Comparaison de relations d'inférence non monotone : etude de complexité

Claudette Cayrol, Marie-Christine Lagasquie-Schiex

Comparaison de relations d'inférence non monotone : étude de complexité

Claudette CAYROL
Marie-Christine LAGASQUIE-SCHIEX

Rapport IRIT / 93.23. R

Révision numéro 1 de mai 1994.

Comparaison de relations d'inférence non monotone : étude de complexité

Claudette Cayrol
Marie-Christine Lagasquie-Schiex
Institut de Recherche en Informatique de Toulouse
Université Paul Sabatier
118 route de Narbonne
31062 Toulouse Cedex
FRANCE

e-mail : {testemal, lagasq}@irit.fr

Résumé

Ce rapport présente des travaux de recherche menés à l'IRIT ("équipe Intelligence Artificielle et Robotique – Communication, Décision, Raisonnement").

L'objectif de ces travaux est l'étude de la complexité de certains processus d'inférence non-monotone sur une base de croyances E ordonnée ou pas et pouvant être initialement inconsistante.

Nous commençons donc par rappeler certains mécanismes de génération des ensembles de croyances consistants issus de E et pouvant être ordonnés ou pas.

Puis dans une seconde partie, nous citons les divers principes d'inférence utilisables sur de tels ensembles.

Au croisement de chaque mécanisme de génération et de chaque principe d'inférence se trouve une relation d'inférence du type $(E, <) \mid\!\sim \Phi$ qu'il faut étudier, en particulier du point de vue de la complexité, ce qui constitue la troisième et dernière partie de ce rapport.

Les résultats obtenus sont peu encourageants, toutes nos relations d'inférence non-monotone ayant, dans le pire des cas, une complexité largement supérieure à la complexité de l'inférence monotone.

TABLE DES RÉVISIONS

Numéro de la révision	Date de la révision	Objet de la révision
Version initiale	Septembre 1993	
1	Mai 1994	Suite aux remarques de différents relecteurs (Pr. Michel Cayrol, Pr. Martin Cooper, Mr. Jérôme Lang, Pr. Daniel Lehmann, Mr. Thomas Schiex) et à la poursuite de notre étude, ce rapport a été modifié comme suit : ▪ Prise en compte de l'erratum fourni avec la version initiale. ▪ Correction d'une erreur dans la démonstration de complétude de 1/STRATE-EXI-BO. ▪ Étude d'un cas particulier supplémentaire : les formules CNF. ▪ Étude de 3 nouvelles relations d'inférence : UNI-CAR, EXI-CAR, ARG-CAR (ordre basé sur la cardinalité appliqué à une base de formules non ordonnée) et des cas particuliers associés. ▪ Nouveaux résultats sur la relation d'inférence UNI-LEX, en particulier un algorithme Δ_2^p et une preuve de complétude. ▪ Preuve de complétude pour 1/STRATE. ▪ Rajout de quelques précisions sur la notion de multi-ensembles et son utilisation dans la démonstration de UNI (EXI, ARG) - CAR (LEX) -HORN. ▪ Rajout d'une nouvelle annexe : "Tableau récapitulatif des transformations polynomiales utilisées". ▪ Raffinage des résultats sur ARG-E-HORN. ▪ Quelques modifications de présentation.

Plan :

1. Introduction
   - quels formalismes comparer (situer parmi tous les formalismes, ceux qui vont être comparés),
   - intérêt de cette comparaison,
   - méthode utilisée pour comparer,
   - introduction des chapitres suivants.
2. Que compare-t-on ?
   - les ensembles de croyances éventuellement ordonnés :
     - sous-bases consistantes maximales pour l'inclusion (ordre "INCLUSION BASED PREFERENCE") :
       - sous-théories de [Bre89b],
       - thèses de [Cay90], [Cay92] et [CRS92] (ordres démocratique, élitiste),
       - sous-bases fortement maximales consistantes de la logique possibiliste [DLP91],
     - sous-bases consistantes avec ordre lexicographique de [BCD$^+$93],
     - extensions de la logique des défauts de [Rei80],
     - sous-bases consistantes avec ordre "BEST-OUT" [BCD$^+$93],
     - ensembles issus de [GM94].
   - pour chaque cas cité précédemment, il faut :
     - donner la définition des ensembles de croyances (mécanisme de décomposition de la base initiale en sous-bases représentant les ensembles de croyances étudiés),
     - donner la définition des ordres appliqués ou non à ces ensembles.
3. Les principes d'inférence à appliquer :
   - ceux cités par les concepteurs du formalisme étudié,
   - ceux cités par [PL92] et [GM94], et éventuellement applicables aux cas étudiés.
4. Étape préliminaire : étude de complexité.

   Au croisement de chaque cas et de chaque principe d'inférence se trouve une relation d'inférence du type $(E, <) \mid\sim \Phi$ qu'il faut étudier, en particulier du point de vue de la complexité (voir [Neb91], [Joh90], [GJ79]).
5. Conclusion : synthèse des résultats.

Table des matières

Chapitre 1

Introduction

Un des buts de l'intelligence artificielle est la manipulation de connaissances. Tant que ces connaissances sont cohérentes, la logique classique s'avère être un outil parfait. Malheureusement, dès qu'il s'agit de traiter des connaissances incomplètes, incertaines ou inconsistantes (on parle alors de croyances), la logique classique ne s'applique plus (par exemple, à partir d'une information inconsistante en logique classique, on peut déduire n'importe quoi).

Or la manipulation de telles informations est capitale si on veut pouvoir modéliser un raisonnement de "sens commun". Nous avons donc 2 motivations :

- pouvoir traiter les inconsistances éventuelles, donc gérer des conflits,
- pouvoir rendre compte d'un fait incontestable : toutes les informations n'ont pas la même importance (par exemple, on peut faire intervenir le degré de certitude en une information).

Ce raisonnement appelé *raisonnement non monotone* est un des sujets de recherche les plus riches en intelligence artificielle actuellement. Beaucoup de chercheurs ont proposé de nouvelles logiques destinées à modéliser un tel raisonnement appelées logiques non monotones ; parmi celles-ci, on trouve, par exemple, la logique des défauts de Reiter [Rei80]. D'autres ont essayé de garder le cadre de la logique classique en rajoutant des systèmes numériques ou symboliques modélisant des relations d'ordre entre les croyances.

Nous nous intéressons plus particulièrement à cette dernière possibilité[1]. Ces relations d'ordre, appelées *relations de priorité*, peuvent se définir :

- soit de manière *sémantique* en utilisant le lien sémantique et la dépendance logique existant entre les croyances, on parle alors d'enracinement épistémique – voir [Gär91, GM94] ;
- soit de manière *syntaxique*, en s'appuyant sur le langage d'expression des croyances et en considérant chaque croyance de la base comme indivisible (elle est soit acceptée en entier, soit rejetée).

Nous nous situons plutôt dans la deuxième approche[2] qui induit une partition de la base de croyances dite alors base *stratifiée* (voir [Bre89b, Neb91, CRS92, BCD+93]) et nous parlons de raisonnement non monotone syntaxique. Nous définissons ainsi un paradigme des croyances et des préférences constitué d'une base de croyances E et d'une relation notée $<$ sur les éléments de E (la relation de priorité).

Parmi tous les aspects possibles d'un raisonnement (déductif, abductif, etc.), notre intérêt va se porter plus particulièrement sur la déduction. Cet aspect dans le cadre d'un raisonnement non monotone syntaxique est modélisé par une *relation d'inférence non monotone syntaxique*.

[1] Toutefois, à titre de comparaison, nous présenterons aussi un exemple de logique non monotone : la logique des défauts de Reiter [Rei80].

[2] Nous présentons quand même en section 2.5 les travaux de Gärdenfors et Makinson, afin de bien voir les différences entre ces deux approches.

Cette relation est vue, suivant en cela Pinkas et Loui ([PL92]), comme une procédure en deux étapes qui, premièrement, génère et sélectionne les états de croyances préférés (*mécanisme de génération*), puis qui utilise l'inférence classique sur tout ou partie de ces multiples états afin de conclure (*principe de résolution des conflits*), le but évident étant de continuer à pouvoir utiliser les mécanismes de la logique classique sur des parcelles de croyances cohérentes puisqu'on ne peut plus le faire sur la totalité des croyances, devenue incohérente ; par exemple, on définit l'inférence étudiée dans [BCD$^+$93] par : *"E infère Φ ssi Φ est classiquement inférée par tous les sous-ensembles préférés de E"*.

Une classification de ces principes de résolution des conflits, des plus crédules aux plus sceptiques, est donnée dans [PL92]. Quant à la sélection des sous-ensembles préférés, elle repose sur la définition de modes d'agrégation permettant d'étendre la relation de priorité donnée sur la base initiale en une préférence entre les sous-ensembles (voir [BCD$^+$93, CRS92]).

Dans le contexte décrit ci-dessus, notre contribution consiste en une étude comparative de quelques relations d'inférence non monotone syntaxique du point de vue de la complexité. Cette étude s'inscrit dans le cadre plus général d'une synthèse des différents formalismes de raisonnement non monotone syntaxique. En effet, la complexité est un aspect primordial pour toute application pratique et donc par là-même un angle de vue à prendre en compte lors du choix de tel ou tel formalisme. À notre connaissance, peu de travaux ont été publiés sur cet sujet. Parmi eux, on trouve les études de Nebel sur la complexité de certaines *procédures de révision* syntaxique [Neb91], ainsi que celles d'Eiter et Gottlob sur la logique des défauts, certains processus abductifs, etc. – voir [Got92, EG92].

Ce rapport est organisé comme suit :

1. à partir d'une base de croyances E et d'un pré-ordre sur E, énumération des formalismes étudiés avec la description succincte des mécanismes de production des sous-bases consistantes de E ; nous rappellerons quelques définitions :
   - celle de la décomposition de la base initiale E en un ensemble (noté P^E) de sous-bases consistantes,
   - celle de l'ordre ou des ordres à appliquer ou non sur ces sous-bases (noté $\ll$ pour le distinguer de l'ordre $<$ entre les formules de E) ;
2. énumération des principes de définition des relations d'inférence ;
3. étude de la relation d'inférence ($(E, <) \mid\!\sim^{p,m} \Phi$) issue de chaque couple ((ensemble de sous-bases P^E, ordre $\ll$)[3], principe p) du point de vue de la complexité[4].
4. étude des trois cas particuliers d'une base strictement ordonnée, d'une base de clauses (forme conjonctive normale) et d'une base de clauses de Horn.

Le but de cette synthèse est de permettre la constitution d'un répertoire de formalismes comparés tous du même point de vue : la manière d'inférer une nouvelle information dans un même paradigme.

[3] Ce couple $(P^E, \ll)$ est issu de $(E, <)$ par le mécanisme de génération m.

[4] Pourquoi avoir noté la relation d'inférence $(E, <) \mid\!\sim^{p,m} \Phi$ et pas $(P^E, \ll) \mid\!\sim^{p,m} \Phi$? Tout simplement pour bien mettre en évidence le cheminement complet entre la base de formules initiale, ordonnée ou pas, et la relation d'inférence non monotone permettant, à partir de cette base, d'inférer une formule.

Chapitre 2

La génération des sous-bases

Dans toute la suite, E dénotera la base de croyances (a priori ensemble de formules d'un langage propositionnel). Lorsqu'on disposera d'une relation de préférence sur les éléments de E, elle sera notée $<$. Les propriétés requises pour cette relation seront précisées lorsque ce sera nécessaire (en fonction des approches décrites). Les formalismes présentés ici correspondent essentiellement à l'approche syntaxique. Le seul formalisme issu d'une approche sémantique est celui de Gärdenfors et Makinson que nous citons afin de bien voir les différences entre les deux approches syntaxiques et sémantiques. Nous présenterons aussi en section 2.3 un exemple de logique non-monotone : la logique des défauts [Rei80].

Nous appellerons mécanisme initial, tout mécanisme permettant d'obtenir un ensemble de sous-bases consistantes ordonné ou pas. Nous ne nous étendrons pas sur la description des dits mécanismes, suivant en cela la méthode préconisée par [PL92]. Toutefois, pour les démonstrations de complexité, il faudra quand même disposer des définitions des sous-bases consistantes et de l'ordre entre ces sous-bases. Nous allons donc donner quelques rappels pour chacun des formalismes étudiés.

2.1 Les sous-bases consistantes maximales pour l'inclusion et les ordres associés

2.1.1 Définitions principales

Soit E un ensemble fini non vide quelconque de formules, on définit :

Définition 2.1.1 *Les sous-bases consistantes A de E sont les sous-ensembles de E consistants au sens de la logique classique.*

Définition 2.1.2 *Une sous-base A de E est une sous-base consistante maximale pour l'inclusion (sous-base maximale consistante) de E ssi :*

- *A est une sous-base consistante,*
- *il n'existe pas de sous-base consistante de E contenant strictement A.*

Les sous-bases de E maximales consistantes seront aussi appelées thèses de E.

Le langage utilisé pour énoncer les formules de E peut être soit le langage de la logique propositionnelle, soit le langage de la logique des prédicats avec des formules fermées. Toutefois, dans le cadre de notre étude de complexité, nous nous limiterons à des formules de la logique propositionnelle (voir chapitre 6).

Il existe de nombreux travaux qui utilisent les thèses de E. On les rassemble sous le terme "INCLUSION BASED PREFERENCE", c'est-à-dire que parmi toutes les sous-bases consistantes, on privilégie les sous-bases maximales consistantes. Nous allons voir dans les sections suivantes les diverses utilisations de ces thèses ainsi que la terminologie utilisée suivant les auteurs des différents travaux.

2.1.2 Les sous-théories préférées

Soit une base E de formules munie d'une relation de pré-ordre $<$ (réflexive et transitive), il existe plusieurs mécanismes pour partitionner E, par exemple la stratification uniforme. On appelle stratification uniforme de E le résultat de l'application d'un tri topologique sur E[1]. Dans ce cas, on dit que E est stratifiée et on note : $E = E_1 \cup \ldots \cup E_n$ avec n nombre de strates dans E.

Définition 2.1.3 *Soit E une base stratifiée, une sous-théorie préférée est un ensemble $S = S_1 \cup \ldots \cup S_n$ tel que $\forall k \in [1, n]$, $S_1 \cup \ldots \cup S_k$ est un sous-ensemble maximal consistant de $E_1 \cup \ldots \cup E_k$ (voir [Bre89b], [Bre89a]).*

Par définition, les sous-théories préférées de Brewka sont des thèses de E.

EXEMPLE 1 : Soit E la base propositionnelle suivante (avec en première colonne le numéro de strate E_i, en seconde colonne les formules de la base, en dernière colonne la signification des formules de la base) :

E_1	$\rightarrow P$	titi est un pingouin
E_2	$P \rightarrow O$	un pingouin est un oiseau
E_3	$P \rightarrow \neg V$	un pingouin ne sait pas voler
E_4	$O \rightarrow V$	un oiseau sait voler
E_5	$O \rightarrow A$	un oiseau a des ailes
E_6	$A \rightarrow V$	Si on a des ailes, on sait voler

Si on calcule les thèses, on trouve 5 solutions :

$\rightarrow P$	$\rightarrow P$	$\rightarrow P$	$\rightarrow P$	$P \rightarrow O$
$P \rightarrow O$	$P \rightarrow O$	$P \rightarrow O$	$P \rightarrow \neg V$	$P \rightarrow \neg V$
$P \rightarrow \neg V$	$P \rightarrow \neg V$	$O \rightarrow V$	$O \rightarrow V$	$O \rightarrow V$
$O \rightarrow A$	$A \rightarrow V$	$O \rightarrow A$	$O \rightarrow A$	$O \rightarrow A$
		$A \rightarrow V$	$A \rightarrow V$	$A \rightarrow V$

Ensuite, nous appliquons la méthode de Brewka et nous obtenons une seule sous-théorie préférée :

$\rightarrow P$
$P \rightarrow O$
$P \rightarrow \neg V$
$O \rightarrow A$

EXEMPLE 2 : Les résultats obtenus dépendent bien sûr de la manière dont les formules sont ordonnées dans les strates. Si on a la base E suivante :

[1] Il existe plusieurs possibilités de tris topologiques. Le plus utilisé en raisonnement non monotone est le suivant. Soit E une base de formules munie d'un pré-ordre $<$, on applique l'algorithme :

$i \leftarrow 1$
$E_1 \leftarrow \varnothing$
tant que E non vide faire
    *$E_i \leftarrow$ Max(E) (*Max(E) = les éléments maximaux de E pour $<$ *)*
    $E \leftarrow E$ - E_i
    $i \leftarrow i + 1$
fin tant que

E_1	$\rightarrow P$	titi est un pingouin
	$P \rightarrow O$	un pingouin est un oiseau
E_2	$P \rightarrow \neg V$	un pingouin ne sait pas voler
	$O \rightarrow V$	un oiseau sait voler
E_3	$O \rightarrow A$	un oiseau a des ailes
	$A \rightarrow V$	si on a des ailes, on sait voler

Dans ce cas là, les thèses sont toujours les mêmes, et on trouve 3 sous-théories préférées :

$\rightarrow P$	$\rightarrow P$	$\rightarrow P$
$P \rightarrow O$	$P \rightarrow O$	$P \rightarrow O$
$P \rightarrow \neg V$	$P \rightarrow \neg V$	$O \rightarrow V$
$O \rightarrow A$	$A \rightarrow V$	$O \rightarrow A$
		$A \rightarrow V$

EXEMPLE 3 : Soit E la base propositionnelle suivante ne contenant qu'une seule strate :

E_1	$\rightarrow P$	titi est un pingouin
	$P \rightarrow O$	un pingouin est un oiseau
	$P \rightarrow \neg V$	un pingouin ne sait pas voler
	$O \rightarrow V$	un oiseau sait voler
	$O \rightarrow A$	un oiseau a des ailes
	$A \rightarrow V$	Si on a des ailes, on sait voler

Si on calcule les thèses, on trouve toujours les mêmes 5 solutions. Puis, nous appliquons la méthode de Brewka et nous obtenons alors 5 sous-théories préférées. Toutes les thèses sont des sous-théories préférées.

CONCLUSIONS :

1. Brewka calcule directement les sous-théories préférées sans passer par le calcul des thèses.
2. Dans son processus, Brewka ne définit pas de relation de préférence entre ses sous-théories préférées. Donc, dans le cas des exemples 2 et 3 ci-dessus, les solutions obtenues ne sont pas comparables.
3. Brewka signale dans son article que dans certains cas, on retrouve le même type de résultats qu'avec les extensions de la logique des défauts de Reiter (voir [Rei80])[2].
4. Dans le cas d'une base ne contenant qu'une seule strate, l'ensemble des sous-théories préférées est égal à celui des thèses.

Remarque : Ces travaux ont inspiré d'autres travaux (voir [Cay90], [Cay92], [CRS92]).

2.1.3 Les thèses préférées

Dans les articles [Cay90] et [Cay92], Cayrol utilise le terme d'interprétation pour parler de sous-bases maximales consistantes. Désormais ce terme est remplacé par celui de thèse.

Puis dans [CRS92], Cayrol, Royer, Saurel ont défini, à partir d'une relation $<$ (dite de préférabilité) sur les formules de E, plusieurs relations de préférence $\ll$ pour classer les sous-bases consistantes de E.

[2] À condition que les défauts exprimés soient des défauts normaux sans pré-requis, que l'ensemble W de Reiter soit égal à la strate la plus prioritaire de E et que l'ordre soit judicieux entre les formules des strates inférieures qui sont traduites sous forme de défaut. Il serait intéressant de formaliser ce que l'on entend par "ordre judicieux", mais ce n'est pas notre propos. Constatons quand même que la méthode de Brewka sur la base de l'exemple 2 fournit exactement les mêmes ensembles que les extensions de Reiter (la fermeture logique mise à part). Et si on modifie la base en transformant la strate numéro 1 en 2 strates 1 et 1bis, on obtient les sous-théories préférées $\{\rightarrow P, P \rightarrow O, P \rightarrow \neg V, O \rightarrow A\}$ et $\{\rightarrow P, P \rightarrow O, P \rightarrow \neg V, A \rightarrow V\}$ qui ne correspondent pas aux extensions.

L'objectif premier était de généraliser les travaux de Brewka en définissant une préférence $\ll$ sur les sous-bases consistantes de E telle que les éléments maximaux pour cet ordre soient les sous-théories préférées définies dans [Bre89b].

Il a été démontré alors que la préférence $\ll$ sur les sous-bases consistantes généralise l'inclusion ensembliste. C'est donc une préférence du type "INCLUSION BASED PREFERENCE".

On a plusieurs résultats suivant que E est ordonné ou pas :

- si E n'est pas ordonné[3], alors $\ll$ correspond exactement à l'ordre basé sur l'inclusion ensembliste,
- si E est ordonné avec un pré-ordre $<$[4], alors on a 2 définitions possibles de l'ordre $\ll$ appelé ordre démocratique :
  - définition 2.1.4 valable dans le cas général, que $<$ soit total ou partiel,
  - définition 2.1.5 valable uniquement si $<$ est total et qui fournit un processus constructif.

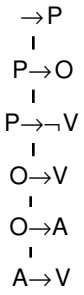
L'ORDRE DÉMOCRATIQUE : Les définitions de l'ordre démocratique sur les sous-bases consistantes de la base initiale ordonnée $(E, <)$ sont :

Définition 2.1.4 *Soient F et G, 2 sous-bases consistantes de $(E, <)$, F est préférée démocratiquement à G (noté $G \ll^d F$) ssi $\forall g \in G \setminus F$, $\exists f \in F \setminus G$ tel que $g < f$ et que l'on n'ait pas $f < g$.*

Définition 2.1.5 *Soit E la base initiale stratifiée, $E = E_1 \cup \ldots \cup E_n$ (E_1 strate de plus haute priorité, E_n strate de plus faible priorité). $\forall i$ et $\forall F$ un sous-ensemble de E, on note $F_i = F \cap E_i$. Soient F et G, 2 sous-bases consistantes de $(E, <)$, F est préférée démocratiquement à G (noté $G \ll^d F$) ssi $\exists i$ tel que F_i contient strictement G_i et $\forall E_j$ telle que E_j soit une strate de priorité supérieure à E_i, on a $F_j = G_j$.*

Dans le cas où $<$ est total sur E, les 2 définitions sont équivalentes et les éléments maximaux de cet ordre sont appelées thèses démo-préférées.

EXEMPLE 1 : Appliquons cette formule à l'exemple des oiseaux, donc à la base E suivante représentée sous forme d'arbre pour mettre en évidence la relation d'ordre entre les formules (avec $\rightarrow P$ formule la plus prioritaire et $A \rightarrow V$ formule la moins prioritaire) :



On trouve alors les 5 thèses suivantes :

Thèse 1	Thèse 2	Thèse 3	Thèse 4	Thèse 5
$\rightarrow P$	$\rightarrow P$	$\rightarrow P$	$\rightarrow P$	$P \rightarrow O$
$P \rightarrow O$	$P \rightarrow O$	$P \rightarrow O$	$P \rightarrow \neg V$	$P \rightarrow \neg V$
$P \rightarrow \neg V$	$P \rightarrow \neg V$	$O \rightarrow V$	$O \rightarrow V$	$O \rightarrow V$
$O \rightarrow A$	$A \rightarrow V$	$O \rightarrow A$	$O \rightarrow A$	$O \rightarrow A$
		$A \rightarrow V$	$A \rightarrow V$	$A \rightarrow V$

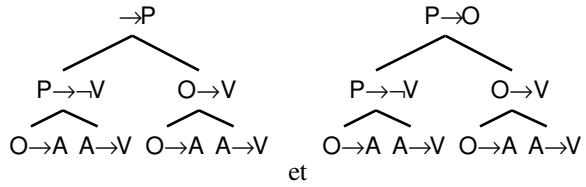
Puis en appliquant la définition de l'ordre démocratique, on trouve l'ordre suivant sur les thèses :

$$\text{thèse 5} \ll^d \text{thèse 4} \ll^d \text{thèse 3} \ll^d \text{thèse 2} \ll^d \text{thèse 1}$$

[3] E est alors composé d'une seule strate.

[4] Ce pré-ordre peut être total, on retrouve alors la stratification comme dans [Bre89b], ou bien seulement partiel.

EXEMPLE 2 : La base E est la suivante :



et

avec $\rightarrow P$ et $P \rightarrow O$, 2 formules de même priorité.

Ici la base E est partiellement ordonnée.

On retrouve les mêmes 5 thèses que dans l'exemple précédent.

Puis en appliquant la définition de l'ordre démocratique, on trouve l'ordre suivant sur les thèses :

thèse 5 $\ll^d$ thèse 4 $\ll^d$ thèse 1,
thèse 5 $\ll^d$ thèse 4 $\ll^d$ thèse 2,
thèse 5 $\ll^d$ thèse 4 $\ll^d$ thèse 3
(les thèses 1, 2 et 3 sont incomparables)

EXEMPLE 3 : Soit E la base propositionnelle suivante ne contenant qu'une seule strate :
$E = \{\rightarrow P, P \rightarrow O, P \rightarrow \neg V, O \rightarrow V, O \rightarrow A, A \rightarrow V\}$. Il n'y a pas d'ordre entre les formules.
Si on calcule les thèses, on trouve toujours les mêmes 5 solutions. Puis, lorsque nous appliquons l'ordre démocratique, nous obtenons 5 thèses préférées incomparables entre elles.

L'ORDRE ÉLITISTE : Dans [CRS92], on trouve aussi une autre relation de préférence : la préférence élitiste qui correspond à la recherche d'explications préférées par rapport à un but à atteindre et pour laquelle sont aussi données 2 définitions :

- définition 2.1.6 valable dans le cas général, que $<$ soit total ou partiel,
- définition 2.1.7 valable uniquement si $<$ est total et qui fournit un processus constructif.

Les définitions de l'ordre élitiste sur les sous-bases consistantes de la base initiale ordonnée $(E, <)$ sont :

Définition 2.1.6 *Soient F et G, 2 sous-bases consistantes de $(E, <)$, F est préférée élitiquement à G (noté $G \ll^e F$) ssi $\forall f \in F \setminus G$, $\exists g \in G \setminus F$ tel que $g < f$ et que l'on n'ait pas $f < g$.*

Définition 2.1.7 *Soit E la base initiale stratifiée, $E = E_1 \cup \ldots \cup E_n$ (E_1 strate de plus haute priorité, E_n strate de plus faible priorité). $\forall i$ et $\forall F$ un sous-ensemble de E, on note $F_i = F \cap E_i$. Soient F et G, 2 sous-bases consistantes de $(E, <)$, F est préférée élitiquement à G (noté $G \ll^e F$) ssi $\exists i$ tel que G_i contient strictement F_i et $\forall E_j$ telle que E_j soit une strate de priorité inférieure à E_i, on a $F_j = G_j$.*

Il n'est pas intéressant d'appliquer cet ordre à l'exemple précédent. En effet, les sous-bases préférées sont ici minimales pour l'inclusion. Cet ordre est surtout destiné à être utilisé dans des problèmes de type abductif (recherche d'explication, diagnostic).

CONCLUSIONS :

1. Si l'ordre $<$ est partiel sur E alors, l'ordre démocratique issu de $<$ ne permet pas d'ordonner toutes les thèses.
2. Les éléments maximaux obtenus avec l'ordre démocratique sont les thèses démo-préférées et, dans le cas d'un ordre total $<$ sur E, elles correspondent exactement aux sous-théories préférées.

3. L'ordre démocratique permet donc de généraliser les travaux de Brewka car :
   - il permet d'utiliser une base ordonnée partiellement (donc non stratifiée),
   - dans le cas d'une base stratifiée, il ne se contente pas de fournir les sous-théories préférées mais il permet d'ordonner toutes les thèses.
4. Dans le cas d'une base avec une strate unique, nous retrouvons les mêmes résultats qu'avec la méthode de Brewka : toutes les thèses sont préférées.

2.1.4 Les sous-bases fortement maximales consistantes

À toute base initiale $E = E_1 \cup \ldots \cup E_n$ stratifiée, on peut associer une base de croyances possibiliste B et vice-versa ; une base possibiliste est une base composée de couples (Φ, α) avec Φ une formule et α appelé le poids de cette formule et représentant la borne inférieure de la mesure de nécessité associée à la formule (voir [DLP91]).
En utilisant les sous-bases maximales consistantes de B, notées $B_1 \ldots B_p$, Dubois, Lang et Prade définissent la notion suivante :

Définition 2.1.8 *Soit K un ensemble de formules possibiliste, on définit $Inc(K)$ par :*

- *soit il existe un α_j tel que :*
  - $\cup_{i=1\ldots(j-1)}(\phi, \alpha_i)$ *est consistant,*
  - $\cup_{i=1\ldots j}(\phi, \alpha_i)$ *est inconsistant.*

  alors $Inc(K) = \alpha_j$,
- *soit $Inc(K) = 0$.*

On dit que $Inc(K)$ est le degré d'inconsistance de K.

$Inc(K)$ peut être calculé en utilisant le principe de résolution possibiliste (voir [DLP91]). D'autre part, on a une propriété importante : K est consistante ssi $Inc(K) = 0$.

Définition 2.1.9 *B_i est une sous-base fortement maximale consistante de B ssi B_i est consistante et $\forall(\Phi, \alpha) \in B \setminus B_i$, $Inc(B_i \cup \{(\Phi, \alpha)\}) = \alpha$.*

Il a été démontré que les sous-bases fortement maximales consistantes ainsi calculées sur B sont équivalentes aux thèses démo-préférées obtenues sur la base E stratifiée adéquate associée à B (voir dans [BDLP92]).

EXEMPLE 1 :

$$\begin{array}{l} B = \{(\rightarrow P, 1) \\ (P \rightarrow O, 0.9) \\ (P \rightarrow \neg V, 0.8) \\ (O \rightarrow V, 0.5) \\ (O \rightarrow A, 0.3) \\ (A \rightarrow V, 0.1)\} \end{array}$$

Comme précédemment, nous avons 5 sous-bases maximales consistantes :

$B_1 =$	$B_2 =$	$B_3 =$	$B_4 =$	$B_5 =$
$\{(\rightarrow P, 1)$	$\{(\rightarrow P, 1)$	$\{(\rightarrow P, 1)$	$\{(\rightarrow P, 1)$	$\{(P \rightarrow O, 0.9)$
$(P \rightarrow O, 0.9)$	$(P \rightarrow O, 0.9)$	$(P \rightarrow O, 0.9)$	$(P \rightarrow \neg V, 0.8)$	$(P \rightarrow \neg V, 0.8)$
$(P \rightarrow \neg V, 0.8)$	$(P \rightarrow \neg V, 0.8)$	$(O \rightarrow V, 0.5)$	$(O \rightarrow V, 0.5)$	$(O \rightarrow V, 0.5)$
$(O \rightarrow A, 0.3)\}$	$(A \rightarrow V, 0.1)\}$	$(O \rightarrow A, 0.3)$	$(O \rightarrow A, 0.3)$	$(O \rightarrow A, 0.3)$
		$(A \rightarrow V, 0.1)\}$	$(A \rightarrow V, 0.1)\}$	$(A \rightarrow V, 0.1)\}$

Une seule sous-base est fortement maximale consistante : B_1.

EXEMPLE 2 :

$$
\begin{array}{l}
B = \{(\rightarrow P, 1) \\
(P \rightarrow O, 1) \\
(P \rightarrow \neg V, 0.8) \\
(O \rightarrow V, 0.8) \\
(O \rightarrow A, 0.3) \\
(A \rightarrow V, 0.3)\}
\end{array}
$$

Comme précédemment, nous avons 5 sous-bases maximales consistantes :

$B_1 =$	$B_2 =$	$B_3 =$	$B_4 =$	$B_5 =$
$\{(\rightarrow P, 1)$	$\{(\rightarrow P, 1)$	$\{(\rightarrow P, 1)$	$\{(\rightarrow P, 1)$	$\{(P \rightarrow O, 1)$
$(P \rightarrow O, 1)$	$(P \rightarrow O, 1)$	$(P \rightarrow O, 1)$	$(P \rightarrow \neg V, 0.8)$	$(P \rightarrow \neg V, 0.8)$
$(P \rightarrow \neg V, 0.8)$	$(P \rightarrow \neg V, 0.8)$	$(O \rightarrow V, 0.8)$	$(O \rightarrow V, 0.8)$	$(O \rightarrow V, 0.8)$
$(O \rightarrow A, 0.3)\}$	$(A \rightarrow V, 0.3)\}$	$(O \rightarrow A, 0.3)$	$(O \rightarrow A, 0.3)$	$(O \rightarrow A, 0.3)$
		$(A \rightarrow V, 0.3)\}$	$(A \rightarrow V, 0.3)\}$	$(A \rightarrow V, 0.3)\}$

Trois sous-bases sont fortement maximales consistantes : B_1, B_2 et B_3.

EXEMPLE 3 : Soit B la base possibiliste propositionnelle suivante ne contenant qu'une seule strate :

$$
\begin{array}{l}
B = \{(\rightarrow P, 0, 7) \\
(P \rightarrow O, 0.7) \\
(P \rightarrow \neg V, 0.7) \\
(O \rightarrow V, 0.7) \\
(O \rightarrow A, 0.7) \\
(A \rightarrow V, 0.7)\}
\end{array}
$$

$B_1, \ldots, B_5$ sont toujours les sous-bases maximales consistantes et, ici, elles sont toutes fortement maximales consistantes.

CONCLUSIONS :

1. Les sous-bases fortement maximales consistantes de Dubois, Lang, Prade correspondent aux thèses démo-préférées.
2. Dubois, Lang, Prade ont aussi défini un ordre entre les sous-bases fortement maximales consistantes appelé ordre lexicographique décrit dans la section 2.2.
3. Nous pouvons donc dire, ici aussi, que l'ordre démocratique permet de généraliser le concept de sous-base fortement maximale consistante de [DLP91] par le fait qu'il traite les ensembles partiellement ordonnés.
4. Nous retrouvons toujours le même type de résultat dans le cas d'une base avec une seule strate.

2.1.5 Conclusion sur les méthodes "INCLUSION BASED PREFERENCE"

Toutes les méthodes vues dans cette section sont regroupées sous le terme "INCLUSION BASED PREFERENCE" car elles privilégient parmi les sous-bases consistantes celles qui sont maximales pour l'inclusion, nommées ici des thèses et permettent :

- soit de sélectionner parmi ces thèses la ou les thèses préférées,
- soit d'ordonner partiellement ou totalement les thèses.

Elles fournissent les mêmes résultats, même si les motivations sont au départ différentes.

2.2 Les sous-bases consistantes et l'ordre lexicographique

Ici, il ne s'agit pas de travailler à partir de sous-bases consistantes maximales pour l'inclusion, mais d'utiliser des sous-bases consistantes maximales pour la cardinalité. C'est la raison pour laquelle cette méthode, bien que citée dans le cadre des sous-bases fortement maximales consistantes de Dubois, Lang et Prade (voir [DLP91]), n'a pas été présentée avec les méthodes "INCLUSION BASED PREFERENCE".

La définition de base (ordre basé sur la cardinalité) correspondant au cas d'une base de croyances non ordonnée est (voir [DLP91]) :

Définition 2.2.1 *Soit E une base initiale finie, soit A un sous-ensemble de E, on dit que A est une sous-base consistante maximale pour la cardinalité ssi A est consistante et $\forall B$ sous-ensemble de E tel que B soit consistant alors $|B| \leq |A|$*[5].

Lorsqu'on se trouve en présence d'une base de croyances totalement ordonnée on définit :

Définition 2.2.2 *Soit E une base stratifiée finie ($E = E_1 \cup \ldots \cup E_n$), soient A et B deux sous-bases consistantes de E, on définit l'ordre lexicographique de la manière suivante : $A \ll^{lex} B$ ssi $\exists i$ tel que $|A_i| < |B_i|$ et pour tout $j < i$ on a $|A_j| = |B_j|$.*

Définition 2.2.3 *Soit E une base stratifiée finie ($E = E_1 \cup \ldots \cup E_n$), soit S un sous-ensemble de E, S est une sous-base préférée pour l'ordre lexicographique ssi $\forall k = 1 \ldots n$, $(S_1 \cup \ldots \cup S_k)$ est un sous-ensemble consistant maximal pour la cardinalité de $(E_1 \cup \ldots \cup E_k)$.*

Propriété 2.2.1 *Il est facile de montrer que l'ordre lexicographique raffine l'"INCLUSION BASED PREFERENCE" : toute sous-base de E se trouvant être élément maximal pour l'ordre $\ll^{lex}$ sera nécessairement élément maximal pour l'ordre "INCLUSION BASED".*

Voyons ce que cela nous donne sur un exemple.

EXEMPLE 1 : Soit E la base propositionnelle suivante (avec en première colonne le numéro de strate E_i, en seconde colonne les formules de la base, en dernière colonne la signification des formules de la base) :

E_1	$\rightarrow P$	titi est un pingouin
E_2	$P \rightarrow O$	un pingouin est un oiseau
E_3	$P \rightarrow \neg V$	un pingouin ne sait pas voler
E_4	$O \rightarrow V$	un oiseau sait voler
E_5	$O \rightarrow A$	un oiseau a des ailes
E_6	$A \rightarrow V$	Si on a des ailes, on sait voler

Nous avons ici une base ordonnée. Il n'est pas raisonnable de calculer toutes les sous-bases consistantes. Grâce à la propriété 2.2.1, nous pouvons donc nous contenter de partir des sous-bases maximales consistantes, donc des thèses :

Thèse 1	Thèse 2	Thèse 3	Thèse 4	Thèse 5
$\rightarrow P$	$\rightarrow P$	$\rightarrow P$	$\rightarrow P$	$P \rightarrow O$
$P \rightarrow O$	$P \rightarrow O$	$P \rightarrow O$	$P \rightarrow \neg V$	$P \rightarrow \neg V$
$P \rightarrow \neg V$	$P \rightarrow \neg V$	$O \rightarrow V$	$O \rightarrow V$	$O \rightarrow V$
$O \rightarrow A$	$A \rightarrow V$	$O \rightarrow A$	$O \rightarrow A$	$O \rightarrow A$
		$A \rightarrow V$	$A \rightarrow V$	$A \rightarrow V$

Ensuite, nous appliquons l'ordre lexicographique et nous obtenons :

$$\text{thèse 5} \ll^{lex} \text{thèse 4} \ll^{lex} \text{thèse 3} \ll^{lex} \text{thèse 2} \ll^{lex} \text{thèse 1}$$

Remarque : nous trouvons ici le même ordre que celui obtenu par la préférence démocratique.

[5] $|A|$ dénote le cardinal de A.

EXEMPLE 2 : Les résultats obtenus dépendent bien sûr de la manière dont les formules sont rangées dans les strates. Si on a la base E suivante :

E_1	$\rightarrow P$	titi est un pingouin
	$P \rightarrow O$	un pingouin est un oiseau
E_2	$P \rightarrow \neg V$	un pingouin ne sait pas voler
	$O \rightarrow V$	un oiseau sait voler
E_3	$O \rightarrow A$	un oiseau a des ailes
	$A \rightarrow V$	si on a des ailes, on sait voler

Dans ce cas là, les thèses sont toujours les mêmes, et on trouve les résultats suivants[6] :

$$\text{thèse 5} \equiv^{lex} \text{thèse 4} \ll^{lex} \text{thèse 2} \equiv^{lex} \text{thèse 1} \ll^{lex} \text{thèse 3}$$

(contrairement aux résultats obtenus avec l'ordre démocratique sur les thèses, on a ici comme élément maximal pour l'ordre lexicographique la thèse 3 tandis que les thèses 1 et 2 sont équivalentes ainsi que les thèses 4 et 5)

EXEMPLE 3 : Soit E la base propositionnelle suivante ne contenant qu'une seule strate :

E_1	$\rightarrow P$	titi est un pingouin
	$P \rightarrow O$	un pingouin est un oiseau
	$P \rightarrow \neg V$	un pingouin ne sait pas voler
	$O \rightarrow V$	un oiseau sait voler
	$O \rightarrow A$	un oiseau a des ailes
	$A \rightarrow V$	Si on a des ailes, on sait voler

Si on calcule les thèses, on trouve toujours les mêmes 5 solutions. Et si ensuite, nous appliquons l'ordre lexicographique, nous obtenons que :

$$\text{thèse 2} \equiv^{lex} \text{thèse 1} \ll^{lex} \text{thèse 5} \equiv^{lex} \text{thèse 4} \equiv^{lex} \text{thèse 3}$$

CONCLUSIONS :

1. Bien que l'ordre lexicographique soit défini sur des sous-bases consistantes, il est évident que les éléments maximaux de cet ordre sont pris parmi les thèses : $\forall T$ une thèse, $\forall S$ une sous-base consistante qui n'est pas une thèse, on aura toujours $S \ll^{lex} T$; d'où l'idée d'appliquer cet ordre uniquement sur les thèses afin de limiter les calculs[7].
2. Dans le cas d'une base stratifiée avec des strates ne contenant qu'une seule formule, l'ordre lexicographique appliqué aux thèses fournit des résultats identiques à ceux obtenus par toutes les méthodes précédentes.
3. Dans le cas d'une base stratifiée avec des strates pouvant contenir plusieurs formules et en particulier dans le cas d'une base avec une strate unique, l'ordre lexicographique appliqué aux thèses fournit des résultats différents de ceux obtenus par les méthodes précédentes. C'est son principal avantage, puisqu'il y a *au plus* autant de thèses préférées lexicographiquement que de thèses préférées au sens "INCLUSION BASED".
4. L'ordre lexicographique permet d'ordonner de manière totale les thèses, contrairement à l'ordre démocratique qui ne fournit qu'un ordre partiel sur les thèses.
5. Lehmann dans [Leh92] a défini un ordre semblable.

[6] Le symbole $\equiv^{lex}$ représente la relation d'équivalence issue de l'ordre $\ll^{lex}$: soient A et B, 2 sous-bases, $A \equiv^{lex} B \Leftrightarrow (A \ll^{lex} B$ et $B \ll^{lex} A)$.

[7] Il est bien sûr évident que cette idée est applicable dans le cadre du petit exemple présenté. Toutefois, dans le cas général, il vaut mieux chercher une méthode constructive ne nécessitant ni le calcul des sous-bases consistantes, ni celui des thèses.

2.3 Les extensions de la logique des défauts

Ici nous sortons du cadre d'utilisation des thèses (sous-bases maximales consistantes). Les travaux de Reiter (voir [Rei80]) ont consisté à définir une nouvelle logique avec de nouvelles règles d'inférence permettant de prendre en compte des règles générales pouvant comporter des défauts et ainsi permettant de gérer une inconsistance de la base de croyances.

Nous allons toutefois citer quelques-uns des résultats obtenus par Reiter puisque ces résultats, dans certains cas très particuliers, sont équivalents à ceux obtenus par l'utilisation de thèses (voir [Bre89b]).

La base de croyances E est ici partitionnée en :

- $W =$ ensemble de formules fermées propositionnelles ou du premier ordre (prémisses),
- $D =$ ensemble de défauts c'est-à-dire de formules de la forme $a : b/c$ avec a[8], b[9], c[10] des formules propositionnelles ou fermées du premier ordre,

Définition 2.3.1 *L'interprétation d'un défaut $a : b/c$ est : "si a est connu et b consistant avec tout ce qui est connu alors inférer c". Le défaut $a : b/c$ est donc une nouvelle règle d'inférence*[11].

Définition 2.3.2 *Les extensions sont des ensembles de croyances contenant W et fermés pour les 2 types d'inférence : l'inférence classique et l'inférence des défauts.*

Dans [Rei80] on trouve un processus de construction des extensions.

EXEMPLE : Contrairement aux sections précédentes, nous n'aurons pas 3 exemples à traiter, puisque la base initiale E n'est pas ordonnée. Nous allons choisir de traduire notre exemple des oiseaux avec des défauts normaux sans pré-requis afin de voir les points communs avec les méthodes présentées précédemment (voir [Rei80]) :

$E = (W, D)$ avec :

- $W = \{ \rightarrow P, P \rightarrow O \}$,
- $D = \{ \ :O \rightarrow V/O \rightarrow V, \ :P \rightarrow \neg V/P \rightarrow \neg V, \ :O \rightarrow A/O \rightarrow A, \ :A \rightarrow V/A \rightarrow V \}$

Nous construisons alors l'arbre de calcul des extensions et trouvons 3 extensions possibles E_1, E_2 et E_3 (voir la figure 2.1).

Remarque : ces trois extensions correspondent bien aux thèses préférées obtenues par les méthodes "INCLUSION BASED PREFERENCE" lorsque les formules $(P \rightarrow O)$ et $(\rightarrow P)$ constituent la strate de plus forte priorité dans la base initiale E (voir note sur ce sujet dans la conclusion de la section 2.1.2).

CONCLUSIONS :

1. Chez Reiter et dans le cas d'une base (W, D) ne contenant que des défauts normaux sans pré-requis $(D = \{ : b_i/b_i \})$ les extensions calculées correspondent aux thèses démo-préférées obtenues sur une base initiale comportant 2 strates, la plus prioritaire pour les formules de W et la suivante pour les formules b_i correspondant aux défauts, quelle que soit la méthode utilisée pour les calculer.

2. Reiter n'a pas prévu d'ordre sur ses extensions. Poole, en introduisant les défauts nommés, s'est donné la possibilité d'ordonner la base initiale mais n'a rien défini au niveau extension (voir [Poo88]).

[8] a est appelé le pré-requis.
[9] b est appelé la justification.
[10] c est appelé le conséquent.
[11] Les définitions données ici sont volontairement simplifiées. Pour plus de précision, se reporter à [Rei80].

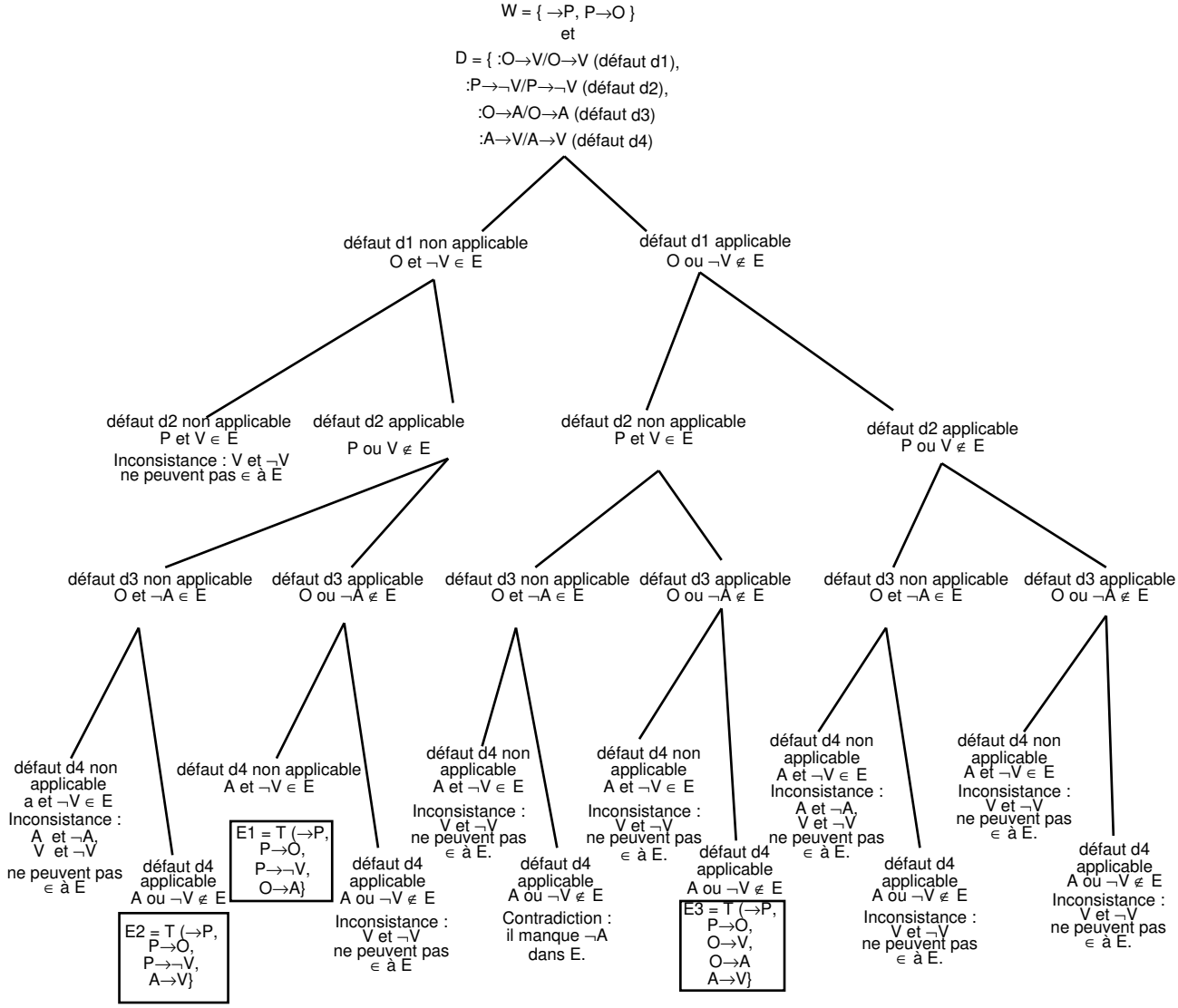


Figure 2.1: Arbre des extensions

2.4 Les sous-bases consistantes et l'ordre "BEST-OUT"

L'ordre "BEST-OUT" décrit dans [BCD+93] et [DLP91] est défini, comme pour l'ordre lexicographique, à partir de sous-bases consistantes et non à partir des thèses.

Soit une base E initiale stratifiée ($E = E_1 \cup \ldots \cup E_n$), on définit :

Définition 2.4.1 *Soit A un sous-ensemble de E ($A = A_1 \cup \ldots \cup A_n$), on note $a(A)$ la plus haute priorité d'une formule de E qui n'est pas dans A ; $a(A) = min\{i, \exists \Phi \in E_i \setminus A\}$.*

Définition 2.4.2 *Soient A et B, 2 sous-bases consistantes de E, on définit l'ordre "BEST-OUT" de la manière suivante : $A \ll^{bo} B$ ssi $a(A) \leq a(B)$ (on dit alors que B est préférée à A pour l'ordre $\ll^{bo}$).*

Attention, contrairement à ce qui se produit pour l'ordre lexicographique, les éléments maximaux de l'ordre "BEST-OUT" ne sont pas nécessairement des éléments maximaux pour l'inclusion ensembliste. On ne peut donc pas se restreindre au calcul des thèses. Il faut calculer toutes les sous-bases consistantes.

