Journées d'Intelligence Artificielle Fondamentale 2008 (IAF'08)

21-23 Novembre 2008 - Paris Dauphine

Actes électroniques

Ces journées sont la principale animation du thème Intelligence Artificielle Fondamentale (thème 1) du GDR Information-Interaction-Intelligence.

Les journées étaient composées d'exposés de synthèse et de contributions sélectionnées.

Contributions sélectionnées

Exposés de synthèse

Comité de Programme

- Laurence Cholvy (ONERA, Toulouse)
- Marie-Odile Cordier (IRISA, Rennes)
- Philippe Dague (LRI, Paris)
- Didier Dubois (IRIT, Toulouse)
- Laurent Garcia (LERIA, Angers)
- Robert Jeansoulin (IGM, Marne la Vallée)
- Daniel Kayser (LIPN, Paris)
- Sébastien Konieczny (CRIL, Lens)
- Jérôme Lang (LAMSADE, Paris)
- Pierre Marquis (CRIL, Lens)
- Nicolas Maudet (LAMSADE, Paris)
- Marie Laure Mugnier (LIRMM, Montpellier)
- Amedeo Napoli (LORIA, Nancy)
- Pascal Nicolas (LERIA, Angers)
- Odile Papini (LSIS, Marseille)
- Henri Prade (IRIT, Toulouse)
- Marie Christine Rousset (LIG, Grenoble)
- Lakhdar Sais (CRIL, Lens)
- Thomas Schiex (INRA, Toulouse)
- Michèle Sebag (LRI, Paris)
- Pierre Siegel (LSIS, Marseille)
- Laurent Simon (LRI, Paris)

Comité d'Organisation

- Nicolas Maudet (Président)
- Fabien Badeig
- Flavien Balbo
- Gauvain Bourgne
- Tristan Cazenave
- Yann Chevaleyre
- Wassila Ouerdane
- Guillaume Ravilly-Abadie

Vers une pertinence orientée agent des informations ⋆

Stéphanie Roussel[1] Laurence Cholvy[1]

ONERA, Centre de Toulouse
2, avenue Edouard Belin, 31055 Toulouse
Stephanie.Roussel@onera.fr

Résumé : Dans ce papier, on s'intéresse à la définition d'une pertinence orientée-agent. Pour cela, on définit un nouvel opérateur modal R_a^Q tel que $R_a^Q\varphi$ signifie qu'une information φ est pertinente pour l'agent a vis-à-vis d'une requête Q. On discute les propriétés de ce nouvel opérateur, ainsi que ses extensions et limites.

Mots-clés : Pertinence d'une information, système multi-agent, besoin en information

1 Introduction

Ce travail se place dans un cadre de modélisation des systèmes multi-agents dans lesquels des entités doivent coopérer de façon à réaliser une tâche qu'à priori aucune d'elles ne pourrait réaliser seule. Dans un tel contexte, ces entités qui coopèrent (que l'on appelle désormais agents), doivent communiquer, c'est-à-dire échanger des informations, par exemple pour avoir une vue globale de leur environnement. Cependant, si un agent reçoit l'ensemble de toutes les informations que les autres possèdent, il sera difficile pour lui de trouver dans cet ensemble les informations dont il a réellement besoin. Dans un souci d'efficacité, il faudrait donc que seules les informations dont les agents ont réellement besoin soient échangées, c'est-à-dire les informations qui sont pertinentes pour leurs besoins.

Dans ce papier, on propose une caractérisation formelle de la pertinence des informations dans le contexte d'échange d'informations.

D'après Borlund (2003), deux approches existent pour le concept de pertinence : la pertinence orientée système et la pertinence orientée agent. La première analyse la pertinence en terme de topicalité, d'à-propos, de degrés de ressemblance entre une information et une requête. La pertinence orientée système est au centre du domaine de la Recherche d'Informations (Chevallet (2004); Crestani & Lalmas (2001); Lalmas & van Rijsbergen (1993)). La deuxième approche définit une relation entre une information et l'agent qui la reçoit et analyse donc la pertinence en terme d'utilité, d'informativité pour

⋆la thèse pendant laquelle ce travail a été effectué est financée par la DGA.

l'agent ... le but étant de définir une information pertinente pour un agent en fonction de son besoin en information. Selon Floridi (2007), il n'existe pas de définition pour la pertinence orientée agent. Floridi a proposé une interprétation de la pertinence épistémique basée sur l'analyse du degré de pertinence de la sémantique d'une information pour un agent rationnel qui la reçoit. Cette information est en fait considérée comme une réponse à une question, étant donnée la probabilité que cette question soit posée par l'agent qui reçoit l'information. Floridi s'intéresse en particulier à la précision de l'information, étant donnée la question à laquelle elle est associée.

Dans ce papier, comme Floridi, notre but est de contribuer à l'étude de la pertinence orientée agent. Pour cela, on utilise un modèle d'agent relativement classique à savoir le modèle BDI *belief-desire-intention* (Wooldridge (2000)). Ce modèle suppose que les agents sont caractérisés par des attitudes mentales, à savoir principalement la croyance, le désir et l'intention. La majorité des modèles basés sur BDI sont des logiques modales dont les opérateurs modaux sont utilisés pour représenter les différentes attitudes mentales. La sémantique de ces opérateurs est généralement donnée par la sémantique des mondes possibles (Chellas (1980)). En utilisant ce type de formalisme, notre but est de définir un nouvel opérateur pour représenter la pertinence orientée agent d'une information.

Ce papier est organisé de la façon suivante : la partie 2 présente le formalisme logique sur lequel on base notre travail. L'opérateur pertinence est défini dans la partie 3. On en étudie également diverses propriétés. Dans la partie 4, on propose des hiérarchies pour caractériser les informations les plus pertinentes. Finalement, on conclut dans la partie 5.

2 Outils de formalisation

On se base sur le cadre logique défini dans Herzig & Longin (2002), qui est une logique multi-modale propositionnelle dont les modalités sont la croyance et l'intention.

Soit $\mathcal{A}$ l'ensemble des agents.

- **Croyance** A chaque agent a de $\mathcal{A}$, on associe un opérateur de croyance B_a. La formule $B_a\varphi$ se lit "L'agent a croit que φ". On écrira $Bif_a\varphi$ à la place de $B_a\varphi \vee B_a\neg\varphi$. $Bif_a\varphi$ se lit "L'agent a sait [1] si φ". Classiquement, on utilise la logique KD45 (Chellas (1980)) pour modéliser la croyance. Les règles d'inférence sont :
  - **Modus Ponens** : $\frac{\varphi\ \varphi\rightarrow\psi}{\psi}$
  - **Nécessitation** : $\frac{\varphi}{B_a(\varphi)}$

  et les axiomes :
  - **K** : $B_a(\varphi \rightarrow \psi) \rightarrow (B_a(\varphi) \rightarrow B_a(\psi))$
  - **D** : $B_a(\varphi) \rightarrow \neg B_a(\neg\varphi)$
  - **4** : $B_a(\varphi) \rightarrow B_a(B_a(\varphi))$
  - **5** : $\neg B_a(\varphi) \rightarrow B_a(\neg B_a(\varphi))$

  On suppose donc que les agents n'ont pas de croyances incohérentes (D) et qu'ils sont conscients de leurs croyances (4) ainsi que de leurs non-croyances (5).

[1] On utilise ici le verbe "savoir" au lieu de croire car, dans le langage courant on ne dit pas qu'un agent croit si une information est vraie ou fausse.

– **Intention** A chaque agent a de $\mathcal{A}$, on associe un opérateur I_a. La formule $I_a\varphi$ se lit "L'agent a a l'intention que φ". Pour cet opérateur, on a seulement : $\frac{\varphi \leftrightarrow \psi}{I_a\varphi \leftrightarrow I_a\psi}$
– **Relation entre croyance et intention** Comme dans Herzig & Longin (2002), on suppose qu'il y a réalisme fort entre intention et croyance, c'est-à-dire que si un agent a l'intention qu'une formule soit vraie, alors c'est qu'il croit que cette formule est fausse. Cette hypothèse peut être exprimée ainsi : $I_a\varphi \rightarrow B_a\neg\varphi$. On peut alors déduire la relation de réalisme faible : $I_aA \rightarrow \neg B_aA$ (si un agent a l'intention qu'une formule soit vraie, c'est qu'il ne croit pas que celle-ci est vraie). On montre également que $B_aA \rightarrow \neg I_aA$; $\neg B_aA \rightarrow \neg I_a\neg B_aA$.
De plus, on suppose les introspections positive et négative des intentions, c'est-à-dire $I_aA \rightarrow B_aI_aA$ et $\neg I_aA \rightarrow B_a\neg I_aA$

Dans toute la suite, on appelle *requête* une formule sans opérateur modal.

3 Pertinence

3.1 Définition

Dans cette partie, on définit un nouvel opérateur R_a^Q. $R_a^Q\varphi$ signifie que la formule φ est pertinente pour l'agent a par rapport à la requête Q. On remarque ici que la pertinence dépend d'un agent. En effet, dans le cadre d'échange d'informations, on considère qu'une information ne peut pas être pertinente dans l'abolu mais doit être associée à un agent.

Avant de donner la définition formelle de cet opérateur, on note $\otimes$ la disjonction exclusive généralisée à n formules, c'est-à-dire que si $\varphi_1, ...\varphi_n$ sont n formules alors $\varphi_1 \otimes ... \otimes \varphi_n$ est vraie si et seulement si $\varphi_1 \vee ... \vee \varphi_n$ est vraie et $\forall i, j\ (\varphi_i \wedge \varphi_j)$ est fausse.

Définition 1
Soit a un agent de $\mathcal{A}$, φ une formule et Q une requête.

$$R_a^Q\varphi \equiv I_aBif_aQ \wedge (B_a(\varphi \rightarrow Q) \otimes B_a(\varphi \rightarrow \neg Q)) \wedge \varphi$$

Dans cette définition, trois éléments apparaissent.
– **Le besoin en information d'un agent** I_aBif_aQ : On suppose que les agents qui échangent des informations ont des besoins en information. De plus, on suppose qu'un besoin en information est relativement simple et qu'il peut être modélisé de la façon suivante : l'agent a veut savoir si Q ou si $\neg Q$, Q étant une requête. [2]. Formellement, le besoin en information s'écrit donc I_aBif_aQ, c'est-à-dire l'agent a a l'intention de savoir si Q.
– **Les croyances de l'agent** $B_a(\varphi \rightarrow Q) \otimes B_a(\varphi \rightarrow \neg Q)$: A partir de ses croyances et de l'information φ, l'agent doit être capable de répondre à son besoin en information Q, c'est-à-dire qu'il peut déduire soit Q soit $\neg Q$. Pour représenter cette déduction, on choisit ici l'implication logique.
Pour ce concept de croyance dans la notion de pertinence, deux points sont à noter :

[2] On ne s'intéresse pas dans cet article à la façon dont on passe du besoin en information à la (ou aux) requête(s)

  - L'agent doit obligatoirement utiliser ses croyances pour répondre à son besoin en information.
  - Si un agent, à partir d'une information φ, peut déduire à la fois Q et $\neg Q$, alors φ ne permet pas vraiment de répondre au besoin en information. L'utilisation de $\otimes$ permet d'éviter ce cas. [3]
- **La valeur de vérité de l'information** φ : On considère ici qu'une information fausse ne peut pas être pertinente. La fausse information, bien qu'ayant un sens, est fausse. Si on analyse le pertinence épistémique en terme d'efforts cognitifs, la mésinformation est délétère. Par exemple, considérons un agent qui veut prendre le train pour Paris. Ce train doit partir à 13h05. Dans ce contexte, communiquer à l'agent le fait que le train part à 13h15 est nuisible (car il risque de rater son train). On ne peut donc pas considérer que l'information "le train part à 13h15" est pertinente pour l'agent.[4]

L'exemple qui suit permet d'illustrer ces différents éléments.

Exemple 1
Considérons trois agents a, b et c qui doivent prendre le train. Hélas, des incidents (modélisé par $incident$) en gare peuvent bloquer les trains et donc les retarder. Ainsi, les agents veulent savoir si leur train est en retard (modélisé par $retard$) ou non. Ces agents ont donc le même besoin en information que l'on peut respectivement formaliser par $I_aBif_a(retard)$, $I_bBif_b(retard)$ et $I_cBif_c(retard)$ pour a, b et c.

Supposons que les croyances des agents soient différentes.

- $B_a(incident \rightarrow retard)$,
- $B_b(\neg incident \rightarrow \neg retard)$,
- $B_c(incident \leftrightarrow retard)$.

Considérons l'information $incident$. Si cette information est vraie, alors les agents a et c peuvent en déduire que leur train est en retard. L'agent b, quant à lui, n'a pas les croyances suffisantes pour déduire quoi que ce soit sur le retard de son train. L'information $incident$ est pertinente pour a et c mais pas pour b. Les agents utilisent donc leur croyances pour répondre à leurs besoins en information. De la même façon, si $\neg incident$ est une information vraie, alors elle est pertinente pour b et c.

Par contre, la formule $incident$ est considérée comme non pertinente pour a car, certes elle permettrait de répondre à son besoin en information mais elle l'induirait en erreur.

3.2 Propriétés

Dans cette sous-partie, on énonce quelques propriétés de l'opérateur pertinence défini précédemment. La liste des propriétés n'est bien sûr pas exhaustive. On considère a

[3] Le $\otimes$ permet également de ne pas s'occuper du cas où l'agent croyait $\neg\varphi$. En effet, dans ce cas, l'agent, à partir de φ peut déduire n'importe quoi.

[4] Cette pertinence peut être caractérisée comme contingente. La pertinence dans laquelle on ne considère pas la valeur de vérité de l'information est également intéressante à étudier. Elle peut être qualifiée de pertinence intrinsèque. Dans cet article, on se restreint à la pertinence contingente.

un agent de $\mathcal{A}$, Q, Q_1 et Q_2 sont des requêtes, φ, φ_1 et φ_2 sont des formules. Les propositions qui suivent sont des théorèmes de notre logique [5].

Proposition 1
$R_a^Q\varphi \rightarrow \neg B_a\varphi \wedge \neg B_a\neg\varphi$

Si une information φ est pertinente pour un agent a, alors a ne croit ni φ (sinon il pourrait déjà répondre à son besoin en information), ni $\neg\varphi$ (sinon il aurait une base de croyances inconsistante).

Proposition 2
Soit $*$ un opérateur de révision de croyances satisfaisant les postulats AGM Alchourrón et al. *(1985). On note Bel_a l'ensemble de croyances de l'agent a et $Bel_a * \varphi$ l'ensemble des croyances de a après révision de Bel_a par φ avec l'opérateur de révision $*$. On a alors $R_a^Q\varphi \rightarrow ((Bel_a * \varphi) \rightarrow Q) \otimes ((Bel_a * \varphi) \rightarrow \neg Q)$*

Cette proposition montre que l'opérateur de déduction choisi, l'implication, correspond à un opérateur "basique" de révision de croyance. En effet, en révisant sa base de croyances avec l'information pertinente, l'agent peut, dans sa nouvelle base, répondre à son besoin en information.

Proposition 3
- $I_aBif_aQ \rightarrow R_a^QQ \otimes R_a^Q\neg Q$ *: une des informations Q ou $\neg Q$ est pertinente pour un besoin en information Q.*
- $(Q_1 \leftrightarrow Q_2) \rightarrow (R_a^{Q_1}\varphi \leftrightarrow R_a^{Q_2}\varphi)$ *: une information pertinente pour une requête est également pertinente pour une requête équivalente à la première.*
- $R_a^Q\varphi \leftrightarrow R_a^{\neg Q}\varphi$ *: une information pertinente pour une requête Q est également pertinente pour la requête $\neg Q$.*
- $\neg(\varphi_1 \wedge \varphi_2) \rightarrow \neg(R_a^{Q_1}\varphi_1 \wedge R_a^{Q_2}\varphi_2)$ *: deux informations contradictoires ne peuvent pas être pertinentes simultanément.*

Proposition 4
$R_a^Q\varphi \rightarrow \neg B_aR_a^Q\varphi$

Si une information φ est pertinente pour un agent a alors a ne le sait pas. En effet, la valeur de vérité est contenue dans la pertinence. Si l'agent croit que l'information est pertinente, il croit donc cette information. S'il croit cette information, alors il possède les croyances pour déduire à partir de celle-ci une réponse à son besoin en information. Or si l'agent peut par lui-même répondre à son besoin en information, c'est que ce besoin n'existe plus.

Notation. Dans ce qui suit, on notera $B_a(\varphi_1, \varphi_2/Q)$ au lieu de $\neg(B_a(\varphi_1 \rightarrow Q) \wedge B_a(\varphi_2 \rightarrow \neg Q)) \wedge \neg(B_a(\varphi_1 \rightarrow \neg Q) \wedge B_a(\varphi_2 \rightarrow Q))$. Cette formule signifie que a croit que φ_1 et φ_2 ne permettent pas de déduire de contradiction par rapport à Q.

[5] Pour alléger, on omettra le symbole $\vdash$ devant les théorèmes.

Proposition 5
$B_a(\varphi_1, \varphi_2/Q) \rightarrow (\varphi_2 \wedge R_a^Q \varphi_1 \rightarrow R_a^Q(\varphi_1 \wedge \varphi_2))$

Proposition 6
$B_a(\varphi_1, \varphi_2/Q) \rightarrow (R_a^Q \varphi_1 \wedge R_a^Q \varphi_2 \rightarrow R_a^Q(\varphi_1 \vee \varphi_2))$

Ces deux propositions montrent que l'opérateur pertinence que nous avons défini caractérise trop d'informations comme étant pertinentes. C'est ce qui est illustré dans l'exemple suivant :

Exemple 2
Reprenons l'exemple du train possiblement en retard à cause d'un incident. a est un agent qui a besoin de savoir si son train est en retard. Supposons que $incident$ soit une information pertinente pour lui et supposons également que l'information "il pleut" modélisée par $pluie$ soit une information vraie. Alors l'information $incident \wedge pluie$ est une information pertinente pour a. En effet, elle contient les éléments nécessaires pour que l'agent réponde à son besoin en information. Cependant, on voudrait dire que l'information $incident$ est plus pertinente que l'information $incident \wedge pluie$ car cette dernière contient un élément $(pluie)$ non-nécessaire pour répondre au besoin en information.

Toutes les informations caractérisées comme pertinentes le sont de façon suffisante dans le sens où elles permettent toutes de répondre au besoin en information. Le problème qui se pose ici est de caractériser, parmi ces informations suffisantes, celles qui ont le plus d'éléments nécessaires pour répondre à ce besoin en information. Les informations qui ne possèdent que des éléments nécessaires sont celles qui sont considérées comme étant le plus pertinentes.

4 Hiérachies dans la pertinence

Dans cette sous-partie, on donne des exemples de hiérarchie caractérisant ce "nécessaire".

4.1 Cas des clauses et des cubes

On note $\mathcal{R}_a^Q$ l'ensemble des formules qui sont pertinentes. On traite tout d'abord le cas des clauses et des cubes.

Clauses.
Considérons le cas où $\mathcal{R}_a^Q$ est un ensemble de clauses.

Définition 2
Soient φ_1 et φ_2 deux clauses. On définit $\varphi_1 \leq_{Cl} \varphi_2$ ssi $\vdash \varphi_2 \rightarrow \varphi_1$

Si l'on considère que les informations pertinents sont des clauses, alors les plus pertinentes sont les maxima du préordre défini ci-dessus. En effet, les maxima de ce préordre

sont les clauses qui ne sont subsumées par aucune autre. Ces informations sont donc les plus précises et peuvent être considérées comme étant les informations nécessaires. On peut alors définir des degrés de pertinence en prenant les maxima successifs du préordre.

Par exemple, supposons que $\mathcal{R}_a^Q = \{incident, incident \vee pluie, incident \vee pluie \vee greve, greve\}$. Alors les informations les plus pertinentes sont $\{incident, greve\}$. En effet, ce sont les plus précises. Ensuite, viennent $\{incident \vee pluie\}$ puis $\{incident \vee pluie \vee greve\}$.

Cubes.

Considérons le cas où $\mathcal{R}_a^Q$ est un ensemble de cubes (conjonction de littéraux).

Définition 3

Soient φ_1 et φ_2 deux cubes. On définit $\varphi_1 \leq_{Cu} \varphi_2$ ssi $\vdash \varphi_1 \rightarrow \varphi_2$

Si l'on considère que les informations pertinentes sont des cubes, alors les plus pertinentes sont les maxima par ce préordre. En effet, les maxima par ce préordre sont les impliquants premiers de Q ou de $\neg Q$[6]Enderton (1972). Les informations les plus pertinentes sont alors bien celles qui contiennent juste les éléments nécessaires pour répondre au besoin en information. De même que pour les clauses, on peut définir des degrés de pertinence en prenant les maxima successifs.

Par exemple, supposons que $\mathcal{R}_a^Q = \{incident, incident \wedge pluie, incident \wedge pluie \wedge greve, greve\}$. Alors les informations les plus pertinentes sont $\{incident, greve\}$. En effet, ce sont celles qui contiennent les éléments nécessaires. Ensuite, viennent $\{incident \wedge pluie\}$ puis $\{incident \wedge pluie \wedge greve\}$.

Formules générales.

En ce qui concerne les formules générales, la hiérarchie est beaucoup plus délicate à établir. En effet, les informations qui sont les plus pertinentes pour le préordre des clauses sont celles qui sont le moins pertinentes pour le préordre des cubes. Néanmoins, on verra dans les paragraphes suivants qu'une telle hiérarchie est possible mais qu'elle est syntaxique.

4.2 La "relevance" de Lakemeyer

Dans ce paragraphe, on compare notre opérateur de pertinence avec une approche de la littérature : la *relevance* [7] de Lakemeyer (1997). Dans cet article, Lakemeyer définit différentes relevances et introduit également d'autres travaux (Marquis (1991), Lin & Reiter (1994)). Cependant, dans un souci de concision, nous ne regarderons de plus près

[6]On rappelle qu'un impliquant est premier s'il cesse d'être un impliquant dès qu'on enlève un de ses littéraux.

[7]Pour éviter toute confusion entre la pertinence de Lakemeyer et celle introduite dans cet article, on gardera la terminologie anglaise "relevance" lorsque l'on traitera de la pertinence de Lakemeyer.

que celle qui paraît la plus proche de notre pertinence. [8]

Tout d'abord, rappelons quelques définitions et notations de Lakemeyer (1997), nécessaires pour sa définition de la relevance.

- une formule ou un ensemble de formules Δ *mentionne* un atome p si p ou $\neg p$ apparaît dans Δ.
- une formule est *objective* si elle ne contient pas d'opérateur modal.
- un *sujet d'intérêt* π est un ensemble d'atomes.
- $\pi_\Delta = \{p \mid p \textit{ est un atome mentionné dans } \Delta\}$

Lakemeyer définit ensuite la notion d'explication minimale.

Définition 4
Soit Δ une ensemble fini de formules.
Si $\not\models \Delta$ alors $lits(\Delta)$ est l'ensemble de littéraux de la forme CNF de Δ.
Si $\models \Delta$ (Δ ne contient que des tautologies) alors $lits(\Delta) = \{\}$

Définition 5
Soient Δ un ensemble fini de formules objectives, α et β deux formules objectives.
β est une explication de α ssi $\models B\Delta \rightarrow B(\beta \rightarrow \alpha)$ et $\not\models B\Delta \rightarrow B(\neg\beta)$.
β est une explication minimale de α ssi β est une explication de α et qu'il n'y a pas d'explication β' de α telle que $lits(\beta') \subseteq lits(\beta)$

Lakemeyer définit alors la relevance de la façon suivante :

Définition 6
Un sujet d'intérêt π est relevant pour α par rapport à Δ (noté $RX_\Delta(\pi, \alpha)$) ssi il existe une explication minimale non triviale [9] de α qui mentionne un $p \in \pi$.

Intuitivement, un sujet d'intérêt est relevant pour une formule s'il contient des atomes nécessaires à l'explication de cette formule.

Exemple 3
Considérons l'ensemble $\Delta = \{incident \wedge greve \rightarrow retard, incident \wedge greve \wedge pluie \rightarrow retard\}$. Ici, l'explication minimale de $retard$ est $incident \wedge greve$. Ainsi, tout sujet d'intérêt contenant un des atomes $incident$ ou $greve$ est considéré comme relevant pour $retard$ par rapport à Δ.

Pour pouvoir étudier le lien entre la relevance et la pertinence, on considère un agent et l'ensemble de ses croyances. Cet ensemble correspond à l'ensemble de formules Δ de Lakemeyer. Les formules que l'on va chercher à expliquer sont la requête et sa négation. Le sujet d'intérêt potentiellement relevant pour ces formules est l'ensemble des atomes de la formule φ pertinente.

[8] Dans Lang & Liberatore & Marquis (2003), les auteurs introduisent une notion de dépendance entre variables et retrouvent plusieurs concepts de la littérature dont la pertinence de Lakemeyer. Nous gardons ici l'approche de Lakemeyer (1997).

[9] Une formule est triviale si elle est équivalente à une tautologie.

