridiques). La question devient alors : existe-t-il une stratégie gagnante finie pour le premier joueur ? Une autre variante du jeu (non traitée ici) permettrait au second joueur de répondre avec un bruit (adversarial).

Bien que le problème soit formulé de manière purement formelle, il se retrouve bien dans des cas d'usages concrets. Par exemple, un supermarché souhaite généralement adapter la liste de ses produits à sa base de clientèle et sélectionner ces produits sur la base de leurs préférences. Toutefois, une telle population montre généralement des compertements et préférences très différents et il est standard de considérer une segmentation de la clientèle. Chaque segment représente alors un groupe de clients aux préférences homogènes, et dans notre cas un décideur dont la fonction de valeur est à éliciter.

Exemple 1 *Dans cet article, nous utiliserons comme fil conducteur l'expression de préférences pour le choix d'une voiture électrique selon deux critères : l'autonomie (de 100 km à 600 km) et le prix (de 10 000€ à 50 000€). Les réponses aux requêtes de préférence sont obtenues via un système en ligne garantissant l'anonymat. Deux décideurs expriment leurs préférences :* Commuter *utilise la voiture*

dans un contexte urbain pour de courtes distances, tandis que Traveler *effectue de longs trajets.* Commuter *et* Traveler *disposent respectivement de 30 000€ et 40 000€ d'épargne liquide ; ils doivent contracter un prêt pour les voitures dont le prix dépasse leur épargne.*

Ce type de problème a déjà été abordé dans une perspective d'apprentissage de modèle de préférence [1]. Le résultat d'identifiabilité proposé ici est fondamental pour justifier la recherche d'algorithmes efficaces permettant d'inférer plusieurs modèles UTA à partir de déclarations de préférences. Notre objectif est de fournir une procédure d'élaboration systématique permettant d'identifier deux fonctions de valeur additives linéaires par morceaux à partir d'une séquence finie de requêtes de matching. Dans cette introduction, nous proposons une présentation générale de cette procédure. Le principal résultat de l'article peut se formuler comme suit.

Theorem 1.1 *Il est possible d'éliciter deux modèles de préférence additifs linéaires par morceaux à partir d'informations de préférence anonymisées en utilisant un nombre fini de requêtes d'indifférence.*

Dans notre cadre, chaque requête de matching, ou d'indifférence, implique des alternatives qui varient uniquement selon deux critères i et j. Trois valeurs sont fixées par l'Éliciteur, et les réponses sont fournies par les différents Décideurs de telle sorte à ce qu'elles correspondent à deux couples d'alternatives indifférentes. L'espace de valeur des critères (i, j) définit un plan où les requêtes peuvent être entièrement représentées, tout le reste étant égal. En particulier, étant donné que les fonctions de valeur marginales sont linéaires par morceaux, les courbes d'indifférence sont également représentées par des fonctions linéaires par morceaux (différentes). Dans ce plan des critères, il est utile de définir deux types de requêtes à la géométrie spécifique que nous utiliserons comme «blocs de construction» de notre procédure d'élaboration :
Requête sur un seul rectangle : les valeurs de la requête ainsi que les réponses sont contenues dans un seul rectangle défini par les segments linéaires, définis a priori, sur les deux échelles de critères.
Requête sur des rectangles adjacents : les valeurs de la requête sont choisies de manière à impliquer deux rectangles voisins définis par les segments linéaires sur les échelles de critères.
Les requêtes sur un seul rectangle permettent d'éliciter les pentes anonymisées des fonctions de valeur marginales sur un intervalle donné, tandis que les requêtes sur des rectangles adjacents fournissent des contraintes de couplage permettant d'assigner les pentes aux deux fonctions de valeur.
En utilisant ces requêtes standards, la procédure peut être résumée comme suit :
Initialisation : la procédure commence avec les deux premiers critères et postule (sans perte de généralité) que la pente sur le critère 1 est égale à un sur le premier segment. Une requête sur un seul rectangle identifie la pente de chaque modèle pour le premier segment du critère 2.
Itération : étant donné un segment ℓ élaboré sur le critère 1, une succession de requêtes permet d'identifier les pentes des deux modèles sur le segment $\ell + 1$; il en va de même pour le critère 2.
Généralisation : pour obtenir l'ensemble des fonctions de valeur marginales, on applique le même principe à toutes les paires de critères (1,i), $i \neq 1$.
La suite de l'article est organisée comme suit : après une revue des travaux connexes en Section 2, la Section 3 introduit les notations générales. La Section 4 présente les deux types de requêtes servant de blocs de construction pour la procédure d'identification. La Section 5 présente les résultats justifiant la nature itérative de la procédure générale, développée en Section 6.

Références

[1] Vincent Auriau, Khaled Belahcène, Emmanuel Malherbe, and Vincent Mousseau. Learning multiple multicriteria additive models from heterogeneous preferences. In Ruper Freeman and Nicholas Mattei, editors, *Algorithmic Decision Theory*, pages 207–224. Springer, 2025.

[2] Denis Bouyssou and Marc Pirlot. Conjoint measurement tools for mcdm. In Salvatore Greco, Matthias Ehrgott, and José Rui Figueira, editors, *Multiple Criteria Decision Analysis : State of the Art Surveys*, pages 97–151. Springer New York, 2016.

[3] D.H. Krantz, R.D. Luce, P. Suppes, and A. Tversky. *Foundations of Measurement - Volume 1 : Additive and Polynomial Representations*. Academic Press, Incorporated, 1971.

[4] D. von Winterfeldt and W. Edwards. *Decision analysis and behavioral research*. Cambridge University Press, Cambridge, 1986.

Accord entre Règles de Vote sous des Distributions de Préférences “Single-Peaked"

Vincent Mousseau[1], Henri Surugue[1], Anaëlle Wilczynski[1]

[1] MICS, CentraleSupélec, Paris-Saclay

Résumé

De nombreuses règles de vote ont été proposées dans la littérature et elles peuvent sélectionner des alternatives très différentes. La question se pose alors de savoir si cette diversité de gagnants se produit souvent en pratique. Des travaux antérieurs ont montré que la probabilité que les règles de vote s'accordent sur le même résultat est généralement assez faible sous des cultures de préférences uniformes. Dans cet article, nous utilisons une approche probabiliste similaire en se penchant sur des cultures “single-peaked", qui sont plus structurées et plus réalistes que les cultures uniformes. Nous fournissons des conditions pour que les règles de vote s'accordent sous ces cultures et nous montrons que la probabilité d'accord d'une grande famille de règles de vote est beaucoup plus élevée dans ce cadre, avec une convergence rapide en fonction du nombre de votants. Enfin, nous donnons un aperçu d'autres distributions de préférences structurées, en observant que nombre d'entre elles présentent une convergence similaire, y compris la distribution de Mallows. Notre étude révèle que plusieurs cultures de vote bien établies ont tendance à biaiser le résultat des règles de vote dans une même direction, ce qui vaudrait d'être connu avant de mener des expériences sur des données synthétiques.

Mots-clés

Choix social computationnel, Théorie du vote, Restrictions de préférences, Etude statistique.

1 Introduction

L'un des principaux thèmes de la théorie du vote est la conception de bonnes règles de vote. Cependant, la littérature sur le choix social est célèbre pour ses théorèmes d'impossibilité, par exemple les théorèmes d'Arrow ou de Gibbard-Satterthwaite, qui suggèrent qu'il n'existe pas de règle de vote parfaite. De nombreuses règles de vote différentes ont été conçues dans la littérature, et elles peuvent sélectionner des alternatives très différentes. On peut alors se demander si ce comportement se produit souvent. Cette question a été soulevée par de nombreux articles [6] qui étudient la probabilité que différentes règles de vote soient en accord sur leur résultat.

La plupart des travaux sur l'accord des règles de vote se concentrent sur les cultures impartiales. Cependant, ces cultures ne rendent pas compte de préférences réelles de votants, qui sont généralement loin d'être uniformément distribuées. En outre, la plupart des résultats sur les cultures impartiales soulignent que les règles de vote donnent rarement le même résultat. Par conséquent, l'exploration de cultures plus structurées et plus réalistes peut fournir de nouvelles informations sur les différences entre les règles de vote. Dans cet article, nous nous concentrerons sur des cultures générant des préférences dites “single-peaked" [1], qui sont pertinentes dans plusieurs contextes tels que, par exemple, les élections politiques où un axe gauche-droite peut structurer les préférences de la plupart des votants. Même si les cultures “single-peaked" sont encore loin de correspondre parfaitement à des données réelles, elles sont beaucoup plus réalistes et peuvent être considérées comme une meilleure approximation de la réalité dans certains contextes.

D'un autre point de vue, l'étude de l'accord entre les règles de vote dans les cultures “single-peaked" peut également améliorer la compréhension de ces dernières tant d'un point de vue théorique que pour les expériences. En effet, une question clé en choix social computationnel, et en particulier dans la théorie du vote, est de savoir comment générer des données synthétiques pertinentes pour des expériences sur les élections [2]. La réalisation d'une étude empirique à l'aide de simulations numériques peut en effet s'avérer très utile pour étayer ou compléter des résultats théoriques dans de nombreux problèmes de vote [3]. Par conséquent, l'étude de l'accord des règles de vote dans les cultures “single-peaked" est pertinente pour mieux comprendre ces cultures qui sont couramment utilisées, et mieux interpréter les études expérimentales. Dans l'absolu, la connaissance du fonctionnement de l'outil statistique est une condition préalable à une bonne étude empirique.

Dans cet article, nous étudions la probabilité d'accord de différentes règles de vote dans des cultures “single-peaked". À notre connaissance, cette question a été étonnamment négligée pour des cultures plus structurées que les cultures impartiales. Une exception notable est le travail de Chatterjee and Storcken [4] sur les profils unimodaux.

2 Le Modèle

Soient N un ensemble de n votants, et M un ensemble de m candidats. Les préférences de chaque votant $i \in N$ sont représentées par un ordre linéaire $\succ_i$ sur M. Le profil de préférences est noté $\succ = (\succ_i)_{i \in N}$ et Π^m désigne l'ensemble de tous les ordres de préférence possibles sur m candidats. Un profil de préférences $\succ \in (\Pi^m)^n$ est dit "single-peaked" s'il existe un axe $>$ sur M tel que, pour tout votant $i \in N$ et tout triplet de candidats $x > y > z$, on ait $y \succ_i x$ ou $y \succ_i z$.
Les gagnants de l'élection sont déterminés par une règle de vote qui associe à chaque profil de préférences un sous-ensemble non vide de candidats. Une règle de vote par score sélectionne les candidats maximisant une fonction de score attribuant un score à chaque candidat. Si le score d'un candidat est calculé en fonction de son rang dans l'ordre de préférence de chaque votant, la règle de vote associée est appelée une *règle de score positionnelle (PSR)*. Les PSRs incluent les règles de vote comme pluralité, veto, k-approbation, ainsi que la règle de Borda. Une règle de vote est dite *Condorcet-consistante* si elle élit le vainqueur de Condorcet lorsqu'il existe, c'est-à-dire le candidat qui bat tous les autres en comparaison par paires.

3 Résultats

Nous étudions d'abord la distribution de Walsh [7], qui correspond à la culture impartiale sur le domaine "single-peaked". Nous établissons que cette distribution favorise fortement les candidats médians, c'est-à-dire les candidats situés au centre de l'axe "single-peaked", puisqu'ils sont les gagnants espérés de toute PSR. Ensuite, nous donnons une caractérisation de toutes les PSRs pour lesquelles les candidats médians sont les seuls gagnants espérés, et montrons qu'elle correspond asymptotiquement à toutes les règles Condorcet-consistantes. Nous donnons également une borne inférieure exponentielle pour la probabilité d'accord de ces règles de vote.
Ensuite, nous examinons la distribution de Conitzer [5], qui sélectionne uniformément le candidat en tête pour générer des ordres "single-peaked". Nous caractérisons les gagnants espérés de toutes les règles de k-approbation, ainsi que de toutes les PSRs qui tendent à élire les candidats médians, lesquels sont également le résultat asymptotique des règles Condorcet-consistantes, et nous donnons une borne inférieure exponentielle pour cette probabilité.
Dans une étude orthogonale, nous identifions des distributions "single-peaked" qui ne favorisent, par construction, aucun candidat, relativement à une PSR donnée.
Enfin, nous explorons d'autres distributions structurées de préférences afin de déterminer si des résultats similaires peuvent être obtenus. Nous étudions des distributions unimodales, incluant la célèbre distribution de Mallows, et donnons quelques perspectives sur les urnes de Pólya-Eggenberger.

4 Conclusions et Perspectives

Nous avons identifié des cultures dans lesquelles l'accord des différentes règles de vote est rapide lorsque le nombre de votants augmente. Ainsi, les conclusions tirées d'expériences testant différentes règles de vote pour un problème donné doivent être interprétées avec prudence. On pourrait imaginer des conclusions très différentes sur un problème, non pas en raison du problème lui-même, mais en raison de la culture utilisée : cultures impartiales contre cultures "single-peaked", par exemple. Les travaux futurs pourraient porter sur la probabilité d'accord lors d'élections finies en nombre de votants avec une culture de Pólya-Eggenberger. La difficulté réside toutefois dans la structure dépendante de cette distribution. Une idée pourrait également être de considérer des distributions quasi "single-peaked" pour combler l'écart entre les cultures impartiales et "single-peaked" et se rapprocher d'élections politiques réelles. Enfin, la même étude pourrait être réalisée dans un modèle stratégique où les votants peuvent manipuler.

Références

[1] Duncan Black. On the rationale of group decision-making. *Journal of Political Economy*, 56(1) :23–34, 1948.

[2] Niclas Boehmer, Piotr Faliszewski, Łukasz Janeczko, Andrzej Kaczmarczyk, Grzegorz Lisowski, Grzegorz Pierczyński, Simon Rey, Dariusz Stolicki, Stanisław Szufa, and Tomasz Wąs. Guide to numerical experiments on elections in computational social choice. In *Proceedings of IJCAI 2024*, pages 7962–7970, 2024.

[3] Felix Brandt, Vincent Conitzer, Ulle Endriss, Jérôme Lang, and Ariel D. Procaccia. *Handbook of computational social choice*. Cambridge University Press, 2016.

[4] Swarnendu Chatterjee and Ton Storcken. Frequency based analysis of collective aggregation rules. *Journal of Mathematical Economics*, 87 :56–66, 2020.

[5] Vincent Conitzer. Eliciting single-peaked preferences using comparison queries. In *Proceedings of AAMAS 2007*, pages 420–427, 2007.

[6] William V. Gehrlein and Dominique Lepelley. *Voting paradoxes and group coherence : the Condorcet efficiency of voting rules*. Springer Science & Business Media, 2010.

[7] Toby Walsh. Generating single peaked votes. *arXiv preprint arXiv :1503.02766*, 2015.

Session 6 : Argumentation

Attaques d'Ordre Supérieur et Incomplétude en Argumentation Abstraite

Sylvie Doutre[1], Marie-Christine Lagasquie-Schiex[2], Jean-Guy Mailly[1], Antonio Yuste-Ginel[3]

[1] Université Toulouse Capitole, IRIT, France
[2] Université de Toulouse, IRIT, France
[3] Universdad de Málaga, Espagne

{doutre,jean-guy.mailly,lagasq}@irit.fr, ayusteginel@uma.es

Résumé

*Nous proposons un cadre d'argumentation abstraite, appelé IHOAF (*Incomplete Higher-Order Argumentation Framework*), dans lequel arguments et attaques peuvent être incertains, et où la cible d'une attaque peut être un argument ou une autre attaque. Une définition des éléments acceptables dans ce cadre est proposée, des problèmes de décision associés sont définis, et leur complexité est analysée.*

Mots-clés

Argumentation Abstraite, Attaques d'Ordre Supérieur, Incomplétude.

Cet article est un résumé des contributions décrites dans [3].

L'argumentation abstraite est un champ de recherche établi, à la frontière entre l'informatique, les mathématiques, la linguistique et la philosophie. La modélisation mathématique de l'argumentation a été profondément influencée par les systèmes d'argumentation abstraits de Dung [4], qui sont des graphes orientés dans lesquels les noeuds représentent les arguments, et les arcs des attaques entre eux. Dans ce cadre, on ignore la nature, la structure interne et l'origine des arguments pour se concentrer sur des aspects dialectiques plus généraux. Différentes sémantiques sont utilisées pour sélectionner les extensions d'un système d'argument, c'est-à-dire des ensembles d'arguments considérés comme collectivement acceptables.

La simplicité du cadre de Dung le rend inadapté pour représenter des notions argumentatives qui ont un impact crucial sur l'acceptabilité des arguments, ce qui explique le développement de nombreuses généralisations de ce cadre par la communauté, comme l'ajout de nouveaux types d'interactions entre arguments, comme les relations de support, les relations d'ordre supérieur ou les interactions collectives [2] ; ou encore l'intégration d'incertitude dans le raisonnement, en particulier concernant la présence des différents éléments [6]. La combinaison de ces généralisations est nécessaire pour une meilleure représentation de scénarios argumentatifs réalistes.

Si [5] a proposé une combinaison qui prend en compte supports entre arguments et incertitude, dans cet article, nous introduisons une combinaison qui permet de représenter à la fois des attaques d'ordre supérieur (c'est-à-dire des attaques dirigées vers des attaques) et une incertitude qualitative concernant la présence des arguments et des attaques, ce qui induit un nouveau cadre appelé *système d'argumentation d'ordre supérieur incomplet*. Considérons l'exemple suivant : une discussion entre amis pour décider d'une activité à faire pendant la journée :

a : « Nous devrions aller à la plage cet après-midi, car il va y avoir du soleil. »

b : « Sam m'a dit qu'il va pleuvoir bientôt. » (b attaque a)

c : « Oui, mais ça ne durera pas d'après le bulletin météo d'hier. » (c attaque l'attaque de b vers a)

L'argument c ne contredit ni a ni b, mais le fait que b attaque a. Il s'agit d'une attaque d'ordre supérieur. De plus, étant donnée la nature imprécise des bulletins météo, l'argument c peut-être considéré comme incertain. De plus, même de courte durée, une averse peut être suffisante pour empêcher une sortie à la plage, ce qui veut dire que l'attaque de c vers l'attaque de b vers a peut potentiellement être ignorée. Ce type de scénario, qui combine les attaques d'ordre supérieur et l'incertitude concernant l'existence des arguments et des attaques, motive l'étude de notre nouveau cadre.

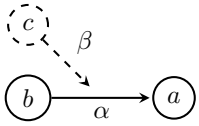
Dans la suite de ce résumé, pour des raisons de concision, nous supposons le lecteur ou la lectrice familier de l'argumentation abstraite [4].

Le nouveau cadre est formellement ainsi défini :

Definition 1 *Soit* $\mathbf{A}$ *un ensemble de noms d'arguments et* $\mathbf{R}$ *un ensemble de noms d'attaques. Un* Système d'Argumentation d'Ordre Supérieur Incomplet *(Incomplete Higher-Order Argumentation Framework,* IHOAF*) est un tuple* $\mathcal{IH} = (\mathcal{A}, \mathcal{A}^?, \mathcal{R}, \mathcal{R}^?, \mathbf{src}, \mathbf{trg})$ *où :*
– $\mathcal{A}$ *sont les* noms d'arguments certains *(*$\mathcal{A} \subseteq \mathbf{A}$*).*
– $\mathcal{A}^?$ *sont les* noms d'arguments incertains *(*$\mathcal{A}^? \subseteq \mathbf{A}$*).*
– $\mathcal{R}$ *sont les* noms d'attaques certaines *(*$\mathcal{R} \subseteq \mathbf{R}$*).*
– $\mathcal{R}^?$ *sont les* noms d'attaques incertaines *(*$\mathcal{R}^? \subseteq \mathbf{R}$*).*
– $\mathbf{src} : (\mathcal{R} \cup \mathcal{R}^?) \to (\mathcal{A} \cup \mathcal{A}^?)$ *est la* fonction source.
– $\mathbf{trg} : (\mathcal{R} \cup \mathcal{R}^?) \to (\mathcal{A} \cup \mathcal{A}^? \cup \mathcal{R} \cup \mathcal{R}^?)$ *est la* fonction cible.
On suppose également que $X \cap X^? = \varnothing$, pour $X \in$

$\{\mathcal{A}, \mathcal{R}\}$; et que pour chaque $\alpha, \beta \in (\mathcal{R} \cup \mathcal{R}^?)$, $\alpha \neq \beta$ implique $\mathbf{v}(\alpha) \neq \mathbf{v}(\beta)$, étant donné que pour $\alpha \in \mathcal{R}$, $\mathbf{v}(\alpha) = (\mathbf{src}(\alpha), \mathbf{trg}(\alpha))$, et pour $X \subseteq \mathcal{R}$, $\mathbf{v}(X) = \{\mathbf{v}(\alpha) | \alpha \in X\}$.

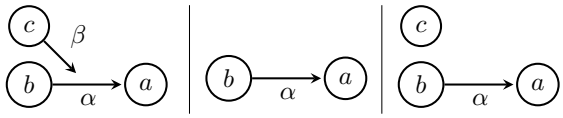
L'IHOAF $\mathcal{IH}$ correspondant à l'exemple de la discussion entre amis peut être représenté sous la forme du graphe suivant, où les noeuds représentent les arguments, les arcs orientés représentent les attaques, et où les éléments certains (resp. incertains) apparaissent avec un trait plein (resp. en pointillé).



Pour raisonner avec un IHOAF, ses *complétions* sont considérées : elles correspondent aux mondes possibles vis à vis de l'information incertaine. Formellement :

Definition 2 *Soit* $\mathcal{IH} = (\mathcal{A}, \mathcal{A}^?, \mathcal{R}, \mathcal{R}^?, \mathbf{src}, \mathbf{trg})$ *un IHOAF. Une complétion de* $\mathcal{IH}$ *est tout HOAF* $\mathcal{H} = (\mathcal{A}_c, \mathcal{R}_c, \mathbf{src}_c, \mathbf{trg}_c)$ *avec :*
– $\mathcal{A} \subseteq \mathcal{A}_c \subseteq (\mathcal{A} \cup \mathcal{A}^?)$ *et* $\mathcal{R}_c \subseteq (\mathcal{R} \cup \mathcal{R}^?)$.
– $\mathbf{v}(\mathcal{R}) \cap (\mathcal{A}_c \times (\mathcal{A}_c \cup \mathcal{R}_c)) \subseteq \mathbf{v}(\mathcal{R}_c) \subseteq (\mathbf{v}(\mathcal{R} \cup \mathcal{R}^?)) \cap (\mathcal{A}_c \times (\mathcal{A}_c \cup \mathcal{R}_c))$.
– $\mathbf{src}_c = \mathbf{src} \cap (\mathcal{R}_c \times \mathcal{A}_c)$.
– $\mathbf{trg}_c = \mathbf{trg} \cap (\mathcal{R}_c \times (\mathcal{A}_c \cup \mathcal{R}_c))$.
On note completions$(\mathcal{IH})$ *l'ensemble de toutes les complétions de* $\mathcal{IH}$.

Trois complétions sont possibles pour l'$\mathcal{IH}$ de l'exemple :



L'acceptabilité des éléments dans un IHOAF est définie relativement à une sémantique σ, et sous forme de *structures*. Les sémantiques complète co, préférée pr, grounded gr et stable st, classiquement définies par [4], peuvent être considérées. Les structures sont des paires (S, Γ) où S est un sous-ensemble des arguments (certains et incertains) du cadre, et Γ est un sous-ensemble de ses attaques (certaines et incertaines). Les structures acceptables sous une sémantique σ donnée (σ-structures) sont celles qui sont acceptables sous σ dans au moins une complétion du cadre.

Definition 3 *Soit* $\mathcal{IH} = (\mathcal{A}, \mathcal{A}^?, \mathcal{R}, \mathcal{R}^?, \mathbf{src}, \mathbf{trg})$ *un IHOAF.* $\sigma(\mathcal{IH})$ *dénote l'ensemble de toutes les* σ-structures *de* $\mathcal{IH}$ *sous la sémantique* σ*, défini comme* $\sigma(\mathcal{IH}) = \{(S, \Gamma) \in 2^{(\mathcal{A} \cup \mathcal{A}^?)} \times 2^{(\mathcal{R} \cup \mathcal{R}^?)} | \exists \mathcal{H} \in$ completions$(\mathcal{IH})$ *et* $(S, \Gamma) \in \sigma(\mathcal{H})\}$.

Dans l'exemple, pour $\sigma \in \{\mathsf{co}, \mathsf{pr}, \mathsf{gr}, \mathsf{st}\}$, chacune des 3 complétions de $\mathcal{IH}$ a une unique σ-structure. Ainsi, $\sigma(\mathcal{IH}) = \{(\{a, b, c\}, \{\beta\}), (\{b\}, \{\alpha\}), (\{b, c\}, \{\alpha\})\}$.

Des *problèmes de décision* classiques sont définis pour les IHOAF : déterminer, pour une sémantique σ donnée, si un élément d'un IHOAF appartient à au moins une (resp. toute) σ-structure d'au moins une complétion de cet IHOAF (acceptabilité crédule possible σ-PCA, resp. nécessaire σ-NCA), ou si x appartient à au moins une (resp. toute) σ-structure de toutes les complétions de cet IHOAF (acceptabilité sceptique possible σ-PSA, resp. nécessaire σ-NSA).

Dans l'$\mathcal{IH}$ exemple, pour $\sigma \in \{\mathsf{co}, \mathsf{pr}, \mathsf{gr}, \mathsf{st}\}$, a, c, α et β, qui appartiennent à toutes les structures de certaines des complétions, sont σ-PSA (et σ-PCA), et b, qui appartient à toutes les structures de toutes les complétions, est σ-NSA (et σ-NCA). Cet argument b est ainsi le seul élément accepté quelle que soit l'interprétation du cadre.

Les résultats de complexité concernant les problèmes de décision sont les suivants (et coïncident avec la complexité obtenue lorsqu'il n'y a pas d'attaques d'ordre supérieur [6]) :

σ	σ-PCA	σ-NCA	σ-PSA	σ-NSA
gr	NP-c	coNP-c	NP-c	coNP-c
co	NP-c	Π_2^{P}-c	NP-c	coNP-c
st	NP-c	Π_2^{P}-c	Σ_2^{P}-c	coNP-c
pr	NP-c	Π_2^{P}-c	Σ_3^{P}-c	Π_2^{P}-c

Un encodage logique du cadre des IHOAFs a été effectué en *Dynamic Logic of Propositional Assignments* (DL-PA) [1], pour calculer les complétions, les structures acceptables, et répondre aux problèmes de décision.

Les détails de cet encodage, le développement de toutes les idées présentées dans ce résumé et les perspectives ouvertes par ce travail, sont à retrouver dans [3].

Références

[1] Philippe Balbiani, Andreas Herzig, and Nicolas Troquard. Dynamic logic of propositional assignments : a well-behaved variant of PDL. In *Proc. of LICS'13*, pages 143–152, 2013.

[2] Pietro Baroni, Dov Gabbay, Guillermo R. Simari, and Matthias Thimm, editors. *Handbook of Formal Argumentation, volume 2*. College Publications, London, England, 2021.

[3] Sylvie Doutre, Marie-Christine Lagasquie-Schiex, Jean-Guy Mailly, and Antonio Yuste-Ginel. Incomplete higher-order abstract argumentation frameworks. In *CLAR 2025*, 2025.

[4] 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.

[5] Marie-Christine Lagasquie-Schiex, Jean-Guy Mailly, and Antonio Yuste-Ginel. How to Manage Supports in Incomplete Argumentation Frameworks. In *Proc. of FOIKS*, 2024.

[6] Jean-Guy Mailly. Yes, no, maybe, I don't know : Complexity and application of abstract argumentation with incomplete knowledge. *Argument Comput.*, 13(3) :291–324, 2022.

Sémantiques de Satisfaction Collective pour l'Argumentation Basée sur les Opinions

Juliete Rossie[1], Jérome Delobelle[2], Sebastien Konieczny[1], Clement Lens[1], Srdjan Vesic[1]

[1] CRIL, CNRS, Univ. Artois, Lens, France
[2] Universite Paris Cité, LIPADE, F-75006 Paris, France

{rossie,konieczny,lens,vesic}@cril.fr, jerome.delobelle@u-paris.fr

Résumé

Le vote sur les arguments au sein d'un débat constitue un moyen intuitif d'atteindre un consensus. Cependant, peu d'approches formelles en argumentation abstraite intègrent l'usage du vote pour la sélection des arguments acceptés. Nous présentons le cadre d'argumentation basée sur les opinions (OBA), qui permet aux individus de voter (ou de s'abstenir) en faveur ou contre des arguments dans un cadre d'argumentation de Dung. Notre objectif est de déterminer les décisions collectives les plus appropriées dans ce contexte. Pour ce faire, nous proposons une nouvelle sémantique, appelée Sémantique de Satisfaction Collective (CSS), visant à évaluer l'acceptabilité des arguments et à analyser leurs propriétés. En outre, nous évaluons de manière axiomatique ces sémantiques en les comparant aux approches alternatives issues de la littérature afin de mieux comprendre leur efficacité relative. Ce papier est une traduction du papier publié à la conférence KR 2023 [21].

Mots-clés

Argumentation, votes, sémantique de satisfaction collective.

Abstract

Voting on arguments within a debate offers an intuitive means of achieving consensus. However, few formal approaches in abstract argumentation address the integration of voting for selecting accepted arguments. We present the Opinion Based Argumentation (OBA) framework, enabling individuals to vote (or abstain) for or against arguments within a Dung argumentation framework. Our goal is to identify the most suitable collective decisions within this context. To this end, we propose a new semantics, termed Collective Satisfaction Semantics (CSS), to assess argument acceptability and analyze their properties. Furthermore, we axiomatically evaluate these semantics against alternative approaches from the literature to gain insights into their comparative effectiveness.

Keywords

Argumentation, Voting, Collective Satisfaction Semantics.

1 Introduction

L'argumentation est un outil puissant pour l'e-démocratie, permettant aux citoyens d'échanger arguments et attaques face à des enjeux cruciaux. Cependant, pour parvenir à une décision collective réellement démocratique, il ne suffit pas de confronter les arguments : il est indispensable d'évaluer le niveau d'accord entre les citoyens et de considérer leur sentiment collectif. Plusieurs plateformes en ligne [1] facilitent la création collective d'un graphe d'argumentation, certaines permettant aux participants de voter pour ou contre les arguments. Cependant, elles se limitent à visualiser l'état actuel du débat sans offrir d'outils de raisonnement pour analyser les arguments et évaluer l'issue collective. La manière la plus simple de représenter formellement les arguments et leurs interactions est le **système d'argumentation abstraite** (AF pour *Argumentation Framework*) de Dung [12], qui peut être vu comme un graphe orienté où les arguments sont les nœuds et les attaques sont les arêtes. Dans ce travail, l'AF est supposé complet, c'est-à-dire qu'il intègre l'ensemble des arguments et attaques liés au débat en cours. Nous considérons notamment que les arguments et attaques invalides ont été éliminés du graphe.

Dans ce travail, nous introduisons le **système d'argumentation basée sur les opinions** (OBAF pour *Opinion Based AF*), qui étend les AF de Dung en permettant aux individus de voter pour ou contre chaque argument, ou de s'abstenir. La question est alors la suivante : étant donné un OBAF, quel est le résultat optimal ? Pour aborder ce problème, nous introduisons une nouvelle famille de sémantiques, appelée **sémantiques de satisfaction collective** (CSS pour *Collective Satisfaction Semantics*), que nous comparons de manière axiomatique avec les travaux existants traitant d'un problème similaire. Ces travaux existants incluent ceux sur l'agrégation de jugements pour l'argumentation de Caminada et Pigozzi [7], où l'AF est également complet et les agents peuvent fournir un labelling [6] sur l'ensemble des arguments de l'AF. Le problème consiste alors à définir un labelling collectif satisfaisant. Nous montrons comment transformer les votes en labellings afin de comparer les approches. Une autre méthode est proposée par Bernreiter et al. [5], qui explorent les AFs sociaux basés sur l'approbation (ABSAF). L'approche commence par un AF prédé-

1. DebateGraph, Kialo, idebate, DebateArt, Arguman, etc.

fini, un ensemble fixe d'agents et des bulletins d'approbation. Deux opérateurs sont utilisés pour attribuer un score à chaque extension en fonction de ces bulletins, puis sélectionner la ou les « meilleures » extensions selon une sémantique prédéfinie. Cette approche présente certaines similitudes avec la nôtre, mais nous mettons en évidence dans la Section 2.3 plusieurs distinctions importantes.

Les différentes manières de représenter les votes sur les arguments dans les approches existantes nous ont conduit à définir notre système unifié et complet **OBAF**, ainsi qu'une sémantique unifiée, la **sémantique d'opinion collective** (COS, pour *Collective Opinion Semantics*). Dans la Section 4, nous montrons que l'approche d'agrégation de labellings [7] et l'approche par bulletins d'approbation [5] peuvent être définies comme des COS. Dans la même section, nous définissons également deux nouvelles approches : i) La première est basée sur une combinaison de l'agrégation de votes inspirée par [19] et de l'élimination des attaques utilisée dans les systèmes d'argumentation basés sur les préférences [3]. ii) La seconde est une famille de méthodes, CSS, basée sur l'agrégation des votes en utilisant trois fonctions de score (satisfaction, insatisfaction et utilité) avec des fonctions d'agrégation utilitaristes et égalitaristes. Dans la Section 5, nous introduisons une liste de dix propriétés pour COS, utilisées pour effectuer une analyse axiomatique dans la Section 6, et montrons en particulier que seule la CSS satisfait toutes les propriétés essentielles que nous avons mises en avant.

2 Notions de base

Dans cette section, nous établissons les définitions et concepts clés issus de la littérature existante afin de poser les bases de notre discussion et de notre analyse.

2.1 Argumentation Abstraite

Un système d'argumentation (AF) est un couple $\mathcal{F} = \langle \mathcal{A}r, att \rangle$, où $\mathcal{A}r$ est un ensemble fini d'arguments et $att \subseteq \mathcal{A}r \times \mathcal{A}r$ est la relation d'attaque [12]. Les sémantiques basées sur les extensions peuvent alors être utilisées pour déterminer quels arguments accepter. Un ensemble $\mathcal{E} \subseteq \mathcal{A}r$ est sans conflit s'il n'existe pas de $(x, y) \in att$ avec $x, y \in \mathcal{E}$. Un argument $x \in \mathcal{A}r$ est acceptable par rapport à $\mathcal{E}$ si pour tout $y \in \mathcal{A}r$ tel que $(y, x) \in att$, il existe un $z \in \mathcal{E}$ avec $(z, y) \in att$. Un ensemble $\mathcal{E}$ est : admissible s'il est sans conflit et que chaque $x \in \mathcal{E}$ est acceptable par rapport à $\mathcal{E}$; complet (co) s'il est admissible et contient tous les arguments acceptables par rapport à $\mathcal{E}$; préféré (pr) s'il est un ensemble admissible $\subseteq$-maximal. Nous notons l'ensemble des extensions selon $\sigma \in \{\mathtt{co}, \mathtt{pr}\}$ par $\mathcal{E}_\sigma(\mathcal{F})$.

Une alternative aux sémantiques basées sur les extensions est l'approche basée sur les labellings [6], où chaque argument est associé à un label parmi $\mathtt{in}, \mathtt{out}, \mathtt{undec}$, représentant respectivement l'acceptation, le rejet ou l'indécision. Caminada [6, 4] montre une correspondance formelle entre les sémantiques basées sur les extensions et les sémantiques basées sur les labellings. Par exemple, un labelling admissible satisfait des conditions sur les arguments acceptés et rejetés en fonction des attaquants, et un labelling complet correspond à un reinstatement labelling satisfaisant une condition de point fixe complet.

Exemple 1. *Considérons le AF $\mathcal{F}$ représenté dans la figure 1 (à gauche). L'ensemble des extensions de $\mathcal{F}$ avec la sémantique basée sur les extensions $\sigma \in \{\mathtt{co}, \mathtt{pr}, \mathtt{stb}\}$ est : $\mathcal{E}_{\mathtt{pr}}(\mathcal{F}) = \mathcal{E}_{\mathtt{stb}}(\mathcal{F}) = \{\{a, b\}, \{b, c\}, \{c, d\}, \{a, e\}, \{c, e\}\}$. $\mathcal{E}_{\mathtt{co}}(\mathcal{F}) = \mathcal{E}_{\mathtt{pr}}(\mathcal{F}) \cup \{\emptyset, \{b\}, \{c\}, \{e\}\}$. $\mathcal{L} = \{(a, \mathtt{out}), (b, \mathtt{out}), (c, \mathtt{in}), (d, \mathtt{undec}), (e, \mathtt{in})\}$ est un labelling de $\mathcal{F}$. Dans la suite de cet article, nous utiliserons aussi la notation simplifiée $\mathcal{L} = (\{c, e\}, \{a, b\}, \{d\})$ où le premier ensemble représente les arguments avec le label $\mathtt{in}$, le second avec $\mathtt{out}$, et le troisième avec $\mathtt{undec}$.*

Caminada et Pigozzi [7] ont introduit les notions de labellings *down-admissible* et *up-complete*. Ces concepts permettent de retrouver le labelling admissible (resp. complet) le plus proche d'un labelling donné, comme formellement défini par Gabbay et Rodrigues [15], nous renvoyons le lecteur à [15] pour les définitions complètes.

2.2 Agrégation de labellings

Caminada et Pigozzi [7] traitent de la combinaison des points de vue individuels en une décision collective cohérente. Leur approche consiste à agréger un ensemble de labellings, chacun représentant la perception d'un agent sur l'acceptabilité des arguments, afin d'obtenir un labelling collectif. Cette agrégation est réalisée par un opérateur de labellings, défini comme une fonction $LA_{\mathcal{F}} : 2^{\mathcal{L}abellings} \setminus \emptyset \rightarrow \mathcal{L}abellings$, où $\mathcal{L}abellings$ désigne l'ensemble des labellings d'un AF $\mathcal{F}$. Ils proposent trois opérateurs d'agrégation : sceptique, crédule et super-crédule.