EXEMPLE 1 : Soit E la base propositionnelle suivante (avec en première colonne le numéro de strate E_i, en seconde colonne les formules de la base, en dernière colonne la signification des formules de la base) :

E_1	$\rightarrow P$	titi est un pingouin
E_2	$P \rightarrow O$	un pingouin est un oiseau
E_3	$P \rightarrow \neg V$	un pingouin ne sait pas voler
E_4	$O \rightarrow V$	un oiseau sait voler
E_5	$O \rightarrow A$	un oiseau a des ailes
E_6	$A \rightarrow V$	Si on a des ailes, on sait voler

Calculons les sous-bases consistantes. Il y a $2^6 - 2$ possibilités à explorer (on ne tient pas compte de la base vide, ni de la base initiale), ce qui nous fait au plus 62 sous-bases consistantes qu'il faudrait comparer 2 à 2 pour pouvoir les ordonner. En fait, si on cherche à calculer les sous-ensembles consistants A de E en fonction des différents $a(A)$, on trouve 58 solutions classées suivant la valeur $a(A)$ (voir table 2.1).

Remarque : les sous-bases solutions correspondant aux thèses sont les solutions 1, 2, 4, 12, 28.

Avec l'ordre "BEST-OUT", on trouve donc :

$$\text{solutions 28 à 58} \ll^{bo} \text{solutions 12 à 27} \ll^{bo} \text{solutions 4 à 11} \ll^{bo} \text{solutions 1 à 3}$$

(avec les solutions 1 à 3 équivalentes, de même que les solutions 4 à 11, 12 à 27, 28 à 58)

Remarque : si on compare avec les résultats obtenus sur les thèses par d'autres méthodes, on obtient des résultats différents, puisqu'ici les 2 thèses (solution 1 et solution 2) sont équivalentes, alors qu'avec les autres méthodes, la solution 1 était préférée à la solution 2.

EXEMPLE 2 : Les résultats obtenus dépendent bien sûr de la manière dont les formules sont rangées dans les strates. Si on a par exemple la base E suivante :

E_1	$\rightarrow P$	titi est un pingouin
	$P \rightarrow O$	un pingouin est un oiseau
E_2	$P \rightarrow \neg V$	un pingouin ne sait pas voler
	$O \rightarrow V$	un oiseau sait voler
E_3	$O \rightarrow A$	un oiseau a des ailes
	$A \rightarrow V$	si on a des ailes, on sait voler

les A consistants tels que $a(A) = 6$: pas de A car l'ensemble $\{\rightarrow P, P \rightarrow O, P \rightarrow \neg V, O \rightarrow V, O \rightarrow A\}$ est inconsistant	les A consistants tels que $a(A) = 2$: 16 possibilités toutes consistantes donc 16 solutions :	les A consistants tels que $a(A) = 1$: 32 possibilités toutes consistantes sauf la base vide donc 31 solutions :
les A consistants tels que $a(A) = 5$: pas de A car l'ensemble $\{\rightarrow P, P \rightarrow O, P \rightarrow \neg V, O \rightarrow V\}$ est inconsistant	$\{\rightarrow P, P \rightarrow \neg V, O \rightarrow V, O \rightarrow A, A \rightarrow V\}$: sol. 12 $\{\rightarrow P, P \rightarrow \neg V, O \rightarrow V, O \rightarrow A\}$: sol. 13 $\{\rightarrow P, P \rightarrow \neg V, O \rightarrow V, A \rightarrow V\}$: sol. 14 $\{\rightarrow P, P \rightarrow \neg V, O \rightarrow V\}$: sol. 15 $\{\rightarrow P, P \rightarrow \neg V, O \rightarrow A, A \rightarrow V\}$: sol. 16	$\{P \rightarrow O, P \rightarrow \neg V, O \rightarrow V$, O$\rightarrow$ A, $A \rightarrow V\}$: sol. 28 $\{P \rightarrow O, P \rightarrow \neg V, O \rightarrow V$, O$\rightarrow$ A$\}$: sol. 29 $\{P \rightarrow O, P \rightarrow \neg V, O \rightarrow V$, A$\rightarrow$ V$\}$: sol. 30 $\{P \rightarrow O, P \rightarrow \neg V, O \rightarrow V\}$: sol. 31 $\{P \rightarrow O, P \rightarrow \neg V, O \rightarrow A$, A$\rightarrow$ V$\}$: sol. 32
les A consistants tels que $a(A) = 4$: 4 possibilités dont une inconsistante donc 3 solutions : $\{\rightarrow P, P \rightarrow O, P \rightarrow \neg V, O \rightarrow A, A \rightarrow V\}$: inconsistante $\{\rightarrow P, P \rightarrow O, P \rightarrow \neg V, O \rightarrow A\}$: sol. 1 $\{\rightarrow P, P \rightarrow O, P \rightarrow \neg V, A \rightarrow V\}$: sol. 2 $\{\rightarrow P, P \rightarrow O, P \rightarrow \neg V\}$: sol. 3	$\{\rightarrow P, P \rightarrow \neg V, O \rightarrow A\}$: sol. 17 $\{\rightarrow P, P \rightarrow \neg V, A \rightarrow V\}$: sol. 18 $\{\rightarrow P, P \rightarrow \neg V\}$: sol. 19 $\{\rightarrow P, O \rightarrow V, O \rightarrow A, A \rightarrow V\}$: sol. 20 $\{\rightarrow P, O \rightarrow V, O \rightarrow A\}$: sol. 21 $\{\rightarrow P, O \rightarrow V, A \rightarrow V\}$: sol. 22 $\{\rightarrow P, O \rightarrow V\}$: sol. 23 $\{\rightarrow P, O \rightarrow A, A \rightarrow V\}$: sol. 24 $\{\rightarrow P, O \rightarrow A\}$: sol. 25 $\{\rightarrow P, A \rightarrow V\}$: sol. 26 $\{\rightarrow P\}$: sol. 27	$\{P \rightarrow O, P \rightarrow \neg V, O \rightarrow V, O \rightarrow A, A \rightarrow V\}$: sol. 33 $\{P \rightarrow O, P \rightarrow \neg V, A \rightarrow V\}$: sol. 34 $\{P \rightarrow O, P \rightarrow \neg V\}$: sol. 35 $\{P \rightarrow O, O \rightarrow V, O \rightarrow A, A \rightarrow V\}$: sol. 36 $\{P \rightarrow O, O \rightarrow V, O \rightarrow A\}$: sol. 37 $\{P \rightarrow O, O \rightarrow V, A \rightarrow V\}$: sol. 38 $\{P \rightarrow O, O \rightarrow V\}$: sol. 39 $\{P \rightarrow O, O \rightarrow A, A \rightarrow V\}$: sol. 40 $\{P \rightarrow O, O \rightarrow A\}$: sol. 41 $\{P \rightarrow O, A \rightarrow V\}$: sol. 42 $\{P \rightarrow O\}$: sol. 43
les A consistants tels que $a(A) = 3$: 8 possibilités toutes consistantes donc 8 solutions : $\{\rightarrow P, P \rightarrow O, O \rightarrow V, O \rightarrow A, A \rightarrow V\}$: sol. 4 $\{\rightarrow P, P \rightarrow O, O \rightarrow V, O \rightarrow A\}$: sol. 5 $\{\rightarrow P, P \rightarrow O, O \rightarrow V, A \rightarrow V\}$: sol. 6 $\{\rightarrow P, P \rightarrow O, O \rightarrow V\}$: sol. 7 $\{\rightarrow P, P \rightarrow O, O \rightarrow A, A \rightarrow V\}$: sol. 8 $\{\rightarrow P, P \rightarrow O, O \rightarrow A\}$: sol. 9 $\{\rightarrow P, P \rightarrow O, A \rightarrow V\}$: sol. 10 $\{\rightarrow P, P \rightarrow O\}$: sol. 11		$\{P \rightarrow \neg V, O \rightarrow V, O \rightarrow A$, A$\rightarrow$ V$\}$: sol. 44 $\{P \rightarrow \neg V, O \rightarrow V, O \rightarrow A\}$: sol. 45 $\{P \rightarrow \neg V, O \rightarrow V\}$: sol. 46 $\{P \rightarrow \neg V, O \rightarrow V, A \rightarrow V\}$: sol. 47 $\{P \rightarrow \neg V, O \rightarrow A, A \rightarrow V\}$: sol. 48 $\{P \rightarrow \neg V, O \rightarrow A\}$: sol. 49 $\{P \rightarrow \neg V, A \rightarrow V\}$: sol. 50 $\{P \rightarrow \neg V\}$: sol. 51 $\{O \rightarrow V, O \rightarrow A, A \rightarrow V\}$: sol. 52 $\{O \rightarrow V, O \rightarrow A, \}$: sol. 53 $\{O \rightarrow V, A \rightarrow V\}$: sol. 54 $\{O \rightarrow V\}$: sol. 55 $\{O \rightarrow A, A \rightarrow V\}$: sol. 56 $\{O \rightarrow A\}$: sol. 57 $\{A \rightarrow V\}$: sol. 58

Tableau 2.1: L'ordre "BEST-OUT" (début)

les A consistants tels que $a(A) = 3$: pas de A car l'ensemble $\{\rightarrow P, P \rightarrow O, P \rightarrow \neg V, O \rightarrow V\}$ est inconsistant	les A consistants tels que $a(A) = 2$: 12 possibilités dont une inconsistante donc 11 solutions : $\{\rightarrow P, P \rightarrow O, P \rightarrow \neg V, O \rightarrow A, A \rightarrow V\}$: inconsistante $\{\rightarrow P, P \rightarrow O, P \rightarrow \neg V, O \rightarrow A\}$: sol. 1 $\{\rightarrow P, P \rightarrow O, P \rightarrow \neg V, A \rightarrow V\}$: sol. 2 $\{\rightarrow P, P \rightarrow O, P \rightarrow \neg V\}$: sol. 3 $\{\rightarrow P, P \rightarrow O, O \rightarrow V, O \rightarrow A, A \rightarrow V\}$: sol. 4 $\{\rightarrow P, P \rightarrow O, O \rightarrow V, O \rightarrow A\}$: sol. 5 $\{\rightarrow P, P \rightarrow O, O \rightarrow V, A \rightarrow V\}$: sol. 6 $\{\rightarrow P, P \rightarrow O, O \rightarrow V\}$: sol. 7 $\{\rightarrow P, P \rightarrow O, O \rightarrow A, A \rightarrow V\}$: sol. 8 $\{\rightarrow P, P \rightarrow O, O \rightarrow A\}$: sol. 9 $\{\rightarrow P, P \rightarrow O, A \rightarrow V\}$: sol. 10 $\{\rightarrow P, P \rightarrow O\}$: sol. 11	les A consistants tels que $a(A) = 1$: 48 possibilités toutes consistantes sauf la base vide donc 47 solutions : $\{\rightarrow P, P \rightarrow \neg V, O \rightarrow V, O \rightarrow A, A \rightarrow V\}$: sol. 12 $\{\rightarrow P, P \rightarrow \neg V, O \rightarrow V,$ O$\rightarrow$ A$\}$: sol. 13 $\{\rightarrow P, P \rightarrow \neg V, O \rightarrow V,$ A$\rightarrow$ V$\}$: sol. 14 $\{\rightarrow P, P \rightarrow \neg V, O \rightarrow V\}$: sol. 15 $\{\rightarrow P, P \rightarrow \neg V, O \rightarrow A,$ A$\rightarrow$ V$\}$: sol. 16 $\{\rightarrow P, P \rightarrow \neg V, O \rightarrow A\}$: sol. 17 $\{\rightarrow P, P \rightarrow \neg V, A \rightarrow V\}$: sol. 18 $\{\rightarrow P, P \rightarrow \neg V\}$: sol. 19 $\{\rightarrow P, O \rightarrow V, O \rightarrow A, A \rightarrow V\}$: sol. 20 $\{\rightarrow P, O \rightarrow V, O \rightarrow A\}$: sol. 21 $\{\rightarrow P, O \rightarrow V, A \rightarrow V\}$: sol. 22 $\{\rightarrow P, O \rightarrow V\}$: sol. 23 $\{\rightarrow P, O \rightarrow A, A \rightarrow V\}$: sol. 24 $\{\rightarrow P, O \rightarrow A\}$: sol. 25 $\{\rightarrow P, A \rightarrow V\}$: sol. 26 $\{\rightarrow P\}$: sol. 27 $\{P \rightarrow O, P \rightarrow \neg V, O \rightarrow V,$ O$\rightarrow$ A$, A \rightarrow V\}$: sol. 28 $\{P \rightarrow O, P \rightarrow \neg V, O \rightarrow V,$ O$\rightarrow$ A$\}$: sol. 29 $\{P \rightarrow O, P \rightarrow \neg V, O \rightarrow V,$ A$\rightarrow$ V$\}$: sol. 30 $\{P \rightarrow O, P \rightarrow \neg V, O \rightarrow V\}$: sol. 31 $\{P \rightarrow O, P \rightarrow \neg V, O \rightarrow A,$ A$\rightarrow$ V$\}$: sol. 32 $\{P \rightarrow O, P \rightarrow \neg V, O \rightarrow A\}$: sol. 33 $\{P \rightarrow O, P \rightarrow \neg V, A \rightarrow V\}$: sol. 34 $\{P \rightarrow O, P \rightarrow \neg V\}$: sol. 35 $\{P \rightarrow O, O \rightarrow V, O \rightarrow A, A \rightarrow V\}$: sol. 36 $\{P \rightarrow O, O \rightarrow V, O \rightarrow A\}$: sol. 37 $\{P \rightarrow O, O \rightarrow V, A \rightarrow V\}$: sol. 38 $\{P \rightarrow O, O \rightarrow V\}$: sol. 39 $\{P \rightarrow O, O \rightarrow A, A \rightarrow V\}$: sol. 40 $\{P \rightarrow O, O \rightarrow A\}$: sol. 41 $\{P \rightarrow O, A \rightarrow V\}$: sol. 42 $\{P \rightarrow O\}$: sol. 43 $\{P \rightarrow \neg V, O \rightarrow V, O \rightarrow A,$ A$\rightarrow$ V$\}$: sol. 44 $\{P \rightarrow \neg V, O \rightarrow V, O \rightarrow A\}$: sol. 45 $\{P \rightarrow \neg V, O \rightarrow V\}$: sol. 46 $\{P \rightarrow \neg V, O \rightarrow V, A \rightarrow V\}$: sol. 47 $\{P \rightarrow \neg V, O \rightarrow A, A \rightarrow V\}$: sol. 48 $\{P \rightarrow \neg V, O \rightarrow A\}$: sol. 49 $\{P \rightarrow \neg V, A \rightarrow V\}$: sol. 50 $\{P \rightarrow \neg V\}$: sol. 51 $\{O \rightarrow V, O \rightarrow A, A \rightarrow V\}$: sol. 52 $\{O \rightarrow V, O \rightarrow A, \}$: sol. 53 $\{O \rightarrow V, A \rightarrow V\}$: sol. 54 $\{O \rightarrow V\}$: sol. 55 $\{O \rightarrow A, A \rightarrow V\}$: sol. 56 $\{O \rightarrow A\}$: sol. 57 $\{A \rightarrow V\}$: sol. 58

Tableau 2.2: L'ordre "BEST-OUT" (suite)

On trouve bien sûr le même nombre de sous-bases consistantes mais elles ne sont pas classées de la même façon (voir table 2.2).

Ici aussi, on se rend compte que les résultats obtenus sont différents de ceux obtenus par les autres méthodes :

$$\text{solutions 12 à 58} \ll^{bo} \text{solutions 1 à 11}$$

(avec les solutions 1 à 11 équivalentes, ainsi que les solutions 12 à 58)

Remarque : Si on regarde l'effet de l'ordre "BEST-OUT" sur les thèses, on retrouve que les 3 premières thèses sont de même importance ; par contre, c'est aussi le cas pour les 2 dernières, ce qui n'apparaissait pas dans les méthodes "INCLUSION BASED" pour lesquelles soit il n'y avait pas de comparaison possible, soit la thèse $\{\rightarrow P, P \rightarrow \neg V, O \rightarrow V, O \rightarrow A, A \rightarrow V\}$ était préférée à la thèse $\{P \rightarrow O, P \rightarrow \neg V, O \rightarrow V, O \rightarrow A, A \rightarrow V\}$.

EXEMPLE 3 : Soit E la base propositionnelle suivante avec une seule strate :

E_1	$\rightarrow P$	titi est un pingouin
	$P \rightarrow O$	un pingouin est un oiseau
	$P \rightarrow \neg V$	un pingouin ne sait pas voler
	$O \rightarrow V$	un oiseau sait voler
	$O \rightarrow A$	un oiseau a des ailes
	$A \rightarrow V$	Si on a des ailes, on sait voler

On retrouve bien sûr toujours les mêmes 58 sous-bases consistantes, mais cette fois elles sont toutes préférées (voir table 2.3).

Avec l'ordre "BEST-OUT", on trouve donc que les solutions 1 à 58 sont équivalentes.

CONCLUSION :

1. L'ordre "BEST-OUT" est défini sur les sous-bases consistantes et pas sur les sous-bases maximales consistantes.

2. Contrairement à l'ordre lexicographique, on ne peut pas se contenter de l'appliquer uniquement sur les thèses, puisque parmi toutes les sous-bases consistantes, les éléments maximaux de l'ordre "BEST-OUT" ne sont pas nécessairement les thèses.

3. Il s'agit d'une préférence très peu sélective.

4. Les résultats obtenus avec cet ordre sont tout à fait différents de ceux obtenus par les méthodes vues précédemment, excepté dans le cas d'une base avec une strate unique (voir point suivant), et leur interprétation n'est pas évidente.

5. Dans le cas d'une base avec une strate unique, toutes les sous-bases consistantes sont préférées, donc en particulier les thèses sont toutes préférées.

6. L'ordre "BEST-OUT" permet d'ordonner totalement l'ensemble des sous-bases consistantes.

7. Les relations d'inférence issues de cet ordre présentent un inconvénient majeur : le "drowning effect" (l'effet de noyade) (voir [BCD$^+$93]).

8. L'ordre "BEST-OUT" a une longue histoire. Il est issu du principe du minimum de spécificité (voir [BDP92] et [BDP94]). Il correspond aussi au système Z de [Pea90], à la fermeture rationnelle de Lehmann et Magidor ([LM92]). La liste n'est pas exhaustive.

les A consistants tels que $a(A) = 1$: 58 solutions :	$\{P \to O, P \to \neg V, O \to V, O \to A, A \to V\}$: sol. 28
$\{\to P, P \to O, P \to \neg V, O \to A\}$: sol. 1	$\{P \to O, P \to \neg V, O \to V, O \to A\}$: sol. 29
$\{\to P, P \to O, P \to \neg V, A \to V\}$: sol. 2	$\{P \to O, P \to \neg V, O \to V, A \to V\}$: sol. 30
$\{\to P, P \to O, P \to \neg V\}$: sol. 3	$\{P \to O, P \to \neg V, O \to V\}$: sol. 31
$\{\to P, P \to O, O \to V, O \to A, A \to V\}$: sol. 4	$\{P \to O, P \to \neg V, O \to A, A \to V\}$: sol. 32
$\{\to P, P \to O, O \to V, O \to A\}$: sol. 5	$\{P \to O, P \to \neg V, O \to A\}$: sol. 33
$\{\to P, P \to O, O \to V, A \to V\}$: sol. 6	$\{P \to O, P \to \neg V, A \to V\}$: sol. 34
$\{\to P, P \to O, O \to V\}$: sol. 7	$\{P \to O, P \to \neg V\}$: sol. 35
$\{\to P, P \to O, O \to A, A \to V\}$: sol. 8	$\{P \to O, O \to V, O \to A, A \to V\}$: sol. 36
$\{\to P, P \to O, O \to A\}$: sol. 9	$\{P \to O, O \to V, O \to A\}$: sol. 37
$\{\to P, P \to O, A \to V\}$: sol. 10	$\{P \to O, O \to V, A \to V\}$: sol. 38
$\{\to P, P \to O\}$: sol. 11	$\{P \to O, O \to V\}$: sol. 39
$\{\to P, P \to \neg V, O \to V, O \to A, A \to V\}$: sol. 12	$\{P \to O, O \to A, A \to V\}$: sol. 40
$\{\to P, P \to \neg V, O \to V, O \to A\}$: sol. 13	$\{P \to O, O \to A\}$: sol. 41
$\{\to P, P \to \neg V, O \to V, A \to V\}$: sol. 14	$\{P \to O, A \to V\}$: sol. 42
$\{\to P, P \to \neg V, O \to V\}$: sol. 15	$\{P \to O\}$: sol. 43
$\{\to P, P \to \neg V, O \to A, A \to V\}$: sol. 16	$\{P \to \neg V, O \to V, O \to A, A \to V\}$: sol. 44
$\{\to P, P \to \neg V, O \to A\}$: sol. 17	$\{P \to \neg V, O \to V, O \to A\}$: sol. 45
$\{\to P, P \to \neg V, A \to V\}$: sol. 18	$\{P \to \neg V, O \to V\}$: sol. 46
$\{\to P, P \to \neg V\}$: sol. 19	$\{P \to \neg V, O \to V, A \to V\}$: sol. 47
$\{\to P, O \to V, O \to A, A \to V\}$: sol. 20	$\{P \to \neg V, O \to A, A \to V\}$: sol. 48
$\{\to P, O \to V, O \to A\}$: sol. 21	$\{P \to \neg V, O \to A\}$: sol. 49
$\{\to P, O \to V, A \to V\}$: sol. 22	$\{P \to \neg V, A \to V\}$: sol. 50
$\{\to P, O \to V\}$: sol. 23	$\{P \to \neg V\}$: sol. 51
$\{\to P, O \to A, A \to V\}$: sol. 24	$\{O \to V, O \to A, A \to V\}$: sol. 52
$\{\to P, O \to A\}$: sol. 25	$\{O \to V, O \to A, \}$: sol. 53
$\{\to P, A \to V\}$: sol. 26	$\{O \to V, A \to V\}$: sol. 54
$\{\to P\}$: sol. 27	$\{O \to V\}$: sol. 55
	$\{O \to A, A \to V\}$: sol. 56
	$\{O \to A\}$: sol. 57
	$\{A \to V\}$: sol. 58

Tableau 2.3: L'ordre "BEST-OUT" (fin)

2.5 Les ensembles de Gärdenfors et Makinson

Gärdenfors et Makinson ont défini une nouvelle classe de relations d'inférence non monotone : les relations d'inférence expectatives (voir [Gär91] et [GM94]). Leur but est d'obtenir une relation d'inférence non monotone vérifiant des propriétés choisies parmi celles définies dans [KLM90], [Gär91] et [GM94]. Contrairement à la plupart des formalismes vus jusqu'à présent dans cette section, ils se situent dans le cadre d'une approche sémantique telle que décrite dans la section 1 : utilisation du lien sémantique et de la dépendance logique existant entre les croyances.

Ils présentent 2 méthodes :

- l'utilisation d'ensembles d'"expectations" (sous-ensembles de formules choisis dans l'ensemble de formules initial),
- l'utilisation d'un ordre expectatif (ordre entre les formules du langage possédant certaines propriétés).

Considérons la base de formules E. Les deux définitions de cette relation d'inférence expectative sont :

Définition 2.5.1 (utilisation d'ensembles d'"expectations") $f \mid\sim g$ *ssi*
$g \in C_{E,S}(f) = \cap\{Cn(\{f\} \cup D)$ *avec* $D \in S(E\perp\neg f)\}$ *où :*

- $Cn(X) =$ *ensemble des conséquences logiques de* X *au sens de la logique classique,*
- $S(B) =$ *fonction de sélection renvoyant certains éléments de* B*,*
- $E\perp h =$ *sous-ensemble maximal de* E *qui ne peut impliquer logiquement* h*.*

Définition 2.5.2 (utilisation d'un ordre expectatif) $f \mid\sim g$ *ssi*
$g \in Cn(\{f\} \cup \{h \in E$ *tel que* $\neg f < h\})$ *avec l'ordre* $<$ *entre les formules qui est total et qui vérifie les propriétés de dominance et de conjonctivité (voir [GM94]).*

Notons que $E\perp\neg f$ est donc l'ensemble des sous-bases de E qui sont maximales au sens de l'inclusion et consistantes avec f.

Remarque : la définition de la fonction S donnée dans la définition 2.5.1 est tout à fait informelle.

Dans le cas des ensembles d'"expectations", les propriétés de la relation d'inférence ainsi définie dépendent intimement de la construction de la fonction de sélection S.

Dans le cas d'un ordre expectatif, Gärdenfors et Makinson ont démontré que la relation d'inférence expectative ainsi définie vérifie tout un ensemble de propriétés particulièrement intéressantes (la supra-classicité, l'équivalente logique gauche, le et, la cumulativité, le ou, la monotonie rationnelle et la préservation de la consistance – voir dans [KLM90], [Gär91] et [GM94] la définition de chacune de ces propriétés).

Ils ont d'autre part démontré l'équivalence des 2 méthodes *à condition que* la fonction S soit construite à partir d'un ordre expectatif sur les formules de la base.

Comment situer cette approche par rapport aux approches présentées précédemment ?

Disons tout d'abord que toutes les méthodes précédentes ("INCLUSION BASED", lexicographique, "BEST-OUT", etc.) visent à utiliser conjointement des sous-ensembles de formules choisis dans l'ensemble de formules initial et un ordre entre les formules.

L'utilisation de sous-ensembles choisis correspond à la construction d'une fonction S de sélection. Ces approches pourraient se situer éventuellement dans le cadre des ensembles d'"expectations" de Gärdenfors et Makinson. Toutefois, l'ordre entre les formules sous-jacent à la définition de cette fonction S n'est pas un ordre expectatif ; en effet, nous n'imposons rien à l'ordre $<$, à part dans certains cas le fait qu'il soit total. Ces approches ne correspondent donc pas à une relation d'inférence expectative au sens des ensembles d'"expectations".

D'autre part, pour la même raison, on ne peut pas considérer que les méthodes précédentes correspondent à l'utilisation d'un ordre expectatif. Ce ne sont donc pas des relations d'inférence expectatives au sens de l'ordre expectatif.

CONCLUSION :

1. L'approche de Gärdenfors et Makinson différencie[12] la construction de sous-ensembles de formules particuliers issus de la base initiale et l'utilisation d'un ordre entre les formules de la base, alors que les autres approches vues jusqu'à présent ont plutôt tendance à utiliser conjointement ces aspects.

2. Cette approche vise plutôt la définition d'une relation d'inférence possédant certaines propriétés choisies parmi celles présentées dans [KLM90], [Gär91] et [GM94]. En conséquence, elle impose à l'ordre utilisé (soit directement par la méthode de l'ordre expectatif, soit indirectement par la définition de la fonction de sélection S) un ensemble de propriétés (par exemple, la dominance[13]).

3. Les autres approches vues jusqu'alors n'imposent rien à l'ordre $<$. Ce ne sont donc pas des approches "expectatives".

4. Cette approche est sémantique, contrairement aux méthodes présentées précédemment qui étaient pour la plupart syntaxiques.

[12] Même s'ils montrent ensuite les liens étroits entre ces deux aspects, en particulier du point de vue des propriétés recherchées pour la relation d'inférence expectative.

[13] Pour un ordre $<$ donné, la propriété de dominance est : soient P et Q, 2 formules de notre langage, si $P \vdash Q$ alors $P < Q$.

Chapitre 3

Les différents principes d'inférence

On se donne $P^E = \{P_1^E, \ldots, P_n^E\}$, un ensemble de sous-bases consistantes, et $\ll$, une relation de préférence (au moins un pré-ordre) sur ces sous-bases. On veut étudier différents moyens d'inférer de façon non monotone une formule Φ (par exemple, on peut choisir d'inférer la formule Φ à partir de $(P^E, \ll)$ si et seulement si toutes les sous-bases P_i^E de P^E infèrent classiquement Φ). Pour ce faire, l'objectif à atteindre est d'utiliser au maximum les techniques d'inférence classique.

Remarques importantes :

- Ce chapitre s'inspire très largement des travaux de Pinkas et Loui (voir [PL92]).
- Ici, nous ne nous attachons pas à la construction des sous-bases et à l'élaboration de l'ordre entre les sous-bases. En effet, le mécanisme de définition de l'ensemble P^E et de l'ordre $\ll$ est indépendant du type d'inférence choisi[1].

Nous décrivons ici les principaux mécanismes d'inférence.

3.1 La conséquence forte

Il s'agit de la mise en œuvre d'un raisonnement circonspect (prudent)[2] :

Définition 3.1.1 $(P^E, \ll) \mid\!\sim^{\forall} \Phi$ *ssi quelle que soit la sous-base* P_i^E *élément de* P^E*, on a* $P_i^E \vdash \Phi$.

Ce principe d'inférence est appelé conséquence forte et il sera noté avec le symbole $\mid\!\sim^{\forall}$.

Remarques :

- Dans ce principe d'inférence, l'ordre $\ll$ n'intervient pas.
- Nous appellerons ce principe : le principe UNI (UNI pour universel).
- Ce principe présente la particularité suivante : parmi toutes les sous-bases P_i^E de P^E, on peut avoir la sous-base vide ; la définition devient alors : Si $\exists i$ tel que $P_i^E = \varnothing$ alors $((P^E, \ll) \mid\!\sim^{\forall} \Phi \Leftrightarrow \Phi$ est une tautologie). Ce cas se présente rarement. Lorsqu'il se produira, nous présenterons 2 résultats différents, un pour le cas où $\varnothing$ est inclus dans P^E, et un pour le cas où $\varnothing$ est exclu de P^E.

[1] Même si, historiquement, les auteurs d'un mécanisme de définition de P^E avaient prévu de l'utiliser avec un principe d'inférence particulier.

[2] On prend le plus possible de précautions avant d'inférer une nouvelle information. Par opposition, on parlera de raisonnement crédule. Cette notion de précaution est présentée dans la section 3.7.

3.2 La conséquence faible

Par opposition au principe précédent, nous avons ici la mise en œuvre d'un raisonnement crédule :

Définition 3.2.1 $(P^E, \ll) \mid\!\sim^{\exists} \Phi$ *ssi il existe au moins une sous-base* P_i^E *élément de* P^E *telle que* $P_i^E \vdash \Phi$.

Ce principe d'inférence est appelé conséquence faible et il sera noté avec le symbole $\mid\!\sim^{\exists}$.
Remarques :

- Dans ce principe d'inférence, l'ordre $\ll$ n'intervient pas.
- Nous appellerons ce principe : le principe EXI (EXI pour existentiel).

3.3 La conséquence argumentative

L'idée exploitée ici est la suivante : Φ est conséquence faible mais $\neg\Phi$ n'est pas conséquence faible.

Définition 3.3.1 $(P^E, \ll) \mid\!\sim^{A} \Phi$ *ssi il existe au moins une sous-base* P_i^E *élément de* P^E *telle que* $P_i^E \vdash \Phi$ *et quelle que soit* P_j^E *élément de* P^E, *on a* $P_j^E \not\vdash \neg\Phi$.

Ce principe d'inférence est appelé conséquence argumentative et il sera noté avec le symbole $\mid\!\sim^{A}$.
Remarques :

- Dans ce principe d'inférence, l'ordre $\ll$ n'intervient pas.
- Nous appellerons ce principe : le principe ARG (ARG pour argumentatif).
- Ce principe est défini aussi par [BDP93].

3.4 La conséquence forte avec préférences

Prenons maintenant en compte des préférences entre les sous-bases. En les associant à un raisonnement circonspect, on obtient :

Définition 3.4.1 $(P^E, \ll) \mid\!\sim^{\forall,pref} \Phi$ *ssi quelle que soit la sous-base* P_i^E *élément de* P^E *et préférée selon l'ordre* $\ll$, *on a* $P_i^E \vdash \Phi$.

Ce principe d'inférence est appelé conséquence forte avec préférences et il sera noté avec le symbole $\mid\!\sim^{\forall,\text{pref}}$ avec "pref" fonction de l'ordre $\ll$.
Remarques :

- Dans ce principe d'inférence, l'ordre $\ll$ intervient ; il sert à définir les sous-bases préférées de P^E.
- Nous appellerons ce principe : le principe UNI-PREF (UNI pour universel et PREF pour l'ordre $\ll$).
- Même remarque que pour UNI sur l'inclusion de l'ensemble vide dans P^E.

3.5 La conséquence faible avec préférences

Associons des préférences et un raisonnement crédule :

Définition 3.5.1 $(P^E, \ll) \mid\!\sim^{\exists,pref} \Phi$ *ssi il existe au moins une sous-base* P_i^E *élément de* P^E *et préférée selon l'ordre* $\ll$ *telle que* $P_i^E \vdash \Phi$.

Ce principe d'inférence est appelé conséquence faible avec préférences et il sera noté avec le symbole $\mid\!\sim^{\exists,\text{pref}}$ avec "pref" fonction de l'ordre $\ll$.

Remarques :

- Dans ce principe d'inférence, l'ordre $\ll$ intervient ; il sert à définir les sous-bases préférées de P^E.
- Nous appellerons ce principe : le principe EXI-PREF (EXI pour existentiel et PREF pour l'ordre $\ll$).

3.6 La conséquence argumentative avec préférences

Associons des préférences et l'idée de la conséquence argumentative :

Définition 3.6.1 $(P^E, \ll) \mid\!\sim^{A,pref} \Phi$ *ssi :*

- *il existe au moins une sous-base P_i^E élément de P^E et préférée selon l'ordre $\ll$ telle que $P_i^E \vdash \Phi$,*
- *et quelle que soit P_j^E élément de P^E préférée selon l'ordre $\ll$, on a $P_j^E \nvdash \neg\Phi$.*

Ce principe d'inférence est appelé conséquence argumentative avec préférences et il sera noté avec le symbole $\mid\!\sim^{A,\text{pref}}$ avec "pref" fonction de l'ordre $\ll$.

Remarques :

- Dans ce principe d'inférence, l'ordre $\ll$ intervient ; il sert à définir les sous-bases préférées de P^E.
- Nous appellerons ce principe : le principe ARG-PREF (ARG pour argumentatif et PREF pour l'ordre $\ll$).
- Ce principe est défini aussi par [BDP93].

3.7 Comparaison rapide des divers principes

Ici, nous allons faire une comparaison à la "Pinkas et Loui" (voir [PL92]). C'est à dire que nous allons ordonner les 6 principes d'inférence précédents en fonction de la précaution du raisonnement utilisé. Rappelons d'abord quelques définitions issues de [PL92] :

Définition 3.7.1 *Notation : à chaque principe d'inférence p, est associée la relation R^p qui est l'ensemble des couples (X, Y) liés par le principe d'inférence p. Ce qui revient à dire que $\forall p$, $R^p = \{(X_1, Y_1), \ldots, (X_n, Y_n)\}$ avec $\forall i$, $X_i \mid\!\sim^p Y_i$. On dira que R^p est une relation de conséquence.*

Définition 3.7.2 *Soient R_1 et R_2, deux relations de conséquence. On dit que R_1 est plus "précautionneuse" que R_2 (noté R_1 +prec R_2) ssi R_1 est incluse dans R_2.*

L'ordre +prec est un ordre partiel.

Or, nous pouvons faire diverses remarques sur les principes d'inférence précédents :

- $(\forall P_i^E, P_i^E \vdash \Phi) \Rightarrow (\exists P_i^E, P_i^E \vdash \Phi)$,
- $(\forall P_i^E, P_i^E \vdash \Phi) \Rightarrow (\exists P_i^E, P_i^E \vdash \Phi$ et $\forall P_i^E, P_i^E \nvdash \neg\Phi)$,
- $(\exists P_i^E, P_i^E \vdash \Phi$ et $\forall P_i^E, P_i^E \nvdash \neg\Phi) \Rightarrow (\exists P_i^E, P_i^E \vdash \Phi)$,
- $(\forall P_i^E, P_i^E \vdash \Phi) \Rightarrow (\forall P_i^E$ préférée, $P_i^E \vdash \Phi)$,
- $(\exists P_i^E$ préférée, $P_i^E \vdash \Phi) \Rightarrow (\exists P_i^E, P_i^E \vdash \Phi)$,
- $(\forall P_i^E$ préférée, $P_i^E \vdash \Phi) \Rightarrow (\exists P_i^E$ préférée, $P_i^E \vdash \Phi)$,
- $(\forall P_i^E$ préférée, $P_i^E \vdash \Phi) \Rightarrow (\exists P_i^E$ préférée, $P_i^E \vdash \Phi$ et $\forall P_i^E$ préférée, $P_i^E \nvdash \neg\Phi)$,
- $(\exists P_i^E$ préférée, $P_i^E \vdash \Phi$ et $\forall P_i^E$ préférée, $P_i^E \nvdash \neg\Phi) \Rightarrow (\exists P_i^E$ préférée, $P_i^E \vdash \Phi)$.

On obtient donc la figure 3.1 représentant l'ordre partiel de précaution (le lien $R_1^p \rightarrow R_2^p$ signifie que R_1^p est plus précautionneuse que R_2^p).

Remarque : C'est un schéma de base qui pourra être raffiné en introduisant différents types de préférence.

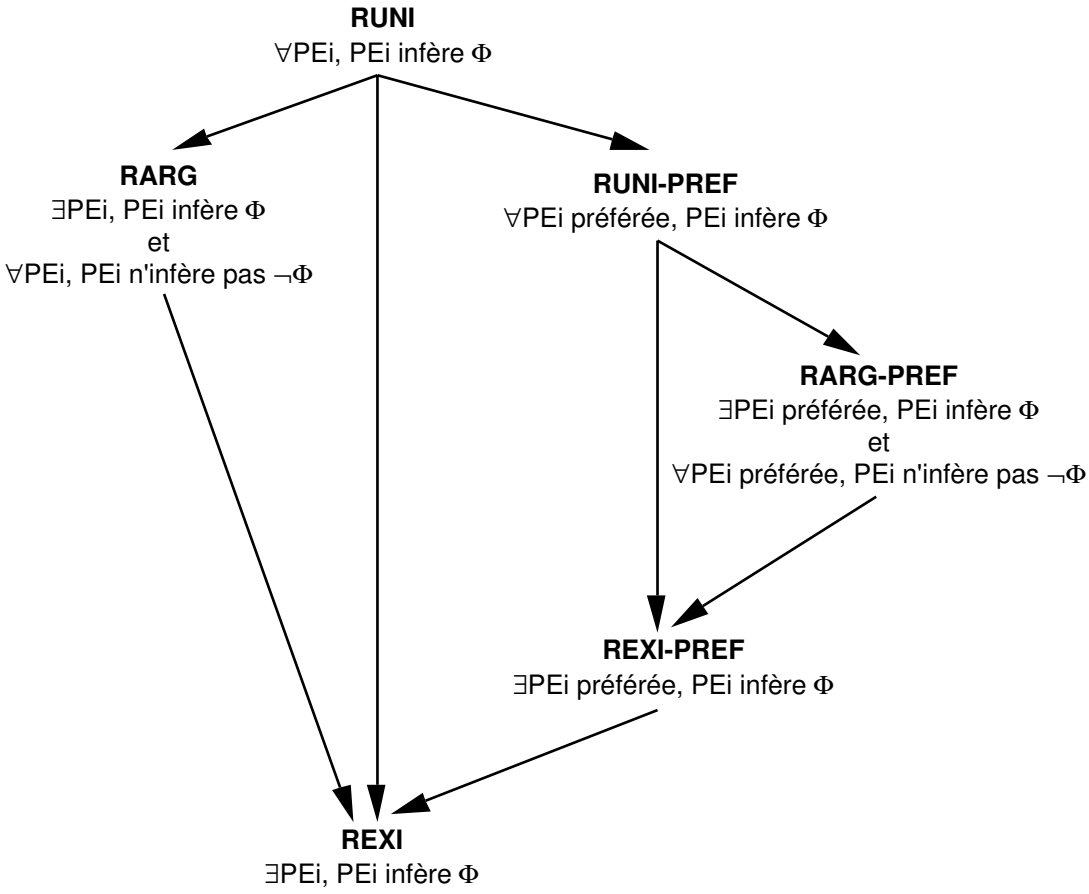


Figure 3.1: Ordre de précaution

3.8 Les autres principes

Dans l'article de Pinkas et Loui, sont cités de nombreux autres principes d'inférence qui proviennent essentiellement d'une combinaison entre les principes vus ci-dessus et que nous nous contenterons de citer sans les étudier plus en détail. On a par exemple :

- $(P^E, \ll) \mid\sim \Phi$ ssi $\forall P_i^E$ préférée selon l'ordre $\ll$, on a $P_i^E \vdash \Phi$ ou $\exists P_j^E$ telle que $P_j^E \nvdash \neg\Phi$.
- $(P^E, \ll) \mid\sim \Phi$ ssi $\exists P_i^E$ préférée selon l'ordre $\ll$ telle que $P_i^E \vdash \Phi$ et $\forall P_j^E$ on a $P_j^E \nvdash \neg\Phi$.
- $(P^E, \ll) \mid\sim \Phi$ ssi $\exists P_i^E$ telle que $P_i^E \vdash \Phi$ et $\forall P_j^E$ telle que $P_i^E \ll P_j^E$, on a $P_j^E \nvdash \neg\Phi$ (avec 2 cas possibles : $P_i^E \ll P_j^E$ vu de manière stricte ou non).

Chapitre 4

Les relations d'inférence à étudier

Il s'agit ici de construire un tableau de synthèse des deux chapitres précédents.

	Sous-bases consis-tantes sans ordre	Sous-bases consis-tantes avec ordre "BEST-OUT"	Thèses sans ordre	Thèses avec ordre basé sur l'inclusion	Thèses avec ordre basé sur la cardinalité	Thèses avec ordre lexi-cographique	Extensions Logique des défauts sans ordre
UNI	UNI-S		UNI-T				UNI-E
EXI	EXI-S		EXI-T				EXI-E
ARG	ARG-S		ARG-T				ARG-E
UNI-PREF		UNI-BO		UNI-INCL	UNI-CAR	UNI-LEX	
EXI-PREF		EXI-BO		EXI-INCL	EXI-CAR	EXI-LEX	
ARG-PREF		ARG-BO		ARG-INCL	ARG-CAR	ARG-LEX	

Légende du tableau : Nous trouvons en colonne, les différents mécanismes de définition d'un ensemble P^E et d'un ordre sur P^E et nous avons en ligne, les différents principes d'inférence considérés (uniquement les principaux).

Pourquoi un tel découpage ? Tout d'abord, l'ensemble P^E peut être constitué par 3 types de sous-bases (voir chapitres précédents) :

- soit de sous-bases consistantes,
- soit uniquement de thèses (sous-bases maximales consistantes),
- soit d'extensions au sens de la logique des défauts.

De plus, dans les chapitres précédents, nous avons aussi défini 4 types de préférences sur des sous-ensembles de formules consistants :

- soit un ordre de type ordre "INCLUSION BASED" ([Bre89b, CRS92, DLP91]), prévu pour ordonner des thèses[1], et qui est une généralisation de l'utilisation de thèses ;

[1] En fait, suivant les auteurs, l'ordre "INCLUSION BASED" est prévu pour ordonner des thèses ou simplement des sous-bases consistantes. Par exemple, Brewka l'a défini sur des thèses, alors que l'ordre démocratique de Cayrol, Royer et Saurel est applicable sur des sous-bases simplement consistantes. Toutefois, on démontre que toute sous-base préférée pour l'ordre "INCLUSION BASED" est systématiquement une thèse. C'est la raison pour laquelle cet ordre ne sera étudié que sur des thèses.

- soit l'ordre basé sur la cardinalité ([DLP91]), prévu pour ordonner initialement des sous-bases consistantes issues d'une base non ordonnée, mais pour lequel les éléments préférés sont par construction des thèses ;
- soit l'ordre lexicographique basé sur la cardinalité ([DLP91]), qui est une généralisation de l'ordre basé sur la cardinalité, tout comme l'ordre "INCLUSION BASED" est une généralisation de l'utilisation de thèses ; cet ordre est lui aussi prévu pour ordonner initialement des sous-bases consistantes mais ses éléments préférés sont par construction des thèses ;
- soit l'ordre "BEST-OUT" ([BCD$^+$93, DLP91]), prévu pour ordonner des sous-bases consistantes.

Partout où les principes UNI, EXI, ARG s'appliquent, on obtient une relation d'inférence notée respectivement UNI-X, EXI-X, ARG-X avec X remplacé par une lettre explicitant le type de l'ensemble P^E utilisé (S = sous-bases consistantes, T = thèses, E = extensions).

Partout où les principes UNI-PREF, EXI-PREF, ARG-PREF s'appliquent, on obtient une relation d'inférence notée respectivement UNI-PREF, EXI-PREF, ARG-PREF avec PREF remplacé par un mot explicitant l'ordre utilisé (BO = ordre "BEST-OUT", INCL = ordre basé sur l'inclusion, CAR = ordre basé sur la cardinalité, LEX = ordre lexicographique) et donc, par là même, le type de l'ensemble P^E (sous-bases consistantes pour BO et thèses pour INCL, CAR et LEX).

Nous nous trouvons donc à la tête de 21 relations d'inférence non monotone, qu'il va falloir étudier.

On peut rajouter des lignes à notre tableau en définissant de nouveaux principes d'inférence (par exemple ceux cités par Pinkas et Loui), ou des colonnes en définissant de nouveaux couples $(P^E, \ll)$.

Signalons que la complexité de certaines relations d'inférence non monotone définies ci-dessus ont déjà été étudiées du point de vue de la complexité, soit directement (par exemple, les relations UNI-E et EXI-E par Gottlob dans [Got92]), soit indirectement (par exemple, les relations UNI-T et UNI-INCL par Nebel dans [Neb91], au travers des mécanismes de révision associés).

Chapitre 5

Rappels en complexité

Les rappels fournis ici sont volontairement succincts et nous conseillons vivement aux personnes intéressées de se reporter aux ouvrages de référence dans le domaine (par exemple, [GJ79]).

La complexité à laquelle on s'intéresse est la complexité temporelle[1] de résolution d'un problème dans le pire cas. Il s'agit de caractériser la façon dont évolue le temps de résolution d'un problème suivant la taille des données et ceci dans le pire cas, afin d'obtenir une borne supérieure de la complexité réelle. Naturellement, les notions de temps, de résolution et de problème devront être formalisées.

Une distinction est habituellement faite entre les problèmes dits "faciles", dont la complexité dans le pire des cas est bornée par une fonction polynomiale de la taille des données (exemple : tri d'une liste d'entiers) et les autres problèmes (exponentiels, super exponentiels...). Ce premier classement voit son intérêt fortement limité par la difficulté qu'il peut y avoir à montrer qu'un problème *ne peut pas être résolu en temps polynomial* : pour de très nombreux problèmes, d'utilité pratique importante, on ne connaît pas d'algorithme polynomial et on n'a pas pu démontrer qu'il n'en existait pas.

La théorie de la complexité offre un palliatif à cette situation en introduisant la notion d'algorithmes non déterministes polynomiaux (NP, contenant la classe des algorithmes polynomiaux) et en distinguant parmi ceux-ci la classe particulièrement riche des algorithmes "les plus difficiles de NP" (ou NP-complets : elle contient de très nombreux problèmes classiques de logique, de recherche opérationnelle...) telle que si l'un de ces problèmes se résout en temps polynomial (personne n'a encore réussi cet exploit), alors tous les problèmes de NP se résolvent en temps polynomial.

Dire d'un problème qu'il est NP-complet montre alors que le problème est aussi difficile que bien d'autres problèmes pour lesquels une gigantesque communauté de chercheurs n'a pas trouvé d'algorithme polynomial depuis plus de 30 ans.

Cette simple distinction P/NP a été ensuite enrichie via la définition d'une hiérarchie de classes de problèmes, de plus en plus difficiles. Dans le cas qui nous occupe, nous cherchons à situer dans cette hiérarchie, les différents mécanismes d'inférence afin de pouvoir les comparer.

Avant toute précision supplémentaire, voyons comment l'on peut formaliser les notions de problème et d'algorithme.

5.1 Qu'est-ce-qu'un problème ? Comment le formaliser ?

Définition 5.1.1 *Un problème est une question générique (qui s'applique à un ensemble d'éléments).*

Exemple de problème : "Déterminer si un entier naturel est pair ou impair".

[1] Il existe aussi une complexité spatiale (voir [GJ79]).

Définition 5.1.2 *Une instance du problème est la question posée pour un élément particulier de l'ensemble. C'est donc une version du problème dans laquelle les paramètres non spécifiés dans le problème initial ont été instanciés.*

Exemple d'un problème et d'une instance de problème : le problème "Soit A un nombre entier naturel, A est-il pair ou impair ?" admet comme instances "3 est-il pair ou impair ?", "15624 est-il pair ou impair ?" etc.

Afin de simplifier les notations, une instance d'un problème donné Q ne sera notée qu'avec l'élément particulier qui la définit. En effet, pour un problème donné et quelle que soit son instance, la question posée reste toujours la même. Ainsi, au lieu de dire que "3 est-il pair ou impair ?" est une instance du problème Q, on dira que "3" est une instance de Q.

À chaque instance de problème est associée une réponse.

Définition 5.1.3 *Un algorithme résout un problème s'il peut être appliqué à toute instance du problème pour fournir une réponse en un temps fini.*

Définition 5.1.4 *Un problème de décision est un problème dont la réponse est oui ou non, quelle que soit l'instance du problème.*

Exemple : "Soit A un nombre entier naturel, A est-il premier ?".

On parlera donc d'instances positives (celles dont la réponse est oui), et d'instances négatives (celles dont la réponse est non). "3" est une instance positive et "15624" est une instance négative du problème défini par "Soit A un entier naturel, A est-il premier ?".

Définition 5.1.5 *Un problème est décidable s'il existe un algorithme pour résoudre toutes ses instances, sinon il est dit indécidable.*

En complexité, on n'étudie que les problèmes décidables.