Proposition 7
Soient φ et Q deux formules objectives. On note Δ_a la base de croyances de l'agent a qui a le besoin en information Q. On suppose que cette dernière est finie. On a alors le théorème suivant :

$$R_a^Q \varphi \rightarrow RX_{\Delta_a}(\pi_\varphi, Q) \otimes RX_{\Delta_a}(\pi_\varphi, \neg Q)$$

Si une information est pertinente pour répondre à une requête (c'est-à-dire qu'elle permet de déduire soit la requête, soit sa négation) alors c'est qu'elle contient des éléments nécessaires à l'explication de la requête ou de sa négation. Cela revient à dire que l'ensemble des informations suffisantes pour répondre au besoin en information contient l'ensemble des informations nécessaires pour répondre au besoin en information.

4.3 Retour sur les formules générales

L'explication minimale introduite par Lakemeyer induit un préordre sur les formules générales.

En effet, on a la proposition suivante :

Proposition 8
Soient φ_1 et φ_2 deux formules objectives et pertinentes pour un agent a. [10]. On introduit la relation $\leq_{exp}$ définie par $\varphi_1 \leq_{exp} \varphi_2$ ssi φ_2 est une explication de φ_1. La relation $\leq_{exp}$ est un préordre.

Ce préordre est utilisé par Lakemeyer pour caractériser les sujets d'intérêts relevants pour une requête. Cependant, on peut l'utiliser pour hiérarchiser les informations pertinentes (selon notre opérateur).

Proposition 9
Soit $\mathcal{R}_a^Q$ l'ensemble des formules pertinentes. On note $\mathcal{R}m_a Q = max_{\leq_{exp}} \mathcal{R}_a^Q$, c'est-à-dire l'ensemble des formules pertinentes maximales selon le préordre $\leq_{exp}$.

Considérons une formule φ appartenant à $\mathcal{R}m_a Q$. Alors non seulement il existe une explication minimale de Q ou de $\neg Q$ qui mentionne au moins un atome de φ ($RX_{\Delta_a}(\pi_\varphi, Q) \otimes RX_{\Delta_a}(\pi_\varphi, \neg Q)$) mais φ est cette explication minimale de Q ou de $\neg Q$.

Avec ce préordre on caractérise donc les informations pertinentes nécessaires (selon l'explication minimale) et suffisantes pour répondre au besoin en information. On remarque que ce préordre sur les formules générales, s'il est réduit aux clauses ou aux cubes, se ramène aux préodres définis dans la partie précédente. Il généralise donc ce qui a déjà été fait. Le seul problème de ce préordre est qu'il est syntaxique et qu'il dépend donc de la forme des formules.

[10] Le fait que les formules soient pertinentes implique que : 1. les formules considérées ne sont pas des contradictions 2. les formules considérées et leurs négations n'appartiennent pas à la base de croyances Δ_a de l'agent

5 Conclusion

La contribution majeure de ce travail est la définition d'une pertinence orientée agent. Dans le cadre des modèles BDI, nous avons défini un nouvel opérateur représentant le fait qu'une information était pertinente pour un agent par rapport à un besoin en information de celui-ci. Nous avons montré que cet opérateur pouvait être utilisé dans le cas de systèmes multi-agents. Cependant, les formules considérées comme pertinentes sont trop nombreuses. Même si elles sont toutes suffisantes pour répondre au besoin en information de l'agent, elles peuvent contenir des éléments non nécessaires. Dans le cas des clauses et des cubes, on propose des hiérarchies pour sélectionner les informations nécessaires et donc les plus pertinentes. Pour les formules générales, une comparaison avec des travaux existants sur la pertinence nous a permis de sélectionner, syntaxiquement, les formules nécessairement pertinentes.

Ce travail peut être étendu de plusieurs manières.

Tout d'abord, il est possible de replacer l'opérateur pertinence défini ici dans un contexte multi-agents. On s'aperçoit alors qu'un agent, pour savoir si une information est pertinente pour un autre, doit connaître non seulement les besoins en information des autres agents mais aussi leurs croyances. Nombreux sont donc les cas où les agents pensent avoir des informations pertinentes pour d'autres alors qu'elles ne le sont pas.

Le cas où les requêtes sont des requêtes à choix multiples a été étudié. Il permet de définir des degrés de pertinence partielle dans le cas où une information ne répond pas complètement au besoin en information mais élimine une partie des réponses possibles. Il serait intéressant d'étendre cette notion de degré de pertinence, par exemple, dans le cas de deux agents qui en combinant leurs informations obtiendraient une information pertinente pour un troisième.

Un passage en logique du premier ordre permettrait également de traiter des besoins en information plus complexes.

Finalement, il serait intéressant d'étudier la pertinence pour d'autres besoins que le besoin en information. Par exemple, un agent pourrait avoir un *besoin de vérification*, c'est-à-dire qu'il aurait besoin de vérifier que ces croyances sont vraies. Dans ce cas, les informations pertinentes seraient celles qui confirment ou qui contredisent ses croyances. De façon symétrique, on agent pourrait avoir un *besoin de complétion*, c'est-à-dire qu'il voudrait être au courant de toutes les informations vraies (dans un domaine donné par exemple).

Références

ALCHOURRÓN C., GÄRDENFORS P. & MAKINSON D. (1985). On the logic of theory change : partial meet contraction and revision functions. In *Journal of Symbolic Logic*, volume 50, p. 510–530.

BORLUND P. (2003). The Concept of Relevance in IR. *Journal of the American Society for Information Science and Technology*, **54(10)**, 913–925.

CHELLAS B. F. (1980). Modal logic : An introduction. Cambridge, MA : Cambridge University Press.

CHEVALLET J.-P. (2004). Modélisation logique pour la recherche d'information. In *Les systèmes de recherche d'information* p. 105–138 : Hermes.

CRESTANI F. & LALMAS M. (2001). Logic and uncertainty in information retrieval. In *Lecture Notes in Computer Science*, p. 179–206.

ENDERTON H. (1972). In *A Mathematical Introduction to Logic* : Academic Press.

FLORIDI L. (2007). Understanding epistemic relevance. *Erkenntniss*.

HERZIG A. & LONGIN D. (2002). A logic of intention with cooperation principles and with assertive speech acts as communication primitives . In C. CASTELFRANCHI & W. L. JOHNSON, Eds., *Proc. 1st Int. Joint Conf. on Autonomous Agent and Multi-Agent System (AAMAS 2002) , Bologna*, p. 920–927 : ACM Press.

LAKEMEYER G. (1997). Relevance from an epistemic perspective. *Artif. Intell.*, **97**(1-2), 137–167.

LANG J. & LIBERATORE P. & MARQUIS P. (2003) Propositional Independence - Formula-Variable Independence and Forgetting In *Journal of Artificial Intelligence Research*, p. 391–443

LALMAS M. & VAN RIJSBERGEN C. (1993). A model of information retrieval system based on situation theory and dempster-shafer theory of evidence. In *Proceedings of the 1st Workshop on Incompleteness and Uncertainty in Information Systems*, p. 62–67.

LIN F. & REITER R. (1994). Forget it ! In R. GREINER & D. SUBRAMANIAN, Eds., *Working Notes, AAAI Fall Symposium on Relevance*, p. 154–159, Menlo Park, California : American Association for Artificial Intelligence.

MARQUIS P. (1991). Novelty revisited. In *ISMIS '91 : Proceedings of the 6th International Symposium on Methodologies for Intelligent Systems*, p. 550–559, London, UK : Springer-Verlag.

WOOLDRIDGE M. (2000). In *Reasoning about rational agents*, Cambridge, Massachusetts : The MIT Press.

Logique de la mise à jour des croyances objectives

Philippe Balbiani[1,2] et Pablo Seban[1,3]

[1] Université de Toulouse
CNRS, Institut de recherche en informatique de Toulouse
118 route de Narbonne, 31062 Toulouse CEDEX 9 (France)

[2] balbiani@irit.fr

[3] seban@irit.fr

Résumé : Le système modal que nous développons consiste à donner une logique pour la mise à jour des croyances objectives. Son langage est construit de la même manière que celui de la logique classique propositionnelle en ajoutant les opérateurs $\Box_i$ (“il est cru par l’agent i que”), $[\phi, G]$ (“après que les agents du groupe G aient appris que ϕ”) et $[G]$ (“après que les agents du groupe G aient appris quoi que ce soit”). Nous définissons les notions de modèle, de formule valide et de formule satisfaite. L’axiomatisation complète de notre système modal est établie. L’étude de la décidabilité et de la complexité de notre système modal est proposée.

Mots-clés : Logique épistémique, croyances objectives, mise à jour des croyances.

1 Introduction

Deux joueurs, Alex et Béa, sont en face à face au poker *Texas Hold’em*. Chacun connaît ses deux cartes et ne sait pas encore quelles seront les cartes communes posées face visible sur la table. Cette donnée est pourtant fixée à l’avance par le mélange du paquet : en effet nos deux joueurs ne peuvent pas en modifier la distribution. Ils l’apprennent par contre au fur et à mesure de leurs enchères, en mettant ainsi à jour leurs croyances sur ces faits objectifs. Ils peuvent se tromper sur les croyances des autres agents ("Alex croit que Béa croit avoir le meilleur jeu") mais leurs croyances concernant telle ou telle carte posée face visible sur la table sont des croyances vraies. Quel modèle nous permet d’exprimer ce caractère objectif de leur croyance, qui fait que s’ils croient un fait objectif c’est que celui-ci est vrai ?

Rentrons dans le cours du jeu. Initialement, chaque joueur a 2 cartes face cachées. Après une phase d’enchères, le donneur pose trois cartes face visible sur la table (le *flop*). S’ensuivent trois phases d’enchères, entrecoupées par la découverte d’une nouvelle carte (le *turn* et la *rivière*). Même s’il pense avoir un jeu perdant, Alex peut tenter un bluff s’il pense que Béa n’est pas sûre d’avoir le jeu gagnant. Comment représenter alors le fait que Béa est sûre d’avoir le jeu gagnant quoi qu’il arrive ultérieurement ?

Supposons maintenant qu'Alex apprenne le jeu de Béa à son insu. Il a alors un avantage sur son adversaire. Il sait en particulier qui a le jeu gagnant, et il peut également déduire si son adversaire est sûre ou non que son jeu sera gagnant. Quelle opération de mise à jour sur notre modèle permet de différencier la découverte commune d'une carte d'une tricherie (dans laquelle seul un joueur apprend quelque chose) ?

Cet article propose un formalisme répondant à ces exigences. Nous présenterons d'abord la logique des croyances objectives et une opération de mise à jour exprimable dans cette logique, ce qui nous permettra d'exprimer des phrases telles que "Après la distribution du flop, Béa envisage d'avoir un jeu perdant" ou "Alex croit que si l'As de pique sort à la rivière, il aura un jeu gagnant". Dans un second temps, nous présenterons le concept de "mise à jour arbitraire" qui nous permettra d'exprimer par exemple "Béa peut apprendre quelque chose de sorte qu'elle sache que son jeu est gagnant et pourtant quoi qu'apprenne Alex, il envisage que Béa envisage d'avoir un jeu perdant".

2 Croyances objectives

2.1 Syntaxe et sémantique

Nous construisons le langage de la logique des croyances objectives à partir d'un ensemble dénombrable AG d'agents et d'un ensemble dénombrable AT d'atomes. L'ensemble des formules est obtenu à partir de AG et AT en itérant un nombre indéfini de fois les opérations suivantes :

- pour tout $p \in AT$, p est une formule,
- $\bot$ ("falsum") est une formule,
- si ϕ est une formule alors $\neg\phi$ ("non ϕ") est une formule,
- si ϕ est une formule et ψ est une formule alors $(\phi \vee \psi)$ ("ϕ ou ψ") est une formule,
- si ϕ est une formule alors pour tout agent $i \in AG$, $\Box_i\phi$ ("il est cru par l'agent i que ϕ") est une formule.

Nous utilisons les abréviation habituelles pour simplifier les écritures. En particulier, $\Diamond_i\phi$ ("il est admis par l'agent i que ϕ") remplace $\neg\Box_i\neg\phi$. La longueur d'une formule ϕ, notation $\mid \phi \mid$, est le nombre d'occurrences de symboles dans ϕ. La formule ϕ est booléenne lorsque pour tout agent $i \in AG$, ϕ ne contient aucune occurrence de l'opérateur $\Box_i$. Il s'agit maintenant d'interpréter sémantiquement les objets syntaxiques que nous avons introduits. Un modèle est une structure relationnelle de la forme $\mathcal{M} = (W, R, V)$ dans laquelle W est un ensemble non vide de mondes possibles, R est une fonction associant à chaque agent $i \in AG$ une relation binaire R_i sur W et V est une fonction associant à chaque monde possible $x \in W$ un ensemble $V(x)$ d'atomes. Relativement à un modèle $\mathcal{M} = (W, R, V)$, nous définissons la relation de satisfaisabilité entre un monde possible $x \in W$ et une formule ϕ, notation $(\mathcal{M}, x) \models \phi$, de la façon suivante :

- $(\mathcal{M}, x) \models p$ ssi $p \in V(x)$,
- $(\mathcal{M}, x) \not\models \bot$,
- $(\mathcal{M}, x) \models \neg\phi$ ssi non $(\mathcal{M}, x) \models \phi$,
- $(\mathcal{M}, x) \models \phi \vee \psi$ ssi $(\mathcal{M}, x) \models \phi$ ou $(\mathcal{M}, x) \models \psi$,
- $(\mathcal{M}, x) \models \Box_i\phi$ ssi pour tout $y \in W$, xR_iy seulement si $(\mathcal{M}, y) \models \phi$.

Une formule ϕ est valide dans un modèle $\mathcal{M} = (W, R, V)$, notation $\mathcal{M} \models \phi$, lorsque pour tout $x \in W$, $(\mathcal{M}, x) \models \phi$. Une formule ϕ est satisfaite dans un modèle $\mathcal{M} = (W, R, V)$, notation $\mathcal{M}$ *sat* ϕ, lorsqu'il existe $x \in W$ tel que $(\mathcal{M}, x) \models \phi$. Soit $\mathcal{M} = (W, R, V)$, $\mathcal{M}' = (W', R', V')$ des modèles et $x_0 \in W$, $x'_0 \in W'$ des mondes possibles. Nous dirons que $(\mathcal{M}, x_0)$ et $(\mathcal{M}', x'_0)$ sont bisimilaires, notation $(\mathcal{M}, x_0) \leftrightarrows (\mathcal{M}', x'_0)$, lorsqu'il existe une relation binaire Z entre W et W' telle que $x_0 Z x'_0$ et pour chaque $x \in W$ et pour chaque $x' \in W'$, si xZx' alors $V(x) = V'(x')$ et pour tout $i \in AG$,

- s'il existe $y \in W$ tel que xR_iy alors il existe $y' \in W'$ tel que $x'R'_iy'$ et yZy',
- s'il existe $y' \in W'$ tel que $x'R'_iy'$ alors il existe $y \in W$ tel que xR_iy et yZy'.

La proposition suivante nous explique à quel point des modèles bisimilaires se ressemblent.

Proposition 1
Soit $\mathcal{M} = (W, R, V)$, $\mathcal{M}' = (W', R', V')$ des modèles et $x_0 \in W$, $x'_0 \in W'$ des mondes possibles. Si $(\mathcal{M}, x_0) \leftrightarrows (\mathcal{M}', x'_0)$ alors $(\mathcal{M}, x_0) \models \phi$ ssi $(\mathcal{M}', x'_0) \models \phi$ pour chaque formule ϕ.

2.2 Axiomatisation/complétude

L'intuition que nous avons des énoncés "il est cru par l'agent i que ϕ" et "il est admis par l'agent i que ϕ" est telle que nous nous intéressons à la logique LCO des croyances objectives dont les axiomes sont les suivants :

(LCP) les axiomes de la logique classique propositionnelle,

(K) $\Box_i(\phi \rightarrow \psi) \rightarrow (\Box_i\phi \rightarrow \Box_i\psi)$,

(4) $\Box_i\phi \rightarrow \Box_i\Box_i\phi$,

(5) $\Diamond_i\phi \rightarrow \Box_i\Diamond_i\phi$,

(T_{bool}) $\Box_i\phi \rightarrow \phi$ (ϕ formule booléenne).

Les axiomes (4) et (5) expriment le caractère introspectif de la croyance : "lorsque l'agent i croit que ϕ, il croit qu'il croit que ϕ" et "lorsque l'agent i admet que ϕ, il croit qu'il admet que ϕ" pour chaque formule ϕ. Les axiomes (T_{bool}) expriment le caractère objectif de la croyance : "lorsque l'agent i croit que ϕ, ϕ" pour chaque formule booléenne ϕ. Autrement dit, nous considérons que les croyances des agents sur le monde réel sont consistantes avec celui-ci. Les théorèmes de LCO sont toutes les formules déductibles des axiomes en utilisant les règles de déduction suivantes :

(MP) si ϕ est un théorème et $\phi \rightarrow \psi$ est un théorème alors ψ est un théorème,

(GD) si ϕ est un théorème alors $\Box_i\phi$ est un théorème.

A la suite de Hommersom *et al.* (11), considérons la classe $\mathcal{C}_0$ des modèles $\mathcal{M} = (W, R, V)$ dans lesquels

- R_i est transitive pour tout agent $i \in AG$: pour tout $x \in W$, pour tout $y \in W$ et pour tout $z \in W$, si xR_iz et zR_iy alors xR_iy,
- R_i est euclidienne pour tout agent $i \in AG$: pour tout $x \in W$, pour tout $y \in W$ et pour tout $z \in W$, si zR_ix et zR_iy alors xR_iy,

– R_i est o-sérielle pour tout agent $i \in AG$: pour tout $x \in W$, il existe $y \in W$ tel que xR_iy et $V(x) = V(y)$.

La proposition suivante établit l'adéquation et la complétude de la logique LCO pour la classe $\mathcal{C}_0$.

Proposition 2
Soit ϕ une formule. Alors ϕ est un théorème de LCO ssi ϕ est valide dans tout modèle de la classe $\mathcal{C}_0$.

Notons que le résultat de la proposition 2 n'apparaît pas dans (11). Considérons la formule $\Diamond_i\top$. Cette formule est valide dans toute structure relationnelle $\mathcal{M} = (W, R, V)$ où R_i est sérielle pour tout agent $i \in AG$: pour tout $x \in W$, il existe $y \in W$ tel que xR_iy. Par conséquent, la logique LCO est une extension de la logique $KD45_{AG}$ obtenue en remplaçant l'axiome (T_{bool}) par l'axiome $\Diamond_i\top$. L'inclusion de $KD45_{AG}$ dans LCO est stricte si $AT \neq \emptyset$. Pour le montrer, associons à toute partie X de AT, la logique LCO_X obtenue en remplaçant l'axiome (T_{bool}) par l'axiome $\Box_i\phi \rightarrow \phi$ pour chaque formule booléenne ϕ basée sur X. Le lecteur montrera sans peine que la fonction $X \longmapsto LCO_X$ est strictement croissante. Qui plus est, $LCO = LCO_{AT}$ et $KD45_{AG} = LCO_{\emptyset}$.

2.3 Décidabilité/complexité

De la proposition suivante, il ressort que la logique LCO possède la propriété dite du modèle fini.

Proposition 3
Soit ϕ une formule. Si ϕ est satisfaite dans un modèle de la classe $\mathcal{C}_0$ alors ϕ est satisfaite dans un modèle fini de la classe $\mathcal{C}_0$.

Décider si ϕ est valide dans tout modèle de la classe $\mathcal{C}_0$ revient à décider si $\neg\phi$ est satisfaite dans un modèle de la classe $\mathcal{C}_0$. Et d'après la proposition précédente, elle est satisfaite dans un modèle de la classe $\mathcal{C}_0$ ssi elle est satisfaite dans un modèle fini de la classe $\mathcal{C}_0$. Par conséquent, le problème de décision suivant est décidable :

– entrée : une formule ϕ,
– sortie : ϕ est-elle satisfaite dans un modèle de la classe $\mathcal{C}_0$?

Plus précisément,

Proposition 4
Le problème de décision ci-dessus est $PSPACE$-complet.

Tester si une formule est satisfaite dans un monde possible relativement à un modèle fini peut certainement être fait en un temps fini. Par conséquent, le problème de décision suivant est décidable :

– entrée : une formule ϕ et un modèle fini $\mathcal{M} = (W, R, V)$ de la classe $\mathcal{C}_0$,
– sortie : ϕ est-elle satisfaite dans un monde possible $x \in W$ relativement à $\mathcal{M}$?

Plus précisément,

Proposition 5
Le problème de décision ci-dessus est P-complet.

Notons que les résultats des propositions 3, 4 et 5 n'apparaissent pas dans (11).

3 Mise à jour

Nous nous posons maintenant la question de la mise à jour des croyances objectives : que se passe-t-il dans un modèle lorsqu'un groupe d'agents apprend qu'une formule booléenne est satisfaite ? Pour répondre à cette question, nous présentons le modèle proposé par Hommersom *et al.* (11), en proposant quelques propriétés nouvelles. Soit $\mathcal{M} = (W, R, V)$ un modèle, ϕ une formule booléenne et $G \subseteq AG$ un groupe d'agents. La mise à jour de $\mathcal{M}$ par ϕ et G est le modèle $MAJ_{\phi,G}(\mathcal{M}) = (W', R', V')$ défini de la façon suivante :

- $W' = W \times \{0, 1\}$,
- $(x, a)R'_i(y, b)$ ssi une des conditions suivantes est vérifiée :
  - $a = 0$, $b = 0$ et xR_iy,
  - $a = 1$, $b = 1$, xR_iy, $(\mathcal{M}, y) \models \phi$ et $i \in G$,
  - $a = 1$, $b = 0$, xR_iy et $i \notin G$.
- $V'(x, a) = V(x)$.

Dans tout cet article, nous considérons seulement des groupes finis d'agents. Pour justifier l'opération MAJ qui associe les modèles à leurs mise à jour par des formules booléennes et des groupes d'agents, examinons les propositions suivantes.

Proposition 6
Soit $\mathcal{M} = (W, R, V)$ un modèle de la classe $\mathcal{C}_0$, ϕ une formule booléenne et $G \subseteq AG$ un groupe d'agents. Pour chaque $x \in W$, si $(\mathcal{M}, x) \models \phi$ alors le sous-modèle de $MAJ_{\phi,G}(\mathcal{M})$ engendré à partir de $(x, 1)$ est un modèle de la classe $\mathcal{C}_0$.

Par conséquent, la classe $\mathcal{C}_0$ est stable par l'opération MAJ.

Proposition 7
Soit $\mathcal{M} = (W, R, V)$ un modèle de la classe $\mathcal{C}_0$, ϕ une formule booléenne et $G \subseteq AG$ un groupe d'agents. Pour chaque $x \in W$ et pour tout $i \in AG$,

- $(MAJ_{\phi,G}(\mathcal{M}), (x, 1)) \models \Box_i\phi$ *lorsque* $i \in G$,
- $(MAJ_{\phi,G}(\mathcal{M}), (x, 1)) \models \Box_i\phi$ *ssi* $(\mathcal{M}, x) \models \Box_i\phi$ *lorsque* $i \notin G$.

Il s'ensuit que les agents du groupe G croient la formule apparaissant dans la mise à jour, et que les autres ne la croient que s'ils la croyaient déjà avant la mise à jour.

Proposition 8
Soit $\mathcal{M} = (W, R, V)$ un modèle de la classe $\mathcal{C}_0$, $x \in W$ un monde possible et $G \subseteq AG$ un groupe d'agents. Alors $(MAJ_{\top,G}(\mathcal{M}), (x, 1)) \iff (\mathcal{M}, x)$.

Proposition 9
Soit $\mathcal{M} = (W, R, V)$ un modèle de la classe $\mathcal{C}_0$, $x \in W$ un monde possible et ϕ une formule booléenne. Alors $(MAJ_{\phi,\emptyset}(\mathcal{M}), (x, 1)) \iff (\mathcal{M}, x)$.

Autrement dit, mettre à jour par la constante booléenne $\top$ ou pour un groupe d'agents vide ne change rien aux croyances des agents.

Proposition 10
Soit $\mathcal{M} = (W, R, V)$ un modèle de la classe $\mathcal{C}_0$, $x \in W$ un monde possible, ϕ, ψ des formules booléennes et $G \subseteq AG$ un groupe d'agents. Si $(\mathcal{M}, x) \models \phi$ et $(\mathcal{M}, x) \models \psi$ alors $(MAJ_{\psi,G}(MAJ_{\phi,G}(\mathcal{M})), ((x, 1), 1)) \iff (MAJ_{\phi\wedge\psi,G}(\mathcal{M}), (x, 1))$.

Deux mises à jour successives sont donc équivalentes à une seule.

Proposition 11
Soit $\mathcal{M} = (W, R, V)$ un modèle de la classe $\mathcal{C}_0$, $x \in W$ un monde possible, ϕ, ψ des formules booléennes et $G, H \subseteq AG$ des groupes d'agents. Si $(\mathcal{M}, x) \models \phi$ et $(\mathcal{M}, x) \models \psi$ alors $(MAJ_{\psi,H}(MAJ_{\phi,G}(\mathcal{M})), ((x, 1), 1)) \iff (MAJ_{\phi,G}(MAJ_{\psi,H}(\mathcal{M})), ((x, 1), 1))$.

L'ordre dans une suite de mises à jour est donc sans importance.

Proposition 12
Soit $\mathcal{M} = (W, R, V)$, $\mathcal{M}' = (W', R', V')$ des modèles de la classe $\mathcal{C}_0$, $x \in W, x' \in W'$ des mondes possibles, ϕ une formule booléenne et $G \subseteq AG$ un groupe d'agents. Si $(\mathcal{M}, x) \models \phi$, $(\mathcal{M}', x') \models \phi$ et $(\mathcal{M}, x) \iff (\mathcal{M}', x')$ alors $(MAJ_{\phi,G}(\mathcal{M}), (x, 1)) \iff (MAJ_{\phi,G}(\mathcal{M}'), (x', 1))$.