2.2.1 L'opérateur sceptique

L'opérateur sceptique exige l'accord unanime de tous les labellings pour qu'un argument soit initialement accepté (in) ou rejeté (out). Tous les autres arguments seront indécis (undec). Une seconde phase consiste à déterminer le labelling *down-admissible* du résultat initial, car celui-ci n'est pas nécessairement admissible.

Définition 1 (*so*)**.** *L'opérateur initial d'agrégation sceptique est une fonction $sio_{\mathcal{F}} : 2^{\mathcal{L}abellings} \setminus \emptyset \rightarrow \mathcal{L}abellings$ telle que, pour un ensemble $\mathcal{L}_1, \ldots, \mathcal{L}_m$ de labellings, soit : $sio_{\mathcal{F}}(\{\mathcal{L}_1, \ldots, \mathcal{L}_m\}) = \{(a, \mathtt{in}) \mid a \in \mathcal{A}r, \forall i \in [1, ..., m], \mathcal{L}_i(a) = \mathtt{in}\} \cup \{(a, \mathtt{out}) \mid a \in \mathcal{A}r, \forall i \in [1, ..., m], \mathcal{L}_i(a) = \mathtt{out}\} \cup \{(a, \mathtt{undec}) \mid a \in \mathcal{A}r, \exists i, \mathcal{L}_i(a) \neq \mathtt{in}$ et $\exists j, \mathcal{L}_j(a) \neq \mathtt{out}\}$.*
L'opérateur d'agrégation sceptique $so_{\mathcal{F}}(\mathcal{L}_1, ..., \mathcal{L}_m)$ est le labelling down-admissible de $sio_{\mathcal{F}}(\mathcal{L}_1, ..., \mathcal{L}_m)$.

Exemple 2. *Considérons le AF $\mathcal{F}$ représenté dans la Figure 1 et l'ensemble de labellings $\mathcal{L}abs = \{\mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_3\}$ tels que : $\mathcal{L}_1 = (\{c, e\}, \{a, b\}, \{d\})$, $\mathcal{L}_2 = (\{e\}, \{a\}, \{b, c, d\})$, $\mathcal{L}_3 = (\{b, c\}, \{a, e\}, \{d\})$. soit alors : $sio_{\mathcal{F}}(\mathcal{L}abs) = (\emptyset, a, \{b, c, d, e\})$. Ce résultat n'est pas un labelling admissible de $\mathcal{F}$, il faut donc déterminer son labelling down-admissible qui est $so_{\mathcal{F}}(\mathcal{L}abs) = (\emptyset, \emptyset, \{a, b, c, d, e\})$, où tous les arguments sont indécis.*

2.2.2 L'opérateur crédule

L'opérateur d'agrégation crédule constitue une version plus permissive de l'opérateur sceptique. L'idée est d'accepter (resp. rejeter) initialement un argument s'il est accepté (resp. rejeté) dans au moins un labelling, à condition qu'aucun labelling ne contredise cette décision. Tous les autres arguments sont laissés indécis (undec). De manière similaire à l'opérateur sceptique, une seconde étape consiste à considérer le labelling *down-admissible* du résultat initial.

Définition 2 (co). *L'opérateur initial d'agrégation crédule est une fonction* $cio_{\mathcal{F}} : 2^{\mathcal{L}abellings} \setminus \emptyset \to \mathcal{L}abellings$ *telle que, pour un ensemble* $\{\mathcal{L}_1, \ldots, \mathcal{L}_m\}$ *de labellings, soit :* $cio_{\mathcal{F}}(\{\mathcal{L}_1, ..., \mathcal{L}_m\}) = \{(a, \texttt{in}) | a \in \mathcal{A}r, \exists i \in [1, ..., m] : \mathcal{L}_i(a) = \texttt{in} \wedge \nexists i \in [1, ..., m] : \mathcal{L}_i(a) = \texttt{out}\} \cup \{(a, \texttt{out}) | a \in \mathcal{A}r, \exists i \in [1, ..., m] : \mathcal{L}_i(a) = \texttt{out} \wedge \nexists i \in [1, ..., m] : \mathcal{L}_i(a) = \texttt{in}\} \cup \{(a, \texttt{undec}) | a \in \mathcal{A}r, \forall i \in [1, ..., m] : \mathcal{L}_i(a) = \texttt{undec} \vee (\exists i \in [1, ..., m] : \mathcal{L}_i(a) = \texttt{in} \wedge \exists i \in [1, ..., m] : \mathcal{L}_i(a) = \texttt{out})\}$. *L'opérateur d'agrégation crédule* $co_{\mathcal{F}}(\{\mathcal{L}_1, \ldots, \mathcal{L}_m\})$ *est le labelling down-admissible de* $cio_{\mathcal{F}}(\{\mathcal{L}_1, \ldots, \mathcal{L}_m\})$.

Exemple 3 (suite). *Considérons l'ensemble de labellings* $\mathcal{L}abs = \{\mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_3\}$ *défini comme suit :* $\mathcal{L}_1 = (\{c, e\}, \{a, b\}, \{d\})$, $\mathcal{L}_2 = (\{e\}, \{a\}, \{b, c, d\})$, $\mathcal{L}_3 = (\{b, c\}, \{a, e\}, \{d\})$. *Nous obtenons :* $cio_{\mathcal{F}}(\mathcal{L}abs) = (\{c\}, \{a\}, \{b, d, e\})$. *Ce résultat est un labelling admissible de* $\mathcal{F}$, *donc* $co_{\mathcal{F}}(\mathcal{L}abs) = cio_{\mathcal{F}}(\mathcal{L}abs)$.

2.2.3 L'opérateur super-crédule

L'opérateur d'agrégation super-crédule étend l'opérateur crédule en appliquant le labelling *up-complet*. Ainsi, les arguments initialement marqués undecpar l'opérateur crédule co peuvent être acceptés (resp. rejetés) si tous leurs attaquants directs sont rejetés (resp. si au moins un attaquant est accepté).

Définition 3 (sco). *L'opérateur d'agrégation super crédule* $sco_{\mathcal{F}}(\mathcal{L}_1, \ldots, \mathcal{L}_m)$ *est défini comme le labelling up-complet de* $co_{\mathcal{F}}(\mathcal{L}_1, \ldots, \mathcal{L}_m)$.

Exemple 4 (suite). *Considérons l'ensemble de labellings* $\mathcal{L}abs = \{\mathcal{L}_1, \mathcal{L}_2, \mathcal{L}_3\}$ *défini par :* $\mathcal{L}_1 = (\{c, e\}, \{b, a\}, \{d\})$, $\mathcal{L}_2 = (\{e\}, \{a\}, \{c, b, d\})$, $\mathcal{L}_3 = (\{b, c\}, \{a, e\}, \{d\})$. *Nous avons* $co_{\mathcal{F}}(\mathcal{L}abs) = (\{c\}, \{a\}, \{b, d, e\})$, *qui est un labelling complet de* $\mathcal{F}$, *donc :* $sco_{\mathcal{F}}(\mathcal{L}abs) = co_{\mathcal{F}}(\mathcal{L}abs)$.

2.3 Systèmes d'argumentation sociale basés sur l'approbation

Bernreiter et al. [5] s'intéressent à la problématique de la combinaison des votes d'approbation avec un AF. Ils introduisent les ABSAFs qui modélisent des débats où des agents expriment leur soutien aux arguments qu'ils jugent convaincants.

Définition 4 (ABSAF). *Un ABSAF* $\mathcal{A}b = (\mathcal{F}, N, \bar{A})$ *est défini par un AF* $\mathcal{F} = \langle \mathcal{A}r, att \rangle$, *un ensemble fini de votants* N, *ainsi qu'un vecteur de bulletins d'approbation* $\bar{A} = (A(i))_{i \in N}$ *où, pour tout* $i \in N$, $A(i) \subseteq \mathcal{A}r$ *est un ensemble non vide d'arguments approuvés par* i.

D'après la définition 4, les votes sont représentés par des ensembles non vides d'arguments approuvés. Le résultat Ω d'un ABSAF $\mathcal{A}b = (\mathcal{F}, N, \bar{A})$ est un ensemble d'extensions au regard d'une sémantique d'extension σ, i.e., $\Omega \subseteq \mathcal{E}_\sigma(\mathcal{F})$. Les votants ne peuvent exprimer qu'une approbation, laissant les autres arguments dans un état ambigu, soit désapprouvés implicitement, soit neutres.

2.3.1 Opérateur de représentation

L'objectif principal de cet opérateur est d'identifier un résultat représentant au mieux la diversité des votants. Cet opérateur quantifie le degré selon lequel une extension représente un votant donné.

Définition 5 (rep). *Soit* $\mathcal{A}b = (\mathcal{F}, N, \bar{A})$ *un ABSAF et* σ *une sémantique d'extension. Le score d'une extension* $\mathcal{E} \in \mathcal{E}_\sigma(\mathcal{F})$ *vis-à-vis d'un votant* $i \in N$ *est défini par* $rep_i(\mathcal{E}) = \frac{|\mathcal{E} \cap A(i)|}{|A(i)|}$. *Le score d'un résultat* $\Omega \subseteq \mathcal{E}_\sigma(\mathcal{F})$ *est donné par le maximum des scores de ses extensions :* $rep_i(\Omega) = \max_{\mathcal{E} \in \Omega} rep_i(\mathcal{E})$.

2.3.2 Opérateur de représentation central

Comme l'ensemble des arguments approuvés par un votant i (c.-à-d. $A(i)$) ne correspond pas nécessairement à une extension existante, Bernreiter et al. proposent une variante de l'opérateur de représentation afin de garantir qu'au moins une extension obtienne le score maximal de 1, même si elle ne correspond pas parfaitement à un $A(i)$.

Définition 6 (rep_c). *Soit* $\mathcal{A}b = (\mathcal{F}, N, \bar{A})$ *un ABSAF et* σ *une sémantique d'extension. Pour* $i \in N$, *soit* $\mu(i) = \max_{\mathcal{E} \in \mathcal{E}_\sigma(\mathcal{F})} |\mathcal{E} \cap A(i)|$. *Si* $\mu(i) = 0$, *le score central d'une extension* $\mathcal{E} \in \mathcal{E}_\sigma(\mathcal{F})$ *est défini par* $rep^c_i(\mathcal{E}) = 1$, *sinon il est donné par* $rep^c_i(\mathcal{E}) = \frac{|\mathcal{E} \cap A(i)|}{\mu(i)}$.

Pour désigner ces deux opérateurs de manière interchangeable, nous utilisons $op \in \{rep, rep^c\}$.

2.3.3 Agrégation des bulletins d'approbation

Bernreiter et al. utilisent une famille de règles basées sur des vecteurs d'agrégation pondérée ordonnée (OWA, pour Ordered Weighted Averaging).

Définition 7 (OWA). *Soit* $\mathcal{A}b = (\mathcal{F}, N, \bar{A})$ *un ABSAF avec* $|N| = n$, σ *une sémantique d'extension et* $k \in \{1, \ldots, n\}$. *Pour un résultat* $\Omega \subseteq \mathcal{E}_\sigma(\mathcal{F})$, *soit* $\vec{s}(\Omega)$ *le vecteur trié dans l'ordre croissant* $\vec{s}(\Omega) = (op_1(\Omega), \ldots, op_n(\Omega))$, *avec* $op \in \{rep, rep^c\}$. *Pour un vecteur de poids décroissant et non négatif* $\vec{w} = (w_1, \ldots, w_n)$, *avec* $w_1 > 0$, *la règle OWA correspondante est définie par :* $OWA^{\vec{w}, op}_\sigma(\mathcal{A}b) \in \text{argmax}_{\Omega \subseteq \mathcal{E}_\sigma(\mathcal{F}) : |\Omega| \leq k} \vec{w} \cdot \vec{s}(\Omega)$.

Les règles égalitaire, utilitaire et harmonique sont notées respectivement e, u, et h. En ajustant le vecteur $\vec{w}$ dans la définition précédente avec, respectivement, $\vec{w}_e = (1, 0, \ldots, 0)$, $\vec{w}_u = (1, \ldots, 1)$, et $\vec{w}_h = (1, 1/2, \ldots, 1/n)$. Une autre méthode consiste à sélectionner l'ensemble des extensions Ω qui maximisent le nombre de votants dont l'ensemble des arguments approuvés obtient un score de 1.

Définition 8 (MaxCov). *Soit* $\mathcal{A}b = (\mathcal{F}, N, \bar{A})$ *un ABSAF avec* $|N| = n$, σ *une sémantique d'extension*

et $k \in \{1, \ldots, n\}$. *Pour un résultat* $\Omega \subseteq \sigma(\mathcal{F})$ *et* $op \in \{rep, rep^c\}$. *La règle de couverture maximale (MaxCov) est définie par :* $MaxCov_\sigma^{op}(\mathcal{A}b) \in \text{argmax}_{\Omega \subseteq \mathcal{E}_\sigma(\mathcal{F}):|\Omega| \leq k} |\{i \in N : op_i(\Omega) = 1\}|$.

La règle MaxCov est notée mc.

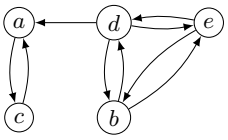
Exemple 5. *Considérons un ABSAF* $\mathcal{A}b = (\mathcal{F}, N, \bar{A})$ *où* $\mathcal{F}$ *est le AF représenté dans la Figure 1, avec les votants* $N = \{1, 2, 3\}$ *et* $\bar{A} = (\{c, e\}, \{e\}, \{b, c\})$. *Rappelons que* $\mathcal{E}_{pr}(\mathcal{F}) = \{\{a, b\}, \{b, c\}, \{c, d\}, \{a, e\}, \{c, e\}\}$. *Par exemple, pour* $\{b, c\}$, *nous obtenons* $\vec{s}(\{b, c\}) = (0, \frac{1}{2}, 1)$ *car* $rep_1(\{b, c\}) = \frac{1}{2}$, $rep_2(\{b, c\}) = 0$, *et* $rep_3(\{b, c\}) = 1$. *Ainsi, en utilisant la règle utilitaire, nous obtenons* $\vec{w}_u \cdot \vec{s}(\{b, c\}) = (1, 1, 1) \cdot (0, \frac{1}{2}, 1) = \frac{3}{2}$. *En appliquant le même raisonnement aux autres extensions, nous obtenons* $OWA_{pr}^{\vec{w}_u, rep}(\mathcal{A}b) = \{\{c, e\}\}$.

Nous clarifions ici plusieurs points relatifs aux approches définies dans [5]. Premièrement, les Définitions 7 et 8 spécifient que k représente le nombre d'extensions à inclure dans l'ensemble final. Dans la Définition 7, pour $k > 1$, tout ensemble d'extensions contenant celle ayant le score maximal obtient ce score (puisque $rep(\Omega)$ utilise la fonction *max*). De même, selon la Définition 8, tout ensemble d'arguments contenant l'extension avec un score de 1 obtient le score maximal. Par conséquent, nous fixons $k = 1$ dans la suite de l'article afin d'évaluer individuellement chaque extension. En cas d'égalité, toutes les extensions ayant le score maximal sont incluses dans le résultat.
Deuxièmement, [5] définit les règles OWA et MaxCov uniquement pour la sémantique préférée. Nous généralisons ces définitions à d'autres sémantiques par extensions, afin d'explorer plus largement leurs propriétés.
Enfin, comme indiqué en Section 4.2, les arguments non mentionnés dans le bulletin d'approbation présentent une ambiguïté (rejet ou abstention). Pour pallier cette ambiguïté, nous adaptons les bulletins d'approbation à notre système de vote.

3 Argumentation basée sur les opinions

En introduisant notre première contribution OBA, notre objectif est de représenter les opinions collectives des votants concernant un ensemble d'arguments. Les systèmes d'OBA (OBAF) servent de structure contenant un AF ainsi qu'un tuple de votes, permettant de représenter diverses perspectives dans un contexte donné. Ce système facilite l'évaluation et l'analyse des opinions collectives exprimées par les votants à travers leurs positions sur les arguments. Dans les OBAF, les votants expriment leurs préférences pour chaque argument en attribuant une valeur : 1 indique l'acceptation, 0 indique l'abstention, et -1 indique le rejet.

Définition 9 (Votes). *Soit* $\mathcal{F} = \langle \mathcal{A}r, att \rangle$ *un AF. Les votes sur* $\mathcal{A}r$, *notés* $\mathcal{V}_{\mathcal{A}r} = \langle v_1, \ldots, v_n \rangle$, *représentent les votes du système. Chaque vote* $v_i \in \mathcal{V}_{\mathcal{A}r}$ *est une fonction* $v_i : \mathcal{A}r \to \{-1, 0, 1\}$ *qui attribue une valeur à chaque argument dans* $\mathcal{F}$, *indiquant la position des votants. Pour tout*



$\mathcal{V}_{\mathcal{A}r}$	a	b	c	d	e
$v1$	1	−1	0	−1	1
$v2$	1	0	0	−1	1
$v3$	1	−1	0	−1	1
$v4$	1	0	0	−1	−1
$v5$	1	1	0	−1	−1
$v6$	−1	0	1	0	1

FIGURE 1 – Un AF basée sur les opinions $\mathcal{O}$

$x \in \mathcal{A}r$, *nous notons* $v^+(x) = \{v_i \in \mathcal{V}_{\mathcal{A}r} \mid v_i(x) = 1\}$, $v^o(x) = \{v_i \in \mathcal{V}_{\mathcal{A}r} \mid v_i(x) = 0\}$ *et* $v^-(x) = \{v_i \in \mathcal{V}_{\mathcal{A}r} \mid v_i(x) = -1\}$ *l'ensemble des votes attribuant respectivement 1, 0 ou -1 à* x.

En pratique, différents types de votes peuvent être envisagés, notamment des échelles plus étendues que l'échelle à trois niveaux (-1, 0, 1). Cependant, nous privilégions cette dernière [16], car dans notre cadre, nous supposons que les votes des participants reflètent leurs croyances. Ainsi, en votant, ils répondent implicitement à la question : "croyez-vous que cet argument est vrai ou non ?". Les croyances, par nature, impliquent des états binaires d'acceptation ou de rejet, à l'image de la logique où une formule est soit déduite, soit non (évaluation binaire). Cela conduit à trois états possibles : la formule est impliquée, sa négation est impliquée ou aucune des deux n'est impliquée. L'échelle à trois niveaux apparaît donc particulièrement adaptée pour représenter ces états de croyance.
Contrairement aux systèmes de préférence, qui permettent des degrés de préférence, les systèmes de croyance sont généralement dichotomiques. Nous considérons néanmoins que l'ajout d'un vote neutre (abstention avec 0) est crucial. Il permet aux votants de ne pas exprimer de croyance, évitant ainsi toute influence indue sur le résultat et garantissant un processus décisionnel plus démocratique. Cependant, notre système OBA peut être adapté à un système de préférence en élargissant l'échelle de vote pour inclure des valeurs additionnelles reflétant divers degrés de préférence. Présentons maintenant formellement les OBAFs.

Définition 10 (OBAF). *Un OBAF est un couple* $\mathcal{O} = \langle \mathcal{F}, \mathcal{V}_{\mathcal{A}r} \rangle$ *où* $\mathcal{F} = \langle \mathcal{A}r, att \rangle$ *est un AF et* $\mathcal{V}_{\mathcal{A}r}$ *sont les votes sur* $\mathcal{A}r$.

Exemple 6. *figure 1 représente un OBAF* $\mathcal{O} = \langle \mathcal{F}, \mathcal{V}_{\mathcal{A}r} \rangle$ *où* $\mathcal{F}$ *est le AF (à gauche) et* $\mathcal{V}_{\mathcal{A}r} = \{v_1, v_2, \ldots, v_6\}$ *(à droite).*

Nous n'imposons qu'une seule contrainte de rationalité pour les votes : un électeur ne peut pas attribuer la valeur positive 1 pour deux arguments impliqués dans une attaque.

Définition 11 (Votes consistency). *Let* $\mathcal{O} = \langle \langle \mathcal{A}r, att \rangle, \mathcal{V}_{\mathcal{A}r} \rangle$ *be an OBAF.* $\mathcal{V}_{\mathcal{A}r}$ *is consistent iff* $\nexists v \in \mathcal{V}_{\mathcal{A}r}$ *s.t.* $v(x) = v(y) = 1$ *and* $(x, y) \in att$.

4 Sémantiques d'opinion collective

Afin d'évaluer les arguments d'un OBAF, nous introduisons une nouvelle famille de sémantiques, COS. Dans cette

section, nous commençons par définir les COS. Ensuite, dans les Sections 4.1 et 4.2, nous adaptons des travaux existants pour les aligner avec les COS, et permettre note étude comparative. Enfin, nous proposons deux nouvelles sémantiques dans les Sections 4.3 et 4.4.

Définition 12 (COS). *Soit $\mathcal{O} = \langle\langle \mathcal{A}r, att\rangle, \mathcal{V}_{\mathcal{A}r}\rangle$ un OBAF. Une sémantique d'opinion collective est une fonction* COS : $\mathcal{O} \to 2^{2^{\mathcal{A}r}}$.

4.1 Approche de Caminada et Pigozzi

L'idée d'adapter l'approche de Caminada et Pigozzi consiste à transformer chaque vote dans $\mathcal{V}_{\mathcal{A}r}$ en un labelling, puis à appliquer l'un des opérateurs définis en Section 2.2 afin d'obtenir le labelling collectif, qui sera ensuite transformé en extension.

Définition 13 (Vote2Lab). *Soit $\mathcal{V}_{\mathcal{A}r}$ l'ensemble des votes sur $\mathcal{A}r$ et $v \in \mathcal{V}_{\mathcal{A}r}$. La fonction $Vote2Lab(v) : \{-1,0,1\}^{\mathcal{A}r} \to \mathcal{L}abellings$ transforme un vote v en son labelling correspondant selon la règle suivante : $Vote2Lab(v) = \{(x, \texttt{in}) \mid v(x) = 1\} \cup \{(x, \texttt{out}) \mid v(x) = -1\} \cup \{(x, \texttt{undec}) \mid v(x) = 0\}$.*

Exemple 7. *Dans la figure 1,soit $Vote2Lab(v_1) = (\{a, e\}, \{b, d\}, \{c\})$ et $Vote2Lab(v_6) = (\{a, b\}, \{d, e\}, \{c\})$.*

Pour obtenir une extension à partir d'un labelling $\mathcal{L}$ dans un AF donné $\mathcal{F}$, on utilise la fonction $Lab2Ext_{\mathcal{F}}(\mathcal{L}) = \{a \in \mathcal{A}r \mid (a, \texttt{in}) \in \mathcal{L}\}$ [6].

Définition 14 ($\texttt{COS}^{LA}$). *Soit $\mathcal{O} = \langle\mathcal{F}, \mathcal{V}_{\mathcal{A}r}\rangle$ un OBAF avec $\mathcal{V}_{\mathcal{A}r} = \langle v_1, \ldots, v_n\rangle$. La* COS *basée sur l'opérateur d'agrégation d'labellings $LA \in \{so, co, sco\}$ est définie par :*

$$\texttt{COS}^{LA}(\mathcal{O}) = \{Lab2Ext_{\mathcal{F}}(\ell) \mid \ell \in LA_{\mathcal{F}}(\{Vote2Lab(v_1), \ldots, Vote2Lab(v_n)\})\}$$

Exemple 7 (suite). *On obtient* $\texttt{COS}^{so}(\mathcal{O}) = \{\emptyset\}$, $\texttt{COS}^{co}(\mathcal{O}) = \{\{c\}\}$, $\texttt{COS}^{sco}(\mathcal{O}) = \{\{c\}\}$.

4.2 Approche de Bernreiter et al.

Dans un ABSAF, les électeurs ne peuvent indiquer que leur approbation, laissant le statut des autres arguments incertain. Pour pallier cette ambiguïté et permettre une comparaison équitable, nous introduisons une définition permettant de convertir nos votes en bulletins d'approbation. La méthode la plus simple consiste à ne considérer que les votes positifs dans le bulletin d'approbation.

Définition 15 (Vote2Bal). *Soit $\mathcal{V}_{\mathcal{A}r}$ l'ensemble des votes sur $\mathcal{A}r$ et $v \in \mathcal{V}_{\mathcal{A}r}$. La fonction $Vote2Bal(v) : \{-1,0,1\}^{\mathcal{A}r} \to 2^{\mathcal{A}r}$ transforme un vote v en son bulletin d'approbation correspondant selon la règle : $Vote2Bal(v) = \{x \mid v(x) = 1\}$.*

Cette transformation facilite la conversion d'un OBAF en ABSAF, permettant ainsi l'adaptation de l'opérateur de Bernreiter et al. [5] dans notre cadre COS.

Définition 16 ($\texttt{COS}_{\sigma}^{AB,op}$). *Soit $\mathcal{O} = \langle\mathcal{F}, \mathcal{V}_{\mathcal{A}r}\rangle$ un OBAF avec $\mathcal{V}_{\mathcal{A}r} = \langle v_1, \ldots, v_n\rangle$ et σ une sémantique à extensions. Un ABSAF est défini comme $\mathcal{A}b = (\mathcal{F}, \{1, \ldots, n\}, \langle Vote2Bal(v) \mid v \in \mathcal{V}_{\mathcal{A}r}\rangle)$. La* COS *basée sur les opérateurs de représentation $op \in \{rep, rep_c\}$ est :*

$$\texttt{COS}_{\sigma}^{AB,op}(\mathcal{O}) = \begin{cases} OWA_{\sigma}^{\vec{w},op}(\mathcal{A}b) & \text{pour } AB \in \{u, e, h\} \\ MaxCov_{\sigma}^{op}(\mathcal{A}b) & \text{pour } AB \in \{mc\} \end{cases}$$

Exemple 8. *On obtient $\texttt{COS}_{pr}^{u,op}(\mathcal{O}) = \{\{c, e\}\}$ pour $op \in \{rep, rep_c\}$.*

4.3 Sémantiques basées sur la suppression d'attaques

Nous définissons une nouvelle approche combinant une agrégation de votes inspirée de [19] et la suppression d'attaques utilisée dans les AFs basés sur les préférences [3]. L'idée est de donner une forte priorité aux arguments les « plus acceptés » au regard des votes. La méthode associe à chaque argument une valeur calculée à partir de ses votes, puis supprime les attaques dans l'AF lorsque l'argument attaqué a une valeur supérieure à celle de l'argument attaquant. Cette modification radicale de l'AF présente un avantage pratique, notamment dans le cadre des débats en ligne, où cette méthode de suppression d'attaques est particulièrement pertinente. Par exemple, si certains arguments sont largement perçus comme incorrects ou perturbateurs, et que la majorité des utilisateurs a voté contre eux, supprimer les attaques associées à ces arguments atténue leur impact sur le résultat final sans procéder à une modification encore plus radicale de l'AF (c.-à-d. la suppression de l'argument lui-même). Cette solution peut considérablement améliorer la qualité et la cohérence du débat.
L'évaluation de ce score pour un argument est effectuée par une fonction générale d'agrégation d'opinions basée sur le nombre de votes à 1, 0, et -1 pour cet argument.

Définition 17 (Fonction d'agrégation d'opinions). *Soit $\mathcal{O} = \langle\langle \mathcal{A}r, att\rangle, \mathcal{V}_{\mathcal{A}r}\rangle$ un OBAF. Une fonction d'agrégation d'opinions $\tau : \mathbb{N} \times \mathbb{N} \times \mathbb{N} \to \mathbb{R}$ produit un score pour un argument à partir de ses votes. Par abus de notation, pour tout $x \in \mathcal{A}r$, on note $\tau(x) = \tau(|v^+(x)|, |v^o(x)|, |v^-(x)|)$.*

Il existe de nombreuses variantes possibles de fonctions d'agrégation d'opinions. Nous utilisons la fonction proposée par Leite et Martins [19] en raison de sa simplicité. Cette fonction peut être facilement ajustée pour s'adapter à d'autres scénarios.

Définition 18 (τ_ϵ). *Soit $\mathcal{O} = \langle\langle \mathcal{A}r, att\rangle, \mathcal{V}_{\mathcal{A}r}\rangle$ un OBAF et $x \in \mathcal{A}r$. La fonction τ_ϵ est une fonction d'agrégation d'opinions, où $\epsilon \geq 0$,*

$$\tau_\epsilon(x) = \begin{cases} 0 & \text{si } |v^+(x)| = |v^-(x)| = 0 \\ \frac{|v^+(x)|}{|v^+(x)|+|v^-(x)|+\epsilon} & \text{sinon} \end{cases}$$

Notre approche pour éliminer certaines attaques repose sur les AFs basés sur les préférences, tels que proposés par Amgoud et Cayrol [3].

Définition 19 ($\mathcal{O}_\tau$). *Soit $\mathcal{O} = \langle\langle \mathcal{A}r, att\rangle, \mathcal{V}_{\mathcal{A}r}\rangle$ un OBAF et τ une fonction d'agrégation d'opinions.*
Un $\mathcal{O}_\tau$ est un triplet $\langle\langle \mathcal{A}r, att^*\rangle, \mathcal{V}_{\mathcal{A}r}, \succeq^\tau_{\mathcal{O}}\rangle$ où :
- *$att^* = \{(x, y) \mid (x, y) \in att$ et $x \succeq^\tau_{\mathcal{O}} y\}$;*
- *$\succeq^\tau_{\mathcal{O}}$ est un préordre total sur $\mathcal{A}r$ tel que $x \succeq^\tau_{\mathcal{O}} y$ si et seulement si $\tau(x) \geq \tau(y)$.*

Nous notons $\mathcal{F}_\tau = \langle \mathcal{A}r, att^*\rangle$ le AF associé à $\mathcal{O}_\tau$.

Il est désormais possible de définir un COS construisant $\mathcal{O}_\tau$ à partir d'un OBAF et retournant les ensembles d'arguments obtenus en appliquant une sémantique basée sur des extensions à $\mathcal{F}_\tau$.

Définition 20 ($\texttt{COS}^{\texttt{AR}}$). *Soit $\mathcal{O} = \langle \mathcal{F}, \mathcal{V}_{\mathcal{A}r}\rangle$ un OBAF. Soit σ une sémantique basée sur des extensions et τ une fonction d'agrégation d'opinions. Le COS fondé sur la suppression d'attaques est défini par $\texttt{COS}^{\texttt{AR}}_{\sigma,\tau}(\mathcal{O}) = \mathcal{E}_\sigma(\mathcal{F}_\tau)$.*

Exemple 9. *Appliquons tout d'abord la fonction d'agrégation d'opinions τ_ϵ à chaque argument avec $\epsilon = 0.1$. Nous obtenons $\tau_\epsilon(a) = 0.82$, $\tau_\epsilon(b) = 0.32$, $\tau_\epsilon(c) = 0.91$, $\tau_\epsilon(d) = 0$ et $\tau_\epsilon(e) = 0.66$. Cela nous donne le préordre total suivant : $c \succeq^{\tau_\epsilon}_{\mathcal{O}} a \succeq^{\tau_\epsilon}_{\mathcal{O}} e \succeq^{\tau_\epsilon}_{\mathcal{O}} b \succeq^{\tau_\epsilon}_{\mathcal{O}} d$. On peut alors définir $\mathcal{O}_\tau = \langle\langle \mathcal{A}r, att^*\rangle, \mathcal{V}_{\mathcal{A}r}, \succeq^{\tau_\epsilon}_{\mathcal{O}}\rangle$ avec $att^* = \{(c, a), (b, d), (e, b), (e, d)\}$. Par conséquent, on obtient $\texttt{COS}^{\texttt{AR}}_{pr,\tau_\epsilon}(\mathcal{O}) = \{\{c, e\}\}$.*

4.4 Sémantique de satisfaction collective

Dans cette section, nous présentons notre deuxième contribution, qui définit des mesures de satisfaction, d'insatisfaction et d'utilité permettant d'agréger les opinions des votants.
Pour faciliter la comparaison entre les votes et les extensions, on peut représenter une extension par un vecteur où les arguments qu'elle contient sont assignés à 1, tandis que les autres prennent la valeur -1.

Définition 21 (Vec). *Soit $\mathcal{F} = \langle \mathcal{A}r, att\rangle$ un AF. Soit σ une sémantique basée sur des extensions. Pour une extension donnée $\mathcal{E} \in \mathcal{E}_\sigma(\mathcal{F})$, la fonction $Vec_\mathcal{E} : \mathcal{A}r \to \{-1, 1\}$ est définie par : $\forall x \in \mathcal{A}r$,*

$$Vec_\mathcal{E}(x) = \begin{cases} 1 & \text{si } x \in \mathcal{E} \\ -1 & \text{sinon} \end{cases}$$

Par abus de notation, $Vec(\mathcal{E})$ représente le (vecteur de) votes correspondant à l'extension $\mathcal{E}$.

Définissons formellement les trois mesures utilisées pour comparer un vote et une extension. La mesure de satisfaction correspond au nombre d'arguments pour lesquels la fonction Vec et le vote attribuent la même valeur. À l'inverse, la mesure d'insatisfaction compte les arguments dont les valeurs diffèrent. Enfin, la mesure d'utilité est la somme des deux mesures précédentes.

	v_1	v_2	v_3	v_4	v_5	v_6	$d^\Sigma_{\mathcal{V}_{\mathcal{A}r}}(\{a, b\})$
$\mathcal{S}_v(\{a, b\})$	2	2	2	0	3	4	13
$\mathcal{D}_v(\{a, b\})$	-2	-1	-2	-3	0	0	-8
$\mathcal{U}_v(\{a, b\})$	0	1	0	-3	3	4	5

TABLE 1 – Valeurs des mesures de satisfaction, d'insatisfaction et d'utilité entre l'extension préférée $\{a, b\}$ et tous les votes de $\mathcal{V}_{\mathcal{A}r}$ dans le OBAF représenté dans la Figure 1.

Définition 22 ((In)satisfaction, utilité). *Soit $\mathcal{O} = \langle \mathcal{F}, \mathcal{V}_{\mathcal{A}r}\rangle$ un OBAF avec $\mathcal{F} = \langle \mathcal{A}r, att\rangle$. Soit σ une sémantique basée sur des extensions. Pour une extension donnée $\mathcal{E} \in \mathcal{E}_\sigma(\mathcal{F})$, la satisfaction $\mathcal{S}_v$, l'insatisfaction $\mathcal{D}_v$ et l'utilité $\mathcal{U}_v$ de $\mathcal{E}$ par rapport à un vote $v \in \mathcal{V}_{\mathcal{A}r}$ sont définies comme suit :*
- *$\mathcal{S}_v(\mathcal{E}) = |\{x \in \mathcal{A}r \mid v(x) = Vec_\mathcal{E}(x)\}|$*
- *$\mathcal{D}_v(\mathcal{E}) = -|\{x \in \mathcal{A}r \mid v(x) = -Vec_\mathcal{E}(x)\}|$*
- *$\mathcal{U}_v(\mathcal{E}) = \mathcal{S}_v(\mathcal{E}) + \mathcal{D}_v(\mathcal{E})$*

Bien que $\mathcal{D}$ et $\mathcal{S}$ puissent sembler complémentaires, elles mesurent en réalité des aspects distincts de la correspondance d'un vote avec une extension.
Nous cherchons à évaluer la distance entre une extension et l'ensemble des votes $\mathcal{V}_{\mathcal{A}r}$, afin de mesurer la proximité de l'extension par rapport au profil de votes. Étant donné que nos fonctions d'agrégation sont la somme, le minimum et le leximin [2], il est souhaitable de maximiser cette distance.

Définition 23 (Distance). *Soit $\mathcal{O} = \langle \mathcal{F}, \mathcal{V}_{\mathcal{A}r}\rangle$ un OBAF avec $\mathcal{F} = \langle \mathcal{A}r, att\rangle$. Soit σ une sémantique basée sur des extensions. Pour une extension donnée $\mathcal{E} \in \mathcal{E}_\sigma(\mathcal{F})$, la distance de $\mathcal{E}$ par rapport à $\mathcal{V}_{\mathcal{A}r}$ est définie comme suit : $d^\otimes_{\mathcal{V}_{\mathcal{A}r}}(\mathcal{E}) = \otimes_{v \in \mathcal{V}_{\mathcal{A}r}} \mathcal{M}_v(\mathcal{E})$ avec $\otimes \in \{\Sigma, \min, leximin\}$ et $\mathcal{M} \in \{\mathcal{D}, \mathcal{S}, \mathcal{U}\}$.*

Ainsi, les extensions maximisant la distance par rapport à l'ensemble des votes seront considérées comme le résultat.