À chaque instance I d'un problème Q, on associe par une fonction de codage e un mot d'un langage formel L (voir [GJ79]). La fonction d'encodage n'est pas quelconque et doit être raisonnable (non redondante et décodable). Désormais, chaque problème Q sera vu à travers son langage L. On identifiera Q et L. On appelle taille de l'instance I, la taille du mot $e(I)$.

Définition 5.1.6 *Dans le cas des problèmes de décision, on définit la notion de problème complémentaire : soit un problème Q, le problème complémentaire de Q, noté co-Q, est le problème dont la solution est oui quand celle de Q est non et vice-versa.*

Remarque : La notation "co-" s'applique aussi aux langages : L et co-L.

Exemple : Le co-problème de "Soit A un nombre entier naturel, A est-il premier ?" sera : "Soit A un nombre entier naturel, A est-il le produit de 2 nombres entiers naturels strictement inférieurs à A ?".

5.2 Les machines de Turing et les classes de problèmes

Nous avons vu à la section précédente que l'on peut coder une instance d'un problème donné. Il faut maintenant définir une machine sur laquelle résoudre le problème encodé. Cette machine est la machine de TURING. C'est un automate auquel on prête la propriété suivante : toute fonction intuitivement calculable peut l'être par une machine de TURING[2].

[2] C'est la thèse de CHURCH-TURING.

Définition 5.2.1 *On dit qu'une machine de TURING résout un problème Q, quand, pour toute instance I de Q et pour un encodage donné, la machine s'arrête en renvoyant la réponse oui quand I est une instance positive (on dit que le mot représentant I est accepté par la machine) et la réponse non quand I est une instance négative (on dit que le mot représentant I est rejeté par la machine).*

Définition 5.2.2 *La complexité temporelle d'un algorithme est une fonction qui associe à la taille n d'une instance (la longueur du mot représentant l'instance) le nombre de pas d'exécution effectués par la machine jusqu'à trouver la solution et ce pour le pire cas (sur tous les mots de longueur n).*

Il existe plusieurs types de machines de TURING, chacune permettant de définir une classe de problèmes.

Définition 5.2.3 *Les machines déterministes*[3] *induisent la classe P des problèmes dits polynomiaux :*

> *un problème est polynomial, s'il existe un algorithme pour le résoudre sur une machine de TURING déterministe dont la complexité est bornée par une fonction polynomiale de la taille du problème.*

Définition 5.2.4 *Les machines non déterministes*[4] *induisent la classe NP des problèmes dits non déterministes polynomiaux :*

> *un problème est non déterministe polynomial, s'il existe un algorithme pour le résoudre sur une machine de TURING non déterministe dont la complexité est bornée par une fonction polynomiale de la taille du problème*[5].

Définition 5.2.5 *Les machines déterministes à oracle de complexité X*[6] *induisent la classe P^X :*

> *un problème est de complexité P^X, s'il existe un algorithme pour le résoudre sur une machine de TURING déterministe à oracle de complexité X dont la complexité est bornée par une fonction polynomiale de la taille du problème.*

Définition 5.2.6 *Les machines non déterministes à oracle de complexité X induisent la classe NP^X :*

> *un problème est de complexité NP^X, s'il existe un algorithme pour le résoudre sur une machine de TURING non déterministe à oracle de complexité X dont la complexité est bornée par une fonction polynomiale de la taille du problème.*

On peut définir une hiérarchie de classes de problèmes. C'est la hiérarchie polynomiale :

- $\Sigma_0^p = \Delta_0^p = \Pi_0^p = \mathrm{P}$ (machines déterministes)
- $\Delta_{k+1}^p = \mathrm{P}^{\Sigma_k^p}$ (machines déterministes à oracle de complexité Σ_k^p)
- $\Sigma_{k+1}^p = \mathrm{NP}^{\Sigma_k^p}$ (machines non déterministes à oracle de complexité Σ_k^p)
- $\Pi_{k+1}^p = \text{co-}\Sigma_{k+1}^p$ (complémentaire)

[3] Informellement, une machine de TURING déterministe est une machine de TURING qui, à partir d'un état donné, ne peut passer que dans un seul autre état. Rappelons que les machines de TURING sont des automates passant d'un état à un autre en fonction des entrées et produisant des sorties.

[4] Par opposition aux machines déterministes, dans une machine non déterministe, à partir d'un état donné, on peut passer dans un état *choisi* parmi plusieurs états différents. Un problème NP est donc un problème qui peut se résoudre efficacement si on sait toujours faire les bons choix.

[5] La complexité temporelle d'un algorithme dans le cas d'une machine de TURING non déterministe = le maximum sur toutes les instances positives, donc pour tous les mots $e(I)$ correspondants, du temps de reconnaissance de ces mots ; sachant que ce temps de reconnaissance pour un mot $e(I)$ donné = le minimum sur tous les choix possibles du nombre de pas d'exécution effectués jusqu'à trouver la solution.

[6] Informellement, une machine de TURING à oracle de complexité X est une machine de TURING faisant appel à un sous-programme exécuté sur une autre machine et sachant résoudre les problèmes de la classe X. Chaque appel compte alors pour un pas d'exécution de la machine appelante.

Remarque : Δ_1^p = P, Σ_1^p = NP, Π_1^p = co-NP.

Quand un problème Q est de complexité X, alors son co-problème sera de complexité co-X.

Introduisons trois dernières notions : la transformation polynomiale, la complétude et la notion de problème X-difficile.

Définition 5.2.7 *Une transformation polynomiale est une fonction f calculable en temps polynomial permettant de "passer" d'un langage L_1 à un langage L_2 (donc d'un problème Q_1 à un problème Q_2) – voir définition précise dans l'annexe A. On note : $L_1 \propto L_2$.*

Définition 5.2.8 *Un problème Q est X-complet avec X représentant une classe de complexité, ssi Q est de complexité X et $\forall Q'$ un problème de complexité X, Q' se ramène à Q par une transformation polynomiale.*

Définition 5.2.9 *Un problème Q est X-difficile avec X représentant une classe de complexité, ssi $\forall Q'$ un problème de complexité X, Q' se ramène à Q par une transformation polynomiale.*

5.3 Les exemples de problèmes

Exemple d'un problème de classe P : "Soient A et B, 2 ensembles finis, A est-il inclus dans B ?".

Exemple d'un problème de classe NP et même NP-complet : le problème de satisfiabilité en logique propositionnelle "Soit Φ une formule propositionnelle, Φ est-elle satisfiable ?". Ce problème est appelé GSAT. Ici nous nous intéressons au problème de satisfiabilité d'une formule propositionnelle quelconque, appelé "SATISFIABILITY OF BOOLEAN EXPRESSIONS" par Garey et Johnson dans [GJ79] et noté le plus souvent GSAT. Il existe aussi un problème SAT défini par Garey et Johnson dans [GJ79] qui ne concerne que le problème de satisfiabilité d'une formule propositionnelle exprimée sous forme normale conjonctive. Ces deux problèmes sont de même complexité.

Exemple d'un problème de classe co-NP et même co-NP-complet : le complémentaire du problème de satisfiabilité en logique propositionnelle "Soit Φ une formule propositionnelle, Φ est-elle non satisfiable ?". Ce problème est appelé UNGSAT[7] (ou co-GSAT ou GVALID[8]).

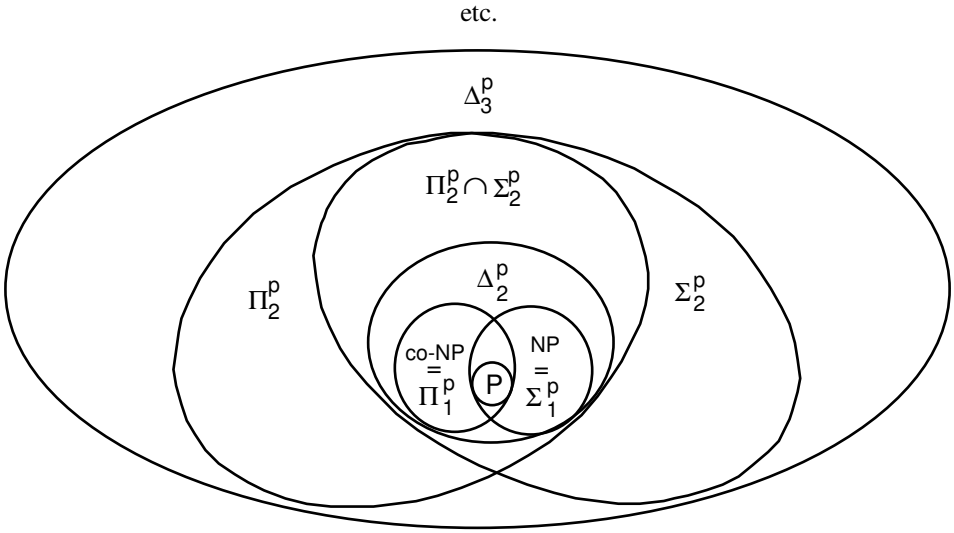
Exemple d'un problème de classe Σ_2^p et même Σ_2^p-complet : le problème 2-QBF défini par : "La formule $\exists a \forall b H(a,b)$ est-elle satisfiable ?". $\exists a \forall b H(a,b)$ est une formule d'une méta-logique dans laquelle a et b sont 2 vecteurs représentant les assignations des ensembles de variables propositionnelles $(a_1, \ldots, a_n)$ et $(b_1, \ldots, b_m)$. "$\exists a \forall b H(a,b)$ satisfiable" signifie donc : "il existe une assignation a des variables propositionnelles $a_1, \ldots, a_n$, telle que, quelque soit l'assignation b des variables propositionnelles $b_1, \ldots, b_m$, la formule $H(a,b)$ est vraie". (voir dans [Neb91] et [Joh90]). On note parfois : 2-QBF-$\exists$.

Exemple d'un problème de classe Π_2^p et même Π_2^p-complet : le complémentaire du problème 2-QBF "La formule $\exists a \forall b H(a,b)$ est-elle non satisfiable ?". Ce qui est équivalent au problème "La formule $\forall a \exists b \neg H(a,b)$ est-elle satisfiable ?", qui est parfois noté : 2-QBF-$\forall$.

[7] Même remarque que pour GSAT.

[8] La non satisfiabilité d'une formule est équivalente à la validité de sa négation.

5.4 Représentation de la hiérarchie polynomiale



P est inclus dans NP et dans co-NP eux-mêmes inclus dans Δ_2^p, etc.

On ne sait pas à l'heure actuelle si ces inclusions sont strictes. Si P $=$ NP alors toute la hiérarchie polynomiale s'effondre sur P (voir section suivante).

5.5 Les conjectures de la théorie de la complexité

Il existe tout un ensemble de conjectures en théorie de la complexité, entre autres :

Conjecture 5.5.1 *$P \neq NP$.*

Conjecture 5.5.2 *$NP \neq$ co-NP.*

Conjecture 5.5.3 $\Sigma_k^p \neq \Pi_k^p \ \forall k$.

Ces conjectures ont fait l'objet de recherches intensives de la part des mathématiciens et des théoriciens de l'informatique depuis plusieurs dizaines d'années. Et personne n'a encore réussi à prouver leur véracité ou leur fausseté.

Remarquons qu'il "suffirait" de trouver un algorithme de complexité polynomiale pour un problème connu comme étant NP-complet pour que les classes NP et co-NP soient ramenées à la classe P. On obtiendrait ainsi un "tassement" de la hiérarchie polynomiale.

Ces conjectures sont sous-entendues dans tout ce qui suit.

Chapitre 6

Étude de complexité des différentes relations d'inférence dans le cas général

Il s'agit d'étudier les 21 relations d'inférence $(E, <) \mid\!\sim^{p,m} H$[1] dans lesquelles on a :

- $E =$ la base initiale (ensemble de formules propositionnelles) supposée finie,
- $< =$ ordre entre les formules de E,
- $p =$ principe d'inférence,
- $m =$ mécanisme de définition de sous-ensembles consistants de E avec un ordre entre ces sous-ensembles induit de $<$,
- $H =$ formule de la logique propositionnelle.

Pourquoi se limiter au cas de la logique propositionnelle ? Tout simplement parce que le mécanisme d'inférence classique auquel on va se ramener, qui est décidable en logique des propositions, est indécidable en logique du premier ordre. Ce qui signifie qu'il n'existe pas d'algorithme pouvant résoudre le problème dans le cas du premier ordre sur toutes ses instances. Or les calculs de complexité s'appuient sur l'existence de tels algorithmes de résolution.

Qu'appelle-t-on le cas général ? Il s'agit du cas où $(E, <)$ est un ensemble ordonné de formules propositionnelles quelconques et où H est une formule propositionnelle quelconque.

Nous verrons dans le chapitre 7 trois cas particuliers pour $(E, <)$ et H (une formule par strate, la forme conjonctive normale, les clauses de Horn) qui, pour deux d'entre eux, simplifient considérablement la complexité des problèmes posés.

Remarque : dans tout ce chapitre, ainsi que dans le chapitre 7, nous utiliserons sans les rappeler divers théorèmes et définitions de la complexité qui sont répertoriés dans l'annexe A[2].

6.1 Étude de la complexité de UNI-T

Rappel : Le problème UNI-T est le suivant : "vérifier que H est une conséquence forte de E[3] en utilisant les thèses définies à partir de E".

Cela se note : $E \mid\!\sim^{\forall,T} H$.

[1] Notre notation $(E, <) \mid\!\sim^{p,m} H$ n'est pas tout à fait la notation traditionnelle qui voudrait que E soit une formule. Elle correspond à : TRUE $\mid\!\sim^{p,m}_{E,<} H$. Toutefois, nous conserverons notre notation un peu "spéciale" dans un souci de lisibilité.

[2] À tous ceux qui sont intéressés par le contenu des preuves, nous conseillons une lecture rapide de l'annexe citée, dans laquelle ils trouveront les théorèmes et définitions utilisés, mais surtout les démonstrations de la complexité de quelques processus de révision de croyances. Ce sont les mécanismes mis en œuvre dans ces démonstrations qui nous ont largement inspirés pour la suite.

[3] Ici, l'ordre $<$ ne sert à rien, nous utiliserons donc la notation E au lieu de $(E, <)$. Cette remarque est valable pour UNI (EXI, ARG)-T (S, CAR, E).

$1^{ère}$ **partie : l'appartenance à une classe.** Notons que ce mécanisme d'inférence non monotone correspond exactement au mécanisme de révision SBR étudié par Nebel dans [Neb91] (voir l'annexe A). La démonstration proposée par Nebel s'applique donc parfaitement au mécanisme UNI-T. Nous allons quand même la reprendre en intégralité[4].

Raisonner sur UNI-T nécessite de passer en revue toutes les thèses de E pour savoir si H est conséquence forte. On va donc faire la démonstration plutôt sur le problème co-UNI-T ($E \not\vdash^{\forall,T} H$) car dans ce cas-là il suffira de trouver une seule thèse qui n'infère pas H[5]. Un algorithme pour co-UNI-T est le suivant :

1. deviner un sous-ensemble Y de E
2. vérifier que Y est une thèse de E
3. vérifier que Y n'infère pas H.

Pour résoudre le point 2 de l'algorithme, on peut vérifier la consistance de Y puis prendre chaque élément z de $E \setminus Y$ et vérifier l'inconsistance de $Y \cup \{z\}$.

On obtient ainsi pour co-UNI-T l'algorithme suivant[6] *:*

1. deviner un sous-ensemble Y de E
2. vérifier que Y est consistant
3. pour chaque élément z de $E \setminus Y$ faire
4. vérifier que $Y \cup \{z\}$ est inconsistant
fin pour
5. vérifier que Y n'infère pas H.

Quant à la résolution des points 2, 4 et 5 de l'algorithme, elle peut se faire en utilisant un oracle non déterministe polynomial, puisque le problème "Soit un ensemble de formules BC et une formule H, $BC \not\vdash H$?" est NP et que le problème "Soit un ensemble de formules BC et une formule H, $BC \vdash H$?" est co-NP. L'algorithme donné pour co-UNI-T est donc non déterministe (à cause du "deviner") polynomial et il fait appel à un oracle lui aussi non déterministe polynomial. Ainsi, le problème co-UNI-T appartient à la classe de complexité NP$^{NP} = \Sigma_2^p$. Le problème UNI-T appartient alors à la classe de complexité Π_2^p.

Attention, il s'agit là d'une limite maximale. Peut-être que UNI-T est de complexité plus faible[7]*.*

$2^{ème}$ **partie : la complétude ?** *On espère montrer que UNI-T est Π_2^p-complet, donc que co-UNI-T est Σ_2^p-complet. On connaît des problèmes Σ_2^p-complets, et si on arrive à ramener un de ces problèmes à co-UNI-T par une transformation polynomiale, on aura démontré la complétude*[8]*. Prenons par exemple le problème 2-QBF (voir entre autres [Neb91], [Joh90] et la démonstration de SBR dans l'annexe A). La question est de savoir si on peut passer de 2-QBF à co-UNI-T grâce à une transformation polynomiale.*

Soit "$\exists a \forall b G(a,b)$" une instance de 2-QBF, posons : $E = \{a_1, \ldots, a_n, \neg a_1, \ldots, \neg a_n, \neg G(a,b)\}$, *et* $H = \neg G(a,b)$.

Remarques :

- *les thèses de E sont de la forme :*
  - *soit $\{l_1, \ldots, l_n\}$ avec $l_i = a_i$ ou $\neg a_i$, $\forall i$, et $\{l_1, \ldots, l_n\}$ inconsistant avec $\neg G(a,b)$,*

[4] Pourquoi la reprendre au lieu de se contenter de citer le résultat de Nebel ? Parce que les mécanismes mis en œuvre ici vont être rediscutés lors de l'étude des cas particuliers (voir chapitre 7).

[5] Cette remarque est valable pour toutes les démonstrations portant sur une relation d'inférence utilisant la conséquence forte. Elle ne sera donc pas répétée.

[6] Cette technique de raffinage de l'algorithme initial sera systématiquement utilisée pour toute étude d'appartenance concernant les relations d'inférence utilisant des thèses. Elle ne sera donc pas répétée.

[7] Cette remarque est valable pour toutes les démonstrations d'appartenance à une classe de complexité. Elle ne sera donc pas répétée.

[8] Ce mécanisme de raisonnement est systématiquement utilisé dès que l'on cherche à prouver la complétude.

- *soit* $\{l_1, \ldots, l_n, \neg G(a,b)\}$ *avec* $l_i = a_i$ *ou* $\neg a_i$, $\forall i$, *et* $\{l_1, \ldots, l_n\}$ *consistant avec* $\neg G(a,b)$,

- *on a dans* E *toutes les "valeurs" possibles pour les variables propositionnelles* $a_1, \ldots, a_n$ *dont on cherche s'il existe un modèle,*
- *on n'impose aucune contrainte sur les variables propositionnelles* $b_1, \ldots, b_m$*, dont tous les modèles doivent être pris en compte.*

On a alors :

$$\exists a \forall b G(a,b) \text{ satisfiable}$$

$$\Leftrightarrow$$

il existe une assignation des variables propositionnelles $a_1, \ldots, a_n$ *telle que sans la moindre contrainte sur les variables propositionnelles* $b_1, \ldots, b_m$ *(donc pour toute assignation de* $b_1, \ldots, b_m$*) on a* $G(a,b)$ *vraie*[9]

$$\Leftrightarrow$$

$$\exists \{l_1, \ldots, l_n\} \text{ telle que } \{l_1, \ldots, l_n\} \models G(a,b)$$

$$\Leftrightarrow$$

$$\exists \{l_1, \ldots, l_n\} \text{ telle que } \{l_1, \ldots, l_n\} \text{ inconsistante avec } \neg G(a,b)$$

$$\Leftrightarrow$$

$$\exists \text{ une thèse } \{l_1, \ldots, l_n\} \not\vdash \neg G(a,b)^{10}$$

$$\Leftrightarrow$$

$$E \not\sim^{\forall,T} H$$

On a donc montré que 2-QBF $\propto$ *co-UNI-T. Or 2-QBF est* Σ_2^p*-complet et on a vu que co-UNI-T* $\in \Sigma_2^p$*. On en déduit que co-UNI-T est* Σ_2^p*-complet et donc que UNI-T est* Π_2^p*-complet.*

En conclusion : *UNI-T est* Π_2^p*-complet.*

6.2 Étude de la complexité de EXI-T

Rappel : *Le problème EXI-T est le suivant : "vérifier que* H *est une conséquence faible de* E *en utilisant les thèses définies à partir de* E*".*

Cela se note : $E \mid\sim^{\exists,T} H$.

$1^{ère}$ **partie : l'appartenance à une classe.** *Un algorithme pour EXI-T est le suivant :*

1. *deviner un sous-ensemble* Y *de* E
2. *vérifier que* Y *est une thèse de* E
3. *vérifier que* Y *infère* H.

Cet algorithme est non déterministe (à cause du "deviner") polynomial et il fait appel à un oracle lui aussi non déterministe polynomial. Le problème EXI-T appartient donc à la classe de complexité $NP^{NP} = \Sigma_2^p$.

[9] Cette étape du raisonnement sur 2-QBF permettant le passage de "$\exists a \forall b G(a,b)$ satisfiable" à "$\exists \{l_1, \ldots, l_n\}$ telle que $\{l_1, \ldots, l_n\} \vdash G(a,b)$" est systématique et ne sera donc pas répétée.

[10] En logique propositionnelle, nous avons la propriété suivante : "$\forall A, \forall B$, ensembles de formules propositionnelles $A \models B \Leftrightarrow A \vdash B$. Cette remarque est utilisée dans la plupart des démonstrations, elle ne sera donc pas répétée.

$2^{ème}$ **partie : la complétude ?** *Reprenons le problème 2-QBF et essayons de passer de 2-QBF à EXI-T grâce à une transformation polynomiale.*

Soit "$\exists a \forall b G(a,b)$", une instance de 2-QBF, posons : $H = G(a,b)$ *et* $E = \{a_1, \ldots, a_n, \neg a_1, \ldots, \neg a_n\}$.

Les remarques faites lors de la démonstration de UNI-T sont toujours valables, exception faite de la première qui devient :

- *la base E est égale à $\{a_1, \ldots, a_n, \neg a_1, \ldots, \neg a_n\}$, ce qui signifie que quelle que soit la thèse Y, c'est-à-dire une sous-base maximale consistante, on a $Y = \{l_1, \ldots, l_n\}$ avec $l_i = a_i$ ou $\neg a_i$, $\forall i = 1 \ldots n$.*

Montrons alors que : "$\{a_1, \ldots, a_n, \neg a_1, \ldots, \neg a_n\}$ infère $G(a,b)$ avec la méthode EXI-T" équivaut à "$\exists a \forall b G(a,b)$ est satisfiable" :

$$\{a_1, \ldots, a_n, \neg a_1, \ldots, \neg a_n\} \text{ infère } G(a,b) \text{ avec la méthode EXI-T}$$
$$\Leftrightarrow$$
$$\text{il existe un sous-ensemble } Y \text{ de } E \text{ tel que } Y \text{ soit une thèse et infère } G(a,b)$$
$$\Leftrightarrow$$
$$\text{il existe un sous-ensemble } Y = \{l_1, \ldots, l_n\} \text{ avec } l_i = a_i \text{ ou } \neg a_i, \forall i = 1, \ldots, n \text{ tel que } Y \vdash G(a,b)$$
$$\Leftrightarrow$$
$$\text{il existe } \{l_1, \ldots, l_n\} \text{ qui infère } G(a,b)$$
$$\Leftrightarrow$$
$$\exists a \forall b G(a,b) \text{ est satisfiable}$$

On a donc montré que 2-QBF $\propto$ EXI-T. Or 2-QBF est Σ_2^p-complet et EXI-T $\in \Sigma_2^p$. On en déduit que EXI-T est Σ_2^p-complet[11].

En conclusion : *EXI-T est Σ_2^p-complet.*

6.3 Étude de la complexité de ARG-T

Rappel : *Le problème ARG-T est le suivant : "vérifier que H est une conséquence argumentative de E en utilisant les thèses définies à partir de E".*
Cela se note : $E \mid\!\sim^{A,T} H$.

$1^{ère}$ **partie : l'appartenance à une classe.** *Un algorithme pour ARG-T est le suivant :*

1. *vérifier que $E \not\mid\!\sim^{\exists,T} \neg H$*
2. *vérifier que $E \mid\!\sim^{\exists,T} H$*

Cet algorithme est polynomial et il fait appel à un oracle de complexité Σ_2^p (EXI-T)[12].
La complexité de ARG-T est donc au plus $P^{\Sigma_2^p} = \Delta_3^p$.

[11] Ce résultat n'est pas surprenant. Dans [EG93], Eiter et Gottlob ont défini le problème abductif suivant :

instance : $P = (V, H, M, T)$ un PAP (problème d'abduction propositionnel) avec V un ensemble de variables propositionnelles, H un ensemble d'hypothèses (atomes propositionnels), M un ensemble de manifestations (formules propositionnelles), T une théorie consistante (formules propositionnelles) ;
question : Existe-t-il une explication pour P ? (une explication est un sous-ensemble H' tel que $H' \subseteq H$, $T \cup H'$ est consistant, $(T \cup H') \models M$).

Ce problème peut être transformé en EXI-T de la manière suivante : $E = T \cup H$ et $G = M$.

[12] Voir la démonstration de la complexité du mécanisme de révision FMR (théorème A.2.1 de l'annexe A) pour l'utilisation dans un algorithme de deux oracles, l'un de complexité X et l'autre de complexité co-X.

$2^{ème}$ **partie : la complétude ?** *Nous avons ici des difficultés à étudier la complétude. En effet, on connaît quelques problèmes Δ_3^p-complets, par exemple le problème du double sac à dos de [Joh90], ou quelques problèmes d'abduction de [EG93]. Toutefois, nous n'avons pas réussi à ramener polynomialement un de ces problèmes à ARG-T*[13]*. On ne peut donc pas prouver simplement la complétude, si complétude il y a.*

Par contre, on peut peut-être raffiner comme Nebel l'a fait dans le calcul des complexités de FMR et UBR (voir dans [Neb91] et dans l'annexe A), c'est-à-dire montrer que ARG-T $\in \Delta_3^p$ - ($\Sigma_2^p \cup \Pi_2^p$).

Pour cela on va essayer de comparer le problème ARG-T à des problèmes de classe inférieure dans la hiérarchie polynomiale.

On constate alors que :

- *le problème EXI-T se ramène polynomialement à ARG-T :*

Soit "E, H", une instance de EXI-T, posons : *- $f(E) = E \cup \{H \rightarrow G\}$ avec G nouvelle variable propositionnelle* *(G n'apparaît pas dans E),* *- $f(H) = G$.*

  Remarques préliminaires :

  - *$H \rightarrow G$ est consistante avec n'importe quelle thèse Y' de E : soit Y' thèse $\Rightarrow$ Y' est consistante, donc admet un modèle M'. Or G n'apparaît pas dans E, donc n'apparaît pas dans Y'. On peut ainsi étendre le modèle M' de Y' à un modèle M en rajoutant la valeur VRAI pour la variable propositionnelle G. On a ainsi M modèle de Y' et modèle de $H \rightarrow G$. $H \rightarrow G$ est donc consistante avec toute thèse de E.*
  - *Quelle que soit Y thèse de $f(E)$, Y sera de la forme $Y = Y' \cup \{H \rightarrow G\}$ avec Y' thèse de E. Ceci vient du fait que la formule $H \rightarrow G$ est consistante avec toutes les thèses de E.*
  - *Quelle que soit Y thèse de $f(E)$, $Y \vdash G$ ssi $Y' \vdash H$, car :*
    - *$Y' \vdash H \Rightarrow \forall M'$ modèle de Y', $M'(H) =$ VRAI ; on cherche à montrer que $Y \vdash G$, donc que $\forall M$ modèle de Y, M est un modèle de G ; M étant un modèle de Y et $Y = Y' \cup \{H \rightarrow G\}$, M est donc un modèle de Y' et de $H \rightarrow G$; ainsi, on a $M(H) =$ VRAI et $M(H \rightarrow G) =$ VRAI, donc $M(G) =$ VRAI, donc $Y \vdash G$.*
    - *$Y \vdash G$ et supposons que $Y' \not\vdash H \Rightarrow \exists M'$ modèle de Y' tel que $M'(H) =$ FAUX $\Rightarrow \exists M$ modèle de Y défini par M' étendu avec $M(G) =$ FAUX ce qui contredit l'hypothèse $Y \vdash G$.*

  Montrons alors que : $f(E) \mid\!\sim^{A,T} f(H) \Leftrightarrow E \mid\!\sim^{\exists,T} H$:

$$f(E) \mid\!\sim^{A,T} G$$
$$\Leftrightarrow$$
il existe Y thèse de $f(E)$ telle que $Y \vdash G$
et aucune des autres thèses n'infère $\neg G$.
$$\Leftrightarrow$$
il existe Y thèse de $f(E)$ telle que $Y \vdash G$
(voir remarques préliminaires)
$$\Leftrightarrow$$
il existe Y' thèse de E telle que $Y' \vdash H$
(voir remarques préliminaires)
$$\Leftrightarrow$$
$$E \mid\!\sim^{\exists,T} H$$

- *le problème co-EXI-T se ramène polynomialement à ARG-T :*

Soit "E, H", une instance de co-EXI-T, posons : *- $f(E) = E \cup \{\neg H\}$,* *- $f(H) = \neg H$.*

[13] Le problème du double sac à dos de [Joh90] se ramène sans difficulté majeure à un problème de déduction en logique des prédicats. Malheureusement, il faut que la logique utilisée dans le problème ARG-T étudié ici soit la logique propositionnelle, sinon les problèmes de satisfiabilité deviennent indécidables et donc n'appartiennent plus à NP.

Remarque préliminaire : Soit Y une thèse de $f(E) = E \cup \{\neg H\}$:

- *soit Y contient $\neg H$:*
  - *soit $Y = Y' \cup \{\neg H\}$ avec Y' thèse de E consistante avec $\neg H$ et $\neg H \notin Y'$ donc $Y' \not\vdash H$ et Y infère $\neg H$,*
  - *soit $Y = Y'$ avec Y' thèse de E consistante avec $\neg H$ et $\neg H \in Y'$ donc $Y' \not\vdash H$ et Y infère $\neg H$,*
  - *soit il n'existe pas Y' thèse de E consistante avec $\neg H$ et alors il existe un sous-ensemble S de E consistant, qui peut être égal à l'ensemble vide, tel que $Y = S \cup \{\neg H\}$,*
- *soit Y ne contient pas $\neg H \Rightarrow Y = Y'$ avec Y' thèse de E inconsistante avec $\neg H$ donc inférant H.*

On a alors les propriétés suivantes :

Propriété 6.3.1 *Si toutes les thèses Y' de E sont consistantes avec $\neg H$ (donc n'infèrent pas H) alors toutes les thèses de $f(E)$ sont de la forme $Y = Y' \cup \{\neg H\}$ si $\neg H \notin Y'$ ou $Y = Y'$ si $\neg H \in Y'$ et, bien sûr, elles infèrent $\neg H$ et n'infèrent pas H ;*

Propriété 6.3.2 *S'il existe une thèse Y' de E qui infère H (donc inconsistante avec $\neg H$) alors il existe une thèse Y de $f(E)$ de la forme $Y = Y'$ qui infère H ;*

Propriété 6.3.3 *(contraposée de la propriété 6.3.2) si aucune thèse Y de $f(E)$ n'infère H alors aucune thèse Y' de E n'infère H.*

Montrons alors que : $f(E) \mid\!\sim^{A,T} f(H) \Leftrightarrow E \not\mid\!\sim^{\exists,T} H$.
Tout d'abord :

$$f(E) \mid\!\sim^{A,T} \neg H$$
$$\Leftrightarrow$$
$$\exists Y \text{ thèse de } f(E) \text{ telle que } Y \vdash \neg H$$
$$\text{et } \forall Y \text{ thèse de } f(E)\ Y \not\vdash H$$
$$\Rightarrow$$
$$\text{(à cause de la propriété 6.3.3)}$$
$$\forall Y' \text{ thèse de } E,\ Y' \not\vdash H$$
$$\Leftrightarrow$$
$$E \not\mid\!\sim^{\exists,T} H$$

D'autre part :

$$E \not\mid\!\sim^{\exists,T} H$$
$$\Leftrightarrow$$
$$\forall Y' \text{ thèse de } E,\ Y' \not\vdash H$$
$$\Rightarrow$$
$$\text{(à cause de la propriété 6.3.1)}$$
$$\forall Y \text{ thèse de } f(E),\ Y \not\vdash H \text{ et } Y \vdash \neg H$$
$$\Rightarrow$$
$$\exists Y \text{ thèse de } f(E) \text{ telle que } Y \vdash \neg H$$
$$\text{et } \forall Y \text{ thèse de } f(E)\ Y \not\vdash H$$
$$\Leftrightarrow$$
$$f(E) \mid\!\sim^{A,T} \neg H$$

On a donc EXI-T $\propto$ ARG-T et co-EXI-T $\propto$ ARG-T (avec EXI-T qui est Σ_2^p-complet et co-EXI-T qui est Π_2^p-complet). Donc ARG-T $\in \Delta_3^p$ et est Σ_2^p-difficile et Π_2^p-difficile. En utilisant le théorème A.1.4 donné dans l'annexe A[14]*, nous arrivons à la conclusion suivante.*

[14] Ce théorème A.1.4 sera utilisé de manière systématique dans les démonstrations du type "le problème $\in \Delta_k^p$ - ($\Sigma_{k-1}^p \cup \Pi_{k-1}^p$)". Il ne sera donc pas cité à chaque fois.

En conclusion : *Si $\Sigma_2^p \neq \Pi_2^p$, ARG-T $\in \Delta_3^p$ - ($\Sigma_2^p \cup \Pi_2^p$).*

Nous avons choisi d'exprimer ce résultat sous cette forme afin de bien "visualiser" la place occupée par le problème étudié dans la classe Δ_3^p, plutôt que d'utiliser la forme "ARG-T $\in$ Δ_3^p et est Σ_2^p-difficile et Π_2^p-difficile".

6.4 Étude de la complexité de UNI-S

Rappel : *Le problème UNI-S est le suivant : "vérifier que H est une conséquence forte de E en utilisant les sous-bases consistantes définies à partir de E".*

Cela se note : $E \mid\!\sim^{\forall,S} H$.

$1^{ère}$ **partie : l'appartenance à une classe.** *Un algorithme pour UNI-S est le suivant :*

1. pour tout élément G_i de la base E faire
2. vérifier que $G_i \vdash H$
fin pour

Or, on sait que : ($\forall G_i \in E, i = 1 \ldots n$, $G_i \vdash H$) $\Leftrightarrow$ ($(\bigvee_i G_i) \vdash H$). Donc l'algorithme ci-dessus se simplifie et devient :

1. vérifier que $(\bigvee_i G_i) \vdash H$ avec $i = 1 \ldots n$

Le problème UNI-S se ramène donc exactement au problème de l'inférence classique et est donc co-NP-complet. D'autre part, se posent les problèmes de la prise en compte des tautologies et de l'ensemble vide en tant que sous-bases consistantes. Pour cette raison, cette relation d'inférence non monotone ne présente aucun intérêt, et ne sera donc pas étudiée plus en détail.

En conclusion : *UNI-S est co-NP-complet.*

6.5 Étude de la complexité de EXI-S

Rappel : *Le problème EXI-S est le suivant : "vérifier que H est une conséquence faible de E en utilisant les sous-bases consistantes définies à partir de E".*

Cela se note : $E \mid\!\sim^{\exists,S} H$.

$1^{ère}$ **partie : l'appartenance à une classe.** *Remarquons d'abord que nous avons la propriété suivante :*

Propriété 6.5.1 *$\exists S$ sous-base consistante telle que $S \vdash H \Leftrightarrow \exists T$ thèse telle que $T \vdash H$*

Il suffit effectivement de constater que si S infère H alors n'importe quelle thèse T contenant S[15] inférera aussi H. De même si T infère H alors il existe une sous-base consistante $S = T$ qui infère H.

Cela revient à dire que toute instance positive pour EXI-T est aussi une instance positive pour EXI-S, et vice-versa. Donc EXI-T = EXI-S. Ainsi, EXI-S aura la même complexité que EXI-T.

En conclusion : *EXI-S est Σ_2^p-complet.*

[15] Et il en existera forcément une !

6.6 Étude de la complexité de ARG-S

Rappel : *Le problème ARG-S est le suivant : "vérifier que H est une conséquence argumentative de E en utilisant les sous-bases consistantes définies à partir de E".*

Cela se note : $E \mid\!\sim^{A,S} H$.

$1^{ère}$ **partie : l'appartenance à une classe.** *Nous avons toujours la propriété 6.5.1 déjà vue pour la démonstration de EXI-S.*

Or l'algorithme que nous proposons pour le problème ARG-S utilise les oracles EXI-S et co-EXI-S :

1. *vérifier que* $E \not\mid\!\sim^{\exists,S} \neg H$
2. *vérifier que* $E \mid\!\sim^{\exists,S} H$

Ce qui est donc équivalent à

1. *vérifier que* $E \not\mid\!\sim^{\exists,T} \neg H$
2. *vérifier que* $E \mid\!\sim^{\exists,T} H$

Donc ARG-T = ARG-S. Ainsi, ARG-S aura la même complexité que ARG-T.

En conclusion : *Si* $\Sigma_2^p \neq \Pi_2^p$, *ARG-S* $\in \Delta_3^p$ - $(\Sigma_2^p \cup \Pi_2^p)$.

6.7 Étude de la complexité de UNI-BO

Rappel : *Le problème UNI-BO est le suivant : "vérifier que H est une conséquence forte avec préférences de $(E,<)$[16] en utilisant l'ordre "BEST-OUT" sur les sous-bases consistantes".*

Cela se note : $(E,<) \mid\!\sim^{\forall,Bo} H$.

$1^{ère}$ **partie : l'appartenance à une classe.** *Il faut d'abord remarquer, vu le mécanisme de construction des sous-bases consistantes préférées pour l'ordre "BEST-OUT", que :*

- *les sous-bases consistantes préférées pour l'ordre "BEST-OUT" sont les sous-ensembles Y qui ont tous le même $a(Y) = amax$ et tels que $\cap Y = \cup_{i=1\ldots amax-1} E_i$ (cette intersection étant consistante),*
- *pour que toutes les sous-bases consistantes préférées pour l'ordre "BEST-OUT" infèrent une même formule, il faut que cette formule soit inférée par leur intersection :* $\forall Y_i,\ Y_i \vdash H \Leftrightarrow \cap Y_i \vdash H$.

Un algorithme pour UNI-BO est le suivant :

1. *calculer le amax de* $(E,<)$
2. *posons* $Y = \cup_{i=1\ldots amax-1} E_i$ *(*Y est consistant*)*
3. *vérifier que Y infère H.*

Décomposons l'étape 1 : calculer le amax de $(E,<)$ *:*

[16] À partir de maintenant, l'ordre $<$ est important ! On utilisera donc la notation $(E,<)$ au lieu de E. Cette remarque est valable pour UNI(EXI, ARG)-BO(INCL, LEX).

1. $nb \leftarrow 1$
2. $Z \leftarrow E_1$
3. tant que Z *est consistant et* $nb \leq$ *nombre total de strates dans* E *faire*
4. $\quad nb \leftarrow nb + 1$
5. $\quad Z \leftarrow Z \cup E_{nb}$
fin tant que
6. $amax \leftarrow nb$

L'algorithme complet est donc polynomial et il fait appel à un oracle non déterministe polynomial. Le problème UNI-BO appartient donc à la classe de complexité $P^{NP} = \Delta_2^p$.

$2^{ème}$ **partie : la complétude ?** *On ne connaît pas de problème* Δ_2^p*-complet. Essayons de comparer le problème UNI-BO avec des problèmes de classe inférieure dans la hiérarchie polynomiale, c'est-à-dire des problèmes NP et co-NP, par exemple les problèmes GSAT et UNGSAT. On constate alors que :*

- *le problème GSAT se ramène au problème UNI-BO par une transformation polynomiale :*

Soit "G*" une instance de GSAT, posons :* $E = \{G\}$ *et* $H = G$.

  Remarquons que $(E, <)$ *est réduit à une seule strate, il est donc noté* E[17]*. Nous avons deux cas possibles :*
  - *si* G *est consistante (donc satisfiable) alors* $amax =$ *2 et la seule sous-base consistante préférée pour l'ordre "BEST-OUT" est* $Y = \{G\}$ *; le problème UNI-BO se ramène donc à montrer que* $G \vdash G$*, ce qui est toujours vrai ;*
  - *sinon (*G *est inconsistante donc non satisfiable)* $amax = 1$ *et la seule sous-base consistante préférée pour l'ordre "BEST-OUT" est* $Y = \varnothing$ *; le problème UNI-BO se ramène donc à montrer que* $\vdash G$*, ce qui est faux quand* G *est non satisfiable.*

  Ainsi on aura bien : G *est satisfiable* $\Leftrightarrow E \mid\!\sim^{\forall,Bo} H$.
  En conclusion, GSAT se transforme polynomialement en UNI-BO (noté GSAT $\propto$ *UNI-BO).*
- *le problème UNGSAT se ramène au problème UNI-BO de manière polynomiale :*

Soit "G*" une instance de UNGSAT, posons :* $E = \varnothing$ *et* $H = \neg G$.

  Remarquons alors que $amax = 1$ *et que la seule sous-base consistante préférée pour l'ordre "BEST-OUT" est* $Y = \varnothing$ *; le problème UNI-BO se ramène donc à montrer que* $\vdash \neg G$*, ce qui est faux quand* G *est satisfiable et vrai quand* G *est non satisfiable.*
  On a donc : UNGSAT $\Leftrightarrow G$ *non satisfiable* $\Leftrightarrow E \mid\!\sim^{\forall,Bo} H$.
  En conclusion, on a UNGSAT $\propto$ *UNI-BO.*

En faisant le même raisonnement que celui pour ARG-T, on arrive à la conclusion suivante.

En conclusion : | *Si NP* $\neq$ *co-NP, UNI-BO* $\in \Delta_2^p$ *- (NP* $\cup$ *co-NP).* |

6.8 Étude de la complexité de EXI-BO

Rappel : *Le problème EXI-BO est le suivant : "vérifier que* H *est une conséquence faible avec préférences de* $(E, <)$ *en utilisant l'ordre "BEST-OUT" sur les sous-bases consistantes".*

Cela se note : $(E, <) \mid\!\sim^{\exists,Bo} H$.

[17] Ce cas se produit souvent. Cette remarque ne sera donc pas répétée.

$1^{ère}$ **partie : l'appartenance à une classe.** *Un algorithme pour EXI-BO est le suivant :*

1. deviner un sous-ensemble Y de $(E, <)$
2. vérifier que Y est une sous-base consistante préférée pour l'ordre "BEST-OUT" de $(E, <)$
3. vérifier que Y infère H.

Décomposons l'étape 2 : vérifier que Y est une sous-base consistante préférée pour l'ordre "BEST-OUT" de $(E, <)$:

1. vérifier que Y est consistant
2. calculer $a(Y)$
3. $nb \leftarrow 1$
4. $Z \leftarrow E_1$
*5. tant que Z est consistant et $nb \leq$ nombre total de strates dans E faire (*calcul de $amax$*)*
6. $nb \leftarrow nb + 1$
7. $Z \leftarrow Z \cup E_{nb}$
fin tant que
8. $amax \leftarrow nb$
9. vérifier que $a(Y) = amax$

L'algorithme complet est non déterministe (à cause du "deviner") polynomial et il fait appel à un oracle lui aussi non déterministe polynomial. Le problème EXI-BO appartient donc à la classe de complexité $NP^{NP} = \Sigma_2^p$.

$2^{ème}$ **partie : la complétude ?** *On connaît des problèmes Σ_2^p-complets. La question est donc de savoir si on peut passer de l'un de ces problèmes à EXI-BO grâce à une transformation polynomiale. Choisissons par exemple le problème 2-QBF.*

Soit "$\exists a \forall b G(a, b)$", une instance de 2-QBF, posons : $H = G(a, b)$, $E = \{a_1, \ldots, a_n, \neg a_1, \ldots, \neg a_n\}$.

Ainsi :

- *on a dans E toutes les "valeurs" possibles pour les variables propositionnelles $a_1, \ldots, a_n$ dont on cherche s'il existe une assignation,*
- *on n'impose aucune contrainte sur les variables propositionnelles $b_1, \ldots, b_m$, dont toutes les assignations doivent être prises en compte.*

Remarques préliminaires :

- *La base E est égale à $\{a_1, \ldots, a_n, \neg a_1, \ldots, \neg a_n\}$ avec une seule strate, ce qui signifie que quelle que soit Y sous-base consistante, $a(Y) = 1$ donc Y est préférée pour l'ordre "BEST-OUT" et Y est un sous-ensemble de $\{l_1, \ldots, l_n\}$ avec $l_i = a_i$ ou $\neg a_i$, $\forall i = 1 \ldots n$.*
- *Soit une formule H, s'il existe une sous-base consistante Y telle que Y infère H, alors pour toute $Y' = Y \cup \{lk_1, \ldots, lk_m\}$ avec les lk_j représentant les variables propositionnelles a_k non affectées par Y, Y' sera consistant et inférera H puisque la relation d'inférence classique est monotone.*
- *Soit une formule H, s'il existe un ensemble $Y' = \{l_1, \ldots, l_n\}$ avec $l_i = a_i$ ou $\neg a_i$, $\forall i = 1 \ldots n$ tel que Y' infère H, alors Y' est une sous-base préférée pour l'ordre "BEST-OUT".*

Montrons alors que : "$\{a_1, \ldots, a_n, \neg a_1, \ldots, \neg a_n\}$ infère $G(a, b)$ avec la méthode EXI-BO" équivaut à "$\exists a \forall b G(a, b)$ est satisfiable" :

$\{a_1, \ldots, a_n, \neg a_1, \ldots, \neg a_n\}$ *infère* $G(a, b)$ *avec la méthode EXI-BO*

$\Leftrightarrow$

il existe un sous-ensemble Y *de* E *tel que* Y *soit une sous-base consistante préférée pour l'ordre "BEST-OUT" et infère* $G(a, b)$

$\Leftrightarrow$

il existe une sous-base consistante Y*, sous-ensemble de* $\{l_1, \ldots, l_n\}$ *avec* $l_i = a_i$ *ou* $\neg a_i$*,* $\forall i = 1 \ldots n$*, telle que* $Y \vdash G(a, b)$ *et* $a(Y) = 1$

$\Leftrightarrow$

il existe $\{l_1, \ldots, l_n\}$ *qui infère* $G(a, b)$

$\Leftrightarrow$

$\exists a \forall b G(a, b)$ *est satisfiable*

On a donc montré que 2-QBF $\propto$ *EXI-BO. Or 2-QBF est* Σ_2^p*-complet et EXI-BO* $\in \Sigma_2^p$*. On en déduit que EXI-BO est* Σ_2^p*-complet.*

En conclusion : *EXI-BO est* Σ_2^p*-complet.*

6.9 Étude de la complexité de ARG-BO

Rappel : *Le problème ARG-BO est le suivant : "vérifier que* H *est une conséquence argumentative avec préférences de* $(E, <)$ *en utilisant l'ordre "BEST-OUT" sur les sous-bases consistantes".*

Cela se note : $(E, <) \mid\!\sim^{A,Bo} H$.

$1^{ère}$ **partie : l'appartenance à une classe.** *Un algorithme pour ARG-BO est le suivant :*

1. *vérifier que* $(E, <) \not\mid\!\sim^{\exists,Bo} \neg H$
2. *vérifier que* $(E, <) \mid\!\sim^{\exists,Bo} H$

Cet algorithme est polynomial et il fait appel à un oracle de complexité Σ_2^p *(EXI-BO).*

La complexité de ARG-BO est donc au plus $P^{\Sigma_2^p} = \Delta_3^p$.

$2^{ème}$ **partie : la complétude ?** *Essayons de comparer le problème ARG-BO avec des problèmes de classe inférieure dans la hiérarchie polynomiale.*

On constate alors que :

- *le problème EXI-BO se ramène polynomialement à ARG-BO :*

> *Soit "*$(E, <), H$*", une instance de EXI-BO, posons :*
> - $f(H) = G$,
> - $f((E, <)) = (E \cup \{H \rightarrow G\}, <)^a$
>
> *avec* G *nouvelle variable propositionnelle (*G *n'apparaît pas dans* $(E, <)$*) et la formule* $H \rightarrow G$ *placée seule en première strate.*

[a] En fait, il ne s'agit pas du pré-ordre $<$ (défini uniquement sur E) mais du pré-ordre $<'$ correspondant à l'extension de $<$ en rajoutant $\Phi < (H \rightarrow G), \forall \Phi \in E$. Toutefois, afin de ne pas alourdir les notations, nous continuerons à utiliser $<$ au lieu de $<'$. Cette remarque est valable pour tous les cas où l'on modifie la base E initiale stratifiée en rajoutant un nouvel élément.