Deux modèles bisimilaires le restent donc après une même mise à jour. Notons que les résultats des propositions 8, 9, 10 et 12 n'apparaissent pas dans (11).

4 Mise à jour des croyances objectives

4.1 Syntaxe et sémantique

La logique des croyances objectives utilise uniquement les opérateurs $\Box_i$. Par conséquent, elle est loin d'être satisfaisante pour modéliser la mise à jour. Que pouvons-nous faire pour introduire dans LCO de nouveaux opérateurs donnant la possibilité d'analyser la dynamique de la mise à jour des croyances objectives ? En adaptant les travaux de Hommersom *et al.* (11), nous développons une logique de la mise à jour des croyances objectives dont le langage contient, en plus des opérateurs $\Box_i$, des opérateurs de la forme $[\phi, G]$ pour chaque formule booléenne ϕ et pour chaque groupe d'agents $G \subseteq AG$. Nous complétons ensuite cette présentation par des résultats nouveaux d'expressivité et de décidébilité/compexité. $[\phi, G]\psi$ ayant "après que les agents du groupe G aient appris que ϕ, ψ" pour signification, ces opérateurs introduisent une idée de mise à jour. Nous généralisons la relation de satisfaisabilité de la façon suivante :

– $(\mathcal{M}, x) \models [\phi, G]\psi$ ssi $(\mathcal{M}, x) \models \phi$ seulement si $(MAJ_{\phi,G}(\mathcal{M}), (x, 1)) \models \psi$.

La proposition suivante nous explique à quel point des modèles bisimilaires se ressemblent.

Proposition 13
Soit $\mathcal{M} = (W, R, V)$, $\mathcal{M}' = (W', R', V')$ des modèles et $x_0 \in W$, $x'_0 \in W'$ des mondes possibles. Si $(\mathcal{M}, x_0) \leftrightarrows (\mathcal{M}', x'_0)$ alors $(\mathcal{M}, x_0) \models \phi$ ssi $(\mathcal{M}', x'_0) \models \phi$ pour chaque formule ϕ.

4.2 Axiomatisation/complétude

Soit $LMAJCO$ la logique de la mise à jour des croyances objectives dont les axiomes sont les suivants :

(LCO) les axiomes de la logique des croyances objectives,

$(R0)$ $[\phi, G](\psi \to \chi) \to ([\phi, G]\psi \to [\phi, G]\chi)$,

$(R1)$ $[\phi, G]p \leftrightarrow (\phi \to p)$,

$(R2)$ $[\phi, G]\bot \leftrightarrow (\phi \to \bot)$,

$(R3)$ $[\phi, G]\neg\psi \leftrightarrow (\phi \to \neg[\phi, G]\psi)$,

$(R4)$ $[\phi, G](\psi \vee \chi) \leftrightarrow [\phi, G]\psi \vee [\phi, G]\chi$,

$(R5)$ $[\phi, G]\Box_i\psi \leftrightarrow (\phi \to \Box_i[\phi, G]\psi)$ lorsque $i \in G$,

$(R6)$ $[\phi, G]\Box_i\psi \leftrightarrow (\phi \to \Box_i\psi)$ lorsque $i \notin G$.

Les axiomes $(R1)$, $(R2)$, $(R3)$, $(R4)$, $(R5)$ et $(R6)$ ont tous une interprétation simple. Nous le verrons plus bas, leur présence implique que le langage de $LMAJCO$ n'est pas plus expressif que le langage de LCO. Les théorèmes de $LMAJCO$ sont toutes les formules déductibles des axiomes en utilisant les règles de déduction (MP) et (GD) ainsi que la règle de déduction suivante :

$(GMAJ)$ si ψ est un théorème alors $[\phi, G]\psi$ est un théorème.

La proposition suivante établit l'adéquation et la complétude de la logique $LMAJCO$ pour la classe $\mathcal{C}_0$.

Proposition 14
Soit ϕ une formule. Alors ϕ est un théorème de $LMAJCO$ ssi ϕ est valide dans tout modèle de la classe $\mathcal{C}_0$.

Soit τ la traduction des formules du langage de $LMAJCO$ en des formules du langage de LCO définie de la façon suivante :

- $\tau([\phi_1, G_1] \dots [\phi_n, G_n], p) = \phi_1 \wedge \dots \wedge \phi_n \to p$,
- $\tau([\phi_1, G_1] \dots [\phi_n, G_n], \bot) = \phi_1 \wedge \dots \wedge \phi_n \to \bot$,
- $\tau([\phi_1, G_1] \dots [\phi_n, G_n], \neg\phi) = \phi_1 \wedge \dots \wedge \phi_n \to \neg\tau([\phi_1, G_1] \dots [\phi_n, G_n], \phi)$,
- $\tau([\phi_1, G_1] \dots [\phi_n, G_n], \phi \vee \psi) = \tau([\phi_1, G_1] \dots [\phi_n, G_n], \phi) \vee \tau([\phi_1, G_1] \dots [\phi_n, G_n], \psi)$,
- $\tau([\phi_1, G_1] \dots [\phi_n, G_n], \Box_i\phi) = \phi_1 \wedge \dots \wedge \phi_n \to \Box_i\tau(\mu, \phi)$ où μ est la séquence obtenue de $[\phi_1, G_1] \dots [\phi_n, G_n]$ en éliminant les $[\phi, G]$ qui ne concernent pas i,
- $\tau([\phi_1, G_1] \dots [\phi_n, G_n], [\phi, G]\psi) = \tau([\phi_1, G_1] \dots [\phi_n, G_n][\phi, G], \psi)$.

Nous laissons au lecteur le soin de montrer que

- $\mid \tau([\phi_1, G_1] \dots [\phi_n, G_n], \psi) \mid \leq (\mid \phi_1 \mid + \dots + \mid \phi_n \mid + \mid \psi \mid) \times \mid \psi \mid$,

– $\tau([\phi_1, G_1] \dots [\phi_n, G_n], \psi) \leftrightarrow [\phi_1, G_1] \dots [\phi_n, G_n]\psi$ est valide dans tout modèle de la classe $\mathcal{C}_0$.

Par conséquent,

– $\mid \tau(\epsilon, \psi) \mid \leq \mid \psi \mid^2$,
– $\tau(\epsilon, \psi) \leftrightarrow \psi$ est valide dans tout modèle de la classe $\mathcal{C}_0$.

Le langage de $LMAJCO$ est-il plus expressif que le langage de LCO ? La traduction ci-dessus nous autorise à dire que non : toute formule de $LMAJCO$ est équivalente à une formule de LCO.

4.3 Décidabilité/complexité

De la proposition suivante, il ressort que la logique $LMAJCO$ possède la propriété dite du modèle fini.

Proposition 15
Soit ϕ une formule. Si ϕ est satisfaite dans un modèle de la classe $\mathcal{C}_0$ alors ϕ est satisfaite dans un modèle fini de la classe $\mathcal{C}_0$.

Par conséquent, le problème de décision suivant est décidable :

– entrée : une formule ϕ,
– sortie : ϕ est-elle satisfaite dans un modèle de la classe $\mathcal{C}_0$?

Plus précisément,

Proposition 16
Le problème de décision ci-dessus est $PSPACE$-complet.

Quant au problème de décision suivant, il est également décidable :

– entrée : une formule ϕ et un modèle fini $\mathcal{M} = (W, R, V)$ de la classe $\mathcal{C}_0$,
– sortie : ϕ est-elle satisfaite dans un monde possible $x \in W$ relativement à $\mathcal{M}$?

Plus précisément,

Proposition 17
Le problème de décision ci-dessus est P-complet.

5 Mise à jour arbitraire des croyances objectives

5.1 Syntaxe et sémantique

La logique de la mise à jour des croyances objectives utilise uniquement les opérateurs $\Box_i$ et $[\phi, G]$. Par conséquent, elle est loin d'être satisfaisante pour modéliser la mise à jour arbitraire. Que pouvons-nous faire pour introduire dans $LMAJCO$ de nouveaux opérateurs donnant la possibilité d'analyser la dynamique de la mise à jour arbitraire des croyances objectives ? En nous inspirant des travaux de Balbiani *et al.* (3), nous développons une logique de la mise à jour arbitraire des croyances objectives dont le langage contient, en plus des opérateurs $\Box_i$ et $[\phi, G]$, des opérateurs de la forme $[G]$ pour chaque groupe d'agents $G \subseteq AG$. $[G]\psi$ ayant "après que les agents du groupe G

aient appris quoi que ce soit, ψ" pour signification, ces opérateurs introduisent une idée de mise à jour arbitraire. Nous généralisons la relation de satisfaisabilité de la façon suivante :

– $(\mathcal{M}, x) \models [G]\psi$ ssi pour toute formule booléenne ϕ, $(\mathcal{M}, x) \models [\phi, G]\psi$.

Proposition 18
Les formules suivantes sont valides dans tout modèle de la classe $\mathcal{C}_0$:

(T) $[G]\psi \rightarrow \psi$,

(4) $[G]\psi \rightarrow [G][G]\psi$,

(CR) $\langle G\rangle[H]\psi \rightarrow [H]\langle G\rangle\psi$.

Remarquons d'abord que les formules de la forme $[G \cup H]\psi \rightarrow [G][H]\psi$ ne sont pas toutes valides dans tout modèle de la classe $\mathcal{C}_0$. Par exemple, la formule $[\{i, j\}](\Box_i p \rightarrow \Box_i\Box_j p) \rightarrow [\{i\}][\{j\}](\Box_i p \rightarrow \Box_i\Box_j p)$ n'est pas valide dans le modèle présenté sur la figure 1 (qui est de la classe $\mathcal{C}_0$). Remarquons par ailleurs que les formules de la forme $[G]\langle H\rangle\psi \rightarrow \langle H\rangle[G]\psi$ ne sont pas toutes valides dans tout modèle de la classe $\mathcal{C}_0$. Par exemple, la formule $[\{i\}]\langle\{j\}\rangle(\Box_i p \oplus \Box_j p) \rightarrow \langle\{j\}\rangle[\{i\}](\Box_i p \oplus \Box_j p)$ — dans laquelle l'opérateur $\oplus$ dénote la disjonction exclusive — n'est pas valide dans ce même modèle.

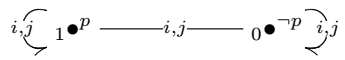


FIG. 1 –

La proposition suivante nous explique à quel point des modèles bisimilaires se ressemblent.

Proposition 19
Soit $\mathcal{M} = (W, R, V)$, $\mathcal{M}' = (W', R', V')$ des modèles et $x_0 \in W$, $x'_0 \in W'$ des mondes possibles. Si $(\mathcal{M}, x_0) \leftrightarrows (\mathcal{M}', x'_0)$ alors $(\mathcal{M}, x_0) \models \phi$ ssi $(\mathcal{M}', x'_0) \models \phi$ pour chaque formule ϕ.

5.2 Axiomatisation/complétude

Soit $LMAJACO$ la logique de la mise à jour arbitraire des croyances objectives dont les axiomes sont les suivants :

$(LMAJCO)$ les axiomes de la logique de la mise à jour des croyances objectives,

$(S0)$ $[G](\psi \rightarrow \chi) \rightarrow ([G]\psi \rightarrow [G]\chi)$,

$(S1)$ $[G]\psi \rightarrow [\phi, G]\psi$.

L'axiome $(S1)$ a une interprétation simple. Toutefois, sa seule présence ne suffit pas à rendre $LMAJACO$ complet pour $\mathcal{C}_0$. Pour assurer la complétude de la logique $LMAJACO$ pour la classe $\mathcal{C}_0$, il nous a été nécessaire d'utiliser les règles de déduction suivantes :

$(GMAJA)$ si ψ est un théorème alors $[G]\psi$ est un théorème,

(X) si pour toute formule booléenne ϕ, $\varphi([\phi, G]\psi)$ est un théorème alors $\varphi([G]\psi)$ est un théorème.

Dans la règle de déduction (X), φ représente une forme admissible, i.e. un élément de l'ensemble obtenu à partir d'un symbole dénoté $\sharp$ en itérant un nombre indéfini de fois les opérations suivantes :

- $\sharp$ est une forme admissible,
- si φ est une forme admissible alors pour toute formule ϕ, $(\varphi \vee \phi)$ est une forme admissible,
- si φ est une forme admissible alors pour tout agent $i \in AG$, $\Box_i\varphi$ est une forme admissible.

Remarquons que $\sharp$ apparaît exactement une fois dans toute forme admissible. Pour toute forme admissible φ et pour toute formule ϕ, $\varphi(\phi)$ dénote la formule obtenue de φ en remplaçant l'unique occurrence de $\sharp$ dans φ par ϕ. La proposition suivante établit l'adéquation et la complétude de la logique $LMAJACO$ pour la classe $\mathcal{C}_0$.

Proposition 20
Soit ϕ une formule. Alors ϕ est un théorème de $LMAJACO$ ssi ϕ est valide dans tout modèle de la classe $\mathcal{C}_0$.

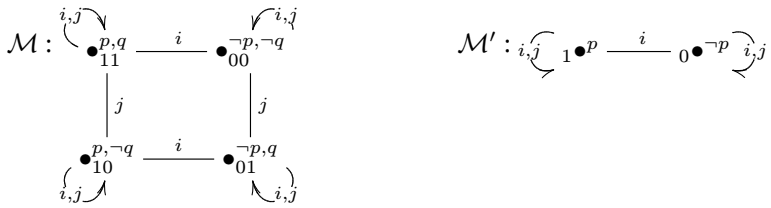


FIG. 2 –

Le langage de $LMAJACO$ est-il plus expressif que le langage de $LMAJCO$? Pour répondre à cette question, il suffit de considérer la formule $\langle\{i,j\}\rangle(\Box_i p \wedge \neg\Box_j\Box_i p)$ et les modèles $\mathcal{M}$ et $\mathcal{M}'$ (de la classe $\mathcal{C}_0$) présentés sur la figure 2. Nous laissons au lecteur le soin de montrer que $(\mathcal{M}, 11) \models \langle\{i,j\}\rangle(\Box_i p \wedge \neg\Box_j\Box_i p)$ et $(\mathcal{M}', 1) \not\models \langle\{i,j\}\rangle(\Box_i p \wedge \neg\Box_j\Box_i p)$. Si la formule $\langle\{i,j\}\rangle(\Box_i p \wedge \neg\Box_j\Box_i p)$ était équivalente à une formule de $LMAJCO$, elle serait satisfaite de la même manière dans les modèles bisimilaires relativement au langage restreint à p. Puisque $(\mathcal{M}, 11) \leftrightarrows (\mathcal{M}', 1)$ relativement au langage restreint à p, alors la formule $\langle\{i,j\}\rangle(\Box_i p \wedge \neg\Box_j\Box_i p)$ n'est équivalente à aucune formule de $LMAJCO$. Par conséquent, le langage de $LMAJACO$ est plus expressif que le langage de $LMAJCO$.

5.3 Décidabilité/complexité

Nous ignorons si la logique $LMAJACO$ possède la propriété dite du modèle fini. Nous ignorons également si le problème de décision suivant est décidable :

- entrée : une formule ϕ,

– sortie : ϕ est-elle satisfaite dans un modèle de la classe $\mathcal{C}_0$?

Par contre, nous savons que le problème de décision suivant l'est :

– entrée : une formule ϕ et un modèle fini $\mathcal{M} = (W, R, V)$ de la classe $\mathcal{C}_0$,

– sortie : ϕ est-elle satisfaite dans un monde possible $x \in W$ relativement à $\mathcal{M}$?

Plus précisément,

Proposition 21
Le problème de décision ci-dessus est $PSPACE$-complet.

6 Conclusion

La logique dynamique épistémique présuppose que le dialogue est un programme cognitif qui met à jour l'état épistémique des agents qui y participent. Son développement récent trouve son origine dans les travaux de Plaza (12) qui définit la première logique des annonces publiques. Dans les travaux de Balbiani *et al.* (3), Baltag et Moss (4) et Gerbrandy (8), des actions plus élaborées que les annonces publiques sont considérées : annonces arbitraires, annonces privées et mise à jour de croyances. D'autres travaux, notamment ceux de Dechesne et Wang (5) et Hommersom *et al.* (11), montrent comment certaines variantes de la logique dynamique épistémique peuvent servir à la modélisation formelle et à la vérification des protocoles cryptographiques. Dans cet article, nous avons enrichi la variante élaborée par Hommersom *et al.* — et qui consiste principalement à ajouter au langage de la logique classique propositionnelle les opérateurs $\Box_i$ ("il est cru par l'agent i que") et $[\phi, G]$ ("après que les agents du groupe G aient appris que ϕ") — de l'opérateur $[G]$ ("après que les agents du groupe G aient appris quoi que ce soit"). De notre système modal, nous avons établi l'axiomatisation complète et proposé l'étude de la décidabilité et de la complexité. Les actions de mise à jour des croyances objectives dont nous venons de parler consistent à faire effectuer par un agent omniscient extérieur à un groupe d'agents des annonces objectives à destination des agents de ce groupe. Un prolongement possible de notre travail concerne, à la manière d'Agotnes et van Ditmarsch (1), l'intégration d'opérateurs de la logique des annonces publiques à notre langage. Que deviennent l'axiomatisation/complétude et la décidabilité/ complexité dans ce contexte ? Un autre prolongement possible de notre travail concerne, à la manière de Dechesne et Wang (5), l'utilisation de notre langage pour la modélisation formelle et la vérification des protocoles cryptographiques. Comment modéliser formellement et vérifier la confidentialité et l'intégrité des communications dans ce contexte ?

Références

[1] Agotnes, T., van Ditmarsch, H. : Coalitions and announcements. In : Autonomous Agents and Multiagent Systems. ACM (2008) 673–680.

[2] Agotnes, T., Balbiani, P., van Ditmarsch, H., Seban, P. : Group announcement logic. En préparation. Voir la version préliminaire sur http ://www.irit.fr/-Publications-

[3] Balbiani, P., Baltag, A., van Ditmarsch, H., Herzig, A., Hoshi, T., de Lima, T. : What can we achieve by arbitrary announcements ? A dynamic take on Fitch's knowability. In : Theoretical Aspects of Rationality and Knowledge. ACM (2007) 42–51.
[4] Baltag, A., Moss, L. : Logics for epistemic programs. Synthese **139** (2004) 165–224.
[5] Dechesne, F., Wang, Y. : Dynamic epistemic verification of security protocols : framework and case study. In : Logic, Rationality and Interaction. College Publications (2007).
[6] Van Ditmarsch, H., van der Hoek, W., Kooi, B. : Dynamic Epistemic Logic. Springer (2007).
[7] French, T., van Ditmarsch, H. : Undecidability for arbitrary public announcement logic. A paraître.
[8] Gerbrandy, J. : The surprise examination in dynamic epistemic logic. Synthese **155** (2007) 21–33.
[9] Grädel, E., Otto, M. : On logics with two variables. Theoretical Computer Science **224** (1999) 73–113.
[10] Halpern, J., Moses, Y. : A guide to completeness and complexity for modal logics of knowledge and belief. Artificial Intelligence **54** (1992) 319–379.
[11] Hommersom, A., Meyer, J.-J., de Vink, E. : Update semantics of security protocols. Synthese **142** (2004) 229–267.
[12] Plaza, J. : Logics of public communications. Synthese **158** (2007) 165–179.

Annexe

Démonstration de la proposition 1 : Par induction sur ϕ.
Démonstration de la proposition 2 : Nous laissons au lecteur le soin de montrer que ϕ est un théorème de LCO seulement si ϕ est valide dans tout modèle de la classe $\mathcal{C}_0$. L'argument habituel basé sur le modèle canonique de LCO permet de montrer que ϕ est un théorème de LCO si ϕ est valide dans tout modèle de la classe $\mathcal{C}_0$.
Démonstration de la proposition 3 : L'argument habituel basé sur la filtration permet de montrer que si ϕ est satisfaite dans un modèle de la classe $\mathcal{C}_0$ alors ϕ est satisfaite dans un modèle fini de la classe $\mathcal{C}_0$.
Démonstration de la proposition 4 : Le problème de décision ci-dessus est $PSPACE$-difficile parce que l'argument développé par Halpern et Moses (10) pour montrer que le problème de la satisfaisabilité des formules de la logique $KD45_{AG}$ est $PSPACE$-difficile s'adapte plutôt facilement. Quant à son appartenance à $PSPACE$, l'argument développé par Halpern et Moses (10) pour montrer que le problème de la satisfaisabilité des formules de la logique $KD45_{AG}$ est dans $PSPACE$ permet de la montrer.
Démonstration de la proposition 5 : Le problème de décision ci-dessus est P-difficile parce que l'argument développé par Grädel and Otto (9) pour montrer que le problème de la satisfaisabilité des formules dans un monde possible relativement à un modèle de la logique K est P-difficile s'adapte plutôt facilement. Quant à son appartenance à P, l'argument développé par Grädel and Otto (9) pour montrer que le problème de la satisfaisabilité des formules dans un monde possible relativement à un modèle de la logique K est dans P permet de la montrer.

Démonstration de la proposition 6 : Simple vérification.
Démonstration de la proposition 7 : Simple vérification.
Démonstration de la proposition 8 : Soit Z la relation binaire entre $W \times \{0,1\}$ et W définie de la façon suivante :

– $(u,a)Zv$ ssi $u = v$.

Nous laissons au lecteur le soin de montrer que Z est une bisimulation entre le sous-modèle de $MAJ_{\top,G}(\mathcal{M})$ engendré à partir de $(x,1)$ et le sous-modèle de $\mathcal{M}$ engendré à partir de x.
Démonstration de la proposition 9 : Soit Z la relation binaire entre $W \times \{0,1\}$ et W définie de la façon suivante :

– $(u,a)Zv$ ssi $u = v$.

Nous laissons au lecteur le soin de montrer que Z est une bisimulation entre le sous-modèle de $MAJ_{\phi,\emptyset}(\mathcal{M})$ engendré à partir de $(x,1)$ et le sous-modèle de $\mathcal{M}$ engendré à partir de x.
Démonstration de la proposition 10 : Soit Z la relation binaire entre $(W \times \{0,1\}) \times \{0,1\}$ et $W \times \{0,1\}$ définie de la façon suivante :

– $((u,a),b)Z(v,c)$ ssi $u = v$, $a = c$ et $b = c$.

Nous laissons au lecteur le soin de montrer que Z est une bisimulation entre le sous-modèle de $MAJ_{\psi,G}(MAJ_{\phi,G}(\mathcal{M}))$ engendré à partir de $((x,1),1)$ et le sous-modèle de $MAJ_{\phi\wedge\psi,G}(\mathcal{M})$ engendré à partir de $(x,1)$.
Démonstration de la proposition 11 : Soit Z la relation binaire sur $(W \times \{0,1\}) \times \{0,1\}$ définie de la façon suivante :

– $((u,a),b)Z((v,c),d)$ ssi $u = v$, $a = d$ et $b = c$.

Nous laissons au lecteur le soin de montrer que Z est une bisimulation entre le sous-modèle de $MAJ_{\psi,H}(MAJ_{\phi,G}(\mathcal{M}))$ engendré à partir de $((x,1),1)$ et le sous-modèle de $MAJ_{\phi,G}(MAJ_{\psi,H}(\mathcal{M}))$ engendré à partir de $((x,1),1)$.
Démonstration de la proposition 12 : Soit Z une bisimulation entre le sous-modèle de $\mathcal{M}$ engendré à partir de x et le sous-modèle de $\mathcal{M}'$ engendré à partir de x'. Soit Z^{maj} la relation binaire entre $W \times \{0,1\}$ et $W' \times \{0,1\}$ définie de la façon suivante :

– $(u,a)Z^{maj}(u',a')$ ssi uZu' et $a = a'$.

Nous laissons au lecteur le soin de montrer que Z^{maj} est une bisimulation entre le sous-modèle de $MAJ_{\phi,G}(\mathcal{M})$ engendré à partir de $(x,1)$ et le sous-modèle de $MAJ_{\phi}(\mathcal{M}')$ engendré à partir de $(x',1)$.
Démonstration de la proposition 13 : Par induction sur ϕ.
Démonstration de la proposition 14 : Nous laissons au lecteur le soin de montrer que ϕ est un théorème de $LMAJCO$ seulement si ϕ est valide dans tout modèle de la classe $\mathcal{C}_0$. L'argument habituel basé sur le modèle canonique de $LMAJCO$ permet de montrer que ϕ est un théorème de $LMAJCO$ si ϕ est valide dans tout modèle de la classe $\mathcal{C}_0$.
Démonstration de la proposition 15 : D'après la proposition 3, la traduction τ des formules du langage de $LMAJCO$ en des formules du langage de LCO définie à la section 4.2 nous autorise à dire que le problème de décision ci-dessus est $PSPACE$-complet.
Démonstration de la proposition 16 : D'après la proposition 4, la traduction τ des formules du langage de $LMAJCO$ en des formules du langage de LCO définie à la

section 4.2 nous autorise à dire que le problème de décision ci-dessus est $PSPACE$-complet.

Démonstration de la proposition 17 : D'après la proposition 5, la traduction τ des formules du langage de $LMAJCO$ en des formules du langage de LCO définie à la section 4.2 nous autorise à dire que le problème de décision ci-dessus est P-complet.