Définition 24 ($\texttt{CSS}$). *Soit $\mathcal{O} = \langle \mathcal{F}, \mathcal{V}_{\mathcal{A}r}\rangle$ un OBAF avec $\mathcal{F} = \langle \mathcal{A}r, att\rangle$. Soit σ une sémantique basée sur des extensions, $\otimes \in \{\Sigma, \min, leximin\}$ et $\mathcal{M} \in \{\mathcal{D}, \mathcal{S}, \mathcal{U}\}$. La sémantique de satisfaction collective est définie par*

$$\texttt{CSS}^{\mathcal{M},\otimes}_\sigma(\mathcal{O}) = \operatorname*{argmax}_{\mathcal{E} \in \mathcal{E}_\sigma(\mathcal{F})}(d^\otimes_{\mathcal{V}_{\mathcal{A}r}}(\mathcal{E}))$$

Exemple 10. *Voici les calculs pour obtenir $\texttt{CSS}^{\mathcal{U},\Sigma}_{pr}(\mathcal{O})$. Pour rappel, nous avons $\mathcal{E}_{pr}(\mathcal{F}) = \{\{a, b\}, \{b, c\}, \{c, d\}, \{a, e\}, \{c, e\}\}$. Tout d'abord, il est nécessaire de convertir chacune de ces extensions en un vecteur de votes. Par exemple, $Vec(\{a, b\}) = \langle 1, 1, -1, -1, -1\rangle$. Ensuite, nous calculons la mesure d'utilité pour chaque couple (extension, vote). Le tableau 1 présente un exemple avec $\{a, b\}$. En suivant le même raisonnement pour $\mathcal{U}_v$, nous obtenons $d^\Sigma_{\mathcal{V}_{\mathcal{A}r}}(\{b, c\}) = -1$, $d^\Sigma_{\mathcal{V}_{\mathcal{A}r}}(\{c, d\}) = -9$, $d^\Sigma_{\mathcal{V}_{\mathcal{A}r}}(\{a, e\}) = 11$, et $d^\Sigma_{\mathcal{V}_{\mathcal{A}r}}(\{c, e\}) = 5$. Enfin,*

2. Appliquée à un vecteur de n nombres réels, la fonction leximin retourne la liste triée par ordre croissant, comparée ensuite selon l'ordre lexicographique induit par l'ordre standard sur les réels.

en appliquant la formule donnée dans la Définition 24, nous obtenons $\texttt{CSS}_{pr}^{\mathcal{U},\Sigma}(\mathcal{O}) = \{\{a,e\}\}$. *En utilisant les deux autres fonctions d'agrégation, nous obtenons* $\texttt{CSS}_{pr}^{\mathcal{U},min}(\mathcal{O}) = \{\{a,e\},\{c,e\}\}$ *et* $\texttt{CSS}_{pr}^{\mathcal{U},leximin}(\mathcal{O}) = \{\{a,e\}\}$.

4.5 Observations

Commençons par examiner les travaux connexes avec figure 1. produit un ensemble vide en raison de l'absence d'unanimité (positive ou négative) sur un argument donné. Pour $\texttt{COS}^{co}$ et $\texttt{COS}^{sco}$, le résultat est également l'ensemble vide. En effet, avec cio, nous avons $\texttt{in} = \{c\}$ et $\texttt{out} = \{d\}$. En appliquant l'admissibilité descendante sur cet labelling, nous obtenons l'ensemble vide pour $\texttt{in}$ et $\texttt{out}$, car bien que $\{c\}$ soit une extension admissible, le labelling n'appartient pas à l'ensemble des labellings admissibles : $\texttt{out}(\{\{c\},\{d\},\{a,b,e\}\}) \not\subseteq \texttt{out}(\{\{c\},\{a\},\{b,d,e\}\})$. Cela constitue un inconvénient notable de la méthode, cette limitation est particulièrement problématique dans des contextes comme l'opinion publique, où l'unanimité est rarement atteinte. Une sémantique efficace devrait pouvoir proposer des solutions réalisables dans de tels cas.

Sous la sémantique ABSAF, nous obtenons $\texttt{COS}_{\texttt{pr}}^{AB,op}(\mathcal{O}) = \{\{a,e\}\}$, où $AB \in \{u,e,h,mc\}$ et $op \in \{rep, rep^c\}$, représentant l'extension souhaitable. Cependant, en utilisant d'autres sémantiques, telles que la sémantique complète, cette méthode ne donne pas toujours la meilleure solution. Cette question est abordée plus en détail dans la Section 6.

En considérant $\texttt{COS}^{\texttt{AR}}$, c est l'argument ayant le score le plus élevé, ce qui conduit à la suppression de l'attaque (a,c). De même, e est considéré comme meilleur que b et d par rapport à $\succeq_{\mathcal{O}}^{\tau_\epsilon}$, il devient non attaqué. Ainsi, nous obtenons $\texttt{COS}_{\texttt{pr},\tau_\epsilon}^{\texttt{AR}}(\mathcal{O}) = \{\{c,e\}\}$.

Enfin, considérons notre CSS. L'extension souhaitée $\{a,e\}$ atteint le score maximal pour $\mathcal{S}$, $\mathcal{D}$ et $\mathcal{U}$ pour $\otimes \in \{\Sigma, leximin\}$, soit $\texttt{CSS}_{\texttt{pr}}^{\mathcal{M},\otimes}(\mathcal{O}) = \{\{a,e\}\}$. Cependant, des différences apparaissent dans cet exemple lorsque $\otimes = min$. En effet, seule la mesure de satisfaction $\mathcal{S}$ donne une véritable solution égalitaire, ce qui conduit à $\texttt{CSS}_{\texttt{pr}}^{\mathcal{S},min}(\mathcal{O}) = \texttt{CSS}_{\texttt{pr}}^{\mathcal{M},\{\Sigma,leximin\}}(\mathcal{O}) \cup \{\{b,c\},\{e,c\}\}$, car trois des six votants (à savoir v_4, v_5 et v_6) ne seraient pas entièrement satisfaits par $\{a,e\}$. En revanche, la dissatisfaction $\mathcal{D}$ diverge de ce résultat, et l'utilité $\mathcal{U}$ s'aligne avec la dissatisfaction $\mathcal{D}$, car la dissatisfaction globale dans le système l'emporte sur la satisfaction. Nous obtenons alors $\texttt{CSS}_{\texttt{pr}}^{\{\mathcal{D},\mathcal{U}\},min}(\mathcal{O}) = \{\{a,e\}\}$. Cet exemple illustre comment nos opérateurs et méthodes d'agrégation permettent d'obtenir des résultats finaux nuancés.

Il est à noter que tous les votants, sauf un, ont voté pour l'argument a et contre l'argument d, l'argument e ayant également reçu quatre votes. Par conséquent, $\{a,e\}$ semble être l'extension la plus souhaitable dans ce AF par rapport aux votes. En conséquence, notre méthode ainsi que l'ABSAF sont les seules approches qui parviennent à ce que l'on peut considérer comme le résultat le plus raisonnable.

5 Propriétés des sémantiques collectives d'opinion

Cette section vise à identifier les propriétés axiomatiques aux $\texttt{COS}$. Nous adaptons des propriétés classiques issues de domaines connexes au contexte de l'agrégation d'opinions. Nous montrons que notre approche satisfait certaines de ces propriétés et discutons pourquoi d'autres ne sont pas vérifiées, ce qui sera approfondi dans la section suivante pour souligner les atouts de notre système. Ces propriétés constituent la base d'une étude axiomatique comparative entre notre méthode et les approches existantes. Nous classons ces propriétés en deux catégories : essentielles et additionnelles. Cette distinction permet de mettre en avant les critères fondamentaux que chaque méthode doit vérifier, tandis que les propriétés additionnelles apportent un complément d'analyse, renforçant la compréhension et la robustesse de $\texttt{COS}$ dans des contextes spécifiques.

5.1 Propriétés essentielles

Le principe d'anonymat des votants, inspiré de [13], stipule que la sémantique collective d'opinion (COS) doit être insensible à l'identité des votants.

Axiome 1 (Anonymat des votes (VA)). *Soit* $\Pi(\mathcal{V}_{\mathcal{A}r})$ *l'ensemble des permutations des votes dans* $\mathcal{V}_{\mathcal{A}r}$ *d'un OBAF* $\mathcal{O} = \langle \mathcal{F}, \mathcal{V}_{\mathcal{A}r}\rangle$. *Une COS satisfait l'anonymat des votes si et seulement si, pour tout* $\mathcal{V}'_{\mathcal{A}r} \in \Pi(\mathcal{V}_{\mathcal{A}r})$, *nous obtenons* $\texttt{COS}(\mathcal{O}) = \texttt{COS}(\mathcal{O}')$ *avec* $\mathcal{O}' = \langle \mathcal{F}, \mathcal{V}'_{\mathcal{A}r}\rangle$.

La neutralité, inspirée de [2], requiert qu'une COS ne dépende pas des noms des arguments ni de la façon dont les votes leur sont attribués.

Définition 25 (Isomorphisme). *Soient* $\mathcal{O} = \langle\langle \mathcal{A}r, att\rangle, \mathcal{V}_{\mathcal{A}r}\rangle$ *et* $\mathcal{O}' = \langle\langle \mathcal{A}r', att'\rangle, \mathcal{V}'_{\mathcal{A}r'}\rangle$ *deux OBAFs. Un isomorphisme de* $\mathcal{O}$ *vers* $\mathcal{O}'$ *est une bijection* $f : \mathcal{A}r \to \mathcal{A}r'$ *telle que, pour tous* $a,b \in \mathcal{A}r$, $(a,b) \in att$ *si et seulement si* $(f(a),f(b)) \in att'$, *et une bijection correspondante entre les ensembles de votes* $\mathcal{V}_{\mathcal{A}r}$ *et* $\mathcal{V}'_{\mathcal{A}r'}$ *telle que pour tout* $a \in \mathcal{A}r$ *et tout vote* $v_i \in \mathcal{V}_{\mathcal{A}r}$, $v_i(a) = v'_i(f(a))$.

Axiome 2 (Neutralité (N)). *Une COS satisfait la neutralité si et seulement si, pour tous OBAFs* $\mathcal{O} = \langle \mathcal{F}, \mathcal{V}\rangle$ *et* $\mathcal{O}' = \langle \mathcal{F}', \mathcal{V}'\rangle$, *et pour tout isomorphisme* f *de* $\mathcal{O}$ *vers* $\mathcal{O}'$, *nous obtenons :* $\texttt{COS}(\mathcal{O}) = \texttt{COS}(f(\mathcal{O}))$.

La monotonie, inspiré de [2], stipule que l'ajout d'un vote correspondant à un ensemble d'arguments déjà sélectionné par la COS ne doit pas retirer cet ensemble du résultat.

Axiome 3 (Monotonie (M)). *Une COS satisfait la monotonie si, pour tout OBAF* $\mathcal{O} = \langle\langle \mathcal{A}r, att\rangle, \mathcal{V}\rangle$, *pour tout ensemble* $\mathcal{E} \in \texttt{COS}(\mathcal{O})$, *on a :* $\mathcal{E} \subseteq \texttt{COS}(\langle\langle \mathcal{A}r, att\rangle, \mathcal{V} \cup \{v\}\rangle)$, *où* $v = Vec(\mathcal{E})$ *est le vote induit par l'extension* $\mathcal{E}$.

La propriété de non-trivialité, inspirée de [13], exige que le résultat d'une COS contienne au moins un ensemble d'arguments non vide dès lors qu'il existe au moins une extension non vide selon une sémantique d'extensions. $\mathbb{F}_{NT,\sigma}$ est l'ensemble des OBAFs non triviaux pour σ.

Axiome 4 (Non-trivialité (NT)). *Soit σ une sémantique basée sur les extensions. Une COS satisfait la non-trivialité si, pour tout $\mathcal{O} = \langle \mathcal{F}, \mathcal{V}_{\mathcal{A}r} \rangle$ tel que $|\mathcal{E}_\sigma(\mathcal{F})| \geq 1$ et $\emptyset \notin \mathcal{E}_\sigma(\mathcal{F})$, alors $|\mathtt{COS}(\mathcal{O})| \geq 1 \quad et \quad \mathtt{COS}(\mathcal{O}) \neq \{\emptyset\}$.*

La propriété d'unanimité sur l'extension, également inspirée de [13], stipule que si tous les votes correspondent à la même extension, alors cette extension doit être retournée par la COS.

Axiome 5 (Unanimité sur l'extension (EU)). *Soit σ une sémantique basée sur les extensions. Une COS satisfait l'unanimité sur l'extension si, pour tout $\mathcal{O} = \langle \mathcal{F}, \mathcal{V}_{\mathcal{A}r} \rangle$ et toute extension $\mathcal{E} \in \mathcal{E}_\sigma(\mathcal{F})$ vérifiant $\forall v \in \mathcal{V}_{\mathcal{A}r}, \quad v = Vec(\mathcal{E})$, on a $\mathtt{COS}(\mathcal{O}) = \{\mathcal{E}\}$.*

La propriété de non-conflictualité garantit la cohérence des résultats vis-à-vis du AF sous-jacent. Cette propriété est adaptée de [17] pour tenir compte des votes.

Axiome 6 (Non-conflictualité (CF)). *Une COS satisfait la non-conflictualité si, pour tout $\mathcal{O} = \langle \mathcal{F}, \mathcal{V}_{\mathcal{A}r} \rangle$, et pour tout ensemble $E \in \mathtt{COS}(\mathcal{O})$, E est conflict-free par rapport à $\mathcal{F}$.*

5.2 Propriétés supplémentaires

Unanimité argumentaire inspirée de [13], cette propriété stipule que si tous les votes acceptent unanimement un argument, alors cet argument doit appartenir à tous les ensembles d'arguments du résultat.

Axiome 7 (Unanimité argumentaire (UA)). *Un COS satisfait l'Unanimité argumentaire si, pour tout $\mathcal{O} = \langle \mathcal{F}, \mathcal{V}_{\mathcal{A}r} \rangle$ tel qu'il existe un argument $x \in \mathcal{A}r$ avec $v(x) = 1$ pour tout $v \in \mathcal{V}_{\mathcal{A}r}$, il en résulte que pour tout $E \in \mathtt{COS}(\mathcal{O})$, $x \in E$.*

Majorité d'extension inspirée de [13], cette propriété exige que si une majorité stricte des votes correspond à une même extension, alors cette extension doit figurer dans le résultat du COS.

Axiome 8 (Majorité d'extension (ME)). *Soit σ une sémantique basée sur les extensions. Un COS satisfait la Majorité d'extension si, pour tout $\mathcal{O} = \langle \mathcal{F}, \mathcal{V}_{\mathcal{A}r} \rangle$ tel qu'il existe une extension $\mathcal{E} \in \mathcal{E}_\sigma(\mathcal{F})$ avec $|\{v \in \mathcal{V}_{\mathcal{A}r} \mid v = Vec(\mathcal{E})\}| > \frac{|\mathcal{V}_{\mathcal{A}r}|}{2}$, alors $\mathtt{COS}(\mathcal{O}) = \{\mathcal{E}\}$.*

Séparabilité inspirée de [10], la séparabilité affirme que si deux ensembles de votes conduisent à des ensembles d'extensions distincts, mais que ces ensembles se recoupent, alors les extensions communes doivent être incluses dans le résultat final.

Axiome 9 (Séparabilité (S)). *Un COS satisfait la propriété de Séparabilité si, pour tout triplet d'OBAFs $\mathcal{O}_1 = \langle \mathcal{F}, \mathcal{V}^1_{\mathcal{A}r} \rangle, \quad \mathcal{O}_2 = \langle \mathcal{F}, \mathcal{V}^2_{\mathcal{A}r} \rangle, \quad \mathcal{O}_3 = \langle \mathcal{F}, \mathcal{V}^1_{\mathcal{A}r} \cup \mathcal{V}^2_{\mathcal{A}r} \rangle$, on a : $\mathtt{COS}(\mathcal{O}_1) \cap \mathtt{COS}(\mathcal{O}_2) \neq \emptyset \implies \mathtt{COS}(\mathcal{O}_1) \cap \mathtt{COS}(\mathcal{O}_2) \subseteq \mathtt{COS}(\mathcal{O}_3)$.*

Continuité, inspirée de [13], la continuité exprime la possibilité d'atteindre exactement une extension donnée en ajoutant un certain nombre de fois le même vote représentant cette extension. Pour cela, nous utilisons la notation $\langle x \rangle^k = \underbrace{\langle x, x, \ldots, x \rangle}_{k \text{ fois}}$ désignant un vecteur de taille k composé de x répété.

Axiome 10 (Continuité (Cn)). *Soit σ une sémantique basée sur les extensions. Un COS satisfait la Continuité si, pour tout $\mathcal{O} = \langle \mathcal{F}, \mathcal{V}_{\mathcal{A}r} \rangle$, tel qu'une extension $\mathcal{E} \in \mathcal{E}_\sigma(\mathcal{F})$ existe et que $\mathcal{E} \notin \mathtt{COS}(\mathcal{O})$, il existe un entier $k \in \mathbb{N}$ tel que $\mathtt{COS}(\langle \mathcal{F}, \mathcal{V}_{\mathcal{A}r} \cup \langle Vec(\mathcal{E}) \rangle^k \rangle) = \{\mathcal{E}\}$.*

6 Résultats et Discussion

Dans la Section 4.5, nous avons montré que, contrairement aux techniques existantes dans la littérature, notre CSS traite efficacement les problèmes d'insatisfaction tout en améliorant la satisfaction globale des votants. De plus, nos sémantiques présentent une meilleure capacité de décision comparativement aux autres approches, ce qui nous conduit à étudier les propriétés essentielles pour une agrégation d'opinions efficace. Dans ce cadre, nous proposons d'établir quelles propriétés, introduites en Section 5, sont satisfaites par les COS présentés en Section 4.
Tout au long de cette étude, nous considérons trois sémantiques d'extensions bien établies : les sémantiques préférée, complète et stable. Ces choix nous offrent un contrôle rigoureux sur les extensions produites, tout en répondant à des besoins et scénarios variés.

Proposition 1. *Soit $LA \in \{so, co, sco\}$. Le COS $\mathtt{COS}^{LA}$ satisfait les propriétés VA, N, M, EU, CF et S.*

Proposition 2. *Soit $\sigma \in \{co, pr, stb\}$. Le COS $\mathtt{COS}^{\mathtt{AR}}_{\sigma,\tau_\epsilon}$ satisfait les propriétés VA, N, M, NT, EU, S, AU et Cn.*

Proposition 3. *Soit $\mathcal{M} \in \{\mathcal{D}, \mathcal{S}, \mathcal{U}\}$ et $\sigma \in \{co, pr, stb\}$. Le COS $\mathtt{CSS}^{\mathcal{M},\Sigma}_\sigma$ satisfait les propriétés VA, N, M, NT, EU, CF, S, EM et Cn.*

Proposition 4. *Soit $\mathcal{M} \in \{\mathcal{D}, \mathcal{S}, \mathcal{U}\}$ et $\sigma \in \{co, pr, stb\}$. Les COS $\mathtt{CSS}^{\mathcal{M},min}_\sigma$ et $\mathtt{CSS}^{\mathcal{M},leximin}_\sigma$ satisfont les propriétés VA, N, M, NT, EU, CF et S.*

Proposition 5. *Soit $AB \in \{u, e, h, mc\}$ et $\sigma \in \{co, pr, stb\}$. Les COS $\mathtt{COS}^{AB,rep^c}_\sigma$ et $\mathtt{COS}^{AB,rep}_{co}$ satisfont les propriétés VA, N, M, NT, CF et S.*

Proposition 6. *Soit $AB \in \{u, h\}$ et $\sigma \in \{pr, stb\}$. Le COS $\mathtt{COS}^{AB,rep}_\sigma$ satisfait les propriétés VA, N, M, NT, EU, CF, S et Cn.*

Proposition 7. *Soit $\sigma \in \{pr, stb\}$. Le COS $\mathtt{COS}^{e,rep}_\sigma$ satisfait les propriétés VA, N, M, NT, EU, CF et S.*

Proposition 8. *Soit $\sigma \in \{pr, stb\}$. Le COS $\mathtt{COS}^{mc,rep}_\sigma$ satisfait les propriétés VA, N, M, NT, EU, CF, S, EM et Cn.*

Plusieurs observations peuvent être formulées à partir des résultats présentés dans la tableau 2.
Tout d'abord, le CSS se distingue comme l'une des rares approches (avec $\mathtt{COS}^{AB,rep}_\sigma$ pour $\sigma \in \mathtt{pr, stb}$) à satisfaire l'ensemble des propriétés essentielles, en particulier CF, NT et EU. Ces propriétés sont fondamentales : la Non-trivialité (NT) garantit la décidabilité, c'est-à-dire que le

	Propriétés essentielles						Propriétés supplémentaires			
Sémantique	VA	N	M	NT	EU	CF	S	AU	EM	Cn
$\texttt{COS}^{so}$	✓	✓	✓	×	✓	✓	✓	×	×	×
$\texttt{COS}^{co}$	✓	✓	✓	×	✓	✓	✓	×	×	×
$\texttt{COS}^{sco}$	✓	✓	✓	×	✓	✓	✓	×	×	×
$\texttt{COS}^{\{u,h\},rep}$	✓	✓	✓	✓	$\times^{\texttt{co}}$	✓	✓	×	×	$\times^{\texttt{co}}$
$\texttt{COS}^{e,rep}$	✓	✓	✓	✓	$\times^{\texttt{co}}$	✓	✓	×	×	×
$\texttt{COS}^{mc,rep}$	✓	✓	✓	✓	$\times^{\texttt{co}}$	✓	✓	×	$\times^{\texttt{co}}$	$\times^{\texttt{co}}$
$\texttt{COS}^{\{u,h\},rep^c}$	✓	✓	✓	✓	×	✓	✓	×	×	×
$\texttt{COS}^{e,rep^c}$	✓	✓	✓	✓	×	✓	✓	×	×	×
$\texttt{COS}^{mc,rep^c}$	✓	✓	✓	✓	×	✓	✓	×	×	×
$\texttt{COS}^{\texttt{AR}}_{\sigma,\tau_e}$	✓	✓	✓	✓	✓	×	✓	✓	×	✓
$\texttt{CSS}^{\mathcal{M},\Sigma}$	✓	✓	✓	✓	✓	✓	✓	×	✓	✓
$\texttt{CSS}^{\mathcal{M},min}$	✓	✓	✓	✓	✓	✓	✓	×	×	×
$\texttt{CSS}^{\mathcal{M},leximin}$	✓	✓	✓	✓	✓	✓	✓	×	×	×

TABLE 2 – Propriétés satisfaites par les COS étudiés dans ce travail. Le symbole ✓(resp. ×) indique que la propriété est respectée (resp. violée) par la sémantique pour $\sigma \in \texttt{co}, \texttt{pr}, \texttt{stb}$. Le symbole $\times^{\texttt{co}}$ signifie que la propriété n'est pas satisfaite lorsque $\sigma = \texttt{co}$ mais l'est lorsque $\sigma \in \texttt{pr}, \texttt{stb}$.

système est toujours capable de produire au moins une solution. La propriété de Conflict-freeness (CF) assure la cohérence avec le AF sous-jacent vis-à-vis d'une sémantique basée sur les extensions (le résultat est donc compatible avec la sémantique choisie). Elle garantit également le respect des contraintes du système, permettant ainsi la sélection d'extensions acceptables conformes aux règles et attentes définies. Enfin, la propriété d'Extension Unanimity (EU) constitue un indicateur clé de la robustesse de la sémantique, reflétant un véritable consensus.

Un autre avantage réside dans la distinction nette entre nos approches égalitariste (leximin) et utilitariste (somme), reflétée dans les propriétés additionnelles. En effet, pour une méthode égalitariste, les propriétés EM et Cn ne sont pas souhaitables, tandis qu'elles sont avantageuses dans les méthodes utilitaristes. Cette flexibilité permet de s'adapter à divers scénarios de vote. Notons que ces propriétés demeurent inchangées, indépendamment du choix parmi les trois sémantiques considérées.

D'après les propositions, $\texttt{COS}^{AB,rep}_{\sigma}$ pour $\sigma \in \{\texttt{pr}, \texttt{stb}\}$ constitue la seule autre approche satisfaisant l'ensemble des propriétés essentielles. Cependant, deux différences majeures par rapport à CSS apparaissent clairement. Premièrement, les résultats semblent dépendre de la sémantique basée sur les extensions utilisée. En effet, on constate que les propriétés EU, EM et Cn ne sont plus satisfaites lorsque la sémantique complète est employée. Cela s'explique par la manière dont cette approche traite les extensions de longueurs différentes. La divergence provient du fait que la normalisation des opérateurs dépend de la longueur du vote et non de la longueur de l'extension (voir la Définition 5). Par exemple, considérons un AF $\mathcal{F}$ tel que $\mathcal{E}_{\texttt{co}}(\mathcal{F}) = \{\{a\}, \{a, d\}\}$. Dans ce cas, les deux extensions obtiendraient le même score si le bulletin d'approbation était $\{a\}$. Bien que cette caractéristique ne soit pas intrinsèquement défavorable, elle peut conduire à un comportement inattendu sous certaines sémantiques, comme la sémantique complète, où les extensions peuvent être incluses les unes dans les autres. Deuxièmement, la distinction entre les approches utilitariste et égalitariste est moins marquée que dans CSS, où les propriétés EM et Cn jouent le rôle d'indicateurs clairs.

Nous mettons en garde contre la satisfaction inconditionnelle de la propriété d'Unanimité Argumentative. En effet, cette propriété permet de mieux comprendre comment les votes sont exploités pour sélectionner l'ensemble (ou les ensembles) d'arguments correspondant au résultat d'un COS. Par exemple, $\texttt{COS}^{\texttt{AR}}$ suppose que les votes prévalent sur la relation d'attaque, de sorte qu'un argument recueillant l'unanimité des votes en faveur de son acceptation sera nécessairement accepté. À l'inverse, dans le cas de CSS, les votes servent à orienter la sélection des extensions du AF initial qui correspondent le mieux aux préférences exprimées par les votes. Ainsi, même si tous les votes sont en faveur de l'acceptation d'un argument qui n'apparaît dans aucune extension, c'est-à-dire qu'il est incompatible avec le AF, cet argument ne fera jamais partie du résultat.

7 Travaux connexes

D'autres contributions ont exploré divers aspects de l'agrégation au sein des cadres d'argumentation, mais sous des perspectives différentesm et donc non comparables aux approches présentés dans ce papier. Certains travaux n'intègrent pas les votes [1], tandis que d'autres se basent sur des structures qui diffèrent de nos AFs, comme les AF bipolaires ou les AF quantitatifs [18, 20]. Dans le domaine de la fusion des systèmes d'argumentation, de nombreux travaux ont été menés, notamment [11, 8, 9]. D'autres recherches portent sur l'agrégation des relations d'attaque ou des relations d'attaque pondérées [22, 14]. Enfin, certaines études se concentrent sur les sémantiques graduées plutôt que sur les sémantiques d'extensions [19].

8 Conclusion

Dans cet article, nous avons proposé un nouveau système d'argumentation, le système d'argumentation basé sur les opinions (OBAF), pour améliorer la résolution démocratique par l'agrégation d'opinions. Les votants y expriment leur préférence quant à l'acceptabilité des arguments, et l'agrégation est assurée par une nouvelle classe de sémantiques, les sémantiques d'opinion collective (COS), conçues pour représenter et agréger efficacement les opinions. Nous démontrons que la sémantique de satisfaction collective (CSS) surpasse systématiquement les approches existantes, offrant des résultats plus précis et robustes en matière d'agrégation d'opinions.

Sur le plan théorique, notre travail fournit un cadre axiomatique pour comparer différentes méthodes d'agrégation grâce à OBAF et aux axiomes proposés. Sur le plan pratique, il offre une méthode applicable aux débats en ligne, conciliant démocratie et liberté individuelle. Nos approches égalitaristes et utilitaristes garantissent équité et inclusivité dans les processus décisionnels. De plus, notre cadre permet d'intégrer facilement d'autres méthodologies et d'en évaluer les propriétés axiomatiques. Notre méthode se montre robuste et flexible, répondant à la plupart des

axiomes sous diverses sémantiques, ce qui confirme son potentiel pour des applications variées.

Références

[1] Stephane Airiau, Elise Bonzon, Ulle Endriss, Nicolas Maudet, and Julien Rossit. Rationalisation of Profiles of Abstract Argumentation Frameworks : Extended Abstract. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI'17)*, pages 4776–4780, 2017.

[2] Leila Amgoud and Vivien Beuselinck. Equivalence of semantics in argumentation. In *Proceedings of the 18th International Conference on Principles of Knowledge Representation and Reasoning (KR'21)*, pages 32–41, 2021.

[3] Leila Amgoud and Claudette Cayrol. A reasoning model based on the production of acceptable arguments. *Annals of Mathematics and Artificial Intelligence*, 34(1-3) :197–215, 2002.

[4] Pietro Baroni, Martin Caminada, and Massimiliano Giacomin. An introduction to argumentation semantics. *The Knowledge Engineering Review*, 26(4) :365–410, 2011.

[5] Michael Bernreiter, Jan Maly, Oliviero Nardi, and Stefan Woltran. Combining Voting and Abstract Argumentation to Understand Online Discussions. In *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS'24)*, page 170–179, 2024.

[6] Martin Caminada. On the Issue of Reinstatement in Argumentation. In *Proceedings of the 10th European Conference on Logics in Artificial Intelligence (JELIA'06)*, page 111–123. Springer, 2006.

[7] Martin Caminada and Gabriella Pigozzi. On judgment aggregation in abstract argumentation. *Autonomous Agents and Multi-Agent Systems*, 22(1) :64–102, 2011.

[8] Weiwei Chen and Ulle Endriss. Preservation of semantic properties during the aggregation of abstract argumentation frameworks. In *Proceedings of the Sixteenth Conference on Theoretical Aspects of Rationality and Knowledge (TARK 2017)*, pages 118–133. OPA, 2017.

[9] Sylvie Coste-Marquis, Caroline Devred, Sébastien Konieczny, Marie-Christine Lagasquie-Schiex, and Pierre Marquis. On the merging of Dung's argumentation systems. *Artificial Intelligence*, 171(10) :730–753, 2007.

[10] Claude d'Aspremont and Louis Gevers. Social welfare functionals and interpersonal comparability. In Kenneth Arrow, Amartya Sen, and Kotaro Suzumura, editors, *Handbook of Social Choice and Welfare*, volume 1, chapter 10, pages 459–541. Elsevier, 2002.

[11] Jérôme Delobelle, Adrian Haret, Sébastien Konieczny, Jean-Guy Mailly, Julien Rossit, and Stefan Woltran. Merging of abstract argumentation frameworks. In *Proceedings of the 15th International Conference on Principles of Knowledge Representation and Reasoning (KR'16)*, pages 33–42, 2016.

[12] Phan Minh Dung. On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, 77(2) :321–357, 1995.

[13] Paul Dunne, Pierre Marquis, and Michael Wooldrige. Argument Aggregation : Basic Axioms and Complexity Results. *Frontiers in Artificial Intelligence and Applications*, 245 :129–140, 2012.

[14] Paul E. Dunne, Anthony Hunter, Peter McBurney, Simon Parsons, and Michael Wooldridge. Weighted argument systems : Basic definitions, algorithms, and complexity results. *Artificial Intelligence*, 175(2) :457–486, 2011.

[15] Dov Gabbay and Odinaldo Rodrigues. A self-correcting iteration schema for argumentation networks. *Frontiers in Artificial Intelligence and Applications*, 266 :377–384, 2014.

[16] Jacob Jacoby and Michael S. Matell. Three-Point Likert Scales Are Good Enough. *Journal of Marketing Research*, 8(4) :495–500, 1971.

[17] Souhila Kaci, Leendert van Der Torre, Srdjan Vesic, and Serena Villata. Preference in Abstract Argumentation. In Pietro Baroni, Dov Gabbay, Massimilino Giacomin, and Leendert van der Torre, editors, *Handbook of Formal Argumentation*, volume 2, chapter 10. College Publications, 2021.

[18] Stefan Lauren, Francesco Belardinelli, and Francesca Toni. Aggregating bipolar opinions. In *Proceedings of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021)*, pages 746–754, 2021.

[19] João Leite and João Martins. Social abstract argumentation. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI'11)*, page 2287–2292, 2011.

[20] Antonio Rago and Francesca Toni. Quantitative Argumentation Debates with Votes for Opinion Polling. In *Proceedings of the 20th International Conference on Principles and Practice of Multi-Agent Systems (PRIMA'17)*, pages 369–385, 2017.

[21] Juliete Rossie, Jérôme Delobelle, Sébastien Konieczny, Clément Lens, and Srdjan Vesic. Collective Satisfaction Semantics for Opinion Based Argumentation. In *Proceedings of the 21st International Conference on Principles of Knowledge Representation and Reasoning (KR' 24)*, pages 631–641, 8 2024.

[22] Fernando A Tohmé, Gustavo A. Bodanza, and Guillermo R. Simari. Aggregation of Attack Relations : A Social-Choice Theoretical Analysis of Defeasibility Criteria. In *Proceedings of the 5th International Conference on Foundations of Information and Knowledge Systems (FoIKS'08)*, pages 8–23. Springer Berlin Heidelberg, 2008.

Systèmes d'argumentation incomplets avec plausibilité

Jean-Guy Mailly

Université Toulouse Capitole, IRIT

jean-guy.mailly@irit.fr

Résumé

Les systèmes d'argumentation incomplets (IAF pour Incomplete Argumentation Framework*) sont des systèmes d'argumentation abstraits avec des arguments et des attaques incertains. Ils ont gagné en popularité ces 10 dernières années. L'approche de raisonnement classique consiste à utiliser des complétions, qui représentent les différents scénarios compatibles avec l'IAF. Les problèmes de décision standard peuvent être adaptés pour vérifier si une propriété est vraie pour certaines ou toutes les complétions. Cependant, cette approche suppose une plausibilité équivalente pour toutes les complétions, ce qui n'est pas toujours réaliste. Nous proposons les IAF avec Plausibilité (pIAF), une généralisation des IAF où les agents peuvent raisonner sur la plausibilité relative des complétions. Nous étudions la complexité des problèmes de décision usuels adaptés à ce modèle et introduisons de nouveaux problèmes liés à la plausibilité relative des extensions, fournissant des bornes supérieures de complexité pour ceux-ci.*

Mots-clés

Argumentation, Connaissance incomplète, Plausibilité.

1 Introduction

L'argumentation abstraite [3] consiste à étudier l'acceptabilité des arguments en fonction de leurs relations uniquement, ignorant leur provenance ou leur structure interne. Dans le cadre de base, on considère des graphes orientés appelés systèmes d'argumentation (ou AF, pour *Argumentation Framework*) dont les noeuds sont les arguments et les arcs les attaques (c'est-à-dire les contradictions) entre arguments. On raisonne classiquement avec ce genre de cadre au moyen de sémantiques à base d'extensions, qui sont des ensembles d'arguments collectivement acceptables.

Parmi les nombreuses généralisations du cadre de Dung, les systèmes d'argumentation incomplets (ou IAF, pour *Incomplete Argumentation Framework*) ont reçu une certaine attention [5]. Dans ce type de cadre, un argument ou une attaque entre deux arguments peut être identifié comme *incertain*, ce qui signifie que l'agent qui raisonne considère deux options à son sujet : soit l'argument (ou l'attaque) existe bel et bien, soit ce n'est pas le cas, mais il n'y a pas de quantification de la certitude de l'agent concernant ces deux scénarios (comme ce serait le cas, par exemple, avec l'introduction de probabilités [4]). Pour raisonner, les problèmes classiques de l'argumentation abstraite sont généralisés en utilisant les complétions de l'IAF, c'est-à-dire les AF « classiques » qui représentent les différents scénarios compatibles avec l'IAF. Par exemple, un argument est *possiblement accepté* s'il est accepté pour au moins une complétion, et *nécessairement accepté* s'il l'est pour toutes les complétions. Cependant, dans ce cas, toutes les complétions ont la même plausibilité pour l'agent (contrairement au cas où des probabilités sont utilisées [4]), ce qui n'est pas forcément réaliste dans toutes les situations.

Par exemple, considérons le cas où un agent modélise ses connaissances (incomplètes) sur un autre agent dans un débat (comme dans [2] qui utilise une généralisation des IAF pour représenter l'adversaire dans une négociation). Si l'agent sait que son adversaire est susceptible d'utiliser soit l'argument b_1 soit l'argument b_2, mais qu'il y a une forme d'incompatibilité entre eux (par exemple, car b_1 a plus de chance d'être utilisé par une personne de droite et b_2 par une personne de gauche), alors les deux complétions qui contiennent soit b_1 soit b_2 (mais pas les deux) sont les plus plausibles, suivies de celle qui ne contient ni b_1 ni b_2, et enfin celle qui contient à la fois b_1 et b_2 est très peu plausible. Pour permettre la modélisation de ce genre de cas, nous introduisons les systèmes d'argumentation incomplets avec plausibilité (pIAF), et adaptons à ce cadre les problèmes de décision classique des IAF. Nous proposons également de nouveaux problèmes liés à la plausibilité des complétions et des extensions. Pour tous ces problèmes, nous fournissons des résultats de complexité préliminaires. Ce travail a été accepté à la conférence SAC 2025, et la version complète de l'article est disponible sur HAL : https://hal.science/hal-04977679.

2 Définitions et synthèse des résultats

Étant donné un IAF $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{R}, \mathcal{R}^? \rangle$ (avec $\mathcal{A}$ et $\mathcal{R}$ les arguments et attaques certain(e)s, et $\mathcal{A}^?$ et $\mathcal{R}^?$ les arguments et attaques incertain(e)s), on note $\mathcal{F}^{\mathbf{A}}$ l'ensemble de tous les AF construits sur l'ensemble d'arguments $\mathbf{A} = \mathcal{A} \cup \mathcal{A}^?$.

Définition 1. *Un* système d'argumentation incomplet avec plausibilité *(pIAF pour* Incomplete Argumentation Framework with Plausibility*) est un tuple* $p\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{R}, \mathcal{R}^?, pl \rangle$ *où* $\langle \mathcal{A}, \mathcal{A}^?, \mathcal{R}, \mathcal{R}^? \rangle$ *est un IAF et* $pl : \mathcal{F}^{\mathbf{A}} \times \mathcal{F}^{\mathbf{A}} \to \{0, 1\}$ *est une* fonction de plausibilité.

Concernant la fonction de plausibilité, il s'agit d'une représentation abstraite de la capacité d'un agent à évaluer si un

AF $\mathcal{F}_1$ (ou une complétion d'un IAF) est au moins aussi plausible qu'un autre AF $\mathcal{F}_2$, auquel cas $pl(\mathcal{F}_1, \mathcal{F}_2) = 1$. On suppose que cette fonction est calculable en temps polynomial (par rapport au nombre d'arguments $|\mathbf{A}|$), et qu'elle induit un pré-ordre total entre les complétions.
Nous introduisons les concepts de complétions les plus plausibles et de plausibilité relative des extensions.