Remarques préliminaires :

- $H \rightarrow G$ *est consistante avec n'importe quelle sous-base consistante préférée pour l'ordre "BEST-OUT"* Y' *de* E *: soit* Y' *sous-base consistante préférée pour l'ordre "BEST-OUT"* $\Rightarrow Y'$ *est consistante, donc admet un modèle* M'*. Or* G *n'apparaît pas dans* E*, donc n'apparaît pas dans* Y'*. On peut ainsi étendre le modèle* M' *de* Y' *à un modèle* M *en rajoutant la valeur VRAI pour la variable propositionnelle* G*. On a ainsi* M *modèle de* Y' *et modèle de* $H \rightarrow G$*.* $H \rightarrow G$ *est donc consistante avec toute sous-base consistante préférée pour l'ordre "BEST-OUT" de* E*.*
- *Quelle que soit* Y *sous-base consistante préférée pour l'ordre "BEST-OUT" de* $f((E, <))$*,* Y *sera de la forme* $Y = Y' \cup \{H \rightarrow G\}$ *avec* Y' *sous-base consistante préférée pour l'ordre "BEST-OUT" de* $(E, <)$*. Ceci vient du fait que la formule* $H \rightarrow G$ *est la plus prioritaire de* $f((E, <))$ *et qu'elle est consistante avec toutes les sous-bases consistantes préférées pour l'ordre "BEST-OUT" de* $(E, <)$*.*
- *Quelle que soit* Y *sous-base consistante préférée pour l'ordre "BEST-OUT" de* $f((E, <))$*,* $Y \vdash G$ *ssi* $Y' \vdash H$ *(voir argumentation dans la démonstration de ARG-T).*

Montrons alors que : "$f((E, <))$ *infère* $f(H)$ *avec la méthode ARG-BO" équivaut à "*$(E, <)$ *infère* H *avec la méthode EXI-BO" :*

$f((E, <))$ *infère* G *avec la méthode ARG-BO*
$\Leftrightarrow$
il existe Y *sous-base consistante préférée pour*
l'ordre "BEST-OUT" de $f((E, <))$ *telle que* $Y \vdash G$
et aucune des autres sous-bases consistantes préférées pour
l'ordre "BEST-OUT" n'infère $\neg G$*.*
$\Leftrightarrow$
il existe Y *sous-base consistante préférée pour l'ordre*
"BEST-OUT" de $f((E, <))$ *telle que* $Y \vdash G$ *(voir remarques préliminaires)*
$\Leftrightarrow$
il existe Y' *sous-base consistante préférée pour l'ordre*
"BEST-OUT" de $(E, <)$ *telle que* $Y' \vdash H$ *(voir remarques préliminaires)*
$\Leftrightarrow$
$(E, <)$ *infère* H *avec la méthode EXI-BO*

- *le problème co-EXI-BO se ramène polynomialement à ARG-BO :*

Soit "$(E, <), H$*", une instance de co-EXI-BO, posons :* - $f(H) = \neg H$*,* - $f((E, <)) = (E \cup \{\neg H\}, <)$ *avec* $\neg H$ *constituant seule la dernière strate.*

Remarques préliminaires :
Soit Y *une sous-base consistante préférée pour l'ordre "BEST-OUT" de* $f((E, <))$*, on a 2 cas possibles :*

- *soit* Y *ne contient pas* $\neg H$ *et alors* Y *est une sous-base consistante préférée pour l'ordre "BEST-OUT" de* $(E, <)$ *qui peut être soit consistante, soit inconsistante avec* $\neg H$*,*
- *soit* Y *contient* $\neg H$ *et :*
  - *soit* $Y = Y' \cup \{\neg H\}$ *où* Y' *est une sous-base consistante préférée pour l'ordre "BEST-OUT" de* $(E, <)$ *qui est consistante avec* $\neg H$ *et qui ne contient pas* $\neg H$*,*
  - *soit* $Y = Y'$ *où* Y' *est une sous-base consistante préférée pour l'ordre "BEST-OUT" de* $(E, <)$ *qui contient* $\neg H$*.*

On constate alors que, si aucune des sous-bases consistantes préférées pour l'ordre "BEST-OUT" de $(E, <)$ *n'infère* H*, alors aucune des sous-bases consistantes préférées pour l'ordre "BEST-OUT" de* $f((E, <))$ *n'inférera* H*, et réciproquement. D'autre part, s'il existe au moins une sous-base consistante préférée pour l'ordre "BEST-OUT" de* $(E, <)$ *consistante avec* $\neg H$*, alors on aura au moins une sous-base consistante préférée pour l'ordre "BEST-OUT" de* $f((E, <))$ *qui contiendra* $\neg H$ *et donc inférera* $\neg H$*. On a donc :*

$$(E, <) \not\sim^{\exists, Bo} H$$
$$\Leftrightarrow$$
il n'existe pas de sous-base consistante Y' préférée pour l'ordre "BEST-OUT" de $(E, <)$ qui infère H (donc toutes les Y' sont consistantes avec $\neg H$)
$$\Leftrightarrow$$
il n'existe pas de sous-base consistante Y préférée pour l'ordre "BEST-OUT" de $f((E, <))$ qui infère H et il existe au moins une sous-base Y préférée pour l'ordre "BEST-OUT" de $f((E, <))$ qui contient $\neg H$ et donc infère $\neg H$
$$\Leftrightarrow$$
$$f((E, <)) \not\sim^{A, Bo} \neg H$$

On a donc EXI-BO $\propto$ ARG-BO et co-EXI-BO $\propto$ ARG-BO (avec EXI-BO qui est Σ_2^p-complet et co-EXI-BO qui est Π_2^p-complet). Donc, comme pour ARG-T, on arrive à la conclusion suivante.

En conclusion : *Si $\Sigma_2^p \neq \Pi_2^p$, ARG-BO $\in \Delta_3^p$ - ($\Sigma_2^p \cup \Pi_2^p$).*

6.10 Étude de la complexité de UNI-INCL

Rappel : *Le problème UNI-INCL est le suivant : "vérifier que H est une conséquence forte avec préférences de $(E, <)$ en utilisant l'ordre "INCLUSION BASED" sur les thèses de E".*

Cela se note : $(E, <) \sim^{\forall, In} H$.

$1^{ère}$ **partie : l'appartenance à une classe.** *Notons que ce mécanisme d'inférence non monotone correspond exactement au mécanisme de révision PBR étudié par Nebel dans [Neb91] (voir l'annexe A). Comme pour UNI-T, nous constatons que la démonstration proposée par Nebel s'applique parfaitement au mécanisme UNI-INCL. Nous allons donc la reprendre en intégralité en utilisant UNI-T à la place de SBR.*

Un algorithme pour co-UNI-INCL est le suivant :

1. *deviner un sous-ensemble Y de $(E, <)$*
2. *vérifier que Y est une thèse préférée pour l'ordre "INCLUSION BASED" de $(E, <)$*
3. *vérifier que Y n'infère pas H.*

Cet algorithme est non déterministe polynomial (à cause du "deviner") et il fait appel à un oracle lui aussi non déterministe polynomial. Le problème co-UNI-INCL appartient à la classe de complexité $NP^{NP} = \Sigma_2^p$.

Le problème UNI-INCL appartient alors à la classe de complexité Π_2^p.

$2^{ème}$ **partie : la complétude ?** *On connaît des problèmes Π_2^p-complets. Par exemple, le problème UNI-T. Or, il se trouve que UNI-INCL est une généralisation de UNI-T.*

Soit "E, H", une instance de UNI-T, considérons toutes les formules de E en une seule strate.

Remarquons alors que nous avons la propriété suivante :

Propriété 6.10.1 *Quand E est constitué d'une seule strate, les thèses de E sont exactement les thèses préférées pour l'ordre "INCLUSION BASED".*

On a alors :

$$E \mid\!\sim^{\forall,T} H$$

$$\Leftrightarrow$$

Quelle que soit Y thèse de E, Y infère H

$$\Leftrightarrow$$

Quelle que soit Y thèse préférée pour l'ordre "INCLUSION BASED" de E, Y infère H

$$\Leftrightarrow$$

$$E \mid\!\sim^{\forall,In} H$$

On a donc montré que UNI-T $\propto$ UNI-INCL. Or UNI-T est Π_2^p-complet et UNI-INCL $\in \Pi_2^p$. On en déduit que UNI-INCL est Π_2^p-complet.

En conclusion : *UNI-INCL est Π_2^p-complet.*

6.11 Étude de la complexité de EXI-INCL

Rappel : *Le problème EXI-INCL est le suivant : "vérifier que H est une conséquence faible avec préférences de $(E, <)$ en utilisant l'ordre "INCLUSION BASED" sur les thèses de E".*
Cela se note : $(E, <) \mid\!\sim^{\exists,In} H$.

$1^{ère}$ **partie : l'appartenance à une classe.** *Un algorithme pour EXI-INCL est le suivant :*

1. *deviner un sous-ensemble Y de $(E, <)$*
2. *vérifier que Y est une thèse préférée pour l'ordre "INCLUSION BASED" de $(E, <)$*
3. *vérifier que Y infère H.*

Cet algorithme est non déterministe polynomial (à cause du "deviner") et il fait appel à un oracle lui aussi non déterministe polynomial. Le problème EXI-INCL appartient donc à la classe de complexité $NP^{NP} = \Sigma_2^p$.

$2^{ème}$ **partie : la complétude ?** *De la même manière et pour les mêmes raisons que pour UNI-INCL, essayons de passer de EXI-T à EXI-INCL grâce à une transformation polynomiale.*

Soit "E, H", une instance de EXI-T, considérons toutes les formules de E en une seule strate.

Nous avons toujours la propriété 6.10.1. On a alors :

$$E \mid\!\sim^{\exists,T} H$$

$$\Leftrightarrow$$

il existe Y thèse de E qui infère H

$$\Leftrightarrow$$

il existe Y thèse préférée pour l'ordre "INCLUSION BASED" de E qui infère H

$$\Leftrightarrow$$

$$E \mid\!\sim^{\exists,In} H$$

On a donc montré que EXI-T $\propto$ EXI-INCL. Or EXI-T est Σ_2^p-complet et EXI-INCL $\in \Sigma_2^p$. On en déduit que EXI-INCL est Σ_2^p-complet.

En conclusion : *EXI-INCL est Σ_2^p-complet.*

6.12 Étude de la complexité de ARG-INCL

Rappel : *Le problème ARG-INCL est le suivant : "vérifier que H est une conséquence argumentative avec préférences de $(E, <)$ en utilisant l'ordre "INCLUSION BASED" sur les thèses de E".*
Cela se note : $(E, <) \mid\!\sim^{A,In} H$.

$1^{ère}$ **partie : l'appartenance à une classe.** *Un algorithme pour ARG-INCL est le suivant :*

1. *vérifier que $(E, <) \not\mid\!\sim^{\exists,In} \neg H$*
2. *vérifier que $(E, <) \mid\!\sim^{\exists,In} H$*

Cet algorithme est polynomial et il fait appel à un oracle de complexité Σ_2^p (EXI-INCL).
La complexité de ARG-INCL est donc au plus $P^{\Sigma_2^p} = \Delta_3^p$.

$2^{ème}$ **partie : la complétude ?** *Nous allons comparer le problème ARG-INCL avec des problèmes de classe inférieure dans la hiérarchie polynomiale.*
Pour cela on constate d'abord que ARG-T se ramène polynomialement à ARG-INCL. En effet, nous avons toujours la propriété 6.10.1.

Soit "E, H", une instance de ARG-T, considérons toutes les formules de E en une seule strate.

On a alors :

$$E \mid\!\sim^{A,T} H$$
$$\Leftrightarrow$$
il existe Y thèse de E qui infère H et aucune thèse de E n'infère $\neg H$
$$\Leftrightarrow$$
il existe Y thèse préférée pour l'ordre "INCLUSION BASED" de E qui infère H et aucune des thèses préférées pour l'ordre "INCLUSION BASED" de E n'infère $\neg H$
$$\Leftrightarrow$$
$$E \mid\!\sim^{A,In} H$$

On a donc bien ARG-T $\propto$ ARG-INCL. Or on a démontré que EXI-T $\propto$ ARG-T et que co-EXI-T $\propto$ ARG-T, donc par la transitivité de $\propto$, on a EXI-T $\propto$ ARG-INCL et co-EXI-T $\propto$ ARG-INCL (avec EXI-T qui est Σ_2^p-complet et co-EXI-T qui est Π_2^p-complet). Comme pour ARG-T, on arrive à la conclusion suivante.

En conclusion : *Si $\Sigma_2^p \neq \Pi_2^p$, ARG-INCL $\in \Delta_3^p$ - ($\Sigma_2^p \cup \Pi_2^p$).*

6.13 Étude de la complexité de UNI-CAR

Rappel : *Le problème UNI-CAR est le suivant : "vérifier que H est une conséquence forte avec préférences de E en utilisant un ordre basé sur la cardinalité sur les thèses de E".*
Cela se note : $E \mid\!\sim^{\forall,Car} H$.

$1^{ère}$ **partie : l'appartenance à une classe.** *Remarque : d'après la définition donnée à la section 2.2, on a la propriété suivante :*

Propriété 6.13.1 *Soit E un ensemble ordonné de formules, toutes les thèses préférées pour la cardinalité dans E ont le même cardinal.*

Cette remarque permet de proposer l'algorithme suivant pour UNI-CAR :

1. $k \leftarrow 0$
2. $nf \leftarrow$ nombre de formules dans E
3. $fini \leftarrow$ faux
4. tant que ($nf \geq 0$) et (non $fini$) faire
5. $k \leftarrow nf$
6. si MAX-GSAT (E,k) alors
7. $fini \leftarrow$ vrai
8. sinon $nf \leftarrow nf - 1$
fin si
fin tant que
9. si NGSAT-CAR (E,H,k) alors
10. échec $\leftarrow$ vrai
11. sinon échec $\leftarrow$ faux
fin si
12. vérifier que l'on n'a pas échec (échec = faux)

Cet algorithme est le résultat de nos recherches sur UNI-LEX dont l'étude a été effectuée avant celle de UNI-CAR (voir la section 6.16) et de suggestions proposées par Daniel Lehmann lors d'une communication personnelle.

On y utilise deux oracles distincts : MAX-GSAT et NGSAT-CAR dont il nous faut déterminer la complexité.

- *Le problème MAX-GSAT est défini par :*

  instance : *E une base et k un entier,*
  question : *existe-t-il une interprétation satisfaisant au moins k formules de E ?*

  Nous proposons l'algorithme suivant pour le résoudre :

  1. deviner une interprétation M
  2. $p \leftarrow 0$
  3. pour chaque formule G de E faire
  4. si M satisfait G alors $p \leftarrow p + 1$
  fin si
  fin pour
  5. vérifier que $p \geq k$

  Il s'agit d'un algorithme non déterministe, donc MAX-GSAT appartient à la classe NP. Montrons maintenant la complétude. C'est immédiat. En effet, il suffit de constater que GSAT se ramène polynomialement à MAX-GSAT.

Soit "G" une instance de GSAT, posons : $E = \{G\}$, $k = 1$.

  On a alors : G est satisfiable $\Leftrightarrow$ il existe un modèle M de G $\Leftrightarrow$ il existe une interprétation M telle que M satisfait $k = 1$ formule de E $\Leftrightarrow$ MAX-GSAT ($\{G\}$, 1).
  On a donc MAX-GSAT NP-complet (remarque : MAX-GSAT reste NP-complet, même pour des clauses)
- *Le problème NGSAT-CAR est défini par :*

instance : *E une base, H une formule propositionnelle, k un entier avec $k =$ nombre maximum de formules pouvant être satisfaites dans E,*
question : *existe-t-il une interprétation M qui satisfait k formules de E telle que M ne satisfait pas H ?*

Un algorithme pour NGSAT-CAR est le suivant :

1. deviner une interprétation M
2. vérifier qu'elle satisfait k formules de E
(c'est à dire :*
a. $k' \leftarrow 0$
b. *pour chaque formule G de E faire*
c. *si M satisfait G alors*
d. $k' \leftarrow k' + 1$
fin si
fin pour
e. *vérifier que $k' = k$*
**)*
3. vérifier que M ne satisfait pas H

Remarquons que, quelle que soit l'interprétation M, vérifier que M satisfait ou pas un ensemble de formules propositionnelles se fait en temps polynomial. L'algorithme pour NGSAT-CAR est non déterministe polynomial, donc NGSAT-CAR appartient à la classe de complexité NP. Quant à la preuve de complétude, elle se fait en ramenant polynomialement GSAT à NGSAT-CAR.

Soit "G" une instance de GSAT, posons : $E = \{G\}$, $H = \neg G$ *et* $k = 1$.

On a alors : G satisfiable $\Leftrightarrow$ il existe une interprétation M qui satisfait $G \Leftrightarrow$ Dans E, le nombre maximal de formules pouvant être satisfaites est 1 $\Leftrightarrow$ il existe une interprétation M satisfaisant k formules qui ne satisfait pas $\neg G$. NGSAT-CAR est donc NP-complet (y compris pour des clauses).

Ce qui nous permet de dire que l'algorithme proposé pour UNI-CAR est déterministe polynomial et qu'il fait appel à un oracle non déterministe polynomial[18]*. Le problème UNI-CAR appartient donc à la classe de complexité $P^{NP} = \Delta_2^p$.*

$2^{ème}$ **partie : la complétude ?** *Dans l'article [EG93] de Eiter et Gottlob, plusieurs problèmes Δ_2^p-complets sont cités. Parmi ceux-ci, nous avons choisi d'utiliser, pour prouver la complétude de UNI-CAR, le problème suivant :*

Instance : *Soit $C = \{C_1, \ldots, C_m\}$ ensemble de clauses, soit $k \in \{1, \ldots, m\}$ un entier,*
Question : *toutes les assignations card-maximales V_c des variables de C vérifient-elles : $V_c(C_k) = vrai$?*

Nous notons ce problème ACM (pour Assignation de Cardinalité Maximum) et rappelons la définition d'une assignation card-maximale :

Définition 6.13.1 *Soit un ensemble de clauses $C = \{C_1, \ldots, C_m\}$ et $X = \{x_1, \ldots, x_n\}$ l'ensemble des variables de C, soient deux assignations V et W de X, on dit que $V >^{card} W$ ssi V satisfait strictement plus de clauses de C que W (en nombre de clauses).*

Définition 6.13.2 *V est une assignation card-maximale ssi V est maximale pour l'ordre $>^{card}$.*

[18] En fait, cet algorithme fait appel à deux oracles $O1$ (MAX-GSAT) et $O2$ (NGSAT-CAR). Mais ces 2 oracles sont NP-complets. Il est donc possible de transformer de manière polynomiale un des oracles en l'autre et ainsi de ne plus faire appel qu'à un seul oracle. Cette remarque est valable pour plusieurs problèmes et ne sera donc pas répétée.

Remarque préliminaire : Soit $C = \{C_1, \ldots, C_m\}$ ensemble de clauses. Soit V une assignation des variables de C, posons $Y(V) = \{C_i$ tel que $V(C_i) =$ vrai , $i = 1 \ldots m\}$. On a alors les propriétés suivantes :

Propriété 6.13.2 *$Y(V)$ est un sous-ensemble de C consistant.*

> ***Preuve :*** *D'après la définition de $Y(V)$, l'assignation V satisfait toutes les clauses de $Y(V)$, donc $Y(V)$ est consistant.*

Propriété 6.13.3 *Pour tout sous-ensemble Y de C maximal (pour l'inclusion) consistant, il existe une assignation V des variables de C telle que $Y = Y(V)$.*

> ***Preuve :*** *Y est consistant donc il existe une assignation V telle que V satisfait toutes les clauses de Y. On peut donc définir $Y(V) = \{C_i$ tel que $V(C_i) =$ vrai , $i = 1 \ldots m\}$. On a alors $Y \subseteq Y(V)$. D'autre part, Y est maximal pour l'inclusion donc il n'existe pas de Y' sous-ensemble de C différent de Y tel que $Y \subseteq Y'$. Donc $Y = Y(V)$.*

Propriété 6.13.4 *Soient V et W assignations des variables de C. $V >^{card} W$ ssi $Y(V) \gg^{card} Y(W)$ (l'ordre $\gg^{card}$ est l'ordre basé sur la cardinalité défini par [BCD+ 93] dans le cas d'une base non stratifiée).*

> ***Preuve :*** *$V >^{card} W \Leftrightarrow V$ satisfait plus de clauses de C que $W \Leftrightarrow |Y(V)| > |Y(W)| \Leftrightarrow$ par définition $Y(V) \gg^{card} Y(W)$*

Propriété 6.13.5 *Soit $C = \{C_1, \ldots, C_m\}$ un ensemble de clauses, V est une assignation card-maximale ssi $Y(V)$ est un sous-ensemble consistant de $\{C_1, \ldots, C_m\}$ maximal pour la cardinalité.*

> ***Preuve :*** *V est une assignation card-maximale $\Leftrightarrow$ il n'existe pas d'assignation W de C telle que $V <^{card} W \Leftrightarrow$ il n'existe pas d'assignation W de C, $Y(V) \ll^{card} Y(W) \Leftrightarrow Y(V)$ est un sous-ensemble consistant de $\{C_1, \ldots, C_m\}$ maximal pour la cardinalité.*

Essayons maintenant d'utiliser ce problème ACM pour prouver la complétude de UNI-CAR.

Soit "$C = \{C_1, \ldots, C_m\}$, $k \in \{1, \ldots, m\}$" une instance de ACM, posons : $H = C_k$ *et* $E = \{C_1, \ldots, C_m\}$

Démontrons alors que ACM $\propto$ UNI-CAR :

- *ACM implique UNI-CAR :*

  Toute assignation V_c card-maximale pour C vérifie : $V_c(C_k) =$ vrai
  $\Rightarrow$ *(à cause de la propriété 6.13.3)*
  $\forall Y$ *thèse préférée pour la cardinalité de E, il existe une assignation V telle que $Y = Y(V)$*
  $\Rightarrow$ *(à cause de la propriété 6.13.5)*
  $\forall Y$ *thèse préférée pour la cardinalité de E, l'assignation V telle que $Y = Y(V)$ est card-maximale et vérifie donc $V_c(C_k) =$ vrai*
  $\Rightarrow$
  $\forall Y$ *thèse préférée pour la cardinalité de E, Y contient C_k*
  $\Leftrightarrow$
  $\forall Y$ *thèse préférée pour la cardinalité de E, Y infère C_k*

- *UNI-CAR implique ACM :*

$\forall Y$ *thèse préférée pour la cardinalité de* E, Y *infère* C_k
$\Leftrightarrow$
$\forall Y$ *thèse préférée pour la cardinalité de* E, Y *contient* C_k
$\Rightarrow$ *(à cause de la propriété 6.13.5)*
$\forall V_c$ *assignation card-maximale de* C, $Y(V_c)$ *est un sous-ensemble maximal pour la cardinalité de* C
$\Rightarrow$
$\forall V_c$ *assignation card-maximale de* C, $Y(V_c)$ *est une des thèses préférées pour la cardinalité de* E
$\Rightarrow$
$\forall V_c$ *assignation card-maximale de* C, $Y(V_c)$ *contient* C_k
$\Rightarrow$
Toute assignation V_c *card-maximale pour* C *vérifie :* $V_c(C_k) = vrai$

Nous en concluons que ACM $\propto$ *UNI-CAR, et ainsi que UNI-CAR est* Δ_2^p*-complet.*

En conclusion : *UNI-CAR est* Δ_2^p*-complet.*

6.14 Étude de la complexité de EXI-CAR

Rappel : *Le problème EXI-CAR est le suivant : "vérifier que* H *est une conséquence faible avec préférences de* E *en utilisant un ordre basé sur la cardinalité sur les thèses de* E*".*
Cela se note : $E \mid\!\sim^{\exists,Car} H$.

$1^{ère}$ **partie : l'appartenance à une classe.** *Un algorithme pour EXI-CAR est le suivant :*

1. deviner un sous-ensemble Y *de* E
2. vérifier que Y *est une thèse préférée pour la cardinalité de* E
3. vérifier que Y *infère* H.

La propriété 6.13.1 déjà citée pour UNI-CAR nous permet le second raffinage pour EXI-CAR dans lequel nous détaillons le point 2 de l'algorithme précédent : "vérifier que Y *est une thèse préférée pour la cardinalité de* E*" :*

1. deviner un sous-ensemble Y *de* E
*(*vérifier que* Y *est une thèse préférée pour la cardinalité de* E**)*
2. échec $\leftarrow$ *faux*
3. $k \leftarrow |Y|$
4. S'il existe une sous-base consistante Z *de* E *telle que* $|Z| > k$ *alors*
5. *échec* $\leftarrow$ *vrai*
fin si
6. si non échec alors
7. *vérifier que* Y *infère* H
fin si

Donc nous obtenons pour EXI-CAR un algorithme non déterministe polynomial faisant appel à deux oracles : un de complexité NP ("vérifier que Y *infère* H*") et l'autre de complexité inconnue ("existe-t-il une sous-base consistante* Z *de* E *telle que* $|Z| > k$*").*

Il nous faut donc détailler cet oracle-là défini par :

instance : E *une base et* k *un entier,*
question : *existe-t-il une interprétation satisfaisant strictement plus de* k *formules de* E *?*

Remarquons que ce problème est très proche du problème MAX-GSAT étudié pour UNI-CAR, sa seule différence venant du type d'inégalité utilisée qui est ici une inégalité stricte. Nous allons donc noter ce problème MAX-GSAT-STRICT et étudier sa complexité.

Un algorithme pour MAX-GSAT-STRICT est le suivant :

1. deviner une interprétation M
2. $p \leftarrow 0$
3. pour chaque formule G *de* E *faire*
4. *si* M *satisfait* G *alors* $p \leftarrow p + 1$
fin si
fin pour
5. vérifier que $p > k$

Le problème MAX-GSAT-STRICT appartient donc à la classe de complexité NP (à cause du "deviner"). Montrons maintenant la complétude. Il suffit de constater que GSAT se ramène polynomialement à MAX-GSAT-STRICT.

Soit "G*" une instance de GSAT, posons :* $E = \{G\}$, $k = 0$.

On a alors : G *est satisfiable* $\Leftrightarrow$ *il existe un modèle* M *de* $G \Leftrightarrow$ *il existe une sous-base consistante* Z *de* E *telle que* $|Z| > 0 \Leftrightarrow$ *MAX-GSAT-STRICT (*$\{G\}$*, 0). On a donc MAX-GSAT-STRICT NP-complet (y compris dans le cas de clauses). Remarquons que nous aurions aussi pu démontrer la complétude à partir du problème MAX-GSAT.*

L'algorithme proposé pour EXI-CAR est donc non déterministe (à cause du "deviner") polynomial et il fait appel à un oracle lui aussi non déterministe polynomial. La complexité de EXI-CAR est donc au plus $NP^{NP} = \Sigma_2^p$.

$2^{ème}$ **partie : la complétude ?** *Essayons de passer de 2-QBF à EXI-CAR grâce à une transformation polynomiale.*

Soit "$\exists a \forall b G(a,b)$*", une instance de 2-QBF, posons :* $H = G(a,b)$ *;* $E = \{a_1, \ldots, a_n, \neg a_1, \ldots, \neg a_n\}$

Ainsi :

- *on a dans* E *toutes les "valeurs" possibles pour les variables propositionnelles* $a_1, \ldots, a_n$ *dont on cherche s'il existe une assignation,*
- *on n'impose aucune contrainte sur les variables propositionnelles* $b_1, \ldots, b_m$*, dont toutes les assignations doivent être prises en compte.*

Remarque préliminaire : Y *est une thèse préférée pour la cardinalité de* $E \Leftrightarrow Y$ *est de la forme* $\{l_1, \ldots, l_n\}$ *avec* $l_i = a_i$ *ou* $\neg a_i$*,* $\forall i = 1 \ldots n$.

Montrons alors que : "$\{a_1, \ldots, a_n, \neg a_1, \ldots, \neg a_n\}$ *infère* $G(a,b)$ *avec la méthode EXI-CAR" équivaut à "*$\exists a \forall b G(a,b)$ *est satisfiable" :*

$\{a_1, \ldots, a_n, \neg a_1, \ldots, \neg a_n\}$ *infère* $G(a,b)$ *avec la méthode EXI-CAR*

$\Leftrightarrow$

il existe un sous-ensemble Y *de* E *tel que* Y *soit une thèse préférée pour la cardinalité et infère* $G(a,b)$

$\Leftrightarrow$

il existe un sous-ensemble $Y = \{l_1, \ldots, l_n\}$ *avec* $l_i = a_i$ *ou* $\neg a_i$, $\forall i = 1 \ldots n$, *tel que* $Y \vdash G(a, b)$

$\Leftrightarrow$

il existe $\{l_1, \ldots, l_n\}$ *qui infère* $G(a, b)$

$\Leftrightarrow$

$\exists a \forall b G(a, b)$ *est satisfiable*

Le problème 2-QBF se ramène donc polynomialement à EXI-CAR. Et on en déduit que le problème EXI-CAR est Σ_2^p*-complet. Remarque : c'est la même démonstration que pour EXI-T.*

En conclusion : *EXI-CAR est* Σ_2^p*-complet.*

6.15 Étude de la complexité de ARG-CAR

Rappel : *Le problème ARG-CAR est le suivant : "vérifier que* H *est une conséquence argumentative avec préférences de* E *en utilisant un ordre basé sur la cardinalité sur les thèses de* E*".*
Cela se note : $E \mid\!\sim^{A,Car} H$.

$1^{ère}$ **partie : l'appartenance à une classe.** *Un algorithme pour ARG-CAR est le suivant :*

1. *vérifier que* $E \not\mid\!\sim^{\exists,Car} \neg H$
2. *vérifier que* $E \mid\!\sim^{\exists,Car} H$

Cet algorithme est polynomial et il fait appel à un oracle de complexité Σ_2^p *(EXI-CAR).*
La complexité de ARG-CAR est donc au plus $P^{\Sigma_2^p} = \Delta_3^p$.

$2^{ème}$ **partie : la complétude ?** *Essayons de comparer le problème ARG-CAR avec des problèmes de classe inférieure dans la hiérarchie polynomiale. On constate alors que :*

- *le problème EXI-CAR se ramène polynomialement à ARG-CAR :*

  Soit "E, H*" une instance de EXI-CAR, posons :*
  - $f(H) = G$,
  - $f(E) = E \cup \{H \rightarrow G\}$
  avec G *nouvelle variable propositionnelle (*G *n'apparaît pas dans* E*).*

  Remarques préliminaires :

  - $H \rightarrow G$ *est consistante avec n'importe quelle thèse préférée pour la cardinalité* Y' *de* E *(voir argumentation donnée pour la démonstration de ARG-T, en sachant que toute thèse préférée pour la cardinalité est avant tout une thèse !).*
  - *Quelle que soit* Y *thèse préférée pour la cardinalité de* $f(E)$, Y *sera de la forme* $Y' \cup \{H \rightarrow G\}$ *avec* Y' *une thèse préférée pour la cardinalité de* E*. Ceci vient du fait que la formule* $H \rightarrow G$ *est consistante avec toutes les thèses préférées pour la cardinalité de* E.
  - *Quelle que soit* Y *thèse préférée pour la cardinalité de* $f(E)$, $Y \vdash G$ *ssi* $Y' \vdash H$ *(voir argumentation dans la démonstration de ARG-T).*

  Montrons alors que : "$f(E)$ *infère* $f(H)$ *avec la méthode ARG-CAR" équivaut à "*E *infère* H *avec la méthode EXI-CAR" :*

$f(E)$ *infère* G *avec la méthode ARG-CAR*
$\Leftrightarrow$
il existe Y *thèse préférée pour la cardinalité de* $f(E)$ *telle que* $Y \vdash G$
et aucune des autres thèses préférées pour la cardinalité n'infère $\neg G$.
$\Leftrightarrow$
il existe Y *thèse préférée pour la cardinalité de* $f(E)$ *telle que* $Y \vdash G$ *(voir remarques préliminaires)*
$\Leftrightarrow$
il existe Y' *thèse préférée pour la cardinalité de* E *telle que* $Y' \vdash H$ *(voir remarques préliminaires)*
$\Leftrightarrow$
E *infère* H *avec la méthode EXI-CAR*

- *par contre, nous n'avons pas réussi à montrer qu'un problème* Π_2^p*-complet (par exemple : co-EXI-CAR) se ramène polynomialement à ARG-CAR.*

Nous ne pouvons donc pas faire le même raisonnement que celui sur les autres relations d'inférence utilisant la conséquence argumentative (ARG-T, etc.). Seul le résultat d'appartenance demeure valable.

En conclusion : $\boxed{\textit{ARG-CAR} \in \Delta_3^p \textit{ et est } \Sigma_2^p\textit{-difficile.}}$

6.16 Étude de la complexité de UNI-LEX

Rappel : *Le problème UNI-LEX est le suivant : "vérifier que* H *est une conséquence forte avec préférences de* $(E, <)$ *en utilisant un ordre lexicographique sur les thèses de* $(E, <)$*".*

Cela se note : $(E, <) \mid\!\sim^{\forall, Lex} H$.

$1^{ère}$ **partie : l'appartenance à une classe.** *Remarque : Cette relation d'inférence est une généralisation de la relation UNI-CAR à une base stratifiée. Et, comme pour UNI-CAR, nous avons la propriété suivante généralisée cette fois à une base stratifiée :*

Propriété 6.16.1 *Soit* $(E, <)$ *un ensemble ordonné de formules, toutes les thèses préférées lexicographiquement dans* $(E, <)$ *ont le même cardinal strate par strate.*

La réflexion que nous avons menée nous a alors conduit aux faits suivants.

Soient $(E, <)$ *une base stratifiée et* H *une formule, posons :*
$f((E, <)) = (E \cup \{H \rightarrow G, \neg G\}, <)$ *et* $f(H) = \neg G$ *avec :*
- G *nouvelle variable propositionnelle n'apparaissant pas dans* E*,*
- $H \rightarrow G$ *constituant seule la première strate,*
- $\neg G$ *constituant seule la dernière strate.*

Puis, calculons les thèses Y *préférées lexicographiquement de* $f((E, <))$ *en utilisant la propriété 6.16.1 :*

Y' *thèse préférée lexicographiquement de* $(E, <)$ $(\|Y'\| = n)$	Y'' *thèse préférée lexicographiquement de* $(E \cup \{H \rightarrow G\}, <)$ *avec* $1^{ère}$ *strate* $= \{H \rightarrow G\}$	Y *thèse préférée lexicographiquement de* $f((E, <)) = (E \cup \{H \rightarrow G, \neg G\}, <)$ *avec* $1^{ère}$ *strate* $= \{H \rightarrow G\}$ *et dernière strate* $= \{\neg G\}$
Y' *infère* H	$Y'' = Y' \cup \{H \rightarrow G\}$ $(\|Y''\| = n + 1)$ *avec* $Y'' \vdash H$, $Y'' \vdash G$ *donc inconsistante avec* $\neg G$	Y *(noté* $Y_{1^{er}\ cas}$*)* $= Y'' = Y' \cup \{H \rightarrow G\}$ $(\|Y\| = n + 1)$ *avec* $Y \vdash H$, $Y \vdash G$ *et* $Y \nvdash \neg G$ *et il existe une telle* Y seulement si le cas donné dans la case ci-dessous ne se produit jamais !! *(puisque* $\|Y_{1^{er}\ cas}\| < \|Y_{2^{ème}\ cas}\|$ *et que l'on a la propriété 6.16.1)*
Y' *n'infère pas* H	$Y'' = Y' \cup \{H \rightarrow G\}$ $(\|Y''\| = n + 1)$ *avec* $Y'' \nvdash H$, $Y'' \nvdash G$ *donc consistante avec* $\neg G$	*Sachant que* $\neg G \notin Y'$ *donc* $\neg G \notin Y''$, Y *(noté* $Y_{2^{ème}\ cas}$*)* $= Y'' \cup \{\neg G\} =$ $Y' \cup \{H \rightarrow G, \neg G\}$ $(\|Y\| = n + 2)$ *avec* $Y \nvdash H$, $Y \nvdash G$ *et* $Y \vdash \neg G$ *et c'est la* seule forme possible *pour toute thèse préférée lexicographiquement de* $f((E, <))$ *(pour la même raison que celle fournie dans le* 1^{er} *cas). Ainsi on masque les* Y' *qui inféraient* H.

On a alors les propriétés suivantes :

Propriété 6.16.2 *Il existe une thèse préférée lexicographiquement de* $f((E, <))$ *qui infère* G *ssi toute thèse préférée lexicographiquement de* $(E, <)$ *infère* H.

Propriété 6.16.3 *Il existe une thèse préférée lexicographiquement de* $(E, <)$ *qui n'infère pas* H *ssi toute thèse préférée lexicographiquement de* $f((E, <))$ *infère* $\neg G$.

De ces deux propriétés, on déduit facilement que :

- *UNI-LEX* $\propto$ *EXI-LEX,*
- *co-UNI-LEX* $\propto$ *UNI-LEX.*

Ce qui nous donne pour UNI-LEX, l'algorithme suivant[19] *:*

1. $E' \leftarrow \{H \rightarrow G\} \cup E \cup \{\neg G\}$
2. $k \leftarrow < 0, 0, \ldots, 0 >$ *(*vecteur de dimension* n *= nombre de strates dans* E' **)*
3. pour $ns \leftarrow 1$ *à* n *faire*
4. $nf \leftarrow$ *nombre de formules dans la strate* E'_{ns}
5. $fini \leftarrow$ *faux*
6. tant que $(nf \geq 0)$ *et (non* $fini$*) faire*
7. $k[ns] \leftarrow nf$
8. si MAX-GSAT-ARRAY (E'*,*k*) alors*
9. $fini \leftarrow$ *vrai*
10. sinon $nf \leftarrow nf - 1$
fin si
fin tant que
fin pour
11. vérifier que $k[n]$ *est différent de 1*

[19] Cet algorithme repose en partie sur le fait que la base est stratifiée et que la formule $\neg G$ est placée seule dans la dernière strate. Il n'a donc pas pu être utilisé pour UNI-CAR, toutes les formules devant être dans la même strate.

Nous retrouvons ici le problème MAX-GSAT présenté pour UNI-CAR mais généralisé pour les bases stratifiées et noté alors MAX-GSAT-ARRAY :

> ***instance :*** *$(E, <)$ une base et k un vecteur d'entiers de dimension n (n = nb de strates dans E),*
> ***question :*** *existe-t-il une interprétation satisfaisant pour chaque strate E_i de E au moins $k[i]$ formules) ?*

Ce problème peut être résolu avec l'algorithme suivant :

1. deviner une interprétation M
*2. $i \leftarrow 1$ (*strate la plus prioritaire*)*
3. échec $\leftarrow$ faux
4. tant que $i \leq n$ (nombre de strates de E) et (non échec) faire
5. *$p \leftarrow 0$*
6. *pour chaque formule G de $(E, <)$ faire*
7. *si M satisfait G alors $p \leftarrow p + 1$*
fin si
fin pour
8. *si $p \geq k[i]$ alors*
9. *$i \leftarrow i + 1$*
10. *sinon échec $\leftarrow$ vrai*
fin si
fin tant que
11. vérifier que l'on n'a pas échec (échec = faux)

Il s'agit d'un algorithme non déterministe, donc MAX-GSAT-ARRAY appartient à la classe NP. Montrons maintenant la complétude. C'est immédiat. En effet, MAX-GSAT qui est NP-complet (voir démonstration pour UNI-CAR) est une restriction de MAX-GSAT-ARRAY.

Soit "E, k" une instance de MAX-GSAT, posons : *- $f(E) = E$,* *- $f(k) =< k >$* *avec $f(E)$ une base stratifiée constituée d'une seule strate.*

On a alors : MAX-GSAT (E,k) $\Leftrightarrow$ MAX-GSAT-ARRAY ($f(E)$,$f(k)$).

On a donc MAX-GSAT-ARRAY NP-complet. Ce qui nous permet de dire que l'algorithme proposé pour UNI-LEX est déterministe polynomial et qu'il fait appel à un oracle non déterministe polynomial. Le problème UNI-LEX appartient donc à la classe de complexité $P^{NP} = \Delta_2^p$[20].

[20] Il existe un autre algorithme possible pour UNI-LEX qui s'inspire de celui proposé pour UNI-CAR. On y utilise le problème : NGSAT-LEX : **instance :** $(E, <)$ une base stratifiée, H une formule propositionnelle, k un vecteur de dimension n (n = nb de strates dans E) avec $k[i]$ = nombre maximum de formules pouvant être satisfaites dans la strate i (même vecteur k que celui décrit dans l'algorithme précédent), **question :** existe-t-il une interprétation M qui satisfait $k[i]$ formules de chaque strate i de E telle que M ne satisfait pas H ? Ce problème NGSAT-LEX est une généralisation du problème NGSAT-CAR défini lors de la démonstration pour UNI-CAR de la même manière que MAX-GSAT-ARRAY est une généralisation de MAX-GSAT. NGSAT-LEX est NP-complet. Cela se démontre en utilisant l'algorithme suivant :

1. deviner une interprétation M
2. vérifier qu'elle satisfait $k[i]$ formules de chaque strate i de E
(c'est à dire :*
a. *$k' \leftarrow < 0, 0, \ldots, 0 >$ (*k' de même dimension n que k*)*
b. *pour chaque strate i de 1 à n faire*
c. *pour chaque formule G de la strate i faire*
d. *si M satisfait G alors*
e. *$k'[i] \leftarrow k'[i] + 1$*
fin si

$2^{ème}$ **partie : la complétude ?** *On sait que UNI-CAR est Δ_2^p-complet. Or UNI-CAR est une restriction de UNI-LEX, donc on peut prouver que UNI-CAR $\propto$ UNI-LEX en utilisant la transformation polynomiale suivante :*

Soit "E, H" une instance de UNI-CAR, posons : *- $f(H) = H$,* *- $f(E) = E$,* *$f(E)$ étant alors une base stratifiée ne contenant qu'une seule strate.*

Donc UNI-LEX est Δ_2^p-complet[21].

---

fin pour
fin pour
f. vérifier que $k' = k$
**)*
3. vérifier que M ne satisfait pas H

L'algorithme pour NGSAT-LEX est non déterministe polynomial, donc NGSAT-LEX appartient à la classe de complexité NP. Quant à la preuve de complétude, elle se fait en ramenant polynomialement NGSAT-CAR à NGSAT-LEX. Soit "E,H,k" une instance de NGSAT-CAR, posons : $f(E) = E$, $f(H) = H$ et $f(k)$ vecteur de dimension 1 égal à $< k >$. On a alors : NSAT-CAR $\propto$ NGSAT-LEX. Nous avons ainsi un nouvel algorithme pour UNI-LEX :

(calcul du vecteur k *)*
1. $k \leftarrow < 0, 0, \ldots, 0 >$ (vecteur de dimension n (nombre de strates dans E)*)*
2. pour $ns \leftarrow$ 1 à n faire
3. $nf \leftarrow$ nombre de formules dans la strate E_{ns}
4. $fini \leftarrow$ faux
5. tant que ($nf \geq 0$) et (non $fini$) faire
6. $k[ns] \leftarrow nf$
7. si MAX-GSAT-ARRAY (E,k) alors
8. $fini \leftarrow$ vrai
9. sinon $nf \leftarrow nf - 1$
fin si
fin tant que
fin pour
10. si NGSAT-LEX (E,H,k) alors
11. échec $\leftarrow$ vrai
12. sinon échec $\leftarrow$ faux
fin si
13. vérifier que l'on n'a pas échec (échec = faux)

Cet algorithme montre lui-aussi que UNI-LEX appartient à la classe Δ_2^p. Remarquons que l'on utilise ici aussi l'oracle MAX-GSAT-ARRAY qui a été présenté avec le premier algorithme de UNI-LEX. Signalons toutefois que le premier algorithme proposé pour UNI-LEX est plus efficace que celui puisqu'il ne fait appel qu'à un seul oracle NP alors que le second algorithme en utilise 2.

[21] On peut démontrer directement ce résultat à partir du même type de problème que celui utilisé pour UNI-CAR. Nous allons présenter cette preuve ici car elle nous servira lors de l'étude de la complétude du problème 1/STRATE (cf. section 7.1.2). Le problème choisi est (cf. [EG93]) :

> **Instance :** Soit $C = \{C_1, \ldots, C_m\}$ ensemble de clauses satisfiable, soit $X = \{x_1, \ldots, x_n\}$ ensemble des variables propositionnelles apparaissant dans C, soit $O(X) =< x_1, \ldots, x_n >$ une prioritisation de X (une "prioritisation" de X est une suite strictement ordonnée $< x_1, \ldots, x_n >$ des variables de X),
>
> **Question :** L'assignation lex-maximale V_M pour la prioritisation $O(X)$ satisfaisant C vérifie-t-elle : $V_M(x_n) =$ vrai ?

Nous notons ce problème ALM (pour Assignation Lexicographiquement Maximale) et rappelons la définition d'une assignation lex-maximale :

Définition 6.16.1 *Soient un ensemble de variables $X = \{x_1, \ldots, x_n\}$ et $O(X) =< x_1, \ldots, x_n >$ une prioritisation de X, avec x_1 la plus prioritaire et x_n la moins prioritaire, soient deux assignations V et W sur cet ensemble de variables, on dit que $V >^{lex} W$ pour $O(X)$ ssi $\exists i \in [1 \ldots n]$ tel que $\forall j < i$, $V(x_j) = W(x_j)$, $V(x_i) =$ vrai, $W(x_i) =$ faux.*

Définition 6.16.2 *Pour une prioritisation de X donnée, V_M est une assignation lex-maximale ssi V_M est maximale pour l'ordre $>^{lex}$.*

Remarques préliminaires : Soit $C = \{C_1, \ldots, C_m\}$ ensemble de clauses satisfiable, soient $X = \{x_1, \ldots, x_n\}$ ensemble des variables propositionnelles apparaissant dans C et $O(X) =< x_1, \ldots, x_n >$ prioritisation de X.

1. L'ordre $>^{lex}$ est strict et total, donc il n'existe qu'une seule assignation lex-maximale satisfaisant C pour une prioritisation de X donnée.

En conclusion : *UNI-LEX est Δ_2^p-complet.*

6.17 Étude de la complexité de EXI-LEX

Rappel : *Le problème EXI-LEX est le suivant : "vérifier que H est une conséquence faible avec préférences de $(E, <)$ en utilisant un ordre lexicographique sur les thèses de $(E, <)$".*

2. Soit V une assignation de X satisfaisant C, posons $Y(V) = \{x_i$ tel que $V(x_i) =$ vrai , $i = 1 \ldots n\}$. On a alors les propriétés suivantes :

   Propriété 6.16.4 *$Y(V)$ est une sous-base de X consistante avec C.*

   Propriété 6.16.5 *Soient V et W assignations de X, soit $O(X)$ une prioritisation de X. Si $V >^{lex} W$ pour $O(X)$ alors $Y(V) \gg^{lex} Y(W)$ (l'ordre $\gg^{lex}$ est l'ordre lexicographique tel que défini par [BCD+ 93]).*

   Preuve : Avec $O(X) =< x_1, \ldots, x_n >$, $V >^{lex} W$ pour $O(X) \Leftrightarrow \exists i \in [1 \ldots n]$ tel que $\forall j < i, V(x_j) = W(x_j), V(x_i) =$ vrai, $W(x_i) =$ faux $\Rightarrow \exists i \in [1 \ldots n]$ tel que $\forall j < i$ on a :

   - soit ($x_j \in Y(V)$ et $x_j \in Y(W)$), soit ($x_j \notin Y(V)$ et $x_j \notin Y(W)$)
   - et $x_i \in Y(V)$ et $x_i \notin Y(W)$

   $\Rightarrow$ par définition $Y(V) \gg^{lex} Y(W)$

Essayons maintenant d'utiliser ce problème ALM pour prouver la complétude de UNI-LEX.

Soit "$C = \{C_1, \ldots, C_m\}$ satisfiable, $X = \{x_1, \ldots, x_n\}$, $O(X) =< x_1, \ldots, x_n >$" une instance de ALM, posons :
$H = x_n$ et
$(E, <) = \{C_1 \wedge \ldots \wedge C_m, x_1, \ldots, x_n\}$,
l'ordre $<$ étant le suivant : la formule $CC = C_1 \wedge \ldots \wedge C_m$ est plus prioritaire que la formule x_1 elle-même plus prioritaire que la formule x_2 elle-même plus prioritaire que …plus prioritaire que x_n.

Remarques préliminaires :

- La première strate E_1 de E est consistante puisque C est satisfiable.
- Toute thèse préférée lexicographiquement contient la formule de la première strate.
- Comme il n'y a qu'une formule par strate dans E, il n'existe alors qu'une seule et unique thèse préférée lexicographiquement qui est aussi la seule préférée pour l'inclusion (voir propriété 7.1.1). Notons Y_0 cette thèse. $Y_0 = E_1 \cup (\{\alpha_1, \ldots, \alpha_n\} \setminus \{\top\}$ avec :
  - $\top =$ tautologie,
  - $\alpha_1 = x_1$ s'il existe un modèle de CC avec x_1 à vrai, sinon $\alpha_1 = \top$,
  - $\alpha_2 = x_2$ s'il existe un modèle de CC avec α_1, x_2 à vrai, sinon $\alpha_2 = \top$,
  - $\vdots$
  - $\alpha_n = x_n$ s'il existe un modèle de CC avec $\alpha_1, \ldots, \alpha_{n-1}, x_n$ à vrai, sinon $\alpha_n = \top$.

  Définition 6.16.3 *Définissons alors l'assignation V_0 de la façon suivante : $V_0(x_i) =$ vrai si $x_i \in Y_0$, faux sinon. On a alors $Y(V_0) \cup E_1 = Y_0$ et V_0 satisfait C.*

  On a alors la propriété suivante :

  Propriété 6.16.6 *V_0 ainsi construite est exactement l'assignation lex-maximale pour $O(X)$.*

  Preuve : Raisonnons par l'absurde. Si V_0 n'est pas l'assignation lex-maximale pour $O(X)$ satisfaisant C alors il existe une autre assignation notée V_M qui est l'assignation lex-maximale pour $O(X)$ satisfaisant C. Donc $V_M >^{lex} V_0$ pour $O(X)$ alors $Y(V_M) \gg^{lex} Y(V_0)$. Cela implique que $Y(V_0) \cup E_1$ n'est pas lex-préférée pour E. On a donc une contradiction avec la condition imposée à la construction de V_0 (cf. définition 6.16.3).
- $V_M(x_n) =$ vrai ssi Y_0 contient x_n

Démontrons alors que ALM $\propto$ UNI-LEX :

Soient C, X et $O(X)$, on a $V_M(x_n) =$ vrai
$\Leftrightarrow$
$\forall Y$ thèse préférée lexicographiquement de E, Y contient x_n
$\Leftrightarrow$
$\forall Y$ thèse préférée lexicographiquement de E, Y infère x_n

Nous en concluons que ALM $\propto$ UNI-LEX, et ainsi que UNI-LEX est Δ_2^p-complet.

Cela se note : $(E, <) \mid\sim^{\exists, Lex} H$.

$1^{ère}$ **partie : l'appartenance à une classe.** *Remarquons que EXI-LEX est une généralisation de EXI-CAR pour des bases stratifiées.*

Un algorithme pour EXI-LEX est le suivant :

1. deviner un sous-ensemble Y *de* $(E, <)$
2. vérifier que Y *est une thèse préférée lexicographiquement de* $(E, <)$
3. vérifier que Y *infère* H.