Démonstration de la proposition 18 : Soit $\mathcal{M} = (W, R, V)$ un modèle, $x \in W$ un monde possible et ψ une formule.

(T) : Supposons que $(\mathcal{M}, x) \models [G]\psi$ et $(\mathcal{M}, x) \not\models \psi$. Donc, $(\mathcal{M}, x) \models [\top, G]\psi$. Par conséquent, $(MAJ_{\top,G}(\mathcal{M}), (x, 1)) \models \psi$. D'après la proposition 8, le sous-modèle de $MAJ_{\top,G}(\mathcal{M})$ engendré à partir de $(x, 1)$ et le sous-modèle de $\mathcal{M}$ engendré à partir de x sont bisimilaires : une contradiction.

(4) : Supposons que $(\mathcal{M}, x) \models [G]\psi$ et $(\mathcal{M}, x) \not\models [G][G]\psi$. Donc, il existe une formule booléenne ϕ_1 telle que $(\mathcal{M}, x) \not\models [\phi_1, G][G]\psi$. Par conséquent, $(\mathcal{M}, x) \models \phi_1$ et $(MAJ_{\phi_1,G}(\mathcal{M}), (x, 1)) \not\models [G]\psi$. Donc, il existe une formule booléenne ϕ_2 telle que $(MAJ_{\phi_1,G}(\mathcal{M}), (x, 1)) \not\models [\phi_2, G]\psi$. Par conséquent, $(MAJ_{\phi_1,G}(\mathcal{M}), (x, 1)) \models \phi_2$ et $(MAJ_{\phi_2,G}(MAJ_{\phi_1,G}(\mathcal{M})), ((x, 1), 1)) \not\models \psi$. D'après la proposition 10, le sous-modèle de $MAJ_{\phi_1,G}(MAJ_{\phi_2,G}(\mathcal{M}))$ engendré à partir de $((x, 1), 1)$ et le sous-modèle de $MAJ_{\phi_1\wedge\phi_2,G}(\mathcal{M})$ engendré à partir de $(x, 1)$ sont bisimilaires : une contradiction.

(CR) : Supposons que $(\mathcal{M}, x) \models \langle G\rangle[H]\psi$ et $(\mathcal{M}, x) \not\models [H]\langle G\rangle\psi$. Donc, il existe une formule booléenne ϕ_1 telle que $(\mathcal{M}, x) \models \langle\phi_1, G\rangle[H]\psi$ et il existe une formule booléenne ϕ_2 telle que $(\mathcal{M}, x) \not\models [\phi_2, H]\langle G\rangle\psi$. Par conséquent, $(\mathcal{M}, x) \models \phi_1$, $(MAJ_{\phi_1,G}(\mathcal{M}), (x, 1)) \models [H]\psi$, $(\mathcal{M}, x) \models \phi_2$, $(MAJ_{\phi_2,G}(\mathcal{M}), (x, 1)) \not\models \langle G\rangle\psi$. Donc, $(MAJ_{\phi_1,G}(\mathcal{M}), (x, 1)) \models \phi_2$, $(MAJ_{\phi_2,H}(MAJ_{\phi_1,G}(\mathcal{M})), ((x, 1), 1)) \models \psi$, $(MAJ_{\phi_2,H}(\mathcal{M}), (x, 1)) \models \phi_1$ et $(MAJ_{\phi_1,G}(MAJ_{\phi_2,H}(\mathcal{M})), ((x, 1), 1)) \not\models \psi$. D'après la proposition 11, le sous-modèle de $MAJ_{\phi_2,H}(MAJ_{\phi_1,G}(\mathcal{M}))$ engendré à partir de $((x, 1), 1)$ et le sous-modèle de $MAJ_{\phi_1,G}(MAJ_{\phi_2,H}(\mathcal{M}))$ engendré à partir de $((x, 1), 1)$ sont bisimilaires : une contradiction.Soit $\mathcal{M} = (W, R, V)$ un modèle, $x \in W$ un monde possible et ψ une formule.

Démonstration de la proposition 19 : Par induction sur ϕ.

Démonstration de la proposition 20 : Nous laissons au lecteur le soin de montrer que ϕ est un théorème de $LMAJCO$ seulement si ϕ est valide dans tout modèle de la classe $\mathcal{C}_0$. L'argument habituel basé sur le modèle canonique de $LMAJCO$ permet de montrer que ϕ est un théorème de $LMAJCO$ si ϕ est valide dans tout modèle de la classe $\mathcal{C}_0$.

Démonstration de la proposition 21 : Le problème de décision ci-dessus est $PSPACE$-difficile parce que l'argument développé par Agotnes *et al.* (2) pour montrer que le problème de la satisfaisabilité des formules dans un monde possible relativement à un modèle de la logique GAL est $PSPACE$-difficile s'adapte plutôt facilement. Quant à son appartenance à $PSPACE$, l'argument développé par Agotnes *et al.* (2) pour montrer que le problème de la satisfaisabilité des formules dans un monde possible relativement à un modèle de la logique GAL est dans $PSPACE$ permet de la montrer.

Apprentissage de réseaux de préférences *ceteris paribus*⋆

Frédéric Koriche[1] et Bruno Zanuttini[2]

[1] LIRMM, CNRS UMR 5526, Université Montpellier II
frederic.koriche@lirmm.fr

[2] GREYC, CNRS UMR 6072, Université de Caen Basse-Normandie, ENSICAEN
bruno.zanuttini@info.unicaen.fr

Résumé : Nous étudions l'acquisition de réseaux de préférences *ceteris paribus* (CP-nets), selon le paradigme de l'apprentissage exact avec requêtes défini par Angluin. Dans ce contexte, l'apprenant cherche à découvrir un CP-net cible représentant le modèle cognitif de l'utilisateur en lui posant des questions. Nous montrons que la classe générale des CP-nets booléens n'est pas apprenable avec requêtes d'équivalence et d'appartenance. Puis nous donnons un algorithme montrant que l'importante sous-classe des CP-nets *acycliques complets de taille polynomiale en le nombre de variables* est apprenable avec requêtes d'équivalence et d'appartenance. Ce résultat encourageant indique que les réseaux de « petit degré » utilisés en pratique sont efficacement identifiables par le biais de l'apprentissage interactif.

1 Introduction

La notion de *préférence* joue un rôle central en intelligence artificielle pour la conception d'agents autonomes capables d'assister les humains dans la prise de décision. Par exemple, dans le commerce électronique, l'agent peut aider l'utilisateur à choisir un produit en triant la base de données selon ses préférences. En planification routière, l'agent peut aussi aider l'utilisateur en lui présentant des itinéraires qui optimisent ses préférences en matière de consommation d'énergie, de durée du trajet, de trafic routier, etc. Comme l'ont souligné plusieurs auteurs (Keeney & Raiffa, 1976; French, 1986; Ha & Haddawy, 1998; Kahneman & Tversky, 2000), un des problèmes fondamentaux dans ce type d'applications est d'*acquérir* (ou *éliciter*) les préférences d'un utilisateur dans une représentation permettant la fois de décrire fidèlement le modèle cognitif de l'utilisateur et de raisonner efficacement sur ce modèle.

Parmi les nombreux formalismes de représentation des préférences proposés dans la littérature, les réseaux de préférences *ceteris paribus* ou *CP-nets* ont fait l'objet d'un

⋆Ce travail a été réalisé dans le cadre du projet CANAR, financé par l'ANR (ANR-06-BLAN-0383-02).

intérêt croissant, en proposant une représentation graphique à la fois intuitive et efficace (Boutilier *et al.*, 1999, 2001, 2004; Brafman & Domshlak, 2002; Goldsmith *et al.*, 2005; Wilson, 2004, 2006). Intuitivement, un CP-net peut être vu comme un réseau de croyances où les tables de probabilités conditionnelles associées à chaque variable sont simplement remplacées par des tables de préférences conditionnelles de la forme « toutes choses étant égales par ailleurs, je préfère le littéral p à sa négation $\overline{p}$ dans les états du monde vérifiant le motif t ». D'un point de vue algorithmique, certaines classes de CP-nets, telles que celle des CP-nets booléens arborescents, peuvent supporter en temps polynomial divers processus de raisonnement comme le test de dominance qui consiste à savoir, parmi deux états du monde x et x', si x' est préféré ou non à x.

Dans cet article, nous étudions l'acquisition de CP-nets booléens selon le paradigme de l'apprentissage exact avec requêtes introduit par Angluin (1988). Ce paradigme a suscité beaucoup d'intérêt dans la littérature, en posant un cadre formel au processus d'apprentissage interactif et en permettant d'acquérir efficacement, par interaction, des classes expressives utilisées en représentation des connaissances telles que les formules existentielles conjonctives (Haussler, 1989), les théories de Horn (Angluin *et al.*, 1992), ou encore certaines logiques de description (Frazier & Pitt, 1996).

Dans le cadre des CP-nets, le modèle cible n'est pas utilisé pour classifier des données mais plutôt pour les comparer entre elles. Un *exemple* est alors une paire (x, x') d'états du monde. L'apprenant cherche à identifier le modèle de l'utilisateur en lui posant des questions. Dans une *requête d'équivalence*, l'apprenant présente un CP-net à l'utilisateur et lui demande s'il correspond à son modèle. Si le CP-net n'est pas correct, l'apprenant reçoit un contre-exemple (x, x') qui lui permet de réviser son hypothèse. Dans une *requête d'appartenance*, l'apprenant présente un exemple (x, x') à l'utilisateur et lui demande si effectivement x est préféré à x'. Le but est d'identifier le modèle cible en utilisant le moins de ressources possibles (requêtes et temps de calcul).

Cet article établit des résultats d'apprenabilité pour deux classes de réseaux de préférences. Nous commençons par montrer que la classe générale des CP-nets booléens n'est pas apprenable avec requêtes d'équivalence et d'appartenance. Pour cette classe, qui autorise un nombre arbitraire de parents par sommet du graphe, la dimension de Vapnik-Chervonenkis (Blumer *et al.*, 1989) s'avère en effet au moins exponentielle en le nombre de variables. Par contraste, nous démontrons par un algorithme que l'importante sous-classe des CP-nets acycliques complets de taille polynomiale en le nombre de variables est efficacement apprenable avec requêtes d'équivalence et requêtes d'appartenance.

Comme la taille d'un CP-net augmente de manière exponentielle avec le degré de son graphe, la plupart des modèles exploités en pratique sont fondés sur des graphes de petit degré (Boutilier *et al.*, 2004). Ainsi, notre résultat positif indique que les réseaux de préférences avec « petit degré » sont effectivement identifiables par le biais de l'apprentissage interactif. Notamment, les réseaux arborescents nécessitent seulement un nombre quadratique de requêtes pour être identifiés.

2 Réseaux de préférences ceteris paribus

Si v est une variable booléenne, les *littéraux* sur v sont le littéral *positif* v et le littéral *négatif* $\overline{v}$. Si p est un littéral, alors $\overline{p}$ désigne le littéral opposé. Par exemple, si p est le littéral $\overline{v}$, alors $\overline{p}$ est le littéral v. Un *terme* t est un ensemble de littéraux, vu comme leur conjonction. L'ensemble des variables apparaissant dans t est noté $var(t)$. Si V est un ensemble de variables, un terme t est dit *maximal* sur V s'il contient exactement un littéral par variable de V, c'est à dire $var(t) = V$.

Étant donné un ensemble de variables V, un *état* x du monde est un terme maximal sur V. L'espace des états est noté X. Si t est un terme sur V, on dit que x *satisfait* t si l'on a $t \subseteq x$. Enfin, si x est un état et t un terme, on note $x[t]$ l'état obtenu à partir de x en remplaçant les littéraux de x définis sur $var(t)$ par les littéraux de t. Formellement, $x[t] = \{p \mid p \in t \text{ ou } p \in x \text{ et } \overline{p} \notin t\}$. Par exemple, si x est l'état $\{v_1, v_2, \overline{v}_3, \overline{v}_4\}$ et t est le terme $v_1 \wedge \overline{v}_2 \wedge v_3$, alors $x[t]$ est l'état $\{v_1, \overline{v}_2, v_3, \overline{v}_4\}$.

Nous présentons maintenant les CP-nets booléens. Soient v une variable booléenne et t un terme avec $v \notin var(t)$. Une *règle de préférence ceteris paribus* (CP-règle) sur v sachant t est une expression de la forme $t : v > \overline{v}$ ou $t : \overline{v} > v$. Intuitivement, $t : v > \overline{v}$ signifie que, toutes choses étant égales par ailleurs, v est préférable à $\overline{v}$ dans les états du monde satisfaisant t.

Soient v une variable booléenne et Pa un ensemble fini de variables booléennes avec $v \notin Pa$. Une *table de préférences ceteris paribus* (CP-table) pour v selon Pa, notée $cpt(v)$, est la donnée d'au plus une CP-règle sur v sachant t, pour chaque terme maximal t sur Pa. La CP-table $cpt(v)$ est dite *complète* si elle associe exactement une CP-règle sur v sachant t à chaque terme maximal t sur Pa.

Définition 1 (CP-net)
Soit $V = \{v_1, \ldots, v_n\}$ un ensemble de variables booléennes. Un réseau de préférences ceteris paribus *(CP-net) sur V est la donnée d'un graphe orienté G sur V et, pour tout $i = 1, \ldots, n$, d'une table $cpt(v_i)$ selon l'ensemble Pa_i des parents de v_i dans G.*

Un CP-net N sur le graphe G est dit *complet* si chacune de ses tables est complète. Il est dit *acyclique* si G est acyclique, et *arborescent* si G est une forêt, c'est-à-dire si chaque variable a au plus un arc entrant/parent dans G, et G ne contient pas de cycle.

Le *degré* de N, noté $deg(N)$, est défini par le degré du graphe de N, c'est-à-dire le nombre maximum de parents d'une variable de N. La *taille* d'un CP-net N, notée $||N||$, est définie comme le nombre de CP-règles distinctes apparaissant dans N. Si l'on note $V = \{v_1, \ldots, v_n\}$ l'ensemble des variables de N, on a donc $deg(N) = \max_{v_i \in V} |Pa_i|$, et $||N|| \leq \sum_{i=1}^{n} 2^{|Pa_i|}$, avec une égalité stricte si N est complet.

Exemple 2 (tenue de soirée)
Le CP-net est décrit dans la partie gauche de la figure 1. Il fait intervenir les trois variables v, p et c, pour le choix de la veste, celui du pantalon et celui de la chemise. De manière inconditionnelle, notre utilisatrice préfère le noir ($\overline{v}$) au blanc (v) pour la couleur de la veste, et de même pour celle du pantalon. Si le pantalon et la veste sont de la même couleur, elle préfère le rose $\overline{c}$ au blanc c pour la couleur de la chemise. En revanche, si le pantalon et la chemise sont de couleurs différentes, elle choisira une

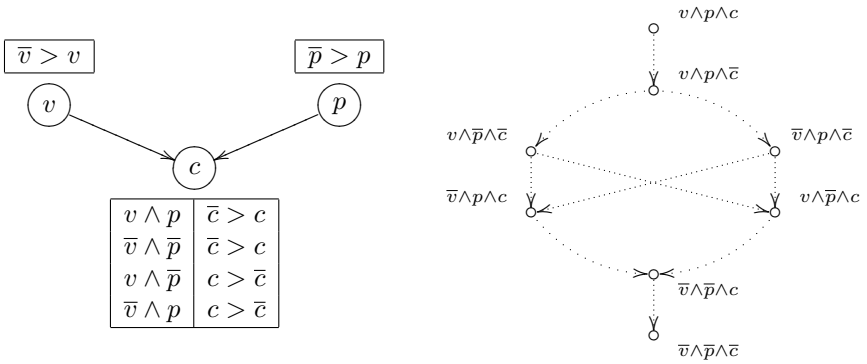


FIG. 1 – Scénario « tenue de soirée » avec à gauche un CP-net, et à droite un préordre partiel sur l'espace d'états qui satisfait le CP-net.

chemise blanche. Notons que le CP-net est acyclique, booléen, complet, et de degré 2*. La partie droite de la figure 1 donne un préordre partiel satisfaisant ce CP-net.*

D'un point de vue sémantique, un CP-net N représente un ensemble de fonctions d'utilité sur les états du système. Rappelons qu'une fonction d'utilité f associe à chaque état x de X un réel $f(x)$ appelé utilité de x. Nous disons que f *satisfait* une CP-règle $t : v > \overline{v}$ si pour tout état x dans X satisfaisant t, nous avons $f(x[v]) > f(x[\overline{v}])$. La satisfaction d'une CP-règle $t : \overline{v} > v$ est définie de manière duale. Nous disons que f *satisfait* une CP-table $cpt(v)$ sur une variable booléenne v si f satisfait chaque règle de $cpt(v)$. Enfin, nous disons que f *satisfait* un CP-net N sur un ensemble de variables booléennes $V = \{v_1, \ldots, v_n\}$ si f satisfait chaque table $cpt(v_i)$ de N.

Définition 3 (dominance)
Soient V un ensemble de variables, N un CP-net sur V, et x, x' deux états sur X. Alors on dit que x domine *x' selon N, et on note $x \succ_N x'$, si $f(x) > f(x')$ pour toute fonction d'utilité f sur X satisfaisant N.*

Nous dirons que deux CP-nets $\hat{N}, N$ sont *équivalents* si pour tout couple d'états (x, x'), x domine x' selon $\hat{N}$ si et seulement s'il le domine selon N.

La proposition suivante caractérise la dominance entre deux états ne diffèrant qu'en un littéral. La preuve est omise par manque de place, mais elle découle de la caractérisation de la dominance par les séquences améliorantes (Boutilier *et al.* (2004)).

Proposition 4
Soit N un CP-net acyclique, x un état, et p un littéral. Alors $x[p] \succ_N x[\overline{p}]$ si et seulement s'il existe dans N une règle de la forme $t : p > \overline{p}$ où $t \subseteq x$.

3 Apprentissage exact de CP-nets

L'apprentissage exact introduit par Angluin (1988) peut être vu comme un jeu de découverte entre l'utilisateur et l'apprenant. Le premier dispose d'un modèle qui est inconnu du second. Il peut néanmoins lui apporter une aide limitée, en répondant à certaines requêtes. Le but du jeu, pour l'apprenant, est de trouver le modèle de l'utilisateur en dépensant le moins de ressources (requêtes et temps de calcul) possible.

Nous appelons *exemple* une paire d'états (x, x') qui ne diffèrent que sur la valeur d'une seule variable. D'un point de vue cognitif, cette restriction est justifiée par le fait que de tels exemples sont relativement simples à trouver pour un utilisateur humain ; ils correspondent à des situations du type « pour cette voiture, je la préfèrerais en rouge plutôt qu'en blanc », où la couleur est une des multiples variables décrivant la voiture. D'un point de vue algorithmique, cette restriction est aussi justifiée par le fait qu'en appliquant la proposition 4, le test de dominance entre x et x' est linéaire pour tout réseau acyclique N, alors qu'il deviendrait NP-difficile si x et x' étaient deux états arbitraires (Boutilier *et al.*, 2004). Enfin, notons que dans tout CP-net complet N, si x et x' diffèrent en une seule variable, on a toujours $x \succ_N x'$ ou $x' \succ_N x$, et que deux CP-nets sont équivalents si et seulement s'ils le sont sur de tels couples d'états.

Une *classe* de CP-nets est un ensemble $\mathcal{N}$ de CP-nets. Nous disons qu'un exemple (x, x') *appartient* à un réseau cible N dans $\mathcal{N}$ si x domine x' dans N. En se basant sur cette notation, nous pouvons donc voir tout CP-net comme un concept. Etant donné un ensemble E d'exemples, soit $\mathcal{N}(E)$ l'ensemble de tous les CP-nets dans $\mathcal{N}$ identifiant une partie de E, c'est à dire $\mathcal{N}(E) = \{E' \subseteq E : \exists N \in \mathcal{N} : E' = E \cap N\}$. La *dimension de Vapnik-Chervonenkis*, ou *VC-dimension*, d'une classe $\mathcal{N}$ de CP-nets, notée $dim(\mathcal{N})$, est le cardinal du plus grand ensemble d'exemples pour lequel nous avons $|\mathcal{N}(E)| = 2^{|E|}$.

Les deux types de requêtes généralement employées en apprentissage exact sont les requêtes d'*appartenance*, utilisées pour connaître la valeur d'un exemple, et les requêtes d'*équivalence*, utilisées pour verifier si l'hypothèse apprise correspond au modèle cible.

Plus précisément, pour une requête d'appartenance $\mathrm{MQ}(x, x')$, l'apprenant fournit un couple d'états (x, x') à l'utilisateur, qui répond oui si (x, x') appartient à N, c'est à dire $x \succ_N x'$, et non sinon. Pour une requête d'équivalence $\mathrm{EQ}(\hat{N}, N)$, l'apprenant fournit un CP-net $\hat{N}$ à l'utilisateur, qui répond oui si N et $\hat{N}$ sont équivalents, et non sinon. Dans ce dernier cas, l'utilisateur fournit de plus un *contre-exemple* (x, x') pour lequel $\hat{N}$ et N impliquent une préférence différente.

Définition 5 (apprentissage exact)
Un algorithme A est dit d'apprentissage avec requêtes d'équivalence et d'appartenance *pour une classe $\mathcal{N}$ de CP-nets, s'il existe un polynôme p tel que pour tout CP-net N de $\mathcal{N}$ sur n variables, après $p(n)$ requêtes d'équivalence et d'appartenance, A converge vers un CP-net $\hat{N}$ de $\mathcal{N}$ qui est équivalent à N. A est dit de plus* efficace *s'il existe un polynôme q tel que la complexité temporelle de A est bornée par $q(n)$. Dans ce cas, $\mathcal{N}$ est dite* apprenable avec requêtes d'équivalence et d'appartenance.

Notons que nous définissons l'efficacité de l'apprentissage en fonction du nombre de variables, et non de la taille du réseau cible.

4 Résultats d'apprenabilité

Après une présentation du formalisme des CP-nets et du modèle d'apprentissage, nous pouvons passer au cœur de l'article en examinant trois résultats d'apprenabilité. Nous commençons par montrer que la classe générale $\mathcal{N}_{\text{BIN}}$ des CP-nets booléens complets n'est pas apprenable avec un nombre de requêtes d'équivalence et d'appartenance polynomial en le nombre de variables. Notons que ce résultat n'est pas surprenant, étant donné que l'on cherche a apprendre un objet de taille exponentielle. Mais le résultat est vrai même sans limite sur le temps de calcul (seulement sur le nombre de requêtes).

Lemme 6
La VC-dimension de $\mathcal{N}_{\text{BIN}}$ est au moins exponentielle en le nombre de variables.

Preuve À chaque ensemble de variables $\{v_1, \ldots, v_n\}$, nous faisons correspondre un ensemble E_n contenant 2^{n-1} exemples tels que pour toute partie de E_n, il existe dans $\mathcal{N}_{\text{BIN}}$ un CP-net de taille n contenant exactement cette partie. Par définition de la VC-dimension, on en déduit donc $dim(\mathcal{N}_{\text{BIN}}) \geq 2^{n-1}$. Soit E_n l'ensemble de tous les exemples de la forme $(\{p_1, \ldots, p_{n-1}, v_n\}, \{p_1, \ldots, p_{n-1}, \overline{v}_n\})$ tels que pour chaque $i \in \{1, \ldots, n-1\}$, p_i est un littéral sur la variable v_i. Donc E_n est l'ensemble des paires d'états du monde qui ne diffèrent qu'en v_n. Cet ensemble est de taille 2^{n-1}, mais toute partie E'_n de E_n peut être uniquement identifiée par un CP-net (acyclique et complet) contenant des préférences inconditionnelles quelconques sur $v_1, \ldots, v_{n-1}$, et pour v_n, la CP-règle $p_1 \wedge \cdots \wedge p_{n-1} : v_n > \overline{v}_n$ si $(\{p_1, \cdots, p_{n-1}, v_n\}, \{p_1, \cdots, p_{n-1}, \overline{v}_n\})$ est dans E'_n, et la règle opposée sinon. □

Théorème 7 (non-apprenabilité des CP-nets de degré quelconque)
La classe $\mathcal{N}_{\text{BIN}}$ n'est pas apprenable avec requêtes d'équivalence et d'appartenance. Ceci s'applique aussi pour la classe $\mathcal{N}_{\text{ACY}}$ des CP-nets booléens complets acycliques.

Preuve (Maass & Turán, 1992, Théorème 6.5) montrent que la VC-dimension d'une classe de concepts constitue une borne inférieure au nombre minimum de requêtes d'équivalence et d'appartenance nécessaires à l'identification exacte des concepts de la classe. Comme la VC-dimension de $\mathcal{N}_{\text{BIN}}$ est au moins exponentielle en le nombre de variables, il est donc impossible d'apprendre cette classe avec requêtes d'équivalence et d'appartenance. Enfin, comme la preuve de la VC-dimension s'applique aussi pour les réseaux acycliques, la classe $\mathcal{N}_{\text{ACY}}$ n'est donc pas non plus apprenable avec requêtes d'équivalence et d'appartenance. □

À présent, nous montrons que la classe $\mathcal{N}^{poly}_{\text{ACY}}$ des CP-nets (booléens) acycliques complets de taille polynomiale en le nombre de variables est identifiable efficacement en utilisant des requêtes d'appartenance et d'équivalence. Dans ce cadre, l'apprenant sait que le réseau est complet et acyclique, mais *ne connaît pas* son degré. Ceci démarque notre résultat de l'approche directe pour l'élicitation de petits réseaux, approche consistant à considérer tous les ensembles de parents possibles ou à utiliser la connaissance du graphe des parents.