Définition 2. *Soit $p\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{R}, \mathcal{R}^?, pl \rangle$ un pIAF. Ses* complétions les plus plausibles *(MPC pour* most plausible completions*) sont* $\mathrm{mpc}(p\mathcal{I}) = \{\mathcal{F}^* \in \mathrm{comp}(\widetilde{p\mathcal{I}}) \mid \forall \mathcal{F}' \in \mathrm{comp}(\widetilde{p\mathcal{I}}), pl(\mathcal{F}^*, \mathcal{F}') = 1\}$ *où $\widetilde{p\mathcal{I}} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{R}, \mathcal{R}^? \rangle$ est l'IAF (classique) obtenu à partir de $p\mathcal{I}$ en ignorant la fonction de plausibilité, et $\mathrm{comp}(\widetilde{p\mathcal{I}})$ sont ses complétions.*

Définition 3. *Soient $p\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{R}, \mathcal{R}^?, pl \rangle$ un pIAF, et $S, S' \subseteq \mathcal{A} \cup \mathcal{A}^?$ deux ensembles d'arguments. S est* plus plausible que S' *pour la sémantique σ si $\mathrm{comp}^\sigma_{S'}(p\mathcal{I}) = \emptyset$, ou il existe $\mathcal{F} \in \mathrm{comp}^\sigma_S(p\mathcal{I})$ tel que pour tout $\mathcal{F}' \in \mathrm{comp}^\sigma_{S'}(p\mathcal{I})$, $pl(\mathcal{F}, \mathcal{F}') = 1$, avec $\mathrm{comp}^\sigma_S(p\mathcal{I}) = \{\mathcal{F}^* \in \mathrm{comp}(\widetilde{p\mathcal{I}}) \mid S \in \sigma(p\mathcal{I})\}$.*

Les problèmes de décision étudiés sont les suivants :
• σ-PVER Soient $p\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{R}, \mathcal{R}^?, pl \rangle$ et $S \subseteq \mathcal{A}$. Est-ce que $S \in \sigma(\mathcal{F}^*)$ pour un $\mathcal{F}^* \in \mathrm{mpc}(p\mathcal{I})$?
• σ-PCA Soient $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{R}, \mathcal{R}^? \rangle$ et $a \in \mathcal{A}$. Est-ce que $a \in \bigcup_{S \in \sigma(\mathcal{F}^*)} S$ pour un $\mathcal{F}^* \in \mathrm{mpc}(p\mathcal{I})$?
• σ-PSA Soient $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{R}, \mathcal{R}^? \rangle$ et $a \in \mathcal{A}$. Est-ce que $a \in \bigcap_{S \in \sigma(\mathcal{F}^*)} S$ pour un $\mathcal{F}^* \in \mathrm{mpc}(\mathcal{I})$?
• σ-NVER Soient $p\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{R}, \mathcal{R}^?, pl \rangle$ et $S \subseteq \mathcal{A}$. Est-ce que $S \in \sigma(\mathcal{F}^*)$ pour tout $\mathcal{F}^* \in \mathrm{mpc}(p\mathcal{I})$?
• σ-NCA Soient $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{R}, \mathcal{R}^? \rangle$ et $a \in \mathcal{A}$. Est-ce que $a \in \bigcup_{S \in \sigma(\mathcal{F}^*)} S$ pour tout $\mathcal{F}^* \in \mathrm{mpc}(p\mathcal{I})$?
• σ-NSA Soient $\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{R}, \mathcal{R}^? \rangle$ et $a \in \mathcal{A}$. Est-ce que $a \in \bigcap_{S \in \sigma(\mathcal{F}^*)} S$ pour tout $\mathcal{F}^* \in \mathrm{mpc}(\mathcal{I})$?
• MPC Soient $p\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{R}, \mathcal{R}^?, pl \rangle$ et $\mathcal{F}^* \in \mathrm{comp}(\widetilde{p\mathcal{I}})$. Est-ce que $\mathcal{F}^* \in \mathrm{mpc}(p\mathcal{I})$?
• σ-RPEC Soient $p\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{R}, \mathcal{R}^?, pl \rangle$, $\mathcal{F}^* \in \mathrm{comp}(\widetilde{p\mathcal{I}})$ et $S' \subseteq \mathcal{A} \cup \mathcal{A}^?$. Est-ce que pour chaque $\mathcal{F}' \in \mathrm{comp}^\sigma_{S'}(p\mathcal{I})$, $pl(\mathcal{F}^*, \mathcal{F}') = 1$?
• σ-RPE Soient $p\mathcal{I} = \langle \mathcal{A}, \mathcal{A}^?, \mathcal{R}, \mathcal{R}^?, pl \rangle$ et S, $S' \subseteq \mathcal{A} \cup \mathcal{A}^?$. Est-ce que S est plus plausible que S' ?

Les six premiers sont des adaptations des problèmes standard des IAF [5] en ne tenant compte que des complétions les plus plausibles. MPC vérifie si une complétion fait partie des plus plausibles. σ-RPEC vérifie si une complétion donnée est plus plausibles que toutes les complétions correspondant à une extension donnée. Enfin, σ-RPE vérifie si une extension est plus plausible qu'une autre.
La Table 1 synthétise les résultats de complexité obtenus pour les différents problèmes étudiés, pour les sémantiques de Dung [3] : adm, stb, co, gr et pr correspondant respectivement aux sémantiques *admissible*, *stable*, *complète*, *de base* (*grounded*) et *préférée*. L'observation générale est une complexité située généralement entre le premier et le deuxième niveau de la hiérarchie polynomiale, ce qui permet d'envisager des approches de résolution utilisant des algorithmes de type CEGAR, comme pour les IAF [1].

Problème	adm	stb	co	gr	pr
σ-PVER	NP-h $\in \Sigma_2^P$	NP-h $\in \Sigma_2^P$	NP-h $\in \Sigma_2^P$	NP-h $\in \Sigma_2^P$	Σ_2^P-c
σ-PCA	NP-h $\in \Sigma_2^P$	NP-h $\in \Sigma_2^P$	NP-h $\in \Sigma_2^P$	NP-h $\in \Sigma_2^P$	NP-h $\in \Sigma_2^P$
σ-PSA	trivial	Σ_2^P-c	NP-h $\in \Sigma_2^P$	NP-h $\in \Sigma_2^P$	Σ_3^P-c
σ-NVER	coNP-h $\in \Pi_2^P$	coNP-h $\in \Pi_2^P$	coNP-h $\in \Pi_2^P$	coNP-h $\in \Pi_2^P$	coNP-h $\in \Pi_2^P$
σ-NCA	Π_2^P-c	Π_2^P-c	Π_2^P-c	coNP-h $\in \Pi_2^P$	Π_2^P-c
σ-NSA	trivial	coNP-h $\in \Pi_2^P$	coNP-h $\in \Pi_2^P$	coNP-h $\in \Pi_2^P$	Π_2^P-h $\in \Pi_3^P$
MPC σ-RPEC σ-RPE	$\in$ coNP $\in \Sigma_2^P$	$\in$ coNP $\in \Sigma_2^P$	coNP-c $\in$ coNP $\in \Sigma_2^P$	$\in$ coNP $\in \Sigma_2^P$	$\in \Pi_2^P$ $\in \Sigma_3^P$

TABLE 1 – Complexité du raisonnement avec les pIAF. C-c signifie « complet pour la classe C », et C-h « difficile pour la classe C ». MPC ne dépend pas de la sémantique.

3 Conclusion

Nous avons proposé la première approche *qualitative* pour raisonner sur la plausibilité des scénarios encodés dans un système d'argumentation incomplet. C'est un premier pas, par exemple, vers une meilleure représentation de l'adversaire dans des protocoles de négociation argumentée [2]. Le principal défi pour de futurs travaux est la proposition d'une représentation moins abstraite de la fonction de plausibilité qui conserverait une forme de compacité spatiale, dans le but d'éviter de définir un pré-ordre total qui reviendrait à énumérer toutes les complétions possibles.

Remerciements

Ce travail est soutenu par le projet AGGREEY (ANR-22-CE23-0005) et la CPJ AIDAL (ANR-22-CPJ1-0061-01).

Références

[1] D. Baumeister, M. Järvisalo, D. Neugebauer, A. Niskanen, and J. Rothe. Acceptance in incomplete argumentation frameworks. *Artif. Intell.*, 295 :103470, 2021.

[2] Y. Dimopoulos, J.-G. Mailly, and P. Moraitis. Arguing and negotiating using incomplete negotiators profiles. *Auton. Agents Multi Agent Syst.*, 35(2) :18, 2021.

[3] 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.

[4] H. Li, N. Oren, and T. J. Norman. Probabilistic argumentation frameworks. In *Proc. of TAFA'11*, pages 1–16, 2011.

[5] 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.

Cadres d'argumentation bipolaire quantitatifs avec incertitude

Jordan Thieyre[1], Caren Al Anaissy[1], Aurélie Beynier[1], Sébastien Destercke[2], Nicolas Maudet[1], Srdjan Vesic[3]

[1] LIP6 - CNRS, Sorbonne Université, 4 place Jussieu, F-75005 Paris, France
[2] UMR CNRS 7253, Heudiasyc, Sorbonne Université, Université de Technologie de Compiègne, Compiègne, France
[3] CRIL - CNRS - Univ. Artois

{jordan.thieyre, caren.al-anaissy, aurelie.beynier, nicolas.maudet}@lip6.fr
sebastien.destercke@hds.utc.fr
vesic@cril.fr

Résumé

Les débats sur les plateformes de délibération en ligne permettant l'échange d'arguments et les votes peuvent être modélisés à l'aide de l'argumentation abstraite, mais la rareté des votes sur les contributions des utilisateurs reste un défi. Dans ce papier, nous proposons une généralisation des Quantitative Bipolar Argumentation Frameworks en y incorporant de l'incertitude. Concrètement, nous utilisons les votes sur les arguments pour initialiser les intervalles de poids à l'aide du modèle de Dirichlet imprécis, permettant une représentation plus fiable de la force des arguments en présence d'un nombre limité de données.

1 Introduction

La délibération vise à favoriser des décisions collectives équilibrées. Les plateformes de débat en ligne, comme Debatabase [1], Kialo [2] ou Debategraph [3], facilitent ce processus en structurant les échanges autour d'arguments "POUR" et "CONTRE", et permettent parfois aux utilisateurs de voter sur la pertinence des arguments.

Avec l'essor de ces outils dans les pratiques démocratiques numériques, il devient crucial de comprendre comment évaluer les arguments à grande échelle, en prenant en compte leurs interactions et, lorsque disponibles, les votes des participants. Lire l'ensemble des contributions d'un débat étant en général peu réaliste, il est pertinent de recourir à des méthodes automatisées d'analyse.

L'argumentation computationnelle fournit un cadre formel pour traiter l'information conflictuelle, incomplète ou incertaine. Elle modélise les débats sous forme de graphes d'arguments reliés par des relations d'attaque et de support, et utilise des sémantiques pour évaluer la force de chaque argument en fonction de ces relations. Ces sémantiques abstraient le contenu des arguments, réduisant ainsi les biais potentiels dans l'évaluation.

1. https://idebate.net/resources/debatabase
2. https://www.kialo.com/
3. http://debategraph.org

Parmi les nombreux cadres proposés, les *cadres d'argumentation bipolaires quantitatifs* (QBAFs) sont particulièrement adaptés aux débats en ligne. Ils intègrent à la fois les relations bipolaires entre arguments et des poids initiaux, souvent issus des votes. Les *sémantiques graduelles bipolaires* [2, 5, 1, 4] permettent ensuite de calculer un degré d'acceptabilité finale pour chaque argument.

Problème de recherche. Un défi majeur dans les débats en ligne est la *parcimonie des votes* : la plupart des utilisateurs ne votent que sur une faible proportion des arguments [3]. Ce phénomène limite la fiabilité des évaluations fondées sur les seuls votes. Plutôt que de tenter de prédire les votes manquants ou de solliciter davantage les utilisateurs, une approche conservatrice consiste à reconnaître cette incertitude et à s'assurer qu'elle soit correctement prise en compte dans les résultats affichés.

Notre approche. Nous généralisons le cadre d'argumentation bipolaire quantitatif afin de prendre en compte l'incertitude sur les poids initiaux des arguments. Pour cela, chaque argument est associé à un *intervalle* de poids possibles, reflétant la variabilité et la rareté des votes exprimés. Ces intervalles sont initialisés à partir des votes observés grâce au *modèle de Dirichlet imprécis*, qui permet de quantifier l'incertitude tout en évitant les inférences prématurées.

Nos contributions sont les suivantes :

- Introduction du cadre d'argumentation bipolaire quantitatif imprécis (*IQBAF*) pour modéliser les poids initiaux sous forme d'intervalles ;
- Utilisation du modèle de Dirichlet imprécis pour estimer ces poids à partir de données de vote partiellement observées ;
- Proposition d'une méthode pour appliquer une famille de sémantiques graduelles bipolaires à ces *IQBAFs* ;
- Validation expérimentale sur un corpus réel issu de la plateforme Kialo.

2 Cadres d'argumentation bipolaire quantitatifs imprécis

Lorsqu'un argument dispose de peu d'informations — notamment en cas de votes clairsemés — il devient difficile de lui attribuer un poids initial précis reflétant l'opinion des utilisateurs. Pour remédier à cela, nous nous inspirons de la théorie des probabilités imprécises et proposons un cadre dans lequel le poids initial d'un argument est représenté non plus par une valeur unique, mais par un *intervalle de valeurs possibles*. La borne inférieure (resp. supérieure) indique la valeur minimale (resp. maximale) que peut prendre ce poids. Ce formalisme donne lieu à la définition des *cadres d'argumentation bipolaires quantitatifs imprécis* (IQBAFs) comme suit :

Définition 1. *Soit $\mathcal{I}_{[0;1]}$ l'ensemble des intervalles inclus ou égaux dans $[0;1]$. Un cadre d'argumentation bipolaire quantitatif imprécis (IQBAF) est un quadruplet $(\mathcal{A}, \mathcal{R}^-, \mathcal{R}^+, i^i)$ où : (i) $\mathcal{A}$ est un ensemble fini d'arguments, (ii) $\mathcal{R}^- \subseteq \mathcal{A} \times \mathcal{A}$ représente une relation binaire acyclique d'attaque, (iii) $\mathcal{R}^+ \subseteq \mathcal{A} \times \mathcal{A}$ représente une relation binaire acyclique de support, (iv) $i^i : \mathcal{A} \to \mathcal{I}_{[0;1]}$ associe à chaque argument a un intervalle de poids initiaux $i^i(a) = [\underline{i}^i(a), \overline{i}^i(a)]$.*

3 Calcul des intervalles d'acceptabilité finale

Plutôt que d'introduire une nouvelle sémantique graduelle bipolaire dédiée aux *IQBAFs*, nous proposons une méthode pour adapter les sémantiques existantes afin de calculer les intervalles d'acceptabilité finale i^f. Pour chaque argument a, $i^f(a) = [\underline{i^f}(a), \overline{i^f}(a)]$ est défini à partir des bornes de ses attaquants et supporters :

$$\underline{i^f}(a) = \min_{\substack{x \in i^i(a) \\ Y \in \{y \in i^f(b) | b \in \mathcal{R}^-(a)\} \\ Z \in \{z \in i^f(c) | c \in \mathcal{R}^+(a)\}}} s^f(x, Y, Z)$$

et

$$\overline{i^f}(a) = \max_{\substack{x \in i^i(a) \\ Y \in \{y \in i^f(b) | b \in \mathcal{R}^-(a)\} \\ Z \in \{z \in i^f(c) | c \in \mathcal{R}^+(a)\}}} s^f(x, Y, Z)$$

Nous montrons que si la sémantique satisfait un ensemble de propriétés peu contraignantes, alors les bornes de $i^f(a)$ peuvent être obtenues en combinant : poids initial minimal (resp. maximal), degré d'acceptabilité finale des attaquants maximaux (resp. minimaux), et ceux des supporters minimaux (resp. maximaux). L'intervalle d'acceptabilité finale d'un argument a peut alors être calculé directement à partir des bornes inférieure et supérieure de ses attaquants et de ses supporters comme suit :

$$\underline{i^f}(a) = \sigma(\underline{i^i}(a), \{\overline{i^f}(b) | b \in \mathcal{R}^-(a)\}, \{\underline{i^f}(c) | c \in \mathcal{R}^+(a)\})$$

et

$$\overline{i^f}(a) = \sigma(\overline{i^i}(a), \{\underline{i^f}(b) | b \in \mathcal{R}^-(a)\}, \{\overline{i^f}(c) | c \in \mathcal{R}^+(a)\})$$

Nous montrons également que l'ajout d'information (en particulier sous forme de nouveaux votes) ne réduit pas nécessairement les intervalles d'incertitudes. Nous avons donc mené une expérimentation sur un jeu de données de presque 3000 débats issus de la plateforme Kialo.

4 Résultats expérimentaux

Pour chaque argument, nous avons appliqué la méthode *IDM* pour estimer un intervalle de poids initial à partir des votes, puis utilisé la sémantique *QuEM* pour calculer l'intervalle final. Seuls quelques débats montrent une réduction nette de l'incertitude, souvent liés à un nombre suffisant de votes ou à une structure dominée par un type de relation (attaque ou support). Concernant l'acceptabilité, on observe un biais global vers une moindre acceptation finale des thèses. Certains débats présentent un basculement marqué, l'intervalle final ne recoupant pas l'initial. De façon générale, bien que la thèse centrale concentre de nombreux votes, la prise en compte des arguments "POUR" et "CONTRE" élargit l'éventail des conclusions possibles.

Remerciements

Ce travail a été soutenu par le projet AGGREEY (ANR-22-CE23-0005) de l'ANR (Agence Nationale de la Recherche), ainsi que par une subvention de l'État français gérée par l'Agence Nationale de la Recherche dans le cadre du programme France 2030, référence ANR-22-EXEN-0004 (PEPR eNSEMBLE / PC3 MATCHING).

Références

[1] Leila Amgoud and Jonathan Ben-Naim. Evaluation of arguments in weighted bipolar graphs. *International Journal of Approximate Reasoning*, 99 :39–55, 2018.

[2] Pietro Baroni, Marco Romano, Francesca Toni, Marco Aurisicchio, and Giorgio Bertanza. Automatic evaluation of design alternatives with quantitative argumentation. *Argument & Computation*, 6(1) :24–49, 2015.

[3] Beth Goldberg, Diana Acosta-Navas, Michiel Bakker, Ian Beacock, Matt Botvinick, Prateek Buch, Renée DiResta, Nandika Donthi, Nathanael Fast, Ravi Iyer, Zaria Jalan, Andrew Konya, Grace Kwak Danciu, Hélène Landemore, Alice Marwick, Carl Miller, Aviv Ovadya, Emily Saltz, Lisa Schirch, Dalit Shalom, Divya Siddarth, Felix Sieker, Christopher Small, Jonathan Stray, Audrey Tang, Michael Henry Tessler, and Amy Zhang. AI and the future of digital public squares, ArXiv, https ://arxiv.org/abs/2412.09988, 2024.

[4] Nico Potyka. Continuous Dynamical Systems for Weighted Bipolar Argumentation. In *Proceedings of the Sixteenth International Conference, KR 2018, Tempe, Arizona*, July 2018.

[5] Antonio Rago, Francesca Toni, Marco Aurisicchio, Pietro Baroni, et al. Discontinuity-free decision support with quantitative argumentation debates. *Fifteenth International Conference on Principles of Knowledge Representation and Reasoning*, 16 :63–73, 2016.

Session 7 : Raisonnement par analogie et à partir de cas

L'analogie numérique revisitée et étendue, précisée, rétrécie ou agrandie

Yves Lepage[1] Miguel Couceiro[2]

[1] IPS, université Waseda, Japon
[2] INESC-ID, IST, Universidade de Lisboa, Portugal

yves.lepage@waseda.jp
miguel.j.couceiro@tecnico.ulisboa.pt

Résumé

Le lien entre analogie et moyennes généralisées a récemment été établi. Nous revisitons *ce lien avec des moyens algébriques, ce qui conduit à* étendre *l'ensemble support des analogies pour inclure le cas où l'un au moins des termes de l'analogie est égal à zéro. Nous* précisons *l'algorithme dichotomique de calcul des puissances analogiques et donnons les bornes pour son initialisation. Enfin, nous montrons comment la visualisation des analogies comme sortes de carrés pousse à créer d'autres analogies par* rétrécissement *ou* agrandissement.

Mots-clés

Analogie numérique ; algorithme ; bornes des puissances.

Abstract

The link between analogies and generalised means has recently been established. We re-explain this link from an algebraic point of view, which leads us to extend the support set of analogies, including the case where at least one of the terms of the analogy is zero. We give more details on the dichotomic algorithm for the computation of analogical powers, and provide bounds for its initialisation. Finally, the visualization of analogies as kinds of squares naturally leads to the creation of other analogies by shrinking or enlarging.

Keywords

Numerical analogy ; algorithm ; power bounds.

1 Introduction

L'analogie est une faculté cognitive fondamentale à l'œuvre dans la métaphore (l'atome est comme un système solaire) ou la comparaison (les nageoires sont au poisson ce que les ailes sont à l'oiseau). Afin d'équiper les modèles génératifs de cette faculté, il est nécessaire de mettre au clair des outils mathématiques précis applicables à des espaces de représentations numériques.

La lecture de maints travaux récents sur l'analogie [10, 15, 3] incite à conclure que trois propriétés la caractérisent :

— la réflexivité de la conformité
$(a : b :: a : b$ toujours vrai$)$;
— la symétrie de la conformité
$(a : b :: c : d \Leftrightarrow c : d :: a : b)$;
— et l'échange des moyens (lat. *permutando*)
$(a : b :: c : d \Leftrightarrow a : c :: b : d)$ [1].

Les deux premières caractérisent le symbole :: comme relation de ressemblance [8]. [2] La dernière est intrinsèque de l'analogie selon les Anciens [4]. L'application successive des deux dernières implique huit formes équivalentes pour une même analogie. Par exemple, en appliquant l'échange des moyens, puis la symétrie de la conformité, puis à nouveau l'échange des moyens, on obtient l'inversion des rapports (lat. *invertendo*) $(a : b :: c : d \Leftrightarrow a : c :: b : d \Leftrightarrow b : d :: a : c \Leftrightarrow b : a :: d : c)$.

La tradition, issue de l'analogie comme égalité de rapports [7], privilégie l'échange des moyens (*permutando*) et l'inversion des rapports (*invertendo*) [18, 14], mais une approche plus moderne donnerait comme transformations de base l'une des quatre opérations de symétrie de la conformité $(c : d :: a : b)$, d'inversion des rapports $(b : a :: c : d)$, d'échange des moyens $(a : c :: b : d)$ ou d'échange des extrêmes $(d : b :: c : a)$, comme choix pour l'élément s d'ordre 2 [3] et l'une des deux rotations des termes $b : d :: a : c$ ou $c : a :: d : b$ comme choix pour l'élément r d'ordre 4 afin de construire les huit éléments du groupe diédral D_8 des transformations invariantes du carré, c'est-à-dire $e, er = r, er^2 = r^2, er^3 = r^3, s, sr, sr^2, sr^3$ où e est l'élément neutre [12]. Ces huit éléments sont équivalents aux huit formes possibles d'une même analogie mentionnées plus haut. La liste ci-dessous fait un choix possible de s et r parmi bien d'autres.

$$\begin{array}{ll} a : b :: c : d = e & c : a :: d : b = r^3 \\ a : c :: b : d = sr^3 & c : d :: a : b = s \\ b : a :: d : c = sr^2 & d : b :: c : a = sr \\ b : d :: a : c = r & d : c :: b : a = r^2 \end{array}$$

1. Raison alterne ou raison par échange chez Ozanam [17].
2. L'ajout de la transitivité donnerait une relation d'équivalence.
3. L'ordre d'un élément dans un groupe est le nombre minimal de fois qui permet de retomber sur l'élément neutre. Ainsi, si $a * a \neq e$ et $a * a * a = e$, l'ordre de a est 3.

2 L'analogie numérique revisitée

2.1 Définition

Un précédent article [13] montrait comment définir l'analogie numérique entre nombres réels positifs et non nuls à partir des moyennes généralisées en posant qu'une analogie en p existe si la moyenne généralisée en p de ce que l'on appelle les *moyens*, b et c, est égale à la moyenne généralisée en p de ce que l'on appelle les *extrêmes*, a et d.

Définition : analogie numérique

$$\forall p \in \mathbb{R} \cup \{-\infty, +\infty\},\ \forall (a,b,c,d) \in \left(\mathbb{R}_+^*\right)^4,$$

$$a : b ::^p c : d \overset{\text{déf.}}{\Leftrightarrow} m_p(a,d) = m_p(b,c)$$

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 $p \in \mathbb{R} \cup \{-\infty, +\infty\}$ [9]. On retrouve :

- 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

$$m_0(x_1, \dots, x_N) = \lim_{p \to 0} m_p(x_1, \dots x_N) = \sqrt[N]{\prod_{i=1}^{N} x_i} ;$$

- le maximum des nombres quand p tend vers $+\infty$;
- le minimum des nombres quand p tend vers $-\infty$.

Dans le cas de l'analogie numérique, $N = 2$.

2.2 Propriétés

Le précédent article vérifiait que la définition donnée ci-dessus correspond bien aux idées que l'on se fait ordinairement de l'analogie mathématique, à savoir que l'on a bien les trois propriétés caractéristiques données dans l'introduction. Ils démontrait ces résultats à partir de la définition donnée.

De plus, il démontrait plusieurs résultats majeurs parmi lesquels le plus important est que, pour quatre nombres réels positifs non nuls quelconques, à un réordonnancemant près, il existe toujours une analogie numérique entre ces termes et elle est unique. Autrement dit, on peut toujours trouver un p tel que soit $a : b ::^p c : d$, soit $a : c ::^p d : b$, soit $a : d ::^p b : c$.

2.3 L'analogie numérique comme analogie induite par des structures algébriques

Nous montrons ci-dessous, en nous fondant toujours sur la définition de la moyenne généralisée, que l'analogie numérique peut en fait se déduire simplement de résultats algébriques connus.

Dans un premier temps, sans étendre l'ensemble sur lequel est définie l'analogie, en utilisant la moyenne généralisée comme la loi interne, commutative, du magma, on peut égaliser l'analogie numérique pour un p donné avec celle induite par la structure de magma commutatif.

Dans un deuxième temps, on montrera qu'en ajoutant à l'ensemble de départ une demi-droite dans le plan complexe, l'analogie numérique pour un p donné se déduit directement de la structure de groupe commutatif obtenue.

2.3.1 Analogie induite par la structure de magma commutatif

Un ensemble muni d'une loi interne est appelé magma. Tout magma commutatif induit une analogie [19, 11] définie par l'égalité de la composition des extrêmes et de la composition des moyens par la loi interne. Les trois propriétés énoncées dans l'introduction sont facilement vérifiables.

Définition : analogie du magma commutatif

Soit $(M, *)$ un magma commutatif.
$\forall (A,B,C,D) \in M^4$,

$$A : B ::_* C : D \overset{\text{déf.}}{\Leftrightarrow} A * D = C * B.$$

2.3.2 Analogie sur $(\mathbb{R}_+^*, m_p)$

On peut appliquer directement la définition précédente, pour n'importe quel p de $\mathbb{R}$, à l'ensemble $\mathbb{R}_+^*$ muni de la loi interne m_p, c'est-à-dire la moyenne généralisée en p :

$$\begin{array}{rll} m_p : & \mathbb{R}_+^* \times \mathbb{R}_+^* & \to \mathbb{R}_+^* \\ & (x,y) & \mapsto m_p(x,y) = \lim_{r \to p} \left(\frac{1}{2}(x^r + y^r) \right)^{1/r} \end{array}$$

Cette loi est commutative. On a donc une structure de magma commutatif, ce qui induit une analogie qui n'est autre, par définition, que l'analogie numérique en p.

$$\begin{aligned} a : b ::_{m_p} c : d &\Leftrightarrow m_p(a,d) = m_p(c,b) = m_p(b,c) \\ &\Leftrightarrow a : b ::^p c : d \end{aligned}$$

Comme c'est un magma, il n'y a pas lieu de parler d'élément neutre ni d'élément opposé.

On peut observer qu'il est loisible d'enlever le facteur un demi dans la loi interne $(x^p + y^p)^{1/p}$ et dans la limite pour les p finis non nuls, ce qui donne une autre loi interne. Pour le cas $p = 0$, on pose directement xy. Notons que, quel que soit p, les deux lois sont équivalentes et induisent la même analogie. $\forall p \in \mathbb{R}^*$,

$$m_p(a,d) = m_p(b,c) \Leftrightarrow (a^p + d^p)^{1/p} = (b^p + c^p)^{1/p}$$

et

$$m_0(a,d) = m_0(b,c) \Leftrightarrow ad = bc$$

2.3.3 Analogie induite par la structure de groupe commutatif

La structure de groupe commutatif induit aussi une analogie [19, 11]. On exprime directement le rapport comme application de la loi interne à l'élément opposé. Les trois propriétés données dans l'introduction sont facilement vérifiables.

Définition : analogie du groupe commutatif

Soit $(G, *)$ un groupe commutatif où X^{-1} note l'opposé de X. $\forall (A, B, C, D) \in G^4$,

$$A : B ::_* C : D \overset{\text{déf.}}{\Leftrightarrow} A * B^{-1} = C * D^{-1}.$$

Pour l'analogie numérique en p, la question est de trouver, pour un p donné, un sous-ensemble de $\mathbb{R}$ ou de $\mathbb{C}$ sur lequel on ait une loi interne de groupe commutatif dont la structure induise l'analogie numérique en p.

La moyenne généralisée en p, m_p, ne constitue pas une loi de groupe, car on n'a pas d'élément neutre. En effet, supposons qu'il en existe un, appelé e. On aurait alors :

$$\begin{aligned}\forall x \in \mathbb{R}_+^*, \quad \left(\frac{1}{2}(x^p + e^p)\right)^{1/p} = x &\Leftrightarrow \frac{1}{2}(x^p + e^p) = x^p \\ &\Leftrightarrow x^p + e^p = 2x^p \\ &\Leftrightarrow e^p = x^p \\ &\Leftrightarrow e = x\end{aligned}$$

Il n'y aurait pas unicité de l'élément neutre, ce qui est impossible car, dans un groupe, l'élément neutre est unique.

En revanche, nous allons montrer que la loi $+_p$ définie par $x +_p y = (x^p + y^p)^{1/p}$ constitue bien une loi commutative de groupe sur une extension de $\mathbb{R}_+^*$. Cette loi est obtenue en enlevant le facteur un demi de l'écriture de la moyenne généralisée, ainsi que la limite. Comme vu plus haut, on posera $x +_0 y = xy$. Il est alors nécessaire d'étendre $\mathbb{R}_+^*$ premièrement pour y ajouter l'élément neutre, et deuxièmement pour y ajouter les éléments opposés des éléments de $\mathbb{R}_+^*$. Si on y arrive, on aura alors immédiatement l'analogie numérique en p comme analogie induite par cette structure de groupe commutatif.

On peut au passage remarquer que, contrairement à m_p, la loi $+_p$ n'est pas une moyenne de Kolmogoroff. Les cinq axiomes définissant la notion de moyenne de Kolmogoroff sont : la symétrie, le point fixe, la croissance, la continuité et la substitution. La propriété de point fixe demande que $x +_p x = x$; or $x +_p x = 2^{1/p} x \neq x$ en général.

Définition de l'ensemble Pour $p \in \mathbb{R}$, on définit l'ensemble suivant. La justification sera donnée plus bas.

$$\mathbb{A}_p = \begin{cases} \mathbb{R}_+^* \cup \{0\} \cup \{xe^{i\pi/p}, x \in \mathbb{R}_+^*\} & \text{si } p > 0 \\ \mathbb{R}_+^* & \text{si } p = 0 \\ \mathbb{R}_+^* \cup \{+\infty\} \cup \{xe^{i\pi/p}, x \in \mathbb{R}_+^*\} & \text{si } p < 0 \end{cases}$$

Cet ensemble (voir la figure 1) est, aux variations près,

— la demi-droite ouverte des réels non nuls positifs

— union l'élément neutre de la loi interne

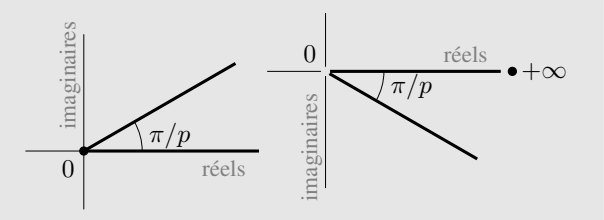


FIGURE 1 – À gauche, ensemble $\mathbb{A}_p$ pour p positif. L'élément neutre 0 est ajouté aux deux demi-droites ouvertes. À droite, ensemble $\mathbb{A}_p$ pour p négatif. L'élément neutre $+\infty$ est ajouté, mais 0 est exclu.

— union la demi-droite ouverte d'angle π/p si p est non-nul.

Définition de la loi interne On munit cet ensemble de la loi de composition interne $+_p$ définie comme dit plus haut.

$$\begin{array}{rll} +_p : & \mathbb{A}_p \times \mathbb{A}_p & \to \mathbb{A}_p \\ & (x, y) & \mapsto (x^p + y^p)^{1/p} \end{array}$$

Dans le cas où p est positif, $+_p$ correspond à la norme L_p. Il suffit de voir qu'un couple (x, y) de $\mathbb{A}_p \times \mathbb{A}_p$ est équivalent à un vecteur de deux dimensions $\vec{v} = \begin{pmatrix} x \\ y \end{pmatrix}$. On a alors l'égalité : $x +_p y = \|\vec{v}\|_p$.

On vérifie maintenant que l'ensemble $\mathbb{A}_p$ muni de $+_p$ est bien un groupe commutatif.

Loi de composition interne. Pour un élément x de $\mathbb{A}_p$, de deux choses l'une. Soit x est un réel positif sur la demi-droite des réels positifs, soit il est sur la demi-droite d'angle π/p. Dans le premier cas, la puissance p de x est un réel positif. Dans le deuxième cas, la puissance p de x est un réel négatif. En effet, si x est situé sur la demi-droite d'angle π/p, c'est un nombre complexe de la forme $xe^{i\pi/p}$ et sa puissance p est de la forme $x^p \times e^{i\pi}$. Maintenant, x étant réel positif et $e^{i\pi}$ étant égal à -1, $\left(xe^{i\pi/p}\right)^p = x^p e^{i\pi} = -(x^p)$ est un réel négatif.

La somme de deux nombres réels étant un nombre réel, pour x et y dans $\mathbb{A}_p$, $x^p + y^p$ est donc un nombre réel. S'il est positif, $(x^p + y^p)^{1/p}$ est un nombre réel. S'il est négatif, alors il est de la forme $\rho e^{i\pi}$ avec ρ réel positif, et $(x^p + y^p)^{1/p}$ est donc de la forme $\rho^{1/p} e^{i\pi/p}$, c'est-à-dire qu'il est situé sur la demi-droite d'angle π/p. Dans tous les cas, donc, la somme au sens de la loi de composition $+_p$ de deux éléments de $\mathbb{A}_p$ est dans $\mathbb{A}_p$. On en conclut que $+_p$ est une loi de composition interne dans $\mathbb{A}_p$.

Associativité de $+_p$. Pour p non nul on démontre l'associativité avec le développement suivant.

$$\begin{aligned}(x +_p y) +_p z &= \left(\left((x^p + y^p)^{1/p}\right)^p + z^p\right)^{1/p} \\ &= ((x^p + y^p) + z^p)^{1/p} \\ &= (x^p + (y^p + z^p))^{1/p} \\ &= \left(x^p + \left((y^p + z^p)^{1/p}\right)^p\right)^{1/p} \\ &= x +_p (y +_p z)\end{aligned}$$

Pour p nul, l'associativité est celle de la multiplication dans $\mathbb{C}$.
Elément neutre pour $+_p$. Pour p non nul positif, l'élément neutre est 0 car on a : $x +_p 0 = (x^p + 0^p)^{1/p} = (x^p)^{1/p} = x$. Et même chose pour $0 +_p x$ par commutativité de $+$.
Pour p non nul négatif, l'élément neutre est $+\infty$. En effet on a : $x +_p +\infty = (x^p + +\infty^p)^{1/p} = (x^p + 0)^{1/p} = x$. Et même chose pour $+\infty +_p x$ par commutativité de $+$.
Pour p nul, l'élément neutre est 1.
Elément opposé. L'élément opposé de tout x dans $\mathbb{R}$ est $xe^{i\pi/p}$. Tout nombre y sur la demi-droite d'angle π/p peut s'écrire comme $xe^{i\pi/p}$ et son élément opposé est alors x. En effet, on a, pour p positif :

$$\begin{aligned} x +_p xe^{i\pi/p} &= \left(x^p + x^p e^{i\pi}\right)^{1/p} \\ &= (x^p + x^p \times -1)^{1/p} \\ &= (x^p - x^p)^{1/p} \\ &= (0)^{1/p} \ = 0 \end{aligned}$$

On a aussi $xe^{i\pi/p} +_p x = 0$ par commutativité de $+$ dans $\mathbb{C}$.
Pour p négatif, le raisonnement est le même en remplaçant 0 par $+\infty$. Pour p nul, l'élément neutre est 1 et l'opposé de x est $1/x$. Ci-dessous, on notera $-_p x$ l'opposé de x pour $+_p$.
Commutativité. Celle de $+$ dans $\mathbb{C}$ implique trivialement que $x +_p y = y +_p x$ pour p non nul. Pour p nul, c'est la commutativité de la multiplication dans $\mathbb{C}$.
En résumé, $\mathbb{A}_p$ muni de $+_p$ est un groupe commutatif.