Comme pour EXI-CAR, on peut raffiner de la manière suivante :

1. deviner un sous-ensemble Y *de* $(E, <)$
*(*vérifier que* Y *est une thèse préférée lexicographiquement de* $(E, <)$**)*
2. échec $\leftarrow$ *faux*
3. $i \leftarrow 1$ *(*strate la plus prioritaire*)*
4. tant que non échec et $i < n + 1$ *faire*
5. $k \leftarrow |Y_1| + |Y_2| + \ldots + |Y_i|$
6. S'il existe une sous-base consistante Z *de* $E_1 \cup E_2 \cup \ldots \cup E_i$ *telle que* $|Z| > k$ *alors*
7. échec $\leftarrow$ *vrai*
8. sinon $i \leftarrow i + 1$
fin si
fin tant que
9. si non échec alors
10. vérifier que Y *infère* H
fin si

Donc nous obtenons pour EXI-LEX un algorithme non déterministe polynomial faisant appel à deux oracles de complexité NP (GSAT et MAX-GSAT-STRICT déjà présenté pour EXI-CAR). Ce qui nous permet de dire que l'algorithme proposé pour EXI-LEX est non déterministe (à cause du "deviner") polynomial et qu'il fait appel à un oracle lui aussi non déterministe polynomial. La complexité de EXI-LEX est donc au plus $NP^{NP} = \Sigma_2^p$.

$2^{ème}$ **partie : la complétude ?** *EXI-CAR étant une restriction de EXI-LEX, on peut définir la transformation polynomiale suivante :*

Soit "E, H*" une instance de EXI-CAR, posons :* - $f(H) = H$, - $f(E) = E$, $f(E)$ *étant alors une base stratifiée ne contenant qu'une seule strate.*

On a donc : EXI-CAR $\propto$ *EXI-LEX. Or EXI-CAR est* Σ_2^p*-complet. Et on en déduit que le problème EXI-LEX est* Σ_2^p*-complet. Remarque : c'est le même type de démonstration que pour EXI-INCL.*

En conclusion : *EXI-LEX est* Σ_2^p*-complet.*

6.18 Étude de la complexité de ARG-LEX

Rappel : *Le problème ARG-LEX est le suivant : "vérifier que* H *est une conséquence argumentative avec préférences de* $(E, <)$ *en utilisant un ordre lexicographique sur les thèses de* $(E, <)$*".*

Cela se note : $(E, <) \mid\sim^{A, Lex} H$.

$1^{ère}$ **partie : l'appartenance à une classe.** *Remarquons que ARG-LEX est une généralisation de ARG-CAR aux bases stratifiées.*

Un algorithme pour ARG-LEX est le suivant :

1. *vérifier que* $(E, <) \not\vdash^{\exists, Lex} \neg H$
2. *vérifier que* $(E, <) \vdash^{\exists, Lex} H$

Cet algorithme est polynomial et il fait appel à un oracle de complexité Σ_2^p *(EXI-LEX).*

La complexité de ARG-LEX est donc au plus $P^{\Sigma_2^p} = \Delta_3^p$.

$2^{ème}$ **partie : la complétude ?** *ARG-CAR étant une restriction de ARG-LEX, on peut définir la transformation polynomiale suivante :*

Soit "E, H*" une instance de ARG-CAR, posons :* - $f(H) = H$, - $f(E) = E$, $f(E)$ *étant alors une base stratifiée ne contenant qu'une seule strate.*

On obtient ainsi que ARG-CAR $\propto$ *ARG-LEX, or on sait que EXI-CAR* $\propto$ *ARG-CAR, donc EXI-CAR* $\propto$ *ARG-LEX. Par contre, comme pour ARG-CAR, nous n'avons pas réussi à montrer qu'un problème* Π_2^p*-complet (par exemple : co-EXI-LEX ou co-EXI-CAR) se ramène polynomialement à ARG-LEX.*

Nous ne pouvons donc pas faire le même raisonnement que celui sur les autres relations d'inférence utilisant la conséquence argumentative (ARG-T, etc.).

Seul le résultat d'appartenance demeure valable.

En conclusion : | *ARG-LEX* $\in \Delta_3^p$ *et est* Σ_2^p*-difficile.* |

6.19 Étude de la complexité de UNI-E

Rappel : *Le problème UNI-E est le suivant : "vérifier que* H *est une conséquence forte de* $E = (W, D)$ *théorie des défauts propositionnelle en utilisant les extensions de* E*".*

Cela se note : $E \vdash^{\forall, E} H$.

Ce problème a été étudié par Gottlob dans [Got92]. Ses conclusions sont les suivantes.

Conclusion de Gottlob : | *UNI-E est* Π_2^p*-complet.* |

Remarques : *Gottlob a montré que UNI-E reste* Π_2^p*-complet même si* $E = (W, D)$ *est une théorie des défauts propositionnelle normale (c'est-à-dire dont les défauts sont de la forme* $a : b/b$*).*

6.20 Étude de la complexité de EXI-E

Rappel : *Le problème EXI-E est le suivant : "vérifier que* H *est une conséquence faible de* $E = (W, D)$ *théorie des défauts propositionnelle en utilisant les extensions de* E*".*

Cela se note : $E \vdash^{\exists, E} H$.

Ce problème a été étudié par Gottlob dans [Got92]. Ses conclusions sont les suivantes.

Conclusion de Gottlob : *EXI-E est Σ_2^p-complet.*

Remarques : *Gottlob a montré que EXI-E reste Σ_2^p-complet même si $E = (W, D)$ est une théorie des défauts propositionnelle normale.*

6.21 Étude de la complexité de ARG-E

Rappel : *Le problème ARG-E est le suivant : "vérifier que H est une conséquence argumentative de $E = (W, D)$ théorie des défauts propositionnelle en utilisant les extensions de E".*

Cela se note : $E \mid\!\sim^{A,E} H$.

À notre connaissance, Gottlob ne s'est pas encore intéressé à ce type de relation d'inférence sur une théorie des défauts.

$1^{ère}$ **partie : l'appartenance à une classe.** *Un algorithme pour ARG-E est le suivant :*

1. *vérifier que $(E, <) \not\mid\!\sim^{\exists,E} \neg H$*
2. *vérifier que $(E, <) \mid\!\sim^{\exists,E} H$*

Cet algorithme est polynomial et il fait appel à un oracle de complexité Σ_2^p (EXI-E).

La complexité de ARG-E est donc au plus $P^{\Sigma_2^p} = \Delta_3^p$.

$2^{ème}$ **partie : la complétude ?** *Nous avons toujours les mêmes difficultés à étudier la complétude, donc essayons de montrer que ARG-E $\in \Delta_3^p$ - ($\Sigma_2^p \cup \Pi_2^p$).*

Pour cela on va essayer de comparer le problème ARG-E avec des problèmes de classe inférieure dans la hiérarchie polynomiale.

On constate alors que :

- *le problème EXI-E se ramène polynomialement à ARG-E :*

Soit "$E = (W, D), H$", une instance de EXI-E, posons : *- $f(H) = G$,* *- $f(E) = (W \cup \{H \rightarrow G\}, D)$* *avec G nouvelle variable propositionnelle (G n'apparaît pas dans W, ni dans D).*

  Remarques préliminaires :

  - *$H \rightarrow G$ est consistante avec n'importe quelle extension Y' de E car : Y' est une extension $\Rightarrow$ elle est consistante $\Rightarrow$ elle admet un modèle M'. Or G n'apparaît pas dans E, donc n'apparaît pas dans Y'. On peut ainsi étendre le modèle M' de Y' à un modèle M en rajoutant la valeur VRAI pour la variable propositionnelle G. On a ainsi M modèle de Y' et modèle de $H \rightarrow G$. $H \rightarrow G$ est donc consistante avec toute extension Y' de E.*
  - *$H \rightarrow G$ ne permet pas l'activation d'autres défauts que ceux activés pour le calcul des extensions de E, puisque G n'apparaît pas dans D (et en particulier, dans les pré-requis des défauts de D).*
  - *Le fait de rajouter cette prémisse à E permet de reprendre en intégralité l'arbre de recherche des extensions de E et de construire l'arbre de recherche des extensions de $f(E)$ à partir de l'étude sur chaque branche de l'arbre de E de l'ajout de la prémisse $H \rightarrow G$ (voir la construction de l'arbre de recherche présenté à la section 2.3).*

  Cela nous donne les résultats présentés dans le tableau 6.1 et on obtient les propriétés suivantes :

État de la branche de l'arbre de E	Résultat sur la construction de l'arbre pour $f(E)$
inconsistance	inconsistance
extension Y'	extension Y de $f(E) = Cn(Y' \cup \{H \rightarrow G\})$
échec car certaines hypothèses sur l'extension n'ont pas été vérifiées	peut-être une solution si la seule hypothèse manquante est satisfaite par le fait : "$H \rightarrow G \in$ l'extension" ; or ce cas est impossible puisque G n'apparaît pas dans E ; on a donc un échec

Tableau 6.1:

Propriété 6.21.1 *Quelle que soit Y extension de $f(E)$, Y est de la forme $Y' \cup \{H \rightarrow G\}$ avec Y' extension de E.*

Propriété 6.21.2 *Quelle que soit Y extension de $f(E)$, $Y \vdash G$ ssi $Y' \vdash H$ (voir argumentation dans la démonstration de ARG-T).*

Propriété 6.21.3 *Aucune extension de E ou de $f(E)$ ne peut inférer $\neg G$.*

Montrons alors que : "$f(E)$ infère $f(H)$ avec la méthode ARG-E" équivaut à "E infère H avec la méthode EXI-E" :

$f(E)$ *infère* G *avec la méthode ARG-E*
$\Leftrightarrow$
il existe Y *extension de* $f(E)$ *telle que* $Y \vdash G$
et aucune des autres extensions n'infère $\neg G$.
$\Leftrightarrow$
(à cause de la propriété 6.21.3)
il existe Y *extension de* $f(E)$ *telle que* $Y \vdash G$
$\Leftrightarrow$
(à cause des propriétés 6.21.1 et 6.21.2)
il existe Y' *extension de* E *telle que* $Y' \vdash H$
$\Leftrightarrow$
E *infère* H *avec la méthode EXI-E*

- *le problème co-EXI-E se ramène polynomialement à ARG-E :*

Soit "$E = (W, D), H$", une instance de co-EXI-E, posons :
- $f(E) = (W, D \cup \{: \neg H / \neg H\}$,
- $f(H) = \neg H$.

Remarque préliminaire :
Le fait de rajouter un défaut à E permet de reprendre en intégralité l'arbre de recherche des extensions de E et de construire l'arbre de recherche des extensions de $f(E)$ à partir de l'arbre pour E et

de l'étude sur chaque branche de l'arbre de E de l'activation ou non du défaut :$\neg H/\neg H$. Cela nous donne les résultats présentés dans le tableau 6.2 et on obtient les propriétés suivantes :

Propriété 6.21.4 *Si toutes les extensions Y' de E sont consistantes avec $\neg H$ (donc n'infèrent pas H) alors toutes les extensions de $f(E)$ infèrent $\neg H$ et n'infèrent pas H ;*

Propriété 6.21.5 *S'il existe une extension Y' de E qui infère H (donc inconsistante avec $\neg H$) alors il existe une extension Y de $f(E)$ qui infère H ($Y = Y'$) ;*

Propriété 6.21.6 *(contraposée de la propriété 6.21.5) Si aucune extension Y de $f(E)$ n'infère H alors aucune extension Y' de E n'infère H.*

Montrons alors que : "$f(E)$ infère $f(H)$ avec la méthode ARG-E" équivaut à "E n'infère pas H avec la méthode EXI-E".
Tout d'abord :

$$
\begin{array}{c}
f(E) \text{ infère } \neg H \text{ avec la méthode ARG-E} \\
\Leftrightarrow \\
\exists Y \text{ extension de } f(E) \text{ telle que } Y \vdash \neg H \\
\text{et } \forall Y \text{ extension de } f(E),\ Y \nvdash H \\
\Rightarrow \\
\text{(à cause de la propriété 6.21.6)} \\
\forall Y' \text{ extension de } E,\ Y' \nvdash H \\
\Leftrightarrow \\
E \text{ n'infère pas } H \text{ avec la méthode EXI-E}
\end{array}
$$

Et réciproquement :

$$
\begin{array}{c}
E \text{ n'infère pas } H \text{ avec la méthode EXI-E} \\
\Leftrightarrow \\
\forall Y' \text{ extension de } E,\ Y' \nvdash H \\
\Rightarrow \\
\text{(à cause de la propriété 6.21.4)} \\
\forall Y \text{ extension de } f(E),\ Y \nvdash H \text{ et } Y \vdash \neg H \\
\Rightarrow \\
\exists Y \text{ extension de } f(E) \text{ telle que } Y \vdash \neg H \\
\text{et } \forall Y \text{ extension de } f(E),\ Y \nvdash H \\
\Leftrightarrow \\
f(E) \text{ infère } \neg H \text{ avec la méthode ARG-E}
\end{array}
$$

On a donc EXI-E $\propto$ ARG-E et co-EXI-E $\propto$ ARG-E (avec EXI-E qui est Σ_2^p-complet et co-EXI-E qui est Π_2^p-complet). Ainsi, comme pour ARG-T, on arrive à la conclusion suivante.

En conclusion : $\boxed{\text{Si } \Sigma_2^p \neq \Pi_2^p, \text{ ARG-E} \in \Delta_3^p \text{ - } (\Sigma_2^p \cup \Pi_2^p).}$

État de la branche de l'arbre de E	Application ou non du défaut $:\neg H/\neg H$	Résultat sur la construction de l'arbre pour $f(E)$
inconsistance	quelle qu'elle soit !	inconsistance
extension Y' consistante avec $\neg H$ donc n'inférant pas H	activation du défaut : H ne doit pas $\in$ à l'extension finale	extension Y de $f(E) = Cn(Y' \cup \{\neg H\})$ si $\neg H \notin Y'$ ou $Cn(Y')$ si $\neg H \in Y'$ Y contient $\neg H$ et n'infère pas H
	non activation du défaut : H doit $\in$ à l'extension finale	échec : H ne peut pas $\in$ à l'extension finale
extension Y' inconsistante avec $\neg H$ donc inférant H	activation du défaut : H ne doit pas $\in$ à l'extension finale	échec : $H \in$ à l'extension finale
	non activation du défaut : H doit $\in$ à l'extension finale	extension Y de $f(E) = Y'$ Y ne contient pas $\neg H$ et infère H
échec car certaines hypothèses sur l'extension n'ont pas été vérifiées	activation du défaut : H ne doit pas $\in$ à l'extension finale	peut-être une solution si la seule hypothèse manquante est satisfaite par le fait : "$\neg H \in$ l'extension" ; dans ce cas là, on a : extension Y de $f(E) = Cn(S \cup \{\neg H\})$ avec $S = W \cup$ {les conséquents des défauts de E activés pour cette branche} Y contient $\neg H$ et n'infère pas H
	non activation du défaut : H doit $\in$ à l'extension finale	échec : la non activation du défaut ne permet pas de vérifier les hypothèses manquant déjà pour l'extension de E

Tableau 6.2:

Chapitre 7

Étude de complexité des différentes relations d'inférence dans des cas particuliers

Nous allons ici refaire une étude de complexité des relations d'inférence étudiées dans le chapitre précédent mais en traitant trois cas particuliers :

- *celui où $(E, <)$ est un ensemble ordonné totalement et strictement ; ce qui revient à dire que l'on a une base initiale E stratifiée avec une seule formule par strate ;*
- *celui où $(E, <)$ est un ensemble totalement ordonné de formules sous forme conjonctive normale (formules CNF) ;*
- *celui où $(E, <)$ est un ensemble totalement ordonné de clauses de Horn.*

Les relations d'inférence UNI (EXI, ARG)-T (S, CAR, E) ne sont pas concernées par le premier cas particulier. En effet, ces relations d'inférence ne tiennent pas compte de l'ordre sur E. On peut aussi montrer que les problèmes UNI (EXI, ARG)-INCL (LEX) deviennent équivalents (voir démonstration dans la section suivante).

L'étude du second cas particulier permet de traiter une sous-famille de la logique propositionnelle très souvent utilisée (les clauses) mais ne permet hélas aucune simplification de la complexité des problèmes étudiés.

Dans le troisième cas particulier, le problème GSAT devient polynomial et sera noté SAT-HORN, ce qui induit bien sûr des complexités plus faibles pour les problèmes posés.

7.1 Cas d'une base stratifiée avec une seule formule par strate

Dans toute cette section, $(E, <)$ dénotera une base stratifiée avec une seule formule par strate.

Dans ce cas particulier, on a la propriété suivante :

Propriété 7.1.1 *Soit $<$ ordre total et strict sur E, il y a une seule thèse préférée pour l'ordre "INCLUSION BASED" et une seule thèse préférée lexicographiquement, et c'est la même.*

Preuve : *Prenons les notations suivantes : $E = E_1 \cup \ldots \cup E_n$ (E_i strate composée d'une seule formule), on note $F_i = E_i \cap F$, $\forall i$ et $\forall F$ sous-ensemble de E.*

Rappelons les deux définitions suivantes :

Définition 7.1.1 *Soient A et B deux sous-ensembles consistants de E, $A = A_1 \cup \ldots \cup A_n$ et $B = B_1 \cup \ldots \cup B_n$, A est préféré à B pour l'ordre "INCLUSION BASED" (noté $B \ll^d A$) ssi il existe i tel que $B_i \subset A_i$[1] et $\forall j < i$, $A_j = B_j$[2].*

Définition 7.1.2 *Soient A et B deux sous-ensembles consistants de E, $A = A_1 \cup \ldots \cup A_n$ et $B = B_1 \cup \ldots \cup B_n$, A est préféré à B pour l'ordre lexicographique (noté $B \ll^{lex} A$) ssi il existe i tel que $|B_i| < |A_i|$ et $\forall j < i$, $|A_j| = |B_j|$.*

Remarquons en premier lieu que, dans le cas d'une base dont les strates sont composées d'une seule formule, on a : $\forall i$, $A_i = \varnothing$ ou $A_i = \{f_i\}$ avec f_i la formule de la strate numéro i. De même pour B.
Montrons alors que les deux ordres donnent le même résultat : $\forall B$ et A, $B \ll^d A \Leftrightarrow B \ll^{lex} A$.

$$B \ll^d A$$
$$\Leftrightarrow$$
$$\exists i \text{ tel que } B_i \subset A_i \text{ et } \forall j < i,\ A_j = B_j$$
$$\Leftrightarrow$$
$$\exists i \text{ tel que } B_i = \varnothing \text{ et } A_i = \{f_i\} \text{ et } \forall j < i, \text{ soit } A_j = B_j = \varnothing, \text{ soit } A_j = B_j = \{f_j\}$$
$$\Leftrightarrow$$
$$\exists i \text{ tel que } |B_i| = 0 \text{ et } |A_i| = 1 \text{ et } \forall j < i, \text{ soit } |A_j| = |B_j| = 0, \text{ soit } |A_j| = |B_j| = 1$$
$$\Leftrightarrow$$
$$\exists i \text{ tel que } |B_i| < |A_i| \text{ et } \forall j < i,\ |A_j| = |B_j|$$
$$\Leftrightarrow$$
$$B \ll^{lex} A.$$

Montrons ensuite l'unicité des thèses préférées :

Définition 7.1.3 *Une thèse préférée pour l'ordre "INCLUSION BASED"*[3] *de E est un ensemble $S = S_1 \cup \ldots \cup S_n$ tel que $\forall k \in [1 \ldots n]$ $S_1 \cup \ldots \cup S_k$ est un sous-ensemble maximal consistant de $E_1 \cup \ldots \cup E_k$.*

Cette définition induit le processus constructif suivant : pour chaque strate k en partant de la plus prioritaire, soit la formule f_k (de la strate k) est consistante avec l'ensemble solution déjà construit, alors le nouvel ensemble solution est égal à l'union de $\{f_k\}$ avec l'ancien ensemble solution, soit f_k est inconsistante avec l'ensemble solution déjà construit, alors l'ensemble solution reste inchangé. Dans tous les cas, à la fin de chaque étape du processus, on a qu'un seul ensemble solution.

7.1.1 Complexité de UNI (EXI, ARG)-BO

Étude d'appartenance. *Contrairement à ce qui se passe pour les méthodes "INCLUSION BASED" et lexicographique (voir propriété 7.1.1), nous n'avons pas ici de simplification des algorithmes. En effet, le fait qu'il n'y ait qu'une formule par strate intervient dans le calcul des $a(Y)$ mais ne permet pas de se ramener à une seule et unique sous-base préférée pour l'ordre "BEST-OUT" (voir l'exemple donné à la section 2.4). On garde donc les mêmes résultats d'appartenance que ceux vus précédemment*[4] *:*

- *1/STRATE-UNI-BO appartient au plus à la classe de complexité Δ_2^p,*
- *1/STRATE-EXI-BO appartient au plus à la classe de complexité Σ_2^p,*
- *1/STRATE-ARG-BO appartient au plus à la classe de complexité Δ_3^p.*

[1] Le symbole $\subset$ dénote l'inclusion stricte.
[2] Nous avons pris ici la définition correspondant à l'ordre démocratique puisque c'est le plus général de tous les ordres "INCLUSION BASED" vus jusqu'ici.
[3] Au sens de Brewka, cette fois ! Cette définition est utilisée pour son aspect constructif.
[4] Toutefois, on modifie les notations pour mettre en évidence le fait qu'il s'agit de problèmes différents. Ainsi, UNI-BO devient 1/STRATE-UNI-BO (1/STRATE : pour une formule par strate). De même EXI-BO devient 1/STRATE-EXI-BO, et ARG-BO devient 1/STRATE-ARG-BO.

Étude de complétude.

- *Pour 1/STRATE-UNI-BO, on peut conserver exactement la même démonstration que celle du cas général, puisque les transformations utilisées définissent des bases avec une seule formule par strate. On prouve donc que : 1/STRATE-UNI-BO* $\in \Delta_2^p$ *- (NP* $\cup$ *co-NP).*
- *Par contre, pour 1/STRATE-EXI-BO, la démonstration du cas général ne s'applique plus puisque la base* $E = \{a_1, \ldots, a_n, \neg a_1, \ldots, \neg a_n\}$ *n'est pas strictement stratifiée. Nous proposons alors une nouvelle transformation polynomiale :*

> *Soit "*E, H*", une instance de EXI-S, posons :*
> *-* $f(H) = H$*,*
> *-* $(f(E), <) = \{\bot\} \cup E_{strat}$
> *avec* $\bot$ *symbolisant la contradiction et constituant seul la première strate et* E_{strat} *l'ensemble constitué des formules de* E *ordonnées strictement dans n'importe quel ordre.*

Nous constatons alors que, d'après la définition de l'ordre "BEST-OUT", le amax correspondant à $f(E)$ *est 1. Ainsi, les sous-bases préférées pour l'ordre "BEST-OUT" de* $f(E)$ *sont exactement les sous-bases consistantes de* E_{strat}*, elles-même étant les sous-bases consistantes de* E*. On a donc :*

$$\exists Y \text{ sous base consistante de } E \text{ telle que } Y \vdash H$$
$$\Leftrightarrow$$
$$\exists Y \text{ sous base consistante préférée pour l'ordre "BEST-OUT" de } f(E) \text{ telle que } Y \vdash H$$

On a ainsi EXI-S $\propto$ *1/STRATE-EXI-BO. Le problème 1/STRATE-EXI-BO est donc* Σ_2^p*-complet.*

- *Pour 1/STRATE-ARG-BO, nous pouvons garder la même démonstration que celle du cas général en partant des problèmes 1/STRATE-EXI-BO et co-1/STRATE-EXI-BO, puisque les deux transformations polynomiales utilisées génèrent des sous-bases stratifiées strictement. Nous en concluons que 1/STRATE-ARG-BO* $\in \Delta_3^p$ *- (*$\Sigma_2^p \cup \Pi_2^p$*) si* $\Sigma_2^p \neq \Pi_2^p$*.*

En conclusion.

> *si NP* $\neq$ *co-NP, 1/STRATE-UNI-BO* $\in \Delta_2^p$ *- (NP* $\cup$ *co-NP)*
> *1/STRATE-EXI-BO est* Σ_2^p*-complet*
> *si* $\Sigma_2^p \neq \Pi_2^p$*, 1/STRATE-ARG-BO* $\in \Delta_3^p$ *- (*$\Sigma_2^p \cup \Pi_2^p$*)*

7.1.2 Complexité de UNI(EXI, ARG)-INCL et UNI(EXI, ARG)-LEX

Comme il existe une unique thèse préférée à la fois pour l'ordre "INCLUSION BASED" et pour l'ordre lexicographique, on constate alors que les problèmes UNI-INCL, EXI-INCL et ARG-INCL deviennent équivalents ainsi que les problèmes UNI-LEX, EXI-LEX et ARG-LEX. Nous obtenons ainsi un unique problème noté 1/STRATE.

Étude d'appartenance. *Un algorithme pour 1/STRATE (noté* $(E, <) \mid\!\sim^{1-s} H$*) est le suivant :*

1. $X \leftarrow \varnothing$
2. $numstrate \leftarrow 1$
3. si $X \cup E_{numstrate}$ *est consistant alors*
4. $\quad X \leftarrow X \cup E_{numstrate}$
fin si
5. $numstrate \leftarrow numstrate + 1$
6. si $numstrate =$ *nb total de strates dans* E *alors*
7. $\quad$ *vérifier que* $X \vdash H$
8. sinon aller étape 3
fin si

Cet algorithme est déterministe polynomial et il fait appel à un oracle de complexité NP. Le problème 1/STRATE appartient donc à la classe de complexité Δ_2^p.

Étude de complétude. *Dans la section 6.16, nous avons présenté une démonstration de complétude pour UNI-LEX utilisant le problème ALM présenté par Eiter et Gottlob dans [EG93]. Or cette preuve s'applique en intégralité pour le problème 1/STRATE puisque la transformation polynomiale utilisée génère une base E strictement ordonnée :*

Soit "$C = \{C_1, \ldots, C_m\}$ satisfiable, $X = \{x_1, \ldots, x_n\}$, $O(X) =< x_1, \ldots, x_n >$" une instance de ALM, posons : $H = x_n$ *et* $(E, <) = \{C_1 \wedge \ldots \wedge C_m, x_1, \ldots, x_n\}$, *l'ordre $<$ étant le suivant : la formule $CC = C_1 \wedge \ldots \wedge C_m$ est plus prioritaire que la formule x_1 elle-même plus prioritaire que la formule x_2 elle-même plus prioritaire que ...plus prioritaire que x_n.*

Donc ALM $\propto$ 1/STRATE et on en conclut que 1/STRATE est Δ_2^p-complet.

En conclusion. *1/STRATE est Δ_2^p-complet.*

7.2 Cas d'une base de formules CNF

Dans cette section, les formules composant la base étudiée $(E, <)$ ainsi que la formule H sont sous forme normale conjonctive. Ce sont donc des conjonctions de disjonctions, ce qui, en logique propositionnelle, correspond à des ensembles de clauses. Dans ce cas précis, le problème de satisfiabilité utilisé n'est plus GSAT mais SAT qui est lui-aussi NP-complet (voir [GJ79])[5]*. Par contre, le problème 2-QBF n'est plus Σ_2^p-complet dans le cas de formules CNF alors qu'il l'est toujours pour des formules DNF (voir [Sto77]). Cela remet donc en cause quelques-unes de nos démonstrations. D'autre part, de nombreuses transformations polynomiales utilisées lors des démonstrations de complétude dans le cas général ne conservent pas la forme conjonctive normale (voir en particulier les démonstrations pour les relations argumentatives).*

7.2.1 Complexité de UNI (EXI, ARG)-T-CNF

Étude d'appartenance. *Tous les algorithmes étant conservés, on obtient les mêmes résultats d'appartenance que dans le cas général :*

- *UNI-T-CNF appartient au plus à la classe de complexité Π_2^p,*
- *EXI-T-CNF appartient au plus à la classe de complexité Σ_2^p,*
- *ARG-T-CNF appartient au plus à la classe de complexité Δ_3^p.*

Étude de complétude.

- *pour UNI-T-CNF, au lieu d'utiliser le problème 2-QBF général comme dans la démonstration de UNI-T, on choisit d'utiliser le problème 2QBF-DNF qui est Σ_2^p-complet (voir [Sto77]) : "$\exists a \forall b G(a, b)$ satisfiable ? avec $G(a, b)$ sous forme DNF". On peut alors garder exactement la même transformation polynomiale puisque l'ensemble E et la formule H sont alors en forme CNF (rappel : $E = \{a_1, \ldots, a_n, \neg a_1, \ldots, \neg a_n, \neg G(a, b)\}$ et $H = \neg G(a, b)$). On a donc UNI-T-CNF qui est Π_2^p-complet.*
- *pour EXI-T-CNF, on va utiliser le problème co-2QBF-DNF qui est Π_2^p-complet.*

[5] Même remarque pour UNGSAT qui devient UNSAT.

Soit "$\forall a \exists b G(a,b)$ avec $G(a,b)$ sous-forme CNF", une instance de co-2-QBF-DNF, posons :
- $H = s$,
- $E = \{a_1, \ldots, a_n, \neg a_1, \ldots, \neg a_n, G(a,b) \vee s\}$
avec s une nouvelle variable propositionnelle.

Les thèses de E sont alors de la forme $Y = \{l_1, \ldots, l_n, G(a,b) \vee s\}$ avec $l_i = a_i$ ou $\neg a_i$, $\forall i$. En effet, il suffit de remarquer que la formule $G(a,b) \vee s$ est toujours consistante avec n'importe quel ensemble $\{l_1, \ldots, l_n\}$ puisque s est un nouveau symbole et qu'il suffit d'étendre les interprétations satisfaisant les l_i en rajoutant la valeur vrai pour s. D'autre part, on remarque aussi que :

- *$\{l_1, \ldots, l_n\}$ consistant avec $G(a,b) \Leftrightarrow$*
  il existe une extension M de l'assignation $\{l_1, \ldots, l_n\}$ pour les autres symboles de $G(a,b)$ (les b_i) telle que M satisfait $G(a,b)$, et M satisfait $G(a,b) \vee s$ quelle que soit la valeur de $s \Leftrightarrow$
  $\{l_1, \ldots, l_n, G(a,b) \vee s\}$ n'infère pas s.
- *$\{l_1, \ldots, l_n\}$ inconsistant avec $G(a,b) \Leftrightarrow$*
  $\{l_1, \ldots, l_n\}$ infère $\neg G(a,b) \Leftrightarrow$
  $\{l_1, \ldots, l_n, G(a,b) \vee s\}$ infère s.

On a alors :

$$\forall a \exists b G(a,b) \text{ satisfiable}$$
$$\Rightarrow$$
$$\forall \{l_1, \ldots, l_n\}, \{l_1, \ldots, l_n\} \text{ consistant avec } G(a,b)$$
$$\Rightarrow$$
$$\forall Y \text{ thèse de } E,\ Y \text{ n'infère pas } s$$

et réciproquement :

$$\forall Y \text{ thèse de } E,\ Y \text{ n'infère pas } s$$
$$\Rightarrow$$
$$\forall \{l_1, \ldots, l_n\} \text{ n'infère pas } \neg G(a,b)$$
$$\Rightarrow$$
$$\forall \{l_1, \ldots, l_n\} \text{ consistant avec } G(a,b)$$
$$\Rightarrow$$
$$\forall a \exists b G(a,b) \text{ satisfiable}$$

On en conclut donc que co-2QBF-DNF $\propto$ co-EXI-T-CNF et donc que EXI-T-CNF est Σ_2^p-complet.

- *pour ARG-T-CNF, les transformations utilisées dans le cas général ne conservent pas la forme conjonctive normale : si E et H sont sous forme CNF, alors $f(E) = E \cup \{H \rightarrow G\}$ ou $f(E) = E \cup \{\neg H\}$, et $f(H) = \neg H$ ne sont pas sous forme CNF mais sous forme quelconque (certaines formules sont en CNF, d'autres en DNF). Par contre, on peut étudier un nouveau problème EXI-T-MIXTE défini par la restriction de EXI-T à une base de formules E CNF et à une formule H DNF. L'algorithme de EXI-T s'applique à EXI-T-MIXTE qui appartient ainsi à la classe Σ_2^p. D'autre part la démonstration de complétude donnée pour EXI-T-CNF s'applique parfaitement à EXI-T-MIXTE, puisque on a la transformation suivante :*

Soit "$\forall a \exists b G(a,b)$ avec $G(a,b)$ sous-forme CNF", une instance de co-2-QBF-DNF, posons :
$E = \{a_1, \ldots, a_n, \neg a_1, \ldots, \neg a_n, G(a,b) \vee s\}$
avec s une nouvelle variable propositionnelle
(E est une base de formules CNF),
et $H = s$ (H est alors une formule DNF).

EXI-T-MIXTE est donc un problème Σ_2^p-complet. On peut alors réutiliser les transformations du cas général :

Soit "E, H", une instance de EXI-T-MIXTE
(E en CNF et H en DNF), posons :
$f(E) = E \cup \{H \rightarrow G\}$
avec G une nouvelle variable propositionnelle
(E est une base de formules CNF),
et $f(H) = G$ ($f(H)$ est alors une formule CNF).

et :

Soit "E, H", une instance de co-EXI-T-MIXTE
(E en CNF et H en DNF), posons :
$f(E) = E \cup \{\neg H\}$
(E est une base de formules CNF),
et $f(H) = \neg H$ ($f(H)$ est alors une formule CNF).

On en déduit ainsi que : ARG-T-CNF $\in \Delta_3^p$ - ($\Sigma_2^p \cup \Pi_2^p$) si $\Sigma_2^p \neq \Pi_2^p$.

En conclusion.

UNI-T-CNF est Π_2^p-complet
EXI-T-CNF est Σ_2^p-complet
ARG-T-CNF $\in \Delta_3^p$ - ($\Sigma_2^p \cup \Pi_2^p$) si $\Sigma_2^p \neq \Pi_2^p$

7.2.2 Complexité de UNI (EXI, ARG)-S-CNF

Étude d'appartenance et de complétude. *On garde exactement les mêmes démonstrations que celles de UNI-S, EXI-S et ARG-S en se ramenant à l'inférence classique sur des formules CNF (pour UNI-S-CNF), à EXI-T-CNF (pour EXI-S-CNF) et à ARG-T-CNF (pour ARG-S-CNF) :*

- *UNI-S est co-NP-complet,*
- *EXI-S-CNF est Σ_2^p-complet.*
- *ARG-S-CNF $\in \Delta_3^p$ - ($\Sigma_2^p \cup \Pi_2^p$) si $\Sigma_2^p \neq \Pi_2^p$.*

En conclusion.

UNI-S-CNF est co-NP-complet
EXI-S-CNF est Σ_2^p-complet
ARG-S-CNF $\in \Delta_3^p$ - ($\Sigma_2^p \cup \Pi_2^p$) si $\Sigma_2^p \neq \Pi_2^p$

7.2.3 Complexité de UNI (EXI, ARG)-BO-CNF

Étude d'appartenance. *Tous les algorithmes étant conservés, on obtient les mêmes résultats d'appartenance que dans le cas général :*

- *UNI-BO-CNF appartient au plus à la classe de complexité Δ_2^p,*
- *EXI-BO-CNF appartient au plus à la classe de complexité Σ_2^p,*
- *ARG-BO-CNF appartient au plus à la classe de complexité Δ_3^p.*

Étude de complétude.

- *pour UNI-BO-CNF, on peut conserver la même transformation polynomiale que pour UNI-BO, pour prouver que SAT $\propto$ UNI-BO-CNF. Par contre, pour la seconde partie de la démonstration, on va utiliser une autre transformation.*

Soit "G" une instance de UNSAT, posons :
$E = \{\neg G \rightarrow s\}$ *et* $H = s$
avec s un nouveau symbole propositionnel.

Remarquons que G est sous forme CNF, donc E et H sont aussi sous forme CNF et que la seule sous-base préférée pour l'ordre "BEST-OUT" est $\{\neg G \rightarrow s\}$.
On a alors :

G *non satisfiable*
$\Leftrightarrow$
s *est UNI-BO-CNF-inférée par* E.

Donc UNSAT $\propto$ UNI-BO-CNF et UNI-BO-CNF appartient à Δ_2^p - (NP $\cup$ co-NP) si NP $\neq$ co-NP.

- *pour EXI-BO-CNF, on fait le même raisonnement que pour EXI-T-CNF. On utilise le problème co-2QBF-DNF qui est Π_2^p-complet.*

Soit "$\forall a \exists b G(a,b)$ avec $G(a,b)$ sous-forme CNF", une instance de co-2-QBF-DNF, posons :
- $H = s$,
- $E = \{a_1, \ldots, a_n, \neg a_1, \ldots, \neg a_n, G(a,b) \vee s\}$
avec s une nouvelle variable propositionnelle.

Les sous-bases préférées pour l'ordre "BEST-OUT" de E sont de la forme $Y = \{l_1, \ldots, l_n, G(a,b) \vee s\}$ avec $l_i = a_i$ ou $\neg a_i$, $\forall i$. Toutes les remarques faites pour EXI-T-CNF restent valables ici dans le cadre des sous-bases préférées pour l'ordre "BEST-OUT".
On a alors :

$\forall a \exists b G(a,b)$ *satisfiable*
$\Rightarrow$
$\forall \{l_1, \ldots, l_n\}, \{l_1, \ldots, l_n\}$ *consistant avec* $G(a,b)$
$\Rightarrow$
$\forall Y$ *sous-base préférée pour l'ordre "BEST-OUT" de* E, Y *n'infère pas* s

et réciproquement :

$\forall Y$ *sous-base préférée pour l'ordre "BEST-OUT" de* E, Y *n'infère pas* s
$\Rightarrow$
$\forall \{l_1, \ldots, l_n\}$ *n'infère pas* $\neg G(a,b)$
$\Rightarrow$
$\forall \{l_1, \ldots, l_n\}$ *consistant avec* $G(a,b)$
$\Rightarrow$
$\forall a \exists b G(a,b)$ *satisfiable*

On en conclut donc que co-2QBF-DNF $\propto$ co-EXI-BO-CNF et donc que EXI-BO-CNF est Σ_2^p-complet.

- *pour ARG-BO-CNF, nous retrouvons les mêmes problèmes que ceux rencontrés pour ARG-T-CNF. Ici aussi, il nous faut étudier le problème EXI-BO-MIXTE, qui pour les mêmes raisons que celles évoquées pour EXI-T-MIXTE, sera Σ_2^p-complet. On garde les transformations données pour le cas général en utilisant les problèmes EXI-BO-MIXTE et co-EXI-BO-MIXTE. On prouve ainsi que ARG-BO-CNF $\in \Delta_3^p$ - ($\Sigma_2^p \cup \Pi_2^p$) si $\Sigma_2^p \neq \Pi_2^p$.*

En conclusion.

UNI-BO-CNF $\in \Delta_2^p$ - (NP $\cup$ co-NP) si NP $\neq$ co-NP
EXI-BO-CNF est Σ_2^p-complet
ARG-BO-CNF $\in \Delta_3^p$ - ($\Sigma_2^p \cup \Pi_2^p$) si $\Sigma_2^p \neq \Pi_2^p$

7.2.4 Complexité de UNI (EXI, ARG)-INCL-CNF

Étude d'appartenance. *Tous les algorithmes étant conservés, on obtient les mêmes résultats d'appartenance que dans le cas général :*

- *UNI-INCL-CNF appartient au plus à la classe de complexité* Π_2^p,
- *EXI-INCL-CNF appartient au plus à la classe de complexité* Σ_2^p,
- *ARG-INCL-CNF appartient au plus à la classe de complexité* Δ_3^p.

Étude de complétude.

- *pour UNI-INCL-CNF, on garde la même démonstration que pour UNI-INCL en utilisant UNI-T-CNF et on prouve que UNI-INCL-CNF est* Π_2^p*-complet.*
- *pour EXI-INCL-CNF, on garde la même démonstration que pour EXI-INCL en utilisant EXI-T-CNF et on prouve que EXI-INCL-CNF est* Σ_2^p*-complet.*
- *pour ARG-INCL-CNF, on garde la même démonstration que pour ARG-INCL en utilisant ARG-T-CNF (donc EXI-T-MIXTE et co-EXI-T-MIXTE) et on prouve que ARG-INCL-CNF* $\in \Delta_3^p$ *-* $(\Sigma_2^p \cup \Pi_2^p)$ *si* $\Sigma_2^p \neq \Pi_2^p$.

En conclusion.

UNI-INCL-CNF est Π_2^p*-complet*
EXI-INCL-CNF est Σ_2^p*-complet*
ARG-INCL-CNF $\in \Delta_3^p$ *-* $(\Sigma_2^p \cup \Pi_2^p)$ *si* $\Sigma_2^p \neq \Pi_2^p$

7.2.5 Complexité de UNI (EXI, ARG)-CAR-CNF

Étude d'appartenance. *Tous les algorithmes étant conservés, on obtient les mêmes résultats d'appartenance que dans le cas général, puisque les oracles utilisés (SAT, MAX-GSAT, NGSAT-CAR) restent NP-complets dans le cas de clauses :*

- *UNI-CAR-CNF appartient au plus à la classe de complexité* Δ_2^p,
- *EXI-CAR-CNF appartient au plus à la classe de complexité* Σ_2^p,
- *ARG-CAR-CNF appartient au plus à la classe de complexité* Δ_3^p.

Étude de complétude.

- *pour UNI-CAR-CNF, on peut garder la même transformation que celle du cas général puisque la base* E *et la formule* H *ainsi générées sont sous forme CNF.*
  Donc UNI-CAR-CNF est Δ_2^p*-complet.*
- *pour EXI-CAR-CNF, on fait le même raisonnement que pour EXI-T-CNF. On utilise le problème co-2QBF-DNF qui est* Π_2^p*-complet.*

Soit "$\forall a \exists b G(a,b)$ *avec* $G(a,b)$ *sous-forme CNF", une instance de co-2-QBF-DNF, posons :*
- $H = s$,
- $E = \{a_1, \ldots, a_n, \neg a_1, \ldots, \neg a_n, G(a,b) \vee s\}$
avec s *une nouvelle variable propositionnelle.*

Les thèses préférées pour la cardinalité de E *sont alors de la forme* $Y = \{l_1, \ldots, l_n, G(a,b) \vee s\}$ *avec* $l_i = a_i$ *ou* $\neg a_i$, $\forall i$. *Toutes les remarques faites pour EXI-T-CNF restent valables ici dans le cadre des thèses préférées pour la cardinalité.*
On a alors :

$$\forall a \exists b G(a,b) \text{ satisfiable}$$
$$\Rightarrow$$
$$\forall \{l_1, \ldots, l_n\}, \{l_1, \ldots, l_n\} \text{ consistant avec } G(a,b)$$
$$\Rightarrow$$
$$\forall Y \text{ thèse préférée pour la cardinalité de } E,\ Y \text{ n'infère pas } s,$$

et réciproquement :

$$\forall Y \text{ thèse préférée pour la cardinalité de } E,\ Y \text{ n'infère pas } s$$
$$\Rightarrow$$
$$\forall \{l_1, \ldots, l_n\} \text{ n'infère pas } \neg G(a,b)$$
$$\Rightarrow$$
$$\forall \{l_1, \ldots, l_n\} \text{ consistant avec } G(a,b)$$
$$\Rightarrow$$
$$\forall a \exists b G(a,b) \text{ satisfiable}$$

On en conclut donc que co-2QBF-DNF $\propto$ co-EXI-CAR-CNF et donc que EXI-CAR-CNF est Σ_2^p-complet.

- *pour ARG-CAR-CNF, comme pour ARG-T-CNF, on garde la même transformation en utilisant le problème EXI-CAR-MIXTE qui est Σ_2^p-complet pour les mêmes raisons que celles évoquées lors de la démonstration de complétude de EXI-T-MIXTE. On prouve ainsi que ARG-CAR-CNF $\in \Delta_3^p$ et est Σ_2^p-difficile. Tout comme pour ARG-CAR, nous ne sommes pas parvenus à prouver que ARG-CAR-CNF est Π_2^p -difficile.*

En conclusion.

UNI-CAR-CNF est Δ_2^p-complet
EXI-CAR-CNF est Σ_2^p-complet
ARG-CAR-CNF $\in \Delta_3^p$ et est Σ_2^p -difficile

7.2.6 Complexité de UNI (EXI, ARG)-LEX-CNF

Étude d'appartenance. *Comme pour UNI (EXI, ARG)-CAR, tous les algorithmes sont conservés, et on obtient les mêmes résultats d'appartenance que dans le cas général :*

- *UNI-LEX-CNF appartient au plus à la classe de complexité Δ_2^p,*
- *EXI-LEX-CNF appartient au plus à la classe de complexité Σ_2^p,*
- *ARG-LEX-CNF appartient au plus à la classe de complexité Δ_3^p.*

Étude de complétude.

- *pour UNI-LEX-CNF, on peut garder la même transformation que celle du cas général qui permet de passer de UNI-CAR-CNF à UNI-LEX-CNF, puis en utilisant le fait que UNI-CAR-CNF est Δ_2^p-complet, on obtient que UNI-LEX-CNF est Δ_2^p-complet.*
- *pour EXI-LEX-CNF, on peut garder la même transformation que celle du cas général qui permet de passer de EXI-CAR-CNF à EXI-LEX-CNF, puis en utilisant les résultats sur EXI-CAR-CNF, on obtient que EXI-LEX-CNF est Σ_2^p-complet.*
- *pour ARG-LEX-CNF, on peut garder la même transformation que celle du cas général qui permet de passer de ARG-CAR-CNF à ARG-LEX-CNF, puis en utilisant les résultats sur ARG-CAR-CNF, on prouve que ARG-LEX-CNF $\in \Delta_3^p$ et est Σ_2^p-difficile. Par contre, comme pour ARG-LEX, nous ne sommes pas parvenus à prouver que ARG-LEX-CNF est Π_2^p -difficile.*

En conclusion.

UNI-LEX-CNF est Δ_2^p-complet
EXI-LEX-CNF est Σ_2^p-complet
ARG-LEX-CNF $\in \Delta_3^p$ et est Σ_2^p -difficile

7.2.7 Complexité de UNI (EXI, ARG)-E-CNF

Il n'existe pas à notre connaissance de travaux sur ces relations d'inférence dans le cas de formules CNF. Tout reste donc à faire et nous laissons aux spécialistes de la logique des défauts le soin de s'y plonger !

7.3 Cas d'une base de clauses de Horn

Ici, on part de l'hypothèse que la base étudiée $(E, <)$ est un ensemble de clauses de Horn et que la formule traitée H est une clause de Horn, c'est-à-dire que chaque formule est de la forme :

- *soit $A_1 \ldots A_n \rightarrow B$*
- *soit $A_1 \ldots A_n \rightarrow$ (avec les A_i et B qui sont des littéraux positifs).*

On constate rapidement que tous les problèmes passent dans la classe de complexité inférieure. En effet, les problèmes GSAT et UNGSAT sont remplacés par SAT-HORN et UNSAT-HORN qui sont polynomiaux, ce qui implique que les problèmes d'inférence classique deviennent polynomiaux eux aussi. On a en effet :

- $B \vdash H \Leftrightarrow B \cup \{\neg H\}$ *inconsistant* $\Leftrightarrow B \cup \{\neg H\}$ *non satisfiable,*
- $B \not\vdash H \Leftrightarrow B \cup \{\neg H\}$ *consistant* $\Leftrightarrow B \cup \{\neg H\}$ *satisfiable.*

7.3.1 Complexité de UNI (EXI, ARG)-T-HORN

Étude d'appartenance. *Tous les algorithmes sont conservés et comme la complexité des problèmes d'inférence a diminué, on a alors :*

- *UNI-T-HORN appartient au plus à la classe de complexité co-NP,*
- *EXI-T-HORN appartient au plus à la classe de complexité NP,*
- *ARG-T-HORN appartient au plus à la classe de complexité Δ_2^p.*

Étude de complétude.

- *pour UNI-T-HORN, on va utiliser la même méthode que pour SBR-HORN (voir annexe A), c'est-à-dire prouver que SAT $\propto$ co-UNI-T-HORN*[6].

Soit "C" une instance de SAT, avec $C = \{C_j\}$ pour $j = 1 \ldots m$ ensemble de clauses quelconques à satisfaire, et $V(C) = \{x_1, \ldots, x_n\}$ l'ensemble des variables propositionnelles de C, posons :
- $H = \neg s$.
- $E = \{p, x_1, \ldots, x_n, y_1, \ldots, y_n, \neg z_1, \ldots, \neg z_n, \neg s\}$ avec $y_1, \ldots, y_n$, $z_1, \ldots, z_n$, s nouveaux symboles de variables propositionnelles, et avec la formule
$p = (z_1 \ldots z_n \rightarrow s) \wedge \bigwedge_{i=1\ldots n}((\neg x_i \vee \neg y_i) \wedge (y_i \rightarrow z_i) \wedge (x_i \rightarrow z_i)) \wedge \bigwedge_{j=1\ldots m} C_j[y]$,
avec la notation suivante : $C_j[y]$ = la clause C_j dans laquelle on remplace tous les x_i positifs par des $\neg y_i$.

[6] Attention, ici il s'agit du problème SAT portant sur la satisfiabilité d'ensembles de clauses quelconques, pas de clauses de Horn !