L'algorithme, présenté dans la figure 2, démarre en affectant une préférence inconditionnelle sur les valeurs de chaque variable, en se basant sur la préférence sur cette

Algorithme 2 : Apprentissage de CP-nets complets acycliques de taille polynomiale

début
  Initialisation des CP-tables
  $\hat{N} \leftarrow \emptyset$
  $x_0 \leftarrow$ l'état mettant tous les littéraux à faux
  pour chaque *variable* v *dans* V **faire**
    $Pa(v) \leftarrow \emptyset$
    si $\mathrm{MQ}(x_0[v], x_0[\overline{v}])$ **alors** $cpt(v) \leftarrow \{\emptyset : v > \overline{v}\}$
    sinon $cpt(v) \leftarrow \{\emptyset : \overline{v} > v\}$
    $\hat{N} \leftarrow \hat{N} \cup \{cpt(v)\}$
  Raffinement des CP-tables jusqu'à identification
  tant que $(x[p], x[\overline{p}]) \leftarrow \mathrm{EQ}(\hat{N}, N) = non$ **faire**
    On note v *la variable de* p
    Soit $t : \overline{p} > p$ la règle de $\hat{N}$ telle que $t \subseteq x$
    $pa \leftarrow$ EXTRAIREPARENT$(p, t : \overline{p} > p, x)$
    Mise à jour des parents de v *dans* $\hat{N}$
    $\hat{N} \leftarrow \hat{N} \setminus \{cpt(v)\}$
    $Pa(v) \leftarrow Pa(v) \cup \{pa\}$
    Mise à jour de la CP-table de v *dans* $\hat{N}$
    $cpt(v) \leftarrow \emptyset$
    pour chaque *terme maximal* t *sur* $Pa(v)$ **faire**
      si $\mathrm{MQ}(x_0[t \wedge v], x_0[t \wedge \overline{v}])$ **alors** $cpt(v) \leftarrow cpt(v) \cup \{t : v > \overline{v}\}$
      sinon $cpt(v) \leftarrow cpt(v) \cup \{t : \overline{v} > v\}$
    $\hat{N} \leftarrow \hat{N} \cup \{cpt(v)\}$
  retourner $\hat{N}$
fin

variable, *tout littéral étant à faux par ailleurs* (état x_0) ; cet état x_0 servira de référence si la préférence inconditionnelle est invalidée par un contre-exemple ultérieur[1].

Par la suite, lorsqu'il reçoit un contre-exemple de la forme $(x[p], x[\overline{p}])$, l'algorithme l'utilise pour déterminer un nouveau parent de la variable v sur laquelle p est formé, en s'appuyant sur l'état x et la règle qui a été mise en échec par le contre-exemple. Puis il complète la CP-table sur v avec le nouvel ensemble de parents, en utilisant des requêtes d'appartenance (toujours avec l'état de référence x_0).

La procédure EXTRAIREPARENT est specifiée dans la figure 3. L'idée est de tester, avec des requêtes d'appartenance, si chaque nouvelle variable n'apparaissant pas dans la condition de la règle mise en échec est un parent de la variable apparaissant dans la tête de la règle. Le lemme suivant détaille et prouve la correction de cet algorithme.

[1]Tout autre état, y compris un état choisi par l'utilisateur, pourrait ainsi servir de référence.

Algorithme 3 : Découverte d'un nouveau parent d'une variable

Entrées : littéral p formé sur v, règle mise en échec $t : \overline{p} > p$, état x
Sorties : parent pa de v tel que $pa \notin var(t)$
début
  $x_c \leftarrow x$;
  pour chaque *variable* $pa \notin var(t)$ *telle que* $pa \neq v$ *et* $x \models pa$ **faire**
    si $\mathrm{MQ}(x[\overline{pa} \wedge p], x[\overline{pa} \wedge \overline{p}])$ **alors** $x_c \leftarrow x_c[\overline{pa}]$;
    sinon retourner pa
fin

Lemme 8
Soient N un CP-net booléen acyclique complet sur un ensemble de variables V, p un littéral sur $v \in V$, t un terme sur $V \setminus \{v\}$, et x un état tel que $x \models t \wedge p$ et $x[p] \succ_N x[\overline{p}]$. Supposons que l'état x_0 qui met tous les littéraux de V à faux est tel que $x_0[t \wedge \overline{p}] \succ_N x_0[t \wedge p]$. Alors l'algorithme 3 retourne un parent pa de v dans N, avec $pa \notin var(t)$.

Preuve Remarquons tout d'abord que l'algorithme maintient l'invariant $x_c[p] \succ_N x_c[\overline{p}]$. Donc l'algorithme ne peut pas terminer sans retourner une variable, car cela signifierait qu'il a transformé, littéral par littéral, l'état x en l'état $x_0[t]$. Du fait de l'invariant, on aurait donc $x_0[t \wedge p] \succ_N x_0[t \wedge \overline{p}]$, ce qui contredit l'hypothèse.

Soit donc pa la variable retournée. On a $x_c[pa \wedge p] \succ_N x_c[pa \wedge \overline{p}]$ par l'invariant et $x_c[\overline{pa} \wedge \overline{p}] \succ_N x_c[\overline{pa} \wedge p]$ par construction, donc pa est un parent de v. □

Lemme 9
Pour toute variable v, l'algorithme 2 maintient un réseau $\hat{N}$ tel que l'ensemble des parents de v dans $\hat{N}$ est inclus dans celui du réseau cible N. En particulier, puisque N est acyclique, $\hat{N}$ l'est également.

Preuve Cet invariant est démontré par induction sur chaque raffinement de l'hypothèse $\hat{N}$. La propriété est clairement vérifiée après l'initialisation des ensembles de parents à $\emptyset$. Donc, supposons par hypothèse d'induction que la propriété soit vraie pour $\hat{N}$ juste avant de recevoir un nouveau contre-exemple. Puisque la requête d'équivalence a fourni un nouveau contre-exemple $(x[p], x[\overline{p}])$, et que le réseau $\hat{N}$ est complet par construction, cela signifie qu'il implique $x[\overline{p}] \succ_{\hat{N}} x[p]$. D'autre part, comme $\hat{N}$ est acyclique par hypothèse d'induction, il contient bien une règle de la forme $t : \overline{p} > p$ telle que $t \subseteq x$ (proposition 4). Puisque cette règle a été construite sur x_0, on conclut en utilisant le lemme 8 que l'ensemble Pa retourné par la procédure d'extraction est inclus dans les parents de v dans N. □

Théorème 10 (apprenabilité des CP-nets acycliques polynomiaux)
L'algorithme 2 identifie exactement les CP-nets acycliques booléens complets de taille polynomiale en le nombre n de variables. Il utilise pour cela $O(n \log n)$ requêtes d'équivalence et $O(n^2 \log n)$ requêtes d'appartenance, et sa complexité temporelle est en $O(n \log n(c_{\mathrm{EQ}} + nc_{\mathrm{MQ}}))$, où c_{EQ} est la complexité d'une requête d'équivalence et c_{MQ} celle d'une requête d'appartenance.

Preuve Si l'algorithme s'arrête, par définition d'une requête d'équivalence, alors il a bien identifié le réseau cible. Pour montrer qu'il s'arrête, on utilise le lemme 9 et le fait qu'à chaque mise à jour d'un ensemble de parents, cet ensemble croît. L'union des ensembles de parents étant finie, l'algorithme converge donc.

Concernant l'analyse de complexité, rappelons que la taille du réseau cible, booléen et complet, est en $O(2^d)$ où d est le degré du réseau. De plus, puisque sa taille est polynomiale en n il s'ensuit que d est en $O(\log n)$. Le nombre d'appels EQ est en $O(dn)$, puisqu'à chaque fois on découvre un nouveau parent d'une variable. Pour chaque contre-exemple reçu, l'algorithme commence à extraire un nouveau parent avec $O(n)$ appels MQ et utilise ensuite une requête MQ pour chaque ligne de la nouvelle table, donc $O(n+2^d)$ requêtes d'appartenance. Puisque d est en $O(\log n)$, la complexité temporelle est bien en $O(n \log n(c_{\mathrm{EQ}} + nc_{\mathrm{MQ}}))$. □

Corollaire 11 (apprenabilité des CP-nets arborescents)
L'algorithme 2 identifie exactement les CP-nets arborescents (booléens complets) en utilisant seulement $O(n)$ requêtes d'équivalence et $O(n^2)$ requêtes d'appartenance.

5 Discussion

Cette étude se situe à l'intersection de deux domaines de l'intelligence artificielle : le raisonnement sur les préférences et l'apprentissage exact. Nous avons démontré que, d'un côté, la classe générale des CP-nets booléens complets, même acycliques, n'est pas apprenable avec requêtes d'équivalence et d'appartenance, et de l'autre, la sous-classe importante des CP-nets acycliques complets de taille polynomiale est apprenable avec seulement des requêtes d'équivalence et d'appartenance.

Concernant les travaux relatifs à ce cadre d'étude, l'apprentissage, ou élicitation, de préférences a suscité un intérêt récent dans la littérature. Notamment, Domshlak & Joachims (2005) emploient des machines à vecteurs de support pour apprendre des fonctions booléennes à seuil permettant de discriminer des préférences. Dans un contexte plus proche, Blum *et al.* (2004) utilisent l'apprentissage exact avec requêtes pour éliciter des préférences numériques pour les problèmes d'enchères. Dans un cadre symbolique, Sachdev (2007) étudie l'apprentissage de préférences dans un cadre proche du nôtre, mais plus général. Les pistes qu'il évoque pour les CP-nets sont donc des approches tirant moins parti de leur structure. Enfin, Athienitou & Dimopoulos (2007) étudient l'apprentissage de CP-nets minimaux (au sens des ensembles de parents) à partir d'exemples, en utilisant une énumération par niveaux des ensembles de parents.

Nous pensons que cette étude préliminaire sur l'apprenabilité des CP-nets ouvre la voie à plusieurs perspectives intéressantes. Notre algorithme 2 peut être optimisé pour éviter de recalculer les CP-tables ; une borne inférieure sur le nombre de requêtes nous permettrait aussi de confirmer l'optimalité. D'autre part, l'utilisation des requêtes d'équivalence peut être sujette à question dans les situations où l'utilisateur ne peut pas toujours déterminer si son modèle correspond au réseau fourni par l'apprenant. Une alternative, suggérée par Angluin (1988), est de simuler les appels EQ par des requêtes stochastiques. Une autre approche est d'explorer si certaines classes de CP-nets ne né-

cessitent, en fait, que des requêtes d'appartenance. Enfin, le problème de savoir si des réseaux non triviaux sont PAC apprenables, sans utiliser des requêtes, reste à explorer.

Références

ANGLUIN D. (1988). Queries and concept learning. *Machine Learning*, **2**(4), 319–342.

ANGLUIN D., FRAZIER M. & PITT L. (1992). Learning conjunctions of Horn clauses. *Machine Learning*, **9**, 147–164.

ATHIENITOU F. & DIMOPOULOS Y. (2007). Learning CP-Networks : A preliminary investigation. In *3rd Multidisciplinary Workshop on Advances in Preference Handling (M-PREF'07)*.

BLUM A., JACKSON J. C., SANDHOLM T. & ZINKEVICH M. (2004). Preference elicitation and query learning. *Journal of Machine Learning Research*, **5**, 649–667.

BLUMER A., EHRENFEUCHT A., HAUSSLER D. & WARMUTH M. K. (1989). Learnability and the Vapnik-Chervonenkis dimension. *Journal of the ACM*, **36**(4), 929–965.

BOUTILIER C., BACCHUS F. & BRAFMAN R. I. (2001). UCP-networks : A directed graphical representation of conditional utilities. In *17th Conference in Uncertainty in Artificial Intelligence (UAI)*, p. 56–64 : Morgan Kaufmann.

BOUTILIER C., BRAFMAN R. I., DOMSHLAK C., HOOS H. H. & POOLE D. (2004). CP-nets : A tool for representing and reasoning with conditional *ceteris paribus* preference statements. *Journal of Artificial Intelligence Research*, **21**, 135–191.

BOUTILIER C., BRAFMAN R. I., HOOS H. H. & POOLE D. (1999). Reasoning with conditional ceteris paribus preference statements. In *15th Conference on Uncertainty in Artificial Intelligence (UAI)*, p. 71–80 : Morgan Kauffmann.

BRAFMAN R. I. & DOMSHLAK C. (2002). Introducing variable importance tradeoffs into CP-nets. In *18th Conference in Uncertainty in Artificial Intelligence (UAI)*, p. 69–76 : Morgan Kaufmann.

DOMSHLAK C. & JOACHIMS T. (2005). Unstructuring user preferences : Efficient non-parametric utility revelation. In *21st Conference in Uncertainty in Artificial Intelligence (UAI)*, p. 169–177 : AUAI Press.

FRAZIER M. & PITT L. (1996). Classic learning. *Machine Learning*, **25**(2-3), 151–193.

FRENCH S. (1986). *Decision Theory*. Halsted Press.

GOLDSMITH J., LANG J., TRUSZCZYNSKI M. & WILSON N. (2005). The computational complexity of dominance and consistency in CP-nets. In *19th International Joint Conference on Artificial Intelligence (IJCAI)*, p. 144–149 : Professional Book Center.

HA V. A. & HADDAWY P. (1998). Toward case-based preference elicitation : Similarity measures on preference structures. In *14th Conference on Uncertainty in Artificial Intelligence (UAI)*, p. 193–201 : Morgan Kaufmann.

HAUSSLER D. (1989). Learning conjunctive concepts in structural domains. *Machine Learning*, **4**, 7–40.

KAHNEMAN D. & TVERSKY A. (2000). *Choices, Values, and Frames*. Cambridge University Press.

KEENEY R. L. & RAIFFA H. (1976). *Decisions with Multiple Objectives : Preferences and Value Tradeoffs*. Wiley.

MAASS W. & TURÁN G. (1992). Lower bounds methods and spearation results for online-learning models. *Machine Learning*, **9**, 107–145.

SACHDEV M. (2007). On Learning of *Ceteris Paribus* Preference Theories. Master's thesis, Graduate Faculty of North Carolina State University.

WILSON N. (2004). Extending CP-nets with stronger conditional preference statements. In *19th National Conference on Artificial Intelligence (AAAI)*, p. 735–741 : AAAI Press.

WILSON N. (2006). An efficient upper approximation for conditional preference. In *Proc. of the 17th European Conference on Artificial Intelligence (ECAI)*, p. 472–476 : IOS Press.

Pondérations et collisions en logique des pénalités*

Nathalie Chetcuti-Sperandio, Sylvain Lagrue

Université Lille-Nord de France, Artois, F-62307 Lens
CRIL, F-62307 Lens
CNRS UMR 8188, F-62307 Lens
{chetcuti,lagrue}@cril.univ-artois.fr

Résumé : La logique des pénalités est un formalisme de représentation des connaissances permettant de manipuler des buts potentiellement incompatibles. Son principal atout est de permettre une certaine forme de compensation entre les différentes sources d'incompatibilité. Cependant, elle est excessivement sensible au choix des poids : il est possible d'obtenir des conclusions extrêmement différentes à partir d'un même ensemble d'informations en changeant simplement quelques poids, y compris en maintenant l'ordre entre les différentes formules.

Se plaçant dans un cadre restreint de la logique des pénalités, cet article se concentre sur le choix des pondérations et plus particulièrement sur le problème des collisions entre interprétations, ces collisions menant à des conclusions faibles. Nous étudions plus particulièrement une famille de pondérations, les σ-pondérations. Nous montrons comment certains éléments de cette famille permettent d'éviter les collisions tout en maintenant une certaine forme de compensation et nous obtenons leur caractérisation logique en considérant seulement les pondérations et non les formules associées.

Mots-clés : Logique des pénalités, collisions, σ-pondérations, Paralex.

1 Introduction

La logique des pénalités est un formalisme de représentation des connaissances permettant de traiter des buts potentiellement incompatibles. Ce formalisme a été proposé par Pinkas (1991, 1995) et développé dans Dupin de Saint-Cyr *et al.* (1994); Dupin de Saint Cyr (1996). La logique des pénalités propose un cadre intuitif pour manipuler des formules pondérées. Une pénalité est associée à chaque interprétation : cette pénalité est la somme des poids des formules falsifiées par l'interprétation. La principale caractéristique de ce formalisme est sa capacité à faire de la compensation par additivité des poids ; même si l'information la plus préférée est falsifiée par une interprétation cette dernière ne sera pas automatiquement rejetée.

*Ce travail est partiellement financé par le projet ANR *jeunes chercheurs* #JC05-41940 *Planevo*.

Toutefois la logique des pénalités est aussi connue pour être dépendante de la syntaxe des formules. Par ailleurs l'un de ses principaux défauts, néanmoins peu étudié, est sa sensibilité au choix des poids pour l'inférence : il est possible d'obtenir des résultats extrêmement différents à partir d'un même ensemble d'informations en changeant simplement quelques poids. Or, un expert ne peut pas prendre en compte les processus de compensation et de déduction de la logique des pénalités quand il donne ses buts. À moins d'utiliser des poids représentant une mesure additive (telle que l'argent), les poids fournis par l'expert sont généralement artificiels. En effet, il est généralement plus facile pour un expert d'évaluer un coût plutôt qu'un niveau de satisfaction (ou d'insatisfaction). De nombreuses méthodes qualitatives ont été proposées pour traiter les informations avec priorités (sans poids) par exemple dans Benferhat *et al.* (1993); Brewka (1989); Geffner (1992); Lehman (1995). Cependant, aucun de ces formalismes ne permet de compensation comme la logique des pénalités, c'est-à-dire falsifier plusieurs formules de peu d'importance peut être équivalent à falsifier une seule formule d'importance plus grande. De même aucune de ces méthodes ne s'intéresse au problème central que nous traitons ici : l'absence de collisions.

Deux interprétations entrent en collision lorsque la même pénalité leur est associée. Dans ce cas ces interprétations ne peuvent être hiérarchisées et les conclusions tirées peuvent être prudentes à l'excès. Idéalement, si deux interprétations falsifient des formules d'importances différentes elles devraient alors avoir des valeurs différentes de façon à obtenir les conclusions les plus fines possibles. Nous montrons dans cet article comment améliorer les résultats fournis par la logique des pénalités uniquement par le choix d'une pondération judicieuse.

Nous nous plaçons dans cet article dans un cadre restreint, mais néanmoins suffisamment expressif, de la logique des pénalités où chaque poids ne peut intervenir qu'une seule fois. Nous présentons dans ce cadre un panorama de différentes pondérations naturelles pouvant être générées de manière automatique à partir des buts pondérés initiaux fournis par l'expert. Nous montrons que certaines de ces pondérations augmentent le risque de collisions alors que d'autres les évitent mais désactivent dans le même temps le mécanisme de compensation de la logique des pénalités et diminuent donc son intérêt. Nous étudions plus particulièrement une famille de pondérations, les σ-pondérations. Nous établissons que l'une d'entre elles, la pondération parabolique, permet d'éviter les collisions tout en maintenant la compensation et nous obtenons sa caractérisation logique en considérant seulement les pondérations et non les formules associées. Cependant cette pondération départage les interprétations falsifiant le même nombre de formules en fonction de la formule la moins importante qu'elles falsifient. C'est la raison pour laquelle nous proposons enfin une pondération originale, appelée pondération Paralex, qui corrige ce défaut.

Dans la première section nous rappelons succinctement comment gérer des bases de buts ordonnées à l'aide de la logique des pénalités. Puis, nous présentons dans ce cadre le problème des collisions, ainsi qu'une étude complète d'une pondération naturelle, la pondération arithmétique. Dans la section suivante nous proposons un panorama de différentes pondérations possibles en nous focalisant plus particulièrement sur une famille de pondérations, les σ-pondérations. Nous montrons que deux d'entre elles possèdent des propriétés intéressantes en terme de collision et de compensation.

2 Bases de buts ordonnées et logique des pénalités

Dans cet article nous considérons un langage propositionnel *fini* $\mathcal{L}$. L'ensemble des interprétations (ou mondes possibles) basé sur $\mathcal{L}$ est noté Ω tandis que ω représente l'un de ses éléments. La conséquence logique est dénotée par $\models$ et $Mod(\varphi)$ désigne l'ensemble des modèles d'une formule φ, c'est-à-dire $Mod(\varphi) = \{\omega \in \Omega : \omega \models \varphi\}$.

2.1 Rappels

La logique des pénalités s'applique aux bases de buts ordonnées. Une base de buts ordonnée B est un ensemble de formules pondérées de la forme $B = \{\langle \varphi_i, r_i \rangle\}$. Le domaine des poids est l'ensemble des entiers naturels $\mathbb{N}$. Plus un poids est élevé, plus la formule est importante. Le poids $+\infty$ est réservé à l'expression de contraintes d'intégrité, c'est-à-dire de formules ne pouvant être insatisfaites.

À partir d'une base de buts ordonnée, il est possible de définir une pondération sur les interprétations en considérant le poids des formules qu'elles falsifient.

Définition 1 (κ-fonction)
Soit B une base de buts ordonnée. La fonction κ *est une fonction de Ω dans $\mathbb{N} \cup \{+\infty\}$ telle que*

$$\kappa(\omega) = \begin{cases} 0 \text{ si } \omega \models B^* \\ \sum_{\langle \varphi, r \rangle \in False_B(\omega)} r \text{ sinon} \end{cases}$$

où B^* correspond à B privé des poids et $False_B(\omega) = \{\langle \varphi, r \rangle \in B : \omega \not\models \varphi\}$.

À partir de la fonction κ, un préordre total sur l'ensemble des interprétations Ω peut être défini comme suit :

$$\omega \leq_\kappa \omega' \text{ ssi } \kappa(\omega) \leq \kappa(\omega').$$

Contrairement aux formules, les interprétations de plus faible valeur sont les préférées. La conséquence logique peut donc être étendue : une formule est une conséquence d'une base de buts ordonnée si et seulement si ses modèles contiennent tous les modèles κ-préférés. Plus formellement :

Définition 2
Soit B une base de buts ordonnée et φ une formule propositionnelle, alors :

$$B \models_\kappa \varphi \text{ ssi } Min(\Omega, \leq_\kappa) \subseteq Mod(\varphi).$$

La fonction κ est basée sur l'opérateur d'addition, ce qui donne la sémantique de la logique des pénalités cf. Pinkas (1995); Dupin de Saint-Cyr *et al.* (1994).

Exemple 1
Soit B une base de buts ordonnée telle que $B = \{(a \wedge b, 3), (\neg a, 2), (\neg b, 1)\}$. Le tableau 1 détaille la fonction κ. Par exemple, ω_2 falsifie deux formules ($(a \wedge b, 3)$ et $(\neg a, 2)$), donc $\kappa(\omega_2) = 3+2 = 5$. La fonction κ entraîne l'ordre suivant sur l'ensemble des interprétations : $\omega_3 \approx_\kappa \omega_0 <_\kappa \omega_1 <_\kappa \omega_2$. Ainsi $B \models_\kappa a \vee \neg b$, par exemple, car $Min(\Omega, \leq_\kappa) = \{\omega_0, \omega_3\} \subseteq \{\omega_0, \omega_2, \omega_3\} = Mod(a \vee \neg b)$.

$\omega_i \in \Omega$	a	b	$\kappa(\omega_i)$
ω_0	0	0	**3**
ω_1	0	1	4
ω_2	1	0	5
ω_3	1	1	**3**

TAB. 1 – Exemple de fonction κ

La logique des pénalités prend en considération la strate la plus basse et est moins affectée par le problème de la noyade cf. Benferhat *et al.* (1993). Par exemple, dans la logique possibiliste cf. Dubois *et al.* (1994), une interprétation qui falsifie la formule la plus importante est automatiquement exclue alors qu'en logique des pénalités elle peut être conservée.

Le préordre entre interprétations induit par la fonction κ dépend fortement des poids donnés.

2.2 De l'impact des collisions

Comme indiqué dans l'introduction, les conclusions obtenues dans le cadre de la logique des pénalités à partir d'une base de buts ordonnée dépendent fortement du choix des poids. Un expert ne peut cependant pas prendre en compte les processus de compensation et de déduction de la logique des pénalités quand il code ses buts : les poids qu'il donne sont généralement artificiels. Ainsi il est plus simple pour un expert de fournir uniquement un ordre entre ses buts ou d'utiliser des entiers naturels consécutifs $(1, 2, 3, 4, ...)$ pour exprimer le niveau de mécontentement atteint lorsqu'un but n'est pas réalisé.

D'après la définition 2, l'inférence de la logique des pénalités est basée sur les interprétations ayant une valeur pour κ (κ-valeur) minimale. Plus l'ensemble des interprétations κ-minimales est grand, plus les inférences produites sont pauvres et peu instructives : les informations inférées doivent contenir *toutes* les interprétations κ-minimales. Intuitivement l'ensemble des interprétations préférées doit être le plus petit possible pour être le plus précis possible.

Soit B une base de buts ordonnée et ω, ω' deux interprétations, alors ω et ω' entrent en collision si et seulement si elles falsifient un ensemble de formules différentes et $\kappa(\omega) = \kappa(\omega')$.