2.3.4 Analogie sur $(\mathbb{A}_p, +_p)$

La structure de groupe commutatif de $(\mathbb{A}_p, +_p)$ induit donc une analogie définie par : $\forall (a,b,c,d) \in (\mathbb{A}_p)^4$,

$$a : b ::_{+_p} c : d \ \Leftrightarrow \ a +_p (-_p b) = c +_p (-_p d)$$

Restreinte aux éléments de $\mathbb{R}_+$, cette définition devient :

$$a : b ::_{+_p} c : d \ \Leftrightarrow \ a +_p (be^{i\pi/p}) = c +_p (de^{i\pi/p})$$

Pour p non nul, on a les équivalences suivantes.

$$\begin{aligned} a : b ::_{+_p} c : d \ &\Leftrightarrow \ (a^p - b^p)^{1/p} = (c^p - d^p)^{1/p} \\ &\Leftrightarrow \ (a^p - b^p) = (c^p - d^p) \\ &\Leftrightarrow \ (a^p + d^p) = (b^p + c^p) \\ &\Leftrightarrow \ \left(\frac{1}{2}(a^p + d^p)\right)^{1/p} = \left(\frac{1}{2}(b^p + c^p)\right)^{1/p} \\ &\Leftrightarrow \ a : b ::^p c : d \end{aligned}$$

Pour p nul, on vérifie aussi $a : b ::_{+_0} c : d \Leftrightarrow a : b ::^0 c : d$.
On remarquera que la commutativité de la loi du groupe implique l'équivalence entre une analogie quelconque et celle sur les opposés des termes [19, 11].

$$a : b :: c : d \ \Leftrightarrow \ a^{-1} : b^{-1} :: c^{-1} : d^{-1}$$

Adapté à notre cas, et à partir de termes réels, cette équivalence énonce que l'analogie sur la demi-droite des réels positifs a son analogie correspondante sur la demi-droite d'angle π/p.

$$a : b ::^p c : d \ \Leftrightarrow \ ae^{i\pi/p} : be^{i\pi/p} ::^p ce^{i\pi/p} : de^{i\pi/p}$$

Enfin, tout groupe commutatif est aussi un magma commutatif. Le résultat précédent énonce que l'analogie du groupe commutatif est équivalente à l'analogie numérique ; l'alinéa 2.3.2 montrait que l'analogie du magma était l'analogie numérique ; on a donc l'équivalence de ces trois analogies sur les réels positifs non nuls. $\forall (a,b,c,d) \in \left(\mathbb{R}_+^*\right)^4$,

$$a : b ::_{m_p} c : d \ \Leftrightarrow \ a : b ::^p c : d \ \Leftrightarrow \ a : b ::_{+_p} c : d$$

3 L'analogie numérique étendue au cas où l'un des termes est égal à zéro

Les seuls groupes commutatifs $\mathbb{A}_p$ donnés plus haut qui contiennent 0 sont pour p strictement positif. On a alors : $\mathbb{A}_p = \mathbb{R}_+^* \cup \{0\} \cup \{xe^{i\pi/p}, x \in \mathbb{R}_+^*\}$. Il est donc possible d'y étendre l'analogie en p, strictement positif, à des analogies avec certains des termes égaux à zéros. Ci-dessous nous considérons les cas où les termes sont dans $\mathbb{R}_+$, car, dans les applications pratiques, les nombres manipulés sont généralement réels et pas complexes.

Théorème : analogie avec terme nul ($p > 0$)

Pour tout p strictement positif et tout couple de nombres réels positifs, éventuellement nuls, on a toujours une analogie en p entre, comme moyens, ces nombres et, comme extrêmes, 0 et leur moyenne généralisée en p fois la racine p-ième de 2.
$\forall p \in \mathbb{R}_+^*, \forall (b,c) \in (\mathbb{R}_+)^2$,

$$0 : b ::^p c : \left(2^{1/p} \times m_p(b,c)\right)$$

Démonstration : on vérifie simplement l'égalité entre les moyennes généralisées en p des extrêmes et des moyens.

$$\begin{aligned} 0^p + \left(2^{1/p}\left(\frac{1}{2}(b^p + c^p)\right)^{1/p}\right)^p \\ = \left(\left(2 \times \frac{1}{2}\right)^{1/p}(b^p + c^p)^{1/p}\right)^p \\ = b^p + c^p \qquad \square \end{aligned}$$

Cela permet d'établir que l'analogie en p entre 0 répété et un nombre réel positif quelconque, éventuellement non nul, répété, est valable quel que soit p positif et non nul.

$$\forall p \in \mathbb{R}_+^*, \ \forall d \in \mathbb{R}_+, \ 0 : 0 ::^p d : d$$

Noter que d peut être égal à 0.
On déduit aussi facilement que l'analogie ayant deux zéros comme extrêmes n'est possible que si les moyens sont égaux eux aussi à zéro.

$$\forall p \in \mathbb{R}_+^*, \ \forall (b,c) \in (\mathbb{R}_+)^2, \ 0 : b :: c : 0 \ \Rightarrow \ b = c = 0$$

Démonstration : trivialement, pour p positif non nul et b, c positifs, $b^p + c^p = 0 \;\Rightarrow\; b = c = 0$. □

Remarquons, que si l'on pose $(+\infty)^p = 0$ pour $p < 0$, on peut énoncer un théorème équivalent pour p négatif. Du point de vue algébrique, il s'agit toujours de l'analogie du groupe commutatif $(\mathbb{A}_p, +_p)$, cette fois-ci avec p négatif et donc $+\infty$ comme élément neutre.

Théorème : analogie avec terme infini ($p < 0$)

$$\forall p \in \mathbb{R}_-^*, \forall (b,c) \in (\mathbb{R}_+)^2,$$
$$\left(2^{1/p} \times m_p(b,c)\right) : b ::^p c : +\infty$$

4 Précision des bornes de la puissance analogique

Le précédent article [13] mentionnait rapidement comment, étant donné un quadruplet de nombres positifs, on peut déterminer la puissance p de l'analogie en p au moyen d'une recherche dichotomique partant de $-\infty$ et $+\infty$. Nous exhibons ci-dessous des bornes plus fines pour p.

4.1 Relation entre les termes du second rapport $c:d$

Pour une analogie en puissance p, avec p positif, les termes les plus grands, c et d par convention, entretiennent une relation d'ordre impliquant l'inverse de la puissance p. Insistons sur le fait que, ci-dessous, p est positif.

Lemme : termes du second rapport

$$\forall p \in \mathbb{R}_+, \forall (a,b,c,d) \in \left(\mathbb{R}_+^*\right)^4, a < b < c < d,$$
$$a : b ::^p c : d \;\Rightarrow\; \begin{cases} d \leq 2^{1/p} \times c \\ c \leq 2^{1/p} \times d \end{cases}$$

Démonstration : elle se fait en deux temps. Dans un premier temps, on exploite un encadrement de la moyenne généralisée en p pour p positif. Dans un deuxième temps, on applique la définition de l'analogie en puissance p qui énonce l'égalité des moyennes généralisées en p des extrêmes et des moyens.

Premier temps : on considère un p positif et deux nombres positifs non nuls tels que $0 < a < d$.

$$\begin{aligned}
& d^p \leq a^p + d^p \leq 2d^p \\
\Leftrightarrow\; & d^p \leq 2 \times \frac{1}{2}\left(a^p + d^p\right) \leq 2d^p \\
\Leftrightarrow\; & (d^p)^{1/p} \leq 2^{1/p} \times \left(\frac{1}{2}\left(a^p + d^p\right)\right)^{1/p} \leq 2^{1/p}(d^p)^{1/p} \\
\Leftrightarrow\; & d \leq 2^{1/p} \times \left(\frac{1}{2}\left(a^p + d^p\right)\right)^{1/p} \leq 2^{1/p} d \\
\Leftrightarrow\; & m_{+\infty}(a,d) \leq 2^{1/p} m_p(a,d) \leq 2^{1/p} m_{+\infty}(a,d) \\
\Leftrightarrow\; & (1/2)^{1/p}\, m_{+\infty}(a,d) \leq m_p(a,d) \leq m_{+\infty}(a,d)
\end{aligned}$$

La première séquence d'inégalités est vraie parce que p est positif, a est positif et d est plus grand que a. L'élévation à la puissance $1/p$ ne change pas l'ordre des inégalités car p est positif. [4]
On applique ce résultat à b et c en posant $0 < b < c$.

$$(1/2)^{1/p}\, m_{+\infty}(b,c) \leq m_p(b,c) \leq m_{+\infty}(b,c)$$

La prise des opposés inverse le sens des inégalités.

$$-m_{+\infty}(b,c) \leq -m_p(b,c) \leq -(1/2)^{1/p}\, m_{+\infty}(b,c)$$

On en dérive : $\forall p \in \mathbb{R}_+,\ \forall (a,b,c,d) \in \left(\mathbb{R}_+^*\right)^4,\ 0 < a < b < c < d$,

$$\begin{aligned}
& (1/2)^{1/p} m_{+\infty}(a,d) - m_{+\infty}(b,c) \\
& \quad \leq m_p(a,d) - m_p(b,c) \\
& \quad\quad \leq m_{+\infty}(a,d) - (1/2)^{1/p}\, m_{+\infty}(b,c)
\end{aligned}$$

Deuxième temps : on applique les inégalités précédentes à l'analogie en puissance p, qui énonce l'égalité des moyennes généralisées en p.
$\forall p \in \mathbb{R}_+,\ \forall (a,b,c,d) \in \left(\mathbb{R}_+^*\right)^4,\ 0 < a < b < c < d$,

$$\begin{aligned}
& a : b ::^p c : d \\
\Rightarrow\; & \left(\frac{1}{2}\right)^{1/p} d - c \leq \underbrace{m_p(a,d) - m_p(b,c)}_{} \leq d - \left(\frac{1}{2}\right)^{1/p} c \\
\Rightarrow\; & \left(\frac{1}{2}\right)^{1/p} d - c \leq 0 \leq d - \left(\frac{1}{2}\right)^{1/p} c \\
\Rightarrow\; & d - 2^{1/p} c \leq 0 \leq 2^{1/p} d - c \\
\Rightarrow\; & \begin{cases} d \leq 2^{1/p}\, c \\ c \leq 2^{1/p}\, d \end{cases}
\end{aligned}$$
□

4.2 Relation entre les termes du premier rapport $a:b$

On peut procéder de la même manière que précédemment directement. Mais en passant par les inverses, qui inversent

4. Il existe un résultat semblable bien connu pour la norme L_p :
$$\forall p \in \mathbb{R}_+,\ \forall \vec{x} \in \mathbb{R}^n,\ \ \|\vec{x}\|_{+\infty} \leq \|\vec{x}\|_p \leq n^{1/p}\, \|\vec{x}\|_{+\infty}.$$

le sens des inégalités, en laissant p positif et en prenant son opposé, on peut écrire :

$$m_{-\infty}(a,d) \leq m_{-p}(a,d) \leq 2^{1/p} m_{-\infty}(a,d)$$

Observez que le résultat final fait intervenir la moyenne généralisée en $-p$, avec p positif. Pour p négatif, on aura donc :

$$m_{-\infty}(a,d) \leq m_p(a,d) \leq \left(\frac{1}{2}\right)^{1/p} m_{-\infty}(a,d)$$

Les mêmes manipulations que plus haut pour $m_{+\infty}$ conduisent au résultat suivant, pour p négatif, attention, pas positif, en prenant garde que le facteur $2^{1/p}$ est placé différemment ici.
$\forall p \in \mathbb{R}_-,\ \forall (a,b,c,d) \in \left(\mathbb{R}_+^*\right)^4,\ 0 < a \leq b \leq c \leq d,$

$$\begin{aligned}
&a : b ::^p c : d \\
\Rightarrow\ & a - \left(\frac{1}{2}\right)^{1/p} b \leq \underbrace{m_p(a,d) - m_p(b,c)}_{} \leq \left(\frac{1}{2}\right)^{1/p} a - b \\
\Rightarrow\ & a - \left(\frac{1}{2}\right)^{1/p} b \leq 0 \leq \left(\frac{1}{2}\right)^{1/p} a - b \\
\Rightarrow\ & 2^{1/p} a - b \leq 0 \leq a - 2^{1/p} b \\
\Rightarrow\ & \begin{cases} 2^{1/p} \times a \leq b \\ 2^{1/p} \times b \leq a \end{cases}
\end{aligned}$$

On peut donc conclure pour le cas où p est négatif.

Lemme : termes du premier rapport

$$\forall (a,b,c,d) \in \left(\mathbb{R}_+^*\right)^4,\ \forall p \in \mathbb{R}_-,\ a < b < c < d,$$

$$a : b ::^p c : d \ \Rightarrow\ \begin{cases} 2^{1/p} \times a \leq b \\ 2^{1/p} \times b \leq a \end{cases}$$

4.3 Bornes pour la puissance analogique

Les lemmes précédents permettent de donner des bornes de la puissance d'une analogie donnée. Pour quatre nombres a, b, c et d, réels positifs, non nuls, différents deux à deux et triés par ordre croissant, on sait qu'il existe une analogie en p, $a : b ::^p c : d$.
Si p est positif, le lemme sur les termes du second rapport permet d'écrire :

$$\begin{aligned}
d \leq 2^{1/p} c \ &\Leftrightarrow\ \frac{d}{c} \leq 2^{1/p} \\
&\Leftrightarrow\ \log_2 \frac{d}{c} \leq 1/p \\
&\Leftrightarrow\ p \leq 1/\log_2 d/c
\end{aligned}$$

Le lemme sur les termes du premier rapport permet d'écrire, en se rappelant que p est négatif :

$$\begin{aligned}
2^{1/p} \times b \leq a \ &\Leftrightarrow\ 2^{1/p} \leq \frac{a}{b} \\
&\Leftrightarrow\ 1/p \leq \log_2 \frac{a}{b} \\
&\Leftrightarrow\ \frac{1}{\log_2 a/b} \leq p
\end{aligned}$$

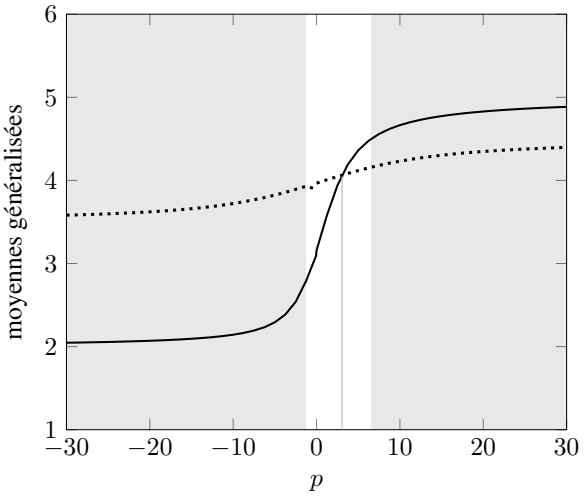


FIGURE 2 – En plein, moyennes généralisées pour $a = 2$ et $d = 5$, avec p en abscisse. En pointillé, même chose pour $b = 3{,}5$ et $c = 4{,}5$. La puissance p de l'analogie $a : b ::^p c : d$ est donnée par l'intersection des courbes pleine et pointillée. En l'occurrence $p \simeq 3{,}06$. La zone en clair, entre les bornes $-1{,}24$ et $6{,}58$, indique l'intervalle à explorer pour trouver cette puissance.

Noter que comme $a < b$, $\log_2 a/b$ est négatif car a/b est inférieur à 1.
En combinant les deux inégalités, on a le théorème ci-dessous. Il peut être étendu pour inclure le cas où l'un des termes est nul. On peut en effet démontrer que quand $a = 0$, a doit être remplacé par d dans la borne inférieure.

Théorème : bornes de la puissance analogique

$$\forall p \in \mathbb{R}, \forall (a,b,c,d) \in \left(\mathbb{R}_+^*\right)^4, 0 < a < b < c < d,$$

$$\begin{cases} a : b ::^p c : d \ \Rightarrow\ \dfrac{1}{\log_2 a/b} \leq p \leq \dfrac{1}{\log_2 d/c} \\ 0 : b ::^p c : d \ \Rightarrow\ \dfrac{1}{\log_2 d/b} \leq p \leq \dfrac{1}{\log_2 d/c} \end{cases}$$

La figure 2 illustre l'encadrement de la puissance analogique entre les bornes calculées par ce théorème, pour une analogie particulière.

4.4 Calcul de p en pratique : algorithme

Pour déterminer en pratique la puissance p de l'analogie entre quatre nombres a, b, c et d, on pourra donc partir des bornes données dans le théorème précédent et effectuer une recherche par dichotomie comme décrite dans l'algorithme de la figure 3. Les cas particuliers où certains des termes sont égaux, et qui mènent à des puissances infinies ou couvrant tout $\mathbb{R}$, comme détaillé dans l'article précédent [13] n'ont pas besoin de ces bornes pour être traitées. Aussi, avant de lancer cette recherche, on peut isoler le cas des

```
def puissance_analogique(a, b, c, d):
    # Initialisation des bornes de p.
    ϵ = 0.0001
    if a == 0: n = d else: n = a
    g = ln 2/ln(n/b) − ϵ # borne à gauche
    d = ln 2/ln(d/c) + ϵ # borne à droite
    # Différences des moy. aux bornes.
    Δ_g = m_g(a,d) −m_g(b,c)
    Δ_d = m_d(a,d) −m_d(b,c)
    # Recherche dichotomique.
    while not (Δ_g ≃ 0.0 or Δ_d ≃ 0.0):
        μ = (g+d)/2 #milieu des bornes
        Δ_μ = m_μ(a,d) − m_μ(b,c)
        if 0 < Δ_μ:
            g, Δ_d = μ, Δ_μ
        else:
            d, Δ_g = μ, Δ_μ
    # Sortie de la puissance.
    if |Δ_g| < |Δ_d|:
        return g
    else:
        return d
```

FIGURE 3 – Algorithme de recherche dichotomique de la puissance d'une analogie. Il s'agit de déterminer le p de valeur $\Delta_p = m_p(a,d) - m_p(b,c)$ la plus plus proche possible de 0 en partant des bornes théoriques g et d. La fonction $\simeq$ décide de l'arrêt. Les termes a, b, c et d doivent être rangés par ordre croissant.

analogies arithmétiques ($p = 1$), géométriques ($p = 0$) ou harmoniques ($p = -1$). C'est ce que nous faisons dans nos implémentations. [5]

Nous avons implémenté l'algorithme de la figure 3 dans deux langages de programmation, Python et C. Pour la version en C, nous avons utilisé deux méthodes, récursive et itérative. Le tableau 1 donne les temps d'exécution de ces différentes implémentations sur la même liste d'un million de quadruplets tirés au hasard. Ces quadruplets contiennent des valeurs réelles positives comprises entre 0 et 100 ; en effet, un terme peut être égal à 0 grâce aux développements donnés dans le présent article.

Les temps d'exécution observés disent l'efficacité des versions en C, langage compilé, par rapport à la version en Python, langage interprété, requérant plus de vérifications, et donc plus lent. Elles sont presque 400 fois plus rapides et permettent le calcul d'un million de puissances analogiques en moins de trois secondes. Contre toute attente et sans que nous n'ayons aucune explication à cela, la version récursive en C semble légèrement plus rapide que la version itérative.

5. Sous http://lepage-lab.ips.waseda.ac.jp/projects/Kakenhi_Project_21K12038/, ouvrir l'onglet « Experimental Results », voir en bas de page : « num_nlg Python package ».

Langage	Méthode	Temps (s)	Nbr. de pas en moy.
Python	itérative	1782,58	19,1±4,0
C	récursive	2,68	idem
	itérative	2,71	idem

TABLEAU 1 – Temps d'exécution pour le calcul de la puissance analogique pour le même million de quadruplets tirés au hasard. Comparaison entre deux langages de programmation et deux méthodes, récursive et itérative.

5 Agrandissements et rétrécissements

L'extension de l'analogie numérique au cas où l'un des termes est égal à 0 introduisait la moyenne généralisée en p des moyens directement dans l'analogie. Dans ce paragraphe, nous examinons plus en détail les relations existant entre les différentes moyennes que l'on peut prendre entre les termes, et explorons la création d'analogies nouvelles à partir d'une analogie en p donnée.

5.1 Egalité des moyennes de tous les termes, des moyennes des extrêmes et des moyennes des moyens

Il est facile de montrer que, pour une analogie donnée, les trois moyennes suivantes :

— de tous les termes,
— des extrêmes et
— des moyens

sont égales. $\forall (a,b,c,d) \in (\mathbb{R}^*_+)^4$,

$$a : b ::^p c : d \;\Rightarrow\; m_p(a,b,c,d) = m_p(a,d) = m_p(b,c)$$

L'égalité $m_p(a,d) = m_p(b,c)$ provient trivialement de la définition de l'analogie. L'égalité avec $m_p(a,b,c,d)$ se déduit de deux propriétés générales des moyennes.
La première propriété est appelée partitionnement pour les moyennes de Kolmogoroff. Elle énonce que l'on peut décomposer une moyenne en moyenne de moyennes de paquets d'éléments de même taille. Autrement dit :

$$\begin{aligned} m(x_1, x_2, \ldots, x_{m\times n}) = m\big(&m(x_1, \ldots, x_n),\\ &m(x_{n+1}, \ldots, x_{2n}),\\ &\vdots\\ &m(x_{((m-1)\times n)+1}, \ldots, x_{m\times n})\big) \end{aligned}$$

La deuxième propriété est une propriété de point fixe vérifiée par les moyennes de Kolmogoroff. La moyenne de la même valeur répétée est égale à cette valeur. Nous l'appliquons en répétant la moyenne de plusieurs nombres : $m(x_1, \ldots, x_n) = m\left(m(x_1, \ldots, x_n), \ldots, m(x_1, \ldots, x_n)\right)$.
On applique les propriétés précédentes aux quatre termes de l'analogie en décomposant en paquets de deux judicieusement choisis, le paquet des deux moyens et le paquet des

deux extrêmes.

$$\begin{aligned} m_p(a,b,c,d) &= m_p(m_p(a,d), m_p(b,c)) \\ &= m_p(m_p(a,d), m_p(a,d)) \\ &= m_p(a,d) \end{aligned}$$

De même pour $m_p(b,c)$.

5.2 Rétrécissement d'analogie aux moyennes des termes des rapports

5.2.1 Rétrécissement vertical

Étant donnée une analogie en puissance p ($p \neq \pm\infty$), on peut former une nouvelle analogie de même puissance dont les deux conséquents (termes b et d) sont les moyennes des termes du premier et du deuxième rapport. La visualisation donnée plus bas dans la figure 4 explique pourquoi nous appelons ce résultat un rétrécissement vertical.
$\forall (a,b,c,d) \in (\mathbb{R}^*{}_+)^4$,

$$a : b ::^p c : d \;\Rightarrow\; a : m_p(a,b) ::^p c : m_p(c,d)$$

Démonstration pour le cas $p \neq 0$:

$$\begin{aligned} m_p(a, m_p(c,d)) &= \left(\frac{1}{2}\left(a^p + \left(\frac{1}{2}(c^p + d^p)\right)^{(1/p)\times p}\right)\right)^{1/p} \\ &= \left(\frac{1}{2}(a^p + \frac{1}{2}c^p + \frac{1}{2}d^p)\right)^{1/p} \end{aligned}$$

$$\begin{aligned} m_p(m_p(a,b),c) &= \left(\frac{1}{2}(\frac{1}{2}a^p + \frac{1}{2}b^p + c^p)\right)^{1/p} \\ &= \left(\frac{1}{2}(a^p - \frac{1}{2}a^p + \frac{1}{2}b^p + \frac{1}{2}c^p + \frac{1}{2}c^p)\right)^{1/p} \\ &= \left(\frac{1}{2}(a^p - \frac{1}{2}a^p + \frac{1}{2}a^p + \frac{1}{2}d^p + \frac{1}{2}c^p)\right)^{1/p} \\ &= \left(\frac{1}{2}(a^p + \frac{1}{2}c^p + \frac{1}{2}d^p)\right)^{1/p} \end{aligned}$$

On a pu remplacer $\frac{1}{2}b^p + \frac{1}{2}c^p$ par $\frac{1}{2}a^p + \frac{1}{2}d^p$ ci-dessus car on suppose l'analogie $a : b ::^p c : d$. On constate donc que $m_p(a, m_p(c,d)) = m_p(m_p(a,b),c)$, c'est-à-dire $a : m_p(a,b) ::^p c : m_p(c,d)$. □

Démonstration pour le cas $p = 0$:

$$\begin{aligned} m_0(a, m_0(c,d))^2 &= \sqrt{a}\sqrt{c}\sqrt{b}\sqrt{c} \\ &= a \times \sqrt{cd} &&= \sqrt{a}\sqrt{b}\sqrt{c}\sqrt{c} \\ &= \sqrt{a}\sqrt{a}\sqrt{c}\sqrt{d} &&= \sqrt{ab} \times c \\ &= \sqrt{a}\sqrt{c}\sqrt{a}\sqrt{d} \nearrow &&= m_0(m_0(a,b),c)^2 \end{aligned}$$

car, l'analogie entre a, b, c et d étant supposée, on a l'égalité $\sqrt{ad} = \sqrt{bc}$. □

5.2.2 Généralisation : rétrécissements vertical et horizontal

Les huit formes équivalentes de l'analogie (voir Introduction) permettent de généraliser le résultat précédent. Il suffit de l'appliquer à ces huit formes équivalentes. Par manque de place, nous abrégeons ci-dessous.

$\forall (a,b,c,d) \in (\mathbb{R}^*{}_+)^4$,

$$a : b ::^p c : d \Rightarrow \begin{cases} a : b ::^p c : d \\ a : c ::^p b : d \\ \quad\vdots \\ d : c ::^p b : a \end{cases}$$

$$\Rightarrow \begin{cases} a : m_p(a,b) ::^p c : m_p(c,d) \\ a : m_p(a,c) ::^p b : m_p(b,d) \\ \quad\vdots \\ d : m_p(c,d) ::^p b : m_p(a,b) \end{cases}$$

L'énumération complète permet de constater que l'on a au total seulement quatre formes. Les deux formes $a : m_p(a,b) ::^p c : m_p(c,d)$ et $m_p(a,b) : b ::^p m_p(c,d) : d$ peuvent être considérées comme des rétrécissements verticaux, et les deux formes $a : m_p(a,c) ::^p b : m_p(b,d)$ et $m_p(a,c) : c ::^p m_p(b,d) : d$ comme des rétrécissements horizontaux de l'analogie initiale. Cette dénomination provient de la visualisation trompeuse de la figure 4 ; trompeuse car nous éclatons sur deux dimensions ce qui n'existe que sur une seule. Mais cette vue est conforme à la représentation intuitive de l'analogie comme carré ou rectangle.

5.2.3 Rétrécissement au demi-carré

On peut poursuivre en appliquant le résultat précédent deux fois : une fois verticalement et une autre fois horizontalement. On constate alors que l'on peut rétrécir un carré analogique en un carré homologue selon les moyennes de même puissance. $\forall (a,b,c,d) \in (\mathbb{R}^*{}_+)^4$, $a : b ::^p c : d \Rightarrow$

$$a : m_p(a,b) ::^p m_p(a,c) : m_p(a,b,c,d)$$

Démonstration pour $p \neq 0$: On développe chacune des moyennes, celle des extrêmes et celle des moyens, élevées à la puissance p sans le facteur $1/2$.

$$\begin{aligned} 2 \times m_p(a, &m_p(a,b,,c,d))^p \\ &= \left(a^p + \left(\frac{1}{4}(a^p + b^p + c^p + d^p)\right)^{(1/p)\times p}\right) \\ &= a^p + \frac{1}{2}b^p + \frac{1}{2}c^p \end{aligned}$$

$$\begin{aligned} 2 \times m_p(m_p(a,b), m_p(a,c))^p \\ &= \left(\frac{1}{2}(a^p + c^p)\right)^{(1/p)\times p} + \left(\frac{1}{2}(a^p + b^p)\right)^{(1/p)\times p} \\ &= a^p + \frac{1}{2}b^p + \frac{1}{2}c^p \end{aligned}$$

Supposer une analogie de puissance p entre a, b, c et d permet d'utiliser l'égalité $a^p + d^p = b^p + c^p$ ci-dessus. Constater l'égalité des deux termes conclut la démonstration pour $p \neq 0$. □

Démonstration pour $p = 0$:

$$\begin{aligned} m_0(a,\, m_0(a,b,c,d)) &= \sqrt{a\sqrt{bc}} \\ &= \sqrt{a\sqrt[4]{abcd}} &&= \sqrt{\sqrt{ab}\sqrt{ac}} \\ &= \sqrt{a\sqrt[4]{(bc)^2}} \nearrow &&= m_0(m_0(a,b), m_0(a,c)) \end{aligned}$$

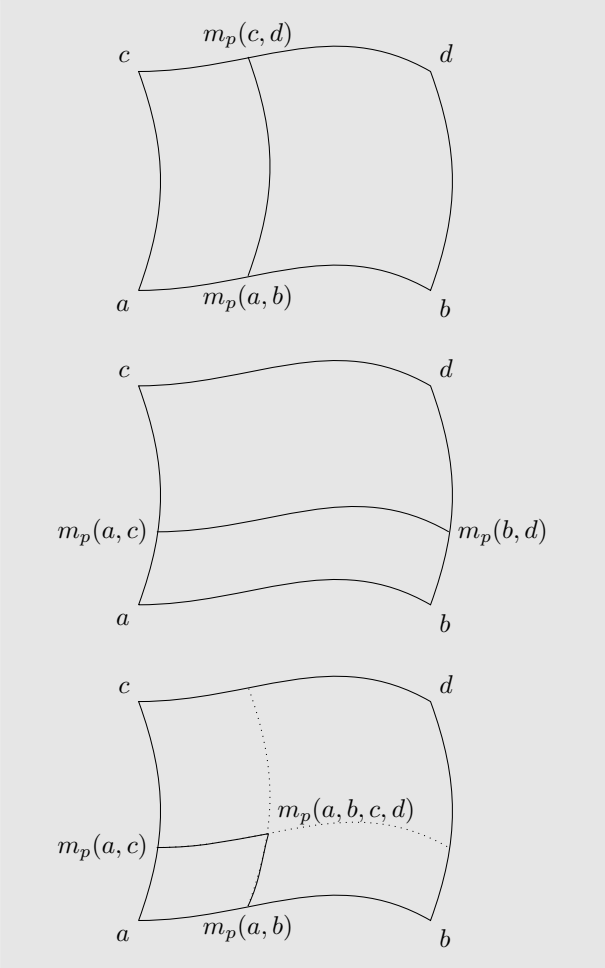


FIGURE 4 – En haut, visualisation du rétrécissement horizontal ; au milieu, du rétrécissement vertical ; en bas, du rétrécissement au demi-carré.

car, l'analogie de puissance 0 entre a, b, c et d étant supposée, on a l'égalité $\sqrt{ad} = \sqrt{bc}$, soit $ad = bc$. □

Visualisation : La visualisation du rétrécissement au demi-carré est donnée dans la figure 4.

5.2.4 Itération

On peut envisager à partir d'une analogie donnée, d'itérer le rétrécissement vertical en passant aux moyennes, puis aux moyennes des moyennes, etc. pour produire une série d'analogies de même puissance.
En appliquant la même opération de moyennage itérativement, les termes moyennés convergent vers les termes non moyennés originaux et donc vers une analogie dégénérée du type $a : a :: c : c$ ou $a : b :: a : b$.
Posons :

$$\begin{aligned} M^1(a,b) &= m_p(a,b) \\ M^2(a,b) &= m_p(a, m_p(a,b)) = m_p(a, M^1(a,b)) \\ &\vdots \\ M^{n+1}(a,b) &= m_p(a, M^n(a,b)) \end{aligned}$$

et des définitions similaires pour $M^n(a,b,c,d)$. On a $\lim_{n\to+\infty} M^n(a,b) = a$. En passant à la limite pour l'analogie, on a alors les trois résultats suivants.

$$\begin{aligned} \lim_{n\to+\infty} \; a : M^n(a,b) ::^p c : M^n(c,d) \; &= \; a : a :: c : c \\ \lim_{n\to+\infty} \; a : b ::^p M^n(a,c) : M^n(b,d) \; &= \; a : b :: a : b \\ \lim_{n\to+\infty} \; a : M^n(a,b) ::^p M^n(a,c) : M^n(a,b,c,d) \; &= \; a : a :: a : a \end{aligned}$$

5.3 Agrandissements

Réciproquement au rétrécissement, on peut établir des propriétés d'agrandissement d'une analogie en puissance p selon les trois directions vues précédemment : verticale, horizontale ou selon les deux à la fois pour le demi-carré. Pour l'agrandissement cela sera un double du carré.

Cela provient du fait que l'on peut résoudre une équation en moyenne : étant donné a et un nombre m donné, il est possible de trouver b tel que $m_p(a,b) = m$ (p est donné aussi). C'est le nombre tel que $b = (m^p - a^p)^{1/p}$ pour $p \neq 0$ et $b = m^2/a$ pour $p = 0$. On voit que a et m doivent vérifier certaines conditions pour que b existe.

6 Conclusion

Revenant sur un précédent article qui définissait l'analogie numérique et montrait qu'à un réordonnancement près il existe une analogie pour tout quadruplet de nombres réels positifs non nuls, nous avons *revisité* cette définition avec des moyens algébriques. Cela nous a permis d'*étendre* à des cas qui n'étaient pas traités précédemment, ceux où l'un des termes est nul.

L'existence d'une analogie équivaut à l'existence d'une puissance analogique. Nous en avons *précisé* les bornes et avons donné un algorithme pour son calcul. Les temps d'exécution obtenus sont raisonnablement courts.

Enfin, l'utilisation explicite de la moyenne généralisée dans les termes d'une analogie nous a permis de produire une infinité d'analogies numériques nouvelles à partir d'une analogie numérique donnée, soit par *rétrécissement*, soit par *agrandissement.*

Cet article a aussi illustré le fait qu'il est possible d'aborder l'étude de l'analogie par différents angles. On peut adopter un point de vue axiomatique [10] comme dans l'introduction de cet article. On peut définir l'analogie en adoptant un point de vue de théorie des modèles [2], comme dans nos développements induisant l'analogie numérique à partir de structures algébriques. On peut encore s'intéresser aux méthodes pratiques pour la détermination des analogies [16, 1, 6] ; notre proposition d'un algorithme de détermination de la puissance analogique va dans ce sens.

La formalisation de l'analogie numérique ouvre la voie à son application dans les modèles d'intelligence artificielle modernes qui travaillent essentiellement sur des représentations vectorielles ou tensorielles.

Références

[1] S. Alsaidi, A. Decker, Puthineath Lay, E. Marquer, P.-A. Murena, and Miguel Couceiro. A neural approach for detecting morphological analogies. In *DSAA*, pages 1–10, Porto, 2021.

[2] C. Antić. Analogical proportions. *Annals Math. and AI*, 90(6) :595–644, 2022.

[3] C. Antić. Boolean proportions. *Log. meth. comp. sci.*, 20(2) :2–20, 2024.

[4] Aristote. *Poétique*. Gallimard, collection *tel*, Paris, 1996. Trad. J. Hardy.

[5] Aristote. *Éthique à Nicomaque*. Librairie philosophique J. Vrin, Paris, [1er tirage 1990] edition, 1997. Trad. J. Tricot.

[6] K. Chan, S. P. Kaszefski-Yaschuk, C. Saran, E. Marquer, and M. Couceiro. Solving morphological analogies through generation. In M. Couceiro and P.-A. Murena, editors, *IARML*, 2022.

[7] Euclide. *Les quinze livres des éléments géométriques d'Euclide*. I. Dédin, Paris, 1632.

[8] Ю. А. Шрейдер. Равенство, сходство, порядок (Égalité, similitude, ordre). Наука, Москва, 1975.

[9] O. L. Hölder. Ueber einen Mittelwerthssatz. *Nach. könig. G. der Wiss. zu Göttingen*, 1889(2) :38–47, 1889.

[10] Y. Lepage. *De l'analogie rendant compte de la commutation en linguistique*. mém. d'hab., univ. Grenoble, 2003.

[11] Y. Lepage. Cahiers d'analogie (2). Notes de cours, niveau maîtrise, univ. Waseda, 2018–2025.

[12] Y. Lepage. Formulae for the solution of an analogical equation between Booleans using the Sheffer stroke (NAND) or the Pierce arrow (NOR). In M. Couceiro, P.-A. Murena, and S. Afantenos, editors, *IARML*, pages 3–14, 2023.

[13] Y. Lepage and M. Couceiro. Analogie et moyenne généralisée. In J.-G. Mailly, F. Schwarzentruber, and A. Wilczynski, editors, *PFIA-JIAF*, pages 114–124, La Rochelle, 2024.

[14] F. Luino and R. G. Boscovich. *Delle progressioni e serie*. Giuseppe Galeazzi, Milano, 1767.

[15] L. Miclet and H. Prade. Handling analogical proportions in classical logic and fuzzy logics settings. In C. Sossai and G. Chemello, editors, *Symb. and Quant. Appr. to Reasoning with Uncertainty*, pages 638–650, Berlin, 2009. Springer.

[16] P.-A. Murena, M. Al-Ghossein, J.-L. Dessalles, and A. Cornuéjols. Solving analogies on words based on minimal complexity transformation. In C. Bessière, editor, *IJCAI*, pages 1848–1854, 2020.

[17] J. Ozanam. *Les elemens d'Euclide, expliquez*. Claude Jombert, Paris, 1711.

[18] J. J. Rallier des Ourmes. Proportion. In *Encyclopédie*. Briasson, Paris, 1751–1772.

[19] N. Stroppa. *Définitions et caractérisation de modèles à base d'analogies pour l'apprentissage automatique des langues naturelles*. Thèse, ENST, 2005.

Proportions analogiques entre probabilités

Henri Prade[1] Gilles Richard[1],

[1] IRIT, CNRS & Université Toulouse III - Paul Sabatier, 118 route de Narbonne, 31062 Toulouse cedex 9, France

{henri.prade, gilles.richard}@irit.fr

Résumé

Les proportions analogiques sont des relations qui lient 4 items a, b, c et d et qui s'énoncent "a est à b comme c est à d". Ces 4 items sont souvent décrits par des vecteurs de valeurs booléennes, nominales, ou numériques. Les proportions analogiques peuvent aussi mettre en relation des formules logiques. L'article propose une première étude des proportions analogiques entre probabilités, qu'elles soient simplement entre des valeurs, ou entre des distributions (ce qui exige la préservation de leur normalisation). Les propriétés de définitions à base de proportion arithmétique, ou combinant cette dernière avec la proportion géométrique sont étudiées, et des utilisations potentielles décrites. Une annexe propose une preuve du théorème de Pythagore en termes de proportions géométriques.