Remarques préliminaires :

- *Il est inutile de définir un ordre entre les formules de la base.*
- *L'idée de cette démonstration est issue d'un article de Eiter et Gottlob (voir [EG92]).*
- *Le but est de transformer un problème portant sur des clauses quelconques en un problème portant sur des clauses de Horn. Il faut donc transformer les clauses initiales en clauses de Horn. Pour cela, on introduit une nouvelle variable propositionnelle y_i pour chaque x_i de C, y_i jouera le rôle de $\neg x_i$. Cela pose alors un problème de cohérence : il ne faut pas que nous puissions avoir x_i et y_i ayant la même valeur de vérité. Nous introduisons alors une nouvelle variable propositionnelle z_i qui sera vraie quand la valeur de vérité de x_i sera différente de la valeur de vérité de y_i ; ce qui revient à dire que z_i est le "ou-exclusif" de x_i et de y_i. La dernière variable rajoutée s sera vraie quand tous les z_i seront vrais, donc quand pour tous les x_i et y_i, on aura valeur-vérité (x_i) $\neq$ valeur-vérité (y_i).*
- *En transformant C_j en $C_j[y]$, on obtient une clause de Horn, quel que soit j.*
- *On a alors la propriété suivante :*

  Propriété 7.3.1 *Si la valeur de vérité de x_i est différente de la valeur de vérité de y_i $\forall i$ alors C sera satisfaite ssi $C[y]$ est satisfaite.*

- *De plus, si on calcule les sous-bases maximales consistantes Y de E, on trouve que :*

  Propriété 7.3.2 *Si C est satisfiable alors il existe un Y tel que $Y \vdash z_i$ ($\forall i$) et $Y \vdash s$. Un tel Y n'inférera pas $\neg s$.*

- *Nous nous trouvons d'autre part dans une optique SYNTAX-BASED, ce qui signifie que, bien que la formule p soit un ensemble de clauses de Horn, nous considérons p dans son intégralité.*
- *La seule sous-base maximale consistante inférant s est une sous-base contenant la formule p, puisque la variable s n'apparaît que dans p.*

Prouvons maintenant que SAT(C) $\Leftrightarrow$ co-UNI-T-HORN (E, H).

- *SAT implique co-UNI-T-HORN :*

  C satisfiable
  $\Leftrightarrow$
  il existe une interprétation Φ de $\{x_1, \ldots, x_n\}$ dans laquelle C est vraie
  $\Rightarrow$
  Posons $W = \{x_i$ tels que $\Phi(x_i)$ soit vrai$\} \cup \{yj$ tels que $\Phi(xj)$ soit faux$\}$.
  $\forall i\ \Phi(x_i) \neq \Phi(y_i)$ puisque $y_i = \neg x_i$.
  On a donc : $\forall j\ C_j[y]$ satisfaite puisque C_j est satisfaite et que x_i et y_i n'ont pas la même valeur de vérité $\forall i$.
  On peut ainsi en déduire que p est satisfaite, donc que $Y = \{p\} \cup W$ est maximale consistante dans E[7].
  $\Rightarrow$
  il existe une sous-base maximale consistante Y de E telle que Y infère tous les z_i et infère s.
  $\Rightarrow$
  il existe une sous-base maximale consistante Y de E telle que Y n'infère pas $\neg s$.
  $\Leftrightarrow$
  co-UNI-T-HORN (E, H)

- *co-UNI-T-HORN implique SAT :*

  co-UNI-T-HORN (E, H)
  $\Leftrightarrow$
  il existe Y sous-base maximale consistante de E
  telle que Y n'infère pas $\neg s$.
  Donc Y est inconsistante avec $\neg s$, puisque $\neg s \in E$, donc Y infère s.

[7] Cette étape du raisonnement reste toujours la même pour les démonstrations sur les problèmes à clauses de Horn montrant l'existence d'une transformation polynomiale entre SAT et le problème traité. Elle ne sera donc pas répétée.

Donc Y contient p, donc Y contient un certain ensemble $\{x_i\} \cup \{y_i\}$ tel que dans une interprétation satisfaisant Y donc p, on aura :
- $\forall i$ la valeur de vérité de x_i est différente de celle de y_i,
- les $C_j[y]$ seront satisfaites.
Posons comme interprétation Φ :
$\Phi(x_i)$ = vrai si x_i appartient à Y et faux sinon
$\Rightarrow$
Dans l'interprétation Φ qui satisfait Y, les $C_j[y]$ sont satisfaites et comme les x_i et les y_i n'ont pas la même valeur de vérité
$\Rightarrow$
les C_j sont satisfaites elles aussi
$\Leftrightarrow$
C satisfiable

Le problème SAT se ramène donc polynomialement à co-UNI-T-HORN. Et on en déduit que le problème co-UNI-T-HORN est NP-complet, et donc que UNI-T-HORN est co-NP-complet.

- *pour EXI-T-HORN, on va faire le même raisonnement que celui pour co-UNI-T-HORN, c'est-à-dire prouver que SAT $\propto$ EXI-T-HORN.*

Soit "C" une instance de SAT, avec $C = \{C_j\}$ pour $j = 1 \ldots m$ ensemble de clauses quelconques à satisfaire, soit $V(C) = \{x_1, \ldots, x_n\}$ l'ensemble des variables propositionnelles de C, posons :
- $H = s$.
- $E = \{p, x_1, \ldots, x_n, y_1, \ldots, y_n, \neg z_1, \ldots, \neg z_n, \neg s\}$ avec $y_1, \ldots, y_n$, $z_1, \ldots, z_n$, s nouveaux symboles de variables propositionnelles, et avec la formule
$p = (z_1 \ldots z_n \rightarrow s) \wedge \bigwedge_{i=1\ldots n}((\neg x_i \vee \neg y_i) \wedge (y_i \rightarrow z_i) \wedge (x_i \rightarrow z_i)) \wedge \bigwedge_{j=1\ldots m} C_j[y]$,
avec la notation suivante : $C_j[y]$ = la clause C_j dans laquelle on remplace tous les x_i positifs par des $\neg y_i$.

Les remarques préliminaires fournies pour la démonstration de UNI-T-HORN restent valables. Prouvons maintenant que SAT(C) $\Leftrightarrow$ EXI-T-HORN (E, H).

- *SAT implique EXI-T-HORN :*

C satisfiable
$\Rightarrow$
de la même façon que pour co-UNI-T-HORN, il existe une sous-base maximale consistante Y de E définie par
$Y = \{p\} \cup \{x_i$ *tels que* $\Phi(x_i)$ *soit vrai*$\} \cup \{yj$ *tels que* $\Phi(xj)$ *soit faux*$\}$
et telle que Y infère tous les z_i et infère s.
$\Leftrightarrow$
EXI-T-HORN (E, H)

- *EXI-T-HORN implique SAT :*

EXI-T-HORN (E, H)
$\Leftrightarrow$
il existe Y sous-base maximale consistante de E telle que Y infère s.
Donc Y contient p, donc Y contient un certain ensemble $\{x_i\} \cup \{y_i\}$ tel que dans une interprétation satisfaisant Y donc p, on aura :
-$\forall i$ la valeur de vérité de x_i est différente de celle de y_i,
- les $C_j[y]$ seront satisfaites.
Posons comme interprétation Φ :
$\Phi(x_i)$ = vrai si x_i appartient à Y et faux sinon

$\Rightarrow$

Dans l'interprétation Φ qui satisfait Y, les $C_j[y]$ sont satisfaites et comme les x_i et les y_i n'ont pas la même valeur de vérité

$\Rightarrow$

les C_j sont satisfaites elles aussi

$\Leftrightarrow$

C satisfiable

Le problème SAT se ramène donc polynomialement à EXI-T-HORN. Et on en déduit que le problème EXI-T-HORN est NP-complet.

- *pour ARG-T-HORN, on ne peut pas garder la même démonstration que pour ARG-T, en effet les transformations polynomiales utilisées pour passer de EXI-T à ARG-T et de co-EXI-T à ARG-T ne conservent pas la "Hornitude" des formules utilisées :*
  - *$H \rightarrow G$ ne sera une clause de Horn que si H est un littéral positif (rappelons que G est une nouvelle variable propositionnelle),*
  - *$\neg H$ ne sera une clause de Horn que si H est de la forme $a \rightarrow b$ avec a et b 2 littéraux positifs.*

Il va donc falloir étudier un nouveau problème appelé EXI-T-HORN-POSITIF :

> ***instance*** *: E une base de clauses de Horn, H un littéral positif,*
> ***question*** *: H est-elle conséquence faible de E en utilisant les thèses définies à partir de E ?*

L'algorithme à appliquer à ce problème est identique à celui de EXI-T et EXI-T-HORN :

1. *deviner un sous-ensemble Y de E*
2. *vérifier que Y est une thèse de E*
3. *vérifier que Y infère H.*

Cet algorithme est non déterministe (à cause du "deviner") polynomial, puisque E est une base de clauses de Horn et que H est un littéral positif (donc une clause de Horn). Le problème EXI-T-HORN-POSITIF appartient à la classe de complexité NP.
EXI-T-HORN-POSITIF est-il NP-complet ? Il suffit de se ramener à la démonstration de complétude faite pour EXI-T-HORN et de constater qu'elle reste valable en intégralité pour EXI-T-HORN-POSITIF puisque on posait : $H = s$ (s étant un littéral positif). Nous en concluons que EXI-T-HORN-POSITIF est NP-complet.
À partir de là, nous pouvons reprendre les transformations polynomiales définies pour ARG-T et les appliquer respectivement à EXI-T-HORN-POSITIF et co-EXI-T-HORN-POSITIF. Ce qui nous donne :

Soit "E, H" une instance de EXI-T-HORN-POSITIF, posons : $f(E) = E \cup \{H \rightarrow G\}$ *et* $f(H) = G$ *(G nouveau symbole propositionnel)*

On a bien $f(E)$ qui est une base de clauses de Horn et $f(H)$ qui est une clause de Horn. Puis avec :

Soit "E, H" une instance de co-EXI-T-HORN-POSITIF, posons : $f(E) = E \cup \{\neg H\}$ *et* $f(H) = \neg H$

On a ici aussi $f(E)$ qui est une base de clauses de Horn et $f(H)$ qui est une clause de Horn.
Ainsi, on obtient que ARG-T-HORN appartient à la classe de complexité Δ_2^p - (NP $\cup$ co-NP) si NP $\neq$ co-NP.

En conclusion.

UNI-T-HORN est co-NP-complet
EXI-T-HORN est NP-complet
ARG-T-HORN $\in \Delta_2^p$ - (NP $\cup$ co-NP) si NP $\neq$ co-NP

7.3.2 Complexité de UNI (EXI, ARG)-S-HORN

Étude d'appartenance et de complétude. *Toutes les démonstrations sont conservées et comme la complexité de l'inférence classique sur clauses de Horn (pour UNI-S-HORN), de EXI-T-HORN (pour EXI-S-HORN) et de ARG-T-HORN (pour ARG-S-HORN) a diminué, on a alors :*

- *UNI-S-HORN est désormais de complexité P,*
- *EXI-S-HORN est NP-complet.*
- *ARG-S-HORN appartient à la classe de complexité Δ_2^p - (NP $\cup$ co-NP) si NP $\neq$ co-NP.*

En conclusion.

UNI-S-HORN est P
EXI-S-HORN est NP-complet
ARG-S-HORN $\in \Delta_2^p$ - (NP $\cup$ co-NP) si NP $\neq$ co-NP

7.3.3 Complexité de UNI (EXI, ARG)-BO-HORN

Étude d'appartenance. *Tous les algorithmes sont conservés et comme la complexité des problèmes d'inférence a diminué, on a alors :*

- *UNI-BO-HORN est désormais de complexité P,*
- *EXI-BO-HORN appartient au plus à la classe de complexité NP,*
- *ARG-BO-HORN appartient au plus à la classe de complexité Δ_2^p.*

Étude de complétude.

- *pour EXI-BO-HORN, on va utiliser la même méthode que pour EXI-T-HORN, c'est-à-dire prouver que SAT $\propto$ EXI-BO-HORN.*

Soit "C" une instance de SAT, avec $C = \{C_j\}$ pour $j = 1 \ldots m$ ensemble de clauses quelconques à satisfaire, soit $V(C) = \{x_1, \ldots, x_n\}$ l'ensemble des variables propositionnelles de C, posons :
- $H = s$.
- $E = \{p, x_1, \ldots, x_n, y_1, \ldots, y_n, \neg z_1, \ldots, \neg z_n, \neg s\}$ avec $y_1, \ldots, y_n$, $z_1, \ldots, z_n$, s nouveaux symboles de variables propositionnelles, et avec la formule
$p = (z_1 \ldots z_n \rightarrow s) \wedge \bigwedge_{i=1\ldots n}((\neg x_i \vee \neg y_i) \wedge (y_i \rightarrow z_i) \wedge (x_i \rightarrow z_i)) \wedge \bigwedge_{j=1\ldots m} C_j[y]$,
avec la notation suivante : $C_j[y]$ = la clause C_j dans laquelle on remplace tous les x_i positifs par des $\neg y_i$. Il est inutile de définir un ordre entre les formules de la base.

Les remarques préliminaires fournies pour UNI-T-HORN restent valables ici. On constate en plus que :

- *Comme il n'y a pas d'ordre entre les formules de E, cela signifie que pour toute sous-base consistante Y de E telle que $Y \neq E$, $a(Y) = 1$.*
- *E est inconsistant ; en effet pour que E soit consistant, il faut que tous les x_i, tous les y_i soient à vrai, tous les z_i et s soient à faux, et que p soit à vrai, or pour que p soit vrai, cela impose que soit x_i, soit y_i soit faux, $\forall i$; on a donc une incohérence.*

- *La seule sous-base consistante préférée dans l'ordre "BEST-OUT"*[8] *inférant s est une sous-base contenant la formule p et un ensemble de x_i et y_i tel que les valeurs de vérité des x_i soient différentes des valeurs de vérité des y_i, puisque la variable s n'apparaît que dans p.*

Prouvons maintenant que SAT(C) $\Leftrightarrow$ EXI-BO-HORN (E, H).

- *SAT implique EXI-BO-HORN :*

$$C \text{ satisfiable}$$
$$\Rightarrow$$
il existe une sous-base consistante Y de E telle que Y infère tous les z_i et infère s.
$$\Leftrightarrow$$
EXI-BO-HORN (E, H)

- *EXI-BO-HORN implique SAT :*

EXI-BO-HORN (E, H)
$$\Leftrightarrow$$
il existe Y sous-base consistante préférée dans l'ordre "BEST-OUT" de E telle que Y infère s. Donc Y contient p, donc Y contient un ensemble $\{x_i\} \cup \{y_i\}$ tel que dans une interprétation satisfaisant Y donc p, on aura :
- $\forall i$ la valeur de vérité de x_i est différente de celle de y_i,
- les $C_j[y]$ seront satisfaites.
Posons comme interprétation Φ : $\Phi(x_i) =$ vrai si $x_i \in Y$ et faux sinon
$$\Rightarrow$$
Dans l'interprétation Φ qui satisfait Y, les $C_j[y]$ sont satisfaites et comme les x_i et les y_i n'ont pas la même valeur de vérité
$$\Rightarrow$$
les C_j sont satisfaites elles aussi
$$\Leftrightarrow$$
C *satisfiable*

Le problème SAT se ramène donc polynomialement à EXI-BO-HORN. Et on en déduit que le problème EXI-BO-HORN est NP-complet.

- *pour ARG-BO-HORN, on ne peut pas garder la même démonstration que pour ARG-BO. En effet, comme pour ARG-T, les transformations polynomiales utilisées pour passer de EXI-BO à ARG-BO et de co-EXI-BO à ARG-BO ne conservent pas la "Hornitude" des formules utilisées (voir remarques sur la démonstration de ARG-T-HORN en section 7.3.1).*
Il va donc falloir étudier un nouveau problème appelé EXI-BO-HORN-POSITIF :

  ***instance** : E une base de clauses de Horn, H un littéral positif,*
  ***question** : H est-elle conséquence faible de $(E, <)$ en utilisant l'ordre "BEST OUT" sur les sous-bases consistantes ?*

Nous démontrons comme pour EXI-T-HORN-POSITIF (voir section 7.3.1) que EXI-BO-HORN-POSITIF est NP-complet. À partir de là, nous pouvons reprendre les transformations polynomiales définies pour ARG-BO et les appliquer respectivement à EXI-BO-HORN-POSITIF et co-EXI-BO-HORN-POSITIF :

Soit "$(E, <)$, H" une instance de EXI-BO-HORN-POSITIF, posons : *- $f(H) = G$ (G nouveau symbole propositionnel),* *- $f((E, <)) = (E \cup \{H \rightarrow G\}, <)$* *avec $H \rightarrow G$ constituant seule la première strate.*

On a bien $f((E, <))$ qui est une base de clauses de Horn et $f(H)$ qui est une clause de Horn. Puis, avec :

Soit "$(E, <), H$" une instance de co-EXI-BO-HORN-POSITIF, posons : *- $f(H) = \neg H$,* *- $f((E, <)) = (E \cup \{\neg H\}, <)$* *avec $\neg H$ constituant seule la dernière strate.*

[8] Étant donné qu'il n'y a qu'une seule strate dans E, toute sous-base consistante est préférée pour l'ordre "BEST-OUT".

On a ici aussi $f((E,<))$ qui est une base de clauses de Horn et $f(H)$ qui est une clause de Horn. Ainsi, on obtient que ARG-BO-HORN appartient à la classe de complexité Δ_2^p - (NP $\cup$ co-NP) si NP $\neq$ co-NP.

En conclusion.

UNI-BO-HORN est P
EXI-BO-HORN est NP-complet
ARG-BO-HORN $\in \Delta_2^p$ - (NP $\cup$ co-NP) si NP $\neq$ co-NP

7.3.4 Complexité de UNI (EXI, ARG)-INCL-HORN

Étude d'appartenance. *Tous les algorithmes sont conservés et comme la complexité des problèmes d'inférence a diminué, on a alors :*

- *UNI-INCL-HORN appartient au plus à la classe de complexité co-NP,*
- *EXI-INCL-HORN appartient au plus à la classe de complexité NP,*
- *ARG-INCL-HORN appartient au plus à la classe de complexité Δ_2^p.*

Étude de complétude.

- *pour UNI-INCL-HORN, on garde exactement la même démonstration que pour UNI-INCL, mais comme maintenant le problème UNI-T-HORN est co-NP-complet, on obtient que UNI-INCL-HORN est co-NP-complet.*
- *pour EXI-INCL-HORN, on garde exactement la même démonstration que pour EXI-INCL, mais comme maintenant le problème EXI-T-HORN est NP-complet, on obtient que EXI-INCL-HORN est NP-complet.*
- *pour ARG-INCL-HORN, on garde exactement la même démonstration que pour ARG-INCL, mais comme maintenant le problème ARG-T-HORN utilise les problèmes co-EXI-T-HORN-POSITIF et EXI-T-HORN-POSITIF qui sont respectivement co-NP-complet et NP-complet, on obtient donc que ARG-INCL-HORN appartient à la classe de complexité Δ_2^p - (NP $\cup$ co-NP) si NP $\neq$ co-NP.*

En conclusion.

UNI-INCL-HORN est co-NP-complet
EXI-INCL-HORN est NP-complet
ARG-INCL-HORN $\in \Delta_2^p$ - (NP $\cup$ co-NP) si NP $\neq$ co-NP

7.3.5 Où l'on parle de multi-ensembles !

Lors de l'étude du cas particulier des clauses de Horn pour les problèmes utilisant la cardinalité, nous nous sommes rendus compte qu'il était indispensable d'introduire une nouvelle notion : les multi-ensembles. *En effet, dans les démonstrations proposées dans les sections 7.3.6 et 7.3.7, nous allons être amenés à répéter des formules de notre base initiale ce qui n'est pas compatible avec la notion d'ensemble. Nous définissons alors :*

Définition 7.3.1 *E est un multi-ensemble de formules propositionnelles ssi :*

- *E est un ensemble de couples (Φ_i, p_i) avec :*
  - *Φ_i une formule propositionnelle,*
  - *p_i est le poids de Φ_i, c'est-à-dire le nombre d'occurrences de Φ_i ;*

- *il ne peut pas y avoir dans E deux couples (Φ_i, p_i) et (Φ_j, p_j) tels que $\Phi_i = \Phi_j$ et $p_i \neq p_j$.*

Définition 7.3.2 *$(E, <) = E_1 \cup \ldots \cup E_n$ est un multi-ensemble stratifié de formules propositionnelles ssi :*

- *$(E, <)$ est un ensemble stratifié de couples (Φ_i, p_i) avec :*
  - *Φ_i une formule propositionnelle,*
  - *p_i est le poids de Φ_i, c'est-à-dire le nombre d'occurrences de Φ_i ;*
- *il ne peut pas y avoir dans $(E, <)$ deux couples (Φ_i, p_i) et (Φ_j, p_j) tels que $\Phi_i = \Phi_j$ et $p_i \neq p_j$.*

Définition 7.3.3 *Les multi-ensembles étant définis comme des ensembles de couples, nous conservons la notion d'inclusion classique : soit E un multi-ensemble, S est un "sous-ensemble" de E ssi $S \subseteq E$. Idem pour le cas stratifié.*

Exemples : *Soit $E = \{(A, 3), (B, 1), (C, 2)\}$, E est un multi-ensemble correspondant à la collection de formules propositionnelles suivante : $\{A, A, A, B, C, C\}$. Soit $E = \{(A, 3), (B, 1), (A, 2)\}$, E n'est pas un multi-ensemble. Soit $E = \{(A, 3), (B, 1), (A, 3)\}$, E est le multi-ensemble : { (A,3), (B,1)}.*

Cette notion de multi-ensemble est essentielle en cardinalité car dire qu'une formule Φ peut être répétée p fois change totalement le calcul des sous-bases préférées pour la cardinalité.

Définition 7.3.4 *Soit S un sous-ensemble du multi-ensemble E, on note $|S|^*$ et on appelle cardinal de S $|S|^* = \sum_i p_i$ tel que $(\Phi_i, p_i) \in S$.*

Définition 7.3.5 *On note E^* l'ensemble de formules propositionnelles issu du multi-ensemble E : $E^* = \{\Phi_i$ telle que $(\Phi_i, p_i) \in E\}$*

Définition 7.3.6 *Soit S un sous-ensemble du multi-ensemble E, S est une sous-base consistante de E ssi S^* est une sous-base consistante de E^*.*

Soit S un sous-ensemble du multi-ensemble E, S est une thèse de E ssi S^* est une thèse de E^*.

Définition 7.3.7 *Soit S un sous-ensemble du multi-ensemble E, S est une sous-base préférée pour la cardinalité de E ssi :*

- *S est une sous-base consistante de E,*
- *il n'existe pas S' sous-base consistante de E telle que $|S|^* < |S'|^*$.*

Définition 7.3.8 *Soit S un sous-ensemble du multi-ensemble $(E, <) = E_1 \cup \ldots \cup E_n$, S est une sous-base préférée pour l'ordre lexicographique de $(E, <)$ ssi $\forall k = 1 \ldots n$, $S_1 \cup \ldots \cup S_k$ est une sous-base préférée pour la cardinalité du multi-ensemble $E_1 \cup \ldots \cup E_k$.*

Nous pouvons alors redéfinir les problèmes d'inférence non-monotone liés à la cardinalité :

Définition 7.3.9 *Les problèmes d'inférence non-monotone UNI (EXI, ARG)-CAR sur des multi-ensembles sont :*

> ***instance :*** *E un multi-ensemble de formules propositionnelles, H une formule propositionnelle,*
> ***question :*** *H est-elle une conséquence forte (faible, argumentative) de E en utilisant l'ordre basé sur la cardinalité sur les thèses de E ?*

Ces problèmes sont notés UNI (EXI, ARG)-CAR-ME.

Définition 7.3.10 *Les problèmes d'inférence non-monotone UNI (EXI, ARG)-LEX sur des multi-ensembles sont :*

> ***instance :*** *E un multi-ensemble de formules propositionnelles, H une formule propositionnelle,*
> ***question :*** *H est-elle une conséquence forte (faible, argumentative) avec préférences de E en utilisant l'ordre lexicographique sur les thèses de E ?*

Ces problèmes sont notés UNI (EXI, ARG)-LEX-ME.

Nous avons pu jusqu'à présent nous passer de la notion de multi-ensemble. Notons toutefois que tous les résultats obtenus jusqu'ici sur des ensembles sont conservés en intégralité quand on travaille sur des multi-ensembles. En effet, le cas des ensembles est une restriction du cas des multi-ensembles donc toutes les preuves de complétude restent valables, quant aux preuves d'appartenance, on constate que les seuls algorithmes dépendants de cet aspect sont les algorithmes utilisant la cardinalité et ils peuvent être modifiés comme suit :

- *algorithme pour UNI-CAR-ME (recherche dichotomique pour conserver un algorithme polynomial) :*

  1. $k \leftarrow 0$
  2. nfmax $\leftarrow$ ($\sum_i p_i$ tel que $(\Phi_i, p_i) \in E$)
  3. si MAX-GSAT-ME (E, nfmax) alors $k \leftarrow$ nfmax
  sinon
  4. nfmin $\leftarrow 0$
  5. tant que ((nfmax - nfmin) > 1) faire
  6. $k \leftarrow$ (nfmax + nfmin) div 2 (div = division entière *)*
  7. si MAX-GSAT-ME (E, k) alors
  8. nfmin $\leftarrow k$
  9. sinon nfmax $\leftarrow k$
  fin si
  fin tant que
  fin si
  10. si NGSAT-CAR-ME (E,H,k) alors
  11. échec $\leftarrow$ vrai
  12. sinon échec $\leftarrow$ faux
  fin si
  13. vérifier que l'on n'a pas échec (échec = faux)

- *algorithme pour MAX-GSAT-ME :*

  1. deviner une interprétation M
  2. $p \leftarrow 0$
  3. pour chaque formule G de E faire
  4. si M satisfait G alors $p \leftarrow p + p_G$ (p_G = poids de G *)*
  fin si
  fin pour
  5. vérifier que $p \geq k$

- *algorithme pour NGSAT-CAR-ME :*

  1. deviner une interprétation M
  2. vérifier qu'elle satisfait k formules de E
  (c'est à dire :*
  a. $k' \leftarrow 0$
  b. pour chaque formule G de E faire
  c. si M satisfait G alors
  d. $k' \leftarrow k' + p_G$ (p_G = poids de G *)*

fin si
fin pour
e. vérifier que $k' = k$
**)*
3. vérifier que M *ne satisfait pas* H

- *algorithme pour EXI-CAR-ME :*

1. deviner un sous-ensemble Y *de* E
*(*vérifier que* Y *est une thèse préférée pour la cardinalité de* E**)*
2. échec $\leftarrow$ *faux*
3. $k \leftarrow |Y|^*$
4. S'il existe une sous-base consistante Z *de* E *telle que* $|Z|^* > k$ *alors*
5. échec $\leftarrow$ *vrai*
fin si
6. si non échec alors
7. vérifier que Y *infère* H
fin si

- *algorithme pour MAX-GSAT-STRICT-ME :*

1. deviner une interprétation M
2. $p \leftarrow 0$
3. pour chaque formule G *de* E *faire*
4. si M *satisfait* G *alors* $p \leftarrow p + p_G$ *(** p_G = poids de G *)*
fin si
fin pour
5. vérifier que $p > k$

- *algorithmes pour UNI-LEX-ME (recherche dichotomique pour conserver un algorithme polynomial) :*

1. $E' \leftarrow \{(H \rightarrow G, 1)\} \cup E \cup \{(\neg G, 1)\}$
2. $k \leftarrow < 0, 0, \ldots, 0 >$ *(*vecteur de dimension* n *= nombre de strates dans* E' **)*
3. pour $ns \leftarrow 1$ *à* n *faire*
4. nfmax $\leftarrow$ *(*$\sum_i p_i$ *tel que* $(\Phi_i, p_i) \in E'_{ns}$*)*
5. $k[ns] \leftarrow$ *nfmax*
6. si (non MAX-GSAT-ARRAY-ME (E'*,* k*)) alors*
7. nfmin $\leftarrow 0$
8. $k[ns] \leftarrow 0$
9. tant que ((nfmax - nfmin) > 1*) faire*
10. $k[ns] \leftarrow$ *(nfmax + nfmin) div 2 (* div = division entière *)*
11. si MAX-GSAT-ARRAY-ME (E'*,* k*) alors*
12. nfmin $\leftarrow k[ns]$
13. sinon nfmax $\leftarrow k[ns]$
fin si
fin tant que
fin si
fin pour
14. vérifier que $k[n]$ *est différent de 1*

ou bien :

(calcul du vecteur* k **)*
1. $k \leftarrow < 0, 0, \ldots, 0 >$ *(* vecteur de dimension* n *(nombre de strates dans* E*)*)*
2. pour $ns \leftarrow 1$ *à* n *faire*
3. nfmax $\leftarrow$ *(*$\sum_i p_i$ *tel que* $(\Phi_i, p_i) \in E_{ns}$*)*
4. $k[ns] \leftarrow$ *nfmax*

5. *si (non MAX-GSAT-ARRAY-ME (E, k)) alors*
6. *nfmin* $\leftarrow$ *0*
7. $k[ns] \leftarrow$ *0*
8. *tant que ((nfmax - nfmin)* $>$ *1) faire*
9. $k[ns] \leftarrow$ *(nfmax + nfmin) div 2 (* div* $=$ *division entière *)*
10. *si MAX-GSAT-ARRAY-ME (E, k) alors*
11. *nfmin* $\leftarrow k[ns]$
12. *sinon nfmax* $\leftarrow k[ns]$
fin si
fin tant que
fin si
fin pour
13. si NGSAT-LEX-ME (E,H,k) alors
14. *échec* $\leftarrow$ *vrai*
15. sinon échec $\leftarrow$ *faux*
fin si
16. vérifier que l'on n'a pas échec (échec = faux)

- *algorithme pour MAX-GSAT-ARRAY-ME :*

1. deviner une interprétation M
2. $i \leftarrow$ *1 (*strate la plus prioritaire*)*
3. échec $\leftarrow$ *faux*
4. tant que $i \leq n$ *(nombre de strates de* E*) et (non échec) faire*
5. $p \leftarrow$ *0*
6. *pour chaque formule* G *de* $(E, <)$ *faire*
7. *si* M *satisfait* G *alors* $p \leftarrow p + p_G$ *(*p_G $=$ *poids de* G *)*
fin si
fin pour
8. *si* $p \geq k[i]$ *alors*
9. $i \leftarrow i + 1$
10. *sinon échec* $\leftarrow$ *vrai*
fin si
fin tant que
11. vérifier que l'on n'a pas échec (échec = faux)

- *algorithme pour NGSAT-LEX-ME :*

1. deviner une interprétation M
2. vérifier qu'elle satisfait $k[i]$ formules de chaque strate i de E
(c'est à dire :*
a. $k' \leftarrow < 0, 0, \ldots, 0 >$ *(*k' *de même dimension* n *que* k*)*
b. *pour chaque strate* i *de 1 à* n *faire*
c. *pour chaque formule* G *de la strate* i *faire*
d. *si* M *satisfait* G *alors*
e. $k'[i] \leftarrow k'[i] + p_G$ *(*p_G $=$ *poids de* G *)*
fin si
fin pour
fin pour
f. *vérifier que* $k' = k$
)
3. vérifier que M *ne satisfait pas* H

- *algorithme pour EXI-LEX-ME :*

1. deviner un sous-ensemble Y *de* $(E, <)$

*(*vérifier que Y est une thèse préférée lexicographiquement de $(E, <)$*)*
2. échec ← faux
*3. $i \leftarrow 1$ (*strate la plus prioritaire*)*
4. tant que non échec et $i < n + 1$ faire
5. $k \leftarrow |Y_1|^* + |Y_2|^* + \ldots + |Y_i|^*$
6. *S'il existe une sous-base consistante Z de $E_1 \cup E_2 \cup \ldots \cup E_i$ telle que $|Z|^* > k$ alors*
7. *échec ← vrai*
8. *sinon $i \leftarrow i + 1$*
fin si
fin tant que
9. si non échec alors
10. *vérifier que Y infère H*
fin si

Tous ces algorithmes donnent la même borne maximum pour la complexité que celle obtenue par les algorithmes correspondant au cas des ensembles.

7.3.6 Complexité de UNI (EXI, ARG)-CAR-HORN

Étude d'appartenance. *Dans tous les algorithmes proposés, que ce soit pour UNI-CAR ou pour EXI-CAR, nous utilisons non seulement un oracle traitant l'inférence classique, mais aussi des oracles appelés MAX-GSAT, NGSAT-CAR (pour UNI-CAR) ou MAX-GSAT-STRICT (pour EXI-CAR). Nous savons que le problème d'inférence classique dans le cas de clauses de Horn devient polynomial. Par contre, nous ne savons rien sur les problèmes MAX-GSAT, NGSAT-CAR et MAX-GSAT-STRICT dans le cas de clauses de Horn (problèmes notés alors MAX-SAT-HORN, NSAT-CAR-HORN et MAX-SAT-STRICT-HORN). Étudions ces problèmes.*

- *Les algorithmes de MAX-SAT-HORN, NSAT-CAR-HORN et MAX-SAT-STRICT-HORN sont les mêmes que ceux de MAX-GSAT, NGSAT-CAR et MAX-GSAT-STRICT. Les problèmes MAX-SAT-HORN, NSAT-CAR-HORN et MAX-SAT-STRICT-HORN appartiennent donc à la classe de complexité NP. Le résultat est le même pour le cas des multi-ensembles : MAX-SAT-ME-HORN, NSAT-CAR-ME-HORN et MAX-SAT-STRICT-ME-HORN appartiennent donc à la classe de complexité NP.*
- *Complétude de MAX-SAT-HORN : on ne peut plus utiliser la même transformation que pour MAX-GSAT puisque SAT devient polynomial quand il s'agit de clauses de Horn. Nous utilisons donc un autre problème NP-complet MAX-2-SAT : "Soit C collection de n clauses à 2 littéraux maximum, soit un entier $k \leq n$, existe-t-il une assignation M satisfaisant au moins k clauses prises dans C ?" (voir [GJ79]).*
  Remarque : Si $k = n$ ou k fixe ($k = n - k0$, avec $k0$ une constante) alors MAX-2-SAT devient polynomial.
  Mais comme ce problème est défini sur une collection de clauses et non sur un ensemble de clauses, nous allons donc démontrer la complétude du problème MAX-SAT-ME-HORN au lieu de démontrer celle de MAX-SAT-HORN.

Soit une collection C de n clauses à 2 littéraux maximum, décomposons C en :
$CH = \{$*les clauses de C qui sont déjà sous forme de clauses de Horn*$\}$
et $C \setminus CH$ et posons :
$E = CH \cup CNH$ *avec :*
$CNH = \{$*les clauses définies de la manière suivante :* $\forall c$ *clause de C de la forme $\rightarrow a_1 a_2$ ($c \in C \setminus CH$), les clauses de Horn $\rightarrow a_1$, $\rightarrow a_2$, $a_1 a_2 \rightarrow \in CNH$*$\}$.

  Remarquons que les clauses $\rightarrow a_1$, $\rightarrow a_2$, $a_1 a_2 \rightarrow$ ainsi générées peuvent ne pas être uniques. C'est donc ici que l'utilisation d'un multi-ensemble prend tout son sens. Bien que E ne soit pas un multi-ensemble tel que nous les avons définis, il est bien évidemment très simple de compter le nombre

d'occurrences de chaque formule de E (ce qui se fait en un temps polynomial) et de créer le multi-ensemble correspondant à E.
Soit p le nombre de clauses de C qui ne sont pas des clauses de Horn ($p = |C \setminus CH|$), soit M une assignation des variables apparaissant dans C, faisons les remarques suivantes :

- *Soit c clause de C sous la forme $\rightarrow a_1 a_2$, M satisfait $c \Leftrightarrow M$ satisfait exactement 2 clauses de l'ensemble $\{\rightarrow a_1, \rightarrow a_2, a_1 a_2 \rightarrow\}$.*
- *Soit c clause de C sous la forme $\rightarrow a_1 a_2$, M ne satisfait pas $c \Leftrightarrow M$ satisfait exactement 1 clause de l'ensemble $\{\rightarrow a_1, \rightarrow a_2, a_1 a_2 \rightarrow\}$ (c'est la clause $a_1 a_2 \rightarrow$).*

Soit M une assignation des variables de C, définissons k_M = nombre de clauses de C satisfaites par M. On a alors : $k_M = k0_M + k1_M$ avec $k0_M$ = nombre de clauses de Horn de C qui sont satisfaites par M et $k1_M$ = nombre de clauses de C sous forme de $\rightarrow a_1 a_2$ qui sont satisfaites par M. Calculons k''_M = nombre de clauses de E satisfaites par M :

$$k''_M = k0_M + 2k1_M + (p - k1_M) = k0_M + k1_M + p = k_M + p$$

Soit k entier $\leq n$ (nombre de clauses de C), posons :
$$f(k) = k' = k + p$$

Montrons alors que MAX-2-SAT $\propto$ MAX-SAT-ME-HORN :

MAX-2-SAT (C,k)
$\Leftrightarrow$
$\exists M$ *assignation satisfaisant au moins k clauses de C, donc $k_M \geq k$*
$\Leftrightarrow$
$\exists M$ *assignation satisfaisant au moins $k + p$ clauses de E,*
puisque $k''_M = k_M + p \geq k + p$
$\Leftrightarrow$
$\exists Y$ *sous-base consistante de E telle que $|Y| \geq k + p$*
$\Leftrightarrow$
$\exists Y$ *sous-base consistante de E telle que $|Y| \geq k' = f(k)$*
$\Leftrightarrow$
MAX-SAT-ME-HORN (E,$f(k)$)

Le problème MAX-SAT-ME-HORN est donc NP-complet.

- *Complétude du problème NSAT-CAR-HORN : il est inutile de prouver la complétude de ce problème puisqu'il ne sert d'oracle que pour un algorithme dans lequel on utilise aussi l'oracle MAX-SAT-HORN dont on vient de montrer que la version multi-ensemble est NP-complète. Et comme on sait que NSAT-CAR-HORN appartient à la classe de complexité NP, il se transforme donc polynomialement en MAX-SAT-ME-HORN. L'algorithme utilisant ces oracles est donc un algorithme faisant appel à un oracle non déterministe polynomial.*
- *Complétude du problème MAX-SAT-STRICT-HORN : on peut prouver que sa version multi-ensemble est NP-complète en utilisant la transformation suivante.*

Soit "E, k" une instance de MAX-SAT-ME-HORN, posons :
- $f(E) = E$ (donc une seule strate),
- $f(k) = k - 1$.

On a alors :

il existe une interprétation satisfaisant au moins k formules de E
$\Leftrightarrow$
il existe une interprétation satisfaisant strictement plus de $k - 1$ formules de E.

On a donc MAX-SAT-ME-HORN $\propto$ MAX-SAT-STRICT-ME-HORN, donc MAX-SAT-STRICT-ME-HORN est NP-complet.

Nous en concluons que les problèmes d'inférence utilisant l'ordre basé sur la cardinalité n'ont donc pas changé de complexité dans le cas de multi-ensembles :

- *UNI-CAR-ME-HORN appartient au plus à la classe de complexité* Δ_2^p,
- *EXI-CAR-ME-HORN appartient au plus à la classe de complexité* Σ_2^p,
- *ARG-CAR-ME-HORN appartient au plus à la classe de complexité* Δ_3^p.

Or le cas des ensembles étant une restriction du cas des multi-ensembles, nous pouvons aussi conclure que :

- *UNI-CAR-HORN appartient au plus à la classe de complexité* Δ_2^p,
- *EXI-CAR-HORN appartient au plus à la classe de complexité* Σ_2^p,
- *ARG-CAR-HORN appartient au plus à la classe de complexité* Δ_3^p.

Étude de complétude.

- *pour UNI-CAR-HORN, on ne peut pas garder la démonstration du cas général, puisque la transformation utilisée ne génère pas des clauses de Horn. Nous ne conservons alors qu'un résultat d'appartenance. Par contre, nous avons trouvé une transformation polynomiale permettant de passer du problème UNI-LEX-HORN au problème UNI-CAR-ME-HORN.*

> *Soit "*$(E, <), H$*" une instance de UNI-LEX-HORN, posons :*
> *-* $f(H) = H$,
> *-* $f(E)$ *= l'ensemble produit par l'algorithme suivant :*
>
> *1.* $f(E) \leftarrow E_n$ *(** E_n *= la strate la moins prioritaire *)*
> *2.* $p \leftarrow 1$
> *3. Pour chaque strate* E_i *de* $(E, <)$ *de* E_{n-1} *à* E_1 *faire*
> *4.* $p \leftarrow p \times (1 + |E_{i+1}|)$
> *5.* $f(E) \leftarrow f(E) \cup \{(\Phi_i, p)$ *pour toutes les formules* $\Phi_i \in E_i\}$
> *fin pour*

Remarques :

- *L'algorithme permettant le calcul de* $f(E)$ *est un algorithme polynomial.*
- $f(E)$ *est un multi-ensemble non stratifié.*

Nous avons alors la propriété suivante :

Propriété 7.3.3 Y *est une thèse préférée lexicographiquement de* $(E, <)$ *ssi* $f(Y)$ *est une thèse préférée de* $f(E)$ *pour l'ordre basé sur la cardinalité.*

Preuve : *La démonstration se fait en 3 étapes :*

- *montrer que* les thèses préférées pour la cardinalité de $f(E)$ sont toujours des images par la fonction f d'un sous-ensemble de $(E, <)$:
  d'après l'algorithme de construction de $f(E)$*, on ne peut avoir comme éléments de* $f(E)$ *que des couples* (Φ_i, p_i) *avec* Φ_i *formule de* $(E, <)$ *et* p_i *le poids calculé par la fonction* f *qui dépend de la strate* i *de* $(E, <)$ *dans laquelle se trouvait* Φ_i *; or la définition des sous-ensembles d'un multi-ensemble donnée en section 7.3.5 impose que les sous-ensembles de* $f(E)$ *ne contiennent que des couples choisis parmi l'ensemble des couples de* $f(E)$ *; on peut alors sans difficulté appliquer la réciproque de la fonction* f *et retrouver le sous-ensemble de* $(E, <)$ *correspondant au sous-ensemble de* $f(E)$ *étudié ; les thèses préférées pour la cardinalité de* $f(E)$ *étant des sous-ensembles de* $f(E)$*, ce sont donc toujours des images par la fonction* f *d'un sous-ensemble de* $(E, <)$*;*
- *montrer que* si Y est une thèse préférée lexicographiquement de $(E, <)$ alors $f(Y)$ est une thèse préférée pour l'ordre basé sur la cardinalité de $f(E)$:

soit $(E,<) = E_1 \cup \ldots \cup E_n$ une base stratifiée, soit A un sous-ensemble de $(E,<)$, d'après l'algorithme de construction de $f(E)$, le cardinal de $f(A)$ se calcule de la façon suivante :

$$\begin{aligned}
|f(A)|^* &= |f(A_1)|^* + |f(A_2)|^* + \ldots + |f(A_n)|^* \\
&= (p_1 \times |A_1|) + (p_2 \times |A_2|) + \ldots + (p_n \times |A_n|) \\
&= [\prod_{j=n-1}^{1} (1 + |E_{j+1}|)] \times |A_1| \\
&\quad +[\prod_{j=n-1}^{2} (1 + |E_{j+1}|)] \times |A_2| \\
&\quad \vdots \\
&\quad +[\prod_{j=n-1}^{n-1} (1 + |E_{j+1}|)] \times |A_{n-1}| \\
&\quad +|A_n|
\end{aligned}$$

soient A et B deux sous-ensembles de $(E,<)$ tels que A est préféré à B pour l'ordre lexicographique sur $(E,<)$, on a alors par définition :
il existe i un numéro de strate de $(E,<)$ tel que $|A_i| > |B_i|$ et $\forall j < i$, $|A_j| = |B_j|$; ainsi, on peut dire que, pour cette strate i, $|A_i| = |B_i| + x$ où $x \geq 1$; d'autre part, on a toujours : $\forall k$, en particulier pour $k > i$, $|B_k| - |A_k| \leq |E_k|$;
calculons alors la différence $|f(A)|^* - |f(B)|^*$:

$$\begin{aligned}
|f(A)|^* - |f(B)|^* &= [\prod_{j=n-1}^{i} (1 + |E_{j+1}|)] \times (|B_i| + x - |B_i|) \\
&\quad +[\prod_{j=n-1}^{i+1} (1 + |E_{j+1}|)] \times (|A_{i+1}| - |B_{i+1}|) \\
&\quad \vdots \\
&\quad +[\prod_{j=n-1}^{n-1} (1 + |E_{j+1}|)] \times (|A_{n-1}| - |B_{n-1}|) \\
&\quad +(|A_n| - |B_n|)
\end{aligned}$$

ce qui implique que :

$$\begin{aligned}
|f(A)|^* - |f(B)|^* &\geq [\prod_{j=n-1}^{i} (1 + |E_{j+1}|)] \\
&\quad -[\prod_{j=n-1}^{i+1} (1 + |E_{j+1}|)] \times |E_{i+1}| \\
&\quad \vdots \\
&\quad -[\prod_{j=n-1}^{n-1} (1 + |E_{j+1}|)] \times |E_{n-1}| \\
&\quad -|E_n| \\
|f(A)|^* - |f(B)|^* &\geq 1 + |E_n| - |E_n| \\
|f(A)|^* - |f(B)|^* &\geq 1
\end{aligned}$$

donc $f(A)$ est préféré pour la cardinalité à $f(B)$.

- *montrer que* si $f(Y)$ est une thèse préférée pour l'ordre basé sur la cardinalité de $f(E)$ alors Y est une thèse préférée lexicographiquement de $(E, <)$:
  soit $f(A)$ une thèse préférée pour l'ordre basé sur la cardinalité de $f(E)$, raisonnons par l'absurde en supposant que A n'est pas une thèse préférée lexicographiquement de $(E, <)$;
  alors il existe B thèse préférée lexicographiquement à A; on sait alors que $f(B)$ est préférée pour la cardinalité à $f(A)$ (voir démonstration précédente) ;
  on a donc une contradiction avec l'hypothèse initiale disant que $f(A)$ est une thèse préférée pour l'ordre basé sur la cardinalité de $f(E)$;
  on en déduit ainsi que A est une thèse préférée lexicographiquement de $(E, <)$.

Grâce à cette propriété, on prouve que :
UNI-LEX-HORN (E, H) $\Leftrightarrow$ UNI-CAR-ME-HORN ($f(E)$, $f(H)$),
ce qui implique que UNI-LEX-HORN $\propto$ UNI-CAR-ME-HORN.
Or on démontre dans la section 7.3.7 que UNI-LEX-HORN est Δ_2^p-complet. Ce qui nous permet de conclure que UNI-CAR-ME-HORN est Δ_2^p-complet.

- *pour EXI-CAR-HORN, on ne peut pas garder la même démonstration que pour EXI-CAR. Toutefois, nous n'avons pas pu trouver une transformation polynomiale d'un problème Σ_2^p-complet en EXI-CAR-HORN. Nous ne conservons alors qu'un résultat d'appartenance.*
- *pour ARG-CAR-HORN, on retrouve les mêmes difficultés que celles rencontrées pour ARG-CAR alourdies encore par le fait que nous ne savons rien sur la complétude de EXI-CAR-HORN. Donc, comme pour EXI-CAR-HORN, nous ne conservons qu'un résultat d'appartenance.*

En conclusion.

UNI-CAR-HORN $\in \Delta_2^p$ mais UNI-CAR-ME-HORN est Δ_2^p-complet
EXI-CAR-HORN $\in \Sigma_2^p$
ARG-CAR-HORN $\in \Delta_3^p$

7.3.7 Complexité de UNI (EXI, ARG)-LEX-HORN

Étude d'appartenance. *L'utilisation de l'ordre lexicographique sur une base stratifiée de clauses de Horn présente les mêmes particularités que l'utilisation de l'ordre basé sur la cardinalité dans une base sans strate. Nous avons déjà étudié l'oracle MAX-GSAT-STRICT dans le cadre de UNI (EXI, ARG)-CAR-HORN, il nous reste à étudier les oracles appelés MAX-GSAT-ARRAY et NSAT-LEX-HORN.*

- *Les algorithmes de MAX-SAT-ARRAY-HORN et de NSAT-LEX-HORN sont les mêmes que ceux de MAX-GSAT-ARRAY et de NGSAT-LEX. Les problèmes MAX-SAT-ARRAY-HORN et NSAT-LEX-HORN appartiennent donc à la classe de complexité NP. Le résultat est le même pour le cas des multi-ensembles : MAX-SAT-ARRAY-ME-HORN et NSAT-LEX-ME-HORN appartiennent donc à la classe de complexité NP.*
- *Complétude du problème MAX-SAT-ARRAY-HORN : on peut utiliser la même transformation que dans le cas général mais uniquement sur les multi-ensembles*[9]*, puisque MAX-SAT-ME-HORN est toujours une restriction de MAX-SAT-ARRAY-ME-HORN. On a donc MAX-SAT-ME-HORN $\propto$ MAX-SAT-ARRAY-ME-HORN, donc MAX-SAT-ARRAY-ME-HORN est NP-complet.*
- *Complétude du problème NSAT-LEX-HORN : comme pour NSAT-CAR-HORN dans UNI-CAR-HORN, il est inutile de montrer la complétude de ce problème, puisqu'il est toujours utilisé en collaboration avec l'oracle MAX-SAT-ARRAY-HORN dont la version multi-ensemble est NP-complète.*

Nous en concluons que les problèmes d'inférence utilisant l'ordre lexicographique n'ont donc pas changé de complexité dans le cas des multi-ensembles :

[9] On ne sait rien sur la complétude de MAX-SAT-HORN.

- *UNI-LEX-ME-HORN appartient au plus à la classe de complexité* Δ_2^p,
- *EXI-LEX-ME-HORN appartient au plus à la classe de complexité* Σ_2^p,
- *ARG-LEX-ME-HORN appartient au plus à la classe de complexité* Δ_3^p.

Or le cas des ensembles étant une restriction du cas des multi-ensembles, nous pouvons aussi conclure que :

- *UNI-LEX-HORN appartient au plus à la classe de complexité* Δ_2^p,
- *EXI-LEX-HORN appartient au plus à la classe de complexité* Σ_2^p,
- *ARG-LEX-HORN appartient au plus à la classe de complexité* Δ_3^p.

Étude de complétude.