Si deux interprétations falsifient exactement les mêmes formules, par définition de κ, elles ont la même κ-valeur. À l'inverse si elles falsifient des formules d'importance différente elle devraient, dans l'idéal, avoir une κ-valeur différente afin d'éviter les collisions. Cette notion d'absence de collision peut être caractérisée comme suit :

$(\mathbf{C_{CF}})$ $\forall \omega, \omega' \in \Omega$, $\kappa(\omega) = \kappa(\omega')$ ssi $False_B(\omega) = False_B(\omega')$. [1]

Il est à noter que, par définition, la κ-valeur 0 ne peut jamais être impliquée dans une collision. En effet deux interprétations ne falsifiant aucune formule ont une même

[1] $False_B(\omega)$ est défini dans la définition 1.

κ-valeur de 0, or elles falsifient le même ensemble de formules -l'ensemble *vide*-, elles ne peuvent donc pas entrer en collision.

Exemple 2
Soit B une base de buts ordonnée telle que $B = \{\langle a \vee b, 6\rangle, \langle a \leftrightarrow b, 4\rangle, \langle \neg a, 3\rangle, \langle \neg b, 2\rangle, \langle \neg a \vee \neg b, 1\rangle, \langle a \wedge \neg b, 1\rangle\}$. Le tableau 2 présente la fonction κ_B induite.

	a	b	κ_B
ω_0	0	0	$7 = 6 + 1$
ω_1	0	1	$7 = 4 + 2 + 1$
ω_2	1	0	$7 = 4 + 3$
ω_3	1	1	$7 = 3 + 2 + 1 + 1$

TAB. 2 – Un exemple extrême de collision

Dans le cas présent toutes les interprétations entrent en collision et il n'est donc possible d'inférer de B que $\top$, inférence la plus pauvre qui soit.

Dans l'exemple précédent toutes les interprétations sont considérées comme équi-préférées. Néanmoins deux d'entre elles, ω_0 et ω_2, falsifient moins de formules que les autres. Si l'on ne considère que ces deux interprétations-là, on peut déduire $\neg b$. Cette remarque nous amène à introduire le critère de préservation de la majorité :

$(\mathbf{C_{MP}})$ $|False_B(\omega)| < |False_B(\omega')|$ implique $\omega <_\kappa \omega'$,

où $|E|$ représente la cardinalité de l'ensemble E. Ce critère spécifie qu'une interprétation qui falsifie moins de formules qu'une autre interprétation doit lui être préférée : cela permet une certaine forme de compensation, tout comme la logique des pénalités.

2.3 La pondération arithmétique

Pour s'affranchir de la structure logique des informations et ne considérer que la pondération, nous imposons la restriction suivante dans le reste de l'article : **chaque poids n'apparaît qu'une seule fois dans une base donnée**. Autrement dit, deux formules ne peuvent avoir le même poids. L'expert peut exprimer le fait qu'il considère deux formules comme équi-préférées **uniquement** en les combinant avec une conjonction. L'expert peut se voir proposer un ensemble de n poids dans lequel il choisit les poids pour ses n formules.

À partir d'une pondération W, toutes les κ-valeurs possibles peuvent être calculées. Cet ensemble de valeurs, noté W_Σ, est tel que :

$$W_\Sigma = \left\{ \sum_{a \in A} a : A \subseteq W \right\}.$$

Notons que comme les pondérations sont appliquées aux formules et non aux interprétations, le poids 0 ne présente aucun intérêt. En effet il n'intervient pas dans le calcul des pénalités, et donc n'apparaît pas dans les pondérations considérées.

Étudier les collisions possibles revient à chercher toutes les sommes d'éléments de W (représentant différents ensembles de formules) donnant la même valeur.

Pour choisir une pondération il est assez naturel de prendre la suite des n premiers entiers naturels pour pondérer n formules données :

$$W^{arith,n} = \{1, 2, 3, 4, ..., n\}.$$

Cette pondération est appelée pondération arithmétique car elle dérive d'une progression arithmétique. Le poids n est associé à la formule la plus préférée alors que la moins préférée a un poids de 1. En fait cette pondération naïve est l'une des pires qui soient en matière de collisions.

En fait, le problème des collisions avec la pondération arithmétique est équivalent à un problème bien connu de la théorie des nombres : la partition d'un entier i en termes distincts de 1 à i cf. Hardy & Wright (1979). Ainsi le nombre de collisions pour chaque valeur de $W_{\Sigma}^{arith,n}$ est lié au développement de :

$$(1 + x)(1 + x^2)(1 + x^3)...(1 + x^n).$$

Par exemple, soit $n = 4$. Le développement de $(1 + x)(1 + x^2)(1 + x^3)(1 + x^4)$ est $1 + x + x^2 + 2x^3 + 2x^4 + 2x^5 + 2x^6 + 2x^7 + x^8 + x^9 + x^{10}$. Le monôme $2x^3$ indique qu'il existe deux manières d'obtenir 3.

Il peut être prouvé que, pour tout n, seules six valeurs ne mènent pas à une collision : 0 (cas dans lequel aucune formule n'est falsifiée), 1, 2 et, de façon symétrique, $\sum_{i=1}^{n} i = n(n+1)$, $\sum_{i=1}^{n-2} i$ et $\sum_{i=1}^{n-1} i$. Toutes les autres valeurs peuvent être obtenues par au moins deux sommes différentes de poids, menant ainsi à une collision. Le nombre maximum de collisions se produit à la médiane de $W_{\Sigma}^{arith,n} = \{1, 2, 3, ..., n(n+1)/2\}$, c'est-à-dire $\lfloor n(n + 1)/4 \rfloor$, avec $\lfloor x \rfloor$ la partie entière de x.

Quoique naturelle et intuitive, la pondération arithmétique ne satisfait pas au critère d'absence de collision (C_{CF}).

3 La famille des σ-pondérations

La section précédente montre que la pondération arithmétique est trop naïve et donne des résultats trop pauvres et peu informatifs en matière d'inférence.

Nous rassemblons dans cette section une famille de pondérations aux propriétés communes que nous appelons les *σ-pondérations*. Chaque membre de cette famille est construit à partir d'une σ-suite dont les éléments sont les sommes des éléments précédents. Une σ-suite est elle-même basée sur une suite Φ qui indique le nombre d'éléments à additionner. La définition générique d'une σ-suite est la suivante :

$$\sigma = \begin{cases} \sigma_1 & = & 1 \\ \sigma_i & = & \sum_{j=i-\Phi_{i-1}}^{i-1} \sigma_j \end{cases}$$

où Φ_i est le i^{e} terme de la suite Φ.

Nous illustrons cette famille avec une suite bien connue : les nombres de Fibonacci.

Exemple 3 (Les nombres de Fibonacci)
Considérons la suite

$$\Phi^{Fibo} = (1, 2, 2, 2, 2, 2, 2, ...).$$

Il est possible de calculer la σ-suite basée sur Φ^{Fibo} et notée σ^{Fibo} :

$$\sigma^{Fibo} = (1, 1, 2, 3, 5, 8, 13, 21, ...).$$

Par exemple pour calculer σ_3^{Fibo}, comme $\Phi_2^{Fibo} = 2$, on calcule $\sum_{j=3-2=1}^{2} \sigma_j = \sigma_1^{Fibo} + \sigma_2^{Fibo} = 1 + 1 = 2$.

Cette suite se trouve être celle de Fibonacci. De façon similaire la suite $\Phi^{Tribo} = (1, 2, 3, 3, 3, ...)$ génère la suite appelée souvent nombres de Tribonacci dans laquelle chaque élément est la somme de ses trois prédécesseurs.

On peut remarquer que, dans le cas général, ni les nombres de Fibonacci ni les nombres de Tribonacci n'ont de bonnes propriétés en ce qui concerne les collisions, puisque chaque poids (sauf les tous premiers) est la somme des deux ou trois poids précédents.

3.1 La pondération lexicographique

Nous étudions dans cette section une autre suite basée sur :

$$\Phi^{lex} = (1, 2, 3, 4, 5, 6, ...).$$

La σ-suite générée à partir de Φ^{lex} peut être utilisée telle quelle comme une pondération :

$$\sigma^{lex} = (1, 2, 4, 8, 16, 32, 64, 128...).$$

Dans ce cas nous obtenons les puissances successives de 2 et la logique des pénalités a alors exactement le même comportement que l'inférence lexicographique cf. Dupin de Saint Cyr (1996). Une interprétation ω est préférée à une interprétation ω' ssi elles falsifient les mêmes formules de plus fort poids jusqu'à un poids r, pour lequel ω' falsifie une formule satisfaite par ω. Plus formellement :

Proposition 1
Soit B une base de buts ordonnée telle que $|B| = n$ et telle que $W_B = W^{lex,n}$, alors $\kappa_B(\omega) < \kappa_B(\omega')$ ssi $\exists r$ tel que :

(i) $\langle \varphi, r \rangle \in B$ *et* $\omega \models \varphi$ *mais* $\omega' \not\models \varphi$

(ii) $\forall \langle \varphi, r' \rangle \in B$ *tel que* $r' > r$, $\omega \models \varphi$ *ssi* $\omega' \models \varphi$.

Il s'ensuit de cette proposition que toute base de buts ordonnée B telle que $W_B = W^{lex,n}$ est exempte de collision donc satisfait C_{CF}. En fait si deux interprétations falsifient des ensembles différents de formules, leurs κ-valeurs sont des sommes de puissances de 2 différentes.

Cette pondération évite les collisions mais dans le même temps désactive le mécanisme de compensation (donc l'intérêt) de la logique des pénalités, comme illustré dans l'exemple suivant.

Exemple 4
Soit B une base de buts ordonnée telle que $W_B = \{\sigma_i^{lex} : i \leq 5\}$ et $B = \{\langle \neg a, 16\rangle, \langle a \wedge b, 8\rangle, \langle a, 4\rangle, \langle a \vee b, 2\rangle, \langle a \vee \neg b, 1\rangle\}$. Le tableau 3 présente la fonction κ induite. Les deux

	a	b	κ_B
ω_0	0	0	$14 = 8 + 4 + 2$
ω_1	0	1	$13 = 8 + 4 + 1$
ω_2	1	0	$24 = 16 + 8$
ω_3	1	1	$16 = 16$

TAB. 3 – Pondération lexicographique

interprétations préférées sont les interprétations qui falsifient le plus de formules, à savoir ω_0 et ω_1. On peut observer que ω_3 est rejetée alors qu'elle ne falsifie qu'une seule formule ($\neg a$), qui n'est d'ailleurs même pas une contrainte d'intégrité.

Nous nous intéressons maintenant à deux pondérations qui évitent les collisions tout en permettant une certaine forme de compensation.

3.2 La pondération parabolique

La pondération parabolique a été proposée dans Alvarez Rodriguez (1983) dans le cadre de la théorie du choix social. Cette pondération est basée sur une suite bien connue en informatique, la suite parabolique :

$$\Phi^{parab} = (1, 2, 2, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 5, ...).$$

Dans cette suite, chaque entier n apparaît exactement n fois consécutivement. Chaque élément de cette suite peut être calculé directement à l'aide de la formule $\Phi_i^{parab} = \lfloor (1 + \sqrt{8i - 7})/2 \rfloor$, voir par exemple Knuth (1968). Historiquement, le nom de cette suite provient de cette formule. Les premiers termes de la σ-suite associée sont :

$$\sigma^{parab} = (1, 1, 2, 3, 6, 11, 20, 40, 77, 148, 285, 570, ...).$$

Cette suite croît plus rapidement que la suite de Fibonacci. Cependant, nous n'allons pas utiliser directement cette suite comme pondération, mais les sommes de ses éléments, calculées comme suit :

$$W_i^{parab,n} = \sum_{j=n-i+1}^{n} \sigma_j^{parab}.$$

Le tableau 4 présente la pondération parabolique pour un nombre de niveaux variant de 1 à 6.

Cette pondération est extrêmement performante : non seulement elle satisfait C_{CF}, mais elle autorise également la forme de compensation définie par le critère C_{MP}.

Proposition 2
Soit B une base de buts ordonnée telle que $|B| = n$ et que $W_B = W^{rep,n}$. Dans ce cas B vérifie C_{CF} et C_{MP}.

n	$W^{parab,n}$					
1	1					
2	1	2				
3	2	3	4			
4	3	5	6	7		
5	6	9	11	12	13	
6	11	17	20	22	23	24

TAB. 4 – $W^{parab,n}$ pour $n < 7$

Par ailleurs, il a été démontré dans Alvarez Rodriguez (1983) que le premier poids de $W^{parab,n}$ est le plus petit poids démarrant une pondération qui satisfait C_{CF} et C_{MP}. Cependant, cette pondération vérifie une propriété inattendue lorsque deux interprétations falsifient le même nombre de formules. Celles-ci sont ordonnées en fonction de la formule *la moins prioritaire* qu'elles falsifient, suivant un critère lexicographique inversé :
($\mathbf{C_{RLC}}$) si $|False_B(\omega)| = |False_B(\omega')|$ alors $\omega <_\kappa \omega'$ ssi $min\{r : \langle \varphi, r\rangle \in False_B(\omega) \setminus False_B(\omega')\} < min\{r' : \langle \varphi', r'\rangle \in False_B(\omega') \setminus False_B(\omega)\}$.

Proposition 3
Soit B une base de buts ordonnée telle que $|B| = n$ et $W_B = W^{parab,n}$. Dans ce cas, B satisfait C_{RLC}.

La pondération parabolique est illustrée par l'exemple suivant.

Exemple 5 (suite)
Considérons à nouveau la base de buts ordonnée B de l'exemple 4, avec le même ordre entre les formules, mais avec une pondération différente, basée sur $W^{parab,n}$. Comme $W^{parab,5} = \{6, 9, 11, 12, 13\}$, nous pouvons considérer B telle que $B = \{\langle \neg a, 13\rangle, \langle a \wedge b, 12\rangle, \langle a, 11\rangle, \langle a \vee b, 9\rangle, \langle a \vee \neg b, 6\rangle\}$. Le tableau 5 résume la fonction κ induite.

	a	b	κ_B
ω_0	0	0	$32 = 12 + 11 + 9$
ω_1	0	1	$29 = 12 + 11 + 6$
ω_2	1	0	$25 = 13 + 12$
ω_3	1	1	$\mathbf{13} = 13$

TAB. 5 – Pondération parabolique

3.3 La pondération Paralex

Le critère C_{RLC} qui permet de départager deux interprétations falsifiant le même nombre de formules n'est pas satisfaisant. Il est en effet plus intuitif de départager ces interprétations en fonction de la formule la plus prioritaire falsifiée par l'une et satisfaite par l'autre. C'est pourquoi nous avons proposé dans Chetcuti-Sperandio & Lagrue (2008) une pondération originale qui satisfait le critère de *cardinalité lexicographique* suivant :

($\mathbf{C_{LC}}$) si $|False_B(\omega)| = |False_B(\omega')|$ alors $\omega <_\kappa \omega'$ ssi $max\{r : \langle\varphi, r\rangle \in False_B(\omega) \setminus False_B(\omega')\} < max\{r' : \langle\varphi', r'\rangle \in False_B(\omega') \setminus False_B(\omega)\}$.

Dans cette optique, nous construisons une pondération, $W^{paralex,n}$, basée elle aussi sur σ^{parab} et telle que :

$$\begin{cases} W_1^{paralex,n} = \sum_{j=\lfloor (n+1)/2 \rfloor}^{n} \sigma_j^{parab}, \\ W_i^{paralex,n} = W_{i-1}^{paralex,n} + \sigma_{i-1}^{parab}. \end{cases}$$

La pondération Paralex $W^{paralex,n}$ est liée à la suite parabolique. Si l'on considère uniquement les n premiers éléments de σ^{parab}, $W^{paralex,n}$ commence par la somme des $\lfloor (n+1)/2 \rfloor$ derniers éléments de σ^{parab} puis chacun des poids suivants est la somme du poids courant et de l'élément courant de σ^{parab}. Le tableau 6 présente la pondération Paralex pour $n < 7$.

n	$W^{paralex,n}$					
1	1					
2	1	2				
3	3	4	5			
4	5	6	7	9		
5	11	12	13	15	18	
6	20	21	22	24	27	33

TAB. 6 – $W^{paralex,n}$ pour $n < 7$

Comme souhaité, cette pondération satisfait la propriété *d'absence de collision*, de *préservation de la majorité* et de *cardinalité lexicographique*.

Proposition 4
Soit B une base de buts ordonnée telle que $|B| = n$ et $W_B = W^{paralex,n}$. Dans ce cas, B satisfait C_{CF}, C_{MP} et C_{LC}.

Bien qu'il puisse être montré que la pondération Paralex n'est pas minimale, elle apporte néanmoins une solution efficace et facilement calculable satisfaisant C_{CF}, C_{MP} et C_{LC}. L'exemple suivant illustre la différence entre la pondération Paralex et la pondération parabolique.

φ	$W^{paralex,6}$	$W^{parab,6}$
$a \rightarrow \neg b$	33	24
b	27	23
a	24	22
$a \vee \neg b$	22	20
$\neg a \vee b$	21	17
$a \leftrightarrow \neg b$	20	11

TAB. 7 – Deux pondérations différentes

Exemple 6
Considérons la base de buts ordonnée B décrite par le tableau 7 associée à deux pondérations différentes, $W^{paralex,n}$ et $W^{parab,n}$.

Ces deux pondérations induisent deux fonctions κ différentes, décrites dans le tableau 8. L'interprétation ω_0, qui falsifie une formule de plus que les autres interprétations, est l'interprétation la moins préférée dans les deux cas.

	a	b	$\kappa_B^{paralex}$	κ_B^{parab}
ω_0	0	0	$71 = 27 + 24 + 20$	$56 = 23 + 22 + 11$
ω_1	0	1	$\mathbf{46 = 24 + 22}$	$42 = 22 + 20$
ω_2	1	0	$48 = 27 + 21$	$40 = 23 + 17$
ω_3	1	1	$53 = 33 + 20$	$\mathbf{35 = 24 + 11}$

TAB. 8 – Fonctions κ induites

L'ordre relatif sur les interprétations induit par la pondération parabolique est :

$$\omega_3 <_{\kappa_B^{parab}} \omega_2 <_{\kappa_B^{parab}} \omega_1 <_{\kappa_B^{parab}} \omega_0$$

tandis que l'ordre induit par la pondération Paralex est :

$$\omega_1 <_{\kappa_B^{paralex}} \omega_2 <_{\kappa_B^{paralex}} \omega_3 <_{\kappa_B^{paralex}} \omega_0.$$

L'ordre issu de $\kappa_B^{paralex}$ est le plus convaincant. En effet, $\kappa_B^{paralex}$ préfère ω_1 qui falsifie autant de formules que ω_2 et ω_3, mais qui en falsifie de moins prioritaires. En particulier ω_1 ne falsifie pas la formule la plus prioritaire de la base, $a \rightarrow \neg b$. À l'inverse, κ_B^{parab} associe la plus petite pénalité à ω_3. De cette interprétation, les formules a et b peuvent être déduites. Or ces formules sont incohérentes avec la formule la plus prioritaire de la base : $a \rightarrow \neg b$.

4 Conclusion et perspectives

Les travaux exposés dans cet article apportent des solutions pratiques à un problème fondamental mais néanmoins ignoré jusqu'à présent dans la littérature, le problème des collisions entre interprétations dans la logique des pénalités. Deux principaux facteurs prévalent pour ces collisions, l'expression logique des buts et les pondérations elles-mêmes. Nous avons présenté ici différentes pondérations permettant de respecter le

critère d'absence de collision tout en gardant certaines propriétés de compensation de la logique des pénalités.

Nous avons montré que les pondérations les plus couramment utilisées s'avèrent trop naïves et n'ont pas réellement de bonnes propriétés en matière de collisions, si ce n'est de les favoriser. D'autres, comme la pondération lexicographique, empêchent les collisions, mais désactivent dans le même temps les mécanismes de compensation de la logique des pénalités. Finalement nous avons présenté deux pondérations possédant de bonnes propriétés et facilement calculables, les pondérations parabolique et Paralex.

Cependant, certains questions restent ouvertes, comme l'existence d'une pondération minimale et aisément calculable qui vérifierait C_{CF}, C_{MP} et C_{LC} ou encore comment autoriser plus d'une formule par strate. De plus, d'autres critères, telle l'inclusion ensembliste, peuvent être étudiés. Enfin, ces résultats peuvent être exploités et adaptés au cadre de la fusion de croyances.

Références

ALVAREZ RODRIGUEZ M. A. (1983). *Étude des propriétés d'une suite numérique liée à un problème de vote pondéré*. Thèse de docteur-ingénieur, Université Pierre et Marie Curie - Paris 6.

BENFERHAT S., CAYROL C., DUBOIS D., LANG J. & PRADE H. (1993). Inconsistency Management and Prioritized Syntax-Based Entailment. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI'93)*, p. 640–647.

BREWKA G. (1989). Preferred Subtheories : An Extended Logical Framework for Default Reasoning. In *Proceedings of the 11th International Joint Conference on Artificial Intelligence (IJCAI'89)*, p. 1043–1048.

CHETCUTI-SPERANDIO N. & LAGRUE S. (2008). How to Choose Weightings to Avoid Collisions in a Restricted Penalty Logic. In *Proceedings of the 11th International Conference on Principles of Knowledge Representation and Reasoning (KR'08)*, p. 340–347 : AAAI Press.

DUBOIS D., LANG J. & PRADE H. (1994). Possibilistic Logic. In *Handbook of Logic in Artificial Intelligence and Logic Programming*, volume 3, p. 439–513 : Oxford University Press.

DUPIN DE SAINT CYR F. (1996). *Gestion de l'évolutif et de l'incertain en logiques pondérées*. Thèse de doctorat, Université Paul Sabatier, Toulouse, France.

DUPIN DE SAINT-CYR F., LANG J. & SCHIEX T. (1994). Penalty Logic and its Link with Dempster-Shafer Theory. In *Proceedings of the 10th conference on Uncertainty in Artificial Intelligence (UAI'94)*, p. 204–211.

GEFFNER H. (1992). *Default Reasoning : Causal and Conditional Theories*. MIT Press.

HARDY G. H. & WRIGHT E. M. (1979). *An Introduction to the Theory of Numbers*, chapter Partitions, p. 273–296. Oxford Science Publications, 5th edition.

KNUTH D. E. (1968). *The Art of Computer Programming*, volume 1, p. 44 and 479. Addison-Wesley, second edition.

LEHMAN D. (1995). Another Perspective on Default Reasoning. *Annals of Mathematics and Artificial Intelligence*, **15**(1), 61–82.

PINKAS G. (1991). Propositional Non-Monotonic Reasoning and Inconsistency in Symmetric Neural Networks. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence (IJCAI'91)*, p. 525–530.

PINKAS G. (1995). Reasoning, Nonmonotonicity and Learning in Connectionist Networks that Capture Propositional Knowledge. *Artificial Intelligence*, **77**(2), 203–247.

De la déduction dans le fragment $\{\exists, \wedge, \neg_a\}$ de la logique du premier ordre à SAT

Khalil Ben Mohamed, Michel Leclère, Marie-Laure Mugnier

LIRMM, Université de Montpellier,
161 rue Ada, 34392 Montpellier Cedex 5
{benmohamed,leclere,mugnier}@lirmm.fr

Résumé : nous considérons le problème de déduction dans le fragment existentiel conjonctif muni de la négation atomique en logique du 1^{er} ordre. Nous proposons une réécriture de ce problème en un problème de test de validité d'une formule propositionnelle, tout en exploitant les résultats antérieurs obtenus sur un problème équivalent, celui de l'inclusion de requêtes conjonctives avec négation (Leclère & Mugnier, 2007). Ces résultats sont basés sur la notion d'homomorphisme de graphes. Premièrement, nous faisons une synthèse de ces résultats et les reformulons dans un cadre logique. Deuxièmement, nous présentons notre nouvelle approche qui conduit à tester la validité d'une forme disjonctive propositionnelle, autrement dit l'insatisfiabilité d'une forme clausale propositionnelle, ce qui permet d'utiliser un solveur SAT.

Mots-clés : logique du 1^{er} ordre, déduction, négation atomique, SAT.

1 Introduction

Dans ce papier, nous nous intéressons à la résolution du problème de *déduction dans le fragment existentiel conjonctif* de la logique du 1^{er} ordre (sans fonctions) : « Étant données deux conjonctions de littéraux fermées existentiellement F et G, F se déduit-elle de G (noté $G \models F$) ? » Ce problème est Π_2^P-$complet$[1]. Il peut être reformulé sous la forme de deux problèmes fondamentaux en informatique : le problème d'inclusion de requêtes en base de données (Abiteboul *et al.*, 1995), fondamental pour l'optimisation des mécanismes d'évaluation de requêtes (Chandra & Merlin, 1977) ou la réécriture de requêtes avec vues (Halevy, 2001), et le problème d'implication de clauses qui est à la base des techniques d'ILP (Inductive Logic Programming), domaine à la croisée de l'apprentissage automatique et de la programmation logique (Muggleton, 1992)(Lavrac & Dzeroski, 1994). Dans le cadre du problème d'inclusion de requêtes, nous considérons des requêtes conjonctives avec négation, i.e. de la forme $ans(u) \leftarrow r_1(u_1), \ldots, r_n(u_n), \neg s_1(y_1), \ldots, \neg s_m(y_m)$, où $n \geq 1$ et $m \geq 0$. Étant données deux requêtes conjonctives avec négation q_1 et q_2, il s'agit de déterminer si q_1 est incluse dans q_2, c'est-à-dire si l'ensemble des réponses à q_1 est inclus dans l'ensemble

[1] $\Pi_2^P = (co\text{-}NP)^{NP}$

des réponses à q_2 pour toute base de données (voir (Ullman, 1997)(Wei & Lausen, 2003)(Leclère & Mugnier, 2007)). Le problème d'implication de clauses est le suivant : « Étant données deux clauses C_1 et C_2, C_1 implique-t-elle C_2, i.e. C_2 se déduit-elle de C_1 ? » Si nous considérons des clauses du premier ordre sans fonctions (comme dans (Gottlob, 1987)), nous obtenons par contraposition une instance du problème de déduction dans le fragment existentiel conjonctif de la logique du 1^{er} ordre sans fonctions. Enfin, si l'on ajoute un ordre partiel sur les prédicats, on obtient exactement le problème de déduction dans les graphes conceptuels polarisés (Leclère & Mugnier, 2006).