Mots-clés

proportion analogique, probabilité, proportion numérique

Abstract

Analogical proportions are relations that link 4 items a, b, c and d and that are expressed as "a is to b as c is to d". These 4 items are often described by vectors of Boolean, nominal, or numerical values. Analogous proportions can also relate logical formulas. The article proposes a first study of analogous proportions between probabilities, whether they are simply between values, or between distributions (which requires the preservation of their normalization). The properties of definitions based on arithmetic proportion, or combining the latter with geometric proportion are studied, and potential uses are described. An appendix proposes a proof of the Pythagorean theorem in terms of geometric proportions.

Keywords

analogical proportion, probability, numerical proportion.

1 Introduction

Les analogies et les probabilités ne sont pas souvent considérées conjointement, bien qu'elles soient toutes deux liées à l'induction [15]. De fait déjà dans les années 1930, Janina Hosiasson-Lindenbaum [18], une philosophe, spécialiste du raisonnement probabiliste et logicienne, considérait le raisonnement analogique suivant : si deux conjectures f_1 et f_2 sont conséquences d'une même hypothèse h, l'observation de f_1 (qui devient donc un fait) induit h et augmente la croyance en la conjecture f_2. Elle cherchait des hypothèses additionnelles lui permettant une justification probabiliste de cette augmentation de croyance. Dans le même esprit, plus tard, Polya [11] considérera que f_1 et f_2 sont analogues s'ils sont impliqués par une hypothèse commune.

Dans cet article on s'intéresse à des analogies exprimées sous forme de proportions analogiques[1] c'est-à-dire des énoncés de la forme "a est à b comme c est à d". Tout comme l'inférence probabiliste bayésienne, l'inférence basée sur les proportions analogiques permet de faire de la classification avec succès [4, 2]. En classification, a, b, c, d sont des vecteurs de valeurs d'attributs décrivant les items à classer. Les attributs peuvent être booléens, nominaux, ou numériques. Cependant, les proportions analogiques peuvent être étendues à des cadres de représentation généraux comme la logique propositionnnelle [16]. Dans cet article, nous considérons le cadre probabiliste.

Cet article comporte quatre sections principales et une annexe. La section 2 présente les rappels nécessaires sur d'une part les proportions analogiques booléennes et d'autre part les proportions analogiques numériques. Ces dernières peuvent être envisagées soit sur le modèle des proportions arithmétiques, soit sur le modèle des proportions géométriques. La section 3 s'intéresse aux proportions analogiques entre des valeurs de probabilité : ces valeurs représentent les probabilités qu'un item ait des valeurs particulières d'attributs. La section 4 considère la possibilité de définir des proportions analogiques entre quatre distributions de probabilité. Pour ce faire deux définitions sont plus particulièrement considérées, l'une à base de proportion arithmétique, l'autre requiert à la fois la satisfaction de proportions arithmétiques et la satisfaction de proportions géométriques. Cette seconde définition est compatible avec une conservation de la divergence de Kullback-Leibler entre les paires de distributions. La section 5 discute des applications potentielles de proportions analogiques entre probabilités. Une annexe fournit une illustration du pouvoir des

1. Ces deux vues de l'analogie sont assez différentes. Cependant, si a et b d'une part, et c et d d'autre part sont "analogues", on peut arguer qu'une proportion analogique "a est à b comme c est à d" doit tenir. C'est ce qui est montré dans [12] où "a analogue à b" est modélisé, de façon un peu différente de Polya, par les relations de conséquence non monotone "si a généralement b" et "si b généralement a".

proportions géométriques en proposant une preuve élégante du théorème de Pythagore.

2 Rappels

Une proportion analogique "a est à b comme c est à d", notée $a : b :: c : d$, est censée satisfaire trois postulats :

- — *réflexivité* : $a : b :: a : b$;
- — *symétrie* : $a : b :: c : d \Rightarrow c : d :: a : b$;
- — stabilité par *permutation centrale* : $a : b :: c : d \Rightarrow a : c :: b : d$.

Quand les items sont représentés de manière vectorielle, c'est-à-dire que $a = (a_1, \cdots, a_n)$, $b = (b_1, \cdots, b_n)$, $c = (c_1, \cdots, c_n)$, et $d = (d_1, \cdots, d_n)$, la proportion analogique $a : b :: c : d$ est définie composante par composante :

Définition 1 $a : b :: c : d$ *ssi* $\forall i = 1, \cdots, n,\ a_i : b_i :: c_i : d_i$.

Nous considérons successivement les cas où les composantes i correspondent à des variables booléennes, nominales, puis numériques. Dans ces sous-sections, on omettra les indices i pour alléger la notation, étant entendu qu'on ne s'intéresse qu'à une seule composante.

2.1 Proportion analogique booléenne

Le modèle booléen minimal des 3 postulats précédents est donné dans la table 1, ou l'on présente les valuations de a, b, c, d que doit contenir tout modèle des 3 postulats. Dans

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 – Table de vérité de $a : b :: c : d$

ce modèle minimal, les 10 autres valuations ne sont pas valides. Une expression logique qui n'est vraie que pour les quadruplets de la table ci-dessus est donnée par la formule [9] :

$$a : b :: c : d = [(a \wedge \neg b) \equiv (c \wedge \neg d)] \wedge [(\neg a \wedge b) \equiv (\neg c \wedge d)] \quad (1)$$

Cette formule exprime précisément que "a diffère de b comme c diffère de d et b diffère de a comme d diffère de c" (et "lorsque a et b ne diffèrent pas, c et d ne diffèrent pas").

L'inférence analogique est basée sur la recherche de d connaissant a, b, et c et s'appuie sur le modèle minimal. L'examen de la table montre que s'il existe, d est unique, mais qu'il n'existe pas toujours de x tel que $a : b :: c : x$ soit vrai. Par exemple, il n'y a pas de solution dans $\{0, 1\}$ pour $1 : 0 :: 0 : x$ et $0 : 1 :: 1 : x$.

La formule (1) s'applique aussi à des variables booléennes représentant des formules propositionnelles quelconques. Elle indique que pour que la proportion $a : b :: c : d$ soit satisfaite il faut que $(a \wedge \neg b) \equiv (c \wedge \neg d)$ et que $(\neg a \wedge b) \equiv (\neg c \wedge d)$. Prenons l'exemple d'une proportion analogique qui apparaît dans [13] et qui est également valable pour toute structure de treillis [1] :

$$p \vee q : p :: q : p \wedge q$$

On peut facilement vérifier que $(p \vee q) \wedge \neg p \equiv q \wedge \neg(p \wedge q) \equiv \neg p \wedge q$ et $p \wedge \neg(p \vee q) \equiv (p \wedge q) \wedge \neg q \equiv \bot$.
Une formule propositionnelle à n variables peut être décrite sur les 2^n interprétations du langage considéré. Par exemple, si nous avons deux variables booléennes p et q, nous avons les interprétations pq, $p\neg q$, $\neg pq$, $\neg p \neg q$. En les prenant dans cet ordre, p est codé par 1100, $p \wedge q$ par 1000, $p \vee q$ par 1110, q par 1010, etc.
On peut montrer que si ces 4 formules logiques a, b, c, d, mettent en jeu un ensemble de n variables, et qu'elles sont donc représentables par des chaînes de bits de taille 2^n, cela revient à avoir une proportion analogique $a_i : b_i :: c_i : d_i$ sur chaque composante i des chaînes de bits qui représentent les formules, c'est-à-dire pour chaque interprétation possible [16] (notons que i fait ici référence à une interprétation, et non à une variable booléenne comme dans la définition 1).
Effectivement dans l'exemple ci-dessus on a bien

$$1110 : 1100 :: 1010 : 1000$$

composante par composante.

Etant donné trois formules propositionnelles, on peut trouver, si elle existe, la formule propositionnelle formant une proportion analogique avec ces trois formules. On peut aussi vérifier si quatre formules propositionnelles forment ou non un proportion analogique [16].

2.2 Proportions analogiques nominales

La description des items peut impliquer des attributs nominaux, c'est-à-dire des attributs avec un domaine fini dont la cardinalité peut être supérieure à 2.
Observons que la Table 1 présente trois types de motifs, à savoir $ssss$, $sstt$ et $stst$ (où $s, t \in \{0, 1\}$) ; les deux derniers motifs sont échangés par permutation centrale.
L'analogie booléenne s'étend alors facilement à un attribut nominal $\mathcal{A}$ prennant ses valeurs dans un domaine fini $\mathcal{D}_\mathcal{A}$. $a : b :: c : d$ est vrai pour une variable nominale si et seulement si (comme suggéré pour la première fois dans [10]) :

$$(a, b, c, d) \in \{(s, s, s, s), (s, t, s, t), (s, s, t, t) | s, t \in \mathcal{D}_\mathcal{A}\} \quad (2)$$

où s, t sont des valeurs distinctes quelconques du domaine de l'attribut $\mathcal{A}$. La condition (2) généralise clairement le cas booléen. Elle garantit la satisfaction des postulats des proportions analogiques.

2.3 Proportions analogiques numériques

Les proportions analogiques ont été imaginées par Aristote sur le modèle des proportions numériques étudiées dans la génération d'avant par Archytas de Tarente et Eudoxe de Cnide.
Parmi les proportions numériques, deux proportions qui mettent en jeu les quatre opérations élémentaires ont une place importante (a, b, c, d sont ici des nombres réels) :

— la proportion arithmétique $a - b = c - d$
— la proportion géométrique $\frac{a}{b} = \frac{c}{d}$

On prendra la convention $\frac{0}{0} = 1$ considérant que $\lim_{x\to 0} \frac{x}{x} = 1$.
Il est facile de vérifier que ces deux proportions sont bien des proportions analogiques puisqu'elles satisfont les trois postulats rappelés au début de cette section. Comme dans le cas booléen, elles expriment bien l'identité des résultats de comparaisons de a et b, et de c et d soit en termes de différence soit en termes de ratios. On peut aussi vérifier l'accord avec la table de verité (Table 1) : pour les 6 lignes de la table, les égalités des proportions numériques sont satisfaites (on prend $\frac{1}{0} = +\infty$) et il n'y a pas d'égalités pour les autres quadruplets possibles. A la différence du cas booléen, le cadre numérique offre des solutions pour les proportions analogiques *continues* qui sont de la forme $a : b :: b : c$. En effet, elles conduisent à la définition de la moyenne arithmétique ($b = \frac{a+c}{2}$) et de la moyenne géométrique ($b = \sqrt{ac}$).
Ces proportions peuvent aussi s'écrire en combinant les extrêmes (a et d) et les moyens (b et c) :

— $a + d = b + c$ (proportion arithmétique)
— $a \cdot d = b \cdot c$ (proportion géométrique)

Cependant pour la proportion géométrique, cela permet d'avoir $0 \cdot 1 = 0 \cdot 0$ alors que $0 : 0 :: 0 : 1$ n'est certainement pas une proportion analogique. On voit donc que l'équivalence entre les formes "quotient" et "produit" exclut 0.
Ces 2 proportions s'échangent par transformation *logarithmique / exponentielle* :
si $\frac{a}{b} = \frac{c}{d}$ alors nous avons $\ln(a) - \ln(b) = \ln(c) - \ln(d)$,
si $a - b = c - d$ alors nous avons $\frac{e^a}{e^b} = \frac{e^c}{e^d}$.
Rappelons enfin que la notation $a : b :: c : d$ a été utilisée (au moins) jusqu'à Gaspard Monge [14] pour signifier $\frac{a}{b} = \frac{c}{d}$ (la notation ' : :' a été introduite par le mathématicien anglais William Oughtred au début du XVII^e). On donne en Annexe une preuve du théorème de Pythagore en termes de proportions géométriques pour illustrer leur pouvoir expressif.

Il existe d'une part des extensions multivaluées de la proportion analogique booléenne [6] qui permettent d'évaluer sur $[0, 1]$ à quel point une proportion analogique entre quatre nombres de $[0, 1]$ tient approximativement. Des cadres généraux [8], [17] ont été récemment proposés pour définir si ou ou non une proportion analogique tient entre quatre nombres de $[0, 1]$. Ces cadres inclut les proportions arithmétiques et géométriques comme cas particuliers importants. C'est pourquoi dans la suite de cet article on ne considérera que ces deux proportions.

3 Proportion analogique entre valeurs de probabilité

Dans cette section, et la suivante, on s'intéresse à des proportions analogiques entre des nombres réels qui sont des probabilités, qui ont donc des valeurs entre 0 et 1.
On considère d'abord le cas de quatre probablités qui réfèrent à quatre populations différentes et qui concernent une certaine valeur d'un même attribut pour les élements de ces quatre ensemble. a, b, c, d sont donc les probabilités qu'un élément d'un ensemble A, respectivement B, C, D prenne la valeur v_i pour l'atttibut i. On peut bien sûr s'intéresser à différents attributs simultanément. C'est par exemple le cas si on dispose d'une collection d'exemples et qu'on peut faire des statistiques sur la valeur d'attributs pour 4 groupes de données différents.
Dans la suite on utilise les notations suivantes $::_{ari}$ et $::_{geo}$ pour distinguer les proportions analogiques arithmétiques er géométriques.
La propriété suivante, facile à vérifier, garantit que si la proportion analogique tient entre quatre probabilités, elle tient aussi si on s'intéresse à l'évènement contraire :

Propriété 1
Si $a : b ::_{ari} c : d$ *alors* $1 - a : 1 - b ::_{ari} 1 - c : 1 - d$.

Cette propriété désirable n'est pas vérifiée en général pour $::_{geo}$. Elle tient cependant si les probabilités sont aussi en proportion arithmétique. Cette propriété peut être considérée comme le pendant de l'implication suivante valide en logique Booléenne :

$$a : b :: c : d \Rightarrow \neg a : \neg b :: \neg c : \neg d$$

où a, b, c, d représentent des valeurs booléennes. Cela exprime l'indépendance au codage de la proportion booléenne, vérifiable sur la Table 1.

Propriété 2
Si $a : b ::_{ari} c : d$ *et si* $a : b ::_{geo} c : d$ *alors on a*
$1 - a : 1 - b ::_{geo} 1 - c : 1 - d$.

Preuve On vérifie que $(1 - a)(1 - d) = (1 - b)(1 - c)$ si $ad = bc$ et $a + d = b + c$. □

Comme dans les cas booléen et nominal, il n'existe pas toujours de x tel que $a : b ::_{ari} c : x$ tienne. En effet

$$0 \leq a \leq 1, 0 \leq b \leq 1, 0 \leq c \leq 1 \not\Rightarrow 0 \leq x = b+c-a \leq 1$$

De la même façon, il n'existe pas toujours de x tel que $a : b ::_{geo} c : x$ tienne, puisque

$$0 \leq a \leq 1, 0 \leq b \leq 1, 0 \leq c \leq 1 \not\Rightarrow 0 \leq x = \frac{bc}{a} \leq 1$$

Dans les deux cas, si b et c sont grands alors on doit avoir a grand, et si a est petit, on doit avoir b ou c petits pour qu'il y ait une solution.

4 Proportion analogique entre distributions de probabilité

Au lieu des vecteurs comme dans la définition 1, nous considérons maintenant des distributions de probabilité normalisées. a, b, c, d désignent maintenant quatre distributions de probabilité sur un domaine fini. On a donc $\Sigma_{i=1,n} a_i = 1$, $\Sigma_{i=1,n} b_i = 1$, $\Sigma_{i=1,n} c_i = 1$, $\Sigma_{i=1,n} d_i = 1$. On va examiner plusieurs définitions possibles de proportions analogiques entre des distributions de probabilité et étudier leurs propriétés.

4.1 Définition arithmétique

Considérons tout d'abord une définition basée sur la proportion arithmétique.

Définition 2 *Soient a, b, c, d quatre distributions de probabilité sur un domaine fini $X = \{x_1, \cdots, x_n\}$. Alors ces distributions sont en proportion analogique* arithmétique*, notée $a : b ::_{ari} c : d$ si pour tout i on a $a_i - b_i = c_i - d_i$, où $a_i = a(x_i)$, $b_i = b(x_i)$, $c_i = c(x_i)$, $d_i = d(x_i)$.*

Il est clair que $::_{ari}$ satisfait les 3 postulats des proportions analogiques. Cette définition préserve la normalisation des probabilités.

Proposition 1 *Soient a, b, c trois distributions de probabilité normalisées, si $a : b ::_{ari} c : d$ et si $0 \leq c_i + b_i - a_i \leq 1$, $\forall i$, alors d est une distribution de probabilité normalisée.*

Preuve
La condition $0 \leq c_i + b_i - a_i \leq 1$ assure que $0 \leq d_i \leq 1$. Comme $\forall i, a_i - b_i = c_i - d_i$, l'addition membre à membre de ces égalités montre que $\Sigma_{i=1,n} d_i = 1$ puisque $\Sigma_{i=1,n} a_i = 1; \Sigma_{i=1,n} b_i = 1$ et $\Sigma_{i=1,n} c_i = 1$. □

La condition $0 \leq c_i + b_i - a_i \leq 1$ est nécessaire pour que d soit une distribution de probabilité, comme le montre le contre-exemple suivant.

Contre-exemple 1 *Prenons $n = 2$. $a_1 = 0{,}7, a_2 = 0{,}3$; $b_1 = 0{,}3, b_2 = 0{,}7$; $c_1 = 0{,}2, c_2 = 0{,}8$.*
Ce qui donne $d_1 = -0{,}2, d_2 = 1{,}2$. □

La définition 2 couvre le cas déterministe :

Observation 1 *Dans le cas déterministe, les distributions de probabilités sont telles que $\exists i, e_i = 1$ and $\forall j \neq i, e_j = 0$, pour $e \in \{a, b, c, d\}$. Il y a alors trois possibilités pour que $a : b ::_{ari} c : d$ tienne :*

- *$\exists i, a_i = b_i = c_i = d_i = 1$;*
- *$\exists i, a_i = b_i = 1$ et $\exists j, c_j = d_j = 1$;*
- *$\exists i, a_i = c_i = 1$ et $\exists j, b_j = d_j = 1$.* □

Donnons maintenant quelques exemples, non extrêmes comme les précédents, de distributions pour lesquelles on obtiendra $a : b ::_{ari} c : d$.

Exemple 1

1. *En utilisant la Propriété 1, on voit que dès que $\exists i, \exists j$, tel que $a_i : b_i ::_{ari} c_i : d_i$, avec $a_j = 1 - a_i, b_j = 1 - b_i, c_j = 1 - c_i, d_j = 1 - d_i$ et $a_k = b_k = c_k = d_k = 0, \forall k \neq i, j$, on a $a : b ::_{ari} c : d$.*
2. *En fait si $a_i : b_i ::_{ari} c_i : d_i$, pour tout réel λ on a $\lambda - a_i : \lambda - b_i ::_{ari} \lambda - c_i : \lambda - d_i$. Ainsi en prenant des λ positifs dont la somme n'excède pas 1, on peut construire des paires $((a_i, b_i, c_i, d_i), (a_j, b_j, c_j, d_j))$ formant deux proportions arithmétiques telles $a_i + a_j = b_i + b_j = c_i + c_j = d_i + d_j = \lambda$, comme dans l'exemple suivant où on a des allocations partielles de probabilité $\lambda = 0{,}4$ et $\lambda' = 0{,}5$ complétées par un quadruplet (a_k, b_k, c_k, d_k) de valeurs égales à 0,1 :*
   $a_1 = 0{,}1; a_2 = 0{,}3; a_3 = 0{,}1; a_4 = 0{,}2; a_5 = 0{,}3.$
   $b_1 = 0{,}3; b_2 = 0{,}1; b_3 = 0{,}1; b_4 = 0{,}3; b_5 = 0{,}2.$
   $c_1 = 0{,}2; c_2 = 0{,}2; c_3 = 0{,}1; c_4 = 0{,}4; c_5 = 0{,}1.$
   $d_1 = 0{,}4; d_2 = 0; d_3 = 0{,}1; d_4 = 0{,}5; d_5 = 0.$
   On vérifie bien que $a : b ::_{ari} c : d$ tient entre des distributions normalisées, toutes différentes.
3. *Pour que $a : b ::_{ari} c : d$ soit satisfait, il n'est pas nécessaire de procéder comme précédemment, c'est-à-dire de construire des paires de quadruplets dont la somme terme à terme est constante. C'est ce que montrent les deux exemples suivants pour $n = 3$ et $n = 4$:*
   $a_1 = 0{,}3; a_2 = 0{,}2; a_3 = 0{,}5.$
   $b_1 = 0{,}5; b_2 = 0{,}1; b_3 = 0{,}4.$
   $c_1 = 0{,}4; c_2 = 0{,}2; c_3 = 0{,}4.$
   $d_1 = 0{,}6; d_2 = 0{,}1; d_3 = 0{,}3.$
4. $a_1 = 0{,}1; a_2 = 0{,}2; a_3 = 0{,}4; a_4 = 0{,}3.$
   $b_1 = 0{,}3; b_2 = 0{,}3; b_3 = 0{,}2; b_4 = 0{,}2.$
   $c_1 = 0{,}2; c_2 = 0{,}2; c_3 = 0{,}2; c_4 = 0{,}4.$
   $d_1 = 0{,}4; d_2 = 0{,}3; \quad d_3 = 0; d_4 = 0{,}3.$ □

Ces exemples montrent qu'ils existent des distributions non triviales satisfaisant la définition 2. Bien entendu, étant donné trois distributions a, b, c, il n'existe pas toujours de distribution d telle que $a : b ::_{ari} c : d$ tienne, puisque on doit avoir $\forall i, 0 \leq c_i + b_i - a_i \leq 1$. Notons cependant que étant donné a, b, il est toujours possible de trouver c tel que ces inégalités soient satisfaites pour chaque i. on peut alors trouver une distribution d en proportion anlogique arithmétique avec a, b, c.

Remarque 1 *Il est naturel de se demander ce qu'il en serait d'une définition analogue à la définition 2 en termes de proportion géométrique, c'est-à-dire considerer la définition suivante :*

Définition 3 *Soient a, b, c, d quatre distributions de probabilité sur un domaine fini $X = \{x_1, \cdots, x_n\}$. Alors ces distributions sont en proportion analogique* géométrique*, notée $a : b ::_{geo} c : d$ si pour tout i on a $\frac{a_i}{b_i} = \frac{c_i}{d_i}$,*
où $a_i = a(x_i)$, $b_i = b(x_i)$, $c_i = c(x_i)$, $d_i = d(x_i)$.

Malheureusement, comme le montre le contre-exemple suivant, la contrepartie de la proposition 1 pour la proportion géométrique est fausse. La normalisation de a, b, c ne suffit pas pour garantir celle de d quand $a : b ::_{geo} c : d$ tient.

Contre-exemple 2 *Prenons $n = 3$. On a $a_1 + a_2 + a_3 = 1$, $b_1 + b_2 + b_3 = 1$, $c_1 + c_2 + c_3 = 1$, et $a_1 \cdot d_1 = b_1 \cdot c_1$, $a_2 \cdot d_2 = b_2 \cdot c_2$, $a_3 \cdot d_3 = b_3 \cdot c_3$, avec la condition $b_i \cdot c_i \leq a_i$.*

- *$b_1 = 0{,}2; b_2 = 0{,}3; b_3 = 0{,}5$, $c_1 = 0{,}4; c_2 = 0{,}3; c_3 = 0{,}3$ donc $b_1 \cdot c_1 = 0{,}08; b_2 \cdot c_2 = 0{,}09; b_3 \cdot c_3 = 0{,}15$. On prend $a_1 = 0{,}3; a_2 = 0{,}4; a_3 = 0{,}3$. Ce qui donne $d_1 = \frac{4}{15}; d_2 = \frac{9}{40}, ; d_3 = \frac{1}{2}$, et finalement $\Sigma_i d_i = 0{,}991666 < 1$*
- *$b_1 = 0{,}1; b_2 = 0{,}5; b_3 = 0{,}4$, $c_1 = 0{,}2; c_2 = 0{,}2; c_3 = 0{,}6$. Donc $b_1 \cdot c_1 = 0{,}02; b_2 \cdot c_2 = 0{,}10; b_3 \cdot c_3 = 0{,}24$. En conservant le même a qu'au dessus, on obtient $d_1 = \frac{1}{15}; d_2 = \frac{1}{4}; d_3 = \frac{4}{5}$ et donc $\Sigma_i d_i = 1{,}116666 > 1$.*

Remarque 2 *Dans le même esprit, on pourrait se poser la question de définir une proportion analogique entre des distributions de* possibilité *[20, 5] a, b, c, d, où la normalisation est exprimée par $\max_i a_i = 1, \max_i b_i = 1, \max_i c_i = 1, \max_i d_i = 1$, les a_i, b_i, c_i, d_i étant compris entre 0 et 1. Comme une distribution de possibilité e peut être vue en termes de ses α-coupes $e_\alpha = \{i \mid e_i \geq \alpha\}$ qui sont des sous-ensembles emboités $e_\alpha \subseteq e_\beta$ si $\alpha \geq \beta$, on peut se ramener à l'approche pour le cas booléen pour chaque α-coupe [16].*

4.2 Définition combinant les deux types de proportions numériques

Dans cette sous-section, on étudie une définition de la proportion analogique entre distributions de probabilité, plus exigeante que la définition 2 car combinant proportion arithmétique et proportion géométrique.

Définition 4 *Soient a, b, c, d quatre distributions de probabilité sur un domaine fini $X = \{x_1, \cdots, x_n\}$. Alors ces distributions sont en proportion analogique* arithmético-géométrique*, notée $a : b ::_{arge} c : d$ si pour tout i on a $a_i - b_i = c_i - d_i$ et $\frac{a_i}{b_i} = \frac{c_i}{d_i}$, où $a_i = a(x_i)$, $b_i = b(x_i)$, $c_i = c(x_i)$, $d_i = d(x_i)$.*

Il est clair que $::_{arge}$ satisfait les 3 postulats des proportions analogiques. Malgré le caractère contraignant de cette définition, il existe des distributions de probabilité non triviales qui la satisfont.

Proposition 2 *Il existe des distributions de probabilité a, b, c, d telles que $a : b ::_{arge} c : d$ tient. Elles sont telles que pour chaque composante i,*

- *cas 1 : soit $a_i = b_i$ et $c_i = d_i$,*
- *cas 2 : soit $a_i = c_i$ et $b_i = d_i$.*

et ce sont les seules solutions.

Preuve
En posant $a_i + d_i = s_i$ et $a_i d_i = p_i$, on obtient $d_i = s_i - a_i$, puis $a_i(s_i - a_i) = p_i$, c'est-à-dire $a_i^2 - s_i \cdot a_i + p_i = 0$. Ainsi, a_i doit être une solution de l'équation du second degré [17] :

$$x^2 - s_i x + p_i = 0$$

Son discriminant est égal à $\Delta = s_i^2 - 4p_i = (b_i + c_i)^2 - 4b_i \cdot c_i = (b_i - c_i)^2$. Les deux solutions sont donc $a_i = \frac{s_i \pm \sqrt{\Delta}}{2} = \frac{b_i + c_i \pm (b_i - c_i)}{2}$. Ce qui donne $a_i = b_i$ ou $a_i = c_i$. Ce qui conduit à $(a_i, d_i) = (b_i, c_i)$ ou $(d_i, a_i) = (b_i, c_i)$.
Donc $a : b ::_{arge} c : d$ si et seulement si pour chaque composante i, on a
- $a_i = b_i$ et $c_i = d_i$, ou
- $a_i = c_i$ et $b_i = d_i$ □

Voici un exemple d'une proportion analogique entre des distributions de probabilité a, b, c, d normalisées, toutes différentes, obéissant à la définition 4 :

Exemple 2
$a_1 = 0.1, a_2 = 0.3, a_3 = 0.2, a_4 = 0.3, a_5 = 0.1$
$b_1 = 0.1, b_2 = 0.4, b_3 = 0.2, b_4 = 0.2, b_5 = 0.1$
$c_1 = 0.1, c_2 = 0.3, c_3 = 0.3, c_4 = 0.3, c_5 = 0$
$d_1 = 0.1, d_2 = 0.4, d_3 = 0.3, d_4 = 0.2, d_5 = 0$

On peut vérifier que les quatre distributions sont bien normalisées, et que pour chaque i, on a à la fois $a_i - b_i = c_i - d_i$ et $\frac{a_i}{b_i} = \frac{c_i}{d_i}$ (rappelons qu'on a pris la convention $\frac{0}{0} = 1$). Observons que pour les i on a trois sortes de motifs possibles pour les valeurs de probabilités : (s, s, s, s), (s, t, s, t) et (s, s, t, t), ce sont précisément ceux déjà rencontrés pour les valeurs nominales en sous-section 2.2.2. Notons que les deux derniers motifs doivent être obligatoirement présents si on veut avoir des distributions distinctes, sinon on a $a = b$ (et donc $c = d$), ou $a = c$ (et donc $b = d$). Le motif (s, s, s, s) peut être absent. Une situation semblable avait déjà été observée dans le cas booléen [3].

Il est facile de trouver des distributions de probabilité a, b, c, d telles que $a : b ::_{arge} c : d$ tient. En effet, on a le résultat suivant.

Proposition 3 *Etant donné deux distributions de probabilité a et b différentes, il existe toujours deux distributions de probabilité c et d telles que $a : b ::_{arge} c : d$ tient. S'il existe au moins un i tel que $a_i = b_i$, les quatre distributions peuvent être différentes.*

Preuve
En effet, si $a_i \neq b_i$, alors $c_i = a_i$ et $d_i = b_i$. Remarquons, puisque $\Sigma_{i \mid a_i = b_i} a_i = \Sigma_{i \mid a_i = b_i} b_i \triangleq \rho$, que $\Sigma_{i \mid a_i \neq b_i} a_i = \Sigma_{i \mid a_i \neq b_i} b_i = 1 - \rho$. Donc on a aussi $\Sigma_{i \mid c_i \neq d_i} c_i = \Sigma_{i \mid c_i \neq d_i} d_i = 1 - \rho$. Pour les i tels que $a_i = b_i$, on affecte à $c_i = d_i$ des parts de la masse restante ρ de façon à ce que $\Sigma_{i=1,n} c_i = \Sigma_{i=1,n} d_i = 1$. □

Ces probabilités a, b, c, d satisfaisant $a : b ::_{arge} c : d$ ont, comme on vient le voir, une forme particulière. Elles satisfont une propriété remarquable en termes de divergence de Kullback-Leibler (notée KL), qui rappelons-le, évalue le

changement entre deux distributions a et b par l'expression $KL(a,b) = \Sigma_{i=1,n} a_i \log \frac{a_i}{b_i}$ (dans le cas discret). En général, $KL(a,b) \neq KL(b,a)$. Cependant, on peut énoncer le résultat suivant :

Proposition 4 *Soient a, b, c, d quatre distributions de probabilité formant une proportion analogique arithmético-géométrique $a : b ::_{arge} c : d$. Alors on a :*

$$KL(a,b) = KL(c,d)$$

Preuve
Puisque $KL(a,b) = \Sigma_{i=1,n} a_i \log \frac{a_i}{b_i}$ et $KL(c,d) = \Sigma_{i=1,n} c_i \log \frac{c_i}{d_i}$, examinons les différents termes i : $a_i \log \frac{a_i}{b_i}$ et $c_i \log \frac{c_i}{d_i}$;
On sait par la Proposition 2, qu'il y deux cas :
- cas (1) $a_i = b_i$: les deux termes sont nuls.
- cas (2) : $a_i = c_i$: On a donc $a_i \log \frac{a_i}{b_i} = c_i \log \frac{c_i}{d_i}$ par simple remplacement puisque $\frac{a_i}{b_i} = \frac{c_i}{d_i}$.
D'où $KL(a,b) = KL(c,d)$. □

Puisque $::_{arge}$ satisfait les postulats des proportions analogiques, on a aussi

Propriété 3 *Si $a : b ::_{arge} c : d$, alors*
- *i) $KL(a,c) = KL(b,d)$;*
- *ii) $KL(b,a) = KL(d,c)$;*
- *iii) $KL(d,b) = KL(c,a)$.*

Preuve i) étant donné la proposition 4 et la permutation centrale ; ii) puisqu'on obtient $a : b ::_{arge} c : d \Rightarrow b : a ::_{arge} d : c$, par applications successives de la permutation centrale, de la symétrie, et de la permutation centrale ; iii) la dernière égalité, qui correspond à la permutation des extrêmes, peut s'obtenir par application de la seconde à la première. □

Un exemple de proportion analogique entre des distributions continues, définies à l'aide de fonctions linéaires par morceaux, est donné dans l'Annexe 2.

5 Perspectives

Dans cette section, on suggère quelques pistes d'applications des proportions analogiques entre probabilités. Puisque les résultats de la section 4 s'appliquent aussi à toute collection de nombres positifs ou nuls dont la somme est 1, on termine par une application aux sommes pondérées en aggrégation multi-critères (puisque comme dans une distribution de probabilité, la somme des coefficients y est égale à 1).
Commençons par rappeler le principle de la classification bayésienne. On a par le théorème de Bayes :

$$P(c|x)P(x) = P(x|c)P(c)$$

où x est un vecteur de valeurs d'attributs décrivant un item, c est une classe, et on attribue à x, parmi les classes possibles, la classe c qui maximise $P(c|x)$. On peut remarquer que cette égalité est une proportion géométrique :

$$P(x) : P(x|c) :: P(c) : P(c|x)$$

On peut au moins penser à trois pistes d'utilisation des proportions analogiques sur des probabilités :

1. **Parallèle analogie-probabilités**
   Tout comme l'approche bayésienne, les proportions analogiques offrent une application à la classification [4, 2]. Ceci suggère de faire un parallèle entre les deux. Considérons quatre items a, b, c, d dont les descriptions vectorielles forment une proportion analogique au sens de la définition 1. De plus on suppose que ces quatre items ont été classés, d'une part par la méthode bayésienne, et d'autre part par la méthode des proportions analogiques. Si les classes fournies pour a, b, c, d par le classifieur analogique sont elles-mêmes en proportion analogique, on peut bien sûr se demander si les deux classifieurs sont en accord, mais aussi plus généralement, si les distributions de probabilité fournies par la règle de Bayes forment une proportion analogique (au moins approximativement au sens de l'extension graduelle numérique des proportions analogiques entre valeurs nominales, appelées extension conservative dans [6]).

2. **Valeurs d'attributs incertaines**
   Dans un problème de classification, certains attributs peuvent présenter une incertitude quant à leur valeur. Considérons le cas d'un seul attribut incertain, qu'il soit booléen ou nominal. Pour un item donné, chaque valeur possible de cet attribut est associée à une probabilité. Supposons que quatre items correspondant à la valeur la plus probable de l'attribut forment une proportion analogique.
   Si les probabilités associées respectent également cette proportion (au moins approximativement comme plus haut) et que l'équation analogique sur les classes permette d'inférer une solution pour le quatrième item, alors on peut accepter cette solution avec la probabilité correspondante.

3. **Proportions analogiques entre distributions**
   Rappelons que l'apprentissage par transfert peut être vu comme une sorte de raisonnement analogique effectué au niveau méta (voir par exemple [19]), puiqu'il s'agit de tirer parti de ce qui a été appris sur un domaine source afin d'améliorer le processus d'apprentissage dans un domaine cible lié au domaine source. Comme le suggère le vocabulaire utilisé, la démarche est assez semblable à celle qui préside au raisonnement à partir de cas [7].
   On peut se demander si des proportions analogiques entre distributions de probabilités, en particulier au sens de la définition 4, pourrait permettre des transferts plus sophistiqués et plus contrôlés, en utilisant les Propositions 2 et 3.
   Plus simplement, on pourrait chercher à savoir pour deux attributs d'intérêt (l'âge et le sexe dans

l'exemple ci-après), si les probabilités d'être dans une classe c

— pour un homme de moins de 40 ans,
— pour une femme de moins de 40 ans,
— pour un homme de plus de 40 ans,
— pour une femme de plus de 40 ans,

forment une proportion analogique pour les distributions de probabilité sur les classes de chacune des quatre populations.

Agrégation multi-critères

Des valeurs positives dont la somme fait 1 ne sont pas nécessairement des probabilités. Cela peut être aussi les coefficients d'une somme pondérée. Ainsi on pourra ajuster des coefficients d par rapport à d'autres jeux de coefficients a, b, c en résolvant une équation analogique $a : b :: c : x$ entre des distributions de coefficients, comme dans l'exemple suivant :

— a : coefficients des matières dans la filière scientifique de l'établissement 1,
— b : coefficients des matières dans la filière littéraire de l'établissement 1,
— c : coefficients des matières dans la filière scientifique de l'établissement 2,
— d : coefficients des matières dans la filière littéraire de l'établissement 2,

6 Conclusion

Cet article est vraisemblablement le premier à s'intéresser à des proportions analogiques sur des distributions de probabilité. On a notamment montré qu'il était possible d'avoir une définition combinant les proportions arithmétiques et géométriques. Cette définition garantit l'invariance, dans le cas discret, de la divergence de Kullback-Leibler entre les les distributions formant les deux paires de la proportion analogique. L'extension au cas continu nécessite une étude minutieuse, pour mieux comprendre quelles sont les situations possibles pour avoir une proportion analogique entre des distributions (au delà du premier exemple donné dans l'Annexe 2), et savoir quand la conservation de la divergence de Kullback-Leibler associée aux paires est préservée. Cela constitue donc une piste de recherche à poursuivre. Plus généralement, les applications concrètes restent à développer.