- *pour UNI-LEX-HORN, nous ne pouvons pas utiliser les démonstrations du cas général, puisque nous ne savons rien sur UNI-CAR-HORN et que la transformation à partir du problème ALM ne génère pas de clauses de Horn. Essayons donc avec d'autres problèmes* Δ_2^p*-complets. Montrons, par exemple, que ACM* $\propto$ *UNI-LEX-HORN.*

  Soit "$C = \{C_1, \ldots, C_m\}$ *un ensemble de clauses (*$X = \{x_1, \ldots, x_n\}$ *les variables de* C*),* $k = \{1, \ldots, m\}$ *un entier" une instance de ACM, posons :*
  - $H = C_k[y] \wedge s$.
  - $(E, <) =$

  $\{p_1, \ldots, p_m\} \cup \{x_1, \ldots, x_n, y_1, \ldots, y_n\} \cup \{\neg z_1, \ldots, \neg z_n, \neg s\}$ *avec* $y_1, \ldots, y_n$, $z_1, \ldots, z_n$, s *nouveaux symboles de variables propositionnelles, et avec les formules*

  $p_j = (z_1 \ldots z_n \rightarrow s) \wedge \bigwedge_{i=1\ldots n}((\neg x_i \vee \neg y_i) \wedge (y_i \rightarrow z_i) \wedge (x_i \rightarrow z_i)) \wedge C_j[y]$,

  avec la notation suivante : $C_j[y]$ *= la clause* C_j *dans laquelle on remplace tous les littéraux positifs* x_i *par des littéraux négatifs* $\neg y_i$.

  L'ordre $<$ *entre les formules de la base est le suivant : les* p_j *plus prioritaires que les* x_i *et les* y_i*, eux-mêmes plus prioritaires que les* $\neg z_i$ *et le* $\neg s$.

  Notons $Y = \{y_1, \ldots, y_n\}$.
  Les remarques préliminaires faites pour UNI-T-HORN restent valables ici :
  - *L'idée de cette démonstration est la composition de deux idées issues d'articles de Eiter et Gottlob (voir [EG92, EG93]).*
  - *Le but est de transformer un problème portant sur des clauses quelconques en un problème portant sur des clauses de Horn. Il faut donc transformer les clauses initiales en clauses de Horn. Pour cela, on introduit une nouvelle variable propositionnelle* y_i *pour chaque* x_i *de* C*,* y_i *jouera le rôle de* $\neg x_i$*. Cela pose alors un problème de cohérence : il ne faut pas que nous puissions avoir* x_i *et* y_i *ayant la même valeur de vérité. Nous introduisons alors une nouvelle variable propositionnelle* z_i *qui sera vraie quand la valeur de vérité de* x_i *sera différente de la valeur de vérité de* y_i *; ce qui revient à dire que* z_i *est le "ou-exclusif" de* x_i *et de* y_i*. La dernière variable rajoutée* s *sera vraie quand tous les* z_i *seront vrais, donc quand pour tous les* x_i *et* y_i*, on aura valeur-vérité (*x_i*)* $\neq$ *valeur-vérité (*y_i*).*
  - *En transformant* C_j *en* $C_j[y]$*, on obtient une clause de Horn, quel que soit* j.
  - *On a alors les propriétés suivantes :*

    Propriété 7.3.4 *Si la valeur de vérité de* x_i *est différente de la valeur de vérité de* y_i $\forall i$ *alors* C *sera satisfaite ssi* $C[y]$ *est satisfaite.*

    $\forall V$ *une assignation de* X*, notons* $D(V)$ *l'ensemble des clauses de* C *satisfaites par* V*. De même,* $\forall V'$ *une assignation de* $X \cup Y$*, notons* $D(V')$ *l'ensemble des clauses de* $C[y]$ *satisfaites par* V'.

Propriété 7.3.5 *Si la valeur de vérité de x_i est différente de la valeur de vérité de y_i $\forall i$ alors il existe une assignation V de X card-maximale pour C satisfaisant l'ensemble de clauses $D(V)$ ssi il existe une assignation V' de $X \cup Y$ card-maximale pour $C[y]$ satisfaisant $D(V)[y]$*

Preuve : *Toute la preuve repose sur la propriété 7.3.4. L'hypothèse de base est que la valeur de vérité de x_i est différente de la valeur de vérité de y_i $\forall i$.*

- *Il existe une assignation V de X satisfaisant l'ensemble de clauses $D(V) \subseteq C \Leftrightarrow$ (à cause de la propriété 7.3.4 et de l'hypothèse de base) il existe une assignation V' satisfaisant l'ensemble $D(V)[y] \subseteq C[y]$ correspondant à $D(V)$. Remarquons au passage que $D(V') = D(V)[y]$ et que $|D(V)| = |D(V)[y]|$.*
- *V est une assignation de X card-maximale pour $C \Leftrightarrow$ il n'existe pas d'assignation W de X card-maximale pour C, donc telle que $|D(V)| < |D(W)|$.*
  Construisons l'assignation V' telle que $V'(x_i) = V(x_i)$ $\forall x_i \in X$ et $V'(y_i) \neq V(x_i)$ $\forall y_i \in Y$, on a ainsi une assignation V' de $X \cup Y$ satisfaisant $D(V)[y]$.
  Raisonnons par l'absurde et supposons que l'assignation V' n'est pas card-maximale. Il existe alors une autre assignation de $X \cup Y$ notée W' qui est card-maximale pour $C[y]$, on a donc $|D(V')| < |D(W')|$. Donc, l'assignation W correspondant à W' (c'est à dire la restriction de W' à X) aura la propriété suivante : $|D(V)| < |D(W)|$, ce qui est en contradiction avec l'hypothèse V card-maximale.
  De la même façon, on prouve que si V' est une assignation de $X \cup Y$ card-maximale pour $C[y]$ alors l'assignation V de X correspondante est card-maximale pour C.

- *Nous nous trouvons d'autre part dans une optique SYNTAX-BASED, ce qui signifie que, bien que les formules p_j soient des ensembles de clauses de Horn, nous considérons chaque p_j dans son intégralité.*

On constate en plus que :

- *Si on calcule les thèses préférées lexicographiquement T de $(E, <)$, on trouve que : s'il existe une C_j satisfiable alors il existe une T telle que $T \vdash z_i$ ($\forall i$) et $T \vdash s$.*
- *Les seules thèses préférées lexicographiquement inférant s sont les sous-bases contenant le plus de formules p_j, puisque la variable s n'apparaît que dans les p_j.*
- *D'autre part, nous avons toujours les propriétés utilisées pour la démonstration de complétude de UNI-CAR.*

Prouvons maintenant que ACM $\propto$ UNI-LEX-HORN.

- *ACM implique UNI-LEX-HORN :*

  Toute assignation V_c de X card-maximale pour C vérifie : $V_c(C_k) =$ vrai.
  Étendons chaque V_c à Y de telle façon que $V_c(x_i) \neq V_c(y_i)$.
  $\Rightarrow$
  $\forall T$ thèse préférée lexicographiquement de E :

  - *T est constituée de 3 strates T_1, T_2 et T_3 :*
    - *T_1 contient le maximum de p_j, T_1 est maximale pour la cardinalité dans E_1 (première strate de E),*
    - *T_2 contient des x_i et des y_i (exactement n en tout puisque à cause des formules p_j, on doit avoir valeur de x_i différente de valeur de y_i $\forall i$),*
    - *T_3 est vide (puisque à cause des formules p_j, on doit avoir tous les z_i et le s) ;*

    donc T infère s.
  - *$\exists V'$ une assignation de $X \cup Y$ telle que l'ensemble des clauses satisfaites par V' est T_1 (cf. propriété 6.13.3). V' satisfait les formules p_j de T_1, donc les formules $C_j[y]$ correspondantes.*

  $\Rightarrow$
  V' est card-maximale pour E_1 (cf. propriété 6.13.5) et pour $C[y]$
  $\Rightarrow$

$\exists V$ *assignation de* X *card-maximale pour* C *(cf. propriété 7.3.5). Or toutes les assignations card-maximales pour* C *satisfont* C_k.

$\Rightarrow$

V' *satisfait* $C_k[y]$ *(cf. propriété 7.3.5)*

$\Rightarrow$

T_1 *contient la formule* p_k

$\Rightarrow$

T_1 *infère* $C_k[y]$

$\Rightarrow$

toutes les thèses préférées lexicographiquement T *de* $(E, <)$ *infèrent* s *et* $C_k[y]$*, donc infèrent* $C_k[y] \wedge s$.

- *UNI-LEX-HORN implique ACM :*

  quelle que soit T *thèse préférée lexicographiquement de* $(E, <)$*,* T *infère* $C_k[y] \wedge s$*, donc* T *infère* $C_k[y]$ *et* s.

  $\Rightarrow$

  $\forall T$, T_1 *(première strate de* T*) contient la formule* p_k *(puisque* $T \vdash C_k[y]$*) et toute assignation de* $X \cup Y$ *satisfaisant* T *vérifie le fait que la valeur de* x_i *est différente de la valeur de* y_i $\forall i$ *(puisque* $T \vdash s$*).*

  $\Rightarrow$

  $\forall V_c$ *assignation de* X *card-maximale pour* C*, il existe* V' *une assignation de* $X \cup Y$ *card-maximale pour* $C[y]$ *(cf. propriété 7.3.5)*

  $\Rightarrow$

  $D(V')$ *est un sous-ensemble maximal pour la cardinalité pour* $C[y]$ *(cf. propriété 6.13.5)*

  $\Rightarrow$

  l'ensemble $P_{D(V')}$ *des formules* p_j *correspondant à* $D(V')$ *est maximal pour la cardinalité pour* E_1 *(première strate de* E*)*

  $\Rightarrow$

  $P_{D(V')}$ *est un des* T_1 *(première strate des* T *préférées lexicographiquement)*

  $\Rightarrow$

  $P_{D(V')}$ *contient* p_k

  $\Rightarrow$

  $D(V')$ *contient* $C_k[y]$

  $\Rightarrow$

  V' *satisfait* $C_k[y]$

  $\Rightarrow$

  Toute assignation V_c *card-maximale pour* C *vérifie :* $V_c(C_k) = vrai$

  Le problème ACM se ramène donc polynomialement à UNI-LEX-HORN. Et on en déduit que UNI-LEX-HORN est Δ_2^p*-complet.*

- *pour EXI-LEX-HORN, nous ne pouvons pas utiliser la démonstration du cas général, puisque nous ne savons rien sur EXI-CAR-HORN. Toutefois, nous n'avons pas pu trouver une autre transformation polynomiale d'un problème* Σ_2^p*-complet en EXI-LEX-HORN. Nous ne conservons alors qu'un résultat d'appartenance.*
- *pour ARG-LEX-HORN, nous ne pouvons pas utiliser la démonstration du cas général, puisque nous ne savons rien sur ARG-CAR-HORN. Donc, comme pour EXI-LEX-HORN, nous ne conservons qu'un résultat d'appartenance.*

En conclusion.

UNI-LEX-HORN est Δ_2^p*-complet*
EXI-LEX-HORN $\in \Sigma_2^p$
ARG-LEX-HORN $\in \Delta_3^p$

7.3.8 Complexité de UNI (EXI, ARG)-E-HORN

Étude d'appartenance et de complétude. *Parmi ces trois relations d'inférence, deux ont déjà été étudiées par Stillman dans [Sti90] et par Kautz et Selman dans [KS91]. Il s'agit de UNI-E-HORN et EXI-E-HORN. Nous nous contenterons donc de citer leurs résultats. Par contre, il n'y a eu à notre connaissance aucune étude sur ARG-E-HORN.*

Définissons avant tout ce que veut dire l'utilisation de clauses de Horn dans le cadre d'une théorie des défauts pour Stillman, Kautz et Selman. Signalons que la définition de Stillman est différente de celle de Kautz et Selman. Ils n'obtiennent donc pas les mêmes résultats[10].

La définition de Stillman est la suivante :

- *l'ensemble des prémisses W est un ensemble de clauses de Horn,*
- *l'ensemble des défauts D est un ensemble de défauts de la forme $a : b/b$ avec a un unique littéral positif, et b un unique littéral (positif ou négatif),*
- *la formule à inférer est un littéral (positif ou négatif).*

La définition de Kautz et Selman est la suivante :

- *l'ensemble des prémisses W est un ensemble de littéraux,*
- *l'ensemble des défauts D est un ensemble de défauts de la forme $a_1, \ldots, an : b/b$ avec a_i littéral positif $\forall i$, et b un unique littéral (positif ou négatif)*[11],
- *la formule à inférer est un littéral (positif ou négatif).*

Le résultat obtenu par Stillman est le suivant :

- *EXI-E-HORN est NP-complet (preuve de Stillman dans [Sti90]).*

Les résultats obtenus par Kautz et Selman sont les suivants :

- *UNI-E-HORN est co-NP-complet (preuve de Kautz et Selman dans [KS91]).*
- *EXI-E-HORN est polynomial (preuve de Kautz et Selman dans [KS91]).*

Ni Stillman, ni Kautz et Selman n'ont étudié ARG-E-HORN. Peut-on conserver la démonstration faite pour ARG-E ?

- *Oui, pour la preuve d'appartenance puisque l'algorithme utilisé reste inchangé avec un oracle (EXI-E-HORN) qui a diminué en complexité. Suivant la définition utilisée, ARG-E-HORN appartient donc :*
  - *soit à Δ_2^p (définition de Stillman),*
  - *soit à P (définition de Kautz et Selman).*
- *Non, pour la preuve de complétude. Du point de vue de Kautz et Selman, le problème de la complétude ne se pose plus puisque ARG-E-HORN est de complexité P. Et du point de vue de Stillman, nous rencontrons la difficulté suivante : une des transformations polynomiales effectuées dans le cadre de la relation ARG-E ne permet pas de respecter les contraintes imposées par Stillman (voir tableau ci-dessous).*

[10] Il est possible de donner encore d'autres définitions de $E = (W, D)$ et de la formule à inférer dans le cas des clauses de Horn et on obtiendrait sûrement d'autres résultats.

[11] Il s'agit donc, dans les 2 cas, d'une sous-classe des défauts normaux.

	Définition de Stillman
Transformation polynomiale de EXI-E (E, H) *à ARG-E* $(f(E),\ f(H))$ *:* - $f(H) = H$, - $f(E) = E$ *avec ajout de* $H \rightarrow G$ *aux prémisses* (G *nouveau symbole propositionnel)*	*Soient* H *un littéral quelconque et* G *un littéral positif,* $H \rightarrow G$ n'est pas une conjonction de clauses de Horn. *(excepté si* H *est un littéral positif) Transformation non utilisable*
Transformation polynomiale de co-EXI-E (E, H) *à ARG-E* $(f(E),\ f(H))$ *:* - $f(E) = E$ *avec ajout de :*$\neg H/\neg H$ *aux défauts* - $f(H) = \neg H$	*Soit* H *un littéral quelconque, le défaut :*$\neg H/\neg H$ est de la forme $a : b/b$ avec b un littéral unique. *(que* H *soit un littéral positif ou négatif) Transformation utilisable*

Nous pouvons donc toujours montrer que co-EXI-E-HORN $\propto$ *ARG-E-HORN du point de vue de Stillman, donc que ARG-E-HORN est co-NP-difficile.*
Par contre, nous ne pouvons plus montrer que EXI-E-HORN $\propto$ *ARG-E-HORN. La seule possibilité restante est d'imposer que* H *soit un littéral positif. Ainsi, nous conservons la transformation polynomiale définie pour ARG-E et nous montrons que EXI-E-HORN-POSITIF* $\propto$ *ARG-E-HORN avec EXI-E-HORN-POSITIF relation correspondant à EXI-E-HORN en imposant que la formule inférée soit uniquement un littéral positif.*
Or, on ne connaît pas la complexité de cette relation (on sait seulement qu'au pire EXI-E-HORN-POSITIF sera NP-complet puisque c'est une restriction de EXI-E-HORN). Il faudrait donc mener une étude semblable à celle de Stillman pour calculer la complexité de EXI-E-HORN-POSITIF et l'utiliser afin de calculer celle de ARG-E-HORN.

En conclusion. *Suivant la définition choisie pour une théorie des défauts de Horn, on obtient des résultats différents :*

Avec la définition de Stillman :
pas d'étude de UNI-E-HORN
EXI-E-HORN est NP-complet
ARG-E-HORN $\in \Delta_2^p$ *et est co-NP-difficile*

Avec la définition de Kautz et Selman :
UNI-E-HORN est co-NP-complet
EXI-E-HORN $\in P$
ARG-E-HORN $\in P$

REMARQUES SUR LE RAISONNEMENT À PARTIR DE LA LOGIQUE DES DÉFAUTS : *Il existe de nombreux autres cas particuliers concernant la logique des défauts (voir [Sti90] et [KS91]).*

Dans [KS91], Kautz et Selman ont proposé, à partir d'un ensemble W *de littéraux, toute une palette de possibilités pour l'ensemble* D*. Ainsi, nous venons de voir une définition portant sur l'utilisation des défauts de Horn qui permet d'obtenir un algorithme polynomial pour EXI-E-HORN.*

Ils ont aussi proposé une théorie des défauts unaire et normale, c'est-à-dire dont les défauts sont de la forme $a : b/b$ *avec* a *un unique littéral positif, et* b *un unique littéral (positif ou négatif), ce qui est un cas particulier des défauts de Horn. Dans ce cas là, ils ont fourni des algorithmes polynomiaux pour UNI-E et pour EXI-E. Les autres possibilités proposées ne permettent pas de réduire la complexité en deçà de la classe NP.*

La hiérarchie de Kautz et Selman a été reprise par Stillman dans [Sti90] en étendant les choix possibles pour W *et en rajoutant certains choix pour l'ensemble* D*. Nous venons de voir par exemple l'utilisation de clauses de Horn pour* W *avec des défauts unaires et normaux.*

Il démontre aussi que le processus d'inférence EXI-E est polynomial dans le cas de défauts unaires, normaux et sans pré-requis et avec un ensemble W ne contenant que des clauses avec 2 littéraux au plus (rappelons que le problème de satisfaction pour des clauses de 2 littéraux au plus, appelé 2-SAT, est polynomial – voir [GJ79]).

Il serait intéressant de voir la correspondance de tous ces cas particuliers de théories des défauts dans le cas d'une base de croyances propositionnelle classique et de calculer la complexité des processus d'inférence associés. Il est probable que l'on trouverait aussi des algorithmes polynomiaux. Malheureusement le temps nous manque !!

Chapitre 8

Synthèse des résultats et conclusion

Dans le tableau ci-dessous, sont répertoriés tous les résultats démontrés dans les chapitres précédents. On retrouve ainsi les quatre cas étudiés : le cas général, le cas d'une stratification stricte, le cas d'une base de formules CNF, et le cas des clauses de Horn[1].

	Cas propositionnel général	*Cas d'une base stratifiée strictement*	*Cas d'une base de formules CNF*	*Cas d'une base de clauses de Horn*
UNI-T thèses non ordonnées + conséquence forte	Π_2^p*-complet*	*non concerné*	*UNI-T-CNF* Π_2^p*-complet*	*UNI-T-HORN co-NP-complet*
EXI-T thèses non ordonnées + conséquence faible	Σ_2^p*-complet*	*non concerné*	*EXI-T-CNF* Σ_2^p*-complet*	*EXI-T-HORN NP-complet*
ARG-T thèses non ordonnées + conséquence argumentative	Δ_3^p *- (*$\Sigma_2^p \cup \Pi_2^p$*) si* $\Sigma_2^p \neq \Pi_2^p$	*non concerné*	*ARG-T-CNF* Δ_3^p *- (*$\Sigma_2^p \cup \Pi_2^p$*) si* $\Sigma_2^p \neq \Pi_2^p$	*ARG-T-HORN* Δ_2^p *- (NP ∪ co-NP) si NP ≠ co-NP*
UNI-S sous-bases consistantes non ordonnées + conséquence forte	*co-NP-complet*	*non concerné*	*UNI-S-CNF co-NP-complet*	*UNI-S-HORN P*
EXI-S sous-bases consistantes non ordonnées + conséquence faible	Σ_2^p*-complet*	*non concerné*	*EXI-S-CNF* Σ_2^p*-complet*	*EXI-S-HORN NP-complet*
ARG-S sous-bases consistantes non ordonnées + conséquence argumentative	Δ_3^p *- (*$\Sigma_2^p \cup \Pi_2^p$*) si* $\Sigma_2^p \neq \Pi_2^p$	*non concerné*	*ARG-S-CNF* Δ_3^p *- (*$\Sigma_2^p \cup \Pi_2^p$*) si* $\Sigma_2^p \neq \Pi_2^p$	*ARG-S-HORN* Δ_2^p *- (NP ∪ co-NP) si NP ≠ co-NP*
UNI-BO sous-bases consistantes "BEST-OUT" ordonnées + conséquence forte	Δ_2^p *- (NP ∪ co-NP) si NP≠ co-NP*	*1/STRATE-UNI-BO* Δ_2^p *- (NP ∪ co-NP) si NP≠ co-NP*	*UNI-BO-CNF* Δ_2^p *- (NP ∪ co-NP) si NP≠ co-NP*	*UNI-BO-HORN P*
EXI-BO sous-bases consistantes "BEST-OUT" ordonnées + conséquence faible	Σ_2^p*-complet*	*1/STRATE-EXI-BO* Σ_2^p*-complet*	*EXI-BO-CNF* Σ_2^p*-complet*	*EXI-BO-HORN NP-complet*

[1] Remarquons que les résultats présentés ici sont aussi valables dans le cas des multi-ensembles. Le seul problème pour lequel nous avons trouvé une différence entre le cas des ensembles et le cas des multi-ensembles est le problème UNI-CAR-HORN (voir tableau des résultats).

ARG-BO *sous-bases consistantes "BEST-OUT" ordonnées + conséquence argumentative*	Δ_3^p - *(*$\Sigma_2^p \cup \Pi_2^p$*)* *si* $\Sigma_2^p \neq \Pi_2^p$	*1/STRATE-ARG-BO* Δ_3^p - *(*$\Sigma_2^p \cup \Pi_2^p$*)* *si* $\Sigma_2^p \neq \Pi_2^p$	*ARG-BO-CNF* Δ_3^p - *(*$\Sigma_2^p \cup \Pi_2^p$*)* *si* $\Sigma_2^p \neq \Pi_2^p$	*ARG-BO-HORN* Δ_2^p - *(NP ∪ co-NP)* *si NP ≠ co-NP*
UNI-INCL *thèses ordonnées par méthode "INCLUSION BASED" + conséquence forte*	Π_2^p*-complet*	*une seule relation d'inférence pour UNI (EXI, ARG) -INCL (LEX) : 1/STRATE* Δ_2^p*-complet*	*UNI-INCL-CNF* Π_2^p*-complet*	*UNI-INCL-HORN* *co-NP-complet*
EXI-INCL *thèses ordonnées par méthode "INCLUSION BASED" + conséquence faible*	Σ_2^p*-complet*	*(voir UNI-INCL)*	*EXI-INCL-CNF* Σ_2^p*-complet*	*EXI-INCL-HORN* *NP-complet*
ARG-INCL *thèses ordonnées par méthode "INCLUSION BASED" + conséquence argumentative*	Δ_3^p - *(*$\Sigma_2^p \cup \Pi_2^p$*)* *si* $\Sigma_2^p \neq \Pi_2^p$	*(voir UNI-INCL)*	*ARG-INCL-CNF* Δ_3^p - *(*$\Sigma_2^p \cup \Pi_2^p$*)* *si* $\Sigma_2^p \neq \Pi_2^p$	*ARG-INCL-HORN* Δ_2^p - *(NP ∪ co-NP)* *si NP ≠ co-NP*
UNI-LEX *thèses ordonnées par ordre lexicographique + conséquence forte*	Δ_2^p*-complet*	*(voir UNI-INCL)*	*UNI-LEX-CNF* Δ_2^p*-complet*	*UNI-LEX-HORN* Δ_2^p*-complet*
EXI-LEX *thèses ordonnées par ordre lexicographique + conséquence faible*	Σ_2^p*-complet*	*(voir UNI-INCL)*	*EXI-LEX-CNF* Σ_2^p*-complet*	*EXI-LEX-HORN* Σ_2^p
ARG-LEX *thèses ordonnées par ordre lexicographique + conséquence argumentative*	Δ_3^p *et* Σ_2^p *-difficile*	*(voir UNI-INCL)*	*ARG-LEX-CNF* Δ_3^p *et* Σ_2^p *-difficile*	*ARG-LEX-HORN* Δ_3^p
UNI-CAR *thèses ordonnées par ordre basé sur la cardinalité + conséquence forte*	Δ_2^p*-complet*	*non concerné*	*UNI-CAR-CNF* Δ_2^p*-complet*	*UNI-CAR-HORN* Δ_2^p *(et UNI-CAR-ME-HORN* Δ_2^p*-complet*
EXI-CAR *thèses ordonnées par ordre basé sur la cardinalité + conséquence faible*	Σ_2^p*-complet*	*non concerné*	*EXI-CAR-CNF* Σ_2^p*-complet*	*EXI-CAR-HORN* Σ_2^p
ARG-CAR *thèses ordonnées par ordre basé sur la cardinalité + conséquence argumentative*	Δ_3^p *et* Σ_2^p *-difficile*	*non concerné*	*ARG-CAR-CNF* Δ_3^p *et* Σ_2^p *-difficile*	*ARG-CAR-HORN* Δ_3^p
UNI-E *extensions non ordonnées + conséquence forte*	Π_2^p*-complet*	*non concerné*	*UNI-E-CNF* *non étudié*	*UNI-E-HORN* Définition Stillman : *non étudié* Définition Kautz et Selman : *co-NP-complet*

EXI-E *extensions non ordonnées* *+ conséquence faible*	Σ_2^p*-complet*	*non concerné*	*EXI-E-CNF* *non étudié*	*EXI-E-HORN* Définition Stillman : *NP-complet* Définition Kautz et Selman : *P*
ARG-E *extensions non ordonnées* *+ conséquence argumentative*	Δ_3^p *- (*$\Sigma_2^p \cup \Pi_2^p$*)* *si* $\Sigma_2^p \neq \Pi_2^p$	*non concerné*	*ARG-E-CNF* *non étudié*	*ARG-E-HORN* Définition Stillman : Δ_2^p *et* *co-NP-difficile* Définition Kautz et Selman : *P*

Que peut-on conclure de tels résultats ?

Remarquons avant tout que seule la complexité temporelle des problèmes d'inférence non monotone a été étudiée ici. En effet, la théorie de la complexité nous apprend que le temps est une ressource plus critique que l'espace.

Ensuite, nous constatons rapidement que la majorité des problèmes d'inférence non monotone dans le cas propositionnel général ont une complexité exponentielle par rapport à la taille du problème. Bien que ces complexités ne dépassent pas le troisième niveau de la hiérarchie polynomiale, elles restent prohibitives.

Cela signifie que l'interrogation d'une base de connaissances propositionnelle E quelconque pour déterminer si une formule donnée est déductible de E, par une inférence non monotone, risque fort de prendre un temps beaucoup trop important pour un opérateur humain (plusieurs heures, voir plusieurs jours et même plusieurs années).

Il existe différents moyens de réduire ce temps qui se ramènent tous à restreindre le cadre du problème étudié. Nous en avons étudié trois :

- *ne travailler que sur une base stratifiée strictement, ce qui réduit, le plus souvent, la complexité d'un demi-niveau de la hiérarchie polynomiale,*
- *ne travailler que sur des formules CNF, ce qui ne réduit en rien la complexité des problèmes étudiés,*
- *ne travailler que sur des formules et des bases constituées de clauses de Horn, ce qui ramène la complexité au niveau inférieur de la hiérarchie polynomiale dans la plupart des cas.*

La dernière méthode est bien sûr la plus intéressante mais aussi la plus contraignante : toutes les formules propositionnelles ne peuvent pas se traduire sous forme de clauses de Horn (par exemple : $A \vee B$).

Et même avec cette contrainte, on reste malheureusement en majorité dans le cadre de problèmes à complexité exponentielle.

Une analyse plus poussée des résultats nous permet de distinguer les relations d'inférence non-monotone UNI-BO *et* UNI-LEX *qui, dans le pire des cas, ne dépassent pas le premier niveau (Δ_2^p, NP, co-NP) de la hiérarchie polynomiale. Malheureusement, même ces relations ont des inconvénients :*

- UNI-BO *présente le problème de la noyade (voir [BCD+93]) ;*
- *et paradoxalement,* UNI-LEX *n'est pas affectée par la restriction à un langage pauvre (les clauses de Horn) alors que presque toutes les autres relations la rejoignent sur le premier niveau de PH.*

Alors que faire ? Les problèmes d'inférence non monotone sont-ils condamnés à ne jamais être résolus ?

En fait, nous ne pensons pas que ces résultats permettent d'affirmer que toute utilisation pratique de ces formalismes est impossible. En effet :

- *Ce problème est aussi celui de la logique classique (l'inférence classique est de complexité co-NP-complet), et il existe désormais un certain nombre de mécanismes, d'algorithmes permettant d'obtenir quelques résultats (voir les travaux sur* GSAT *dans [LMS92]).*
- *Le développement de machines massivement parallèles de plus en plus rapides, de plus en plus puissantes, se poursuit. Elles pourraient peut-être permettre de repousser la frontière de l'"inutilisabilité" pratique suffisamment loin pour pouvoir traiter des instances de taille suffisante.*
- *Et puis surtout les calculs de complexité effectués ici reflètent le pire des cas et bon nombre de problèmes pratiques à traiter ne sont pas dans cette catégorie.*

Il faudrait pour affiner cette conclusion faire aussi une étude de complexité en moyenne, ce que nous ne ferons pas ici par manque de compétence dans ce domaine.

En conclusion, pour traiter l'inférence non monotone, il faut définir le problème le plus précisément possible, afin d'utiliser au mieux ses particularités lors de la définition des algorithmes de résolution du problème. Cela pose alors le problème de la réutilisation de tels algorithmes. Nous pensons qu'il faudra probablement en venir à la définition de diverses catégories de problèmes et à l'écriture d'algorithmes propres à chaque catégorie.

Quant aux perspectives, elles ne manquent pas ! Nous avons 2 axes de recherche en vue :

- *l'étude de la complexité des relations d'inférence du type $\Psi \mid\!\sim^{p,m}_{E,<} \Phi$, c'est-à-dire les relations d'inférence non monotone liant 2 formules quelconques connaissant une base de croyances (ordonnée ou pas) ; de telles relations d'inférence sont à définir par rapport aux relations $(E,<) \mid\!\sim^{p,m} \Phi$ étudiées ici ;*
- *l'étude pour chacune des relations d'inférence citées dans ce document de leurs propriétés au sens de Kraus, Lehmann, Magidor, Gärdenfors et Makinson (voir [KLM90], [Gär91] et [GM94]), par exemple : la réflexivité, l'équivalence logique gauche[2], l'affaiblissement droit[3], le CUT, etc. Cette étude concernera en particulier le type d'inférence $\Psi \mid\!\sim^{p,m}_{E,<} \Phi$.*

[2] En anglais : "Left Logical Equivalence".
[3] En anglais : "Right Weakening".

Annexe A

Les théorèmes et définitions utilisés

Dans cette annexe, nous allons répertorier les principaux théorèmes et définitions utilisés lors des démonstrations de complexité. Pour chacun des théorèmes, nous donnons soit la référence où trouver la preuve, soit la preuve elle-même.

Remarque : dans le cas des théorèmes donnant la complexité des processus de révision FMR, SBR, PBR, UBR auxquels nous nous sommes référés lors nos propres démonstrations de complexité, nous explicitons dans une première partie les démonstrations faites initialement par Nebel dans [Neb91], car les mécanismes mis en œuvre nous ont beaucoup inspirés pour nos propres démonstrations. Puis nous calculons la complexité de ces mêmes processus dans un cas particulier (celui des clauses de Horn) en utilisant la technique donnée par Eiter et Gottlob dans [EG92].

A.1 Les théorèmes et définitions généraux

Définition A.1.1 *Une transformation polynomiale f est une fonction calculable en un temps polynomial et telle que $\forall x$, $x \in L_1 \Leftrightarrow f(x) \in L_2$ (ceci est noté $L_1 \propto L_2$).*

Corollaire A.1.1 *Dans le cadre de problèmes de décision, $L_1 \propto L_2 \Leftrightarrow$ co-$L_1 \propto$ co-L_2.*

Preuve :

$L_1 \propto L_2$

$\Leftrightarrow$

il existe une transformation polynomiale f telle que ($x \in L_1 \Leftrightarrow f(x) \in L_2$) pour tout x

$\Leftrightarrow$

il existe une transformation polynomiale f telle que ($x \notin L_1 \Leftrightarrow f(x) \notin L_2$) pour tout x

Or nous sommes dans le cadre de problèmes de décision. Ce qui signifie que les langages L_1 et L_2 représentent des instances positives de problèmes, c'est-à-dire des instances dont la solution est oui. Donc dire que $x \notin L_1$ est équivalent à dire que $x \in$ co-L_1. On a donc :

$\Leftrightarrow$

il existe une transformation polynomiale f telle que ($x \in$ co-$L_1 \Leftrightarrow f(x) \in$ co-L_2) pour tout x

$\Leftrightarrow$

co-$L_1 \propto$ co-L_2

Corollaire A.1.2 *Dans le cadre de problèmes de décision, pour toute classe de complexité X de la hiérarchie polynomiale, on a : L est X-complet $\Leftrightarrow$ co-L est co-X-complet.*

Preuve :

L *est X-complet*

$\Leftrightarrow$

$L \in$ *X et* $\forall L' \in$ *X,* $L' \propto L$. *Or* $L' \propto L \Leftrightarrow$ *co-*$L' \propto$ *co-*L. *On a donc :*

$L \in$ *X et* $\forall L' \in$ *X, co-*$L' \propto$ *co-*L

$\Leftrightarrow$

co-$L \in$ *co-X et* $\forall$ *co-*$L' \in$ *co-X, co-*$L' \propto$ *co-*L

$\Leftrightarrow$

co-L *est co-X-complet.*

Théorème A.1.1 *Si L est NP-complet et co-*$L \in$ *NP alors NP* $=$ *co-NP*

Preuve : *(voir [GJ79])*

Théorème A.1.2 *Soit X une classe de complexité de la hiérarchie polynomiale,* $\forall L_1, L_2 \in$ *X,* L_1 *étant X-complet,* $L_1 \propto L_2$*, alors* L_2 *est X-complet.*

Preuve :

Pour toute classe X de la hiérarchie polynomiale, L_2 *sera X-complet ssi* $L_2 \in$ *X et pour tout* $L \in$ *X on a* $L \propto L_2$.

Or on sait déjà que $L_2 \in$ *X puisque cela fait partie des hypothèses. Il suffit de montrer que pour tout* $L \in$ *X on a* $L \propto L_2$.

On sait que L_1 *est X-complet, donc que pour tout* $L \in$ *X on a* $L \propto L_1$. *Et on sait que* $L_1 \propto L_2$.

D'autre part, on sait que la relation $\propto$ *est transitive (voir [GJ79]).*

On a donc pour tout $L \in$ *X,* $L \propto L_2$.

L_2 *est donc X-complet.*

Théorème A.1.3 *Soit X une classe de complexité de la hiérarchie polynomiale, si* L *est X-complet et co-*$L \in$ *X alors X* $=$ *co-X*[1].

Preuve :

1. $L \in$ *X-complet* $\Leftrightarrow$ *co-*$L \in$ *co-X-complet* $\Leftrightarrow$ *pour tout* $L' \in$ *co-X on a* $L' \propto$ *co-*L
   Or on sait par hypothèse que co-$L \in$ *X donc comme* $L \in$ *X-complet, on a co-*$L \propto L$.
   La transformation polynomiale étant transitive, on a donc pour tout $L' \in$ *co-X,* $L' \propto L$
   $\Rightarrow$
   pour tout $L' \in$ *co-X,* $L' \in$ *X* $\Rightarrow$ *co-X inclus dans X*
2. $L \in$ *X-complet* $\Leftrightarrow$ *co-*$L \in$ *co-X-complet.*
   D'autre part co-$L \in$ *X donc* $L \in$ *co-X et co-*L *étant co-X-complet, alors* $L \propto$ *co-*L.
   Avec de plus le fait que $L \in$ *X-complet* $\Leftrightarrow$ *pour tout* $L' \in$ *X on a* $L' \propto L$.
   La transformation polynomiale étant transitive, on a donc pour tout $L' \in$ *X,* $L' \propto$ *co-*L
   $\Rightarrow$
   pour tout $L' \in$ *X,* $L' \in$ *co-X* $\Rightarrow$ *X inclus dans co-X*
3. *En conclusion : X* $=$ *co-X.*

[1] Ce théorème est une généralisation du théorème A.1.1 cité précédemment.

Théorème A.1.4 *Soit k un entier représentant un niveau de la hiérarchie polynomiale ($k \geq 1$), les deux propriétés suivantes sont équivalentes :*

(1) si $\Sigma_{k-1}^p \neq \Pi_{k-1}^p$, $L \in \Delta_k^p$ - ($\Sigma_{k-1}^p \cup \Pi_{k-1}^p$),

(2) $L \in \Delta_k^p$ et est Σ_{k-1}^p-difficile et Π_{k-1}^p-difficile.

Preuve :

- *(1) $\Rightarrow$ (2) :*
  (1) $\Leftrightarrow L \in \Delta_k^p$ - ($\Sigma_{k-1}^p \cup \Pi_{k-1}^p$), si $\Sigma_{k-1}^p \neq \Pi_{k-1}^p$. Rappelons que : si $\Sigma_{k-1}^p = \Pi_{k-1}^p$ alors la hiérarchie polynomiale s'effondre sur Σ_{k-1}^p. Tant que l'on suppose vraie la conjecture $\Sigma_{k-1}^p \neq \Pi_{k-1}^p$, on suppose par là-même que les classes Δ_k^p et ($\Sigma_{k-1}^p \cup \Pi_{k-1}^p$) sont distinctes. Ainsi, si L appartient à la différence Δ_k^p - ($\Sigma_{k-1}^p \cup \Pi_{k-1}^p$), alors, par définition de la hiérarchie polynomiale, L est ($\Sigma_{k-1}^p \cup \Pi_{k-1}^p$)-difficile. On a ainsi : $L \in \Delta_k^p$ et est Σ_{k-1}^p-difficile et Π_{k-1}^p-difficile, ce qui est exactement la propriété (2).
- *(2) $\Rightarrow$ (1) :*
  (2) $\Leftrightarrow L \in \Delta_k^p$ et est Σ_{k-1}^p-difficile et Π_{k-1}^p-difficile. Donc, soit L' problème Σ_{k-1}^p-complet, on a : $L' \propto L$ et co-$L' \propto L$. Prenons maintenant comme hypothèse que :
  - *$L \in \Sigma_{k-1}^p$:*
    Utilisons le théorème A.1.3 donné ci-dessus. On sait que co-$L' \propto L$, donc avec l'hypothèse $L \in \Sigma_{k-1}^p$, il est possible de résoudre co-L' avec un algorithme Σ_{k-1}^p, ce qui revient à dire que co-$L' \in \Sigma_{k-1}^p$.
    D'autre part, on sait que L' est Σ_{k-1}^p-complet. Nous obtenons alors : L' est Σ_{k-1}^p-complet, co-$L' \in \Sigma_{k-1}^p$, on a donc $\Sigma_{k-1}^p = \Pi_{k-1}^p$.
  - *$L \in \Pi_{k-1}^p$:*
    On sait que $L_1 \propto L_2 \Leftrightarrow$ co-$L_1 \propto$ co-L_2, $\forall$ L_1 et L_2. De plus on sait que $L' \propto L$ donc co-$L' \propto$ co-L.
    Ainsi, avec l'hypothèse $L \in \Pi_{k-1}^p$, on a co-$L \in \Sigma_{k-1}^p$, et il est donc possible de résoudre co-L' avec un algorithme Σ_{k-1}^p, ce qui revient à dire que co-$L' \in \Sigma_{k-1}^p$.
    Nous pouvons donc à nouveau appliquer le même théorème : L' est Σ_{k-1}^p-complet, co-$L' \in \Sigma_{k-1}^p$, on a donc $\Sigma_{k-1}^p = \Pi_{k-1}^p$.

  Nous avons ainsi, si $\Sigma_{k-1}^p \neq \Pi_{k-1}^p$, $L \in \Delta_k^p$ - ($\Sigma_{k-1}^p \cup \Pi_{k-1}^p$), ce qui est exactement la propriété (1).

A.2 Les théorèmes et définitions pour la révision

A.2.1 Complexité de FMR

Cas général

Théorème A.2.1 *Dans le cas propositionnel général, le processus de révision appelé FMR appartient à la classe de complexité Δ_2^p - (NP $\cup$ co-NP), si NP $\neq$ co-NP.*

Preuve : *(voir [Neb91])*

Il faut calculer la complexité du problème $P : x \in Cn(Z)$ révisé par y en utilisant la méthode FMR[2] ?

Étude $1^{ère}$ ***partie : l'appartenance à une classe.***

Un algorithme pour FMR est le suivant :

1. Si Z n'infère pas $\neg y$ alors
2. $Z \cup y$ infère-t-il x ?
3. sinon y infère-t-il x ?
fin si

On sait d'autre part que le problème $BC \vdash H$ est de complexité co-NP et donc que le co-problème associé $BC \not\vdash H$ est de complexité NP. Cela nous donne donc pour FMR un algorithme polynomial faisant appel à un oracle NP et à un oracle co-NP. Ce qui ne correspond à rien de connu dans la hiérarchie polynomiale. Par contre, on peut sans problème modifier cet algorithme de manière à n'utiliser plus qu'un oracle NP[3] :

1. Si Z n'infère pas $\neg y$ alors
2. si $Z \cup y$ n'infère pas x alors échec $\leftarrow$ vrai
3. sinon échec $\leftarrow$ faux
fin si
4. sinon si y n'infère pas x alors échec $\leftarrow$ vrai
5. sinon échec $\leftarrow$ faux
fin si
fin si
6. vérifier que l'on n'a pas échec (échec = faux)

Dans cet algorithme qui est polynomial, on utilise un oracle de complexité NP.

Le problème FMR est donc dans la classe de complexité $P^{NP} = \Delta_2^p$.

Étude $2^{ème}$ ***partie : la complétude ?***

À ce jour, on ne connaît pas de problème Δ_2^p-complet. On ne peut donc pas prouver de manière simple la complétude, si complétude il y a. Par contre, on peut raffiner l'appartenance à la classe Δ_2^p en vérifiant si FMR $\in$ NP ou co-NP.

Pour cela, on va essayer de comparer FMR à des problèmes connus de NP et co-NP. Les problèmes les plus connus sont GSAT et UNGSAT (satisfaisabilité et insatisfaisabilité d'une formule en logique classique propositionnelle). On constate alors que :

[2] Le problème FMR se définit de la manière suivante : soit Z un ensemble de formules, soient y et x deux formules, Z révisé par y infère-t-il x ? L'opérateur de révision est défini comme suit : l'ensemble "Z révisé par y" $= Z \cup \{y\}$ si Z est consistant avec y, sinon y. Voir [Neb91] pour plus de détail.

[3] Cette technique est systématiquement utilisée y compris pour des oracles de complexité X quelconque et elle justifie le fait que la hiérarchie polynomiale ne soit définie que pour des oracles de classe NP, $\Sigma_2^p, \Sigma_3^p, \ldots, \Sigma_k^p$.

- *Le problème GSAT se ramène au problème FMR par une transformation polynomiale : Soit H une formule propositionnelle, posons :* $Z = \{H\}$, $y =$ *TRUE, et* $x = H$[4].

  $$H \text{ est satisfiable} \Leftrightarrow H \in Cn(\{H\}) \text{ révisé par TRUE.}$$

  En conclusion, on a GSAT qui se transforme polynomialement en FMR (noté GSAT $\propto$ *FMR).*
- *Le problème UNGSAT (complémentaire de GSAT) se ramène au problème FMR de manière polynomiale (on applique le même raisonnement que précédemment). Soit* H *une formule propositionnelle, posons :* $Z = \varnothing$, $y = H$ *et* $x = \bot$.

  $$\begin{array}{c} H \text{ est insatisfiable} \\ \Leftrightarrow \\ \text{il n'existe pas d'ensemble de formules consistant } M \text{ tel que } M \vdash H \\ \Leftrightarrow \\ H \vdash \bot. \\ \Leftrightarrow \\ \bot \in Cn(\varnothing) \text{ révisé par } H. \end{array}$$

  En conclusion, on a UNGSAT $\propto$ *FMR.*

En utilisant le théorème A.1.4 donné ci-dessus, on arrive à la conclusion suivante.

En conclusion :

Si NP $\neq$ *co-NP, FMR* $\in \Delta_2^p$ *- (NP* $\cup$ *co-NP).*

Cas d'une base de clauses de Horn

Théorème A.2.2 *Dans le cas propositionnel avec uniquement des clauses de Horn, le processus de révision appelé FMR appartient à la classe de complexité P.*

Preuve :

L' algorithme pour FMR-HORN est le même que celui de FMR. C'est un algorithme polynomial, dans lequel on utilise un oracle qui est désormais de complexité P puisque le problème de satisfaction d'un ensemble de clauses de Horn est polynomial.

Le problème FMR-HORN est donc dans la classe de complexité P.

A.2.2 Complexité de SBR

Cas général

Théorème A.2.3 *Dans le cas propositionnel général, le processus de révision appelé SBR est* Π_2^p*-complet.*

[4] Cette construction ne "tombe pas du ciel". On y arrive en tenant le raisonnement suivant. Tout d'abord, on remarque que $BC \vdash H$ est de complexité co-NP (donc se ramène à un problème UNGSAT), tandis que $BC \nvdash H$ est de complexité NP (donc se ramène à un problème GSAT) ; ainsi l'algorithme de FMR donné ci-dessus peut s'exprimer de la façon suivante :

1. Si GSAT alors
2. *UNGSAT numéro 1*
3. sinon UNGSAT numéro 2

Pour que GSAT se ramène à cet algorithme, il suffit que :

- *ce problème GSAT corresponde au problème GSAT du Si,*
- *le problème UNGSAT numéro 1 corresponde à un problème donc la réponse est oui quand la réponse au problème GSAT est oui,*
- *le problème UNGSAT numéro 2 corresponde à un problème donc la réponse est non quand la réponse au problème GSAT est non.*

Preuve : *(voir [Neb91])*

Il faut calculer la complexité du problème P : $x \in Cn(Z)$ révisé par y en utilisant la méthode SBR[5] *?*

Dans la méthode SBR, le problème "$x \in Cn(Z)$ révisé par y" consiste à prouver que : "quel que soit Y un sous-ensemble de Z tel que Y n'infère pas $\neg y$ et est maximal au sens de l'inclusion (Y est dit maximal-y-consistant), alors $Y \cup \{y\}$ infère x".

Il va être délicat de calculer la complexité de ce problème sachant qu'il faut passer en revue tous les sous-ensembles Y de Z vérifiant les propriétés énoncées ci-dessus. On va donc plutôt calculer la complexité du co-problème dans lequel il suffit de trouver un sous-ensemble Y vérifiant les propriétés mais n'inférant pas x.

Étude $1^{ère}$ partie : l'appartenance à une classe.

Un algorithme non déterministe pour co-SBR est le suivant :

1. *deviner un sous-ensemble Y de Z*
2. *vérifier qu'il est maximal pour l'inclusion et n'infère pas $\neg y$*
   (c'est à dire que quel que soit z appartenant à $Z - Y$,*
   *on a $Y \cup \{z\}$ qui infère $\neg y$ *).*
3. *vérifier que $Y \cup \{y\}$ n'infère pas x.*

Dans cet algorithme qui est non déterministe (à cause du "deviner") polynomial, on utilise un oracle de complexité NP (même raisonnement que pour FMR).

Le problème co-SBR appartient donc à la classe de complexité $NP^{NP} = \Sigma_2^p$, donc le problème SBR appartient à co-$\Sigma_2^p = \Pi_2^p$.

Étude $2^{ème}$ partie : la complétude ?

On connaît des problèmes Σ_2^p-complets. L'un des plus connus est le problème 2-QBF : "vérifier la satisfaisabilité de la formule $\exists a \forall b H(a,b)$ avec $a = (a_1, \ldots, a_n)$ et $b = (b_1, \ldots, b_m)$".

Pour montrer que co-SBR est Σ_2^p-complet, il suffit de montrer que le problème 2-QBF se ramène polynomialement à co-SBR. Et on pourra alors déduire que co-SBR est Σ_2^p-complet en utilisant le théorème A.1.2.

Comment faire le lien entre co-SBR et 2-QBF ? On cherche à démontrer que le problème 2-QBF se ramène polynomialement à co-SBR. Le principe est le même que celui utilisé pour FMR. Il faut montrer qu'il existe une fonction f telle que l'image par f d'une instance positive du problème numéro 1 "$\exists a \forall b H(a,b)$ est satisfiable" est une instance positive du problème numéro 2 "Z révisé par y avec la méthode SBR n'infère pas x". Posons donc :

$$x = \neg H(a,b),\ y = \textit{TRUE},\ Z = \{a_1, \ldots, a_n, \neg a_1, \ldots, \neg a_n, \neg H(a,b)\}.$$

- *on définit ainsi une fonction f calculable en un temps polynomial,*
- *on a dans Z toutes les "valeurs" possibles pour les variables propositionnelles $a_1, \ldots, a_n$ dont on cherche s'il existe une assignation,*
- *on n'impose aucune contrainte sur les variables propositionnelles $b_1, \ldots, b_m$, dont toutes les assignations doivent être prises en compte,*
- *on a dans Z la négation de la formule dont on cherche à prouver la satisfaisabilité.*

Montrons alors que : "$\{a_1, \ldots, a_n, \neg a_1, \ldots, \neg a_n, \neg H(a,b)\}$ révisé par TRUE avec la méthode SBR n'infère pas $\neg H(a,b)$" équivaut à "$\exists a \forall b H(a,b)$ est satisfiable" :

[5] Le problème SBR se définit de la manière suivante : soit Z un ensemble de formules, soient y et x deux formules, Z révisé par y infère-t-il x ? L'opérateur de révision est défini comme suit : l'ensemble "Z révisé par y" $= (\cap_{Y \in (Z \downarrow \neg y)} Cn(Y)) + y$, avec l'opérateur $\downarrow$ défini informellement par $Z \downarrow y = \{$ensembles de formules n'impliquant pas y et maximaux pour l'inclusion$\}$, la notation $Cn(A)$ signifiant l'ensemble des conséquences logiques de A. Voir [Neb91] pour plus de détail, en particulier pour la définition formelle de $Z \downarrow y$. Remarquons que les Y sont les thèses de Z consistantes avec y.