Dans (Leclère & Mugnier, 2007), le problème d'inclusion de requêtes conjonctives avec négation est étudié par le biais de la notion d'homomorphisme de graphes. Des cas particuliers de requêtes où la complexité du problème décroît sont mis en exergue, et un algorithme de résolution pour le cas général, améliorant ceux existant en base de données, est proposé. Ces résultats seront rapidement présentés dans la section 2, reformulés en termes logiques.

Dans cet article, nous proposons une nouvelle façon de résoudre le problème : nous le reformulons en un problème de test de validité d'une formule propositionnelle, tout en exploitant la notion d'homomorphisme et les résultats antérieurs. La formule obtenue est une forme disjonctive propositionnelle. Sa négation est donc une forme clausale propositionnelle, dont nous pouvons tester l'insatisfiabilité en utilisant un solveur SAT. Cette nouvelle approche sera présentée en section 3.

2 Contexte de l'étude

Nous allons dans un premier temps rappeler les définitions de base et faire une rapide synthèse des résultats de (Leclère & Mugnier, 2007).

2.1 Notions de base

Nous nous plaçons dans le fragment conjonctif existentiel de la logique du 1^{er} ordre muni de la négation atomique, que nous noterons $FOL\{\exists, \wedge, \neg_a\}$, et nous ne considérons pas de symbole de fonctions hormis des constantes.

Définition 1 (Formule et sous-formule)
Une formule *de $FOL\{\exists, \wedge, \neg_a\}$ est une conjonction de littéraux positifs ou négatifs fermée existentiellement. On la verra aussi comme un ensemble $F = \{l_1, \ldots, l_p\}$, où l_i est un littéral pour $i = 1 \ldots p$, composé d'un symbole de prédicat d'arité k (ou de sa négation) et d'un k-tuple de termes. Une* sous-formule *F' de F est alors un ensemble de littéraux tel que $F' \subseteq F$.*

Définition 2 (Formule consistante (ou satisfiable))
Une formule de $FOL\{\exists, \wedge, \neg_a\}$ est consistante *si elle ne contient pas deux littéraux opposés.*

Dans la suite, F et G représenteront deux formules consistantes de $FOL\{\exists, \wedge, \neg_a\}$.

De façon classique, une substitution de F dans G est une application de l'ensemble des variables de F dans l'ensemble des termes (variables ou constantes) de G. Nous

l'étendons à une application des termes de F dans les termes de G, une constante ayant pour image elle-même. On note $s(F)$ l'application d'une substitution s à une formule F. Nous définissons un homomorphisme comme une substitution particulière :

Définition 3 (Homomorphisme)
Un homomorphisme h *de* F *dans* G *est une substitution de* F *dans* G *telle que* $h(F) \subseteq G$.

2.2 Résultats antérieurs

Nous transposons les résultats précédents établis pour le problème d'inclusion de requêtes conjonctives (Leclère & Mugnier, 2007) sous la forme déduction de formules conjonctives existentielles afin de faciliter la comparaison avec la nouvelle approche proposée.

Pour des formules uniquement positives F et G, $G \models F$ si et seulement si il existe un homomorphisme de F dans G. Par contre, dès que l'on considère la négation atomique, un seul sens de la propriété reste vrai :

Propriété 1
Soient deux formules F *et* G *de* $FOL\{\exists, \wedge, \neg_a\}$*, s'il existe un homomorphisme de* F *dans* G *alors* $G \models F$.

En effet, l'homomorphisme ne suffit plus pour répondre au problème de déduction :

Exemple 1
Considérons les formules $F = \exists u \exists v\ (p(u) \wedge r(u,v) \wedge \neg p(v))$ *et* $G = \exists w \exists x \exists y \exists z\ (p(w) \wedge r(w,x) \wedge r(x,y) \wedge r(y,z) \wedge \neg p(z))$. *Il n'y a pas d'homomorphisme de* F *dans* G *alors que* $G \models F$ *: il suffit d'introduire* $(p(x) \vee \neg p(x)) \wedge (p(y) \vee \neg p(y))$ *dans* G *pour s'en persuader. On obtient alors la formule* G' *(équivalente à* G*) suivante :* $G' = \exists w \exists x \exists y \exists z\ ((p(w) \wedge r(w,x) \wedge r(x,y) \wedge r(y,z) \wedge \neg p(z) \wedge p(x) \wedge p(y)) \vee (p(w) \wedge r(w,x) \wedge r(x,y) \wedge r(y,z) \wedge \neg p(z) \wedge \neg p(x) \wedge p(y)) \vee (p(w) \wedge r(w,x) \wedge r(x,y) \wedge r(y,z) \wedge \neg p(z) \wedge p(x) \wedge \neg p(y)) \vee (p(w) \wedge r(w,x) \wedge r(x,y) \wedge r(y,z) \wedge \neg p(z) \wedge \neg p(x) \wedge \neg p(y)))$. *Chacune des quatre conjonctions de* G' *correspond à une façon de compléter* G *par rapport au prédicat* p. *On peut vérifier qu'il existe un homomorphisme de* F *dans chacune de ces quatre conjonctions.* F *se déduit donc de* G'.

Cet exemple servira de fil rouge à notre présentation et sera complété au fur et à mesure. Une façon de résoudre le problème consiste donc à générer toutes les formules conjonctives « complètes » que l'on peut obtenir à partir de G en utilisant les prédicats apparaissant dans G, puis à tester s'il existe un homomorphisme de F dans chacune de ces formules.

Définition 4 (Formule complète)
Une formule consistante G *est dite* complète *par rapport à un ensemble de prédicats* $\mathcal{P}$ *si pour chaque prédicat* p *de* $\mathcal{P}$ *d'arité* k *et chaque k-tuple de termes (non nécessairement distincts)* $t_1, \ldots, t_k$ *de* G, G *contient soit* $p(t_1, \ldots, t_k)$ *soit* $\neg p(t_1, \ldots, t_k)$.

Définition 5 (Complétion)
Une complétion G' *de* G *est une formule obtenue de* G *par ajouts successifs de nouveaux littéraux (composés de prédicats et de termes apparaissant déjà dans* G*) aussi longtemps qu'aucune inconsistance n'apparaît (*$G \subseteq G'$*). Chaque ajout représente une* étape de complétion. *Une complétion de* G *est dite* totale *si c'est une formule complète par rapport à l'ensemble de prédicats considéré.*

Théorème 1
Soient deux formules F *et* G *(avec* G *consistante),* $G \models F$ *si et seulement si quelque soit* G^c*, une formule complète obtenue de* G *par rapport à l'ensemble de prédicats apparaissant dans* G*, il existe un homomorphisme de* F *dans* G^c*.*

Une première propriété de (Leclère & Mugnier, 2007) est que les prédicats n'apparaissant pas à la fois en positif et en négatif dans F et dans G ne sont pas utiles. On peut donc restreindre l'ensemble de prédicats considéré à un ensemble appelé le vocabulaire de complétion :

Définition 6 (Vocabulaire de complétion)
Soient deux formules F *et* G *de* $FOL\{\exists, \wedge, \neg_a\}$*, un* prédicat de complétion *de* F *et* G *est un prédicat apparaissant dans des littéraux de signe opposé, à la fois dans* F *et dans* G*. Le* vocabulaire de complétion *est composé de l'ensemble des prédicats de complétion.*

Une approche brutale (introduite dans (Ullman, 1997)) consiste à calculer l'ensemble des formules complètes de G et à effectuer un test d'homomorphisme de F dans chaque formule complète obtenue. Néanmoins, la complexité d'une telle méthode est exorbitante : $\mathcal{O}(2^{(n_G)^k \times |\mathcal{V}|} \times hom(F, G^c))$, où n_G est le nombre de termes dans G, k est l'arité maximum d'un prédicat de G, $\mathcal{V}$ est le vocabulaire de complétion considéré et $hom(F, G^c)$ est la complexité du test d'existence d'un homomorphisme[2] de F dans G^c.

Deux types d'amélioration de cette méthode sont proposés dans (Leclère & Mugnier, 2007). Premièrement, une exploration incrémentale de l'espace de recherche menant de G à ses complétions totales est effectuée. Cet espace de recherche est partiellement ordonné par la relation d'inclusion. Il est exploré sous la forme d'une arborescence binaire de racine G. Les fils d'un noeud sont obtenus en ajoutant à la formule associée à ce noeud (soit G') un littéral sous sa forme positive et sous sa forme négative (chacune des deux nouvelles formules est donc obtenue par une étape de complétion à partir de G'). Au lieu de construire et tester toutes les complétions totales de G, on recherche un ensemble de complétions partielles couvrant ces complétions totales, i.e. la question de savoir s'il existe un homomorphisme de F dans chaque complétion totale de G devient : « Existe-t-il un ensemble de complétions partielles $\{G_1, \ldots, G_n\}$ tel que (1) il existe un homomorphisme de F dans chaque G_i pour $i = 1 \ldots n$; (2) chaque complétion totale G^c de G est couverte par un G_i, i.e. $G_i \subseteq G^c$? » Après chaque étape de complétion, un test d'homomorphisme de F dans la complétion courante est effectué ; si le résultat est positif, cette complétion partielle est l'un des G_i recherchés, sinon l'exploration continue. La figure 1 donne un aperçu de cette méthode sur le cas très

[2]Ce problème est $NP - Complet$. Il est borné par $n_G^{n_F}$, où n_F est le nombre de termes de F.

simple de l'exemple 1. On crée deux nouvelles formules G_1 et G_2, en ajoutant respec-

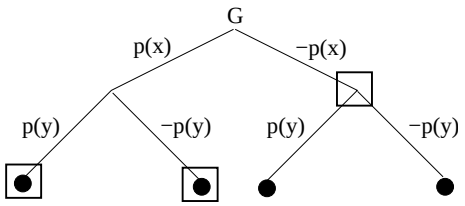


FIG. 1 – Exemple d'arbre de recherche de l'exemple 1. Chaque point noir représente un G^c et chaque carré un G_i.

tivement $p(x)$ et $\neg p(x)$ à G. Il y a un homomorphisme de F dans G_2, on arrête donc de compléter G_2. Il n'y a pas d'homomorphisme de F dans G_1 : on crée deux nouvelles formules G_3 et G_4, en ajoutant respectivement $p(y)$ et $\neg p(y)$ à G_1. Il y a un homomorphisme de F dans G_3 et de F dans G_4. Finalement, l'ensemble prouvant que F se déduit de G est $\{G_2, G_3, G_4\}$ (alors qu'il existe 4 complétions totales de G par rapport à p). Un deuxième niveau d'amélioration consiste à identifier des sous-formules de F pour lesquelles il existe nécessairement un homomorphisme dans G quand $G \models F$. Les deux objectifs principaux liés à de telles sous-formules sont (1) de mettre en place un filtrage (s'il n'y a pas d'homomorphisme d'une de ces sous-formules dans G alors $G \not\models F$) ; (2) de les utiliser comme heuristique de choix du littéral à ajouter lors de la prochaine étape de complétion. Pour améliorer le filtrage et pouvoir les utiliser comme heuristique, on impose que l'homomorphisme de ces sous-formules soit « compatible » avec la formule entière (i.e. que l'homomorphisme de ces sous-formules constitue un homomorphisme de F dans une complétion de G). Cette condition est assurée par la notion d'homomorphisme partiel :

Définition 7 (Homomorphisme partiel)
Soient deux formules F et G de $FOL\{\exists, \wedge, \neg_a\}$ et F' une sous-formule de F, une substitution h de F dans G est un homomorphisme partiel *de F dans G par rapport à F' si :*

- *$h(F') \subseteq G$, i.e. h est un homomorphisme de F' dans G.*
- *pour chaque littéral de prédicat p sur $(c_1, \ldots, c_k)$ de $F - F'$, il n'existe pas un littéral de signe opposé de prédicat p sur $(h(c_1), \ldots, h(c_k))$ dans G.*
- *pour chaque paire de littéraux de signe opposé de prédicat p respectivement sur $(c_1, \ldots, c_k)$ et $(d_1, \ldots, d_k)$ de $F - F'$, $(h(c_1), \ldots, h(c_k)) \neq (h(d_1), \ldots, h(d_k))$.*[3]

Un exemple de sous-formule est celui des *sous-formules pures* : une sous-formule pure de F est une sous-formule ne contenant pas d'occurrences opposées d'un même prédicat (i.e. tout prédicat n'apparaît que sous une forme positive ou négative dans la sous-formule). Une notion plus forte est celle de sous-formule ne possédant pas de

[3]Cette condition n'était pas présente dans (Leclère & Mugnier, 2007) (ce qui ne remet pas en cause les résultats de ce papier), mais elle est nécessaire pour assurer que h constitue un homomorphisme de F dans une complétion de G.

littéraux échangeables, c'est-à-dire de littéraux $p(c_1, \ldots, c_k)$ et $\neg p(d_1, \ldots, d_k)$ tels qu'il existe deux complétions totales de G, G_1 et G_2, et deux homomorphismes h_1 et h_2, respectivement de F dans G_1 et de F dans G_2 avec $(h_1(c_1), \ldots, h_1(c_k)) = (h_2(d_1), \ldots, h_2(d_k))$. On a ainsi :

Théorème 2
(Leclère & Mugnier, 2007) Si $G \models F$ alors pour toute sous-formule F' de F sans littéraux échangeables, il existe un homomorphisme partiel de F dans G par rapport à F'.

L'algorithme finalement proposé dans (Leclère & Mugnier, 2007) prend en compte le filtrage associé à différentes sous-formules, choisit le littéral à ajouter selon un homomorphisme partiel par rapport à une sous-formule spéciale (par exemple, une sous-formule pure maximale pour l'inclusion) et effectue un test d'homomorphisme après chaque étape de complétion.

3 Nouvelle approche

Dans cette partie nous présentons notre nouvelle approche, qui consiste en une réécriture du problème de déduction dans $FOL\{\exists, \wedge, \neg_a\}$ en un problème de validité en logique propositionnelle. Nous allons procéder en plusieurs étapes :

1. Calcul de tous les homomorphismes partiels de F dans G par rapport à une sous-formule spéciale de F.
2. Création d'une formule propositionnelle F_{prop} qui fait la disjonction de toutes les conjonctions de littéraux manquants dans G pour que chaque homomorphisme partiel de l'étape 1 soit total.
3. Test de la validité de F_{prop}, c'est-à-dire de l'insatisfiabilité de sa négation.

Remarquons tout d'abord que l'on peut, et ce sans incidence sur notre problème de déduction, remplacer chaque variable de G par une **nouvelle** constante. Cette modification préserve tous les homomorphismes dans G. Dans la suite, G sera donc supposé contenir uniquement des constantes.

Exemple 2
La formule G de l'exemple 1 devient $G = p(a) \wedge r(a, b) \wedge r(b, c) \wedge r(c, d) \wedge \neg p(d)$.

3.1 Calcul des homomorphismes partiels

Nous pouvons constater qu'une complétion totale G^c code généralement beaucoup plus d'informations que nécessaire pour la recherche d'homomorphismes de F dans G^c. L'idée est d'utiliser la notion d'homomorphisme partiel pour éviter des ajouts superflus en ciblant dès le départ des ensembles de littéraux nécessaires.

Nous observons également qu'un homomorphisme partiel de F dans G par rapport à F' (une sous-formule de F) peut toujours s'étendre à un homomorphisme de F dans une complétion partielle de G. De plus, d'après le théorème 1, si $G \models F$ alors il existe

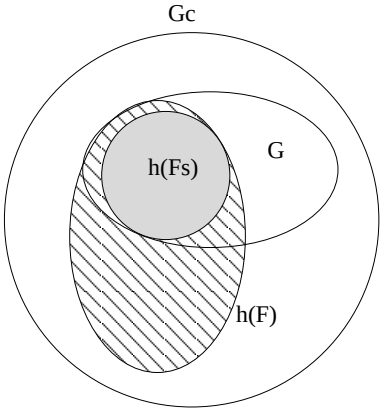


FIG. 2 – Illustration de la propriété 2

un homomorphisme de F dans chaque G^c. Avec le calcul de tous les homomorphismes partiels, on va pouvoir construire un ensemble de complétions partielles $\{G_1, \ldots, G_n\}$ tel qu'il y aura un homomorphisme de F dans G_i pour $i = 1 \ldots n$. Dans le cas où $G \models F$, cet ensemble couvrira l'ensemble des G^c (i.e. pour chaque G^c il existera un G_i tel que $G_i \subseteq G^c$).

Pour obtenir cet ensemble, nous avons besoin d'une sous-formule de F spéciale, que nous appellerons *stable* :

Définition 8 (Sous-formule stable)
La sous-formule stable de F, notée F^s, est la sous-formule composée de tous les littéraux de F dont le prédicat n'est pas un prédicat de complétion de F et G (le vocabulaire de complétion de F^s et G est donc vide). Elle est éventuellement vide.

La propriété ci-dessous est illustrée par la Figure 2.

Propriété 2
Soient deux formules F et G de $FOL\{\exists, \wedge, \neg_a\}$. Soit h un homomorphisme de F dans G^c, une complétion totale de G. h est un homomorphisme partiel de F dans G par rapport à F^s.

Preuve. On a h un homomorphisme de F dans G^c. En restreignant h à F^s, h définit un homomorphisme de F^s dans G. Dès lors que G est consistante, pour chaque littéral l de F qui n'apparaît pas dans F^s, il n'existe pas $h(\bar{l})$ dans G, et G^c est consistante par construction, donc il n'existe pas de paire de littéraux de signe opposé $p(c_1, \ldots, c_k)$ et $\neg p(d_1, \ldots, d_k)$ de $F - F^s$ tels que $(h(c_1), \ldots, h(c_k)) = (h(d_1), \ldots, h(d_k))$. □

Il est à noter que la propriété n'est plus vraie si l'on utilise une sous-formule plus grande que la stable, comme le montre l'exemple suivant :

Exemple 3
Soient F et G les formules de l'exemple 1. $F^s = r(u, v)$. Soient $F^{p_1} = p(u) \wedge r(u, v)$

et $F^{p_2} = r(u,v) \wedge \neg p(v)$ les deux sous-formules pures de F plus grandes que F^s. Soit $G_1^c = p(a) \wedge r(a,b) \wedge p(b) \wedge r(b,c) \wedge \neg p(c) \wedge r(c,d) \wedge \neg p(d)$ une complétion totale de G. Il y a un homomorphisme de F dans G_1^c, $h = \{u \mapsto b, v \mapsto c\}$, mais ce n'est pas un homomorphisme partiel de F dans G que ce soit par rapport à F^{p_1} ou à F^{p_2}.

Ainsi, la sous-formule stable est de taille maximale pour la propriété 2. Cela est dû au fait que les seuls littéraux pouvant servir à la complétion (et ainsi être ajoutés à G) sont ceux possédant un prédicat du vocabulaire de complétion de F et G et la sous-formule stable en est dépourvue.

Exemple 4
Il y a 3 homomorphismes de F dans G par rapport à F^s : $h_1 = \{u \mapsto a, v \mapsto b\}$, $h_2 = \{u \mapsto b, v \mapsto c\}$ et $h_3 = \{u \mapsto c, v \mapsto d\}$

3.2 Création de la formule propositionnelle

Cette étape consiste à se servir des homomorphismes partiels précédemment calculés afin de construire une formule propositionnelle. Comme vu précédemment, un homomorphisme partiel de F dans G peut s'étendre à un homomorphisme complet par l'ajout d'un certain nombre de littéraux à G. Ce sont ces « informations manquantes » que nous codons dans la formule propositionnelle et qui vont nous permettre de répondre au problème de déduction.

Il nous faut donc définir, pour un homomorphisme partiel h de F dans G par rapport à F^s, ce qu'est l'ensemble des littéraux manquants à G afin que h soit égal à un homomorphisme total de F dans une complétion partielle de G. Dans la suite, nous considérons les formules F, G et F^s, et h désigne un homomorphisme partiel de F dans G par rapport à F^s.

Définition 9 (Conjonction minimale)
On appelle conjonction minimale *de G par rapport à h, notée C^m, la conjonction composée de l'atome ■[4] (représentant la tautologie) et des littéraux l tels que $l \in (F - F^s)$ et $h(l) \notin G$.*

Exemple 5
Pour h_1 : $C_1^m = \neg p(b)$; pour h_2 : $C_2^m = p(b) \wedge \neg p(c)$; pour h_3 : $C_3^m = p(c)$.

Définition 10 (Complétion minimale)
On appelle complétion minimale *de G, notée $G^m = G \wedge C^m$, la conjonction composée des littéraux de G et de la conjonction minimale de G par rapport à h.*

Propriété 3
Soit h un homomorphisme partiel de F dans G par rapport à F^s. h est un homomorphisme de F dans la complétion minimale de G issue de h.

[4] nécessaire lorsque h est un homomorphisme total de F dans G car sinon la conjonction est vide.

Propriété 4
Soit C^m une conjonction minimale de G issue d'un homomorphisme partiel de F dans G par rapport à F^s, alors $G \wedge C^m \models F$.

La formule propositionnelle est construite à partir de tous les homomorphismes partiels de F dans G par rapport à F^s :

Définition 11 (Disjonction des Conjonctions Minimales du Stable ($\mathcal{DCMS}$))
On appelle $\mathcal{DCMS}(F, G)$ la disjonction de l'atome $\square$[5] (représentant l'absurde) et des conjonctions minimales de tous les homomorphismes partiels de F dans G par rapport à F^s (i.e. $\square \vee C_1^m \vee \cdots \vee C_n^m$, noté également $\bigvee C^m$).

Exemple 6
Création de la formule propositionnelle : $\mathcal{DCMS}(F, G) = \neg p(b) \vee (p(b) \wedge \neg p(c)) \vee p(c)$.

Nous pouvons maintenant énoncer le théorème validant notre approche (rappelons que G est consistante) :

Théorème 3
$G \models F$ si et seulement si $\mathcal{DCMS}(F, G)$ est valide.

Pour la démonstration, nous avons besoin des définitions et lemmes suivants :

Définition 12 (Conjonction totale)
Soit G^c une complétion totale de G. On appelle conjonction totale, *notée C, la conjonction composée des littéraux $l \in (G^c - G)$.*

Définition 13 (Disjonction des Conjonctions Totales ($\mathcal{DCT}$))
On appelle $\mathcal{DCT}(G)$ la disjonction des conjonctions totales de G (i.e. $C_1 \vee \cdots \vee C_n$).

Lemme 1
$\mathcal{DCT}(G)$ est valide.

Preuve. Par récurrence sur la profondeur de l'arborescence de complétion. □

Lemme 2
Si $G \models F$ alors quelque soit $C \in \mathcal{DCT}(G)$, il existe $C^m \in \mathcal{DCMS}(F, G)$ tel que $C^m \subseteq C$.

Preuve. Soit C une conjonction totale apparaissant dans $\mathcal{DCT}(G)$. Comme $G \models F$, il existe un homomorphisme h de F dans $G \wedge C$ (cf. théorème 1). D'après la propriété 2, h est un homomorphisme partiel de F dans G par rapport à F^s. D'après la propriété 3, h est un homomorphisme de F dans la complétion minimale $G \wedge C^m$ (où C^m est la conjonction minimale de G issue de h). Enfin par définition, C^m ne contient que des littéraux l' tels que $l' = h(l)$ où l est un littéral de F, donc l' est aussi un littéral de C. □

[5] nécessaire lorsqu'il n'y a pas d'homomorphisme partiel de F dans G par rapport à F^s car sinon la disjonction est vide.

Preuve du théorème 3 :

$\Longleftarrow$ Puisque $\mathcal{DCMS}(F,G)$ est valide, $G \equiv G \wedge \bigvee C^m \equiv \bigvee (G \wedge C^m)$. D'après la propriété 4, on a $\bigvee (G \wedge C^m) \models F$. Donc $G \models F$.

$\Longrightarrow$ Soit $G \models F$. Alors d'après le lemme 2 quelque soit $C \in \mathcal{DCT}(G)$, il existe $C^m \in \mathcal{DCMS}(F,G)$ tel que $C = C^m \wedge C'$ où C' est une conjonction de littéraux. *Par l'absurde* : supposons $\mathcal{DCMS}(F,G)$ non valide. Alors il existe une interprétation $\mathcal{I}$ telle que pour tout $C^m \in \mathcal{DCMS}(F,G)$, $v(C^m,\mathcal{I}) = 0$. Donc quelque soit $C \in \mathcal{DCT}(G)$, $v(C,\mathcal{I}) = 0$. Donc $\mathcal{DCT}(G)$ n'est pas valide, ce qui contredit le lemme 1. Donc $\mathcal{DCMS}(F,G)$ est valide. □

3.3 Passage à SAT/UNSAT

Nous avons donc transformé le problème de déduction initial en un problème de test de validité d'une formule propositionnelle, $\mathcal{DCMS}(F,G)$. La négation de cette formule est directement une forme clausale, dont il nous faut tester l'insatisfiabilité. Nous pouvons pour cela utiliser un solveur SAT.