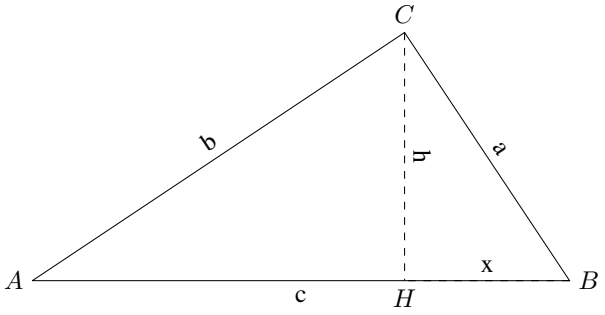
Remerciements

Les auteurs remercient Giuseppe Sanfilippo pour avoir posé la question "Do analogical proportions apply to probabilities ?" lors de la présentation de leur article [17]. Le présent article espère montrer que la question était féconde.

Cette recherche a été soutenue par le projet ANR « Analogies : from Theory to Tools and Applications » (AT2TA), ANR-22-CE23-0023.

Annexe 1 : Preuve du théorème de Pythagore par proportions géométriques

Considérons un triangle ABC rectangle en C, et la hauteur issue de C, comme sur la figure ci-après.
Les angles $\widehat{HAC}$ et $\widehat{HCB}$ sont égaux comme ayant le même complément $\widehat{ACH}$. Les triangles ABC, ACH et CBH sont donc semblables.



Dans cette figure, deux côtés d'un des trois triangles rectangles sont donc proportionnels aux deux côtés correspondants d'un autre triangle rectangle, et forment donc une proportion géométrique, en termes de longueurs.
Par exemple, on a
$AB : CB :: AC : HC$ c'est-à-dire $c : a :: b : h$
et donc $ab = ch$.
Si $a = 3$, $b = 4$, $c = 5$ on obtient $h = 2.4$.
On a aussi
$CH : AH :: BH : CH$ c'est-à-dire $h : c - x :: x : h$.
Ainsi
$h^2 = cx - x^2$, i.e., $h^2 + x^2 = cx$
Mais on a aussi
$AB : CB :: CB : BH$ c'est-à-dire $c : a :: a : x$
et donc $cx = a^2$.
On obtient $h^2 + x^2 = a^2$. QED !

Pour $a = 3$, $c = 5$, on obtient $x = 1.8$.

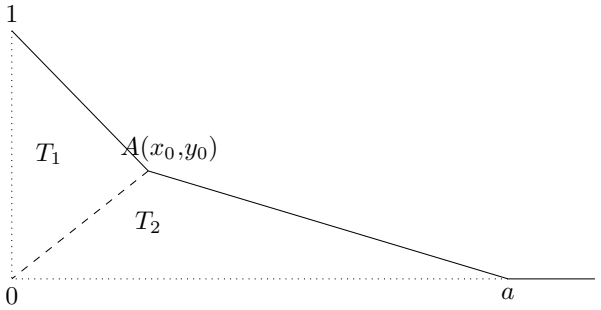
On peut aussi écrire
$CA : AB :: AH : CA$ c'est-à-dire $b : c :: c - x : b$
ce qui donne $b^2 = c^2 - cx$.
Ce qui conduit à
$c^2 = a^2 + b^2$.

NB : Notons l'usage d'une proportion continue
$(c : a :: a : x)$.

Annexe 2 : Un exemple de proportion analogique entre des distributions continues

Un exemple, non trivial, simple, peut être construit de la façon suivante, on prend
$a(x) = p(x)$ sur $(-\infty, +\infty)$
$b(x) = p(x)$ sur $(-\infty, 0]$; $b(x) = q(x)$ sur $[0, +\infty)$
$c(x) = q(x)$ sur $(-\infty, 0]$; $c(x) = p(x)$ sur $[0, +\infty)$
$d(x) = q(x)$ sur $(-\infty, +\infty)$

où p et q sont des fonctions linéaires par morceaux définies à partir d'une fonction paramétrée $S_{x_0,a}$, définie sur $[0, +\infty)$, représentée en trait plein sur la figure ci-après.



Le point A a pour coordonnées (x_0, y_0). On prend ensuite :
$p(x) = S_{x_0,a}(x)$ sur $[0, +\infty)$, $S_{x_0,a}(-x)$ sur $(-\infty, 0]$.
$q(x)$ est définie de manière similaire avec des paramètres x'_0 et a'.
Notons qu'on a $p(0) = q(0) = 1$, et que a, b, c, d sont donc continues.
L'aire en dessous de $S_{x_0,a}$ est égale à $aire(T_1) + aire(T_2) = x_0/2 + ay_0/2$.
Pour assurer la normalisation des distributions a, b, c, d, cette aire doit être égale à $1/2$, on doit donc imposer $(x_0 + ay_0) = 1$ et $(x'_0 + a'y'_0) = 1$. Dans ces conditions, on peut vérifier facilement que $a : b ::_{arge} c : d$ et que $KL(a, b) = KL(c, d)$.

En partant de l'exemple ci-dessus, et en utilisant des fonctions paramétriques $S_{h,x_0,a}$ et $S_{h,x'_0,a'}$ similaires aux précédentes, mais telles que $S_{h,x_0,a}(0) = S_{h,x'_0,a'}(0) = h$, on peut construire des exemples plus complexes :
Soient $p_1 < p_2$. Soient $\lambda < 1/2$, $\lambda' < 1/2$.
On prend alors
$a(x) = p(x)$ sur $(-\infty, p_1]$; $a(x) = r(x)$ sur $[p_1, p_2]$; $a(x) = p(x)$ sur $[p_2, +\infty)$;
$b(x) = p(x)$ sur $(-\infty, p_1]$; $b(x) = r(x)$ sur $[p_1, p_2]$; $b(x) = q(x)$ sur $[p_2, +\infty)$;
$c(x) = q(x)$ sur $(-\infty, p_1]$; $c(x) = r(x)$ sur $[p_1, p_2]$; $c(x) = p(x)$ sur $[p_2, +\infty)$;
$d(x) = q(x)$ sur $(-\infty, p_1]$; $d(x) = r(x)$ sur $[p_1, p_2]$; $d(x) = q(x)$ sur $[p_2, +\infty)$;
où

- $p(x) = S_{h,x_0,a}(x - p_1)$ sur $[p_1, +\infty)$,
  $p(x) = S_{h,x_0,a}(p_1 - x)$ sur $(-\infty, p_1]$, et
- $q(x) = S_{h,x'_0,a'}(x - p_2)$ sur $[p_2, +\infty)$,
  $q(x) = S_{h,x'_0,a'}(p_2 - x)$ sur $(-\infty, p_2]$,
- $r(x)$ est tel que $r(p_1) = r(p_2) = h$ (assurant la continuité des 4 distributions) et $\int_{p_1}^{p_2} r(x)dx = \lambda + \lambda'$ (r peut éventuellement être pris comme uniforme),

avec les constraintes
$(hx_0 + ay_0) = 1 - 2\lambda$,
$(hx'_0 + a'y'_0) = 1 - 2\lambda'$,
et $h(p_2 - p_1) \leq \lambda + \lambda'$ (il y a égalité si r est uniforme).
assurant ainsi la normalisation des 4 distributions.

Références

[1] Nelly Barbot, Laurent Miclet, and Henri Prade. Analogy between concepts. *Artif. Intel.*, 275 :487–539, 2019.

[2] Myriam Bounhas and Henri Prade. Analogy-based classifiers : An improved algorithm exploiting competent data pairs. *Int. J. Approx. Reason.*, 158 :108923, 2023.

[3] Myriam Bounhas and Henri Prade. Revisiting analogical proportions and analogical inference. *Int. J. Approx. Reason.*, 171 :109202, 2024.

[4] Myriam Bounhas, Henri Prade, and Gilles Richard. Analogy-based classifiers for nominal or numerical data. *Int. J. Approx. Reason.*, 91 :36–55, 2017.

[5] Didier Dubois and Henri Prade. *Possibility Theory : An Approach to Computerized Processing of Uncertainty*. Plenum Press, 1988.

[6] Didier Dubois, Henri Prade, and Gilles Richard. Multiple-valued extensions of analogical proportions. *Fuzzy Sets Syst.*, 292 :193–202, 2016.

[7] Béatrice Fuchs, Jean Lieber, Laurent Miclet, Alain Mille, Amedeo Napoli, Henri Prade, and Gilles Richard. Case-based reasoning, analogy, and interpolation. In Pierre Marquis, Odile Papini, and Henri Prade, editors, *A Guided Tour of Artificial Intelligence Research : Volume I : Knowledge Representation, Reasoning and Learning*, pages 307–339. Springer, 2020.

[8] Yves Lepage and Miguel Couceiro. Analogie et moyenne généralisée. In *Actes des 18èmes Journées d'Intelligence Artificielle Fondamentale, La Rochelle, 1-3 Juil.*, pages 114–124, 2024.

[9] Laurent Miclet and Henri Prade. Handling analogical proportions in classical logic and fuzzy logics settings. In *Proc. 10th Eur. Conf. on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU'09),Verona*, pages 638–650. Springer, LNCS 5590, 2009.

[10] Vito Pirrelli and François Yvon. Analogy in the lexicon : a probe into analogy-based machine learning of language. In *Proc. 6th Int. Symp. on Human Communication, Santiago de Cuba*, page 6 p., 1999.

[11] George Polya. *Mathematics and Plausible Reasoning-Vol.1 : Induction and analogy in Mathematics, Vol.2 : Patterns of Plausible Inference*. Princeton Univ. Press, 2nd ed. 1968, 1954.

[12] Henri Prade and Gilles Richard. Cataloguing / analogizing : A nonmonotonic view. *Int. J. Intell. Syst.*, 26(12) :1176–1195, 2011.

[13] Henri Prade and Gilles Richard. From analogical proportion to logical proportions. *Logica Universalis*, 7(4) :441–505, 2013.

[14] Henri Prade and Gilles Richard. Multiple analogical proportions. *AI Commun.*, 34(3) :211–228, 2021.

[15] Henri Prade and Gilles Richard. Analogical proportion- based induction : From classification to creativity. *J. of Applied Logics - IfCoLog J. of Logics and their Applicatiions*, 11 :51–87, 2024.

[16] Henri Prade and Gilles Richard. Diagrammatic analogical reasoning. In J. Lemanski, M. W. Johansen, E. Manalo, P. Viana, R. Bhattacharjee, and R. Burns, editors, *Proc. 14th Int. Conf. on Diagrammatic Representation and Inference (Diagrams'24), Münster, Sept. 27 - Oct. 1*, volume 14981 of *LNCS*, pages 485–489. Springer, 2024.

[17] Henri Prade and Gilles Richard. Frank's triangular norms in Piaget's logical proportions. In S. Destercke, M. V. Martinez, and G. Sanfilippo, editors, *Proc. 16th Int. Conf. on Scalable Uncertainty Management (SUM'24), Palermo, Nov. 27-29*, volume 15350 of *LNCS*, pages 369–377. Springer, 2024.

[18] Marta Sznajder. Janina Hosiasson-Lindenbaum on analogical reasoning : New sources. *Erkenntnis*, 89 :1349–1365, 2024.

[19] Hua-Yan Wang and Qiang Yang. Transfer learning by structural analogy. In Wolfram Burgard and Dan Roth, editors, *Proc. 25th AAAI Conf. on Artificial Intelligence, (AAAI'11), San Francisco, Aug.7-11*, pages 513–518. AAAI Press, 2011.

[20] Lotfi A. Zadeh. Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Syst.*, 1(1) :3–28, 1978.

Des logiques de variations avec attributs numériques pour représenter des règles d'adaptation*

Nicolas François[1], Jean Lieber[2]

[1] Lycée public Chopin, F-54000 Nancy, France
[2] Université de Lorraine, CNRS, LORIA, F-54000 Nancy, France

Résumé

Les connaissances d'adaptation d'un système de raisonnement à partir de cas (RàPC) sont souvent représentées par des règles d'adaptation qui peuvent être apprises à partir de la base de cas, en se fondant sur plusieurs approches proposées dans la littérature du RàPC. Cependant, la représentation formelle de telles règles a été bien moins étudiée. Cet article introduit un formalisme pour représenter de telles règles, s'appuyant sur un formalisme de couples attributs-valeurs dans lequel sont représentés les cas.

Mots-clés

raisonnement à partir de cas, représentation des connaissances, adaptation

Abstract

The adaptation knowledge container of a CBR system is often represented by adaptation rules, that can be learnt from the case base using various approaches proposed in the CBR literature. However, the formal representation of such rules has been much less investigated. This article introduces a formalism for representing such rules, given a formalism that can be used to express cases in an attribute-value formalism.

Keywords

case-based reasoning, knowledge representation, adaptation

1 Introduction

Le raisonnement à partir de cas (RàPC [9]) consiste à résoudre des problèmes en s'appuyant sur la résolution de problèmes antérieurs, appelés cas. L'adaptation est considérée, pour certains membres de la communauté du RàPC, comme un processus clef de ce type de raisonnements ; elle consiste à modifier un cas sélectionné dans la base de cas dans l'optique de la résolution du problème posé. Cet article s'inscrit dans cette tradition et la considère du point de vue de la représentation des connaissances : comment les connaissances d'adaptation peuvent-elle être représentées dans le cadre du RàPC ?

Pour s'attaquer à cette question, le point de départ a été certains travaux sur l'apprentissage de connaissances d'adaptation (ACA). L'ACA a une longue histoire dans le domaine du RàPC, au moins depuis un article de K. Hanney et M. T. Keane [5]. L'ACA s'appuie en général sur ce qu'on appelle parfois l'heuristique des différences [6] : les différences entre couples de cas dans la base de cas sont utilisées comme ensemble d'apprentissage pour un système d'apprentissage artificiel. En particulier, une notation a été introduite pour représenter de telles différences : elle consiste en expressions de la forme a^v où a est un attribut d'un cas et v est un *symbole de variation*. Par exemple, dans le domaine culinaire, pour exprimer que, pour une recette de dessert, 2 œufs peuvent être substitués par 1 banane, l'ensemble suivant d'expressions peut être employé :

$$\{\texttt{dessert}^{=1},\quad \texttt{\#œufs}^{-2},\quad \texttt{\#bananes}^{+1}\}$$

Ici, `dessert` est un attribut booléen : ses valeurs potentielles sont 0 et 1 (assimilées à *faux* et à *vrai*), alors que `#œufs` et `#bananes` sont des attributs à domaines entiers.

Cette notation a été utilisée dans des études relativement anciennes [1] tout comme dans des travaux plus récents [8]. À cette notation (ou syntaxe), une sémantique peut être associée, donnant naissance à des formalismes logiques, les *logiques de variations*, dans lesquelles les règles d'adaptation peuvent être représentées.

La section 2 présente des préliminaires portant principalement sur le RàPC. La base de cas et les connaissances du domaine peuvent être représentées dans ce qu'on pourrait qualifier de *logique statique* ; la section 3 présente la logique statique $(\mathcal{LG}, \models)$ qui est suffisamment expressive pour représenter des couples attributs-valeurs (avec des attributs booléens ou numériques) et des taxonomies (et au-delà). La section 4 définit la logique des variations $(\Delta\mathcal{LG}, \models)$ qui s'appuie sur $(\mathcal{LG}, \models)$: sa syntaxe, sa sémantique et quelques propriétés simples. La section 5 présente la façon dont des règles d'adaptation peuvent être défines en utilisant cette logique. Pour les résultats de ces sections, il est supposé que les logiques $(\mathcal{LG}, \models)$ et $(\Delta\mathcal{LG}, \models)$ s'appuient sur un groupe abélien infini $(\mathcal{G}, +)$; en particulier, le test de satisfiabilité s'appuie sur cette hypothèse. La section 6 explique comment modifier l'approche sous d'autres hypothèses (p. ex. si $(\mathcal{G}, +)$ est un groupe abélien

*Les auteurs tiennent à remercier les relectrices et relecteurs anonymes pour leurs lectures sérieuses et détaillées. Leurs remarques ont permis d'améliorer la qualité de cet article. Elles ont permis en particulier d'enlever une grosse bêtise énoncée en plein milieu d'une proposition ! Certaines suggestions nous seront utiles pour la rédaction d'articles futurs.

fini). La section 7 discute la contribution de cet article, conclut et présente quelques perspectives.

Remarques :

- Cet article est une version modifiée d'un article qui va être publié dans les actes d'ICCBR-2025 (la manifestation internationale et annuelle sur le RàPC).
- On peut noter que le contenu de cet article est assez proche de celui publié par les mêmes auteurs dans les JIAF-JFPDA en 2024 [3]. La différence principale entre le travail présenté ici et ce travail précédent est que la logique statique sur laquelle la logique des variations s'appuyait était la logique propositionnelle, alors que la logique statique considérée dans cet article est $(\mathcal{LG}, \models)$ qui étend la logique propositionnelle.
- La section 6 est une contribution originale de cet article (elle n'est pas dans l'article d'ICCBR-2025).
- Cet article contient des résultats sans leurs preuves : ces dernières peuvent être trouvées dans la version étendue de cet article, qui est accessible en ligne [4].

2 Préliminaires

Cette section présente quelques notations et hypothèses, principalement à propos du RàPC.

2.1 Notions et notations mathématiques

$\mathbb{B}$, l'ensemble des booléens, est assimilé à la paire d'entiers $\mathbb{B} = \{0, 1\}$ (0 pour faux, 1 pour vrai). $\mathbb{Z}$ est l'ensemble des entiers relatifs. Pour $a, b \in \mathbb{Z}$ avec $a \leqslant b$, $[\![a, b]\!] = \{n \in \mathbb{Z} \,|\, a \leqslant n \leqslant b\}$. $\mathbb{Q}$ est l'ensemble des rationnels.

Un *groupe abélien* est un couple $(\mathcal{G}, +)$ où $\mathcal{G}$ est un ensemble non vide et $+$ est une loi de composition interne sur $\mathcal{G}$ ($+ : (r, s) \in \mathcal{G} \mapsto r + s \in \mathcal{G}$) telle que (1) elle a un élément neutre noté 0 (pour tout $r \in \mathcal{G}$, $r + 0 = 0 + r = r$), (2) tout $r \in \mathcal{G}$ a un élément symétrique s ($r + s = s + r = 0$), s est noté $-r$, (3) $+$ est associative ($(r+s)+t = r+(s+t)$ pour tous $r, s, t \in \mathcal{G}$), et (4) $+$ est commutative ($r + s = s + r$ pour tous $r, s \in \mathcal{G}$). Pour $r, s \in \mathcal{G}$, $r - s$ est une abréviation de $r + (-s)$.

Pour l'addition usuelle sur les nombres, $(\mathbb{Q}, +)$ et $(\mathbb{Z}, +)$ sont des groupes. $(\{0, 1, 2\}, +)$ n'est pas un groupe puisque $+$ n'est pas interne dans cet ensemble (p. ex. $2 \in \{0, 1, 2\}$ mais $2 + 2 = 4 \notin \{0, 1, 2\}$).

2.2 Le RàPC : notions, notations et hypothèses

Ces préliminaires au sujet du RàPC ne reflètent pas toute la variété des systèmes de RàPC. En particulier, dans cet article, un cas est considéré comme un couple problème-solution. Un *problème* est, par définition, un élément $\mathtt{x}$ d'un ensemble donné $\mathcal{P}$. Une *solution* est, par définition, un élément $\mathtt{y}$ d'un ensemble donné $\mathcal{S}$. Ces deux ensembles $\mathcal{P}$ et $\mathcal{S}$ sont supposés disjoints. On suppose l'existence d'une relation sur $\mathcal{P} \times \mathcal{S}$ qui n'est pas complètement connue pour le système de RàPC, et qui se lit "a pour solution". Un *cas* est un couple $(\mathtt{x}, \mathtt{y}) \in \mathcal{P} \times \mathcal{S}$. Un *cas correct* est un cas $(\mathtt{x}, \mathtt{y})$ tel que $\mathtt{x}$ a pour solution $\mathtt{y}$. La *base de cas* $\mathtt{BC}$ est un ensemble fini de cas corrects. Un cas de $\mathtt{BC}$ est appelé *cas source* et est noté par $(\mathtt{x}^s, \mathtt{y}^s)$. Enfin, il est supposé que les cas sources représentent des épisodes *spécifiques* de résolution de problèmes.

Modèle de processus. Une session de RàPC consiste en la résolution d'un problème appelé le *problème cible* et noté $\mathtt{x}^{\mathtt{cible}}$. Un schéma de processus fréquent pour le RàPC consiste en (1) la sélection d'un cas source $(\mathtt{x}^s, \mathtt{y}^s)$ qui est considéré comme étant similaire à $\mathtt{x}^{\mathtt{cible}}$ (étape de *remémoration*), (2) la modification de ce cas remémoré afin de proposer une solution $\mathtt{y}^{\mathtt{cible}}$ de $\mathtt{x}^{\mathtt{cible}}$ (étape d'*adaptation*), et d'autres étapes non considérés dans cet article. Dans certains systèmes de RàPC, plusieurs cas sont remémorés avant d'être adaptés; dans cet article, cependant, on ne considérera que les remémorations et adaptation simples (avec un seul cas remémoré puis adapté).

Modèle de connaissances. Un système de RàPC s'appuie sur quatre *conteneurs de connaissances*. Le premier est la base de cas $\mathtt{BC}$ mentionnée plus haut.

Les connaissances du domaine $\mathtt{CD}$ expriment des connaissances à propos du vocabulaire avec lequel les cas sont représentés. $\mathtt{CD}$ peut être considéré comme un ensemble de contraintes sur ce vocabulaire. Par exemple, si "pomme" et "fruit" appartiennent à ce vocabulaire, la connaissances "Toute pomme est un fruit." indique qu'un cas impliquant une pomme qui n'est pas un fruit est incorrect.

Les connaissances pour la remémoration sont souvent représentées à l'aide d'une distance entre problèmes (il n'en sera guère question dans cet article).

Les connaissances d'adaptation $\mathtt{CA}$ sont utilisées durant l'adaptation. Dans cet article, $\mathtt{CA}$ est un ensemble de *règles d'adaptation* $\mathtt{ar}$. De façon générale, une telle règle peut être interprétée comme une fonction qui à $((\mathtt{x}^s, \mathtt{y}^s), \mathtt{x}^{\mathtt{cible}}) \in (\mathcal{P} \times \mathcal{S}) \times \mathcal{P}$ associe $\mathtt{y}^{\mathtt{cible}} \in \mathcal{S} \cup \{\mathtt{échec}\}$. Si $\mathtt{y}^{\mathtt{cible}} = \mathtt{échec}$, cela signifie que la règle d'adaptation n'est pas applicable au problème d'adaptation $((\mathtt{x}^s, \mathtt{y}^s), \mathtt{x}^{\mathtt{cible}})$. Dans le cas contraire, le cas $(\mathtt{x}^{\mathtt{cible}}, \mathtt{y}^{\mathtt{cible}})$ n'est pas nécessairement correct : le RàPC est en général un raisonnement hypothétique et le résultat qu'il produit n'est pas nécessairement correct. Afin d'augmenter la plausibilité que ce résultat soit correct, une valeur strictement positive $\mathtt{coût}$ est associée à une règle d'adaptation : on considérera que plus coût est bas, plus il est plausible que $(\mathtt{x}^{\mathtt{cible}}, \mathtt{y}^{\mathtt{cible}})$ soit correct.

Représentation des connaissances. $\mathtt{BC}$ et $\mathtt{CD}$ sont des conteneurs de connaissances à propos de ce qu'*est* ou *n'est pas* un cas : un $(\mathtt{x}^s, \mathtt{y}^s) \in \mathtt{BC}$ indique que $\mathtt{y}^s$ *est* une solution correcte de $\mathtt{x}^s$, alors qu'une formule α de $\mathtt{CD}$ indique une contrainte sur ce qu'un cas correct doit être. Les connaissances d'adaptation, quant à elles, concernent les transformations, comment un cas est *modifié* en un autre cas.

Pour ces raisons, deux types de logiques pour représenter des connaissances en RàPC sont distinguées : les *logiques statiques* pour $\mathtt{BC}$ et $\mathtt{CD}$ et les *logiques de variations* pour les connaissances d'adaptation.

Dans cet article, la logique statique considérée est $(\mathcal{LG}, \models)$,

qui sera présentée en section 3 et utilisée pour BC et CD. La logique des variations $(\Delta\mathcal{LG}, \models)$ pour représenter les connaissances d'adaptation s'appuie sur la logique précédente et est présentée en section 4.

3 La logique statique $(\mathcal{LG}, \models)$

La logique statique peut être n'importe quelle logique classique utilisée en représentation des connaissances, que ce soit la logique propositionnelle, une logique de descriptions ou autre. Cependant, l'étude présentée dans cet article s'appuie sur la logique $(\mathcal{LG}, \models)$ décrite ci-dessous, associée à une discussion sur son expressivité. Cette logique est paramétrée par un groupe abélien infini $(\mathcal{G}, +)$.

3.1 Syntaxe de $(\mathcal{LG}, \models)$

Soit $\mathcal{V}$ un ensemble non vide donné de symboles : les éléments a de $\mathcal{V}$ sont appelés les *variables*. Un *atome* de $(\mathcal{LG}, \models)$ est une expression de la forme $(a = r)$ où $a \in \mathcal{V}$ et $r \in \mathcal{G}$. Une formule de $(\mathcal{LG}, \models)$ est soit un atome soit une expression d'une des formes suivantes : $\bot$, $\top$, $\neg\alpha$, $\alpha\wedge\beta$ et $\alpha \vee \beta$ où α et β sont deux formules de cette logique. Les autres connecteurs peuvent être définis de manière usuelle, comme des abréviations. En particulier, le connecteur $\rightarrow$ est défini par $\alpha \rightarrow \beta = \neg\alpha \vee \beta$. $\mathcal{LG}$ est l'ensemble des formules de cette logique. L'ensemble des variables apparaissant dans $\alpha \in \mathcal{LG}$ est dénoté par $\mathcal{V}(\alpha)$.
Un *littéral* est soit un atome (qu'on considérera comme un littéral *positif*) soit de la forme $\neg\alpha$ où α est un atome (ce sera alors un littéral *négatif*). Un *cube* est une conjonction de littéraux : il a la forme $\ell_1 \wedge \ell_2 \wedge \ldots \wedge \ell_p$ où ℓ_k est un littéral pour $k \in [\![1, p]\!]$. Une formule est sous *FND* (forme normale disjonctive) si c'est une disjonction de cubes : $\gamma_1 \vee \gamma_2 \vee \ldots \vee \gamma_q$ où γ_k est un cube, pour $k \in [\![1, q]\!]$.

3.2 Sémantique de $(\mathcal{LG}, \models)$

Une *interprétation* $\mathcal{I}$ est une fonction de $\mathcal{V}$ dans $\mathcal{G}$. Soit Ω l'ensemble des interprétations. Pour $\mathcal{I} \in \Omega$ et $\alpha \in \mathcal{LG}$, la relation "$\mathcal{I}$ est un modèle de α" (dénotée par $\mathcal{I} \models \alpha$) est définie récursivement comme suit (pour $a \in \mathcal{V}$, $r \in \mathcal{G}$ et $\alpha, \beta \in \mathcal{LG}$) :

- $\mathcal{I} \models \bot$ n'est jamais vrai ;
- $\mathcal{I} \models \top$ est toujours vrai ;
- $\mathcal{I} \models (a = r)$ si $\mathcal{I}(a) = r$;
- $\mathcal{I} \models \neg\alpha$ si $\mathcal{I} \not\models \alpha$;
- $\mathcal{I} \models \alpha \wedge \beta$ si $\mathcal{I} \models \alpha$ et $\mathcal{I} \models \beta$;
- $\mathcal{I} \models \alpha \vee \beta$ si $\mathcal{I} \models \alpha$ ou $\mathcal{I} \models \beta$.

Pour $\alpha \in \mathcal{LG}$, soit $\mathcal{M}(\alpha)$ l'ensemble des modèles de α. α est *satisfiable* (ou *cohérente*) si $\mathcal{M}(\alpha) \neq \varnothing$ et α est une *tautologie* si $\mathcal{M}(\alpha) = \Omega$. Par exemple, $\bot$ est insatisfiable et $\top$ est une tautologie La relation $\alpha \models \beta$ signifie que $\mathcal{M}(\alpha) \subseteq \mathcal{M}(\beta)$. La relation $\alpha \equiv \beta$ signifie que $\mathcal{M}(\alpha) = \mathcal{M}(\beta)$. Si $\mathcal{B}$ est un ensemble de formules, $\mathcal{M}(\mathcal{B}) = \bigcap\{\mathcal{M}(\alpha) \mid \alpha \in \mathcal{B}\}$. $\mathcal{B} \models \alpha$ signifie que $\mathcal{M}(\mathcal{B}) \subseteq \mathcal{M}(\alpha)$.

3.3 À propos de l'expressivité de $(\mathcal{LG}, \models)$

Chaque variable a de $(\mathcal{LG}, \models)$ a le même domaine $\mathcal{G}$: pour toute $\mathcal{I} \in \Omega$, $\mathcal{I}(a) \in \mathcal{G}$. Une modification de $(\mathcal{LG}, \models)$ consiste à préciser le domaine de chaque variable (p. ex. $\mathbb{B}$, $\mathbb{Z}$, $\mathbb{Q}$, un intervalle d'entiers $[\![m, n]\!]$ ou de rationnels $[r, s]$ —$m, n \in \mathbb{Z}$, $r, s \in \mathbb{Q}$). Par conséquent, ce formalisme contient les formalismes attributs-valeurs (avec des domaines d'attributs limités aux nombres, booléens et types énumérés) et les taxonomies (une taxonomie étant représentée par un ensemble d'implications, p. ex. $(\texttt{apple} = 1) \rightarrow (\texttt{fruit} = 1)$ signifie que toute pomme est un fruit). Pour simplifier la présentation, dans cet article, nous ne considérons que la logique statique $(\mathcal{LG}, \models)$, considérant que l'extension avec des types de variables différents ne devrait pas poser beaucoup de problèmes de modification.

3.4 Représenter des cas dans $(\mathcal{LG}, \models)$

Comme indiqué en section 2, les ensembles $\mathcal{P}$ et $\mathcal{S}$ sont supposés disjoints. Cela suggère de partitionner l'ensemble des variables en $\{\mathcal{V}_\mathcal{P}, \mathcal{V}_\mathcal{S}\}$: $\mathcal{V}_\mathcal{P}$ (resp., $\mathcal{V}_\mathcal{S}$) est l'ensemble des *variables de problèmes* (resp., *variables de solutions*). Ainsi, un problème (resp., une solution) est représenté par une formule dont les variables appartiennent à $\mathcal{V}_\mathcal{P}$ (resp., à $\mathcal{V}_\mathcal{S}$). Donc, un cas $(\mathtt{x}, \mathtt{y}) \in \mathcal{P} \times \mathcal{S}$ peut être écrit $\mathtt{x} \wedge \mathtt{y}$: trouver la partie problème $\mathtt{x}$ et la partie solution $\mathtt{y}$ de $\mathtt{x} \wedge \mathtt{y}$ est facile puisque $\mathcal{V}_\mathcal{P} \cap \mathcal{V}_\mathcal{S} = \varnothing$.
Par ailleurs, il est supposé que les cas sources $(\mathtt{x}^s, \mathtt{y}^s)$ représentent des expériences spécifiques. Cela signifie que pour tout $(\mathtt{x}^s, \mathtt{y}^s) \in \mathtt{BC}$, $\mathcal{M}(\mathtt{x}^s \wedge \mathtt{y}^s)$ est un singleton $\{\mathcal{I}^s\}$. Par conséquent, $\mathtt{x}^s$ et $\mathtt{y}^s$ peuvent être écrits comme conjonctions d'atomes sur toutes les variables de, respectivement, $\mathcal{V}_\mathcal{P}$ et $\mathcal{V}_\mathcal{S}$: $\mathtt{x}^s \equiv \bigwedge_{a\in\mathcal{V}_\mathcal{P}}(a = \mathcal{I}^s(a))$ et $\mathtt{y}^s \equiv \bigwedge_{a\in\mathcal{V}_\mathcal{S}}(a = \mathcal{I}^s(a))$.

4 La logique de variations $(\Delta\mathcal{LG}, \models)$

Étant donné une logique statique qui peut être utilisée pour représenter des cas et des connaissances du domaine, une logique de variations peut être définie pour représenter, en particulier, des connaissances d'adaptation. Cette section décrit la logique de variations $(\Delta\mathcal{LG}, \models)$ construite à partir de la logique statique $(\mathcal{LG}, \models)$: sa syntaxe, sa sémantique, certaines de ses propriétés et les opérateurs sur ces logiques qui vont être utiles pour représenter des règles d'adaptation.

4.1 Syntaxe de $(\Delta\mathcal{LG}, \models)$

Pour commencer par un exemple, soit $\varphi = a^{+5} \wedge \neg b^{8\bullet} \wedge b^{\bullet 2}$ une formule de cette logique, pour $\mathcal{G} = \mathbb{Z}$. φ indique un changement de l'attribut a de la valeur r, pour un $r \in \mathbb{Z}$, à la valeur $r + 5$ et un changement de l'attribut b d'une valeur différente de 8 à la valeur 2. Les expressions $+5$, $8\bullet$ et $\bullet 2$ apparaissant dans φ sont appelés symboles de variation.
Plus généralement, l'ensemble des *symboles de variation* est défini par :

$$\mathcal{D} = \{+r,\ \ r\bullet,\ \ \bullet r \mid r \in \mathcal{G}\}$$

Un symbole de variation additionnel $-r$ est défini comme abréviation de $+(-r)$.
Un *atome* de $(\Delta\mathcal{LG}, \models)$ est une expression de la forme a^v où a est une variable et v est un symbole de variation. Une *formule* de $(\Delta\mathcal{LG}, \models)$ est soit un atome de cette logique,

soit d'une des formes suivantes : $\bot$, $\top$, $\neg\varphi$, $\varphi\wedge\psi$ et $\varphi\vee\psi$ où φ et ψ sont deux formules de cette logique. Les autres connecteurs sont définies comme des abréviations. L'ensemble des formules est noté par $\Delta\mathcal{LG}$. Une *formule de variations* est par définition un élément de $\Delta\mathcal{LG}$. L'ensemble des variables apparaissant dans $\varphi \in \Delta\mathcal{LG}$ est noté $\mathcal{V}(\varphi)$.

Les notions de littéraux, positifs ou négatifs, et de cubes, sont similaires à ceux de $(\mathcal{LG}, \models)$, par exemple, $a^{5\bullet} \wedge \neg a^{+2} \wedge \neg b^{\bullet 6}$ est un cube avec un littéral positif et deux littéraux négatifs. $\bot$ est aussi considéré comme un cube ($\bot$ peut être vu comme la conjonction $A \wedge \neg A$ pour A est un atome fixé de façon arbitraire).

4.2 Sémantique de $(\Delta\mathcal{LG}, \models)$

Chaque symbole de variation est interprété comme une relation binaire sur $\mathcal{G}$. Formellement, à tout $v \in \mathcal{D}$ est associé $\langle v\rangle \subseteq \mathcal{G} \times \mathcal{G}$ défini de la façon suivante (pour $r \in \mathcal{G}$):

$$\langle {+}r\rangle = \{(x, y) \in \mathcal{G}^2 \mid y = x + r\}$$
$$\langle r\bullet\rangle = \{(r, x) \mid x \in \mathcal{G}\}$$
$$\langle \bullet r\rangle = \{(x, r) \mid x \in \mathcal{G}\}$$

Soit $\Delta\Omega = \Omega \times \Omega$: c'est l'ensemble des interprétations utilisées pour définir la sémantique de $(\Delta\mathcal{LG}, \models)$: une interprétation de $\Delta\Omega$ est par conséquent un couple $(\mathcal{I}, \mathcal{J})$ de Ω. Dans la suite, $(\mathcal{I}, \mathcal{J})$ est simplement notée $\mathcal{IJ}$.

La relation affirmant qu'une $\mathcal{IJ} \in \Delta\Omega$ est un modèle de $\varphi \in \Delta\mathcal{LG}$ est définie récursivement comme suit :

- $\mathcal{IJ} \models a^v$ si $(\mathcal{I}(a), \mathcal{J}(a)) \in \langle v\rangle$, pour tout atome a^v de $(\Delta\mathcal{LG}, \models)$;
- $\mathcal{IJ} \models \neg\varphi$ si $\mathcal{IJ} \not\models \varphi$, pour $\varphi \in \Delta\mathcal{LG}$;
- $\mathcal{IJ} \models \varphi \wedge \psi$ si $\mathcal{IJ} \models \varphi$ et $\mathcal{IJ} \models \psi$, pour $\varphi, \psi \in \Delta\mathcal{LG}$;
- $\mathcal{IJ} \models \varphi \vee \psi$ si $\mathcal{IJ} \models \varphi$ ou $\mathcal{IJ} \models \psi$, pour $\varphi, \psi \in \Delta\mathcal{LG}$.

L'ensemble des modèles de $\varphi \in \Delta\mathcal{LG}$ est noté $\mathcal{M}(\varphi)$. Pour une base de connaissances $\mathcal{B}$ de $(\Delta\mathcal{LG}, \models)$, $\mathcal{M}(\mathcal{B}) = \bigcap\{\mathcal{M}(\varphi) \mid \varphi \in \mathcal{B}\}$. La relation de conséquence logique est définie comme d'habitude : $\varphi \models \psi$ si $\mathcal{M}(\varphi) \subseteq \mathcal{M}(\psi)$ et $\mathcal{B} \models \varphi$ si $\mathcal{M}(\mathcal{B}) \subseteq \mathcal{M}(\psi)$. De même, $\varphi \equiv \psi$ si $\mathcal{M}(\varphi) = \mathcal{M}(\psi)$. Finalement, φ est *satisfiable* (ou *cohérente*) si $\mathcal{M}(\varphi) \neq \varnothing$.

Dans l'introduction, un autre symbole de variation a été utilisé : $=r$ (pour $r \in \mathcal{G}$). Il a la sémantique suivante (pour $a \in \mathcal{V}$) : $\mathcal{IJ} \models a^{=r}$ si $\mathcal{I}(a) = \mathcal{J}(a) = r$. Par conséquent, $a^{=r}$ peut être vu comme une abréviation du cube $a^{r\bullet} \wedge a^{\bullet r} \wedge a^{+0}$.