$\{a_1, \ldots, a_n, \neg a_1, \ldots, \neg a_n, \neg H(a,b)\}$ *révisé par TRUE avec la méthode SBR n'infère pas* $\neg H(a,b)$

$\Leftrightarrow$

il existe un sous-ensemble Y *de* Z *tel que* Y *soit maximal pour l'inclusion et n'infère pas* $\neg$ *(TRUE)* $=$ *contradiction* $= \bot$ *(donc maximal consistant) et tel que* Y *n'infère pas* $\neg H(a,b)$

$\Leftrightarrow$

vu la forme de Z *choisie et pour que* Y *soit maximal consistant et n'infère pas* $\neg H(a,b)$*, alors* $Y = \{l_1, \ldots, l_n\}$ *avec* $l_i = a_i$ *ou* $\neg a_i$*,* $\forall i = 1 \ldots n$*.*

$\Leftrightarrow$

$\exists \{l_1, \ldots, l_n\}$ *qui est inconsistant avec* $\neg H(a,b)$

$\Leftrightarrow$

$\exists \{l_1, \ldots, l_n\}$ *qui infère* $H(a,b)$

$\Leftrightarrow$

il existe un modèle des variables propositionnelles $a_1, \ldots, a_n$ *tel que sans la moindre contrainte sur les variables propositionnelles* $b_1, \ldots, b_m$ *(donc pour tout modèle de* $b_1, \ldots, b_m$*) on a* $H(a,b)$ *vraie*

$\Leftrightarrow$

$\exists a \forall b\, H(a,b)$ *est satisfiable*

Le problème 2-QBF se ramène donc polynomialement à co-SBR. Et on en déduit en utilisant le théorème A.1.2 que le problème co-SBR est Σ_2^p*-complet et SBR est* Π_2^p*-complet.*

En conclusion :

Le problème SBR est Π_2^p*-complet.*

Cas d'une base de clauses de Horn

Théorème A.2.4 *Dans le cas propositionnel avec uniquement des clauses de Horn, le processus de révision appelé SBR est co-NP-complet (il sera noté SBR-HORN).*

Preuve :

Étude $1^{ère}$ ***partie : l'appartenance à une classe.***

L'algorithme pour co-SBR-HORN est le même que celui de co-SBR. Il s'agit d'un algorithme non déterministe polynomial (à cause du "deviner"), dans lequel on utilise un oracle de complexité P.

Le problème co-SBR-HORN appartient donc à la classe de complexité NP, donc le problème SBR-HORN appartient à co-NP.

Étude $2^{ème}$ ***partie : la complétude ?***

On connaît des problèmes NP-complets. L'un des plus connus est le problème SAT.

Pour montrer que co-SBR-HORN est NP-complet, il suffit de montrer que le problème SAT se ramène polynomialement à co-SBR-HORN.

Comment faire le lien entre co-SBR-HORN et SAT ? Nous allons nous inspirer de la démonstration utilisée par Eiter et Gottlob dans [EG92]. Soit $C = \{C_j\}$ *pour* $j = 1 \ldots m$ *ensemble de clauses quelconques à satisfaire, soit* $V(C) = \{x_1, \ldots, x_n\}$ *l'ensemble des variables propositionnelles apparaissant dans* C*, posons :*

$Z = \{x_1, \ldots, x_n, y_1, \ldots, y_n, \neg z_1, \ldots, \neg z_n, \neg s\}$ *avec* $y_1, \ldots, y_n$, $z_1, \ldots, z_n$, s *nouveaux symboles de variables propositionnelles,*

$x = \neg s$,

$y = (z_1 \ldots z_n \rightarrow s) \wedge \bigwedge_{i=1\ldots n}((\neg x_i \vee \neg y_i) \wedge (y_i \rightarrow z_i) \wedge (x_i \rightarrow z_i)) \wedge \bigwedge_{j=1\ldots m} C_j[y]$, *avec la notation suivante :* $C_j[y] =$ *la clause* C_j *dans laquelle on remplace tous les* x_i *positifs par des* $\neg y_i$.

Quelques explications :

- *Le but est de transformer un problème portant sur des clauses quelconques en un problème portant sur des clauses de Horn. Il faut donc transformer les clauses initiales en clauses de Horn. Pour cela, on introduit une nouvelle variable propositionnelle* y_i *pour chaque* x_i *de* C, *cette* y_i *jouera le rôle de* $\neg x_i$. *Cela pose alors un problème de cohérence : il ne faut pas que nous puissions avoir* x_i *et* y_i *ayant la même valeur de vérité. Nous introduisons alors une nouvelle variable propositionnelle* z_i *qui sera vraie quand la valeur de vérité de* x_i *sera différente de la valeur de vérité de* y_i *; ce qui revient à dire que* z_i *est le "ou exclusif" de* x_i *et de* y_i. *La dernière variable rajoutée* s *sera vraie quand tous les* z_i *seront vraies, donc quand pour tous les* x_i *et* y_i, *on aura valeur-vérité (*x_i*)* $\neq$ *valeur-vérité (*y_i*).*
- *En transformant* C_j *en* $C_j[y]$, *on obtient une clause de Horn, quel que soit* j.
- *On a alors la propriété suivante :*

  Propriété A.2.1 *Si la valeur de vérité de* x_i *est différente de la valeur de vérité de* y_i $\forall i$ *alors* C *sera satisfaite ssi* $C[y]$ *est satisfaite.*
- *De plus, si on calcule les sous-bases maximales consistantes* Y *de* Z *révisé avec* y, *on trouve que :*

  Propriété A.2.2 *Si* C *est satisfiable alors il existe* Y *tel que* $Y \vdash z_i$ $(\forall i)$ *et* $Y \vdash s$. *Un tel* Y *n'inférera pas* $\neg s$.

  Preuve :
  C *satisfiable* $\Leftrightarrow$ *il existe une interprétation* Φ *dans laquelle* C *est vraie*
  $\Rightarrow$
  Posons :

  $$W = \{x_i \text{ tels que } \Phi(x_i) \text{ soit vrai}\} \cup \{yj \text{ tels que } \Phi(xj) \text{ soit faux}\}.$$

  On sait d'autre part que $\forall i\ \Phi(x_i) \neq \Phi(y_i)$ *puisque* $y_i = \neg x_i$.
  On a donc : $\forall j\ C_j[y]$ *satisfaite puisque* C_j *est satisfaite et que* x_i *et* y_i *n'ont pas la même valeur de vérité* $\forall i$.
  On peut ainsi en déduire que y *est satisfaite, donc que* $Y = W \cup \{y\}$ *est maximale consistante pour* Z *révisé par* y.
  $\Rightarrow$
  il existe une sous-base maximale consistante Y *de* Z *révisé par* y *de la forme :*
  $Y = \{x_i \text{ tels que } \Phi(x_i) \text{ soit vrai}\} \cup \{y_i \text{ tels que } \Phi(x_i) \text{ soit faux}\} \cup \{y\}$.
  Or une telle sous-base infère tous les z_i *et infère* s.
  Donc Y *n'infère pas* $\neg s$
- Z *est un ensemble de clauses de Horn.*
- y *est un ensemble de clauses de Horn puisque les* $C_j[y]$ *sont des clauses de Horn.*
- x *est une clause de Horn.*

Montrons maintenant que SAT implique co-SBR-HORN et que co-SBR-HORN implique SAT.

- *SAT implique co-SBR-HORN :*

C *satisfiable*

$\Rightarrow$

il existe une sous-base maximale consistante Y *(*$Y = W \cup \{y\}$*) de* Z *révisé par* y *telle que* Y *infère tous les* z_i *et infère* s *et* Y *n'infère pas* $\neg s$

$\Leftrightarrow$

co-SBR-HORN

- *co-SBR-HORN implique SAT :*

co-SBR-HORN

$\Leftrightarrow$

il existe Y *une sous-base maximale consistante de* Z *révisé par* y *telle que* Y *n'infère pas* $\neg s$*.*

Or $\neg s$ *appartient à* Z*, donc si* Y *n'infère pas* $\neg s$*, cela signifie que* Y *est inconsistante avec* $\neg s$*, donc que* Y *infère* s*, donc que* Y *contient* y*, que* $\forall i$ *la valeur de vérité de* x_i *est différente de celle de* y_i *et que les* $C_j[y]$ *sont satisfaites par la même interprétation que celle satisfaisant* Y*.*

Posons comme interprétation Φ *qui satisfait* Y *:*

$\Phi(x_i) =$ *vrai si* x_i *appartient à* $Y - \{y\}$ *et faux sinon*

$\Rightarrow$

Dans l'interprétation Φ*, les* $C_j[y]$ *sont satisfaites et comme les* x_i *et les* y_i *n'ont pas la même valeur de vérité*

$\Rightarrow$

les C_j *sont satisfaites elles aussi*

$\Leftrightarrow$

C *satisfiable*

Le problème SAT se ramène donc polynomialement à co-SBR-HORN. Et on en déduit que le problème co-SBR-HORN est NP-complet. Donc le problème SBR-HORN est co-NP-complet.

En conclusion :

Le problème SBR-HORN est co-NP-complet.

A.2.3 Complexité de PBR

Cas général

Théorème A.2.5 *Dans le cas propositionnel général, le processus de révision appelé PBR est* Π_2^p*-complet.*

Preuve :*(voir [Neb91])*

Il faut calculer la complexité du problème P *:* $x \in Cn(Z)$ *révisé par* y *en utilisant la méthode PBR*[6] *?*

Dans la méthode PBR, le problème "$x \in Cn(Z)$ *révisé par* y*" consiste à prouver que : "soit* Z *une base stratifiée, quel que soit* Y *sous-ensemble de* Z *qui n'infère pas* $\neg y$ *et qui est préféré pour l'ordre "INCLUSION BASED" (voir définition de PBR), alors* $Y \cup \{y\}$ *infère* x*".*

Étude $1^{ère}$ ***partie : l'appartenance à une classe.***

Posons $Z = Z_1 \cup \ldots \cup Z_n$ *avec* $Z_i =$ *classe d'équivalence (ensemble des formules ayant la même priorité).*

Un algorithme pour co-PBR est le suivant :

[6] Le problème PBR se définit de la manière suivante : soit Z un ensemble de formules stratifié, soient y et x deux formules, Z révisé par y infère-t-il x ? L'opérateur de révision est défini comme suit : l'ensemble "Z révisé par y" $= Cn((\vee(Z \Downarrow \neg y)) \wedge y)$, avec l'opérateur $\Downarrow$ défini informellement par $Z \Downarrow y = \{$ensembles de formules n'impliquant pas y et préférés par rapport à l'ordre induit par la stratification de $Z\}$, la notation $Cn(A)$ signifiant l'ensemble des conséquences logiques de A. Voir [Neb91] pour plus de détail, en particulier pour la définition formelle de $Z \Downarrow y$. Remarquons que l'ordre utilisé dans le calcul de $Z \Downarrow y$ correspond exactement à l'ordre "INCLUSION-BASED" sur les sous-bases consistantes avec y.

1. deviner un sous-ensemble Y de Z
2. vérifier qu'il est préféré pour l'ordre "INCLUSION BASED" et n'infère pas $\neg y$
(c'est à dire que quel que soit Z_i,*
on a $Y_i = Y \cap Z_i$ qui est maximal pour l'inclusion parmi les
*sous-ensembles de Z_i tels que $(\cup_{j \geq i} Y_j)$ n'infère pas $\neg y$ *).*
3. vérifier que $Y \cup \{y\}$ n'infère pas x.

Dans cet algorithme qui est non déterministe (à cause du "deviner") polynomial, on utilise un oracle de complexité NP (même raisonnement que pour FMR).

La complexité de l'algorithme de co-PBR est donc au plus $NP^{NP} = \Sigma_2^p$, donc la complexité de PBR est au plus co-$\Sigma_2^p = \Pi_2^p$.

Étude $2^{ème}$ partie : la complétude ?

On sait que le problème SBR est Π_2^p-complet.

On vient de montrer que le problème PBR est Π_2^p.

D'autre part, il existe une transformation polynomiale évidente du problème SBR dans le problème PBR : on considère la base de connaissances Z du problème SBR comme étant une base avec priorité dans laquelle toutes les formules ont la même priorité ; Z est donc aussi une base pour PBR. On a ainsi SBR $\propto$ PBR.

On en déduit que le problème PBR est Π_2^p-complet.

En conclusion :

Le problème PBR est Π_2^p-complet.

Cas d'une base de clauses de Horn

Théorème A.2.6 *Dans le cas propositionnel avec uniquement des clauses de Horn, le processus de révision appelé PBR est co-NP-complet (il sera noté PBR-HORN).*

Preuve :

Étude $1^{ère}$ partie : l'appartenance à une classe.

L'algorithme pour co-PBR-HORN est le même que celui de co-PBR. Il s'agit d'un algorithme non déterministe (à cause du "deviner") polynomial, dans lequel on utilise un oracle de complexité P.

Le problème co-PBR-HORN appartient donc à la classe de complexité NP, et le problème PBR-HORN appartient à co-NP.

Étude $2^{ème}$ partie : la complétude ?

On sait que le problème SBR-HORN est co-NP-complet.

On vient de montrer que le problème PBR-HORN est co-NP.

D'autre part, il existe une transformation polynomiale évidente du problème SBR-HORN dans le problème PBR-HORN : la même que celle qui permet de passer de SBR à PBR. On a ainsi SBR-HORN $\propto$ PBR-HORN.

On en déduit que le problème PBR-HORN est NP-complet.

A.2.4 Complexité de UBR

Cas général

Théorème A.2.7 *Dans le cas propositionnel général, le processus de révision appelé UBR appartient à la classe de complexité Δ_2^p - (NP $\cup$ co-NP), si NP $\neq$ co-NP.*

Preuve : *(voir [Neb91])*

Il faut calculer la complexité du problème $P : x \in Cn(Z)$ révisé par y en utilisant la méthode UBR[7] *?*

Dans la méthode UBR, le problème "$x \in Cn(Z)$ révisé par y" consiste à prouver que : "soit Z une base stratifiée telle que chaque strate ne contienne qu'une seule formule, quel que soit Y sous-ensemble de Z qui n'infère pas $\neg y$ et qui est préféré pour l'ordre "INCLUSION BASED", alors $Y \cup \{y\}$ infère x".

Étude $1^{ère}$ ***partie : l'appartenance à une classe.***

$Z = Z_1 \cup \ldots \cup Z_n$ *avec* $Z_i = \{z_i\}$ *= singleton (on a une formule par niveau de priorité et pas plus).*

Au pire, UBR aura la même complexité que PBR. On peut toutefois espérer que le fait que chaque classe d'équivalence Z_i soit un singleton simplifie la résolution du problème.

Un algorithme pour UBR est le suivant :

1. initialiser X à $\varnothing$ et i à 1
2. si $X \cup Z_i$ n'infère pas $\neg y$ alors
3. $X \leftarrow X \cup Z_i$
fin si
4. incrémenter i
5. si on a traité tous les niveaux de priorité alors
6. vérifier que $X \cup \{y\}$ infère x
7. sinon recommencer à partir de l'étape 2 de l'algorithme
fin si

Remarque : cet algorithme peut paraître plus compliqué que ceux de SBR ou PBR. Cela vient du fait que le calcul des thèses préférées est intégré directement à l'algorithme, ce qui n'était pas le cas pour les algorithmes précédents. En fait, nous allons voir que sa complexité est moindre.

Ici, on a un algorithme polynomial dans lequel on fait appel à un oracle de complexité NP. On peut donc en déduire que le problème UBR appartient à la classe des problèmes de complexité $P^{NP} = \Delta_2^p$.

Étude $2^{ème}$ ***partie : la complétude ?***

À ce jour, on ne connaît pas de problème Δ_2^p-complet. On ne peut donc pas prouver de manière simple la complétude, si complétude il y a. Par contre, on peut raffiner l'appartenance à la classe Δ_2^p en vérifiant si UBR $\in$ NP ou co-NP.

Pour cela, on va essayer de comparer UBR à des problèmes connus de NP et co-NP. Les problèmes les plus classiques sont GSAT et UNGSAT (satisfaisabilité et insatisfaisabilité d'une formule en logique classique propositionnelle). On constate alors que :

- *le problème GSAT se ramène au problème UBR de manière polynomiale (même raisonnement que pour la démonstration de FMR) :*
  Soit H formule propositionnelle, posons : $Z = \{H\}$, $y =$ TRUE, $x = H$. On a alors :

  $$H \text{ est satisfiable} \Leftrightarrow H \in Cn(H) \text{ révisé par TRUE.}$$

  Remarque : La base $Z = \{H\}$ est bien une base avec priorité non ambiguë car elle ne contient qu'une seule classe d'équivalence représentant la seule formule de la base qui peut être vue comme la formule à la fois la plus prioritaire et la moins prioritaire.
  En conclusion, on a GSAT qui se transforme polynomialement en UBR (noté GSAT $\propto$ UBR).

[7] Le problème UBR est défini de manière identique au problème PBR mais en imposant que la base Z soit stratifiée strictement (une seule formule par strate).

- *le problème UNGSAT (complémentaire de GSAT) se ramène au problème UBR de manière polynomiale (même raisonnement que pour la démonstration de FMR) :*
  Soit H formule propositionnelle, posons : $Z = \varnothing$, $y = H$, $x = \bot$. On a alors :

  $$H \text{ est insatisfiable} \Leftrightarrow \bot \in Cn(\varnothing) \text{ révisé par } H.$$

  Remarque : La base $Z = \varnothing$ est bien une base avec priorité non ambiguë car elle ne contient aucune formule.
  En conclusion, on a UNGSAT $\propto$ UBR.

En reprenant le même raisonnement que pour FMR, on arrive à la conclusion.

En conclusion :

Si NP $\neq$ co-NP, UBR $\in \Delta_2^p$ - (NP $\cup$ co-NP).

Cas d'une base de clauses de Horn

Théorème A.2.8 *Dans le cas propositionnel avec uniquement des clauses de Horn, le processus de révision appelé UBR appartient à la classe de complexité P (il sera noté UBR-HORN).*

Preuve :

L' algorithme pour UBR-HORN est le même que celui de UBR. C'est un algorithme polynomial, dans lequel on utilise un oracle qui est désormais de complexité P puisque le problème de satisfaction d'un ensemble de clauses de Horn est polynomial.

Le problème UBR-HORN est donc dans la classe de complexité P.

Remarques d'ordre général sur les processus de révision et d'inférence

Il existe entre les processus de révision et les processus d'inférence non monotone des liens étroits (voir par exemple [Gär90]). Typiquement, il y a une correspondance entre le problème "Z révisé par y infère-t-il x ?" et le problème "$(E, <)$ infère-t-il x ? avec $(E, <)$ = résultat de l'opération de révision de Z par y".

Par exemple, on voit que le processus SBR correspond exactement au problème UNI-T, que même que le processus PBR correspond au problème UNI-INCL et que UBR correspond au problème 1/STRATE.

Ce sont toutes ces correspondances qui expliquent la présence de processus de révision dans ce document dédié aux processus d'inférence. Elles justifient les similarités d'algorithmes et de démonstrations.

Annexe B

Tableau récapitulatif des transformations polynomiales utilisées

Dans ce chapitre, nous allons récapituler toutes les transformations polynomiales utilisées dans les chapitres 6 et 7 lors des démonstrations de complexité.

Ce récapitulatif va se faire sous la forme d'un tableau défini comme suit :

- *les lignes correspondent aux problèmes Q étudiés dans ce document qui sont :*
  - *soit les relations d'inférence non monotone dont on a calculé la complexité,*
  - *soit les problèmes rencontrés lors des démonstrations d'appartenance de ces relations d'inférence non monotone ;*
- *les colonnes correspondent aux différents cas traités :*
  - *le cas général,*
  - *le cas d'une base stratifiée strictement,*
  - *le cas des formules CNF,*
  - *le cas des clauses de Horn ;*
- *à la croisée des chemins, on trouve alors le ou les problèmes Q' ainsi que la transformation utilisée pour démontrer que $Q' \propto Q$ dans le but de prouver la complétude de Q ou de raffiner l'appartenance de Q à une classe de complexité.*

	Cas propositionnel général	*Cas d'une base stratifiée strictement*	*Cas des formules CNF*	*Cas des clauses de Horn*
UNI-T	Pb : *2-QBF* Ins : *"*$\exists a \forall b G(a,b)$*"* Transformation : - $H = \neg G(a,b)$, - $E = \{a_1, ..., a_n, \neg a_1, ..., \neg a_n, \neg G(a,b)\}$.	*Non concerné par ce cas.*	Pb : *2QBF-DNF* Ins : *"*$\exists a \forall b G(a,b)$*"* Transformation : - $H = \neg G(a,b)$ - $E = \{a_1, ..., a_n, \neg a_1, ..., \neg a_n, \neg G(a,b)\}$.	Pb : *SAT* Ins : *"*C*" avec* $C = \{C_j\}$ *pour* $j = 1 \dots m$ *ensemble de clauses quelconques à satisfaire, et* $V(C) = \{x_1, \dots, x_n\}$ *l'ensemble des variables propositionnelles de* C Transformation : - $H = \neg s$ - $E = \{p, x_1, ..., x_n, y_1, ..., y_n, \neg z_1, ..., \neg z_n, \neg s\}$ *avec* $y_1, \dots, y_n, z_1, \dots, z_n, s$ *nouveaux symboles de variables propositionnelles, et avec la formule* $p = (z_1 \dots z_n \rightarrow s) \wedge \bigwedge_{i=1\dots n}((\neg x_i \vee \neg y_i) \wedge (y_i \rightarrow z_i) \wedge (x_i \rightarrow z_i)) \wedge \bigwedge_{j=1\dots m} C_j[y]\}$*, avec la notation suivante :* $C_j[y]$ *= la clause* C_j *dans laquelle on remplace tous les* x_i *positifs par des* $\neg y_i$
EXI-T	Pb : *2-QBF* Ins : *"*$\exists a \forall b G(a,b)$*"* Transformation : - $H = G(a,b)$, - $E = \{a_1, ..., a_n, \neg a_1, ..., \neg a_n\}$	*Non concerné par ce cas.*	Pb : *co-2QBF-DNF* Ins : *"*$\forall a \exists b G(a,b)$ *avec* $G(a,b)$ *sous-forme CNF"* Transformation : - $H = s$, - $E = \{a_1, ..., a_n, \neg a_1, ..., \neg a_n, G(a,b) \vee s\}$ *avec* s *une nouvelle variable propositionnelle*	Pb : *SAT* Ins : *"*C*" avec* $C = \{C_j\}$ *pour* $j = 1 \dots m$ *ensemble de clauses quelconques à satisfaire, et* $V(C) = \{x_1, \dots, x_n\}$ *l'ensemble des variables propositionnelles de* C Transformation : - $H = s$ - $E = \{p, x_1, ..., x_n, y_1, ..., y_n, \neg z_1, ..., \neg z_n, \neg s\}$ *avec* $y_1, \dots, y_n, z_1, \dots, z_n, s$ *nouveaux symboles de variables propositionnelles, et avec la formule* $p = (z_1 \dots z_n \rightarrow s) \wedge \bigwedge_{i=1\dots n}((\neg x_i \vee \neg y_i) \wedge (y_i \rightarrow z_i) \wedge (x_i \rightarrow z_i)) \wedge \bigwedge_{j=1\dots m} C_j[y]\}$*, avec la notation suivante :* $C_j[y]$ *= la clause* C_j *dans laquelle on remplace tous les* x_i *positifs par des* $\neg y_i$

EXI-T-MIXTE (relation spéciale étudiée pour la démonstration de complétude de ARG-T-CNF)			*Attention !!* *Il ne s'agit pas ici du cas des formules CNF, mais du cas où E est une base de formules CNF et où H est une formule DNF.* Pb : *co-2QBF-DNF* Ins : *"$\forall a \exists b G(a,b)$ avec $G(a,b)$ sous-forme CNF"* Transformation : - $H = s$, - $E = \{a_1, ..., a_n, \neg a_1, ..., \neg a_n, G(a,b) \vee s\}$ *avec s une nouvelle variable propositionnelle*	
ARG-T	Pb : *EXI-T* Ins : *"E, H"* Transformation : - $f(H) = G$ - $f(E) = E \cup \{H \rightarrow G\}$ *avec G nouvelle variable propositionnelle* *(G n'apparaît pas dans E)* Pb : *co-EXI-T* Ins : *"E, H"* Transformation : - $f(H) = \neg H$ - $f(E) = E \cup \{\neg H\}$	*Non concerné par ce cas.*	Pb : *EXI-T-MIXTE* Ins : *"E, H"* Transformation : - $f(H) = G$ - $f(E) = E \cup \{H \rightarrow G\}$ *avec G nouvelle variable propositionnelle* *(G n'apparaît pas dans E)* Pb : *co-EXI-T-MIXTE* Ins : *"E, H"* Transformation : - $f(H) = \neg H$ - $f(E) = E \cup \{\neg H\}$	Pb : *EXI-T-HORN-POSITIF* *(cas particulier de EXI-T-HORN dans lequel H est un littéral positif)* Ins : *"E, H"* Transformation : - $f(H) = G$ - $f(E) = E \cup \{H \rightarrow G\}$ *avec G nouvelle variable propositionnelle* *(G n'apparaît pas dans E)* Pb : *co-EXI-T-HORN-POSITIF* Ins : *"E, H"* Transformation : - $f(H) = \neg H$ - $f(E) = E \cup \{\neg H\}$
UNI-S	*Pas de transformation.*	*Non concerné par ce cas.*	*Pas de transformation.*	*Pas de transformation.*
EXI-S	Pb : *EXI-T* Ins : *"E, H"* Transformation : *identité*	*Non concerné par ce cas.*	Pb : *EXI-T-CNF* Ins : *"E, H"* Transformation : *identité*	Pb : *EXI-T-HORN* Ins : *"E, H"* Transformation : *identité*
ARG-S	Pb : *ARG-T* Ins : *"E, H"* Transformation : *identité*	*Non concerné par ce cas.*	Pb : *ARG-T-CNF* Ins : *"E, H"* Transformation : *identité*	Pb : *ARG-T-HORN* Ins : *"E, H"* Transformation : *identité*
UNI-BO	Pb : *GSAT* Ins : *"G"* Transformation : - $H = G$ - $E = \{G\}$ Pb : *UNGSAT* Ins : *"G"* Transformation : - $H = \neg G$ - $E = \varnothing$	*idem cas général*	Pb : *SAT* Ins : *"G"* Transformation : - $H = G$ - $E = \{G\}$ Pb : *UNSAT* Ins : *"G"* Transformation : - $H = s$ - $E = \{\neg G \rightarrow s\}$ *avec s un nouveau symbole propositionnel*	*Pas de transformation.*

EXI-BO	Pb : *2-QBF,* Ins : *"*$\exists a \forall b G(a,b)$*"* Transformation : - $H = G(a,b)$, - $E = \{a_1, ..., a_n, \neg a_1, ..., \neg a_n\}$	Pb : *EXI-S* Ins : *"*E, H*"* Transformation : - $f(H) = H$ - $(f(E), <) = \{\bot\} \cup E_{strat}$ *avec* $\bot$ *symbolisant la contradiction et constituant* *seul la première strate et* E_{strat} *l'ensemble constitué des formules de* E *ordonnées strictement dans n'importe quel ordre*	Pb : *co-2QBF-DNF* Ins : *"*$\forall a \exists b G(a,b)$ *avec* $G(a,b)$ *sous-forme CNF"* Transformation : - $H = s$, - $E = \{a_1, ..., a_n, \neg a_1, ..., \neg a_n, G(a,b) \vee s\}$ *avec* s *une nouvelle variable propositionnelle*	Pb : *SAT* Ins : *"*C*" avec* $C = \{C_j\}$ *pour* $j = 1 \dots m$ *ensemble de clauses quelconques à satisfaire, et* $V(C) = \{x_1, \dots, x_n\}$ *l'ensemble des variables propositionnelles de* C Transformation : - $H = s$ - $E = \{p, x_1, ..., x_n, y_1, ..., y_n, \neg z_1, ..., \neg z_n, \neg s\}$ *avec* $y_1, \dots, y_n, z_1, \dots, z_n$, s *nouveaux symboles de variables propositionnelles, et avec la formule* $p = (z_1 \dots z_n \rightarrow s) \wedge \bigwedge_{i=1\dots n}((\neg x_i \vee \neg y_i) \wedge (y_i \rightarrow z_i) \wedge (x_i \rightarrow z_i)) \wedge \bigwedge_{j=1\dots m} C_j[y]\}$, *avec la notation suivante :* $C_j[y]$ *= la clause* C_j *dans laquelle on remplace tous les* x_i *positifs par des* $\neg y_i$
EXI-BO-MIXTE (relation spéciale étudiée pour la démonstration de complétude de ARG-BO-CNF)			*Attention !!* *Il ne s'agit pas ici du cas des formules CNF, mais du cas où* E *est une base de formules CNF et où* H *est une formule DNF.* Pb : *co-2QBF-DNF* Ins : *"*$\forall a \exists b G(a,b)$ *avec* $G(a,b)$ *sous-forme CNF"* Transformation : - $H = s$, - $E = \{a_1, ..., a_n, \neg a_1, ..., \neg a_n, G(a,b) \vee s\}$ *avec* s *une nouvelle variable propositionnelle*	

ARG-BO	Pb : *EXI-BO* Ins : "$(E, <), H$" Transformation : - $f(H) = G$ - $f((E, <)) = (E \cup \{H \rightarrow G\}, <)$ *avec* G *nouvelle variable propositionnelle* *(G n'apparaît pas dans $(E, <)$) et la formule $H \rightarrow G$ placée seule en première strate* Pb : *co-EXI-BO* Ins : "$(E, <), H$" Transformation : - $f(H) = \neg H$ - $f((E, <)) = (E \cup \{\neg H\}, <)$ *avec* $\neg H$ *constituant seule la dernière strate*	Pb : *1/STRATE-EXI-BO* Ins : "$(E, <), H$" Transformation : - $f(H) = G$ - $f((E, <)) = (E \cup \{H \rightarrow G\}, <)$ *avec* G *nouvelle variable propositionnelle* *(G n'apparaît pas dans $(E, <)$) et la formule $H \rightarrow G$ placée seule en première strate* Pb : *co-1/STRATE-EXI-BO* Ins : "$(E, <), H$" Transformation : - $f(H) = \neg H$ - $f((E, <)) = (E \cup \{\neg H\}, <)$ *avec* $\neg H$ *constituant seule la dernière strate*	Pb : *EXI-BO-MIXTE* Ins : "$(E, <), H$" Transformation : - $f(H) = G$ - $f((E, <)) = (E \cup \{H \rightarrow G\}, <)$ *avec* G *nouvelle variable propositionnelle* *(G n'apparaît pas dans $(E, <)$) et la formule $H \rightarrow G$ placée seule en première strate* Pb : *co-EXI-BO-MIXTE* Ins : "$(E, <), H$" Transformation : - $f(H) = \neg H$ - $f((E, <)) = (E \cup \{\neg H\}, <)$ *avec* $\neg H$ *constituant seule la dernière strate*	Pb : *EXI-BO-HORN-POSITIF (cas particulier de EXI-T-HORN dans lequel H est un littéral positif)* Ins : "$(E, <), H$" Transformation : - $f(H) = G$ - $f((E, <)) = (E \cup \{H \rightarrow G\}, <)$ *avec* G *nouvelle variable propositionnelle* *(G n'apparaît pas dans $(E, <)$) et la formule $H \rightarrow G$ placée seule en première strate* Pb : *co-EXI-BO-HORN-POSITIF* Ins : "$(E, <), H$" Transformation : - $f(H) = \neg H$ - $f((E, <)) = (E \cup \{\neg H\}, <)$ *avec* $\neg H$ *constituant seule la dernière strate*
UNI-INCL	Pb : *UNI-T* Ins : "E, H" Transformation : *identité* *(considérons toutes les formules de E en une seule strate)*	*Une seule relation d'inférence pour UNI (EXI, ARG) - INCL (LEX) notée 1/STRATE :* Pb : *ALM (cf. [EG93])* Ins : "$C = \{C_1, \ldots, C_m\}$ *satisfiable,* $X = \{x_1, \ldots, x_n\}$, $O(X) =< x_1, \ldots, x_n >$" Transformation : - $H = x_n$ - $(E, <) = \{C_1 \wedge \ldots \wedge C_m, x_1, \ldots, x_n\}$ *l'ordre $<$ étant le suivant : la formule $CC = C_1 \wedge \ldots \wedge C_m$ est plus prioritaire que la formule x_1 elle-même plus prioritaire que la formule x_2 elle-même plus prioritaire que …plus prioritaire que x_n.*	Pb : *UNI-T-CNF* Ins : "E, H" Transformation : *identité* *(considérons toutes les formules de E en une seule strate)*	Pb : *UNI-T-HORN* Ins : "E, H" Transformation : *identité* *(considérons toutes les formules de E en une seule strate)*
EXI-INCL	Pb : *EXI-T* Ins : "E, H" Transformation : *identité* *(considérons toutes les formules de E en une seule strate)*	*(voir dans UNI-INCL)*	Pb : *EXI-T-CNF* Ins : "E, H" Transformation : *identité* *(considérons toutes les formules de E en une seule strate)*	Pb : *EXI-T-HORN* Ins : "E, H" Transformation : *identité* *(considérons toutes les formules de E en une seule strate)*

ARG-INCL	Pb : *ARG-T* Ins : *"E, H"* Transformation : *identité* *(considérons toutes les formules de E en une seule strate)*	*(voir dans UNI-INCL)*	Pb : *ARG-T-CNF* Ins : *"E, H"* Transformation : *identité* *(considérons toutes les formules de E en une seule strate)*	Pb : *ARG-T-HORN* Ins : *"E, H"* Transformation : *identité* *(considérons toutes les formules de E en une seule strate)*
UNI-LEX	Pb : *UNI-CAR* Ins : *"E, H"* Transformation : *identité* *(considérons toutes les formules de E en une seule strate)*	*(voir dans UNI-INCL)*	Pb : *UNI-CAR-CNF* Ins : *"E, H"* Transformation : *identité* *(considérons toutes les formules de E en une seule strate)*	Pb : *ACM* Ins : *"$C = \{C_1, \ldots, C_m\}$ un ensemble de clauses ($X = \{x_1, \ldots, x_n\}$ les variables de C), $k = \{1, \ldots, m\}$ un entier"* Transformation : - $H = C_k[y] \wedge s$ - $(E, <) = \{p_1, \ldots, p_m\} \cup \{x_1, \ldots, x_n, y_1, \ldots, y_n\} \cup \{\neg z_1, \ldots, \neg z_n, \neg s\}$ *avec* $y_1, \ldots, y_n$, $z_1, \ldots, z_n$, s *nouveaux symboles de variables propositionnelles, et avec les formules* $p_j = (z_1 \ldots z_n \rightarrow s) \wedge \bigwedge_{i=1\ldots n}((\neg x_i \vee \neg y_i) \wedge (y_i \rightarrow z_i) \wedge (x_i \rightarrow z_i)) \wedge C_j[y]$, *avec la notation suivante : $C_j[y]$ = la clause C_j dans laquelle on remplace tous les littéraux positifs x_i par des littéraux négatifs $\neg y_i$. L'ordre $<$ entre les formules de la base est le suivant : les p_j plus prioritaires que les x_i et les y_i, eux-mêmes plus prioritaires que les $\neg z_i$ et le $\neg s$*
EXI-LEX	Pb : *EXI-CAR* Ins : *"E, H"* Transformation : *identité* *(considérons toutes les formules de E en une seule strate)*	*(voir dans UNI-INCL)*	Pb : *EXI-CAR-CNF* Ins : *"E, H"* Transformation : *identité* *(considérons toutes les formules de E en une seule strate)*	*non étudié*
ARG-LEX	Pb : *ARG-CAR* Ins : *"E, H"* Transformation : *identité* *(considérons toutes les formules de E en une seule strate)*	*(voir dans UNI-INCL)*	Pb : *ARG-CAR-CNF* Ins : *"E, H"* Transformation : *identité* *(considérons toutes les formules de E en une seule strate)*	*non étudié*

MAX-GSAT	Pb : *GSAT* Ins : *"G"* Transformation : - $k = 1$ - $E = \{G\}$	*Non concerné par ce cas.*	Pb : *SAT* Ins : *"G"* Transformation : - $k = 1$ - $E = \{G\}$	*Pour la version multi-ensemble :* Pb : *MAX-2-SAT* Ins : *"C, k"* *avec C collection de n clauses à 2 littéraux maximum et k entier $\leq n$ (nombre de clauses de C)* Transformation : - $f(k) = k + p$ *avec* $p = \|C \setminus CH\|$ - $E = CH \cup CNH$ *avec* $CH = \{$*les clauses de C qui sont déjà sous forme de clauses de Horn*$\}$ *et* $CNH = \{$*les clauses définies de la manière suivante :* $\forall c$ *clause de C de la forme* $\rightarrow a_1 a_2$ ($c \in C \setminus CH$), *les clauses de HORN* $\rightarrow a_1$, $\rightarrow a_2$, $a_1 a_2 \rightarrow \in CNH\}$
NGSAT-CAR	Pb : *GSAT* Ins : *"G"* Transformation : - $E = \{G\}$ - $H = \neg G$ - $k = 1$	*Non concerné par ce cas.*	Pb : *SAT* Ins : *"G"* Transformation : - $E = \{G\}$ - $H = \neg G$ - $k = 1$	*non étudié*
MAX-GSAT-STRICT	Pb : *GSAT* Ins : *"G"* Transformation : - $k = 0$ - $E = \{G\}$	*Non concerné par ce cas.*	Pb : *SAT* Ins : *"G"* Transformation : - $k = 0$ - $E = \{G\}$	*Pour la version multi-ensemble :* Pb : *MAX-SAT-ME-HORN* Ins : *"E, k"* Transformation : - $f(k) = k - 1$ - $f(E) = E$ *(donc une seule strate)*
MAX-GSAT-ARRAY	Pb : *MAX-GSAT* Ins : *"E, k"* Transformation : - $f(k) =< k >$ - $f(E) = E$ *avec* $f(E)$ *une base stratifiée constituée d'une seule strate*	*Non concerné par ce cas.*	*idem cas général*	*Pour la version multi-ensemble :* Pb : *MAX-SAT-ME-HORN* Ins : *"E, k"* Transformation : - $f(k) =< k >$ - $f(E) = E$ *avec* $f(E)$ *une base stratifiée constituée d'une seule strate*
NGSAT-LEX	Pb : *NGSAT-CAR* Ins : *"E, H, k"* Transformation : - $f(E) = E$ *avec* $f(E)$ *une base stratifiée constituée d'une seule strate* - $f(H) = H$ - $f(k) =< k >$	*Non concerné par ce cas.*	*idem cas général*	*non étudié*
UNI-CAR	Pb : *ACM* Ins : *"$C = \{C_1, \ldots, C_m\}$, $k \in \{1, \ldots, m\}$"* Transformation : - $H = C_k$ - $E = \{C_1, \ldots, C_m\}$	*Non concerné par ce cas.*	Pb : *ACM* Ins : *"$C = \{C_1, \ldots, C_m\}$, $k \in \{1, \ldots, m\}$"* Transformation : - $H = C_k$ - $E = \{C_1, \ldots, C_m\}$	*Pour la version multi-ensemble :* Pb : *"UNI-LEX-HORN"* Transformation : - $f(H) = H$ - $f(E) =$ *le résultat de l'algorithme donné en section 7.3.6*

EXI-CAR	Pb : *2-QBF* Ins : *"*$\exists a \forall b G(a,b)$*"* Transformation : - $H = G(a,b)$ - $E = \{a_1, ..., a_n, \neg a_1, ..., \neg a_n\}$	*Non concerné par ce cas.*	Pb : *co-2QBF-DNF* Ins : *"*$\forall a \exists b G(a,b)$ *avec* $G(a,b)$ *sous-forme CNF"* Transformation : - $H = s$ - $E = \{a_1, ..., a_n, \neg a_1, ..., \neg a_n, G(a,b) \vee s\}$ *avec* s *une nouvelle variable propositionnelle*	*non étudié*
EXI-CAR-MIXTE *(relation spéciale étudiée pour la démonstration de complétude de ARG-CAR-CNF)*			*Attention !!* *Il ne s'agit pas ici du cas des formules CNF, mais du cas où* E *est une base de formules CNF et où* H *est une formule DNF.* Pb : *co-2QBF-DNF* Ins : *"*$\forall a \exists b G(a,b)$ *avec* $G(a,b)$ *sous-forme CNF"* Transformation : - $H = s$ - $E = \{a_1, ..., a_n, \neg a_1, ..., \neg a_n, G(a,b) \vee s\}$ *avec* s *une nouvelle variable propositionnelle*	
ARG-CAR	Pb : *EXI-CAR* Ins : *"*E, H*"* Transformation : - $f(H) = G$ - $f(E) = E \cup \{H \rightarrow G\}$ *avec* G *nouvelle variable propositionnelle* *(*G *n'apparaît pas dans* E*)*	*Non concerné par ce cas.*	Pb : *EXI-CAR-MIXTE* Ins : *"*E, H*"* Transformation : - $f(H) = G$ - $f(E) = E \cup \{H \rightarrow G\}$ *avec* G *nouvelle variable propositionnelle* *(*G *n'apparaît pas dans* E*)*	*non étudié*
UNI-E	*(voir [Got92])*	*Non concerné par ce cas.*	*non étudié*	*(voir [KS91])*
EXI-E	*(voir [Got92])*	*Non concerné par ce cas.*	*non étudié*	*(voir [KS91] et [Sti90])*
ARG-E	Pb : *EXI-E* Ins : *"*$E = (W, D), H$*"* Transformation : - $f(H) = G$ - $f(E) = (W \cup \{H \rightarrow G\}, D)$ *avec* G *nouvelle variable propositionnelle* *(*G *n'apparaît pas dans* W*, ni dans* D*)* Pb : *co-EXI-E* Ins : *"*$E = (W, D), H$*"* Transformation : - $f(H) = \neg H$ - $f(E) = (W, D \cup \{: \neg H / \neg H\}$	*Non concerné par ce cas.*	*non étudié*	Définition de Stillman : *(voir [Sti90])* Pb : *co-EXI-E-HORN* Ins : *"*$E = (W, D), H$*"* Transformation : - $f(H) = \neg H$ - $f(E) = (W, D \cup \{: \neg H / \neg H\}$

Bibliographie

[BCD+ 93] *Benferhat (Salem), Cayrol (Claudette), Dubois (Didier), Lang (Jérôme) et Prade (Henri). – Inconsistency management and prioritized syntax-based entailment.* In : Proc. of the 13^{th} IJCAI, *éd. par Bajcsy (Ruzena). pp. 640–645. – Chambéry, France, 1993.*

[BDLP92] *Benferhat (Salem), Dubois (Didier), Lang (Jérôme) et Prade (Henri). – Hypothetical reasoning in possibilistic logic : basic notions and implementations issues.* In : Advances in Fuzzy Systems : Applications and Theory - vol 1, *éd. par Wang (P.Z.) et K.F.Loe. – World Scientific Publishing Comp., 1992.*

[BDP92] *Benferhat (Salem), Dubois (Didier) et Prade (Henri). – Representing default rules in possibilistic logic.* In : Proc. of the 3^{rd} KR, *éd. par Allen (J.), Fikes (R.) et Sandewall (E.). pp. 671–684. – Cambridge, MA, 1992.*

[BDP93] *Benferhat (Salem), Dubois (Didier) et Prade (Henri). – Argumentative inference in uncertain and inconsistent knowledge bases.* In : Proc. of the 9^{th} UAI, *éd. par Heckerman (David) et Mamdani (Abe). pp. 411–419. – Washington, DC, 1993.*

[BDP94] *Benferhat (Salem), Dubois (Didier) et Prade (Henri). – Représentation des connaissances conditionnelles.* In : Proc. of the 3^{rd} Maghrebian Conference on Software Engineering and Artificial Intelligence MCSEAI, *éd. par IFIP et AFCET, pp. 79–88. – Rabat, Marocco, 1994.*

[Bre89a] *Brewka (Gerhard). – Nonmonotonic logics – a brief overview.* In : AICOM, 2 (vol. 2), *pp. 88–97.*

[Bre89b] *Brewka (Gerhard). – Preferred subtheories : An extended logical framework for default reasoning.* In : Proc. of the 11^{th} IJCAI, *éd. par Sridharan (N.S.). pp. 1043–1048. – Detroit, MI, 1989.*

[Cay90] *Cayrol (Claudette). –* Assumption-Based Reasoning : A Uniform Framework for Non-Monotonic Reasoning. *– Rapport de recherche n° 90-29R, Institut de Recherche en Informatique de Toulouse (I.R.I.T.), novembre 1990.*

[Cay92] *Cayrol (Claudette). – Un modèle logique général pour le raisonnement révisable.* Revue d'Intelligence Artificielle, *vol. 6, n° 3, 1992, pp. 255–284.*

[CRS92] *Cayrol (Claudette), Royer (Véronique) et Saurel (Claire). – Management of preferences in assumption-based reasoning.* In : Advanced methods in AI. Lecture notes in computer science 682, *éd. par Yager (R.) et Bouchon (B.), pp. 13–22. – Springer Verlag, 1992. Extended version in Technical Report IRIT-CERT, 92-13R (University Paul Sabatier Toulouse).*

[DLP91] *Dubois (Didier), Lang (Jérôme) et Prade (Henri). – Inconsistency in possibilistic knowledge bases - to live or not to live with it.* In : Fuzzy logic for the Management of Uncertainty, *éd. par Zadeh (L.A.) et Kacprzyk (J.), pp. 335–351. – Wiley and sons, 1991.*

[EG92] *Eiter (Thomas) et Gottlob (Georg). – On the complexity of propositional knowledge base revision, updates, and counterfactuals.* Artificial Intelligence, *vol. 57, 1992, pp. 227–270.*

[EG93] *Eiter (Thomas) et Gottlob (Georg). – The complexity of logic-based abduction.* In : Proc. of the 10^{th} Symposium on Theoretical Aspects of Computing STACS, *éd. par Enjalbert (P.), Finkel (A.) et Wagner (K. W.). pp. 70–79. – Würzburg, Germany, 1993. Long version to appear in* Journal of the ACM.

[Gär90] *Gärdenfors (Peter). – Belief revision and nonmonotonic logic : Two sides of the same coin.* In : Proc. of the 9^{th} ECAI, *éd. par Aiello (Luigia Carlucci). pp. 768–773. – Stockhlom, Sweden, 1990.*

[Gär91] *Gärdenfors (Peter). – Nonmonotonic inferences based on expectations: A preliminary report.* In : Proc. of the 2^{nd} KR, *éd. par Allen (J.), Fikes (R.) et Sandewall (E.). pp. 585–590. – Cambridge, MA, 1991.*

[GJ79] *Garey (Michael R.) et Johnson (David S.).* – Computers and Intractability : A Guide to the Theory of NP-completeness. – *New York, W.H. Freeman and Company, 1979.*

[GM94] *Gärdenfors (Peter) et Makinson (David). – Nonmonotonic inference based on expectations.* Artificial Intelligence, *vol. 65, 1994, pp. 197–245.*

[Got92] *Gottlob (Georg). – Complexity results for nonmonotonic logics.* Journal of Logic and Computation, *vol. 2, n° 3, 1992, pp. 397–425.*

[Joh90] *Johnson (David S.). – A catalog of complexity classes.* In : Handbook of Theoretical Computer Science, *éd. par van Leeuwen (Jan), chap. 2, pp. 67–161. – Elsevier, 1990.*

[KLM90] *Kraus (Sarit), Lehmann (Daniel) et Magidor (Menachem). – Nonmonotonic reasoning, preferential models and cumulative logics.* Artificial Intelligence, *vol. 44, 1990, pp. 167–207.*

[KS91] *Kautz (Henry A.) et Selman (Bart). – Hard problems for simple default logics.* Artificial Intelligence, *vol. 49, 1991, pp. 243–279.*

[Leh92] *Lehmann (Daniel).* – Another Perspective on Default Reasoning. – *Rapport de recherche n° 92-12, Israel, Leibniz Center for Research in Computer Science. Hebrew University of Jerusalem, juillet 1992.*

[LM92] *Lehmann (Daniel) et Magidor (Menachem). – What does a conditional knowledge base entail ?* Artificial Intelligence, *vol. 55, 1992, pp. 1–60.*

[LMS92] *Levesque (Hector), Mitchell (David) et Selman (Bart). – A new method for solving hard satisfiability problems.* In : Proc. of AAAI-92, *pp. 440–446. – San Jose, CA, 1992.*

[Neb91] *Nebel (Bernhard). – Belief revision and default reasoning: Syntax-based approaches.* In : Proc. of the 2^{nd} KR, *éd. par Allen (J.A.), Fikes (R.) et Sandewall (E.). pp. 417–428. – Cambridge, MA, 1991.*

[Pea90] *Pearl (Judea). – System Z : A natural ordering of defaults with tractable applications to default reasoning.* In : Proc. of the 3^{rd} Conference of Theoretical Aspects of Reasoning about Knowledge, *éd. par Vardi (M.). pp. 121–135. – Morgan-Kaufmann.*

[PL92] *Pinkas (Gadi) et Loui (Ronald P.). – Reasoning from inconsistency : A taxonomy of principles for resolving conflict.* In : Proc. of the 3^{rd} KR, *éd. par Allen (J.A.), Fikes (R.) et Sandewall (E.). pp. 709–719. – Cambridge, MA, 1992.*

[Poo88] *Poole (David). – A logical framework for default reasoning.* Artificial Intelligence, *vol. 36, 1988, pp. 27–47.*

[Rei80] *Reiter (Raymond). – A logic for default reasoning.* Artificial Intelligence, *vol. 13, n° 1-2, 1980, pp. 81–132.*

[Sti90] *Stillman (Jonathan). – It's not my default : The complexity of membership problems in restricted propositional default logics.* In : Proc. of the 8^{th} AAAI*, éd. par Press (MIT). pp. 571–578. – Boston, MA, 1990.*

[Sto77] *Stockmeyer (Larry J.). – The polynomial-time hierarchy.* Theoretical Computer Science*, vol. 3, 1977, pp. 1–22.*