4.3 Propriétés de base de $(\Delta\mathcal{LG}, \models)$

Proposition 1 *Étant donné la conjonction de deux littéraux ℓ_1 et ℓ_2 portant sur la même variable, un troisième littéral ℓ_3 peut parfois être déduit, qui est une conséquence logique des deux premiers. Dans ce cas, on aura $\ell_1 \wedge \ell_2 \equiv \ell_1 \wedge \ell_2 \wedge \ell_3$. On a, en effet, pour $r, s \in \mathcal{G}$:*

$$a^{r\bullet} \wedge a^{\bullet s} \models a^{+(s-r)} \qquad a^{+r} \wedge a^{s\bullet} \models a^{\bullet(r+s)}$$
$$a^{+r} \wedge a^{\bullet s} \models a^{(s-r)\bullet} \qquad a^{r\bullet} \wedge \neg a^{\bullet s} \models \neg a^{+(s-r)}$$
$$a^{\bullet r} \wedge \neg a^{s\bullet} \models \neg a^{+(r-s)} \qquad a^{+r} \wedge \neg a^{s\bullet} \models \neg a^{\bullet(r+s)}$$
$$a^{+r} \wedge \neg a^{\bullet s} \models \neg a^{(s-r)\bullet} \qquad a^{r\bullet} \wedge \neg a^{\bullet s} \models \neg a^{+(s-r)}$$
$$a^{r\bullet} \wedge \neg a^{+s} \models \neg a^{\bullet(r+s)}$$

Par ailleurs, pour $r, s \in \mathcal{G}$ avec $r \neq s$, les cubes suivants sont insatisfiables : $a^{r\bullet} \wedge a^{s\bullet}$, $a^{+r} \wedge a^{+s}$ et $a^{\bullet r} \wedge a^{\bullet s}$.
Enfin, pour $r, s, t \in \mathcal{G}$, si $t \neq s - r$ alors $a^{r\bullet} \wedge a^{\bullet s} \wedge a^{+t}$ est insatisfiable.

La définition et la proposition suivantes concernent les formes normales de $(\Delta\mathcal{LG}, \models)$ et s'appuient sur le fait que $(\mathcal{G}, +)$ est un groupe abélien infini. La section 6.2 étend ces résultats sous d'autres hypothèses.

Definition 1 (Formes normales FND et FNDCN) *Une formule de variations φ est sous FND (forme normale disjonctive) si c'est une disjonction de cubes: $\varphi = \theta_1 \vee \theta_2 \vee \ldots \vee \theta_p$ où θ_k est un cube de $(\Delta\mathcal{LG}, \models)$.*
Soit θ un cube de $(\Delta\mathcal{LG}, \models)$ avec comme seule variable a. θ est normal si une des conditions suivantes est respectée :

- (i) *$\theta = \bot$;*
- (ii) *θ contient exactement 3 littéraux positifs, des formes $a^{r\bullet}$, $a^{\bullet s}$ et a^{+t} avec $r, s, t \in \mathcal{G}$ et $t = s - r$, et il ne contient aucun littéral négatif ;*
- (iii) *θ contient exactement un littéral positif a^v et p littéraux négatifs ($p \in \mathbb{N}$) m_1, m_2, …, m_p utilisant un symbole de variation d'une forme différente de celle de v (p. ex. si $v = {+}r$ alors $m_k = \neg a^w$ où $w \neq {+}s$ pour tout $s \in \mathcal{G}$) ; de plus, pour tout littéral négatif $\neg a^w$, θ contient aussi la conséquence $\neg a^x$ de $a^v \wedge \neg a^w$ selon la proposition 1 (p. ex. si $v = {+}r$ et $w = s\bullet$, alors $x = \bullet(s + r)$) ;*
- (iv) *θ ne contient que des littéraux négatifs, de tout type.*

Soit θ un cube de $(\Delta\mathcal{LG}, \models)$. Pour a, une variable apparaissant dans θ, soit θ_a la conjonction des littéraux de θ ayant a comme variable. θ est un cube normal *si tout θ_a est normal, pour tout a, variable de θ.*
Une formule $\varphi \in \Delta\mathcal{LG}$ est sous FNDCN (FND avec cubes normaux) si elle est sous FND et si tous les cubes qui la composent sont normaux.

Proposition 2 *Toute formule $\varphi \in \Delta\mathcal{LG}$ peut être mise sous FND et sous FNDCN : il existe $\varphi_{\text{FND}} \in \Delta\mathcal{LG}$ (resp. $\varphi_{\text{FNDCN}} \in \Delta\mathcal{LG}$) qui est sous FND (resp. FNDCN) et telle que $\varphi \equiv \varphi_{\text{FND}}$ (resp. $\varphi \equiv \varphi_{\text{FNDCN}}$).*

Le fait que φ puisse être mise sous FND peut être prouvé comme cela se fait, p. ex., en logique propositionnelle. Le fait que tout cube peut être normalisé s'appuie en particulier sur la proposition 1.
Comme dans beaucoup de logique le problème de la conséquence logique et le problème de la satisfiabilité sont fortement liés dans $(\Delta\mathcal{LG}, \models)$. For exemple, $\{\varphi_1, \varphi_2\} \models \psi$ ssi $\varphi_1 \wedge \varphi_2 \wedge \neg\psi$ n'est pas satisfiable. Or, le problème

de satisfiabilité peut être résolu en utilisant les principes de la méthode des tableaux sémantiques (voir, p. ex., [10]), qui, pour une logique s'appuyant sur les connecteurs propositionnels revient à construire une formule sous FND représentée par un arbre dont chaque branche représente un cube. La proposition suivante est alors utile pour concevoir un algorithme fondé sur la méthode des tableaux sémantiques pour $(\Delta\mathcal{LG}, \models)$:

Proposition 3 *Les résultats ci-dessous peuvent être utilisés pour tester si une formule des variations est satisfiable :*

(a) *Pour $\varphi \in \Delta\mathcal{LG}$, soit φ_{FND} une formule sous FND telle que $\varphi_{\text{FND}} \equiv \varphi$. Alors, φ est satisfiable ssi φ_{FND} est satisfiable.*

(b) *Soit $\varphi \in \Delta\mathcal{LG}$ une formule sous FND : $\varphi = \theta_1 \vee \theta_2 \ldots \vee \theta_p$ où chaque θ_k est un cube de $(\Delta\mathcal{LG}, \models)$. Alors φ est satisfiable ssi au moins un des θ_k est satisfiable ($k \in [\![1, p]\!]$).*

(c) *Soit θ un cube de $(\Delta\mathcal{LG}, \models)$. Pour a, variable apparaissant dans θ, soit θ_a la conjunction des littéraux de θ ayant a comme variable. Alors, θ est satisfiable ssi chaque θ_a est satisfiable.*

(d) *Un cube normal θ_a avec une seule variable est satisfiable ssi $\theta_a \neq \bot$ (cela signifie que, si une des conditions (ii), (iii) et (iv) de la définition 1 est vérifiée, alors θ_a est satisfiable).*

4.4 Opérateurs sur $(\Delta\mathcal{LG}, \models)$

Cette section présente différents opérateurs sur la logique $(\Delta\mathcal{LG}, \models)$ qui seront utilisés dans les sections suivantes. Chacun d'entre eux a une définition sémantique et vérifie le principe d'indépendance à la syntaxe : si une formule apparaissant dans la définition d'un opérateur est représentée par une formule équivalente, alors le résultat de l'application de l'opérateur est inchangé à l'équivalence logique près. Chacun de ces opérateurs est défini soit à travers une définition classique, soit dans une proposition qui affirme son existence. Les preuves de ces propositions sont données dans [4]. Puis, la façon dont ces opérateurs sont calculés est décrite dans le cas où les formules impliquées sont des cubes.

Une formule de variations φ peut être vue comme une représentation d'une relation binaire sur Ω puisque $\mathcal{M}(\varphi) \subseteq \Omega \times \Omega$. La proposition suivante montre que l'inverse d'une relation $\mathcal{M}(\varphi)$ peut être représentée par une formule de variations $\overleftarrow{\varphi}$.

Proposition 4 *Pour toute $\varphi \in \Delta\mathcal{LG}$, il existe une formule de variations notée $\overleftarrow{\varphi}$ telle que*

$$\mathcal{M}\left(\overleftarrow{\varphi}\right) = \{\mathcal{JI} \mid \mathcal{IJ} \in \mathcal{M}(\varphi)\}$$

En d'autres termes, $\mathcal{JI} \models \overleftarrow{\varphi}$ ssi $\mathcal{IJ} \models \varphi$.

Le calcul de $\overleftarrow{\varphi}$ est linéaire : il consiste à remplacer chaque occurrence d'un symbole de variations v par $\overleftarrow{v}$ défini comme suit :

$$\overleftarrow{\text{+}r} = \text{-}r \qquad \overleftarrow{r\bullet} = \bullet r \qquad \overleftarrow{\bullet r} = r\bullet \qquad \text{(pour } r \in \mathcal{G})$$

Par exeample, $\overleftarrow{a^{+1} \wedge a^{\bullet 2} \wedge c^{-2}} = a^{-1} \wedge a^{2\bullet} \wedge c^{+2}$.

Proposition 5 *Pour tous $\alpha, \beta \in \mathcal{LG}$ il existe une formule de $(\Delta\mathcal{LG}, \models)$ notée $\alpha \triangleright \beta$ et qui se lit "α devient β", telle que $\mathcal{M}(\alpha \triangleright \beta) = \mathcal{M}(\alpha) \times \mathcal{M}(\beta)$.*
En d'autres termes, $\mathcal{IJ} \models \alpha \triangleright \beta$ ssi $\mathcal{I} \models \alpha$ et $\mathcal{J} \models \beta$.

À présent, considérons $\bigwedge_{i=1}^{p} \ell_i$ et $\bigwedge_{j=1}^{q} m_j$, deux cubes de $(\mathcal{LG}, \models)$: $\ell_1, \ell_2, \ldots, \ell_p, m_1, m_2, \ldots, m_q$ sont $p+q$ littéraux. L'application de $\triangleright$ sur ces cubes peut être calculée ainsi :

$$\left(\bigwedge_{i=1}^{p} \ell_i\right) \triangleright \left(\bigwedge_{j=1}^{q} m_j\right) \equiv \left(\bigwedge_{i=1}^{p} \ell_i \triangleright \top\right) \wedge \left(\bigwedge_{j=1}^{q} \top \triangleright m_j\right)$$

Enfin, il est suffisant de définir $\alpha \triangleright \beta$ quand l'un parmi α et β est un littéral et l'autre est $\top$ (pour $a \in \mathcal{V}$ et $r \in \mathcal{G}$) :

$$(a = r) \triangleright \top \equiv a^{r\bullet} \qquad \neg(a = r) \triangleright \top \equiv \neg a^{r\bullet}$$
$$\top \triangleright (a = r) \equiv a^{\bullet r} \qquad \top \triangleright \neg(a = r) \equiv \neg a^{\bullet r}$$

L'opérateur $\triangleright$ peut être utilisé pour l'apprentissage de connaissances d'adaptation utilisant l'heuristique des différences introduite en section 1. Considérons les cas sources suivants exprimés dans $(\mathcal{LG}, \models)$ avec $\mathcal{G} = \mathbb{Z}$ et où $\mathcal{V}_\mathcal{P} = \{a, b\}$ et $\mathcal{V}_\mathcal{S} = \{c, d\}$.

$$\begin{aligned} \mathtt{x}^1 \wedge \mathtt{y}^1 &= (a=1) \wedge (b=2) \wedge (c=3) \wedge (d=4) \\ \mathtt{x}^2 \wedge \mathtt{y}^2 &= (a=2) \wedge (b=0) \wedge (c=2) \wedge (d=6) \\ \mathtt{x}^3 \wedge \mathtt{y}^3 &= (a=3) \wedge (b=1) \wedge (c=1) \wedge (d=8) \end{aligned}$$

La variation du cas $(\mathtt{x}^k, \mathtt{y}^k)$ au case $(\mathtt{x}^\ell, \mathtt{y}^\ell)$ est représentée par $\Delta^{k\ell} = (\mathtt{x}^k \wedge \mathtt{y}^k) \triangleright (\mathtt{x}^\ell \wedge \mathtt{y}^\ell)$. Selon ce qui est présenté ci-dessus, comme les cas sources $\mathtt{x}^s \wedge \mathtt{y}^s$ sont des conjonctions de littéraux positifs de $(\mathcal{LG}, \models)$, le résultat $\Delta^{k\ell}$ est une conjonction d'atomes des formes $a^{r\bullet}$ et $a^{\bullet r}$ ($r \in \mathcal{G}$). En utilisant la première conséquence logique de la proposition 1, les atomes des formes a^{+r} sont ajoutés aux $\Delta^{k\ell}$, qui mettent en évidence les variations. Par exemple

$$\begin{aligned} \Delta^{12} \equiv\ & a^{+1} \wedge a^{1\bullet} \wedge a^{\bullet 2} \wedge b^{-2} \wedge b^{2\bullet} \wedge b^{\bullet 0} \\ & \wedge c^{-1} \wedge c^{3\bullet} \wedge c^{\bullet 2} \wedge d^{+2} \wedge d^{4\bullet} \wedge d^{\bullet 6} \end{aligned}$$

Utiliser un algorithme d'extraction de motifs fréquents [7] sur l'ensemble d'apprentissage $\{\Delta^{k\ell}\}_{k\ell}$ permet de mettre en évidence des atomes a^v apparaissant dans des sous-ensembles de cet ensemble d'apprentissage. Par exemple, pour $(k, \ell) \in \{(1,2), (2,3)\}$, le motif $\{a^{+1}, c^{-1}, d^{+2}\}$ peut être mis en évidence et permet de créer la formule $\varphi = a^{+1} \wedge c^{-1} \wedge d^{+2}$ qui peut être utilisée pour représenter une règle d'adaptation, comme on le verra plus loin. On peut noter que $\overleftarrow{\varphi}$ peut également être apprise : elle correspond à $(k, \ell) \in \{(2,1), (3,2)\}$.

Proposition 6 *Pour toute $\varphi \in \Delta\mathcal{LG}$, il existe $\mathsf{G}(\varphi) \in \mathcal{LG}$ et $\mathsf{D}(\varphi) \in \mathcal{LG}$ (les projections* gauche *et* droite *de φ) telles que :*

$$\mathcal{M}(\mathsf{G}(\varphi)) = \{\mathcal{I} \in \Omega \mid \textit{il existe } \mathcal{J} \in \Omega, \mathcal{IJ} \models \varphi\} \textit{ et}$$
$$\mathcal{M}(\mathsf{D}(\varphi)) = \{\mathcal{J} \in \Omega \mid \textit{il existe } \mathcal{I} \in \Omega, \mathcal{IJ} \models \varphi\}$$

Si α *et* β *sont deux formules satisfiables de* $(\Delta\mathcal{LG}, \models)$ *alors* $\mathsf{G}(\alpha \triangleright \beta) \equiv \alpha$ *et* $\mathsf{D}(\alpha \triangleright \beta) \equiv \beta$.

Le calcul des projections gauche et droite d'un cube peut être effectué en mettant un cube sous forme normale et en utilisant les équivalences suivantes :

$$\text{si } \varphi = \bigwedge_{k=1}^{p} \ell_k \text{ est un cube normal de } (\Delta\mathcal{LG}, \models),$$
$$\text{alors } \mathsf{G}(\varphi) \equiv \bigwedge_{k=1}^{p} \mathsf{G}(\ell_k) \text{ et } \mathsf{D}(\varphi) \equiv \bigwedge_{k=1}^{p} \mathsf{D}(\ell_k)$$
$$\mathsf{G}(a^{+r}) \equiv \mathsf{D}(a^{+r}) \equiv \mathsf{G}(a^{\bullet r}) \equiv \mathsf{D}(a^{r\bullet}) \equiv \top$$
$$\mathsf{G}(\neg a^{+r}) \equiv \mathsf{D}(\neg a^{+r}) \equiv \mathsf{G}(\neg a^{\bullet r}) \equiv \mathsf{D}(\neg a^{r\bullet}) \equiv \top$$
$$\mathsf{G}(a^{r\bullet}) \equiv (a = r) \quad \mathsf{G}(\neg a^{r\bullet}) \equiv \neg(a = r)$$
$$\mathsf{D}(a^{\bullet r}) \equiv (a = r) \quad \mathsf{D}(\neg a^{\bullet r}) \equiv \neg(a = r)$$

Definition 2 *Soit* $\varphi \in \Delta\mathcal{LG}$ *et* $a \in \mathcal{V}$. a *est dite* variable invariante *de* φ *s'il existe* $\psi \in \Delta\mathcal{LG}$ *telle que* $\varphi \equiv \psi$ *et* $a \notin \mathcal{V}(\psi)$. *L'ensemble des variables invariantes de* φ *est noté* $\mathsf{Inv}(\varphi)$. *On a donc* $\mathcal{V} \setminus \mathcal{V}(\varphi) \subseteq \mathsf{Inv}(\varphi) \subseteq \mathcal{V}$.

Une variable d'un cube normal est invariante ssi $a \notin \mathcal{V}(\theta)$. En revanche, il existe des formules de variation φ telles que $\mathsf{Inv}(\varphi) \neq \mathcal{V} \setminus \mathcal{V}(\varphi)$. Par exemple, avec $\varphi = (a^{+1} \wedge a^{+2}) \vee b^{4\bullet}$, on a $a \in \mathcal{V}(\varphi)$ mais $a \in \mathsf{Inv}(\varphi)$ puisque $\varphi \equiv b^{4\bullet}$.

5 Représenter des règles d'adaptation

Le formalisme $(\Delta\mathcal{LG}, \models)$ a été conçu pour représenter des variations entre cas sources en s'inspirant de notations issues de l'apprentissage de connaissances d'adaptation. Le résultat d'un tel processus d'apprentissage est un ensemble de règles d'adaptation qui peuvent être représentées à l'aide de cette logique et c'est ce qui est examiné dans cette section.

5.1 Exemple motivant l'introduction de l'opérateur *ceteris paribus*

Considérons à nouveau la règle d'adaptation dans le domaine culinaire qui a été présentée en introduction. Écrite de façon conjonctive, elle correspond au cube suivant de $(\Delta\mathcal{LG}, \models)$, pour $\mathcal{G} = \mathbb{Q}$: $\varphi = \texttt{dessert}^{=1} \wedge \texttt{\#œufs}^{-2} \wedge \texttt{\#bananes}^{+1}$. À présent, soit une recette de dessert avec 2 œufs, aucune banane, 50 grammes de farine et 20 grammes de beurre. Elle peut être représentée par la formule de $(\mathcal{LG}, \models)$ $\alpha = (\texttt{dessert} = 1) \wedge (\texttt{\#œufs} = 2) \wedge (\texttt{\#bananes} = 0) \wedge (\texttt{farine}_{\texttt{gramme}} = 50) \wedge (\texttt{beurre}_{\texttt{gramme}} = 20)$ (en ne considérant pas d'autres variables). La question qui se pose est comment la formule φ peut-elle être appliquée à α ?

Pour traiter cette question, l'idée est d'abord de définir la formule des variations θ qui représente la transformation de α par φ, sachant que le résultat final attendu sera alors $\beta = \mathsf{D}(\theta)$. Soit $\mathcal{IJ}$ un modèle de θ. Cette transformation "part" de α donc $\mathcal{I}$ devrait être un modèle de α, ce qui équivaut à écrire que (a) $\mathcal{IJ} \models \alpha \triangleright \top$. De plus, θ doit spécialiser φ, i.e. $\theta \models \varphi$, ce qui entraîne que (b) $\mathcal{IJ} \models \varphi$. Si (a) et (b) étaient les seules conditions à donner sur $\mathcal{IJ}$, cela impliquerait que $\theta \equiv (\alpha \triangleright \top) \wedge \varphi$ et donc, dans cet exemple, le résultat de l'application sur α de φ serait $\beta = (\texttt{dessert} = 1) \wedge (\texttt{\#œufs} = 0) \wedge (\texttt{\#bananes} = 1)$. Pour les variables $\texttt{farine}_{\texttt{gramme}}$ et $\texttt{beurre}_{\texttt{gramme}}$, cela soulève le problème suivant : elles n'apparaissent pas dans β.En effet, ces deux variables sont invariantes par φ. Pour traiter cela, le postulat suivant est posé : si une variable est invariante dans φ alors, l'application sur α de φ doit laisser cette variable inchangée. Cela suggère une troisième condition : (c) si $a \in \mathsf{Inv}(\varphi)$ alors a est interprétée dans β de la même façon qu'elle est interprétée dans α ; formellement, $\{\mathcal{I}(a) \mid \mathcal{IJ} \in \mathcal{M}(\theta)\} = \{\mathcal{J}(a) \mid \mathcal{IJ} \in \mathcal{M}(\theta)\}$. Si α, comme dans cet exemple, est un cube, la condition (c) signifie simplement que si un littéral ℓ avec la variable a apparaît dans α alors β contient le même littéral ℓ. Une autre manière de formuler cela consiste à ajouter à φ une conjonction avec l'atome a^{+0} pour toute $a \in \mathsf{Inv}(\varphi)$, ce qui mène à une formule notée $\mathsf{cp}(\varphi)$, définie formellement ci-dessous, dans la définition 3.

Par conséquent, en prenant en compte la condition (c) amène à la transformation θ et à l'application β :

$$\theta = \texttt{dessert}^{1\bullet} \wedge \texttt{dessert}^{=1} \wedge \texttt{\#œufs}^{2\bullet} \wedge \texttt{\#œufs}^{-2} \wedge \texttt{\#bananes}^{0\bullet} \wedge \texttt{\#bananes}^{+1} \wedge \texttt{farine}^{50\bullet}_{\texttt{gramme}} \wedge \texttt{farine}^{+0}_{\texttt{gramme}} \wedge \texttt{beurre}^{20\bullet}_{\texttt{gramme}} \wedge \texttt{beurre}^{+0}_{\texttt{gramme}}$$
$$\beta = (\texttt{dessert} = 1) \wedge (\texttt{\#œufs} = 0) \wedge (\texttt{\#bananes} = 1) \wedge (\texttt{farine}_{\texttt{gramme}} = 50) \wedge (\texttt{beurre}_{\texttt{gramme}} = 20)$$

À présent, la définition de $\mathsf{cp}(\varphi)$ peut être donnée :

Definition 3 *Soit* $\varphi \in \Delta\mathcal{LG}$ *et* $\mathcal{W} \subseteq \mathcal{V}$ *avec* $\mathcal{W} \neq \varnothing$. *Le* ceteris paribus [1] *de* φ *pour* $\mathcal{W}$ *est défini par*

$$\mathsf{cp}_{\mathcal{W}}(\varphi) = \varphi \wedge \bigwedge \{a^{+0} \mid a \in \mathsf{Inv}(\varphi) \cap \mathcal{W}\}$$

$\mathsf{cp}(\varphi)$ *est une abréviation de* $\mathsf{cp}_{\mathcal{V}}(\varphi)$ *(toutes les variables sont considérées dans* $\mathsf{cp}(\varphi)$*).*

5.2 Appliquer une formule de variation sur une formule statique

Dans ce qui précède, on a mis en évidence la transformation de α par φ — $\theta = (\alpha \triangleright \top) \wedge \mathsf{cp}(\varphi)$ — et le résultat de cette transformation — $\beta = \mathsf{D}(\theta)$. Une contrainte peut être ajoutée au résultat de cette transformation. Si cette contrainte est représentée par $\gamma \in \mathcal{LG}$, alors il est possible de définir la transformation et l'application d'une formule de variation sur une formule statique en tenant compte de γ :

Definition 4 *Soit* $\alpha, \gamma \in \mathcal{LG}$ *et* $\varphi \in \Delta\mathcal{LG}$. *La transformation sur* α *de* φ *avec contrainte* γ *est la formule* $\mathsf{transf}(\alpha, \varphi, \gamma) \in \Delta\mathcal{LG}$ *définie par*

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

1. L'expression latine *ceteris paribus* signifie "toute chose étant égale par ailleurs".

L'application sur α *de* φ *avec contrainte* γ *est la formule* $\mathsf{appl}(\alpha, \varphi, \gamma) \in \mathcal{LG}$ *définie par*

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

La transformation (resp. application) de α sur φ sans contrainte est simplement $\mathsf{transf}(\alpha, \varphi, \top)$ (resp. $\mathsf{appl}(\alpha, \varphi, \top)$).

5.3 Représentation des règles d'adaptation

Soit $\mathtt{adPb} = ((\mathtt{x}^s, \mathtt{y}^s), \mathtt{x}^{\mathtt{cible}}) \in \mathtt{BC} \times \mathcal{P}$ un problème d'adaptation et $\mathtt{ar}$, une règle d'adaptation. Trois questions peuvent être posées : (Q1) $\mathtt{ar}$ est-elle applicable pour résoudre $\mathtt{adPb}$? (Q2) Si la réponse à la question précédente est oui, quel est le résultat $\mathtt{y}^{\mathtt{cible}} \in \mathcal{S}$ de l'application de $\mathtt{ar}$ sur $\mathtt{adPb}$? (Q3) À quel point l'assertion "$\mathtt{y}^{\mathtt{cible}}$ résout $\mathtt{x}^{\mathtt{cible}}$" est-elle plausible ? Pour répondre aux questions (Q1) et (Q2), une formule de variation φ est utilisée, comme expliqué ci-dessous. La réponse proposée à la question (Q3) utilise un nombre strictement positif $\mathtt{coût}$ qui caractérise l'effort d'adaptation, i.e. le risque pris en appliquant $\mathtt{ar}$ sur le problème d'adaptation (plus haut est ce coût, plus il y a de risque que l'adaptation donne un résultat incorrect).
Ci-dessous, une proposition de formalisation d'une règle d'adaptation est donnée, qui explique comment les questions (Q1) à (Q3) sont traitées :

Definition 5 *Une* règle d'adaptation *est un couple* $\mathtt{ar} = (\varphi, \mathtt{coût})$ *où* φ *est une formule de variation et* $\mathtt{coût}$ *est un rationnel strictement positif appelé* coût *de* $\mathtt{ar}$.
Soit $\mathtt{adPb} = ((\mathtt{x}^s, \mathtt{y}^s), \mathtt{x}^{\mathtt{cible}})$ *un problème d'adaptation. L'application de* $\mathtt{ar}$ *sur* $\mathtt{adPb}$ *est la formule statique suivante :*

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

La règle $\mathtt{ar}$ *est dite* applicable *sur* $\mathtt{adPb}$ *si* β *est satisfiable. Si* $\mathtt{ar}$ *est applicable sur* $\mathtt{adPb}$ *alors l'*application *de* $\mathtt{ar}$ *sur* $\mathtt{adPb}$ *donne une solution satisfiable* $\mathtt{y}^{\mathtt{cible}}$ *telle que* $\beta \equiv \mathtt{x}^{\mathtt{cible}} \wedge \mathtt{y}^{\mathtt{cible}}$.
La valeur $\mathtt{coût}$ *caractérise la fiabilité de l'assertion "*$\mathtt{y}^{\mathtt{cible}}$ *résout* $\mathtt{x}^{\mathtt{cible}}$*" (plus petit ce coût est, plus fiable cette assertion est supposée être).*

Pour reprendre l'exemple de la section 5.1, avec $\mathtt{x}^s \wedge \mathtt{y}^s = \alpha$ et $\mathtt{x}^{\mathtt{cible}} = (\mathtt{dessert} = 1) \wedge (\#\mathtt{œufs} = 0)$ (i.e. "Je veux un dessert sans œufs."), $\mathtt{ar} = (\varphi, \mathtt{coût})$ est applicable au problème d'adaptation $\mathtt{adPb} = ((\mathtt{x}^s, \mathtt{y}^s), \mathtt{x}^{\mathtt{cible}})$ et le résultat est $\mathtt{y}^{\mathtt{cible}} \equiv (\#\mathtt{bananes} = 1) \wedge (\mathtt{farine}_{\mathtt{gramme}} = 50) \wedge (\mathtt{beurre}_{\mathtt{gramme}} = 20)$.

6 Situations dans lesquelles $(\mathcal{G}, +)$ n'est pas un groupe abélien infini

Dans les sections précédentes, $(\mathcal{G}, +)$ est supposé être un groupe abélien infini. Cette section examine d'autres situations conduisant à des modifications dans la vérification de la satisfiabilité des formules de $(\Delta\mathcal{LG}, \models)$. Deux situations sont envisagées : tout d'abord le cas où $(\mathcal{G}, +)$ est un groupe abélien fini, ensuite le cas où $\mathcal{G}$ est un sous-ensemble d'un groupe abélien $(\mathcal{H}, +)$, mais n'a pas une structure de groupe, donc en particulier $\mathcal{G}$ n'est pas stable pour $+$ (autrement dit il existe $r, s \in \mathcal{G}$ tels que $r + s \in \mathcal{H} \setminus \mathcal{G}$) ou pour l'opposé (i.e. il existe $r \in \mathcal{G}$ tel que $-r \in \mathcal{H} \setminus \mathcal{G}$). D'autres situations (comme par exemple le cas où $(\mathcal{G}, +)$ est un groupe non abélien) ne sont pas envisagées ici.
La notion principale considérée ici est celle de cube normal : l'idée derrière cette notion est qu'un cube normal est soit égal à $\bot$, soit satisfiable. Ainsi, le test de satisfiabilité sur les cubes normaux devrait consister en un simple test $\theta \neq \bot$ au niveau syntaxique. Pour les groupes infinis, cela est détaillé dans la section 4.3 : définition 1 et proposition 2.

6.1 $(\mathcal{G}, +)$ est un groupe fini

Quand $(\mathcal{G}, +)$ est un groupe abélien infini, tout cube contenant uniquement des littéraux positifs est satisfiable. Ceci n'est plus le cas si $\mathcal{G}$ est fini. Par exemple, le cube $\neg a^{0\bullet} \wedge \neg a^{1\bullet}$ est satisfiable dans, par exemple, $\mathcal{G} = \mathbb{Z}$, mais pas dans $\mathcal{G} = \mathbb{Z}/2\mathbb{Z}$.
La solution consiste à faire disparaître les littéraux négatifs dans les cubes normaux, ce qui est possible puisque, si $\mathcal{G}$ est fini, tout littéral négatif est équivalent à une disjonction de littéraux positifs :

$$\bigvee_{r \in A} \neg a^{r\bullet} \equiv \bigvee_{s \in \mathcal{G} \setminus A} a^{s\bullet} \quad \neg a^{+r} \equiv \bigvee_{s \in \mathcal{G} \setminus \{r\}} a^{+s} \quad \neg a^{\bullet r} \equiv \bigvee_{s \in \mathcal{G} \setminus \{r\}} a^{\bullet s} \tag{1}$$

Ainsi, après mise de $\varphi \in \Delta\mathcal{LG}$ en forme normale négative (i.e. formule dans laquelle $\neg$ n'apparaît que devant des atomes) et application des équivalences (1) de gauche à droite, la formule obtenue ne contient plus de $\neg$. Appliquer alors la distributivité de $\wedge$ sur $\vee$ donne une FND de cubes contenant seulement des conjonctions de littéraux positifs. On peut alors prouver qu'un tel cube est incohérent si et seulement si il existe $a \in \mathcal{V}$ tel que θ contient trois atomes de la forme $a^{r\bullet}$, a^{+s} et $a^{\bullet t}$ avec $r + s \neq t$. La définition suivante s'appuie sur ces considérations :

Definition 6 (Cubes normaux lorsque $\mathcal{G}$ est fini) *Si* $(\mathcal{G}, +)$ *est un groupe fini, alors un cube* θ *de* $(\Delta\mathcal{LG}, \models)$ *est soit* $\bot$*, soit est une conjonction d'atomes tel que, pour chaque variable* a *apparaissant dans* θ *:*

- *soit* a *apparaît dans un unique atome de* θ *;*
- *soit* a *apparaît dans exactement trois atomes de* θ*, de la forme* $a^{r\bullet}$*,* a^{+s} *et* $a^{\bullet t}$ *avec* $r + s = t$*.*

Proposition 7 *si* $(\mathcal{G}, +)$ *est un groupe abélien fini, un cube normal de* $(\Delta\mathcal{LG}, \models)$ *est satisfiable si et seulement si il est différent de* $\bot$*.*

6.2 $\mathcal{G}$ est un sous-ensemble infini d'un groupe $(\mathcal{H}, +)$

Dans cette situation, la première idée a été de considérer la transformation d'un cube θ de $(\Delta\mathcal{LG}, \models)$ en un cube normal θ' de $(\Delta\mathcal{LH}, \models)$ et d'examiner les nouveaux atomes

produits, gardant ceux appartenant à $(\Delta\mathcal{LG}, \models)$ et enlevant les autres.

Cette approche peut s'appliquer dans certaines situations, mais pas dans toutes comme le contre-exemple qui suit le montre. Par exemple, soit $\mathcal{G} = 2\mathbb{Z} + 1$ (l'ensemble des entiers impairs) : $\mathcal{G} \subseteq \mathcal{H}$ avec $\mathcal{H} = \mathbb{Z}$. Suivre l'idée présentée ci-dessus peut mener, par exemple, à considérer l'atome a^{+1} qui est, effectivement un atome de $(\Delta\mathcal{LG}, \models)$ mais qui n'est pas satisfiable dans cette logique.

Comme ce point n'est pas prioritaire dans nos travaux, nous laissons cela pour d'hypothétiques et lointains travaux futurs.

6.3 $\mathcal{G}$ est un sous-ensemble fini d'un groupe

Soit $\mathcal{G}$ un ensemble fini tel que $\mathcal{G} \subseteq \mathcal{H}$ où $(\mathcal{H}, +)$ est un groupe abélien. Cette situation combine les difficultés des sections 6.1 et 6.2. Elle ne sera pas développée en détails ici.

7 Discussion et conclusion

Cet article a présenté une approche formelle pour représenter des variations entre cas et des règles d'adaptation.

Ce travail théorique peut être prolongé dans différentes directions.

Une première direction serait de confronter cette approche à des applications pratiques et/ou des *benchmarks*, en utilisant un système d'apprentissage de connaissances d'adaptations pour alimenter `CA`.

Ce travail a principalement considéré deux conteneurs de connaissances d'un système de raisonnement à partir de cas : `BC` et `CA` (les connaissances pour la remémoration sont aussi considérées dans [4]). Peu d'attention a été apportée aux connaissances du domaine, représentées par un ensemble de formules statiques `CD`. Pour les connaissances statiques (en particulier les cas), cela consiste principalement en la définition d'une inférence modulo `CD` ($\alpha \models_{\texttt{CD}} \beta$ si $\texttt{CD} \cup \{\alpha\} \models \beta$, α est `CD`-satisfiable si $\texttt{CD} \cup \{\alpha\}$ est satisfiable, etc.). Comment les connaissances statiques représentées par `CD` peuvent-elles utilisées pour définir une inférence sur $(\Delta\mathcal{LG}, \models)$ et des règles d'adaptation est une question qui reste à explorer.

Olaaaf est un moteur d'adaptation fondé sur la révision des croyances dans une logique dont les atomes sont des contraintes linéaires et qui est propositionnellement close [2]. La théorie sur laquelle il est bâti est essentiellement centrée sur les connaissances du domaine `CD` et sur une mesure de dissimilarité `diss` entre interprétations. Certaines règles d'adaptation sont "simulées" dans Olaaaf en ajoutant des connaissances à `CD`, et/ou en modifiant `diss`, bien que cet aspect d'Olaaaf n'ait pas encore été étudié en détail. À l'inverse, l'approche proposée dans cet article est centrée sur `CA` (et sa représentation). Une piste de travail digne d'investigation consiste en la découverte d'une vision intégrée de ces deux approches centrées sur la logique pour l'adaptation.

Références

[1] M. d'Aquin, F. Badra, S. Lafrogne, J. Lieber, A. Napoli, and L. Szathmary. Case Base Mining for Adaptation Knowledge Acquisition. In M. M. Veloso, editor, *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI'07)*, pages 750–755. Morgan Kaufmann, Inc., 2007.

[2] E. Diebold, Y. Kabrit, A. Kril, J. Lieber, P. Malvaud, E. Nauer, and J. Sipp. Olaaaf: A General Adaptation Prototype. In J. A. Recio-Garcia, M. G. Orozco del Castillo, and D. Bridge, editors, *Case-Based Reasoning Research and Development*, volume 14775 of *Lecture Notes in Computer Science*, pages 223–239, Merida, Mexico, July 2024. J. A. Recio-Garcia and M. G. Orozco-del-Castillo and D. Bridge, Springer Nature Switzerland.

[3] N. François and J. Lieber. Appliquer la logique des variations propositionnelles à la représentation de règles d'adaptation pour le raisonnement à partir de cas. In Jean-Guy Mailly, François Schwarzentruber, and Anaëlle Wilczynski, editors, *Journées d'intelligence artificielle fondamentale – Plateforme Intelligence Artificielle*, page 12, La Rochelle, France, July 2024. Jean-Guy Mailly and François Schwarzentruber and Anaëlle Wilczynski.

[4] N. François and J. Lieber. A Knowledge Representation Approach for Reasoning with Adaptation Rules (extended version). Technical report, Loria, 2025. (This report can be found at https://hal.science/hal-04961604).

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

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

[7] J. M. Luna, Ph. Fournier-Viger, and S. Ventura. Frequent itemset mining: A 25 years review. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 9(6):e1329, 2019.

[8] E. Nauer, J. Lieber, and M. d'Aquin. Lazy Adaptation Knowledge Learning Based on Frequent Closed Itemsets. In *Case-Based Reasoning Research and Development*, volume 14141 of *Lecture Notes in Computer Science*, pages 309–324, Aberdeen, United Kingdom, July 2023. Springer Nature Switzerland.

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

[10] R. M. Smullyan. *First-order logic*. Courier Corporation, 1995.