Exemple 7
$FC = \overline{\mathcal{DCMS}(F,G)} = p(b) \wedge (\neg p(b) \vee p(c)) \wedge \neg p(c)$. FC est insatisfiable, donc $\mathcal{DCMS}(F,G)$ est valide, d'où $G \models F$.

3.4 Taille de la formule obtenue

On passe ainsi d'un problème Π_2^P-$complet$ à un problème co-NP-$complet$; toutefois ce deuxième problème prend en entrée une formule dont la taille peut être exponentielle en la taille des formules de départ. Bornons plus précisément la taille de la formule propositionnelle obtenue.

Appelons nb_{hom} le nombre d'homomorphismes partiels de F dans G par rapport à F^s. Ce nombre représente le nombre de conjonctions minimales obtenues dans la formule propositionnelle finale. Il est borné par $n_G^{n_F}$, où n_G et n_F sont respectivement le nombre de termes dans G et F. Néanmoins en pratique, nous nous attendons à ce que plus la taille de F^s soit grande, plus le nombre d'homomorphismes partiels de F dans G par rapport à F^s soit faible, puisque chaque littéral de F^s représente potentiellement une contrainte en plus lors de la recherche d'un homomorphisme partiel de F dans G par rapport à F^s.

La taille d'une conjonction minimale est quant à elle bornée par : $L = |F - F^s|$.

La taille de la formule propositionnelle obtenue est donc bornée par : $nb_{hom} * |F - F^s|$.

D'une part, plus la taille de F^s est grande, plus la taille d'une conjonction minimale est petite, d'autre part on s'attend à ce que nb_{hom} diminue avec la croissance de F^s : ces deux points laissent penser que l'efficacité de cette méthode sera directement corrélée à l'importance de la sous-formule stable dans F, ce qui reste toutefois à vérifier expérimentalement.

Il est également à noter que nb_{hom} peut être borné plus précisément : ce n'est pas le nombre d'homomorphismes partiels de F dans G par rapport à F^s qui importe véritablement, mais le nombre de substitutions différentes sur les termes de L que l'on

obtient par ces homomorphismes partiels. En effet, deux homomorphismes partiels de F dans G par rapport à F^s différents h_1 et h_2 mais tels que $h_1(L) = h_2(L)$ amèneront la création de la même conjonction minimale.

4 Conclusion et perspectives

La démarche présentée dans (Leclère & Mugnier, 2007) consistait à construire incrémentalement un ensemble couvrant toutes les formules complètes générables à partir de G, tel que F se déduise de chaque élément de cet ensemble. Le point crucial est le choix, à chaque étape de complétion, du littéral à ajouter pour construire un ensemble couvrant de taille minimale.

L'approche présentée dans ce papier consiste à construire une formule propositionnelle en tirant parti d'un ensemble d'homomorphismes partiels préalablement calculé, puis d'en tester la validité, ramenant ainsi le problème étudié à un problème de validité en logique propositionnelle. Ceci nous permet de nous servir des avancées réalisées ces dernières années sur le problème SAT.

Nous expérimentons actuellement cette approche afin de la comparer aux précédentes (notamment (Leclère & Mugnier, 2007)). Nous pensons pouvoir identifier des critères permettant de choisir une méthode plutôt qu'une autre selon la structure de F (et peut-être de G) comme par exemple la taille de la sous-formule stable. Enfin, nous espérons trouver un certain nombre de propriétés sur la formule obtenue, allant de bornes sur la taille de cette dernière à des propriétés de filtrage et d'amélioration du test de validité.

Références

ABITEBOUL S., HULL R. & VIANU V. (1995). *Foundations of Databases : The Logical Level*. Addison-Wesley.

CHANDRA A. & MERLIN P. (1977). Optimal implementation of conjunctive queries in relational databases. In *9th ACM Symposium on Theory of Computing*, p. 77–90.

GOTTLOB G. (1987). Subsumption and implication. *Inf. Process. Lett.*, **24**(2), 109–111.

HALEVY A. Y. (2001). Answering queries using views : A survey. *VLDB Journal : Very Large Data Bases*, **10**(4), 270–294.

LAVRAC N. & DZEROSKI S. (1994). *Inductive Logic Programming : Techniques and Applications*. Ellis Horwood.

LECLÈRE M. & MUGNIER M.-L. (2006). Simple Conceptual Graphs with Atomic Negation and Difference. In *Proc. of ICCS'06*, volume 4068 of *LNAI*, p. 331–345 : Springer.

LECLÈRE M. & MUGNIER M.-L. (2007). Some Algorithmic Improvments for the Containment Problem of Conjunctive Queries with Negation. In *Proc. of ICDT'07*, volume 4353 of *LNCS*, p. 401–418 : Springer.

MUGGLETON S. (1992). *Inductive Logic Programming*. Academic Press.

ULLMAN J. D. (1997). Information Integration Using Logical Views. In *Proc. of ICDT'97*, volume 1186 of *LNCS*, p. 19–40 : Springer.

WEI F. & LAUSEN G. (2003). Containment of Conjunctive Queries with Safe Negation. In *International Conference on Database Theory (ICDT)*.

An axiomatization and a tableau calculus for the logic of comparative concept similarity

Régis Alenda[1], Nicola Olivetti[1], Camilla Schwind[2]

[1] LSIS - UMR CNRS 6168
Domaine Universitaire de Saint-Jérôme, Avenue Escadrille Normandie-Niemen ,
13397 MARSEILLE CEDEX 20
regis.alenda@lsis.org et nicola.olivetti@univ-cezanne.fr

[2] LIF - UMR CNRS 6166
Centre de Mathématiques et Informatique
39 rue Joliot-Curie - F-13453 Marseille Cedex13.
camilla.schwind@lif.univ-mrs.fr

Résumé : La logique de *similarité comparative des concepts* $\mathcal{CSL}$ a été introduite en 2005 par Shremet, Tishkowsky, Wolter et Zakharyaschev pour représenter des informations qualitatives sur la similarité entre des concepts, du type "A est plus similaire à B qu'à C". La sémantique utilise des espaces de distances afin de représenter le degré de similarité entre objets du domaine. Dans cet article, nous étudions $\mathcal{CSL}$ sur les *minspaces*, i.e des espaces de distances dans lesquels tout ensemble de distances possède un minimum, et donnons la première axiomatisation directe de cette logique dans ce contexte, ainsi qu'une méthode de preuve à tableaux. A notre connaissance, notre calcul est la première méthode de preuve pratique pour $\mathcal{CSL}$.

Mots-clés : logiques modales, procédures à tableaux, logiques de description, similarité comparative des concepts

1 Introduction

The logics of comparative concept similarity $\mathcal{CSL}$ have been recently proposed by Sheremet, Tishkovsky, Wolter et Zakharyaschev in (Sheremet *et al.*, 2005) to capture a form of qualitative comparison between concept instances. In these logics we can express assertions or judgments of the form : "'Peugeot 207 is more similar to Renault Clio than to Porche Cayenne"', or "'Tuscan order is more similar to Doric order than to Ionic order"', as we might like to express in a KB about archeology. These logics could find a natural application in ontology languages, whose logical base is provided by Description Logics.

The language of $\mathcal{CSL}$ is obtained by the addition of a binary modal connective $\trianglelefteq$ to an underlying language, so that the above examples can be encoded (using a description

logic notation) by :

$$Peugeot207 \sqsubseteq (Clio \leftleftarrows PorcheCayenne)$$

$$TuscanOrder \sqsubseteq (DoricOrder \leftleftarrows IonicOrder)$$

The semantics of $\mathcal{CSL}$ is defined in terms of distance spaces, that is to say structures equipped by a distance function d, whose properties may vary according to the logic under consideration. In this setting, the evaluation of $A \leftleftarrows B$ can be informally stated as follows : $x \in (A \leftleftarrows B)$ iff $d(x, A) < d(x, B)$ meaning that the object x is an instance of the concept $A \leftleftarrows B$ (i.e. it belongs to things that are more similar to A than to B) if x is strictly closer to A-objects than to B-objects according to distance function d, where the distance of an object to a set of objects is defined as the *infimum* of the distances to each object in the set.

In a series of papers (Sheremet *et al.*, 2005, 2008; Kurucz *et al.*, 2005; Sheremet *et al.*, 2007), the authors have investigated the logic $\mathcal{CSL}$ with respect to different classes of distance models, see (Sheremet *et al.*, 2008) for a survey of results about decidability, complexity, expressivity, and axiomatisation. Remarkably it is shown that $\mathcal{CSL}$ is undecidable over subspaces of the reals. Moreover $\mathcal{CSL}$ can be seen as a fragment, indeed a powerful one (including for instance the logic $\mathbf{S4}_u$ of topological spaces), of a general logic for spatial reasoning comprising different modal operators defined by (bounded) quantified distance expressions.

The authors have pointed out that in case the distance spaces are assumed to be *minspaces*, that is spaces where the infimum of a set of distances is actually their *minimum*, the logic $\mathcal{CSL}$ is naturally related to some conditional logics. The semantics of the latter is often expressed in terms of preferential structures, that is to say possible-world structures equipped by a family of strict partial (pre)-orders $\prec_x$ parametrised on objects. The intended meaning of the relation $y \prec_x z$ is namely that x is more similar to y than to z.

In this paper we contribute to the study of $\mathcal{CSL}$ on minspaces. The minspace property is essentially equivalent to the restriction to spaces where the distance function is discrete. This requirement does not seem in contrast with the purpose of representing qualitative comparisons of similarity.

Similarly to conditional logics, we consider a propositional language extended by the $\leftleftarrows$ connective. In this setting (as opposed to concept/subset interpretation), the formula $A \leftleftarrows B$ may be naturally read as "'A is (strictly) more plausible than B"'. We first show that the semantics of $\mathcal{CSL}$ on minspaces can be equivalently restated in terms of preferential models satisfying some additional conditions, namely modularity, centering, and limit assumption. We then give the first sound, complete and direct axiomatisation of this logic ; this problem was left open in the recent (Sheremet *et al.*, 2008). Furthermore, we define a tableaux calculus for checking satisfiability of $\mathcal{CSL}$ formulas. Our tableaux procedure makes use of labelled formulas and pseudo-modalities indexed on worlds $\Box_x$, similarly to the calculi for conditional logics defined in (Giordano *et al.*, 2003, 2009). To the best of our knowledge our calculus provides the first known practical decision procedure for this logic.

2 The logic of *Comparative Concept Similarity* $\mathcal{CSL}$

The langage $\mathcal{L}_{\mathcal{CSL}}$ of $\mathcal{CSL}$ is generated from a set of propositional variables V_i by the following grammar :

$$A, B \ ::= \ V_i \mid \neg A \mid A \wedge B \mid A \leftleftarrows B.$$

The others propositional connectives are defined as usual.

The semantic of $\mathcal{CSL}$ introduced in (Sheremet *et al.*, 2005) makes use of *distance spaces* in order to represent the similarity degree between possible worlds. A distance space is a pair (Δ, d) where Δ is a non-empty set, and $d : \Delta \rightarrow \mathbb{R}^{\geq 0}$ is a *distance function* satisfying the following condition :

(ID) $$\forall x, y \in \Delta, \ \ d(x, y) = 0 \text{ iff } x = y$$

Two further properties are usually assumed : symmetry and triangle inequality. We briefly discuss them at the end of this section.

The distance between an object w and a non-empty subset X of Δ is defined by $d(w, X) = \inf\{d(w, x) \mid x \in X\}$. If $X = \emptyset$, then $d(w, X) = \infty$. If for every onject w and for every (non-empty) subset X we have the following property

(MIN) $$\inf \{d(w, x) \mid x \in X\} = \min \{d(w, x) \mid x \in X\},$$

we will say that (Δ, d) is a *min-space*.

We next define $\mathcal{CSL}$-distance models as Kripke models based on distance spaces :

Definition 2.1 ($\mathcal{CSL}$-distance model)
A $\mathcal{CSL}$-distance model is a triple $\mathcal{M} = (\Delta, d, .^{\mathcal{M}})$ where :

- *Δ is a non-empty set of* objects *(or* possible worlds*).*
- *d is a distance on $\Delta^{\mathcal{M}}$ (so that (Δ, d) is a distance space).*
- *$.^{\mathcal{M}} : \mathcal{V}_p \rightarrow 2^{\Delta}$ is the* evaluation function *which assigns to each propositional variable V_i a set $V_i^{\mathcal{M}} \subseteq \Delta$. $V_i^{\mathcal{M}}$ can be seen as the set of possible worlds where V_i is true. We also stipulate $\bot^{\mathcal{M}} = \emptyset$. For complex formulas, $.^{\mathcal{M}}$ is defined inductively as follows :*

$$(\neg C)^{\mathcal{M}} = \Delta - C^{\mathcal{M}}$$
$$(C \wedge D)^{\mathcal{M}} = C^{\mathcal{M}} \cap D^{\mathcal{M}}$$
$$(C \leftleftarrows D)^{\mathcal{M}} = \left\{w \in \Delta \,\middle|\, d(w, C^{\mathcal{M}}) < d(w, D^{\mathcal{M}})\right\}.$$

If (Δ, d) is a min-space, $\mathcal{M}$ is called a $\mathcal{CSL}$-distance minmodel.

We say that a formula A is valid *in a model $\mathcal{M}$ if $A^{\mathcal{M}} = \Delta$. We say that a formula A is* valid *if A is valid in every $\mathcal{CSL}$-distance model.*

As mentioned above, the distance function might be required to satisfy the further conditions of symmetry (SYM) ($d(x, y) = d(y, x)$) and triangular inequality (TR) ($d(x, z) \leq d(x, y) + d(y, z)$). It turns out that $\mathcal{CSL}$ cannot distinguish between min-models which satisfy (TR) from models which do not. In contrast, $\mathcal{CSL}$ has enough expressive power in order to distinguish between symmetric and non-symmetric min-models[1]. We intend to consider symmetric models in future research.

[1] See (Sheremet *et al.*, 2005).

3 A preferential semantics for $\mathcal{CSL}$

$\mathcal{CSL}$ is a logic of pure qualitative comparisons. This motivates an alternative semantics where the distance function is replaced by a family of comparisons relations, one for each object. We call this semantics *preferential* semantic, similarly to conditional logics. Technically, preferential structures are equipped by a family of strict preorders. We may interprete this relations as expressing a similarity information between objects. For three worlds/objects, $x \prec_w y$ states that w is more similar to x than to y.

The preferential semantic in itself is more general than distance model semantic. However, if we assume the additional conditions of the definition 3.1, it turns out that these two are equivalent (theorem 3.3).

Definition 3.1
We will say that a preferential relation $\prec_w$ over Δ :

(i) is modular *iff* $\forall x, y, z \in \Delta$, $(x \prec_w y) \rightarrow (z \prec_w y \ \vee \ x \prec_w z)$.

(ii) is centered *iff* $\forall x \in \Delta$, $x = w \ \vee \ w \prec_w x$.

(iii) satisfies the limit assumption *iff* $\forall X \subseteq \Delta$, $X \neq \emptyset \ \rightarrow \ \min_{\prec_w}(X) \neq \emptyset$.

Modularity is strongly related to the fact that the preferential relations represents distance comparisons. This is the key property to enforce the equivalence with distance models. Centering states that w is the *unique* minimal element for its preferential relation $\prec_w$, and can be seen as the preferential counterpart of (ID). The limit assumption states that each non-empty set has at least one minimal element wrt. a preferential relation (i.e it does not contain an infinitely descending chain), and corresponds to (MIN).

Definition 3.2 ($\mathcal{CSL}$-preferential model)
A $\mathcal{CSL}$-preferential model is a triple $\mathcal{M} = (\Delta, (\prec_w)_{w\in\Delta}, .^{\mathcal{M}})$ where :

- $\Delta^{\mathcal{M}}$ *is a non-empty set of* objects *(or* possible worlds*).*
- $(\prec_w)_{w\in\Delta}$ *is a family of* preferential relation*, each one being transitive, irreflexive, asymmetric, modular, centered, and satisfying the* limit assumption.
- $.^{\mathcal{M}}$ *is the evaluation function defined as in definition 2.1, except for* $\leftleftarrows$ *:*

$$(A \leftleftarrows B)^{\mathcal{M}} = \left\{ w \in \Delta \middle| \exists x \in A^{\mathcal{M}} \text{ such that } \forall y \in B^{\mathcal{M}},\ x \prec_w y \right\}$$

Validity is defined as in definition 2.1.

We now show the equivalence between preferential models and distance minmodels. We say that a $\mathcal{CSL}$-preferential model $\mathcal{I}$ and a $\mathcal{CSL}$-distance minmodel $\mathcal{J}$ are *equivalent* iff they are based on the same set Δ, and for all formulas $A \in \mathcal{L}_{\mathcal{CSL}}$, $A^{\mathcal{I}} = A^{\mathcal{J}}$.

Theorem 3.3 (Equivalence between $\mathcal{CSL}$-preferential models and $\mathcal{CSL}$-distance models)

1. *For each $\mathcal{CSL}$-distance min-model, there is an equivalent $\mathcal{CSL}$-preferential model.*
2. *For each $\mathcal{CSL}$-preferential model, there is an equivalent $\mathcal{CSL}$-distance min-model.*

Sketch of the proof

1. Contained in (Sheremet *et al.*, 2005).
2. Since the relation $\prec_w$ is modular, we can assume that there exists a *ranking function* r_w such that $x \prec_w y$ iff $r_w(x) < r_w(y)$. Therefore, given a $\mathcal{CSL}$-preferential model $\mathcal{J} = (\Delta^{\mathcal{J}}, (\prec_w)_{w \in \Delta^{\mathcal{J}}}, .^{\mathcal{J}})$, we can define a $\mathcal{CSL}$-distance min-model $\mathcal{I} = (\Delta^{\mathcal{J}}, d, .^{\mathcal{J}})$, where the distance function d is defined as follow : if $w = x$ then $d(w, x) = 0$, and $d(w, x) = r_w(x)$ otherwise. We can easily check that I is a min-space (by the limit assumption), and that $\mathcal{I}$ and $\mathcal{J}$ are equivalent.

4 An axiomatization of $\mathcal{CSL}$ over minspaces

We now give an axiomatisation of $\mathcal{CSL}$ over minspaces. An axiomatisation of $\mathcal{CSL}$ in various classes of models can be found in (Sheremet *et al.*, 2008), but it makes use of an extended language. Moreover, the case of minspaces has not been studied yet, and it does not seem that our axioms can be easily derived from (Sheremet *et al.*, 2008).

$(Ax\bot)$	$\neg(\bot \leftleftarrows A)$	$(AxTR)$	$(A \leftleftarrows B) \wedge (B \leftleftarrows C) \rightarrow (A \leftleftarrows C)$
$(AxAS)$	$\neg(A \leftleftarrows B) \vee \neg(B \leftleftarrows A)$	$(Ax\neg B)$	$(A \leftleftarrows B) \rightarrow \neg B$
$(AxMN)$	$A \wedge \neg B \rightarrow (A \leftleftarrows B)$	$(AxMD)$	$(A \leftleftarrows B) \rightarrow (A \leftleftarrows C) \vee (C \leftleftarrows B)$
$(Ax\vee)$	$(A \leftleftarrows B) \wedge (A \leftleftarrows C) \rightarrow (A \leftleftarrows (B \vee C))$	$(Ax\wedge)$	$(A \leftleftarrows B) \rightarrow ((A \wedge \neg B) \leftleftarrows B)$
$(AxU1)$	$\neg(A \leftleftarrows \bot) \rightarrow \neg((A \leftleftarrows \bot) \leftleftarrows \bot)$	$(AxU2)$	$(A \leftleftarrows \bot) \rightarrow \neg(\neg(A \leftleftarrows \bot) \leftleftarrows \bot)$
(Rr)	$\dfrac{\vdash (A \rightarrow B)}{\vdash (A \leftleftarrows C) \rightarrow (B \leftleftarrows C)}$	(Rl)	$\dfrac{\vdash (A \rightarrow B)}{\vdash (C \leftleftarrows B) \rightarrow (C \leftleftarrows A)}$
$(Taut)$	Classical tautologies and rules.		

FIG. 1 – **CSMS** axioms.

Our axiomatisation, named **CSMS** is presented in figure 1. If we interpret $A \leftleftarrows B$ as "A is more plausible than B", we can give the following intuitive meanings to the axioms : $(Ax\bot)$ states that a contradiction cannot be more plausible than any formula. $(Ax\vee)$ states that if a formula A is more plausible than two others, A is more plausible than their disjunction. $(Ax\wedge)$ states that if a formula A is more plausible than a formula B, then $A \wedge \neg B$ is more plausible than A. $(AxTR)$, $(AxAS)$ and $(AxMD)$ represent respectively transitivity, asymmetry and modularity of the preferential relation. $(Ax\neg B)$ and $(AxMN)$ are needed to express the centering and limit assumption conditions.

$(AxU1)$ and $(AxU2)$ were introduced for a technical purpose. They are the translations in $\mathcal{CSL}$ of the modal **S5** axioms $\Box A \rightarrow \Box\Box A$ and $\Diamond A \rightarrow \Box\Diamond A$[2], meaning that all preference relations have the same range. The rules (Rr) and (Rl) express the monotony of $\leftleftarrows$ in the first argument, and the anti-monotony in the second one.

We can show that our axiomatisation is sound and complete with respect to the preferential semantics introduced in section 3.

[2]Note that we can define $\Diamond A$ in $\mathcal{CSL}$ by $(A \leftleftarrows \bot)$.

Theorem 4.1 (Soundness of CSMS)
(Soundness) If a formula is derivable in **CSMS**, *then it is valid in every $\mathcal{CSL}$-preferential model.*
(Completeness) If a formula is valid in every $\mathcal{CSL}$-preferential model, then it is derivable in **CSMS**.

Sketch of the proof

(Soundness) By induction on the derivation length. We check that the axioms are valid in all $\mathcal{CSL}$-preferential models, and that the rules preserve validity.

(Completeness) We show the contrapositive : if a formula is not derivable in **CSMS**, then its negation is satisfiable in some $\mathcal{CSL}$-preferential model. For this, we show how to construct for a given non-derivable formula C a canonical model in which $\neg C$ is satisfiable.

To begin, let U be the set of all *maximal consistent* sets for $\mathcal{L}_{\mathcal{CSL}}$. We define a binary relation R over U by $R(x, y)$ iff $\forall A \in \mathcal{L}_{\mathcal{CSL}},\ A \in y \rightarrow (A \leftrightharpoons \bot) \in x$. We can prove that R is an equivalence relation (by virtue of $(AxU1)$ and $(AxU2)$). For all $x \in U$, we let $[x]$ be its equivalence class with respect to R.

Since C is not derivable in **CSMS**, $\neg C$ is consistent, and so there is a maximal consistent set $z \in U$ such that $\neg C \in z$. We can define a model (called canonical model) $\mathcal{M}_C = (\Delta, (\prec_w)_{w\in\Delta}, .^{\mathcal{M}_C})$ as follow :

– $\Delta = [z]$.
– $x \prec_w y$ iff there exists a formula $B \in y$ such that for all formulas $A \in x$, $(A \leftrightharpoons B) \in w$.
– $V_i^{\mathcal{M}_C} = \{x \in \Delta \mid V_i \in x\}$, for all propositional variables V_i.

We can check that each preferential relation $\prec_w$ is centered, modular, and satisfies the limit assumption, and that for all formulas A and for all worlds $w \in \Delta$, we have $w \in A^{\mathcal{M}}$ iff $A \in w$. Since $z \in \Delta$ and $\neg C \in z$, we obtain that $(\neg C)^{\mathcal{M}_C} \neq \emptyset$, and thus that C is not valid. Details are in (Alenda, 2008).

By virtue of theorem 3.3, we obtain :

Corollary 4.2
CSMS *is sound and complete wrt. the $\mathcal{CSL}$-distance min-models.*

5 Tableau algorithm

In this section, we sketch a tableau-based decision procedure for $\mathcal{CSL}$. As usual, a tableau is a tree whose branches are sets of formulas. These formulas are either the initial formulas or they are obtained from previous formulas by the application of tableau rules, the rules may produce branching in the tree.

Our calculus makes use of labels to represent possible worlds. In order to check whether a formula A is satisfiable, we initialise a tableau by $x : A$ for an arbitrary label x. Informally, the tableau is construction proceeds as follows : we apply the tableaux rules to each formula in a branch until, either we detect a contradition (the branch is closed), or no new formula is introduced in the branch (the branch is saturated). If every branch

is closed A is not satisfiable, otherwise it is satisfiable and each open branch specifies a model of it. In order to check the validity of a formula A, we check whether its negation is satisfiable, so that A is valid if the tableau initialised with $x : \neg A$ is closed.

Tableau rules encode the semantics of the formulas. It is well known how this works for boolean operators. Let us look at the formulas $(A \leftrightarrows B)$ and $\neg(A \leftrightarrows B)$ under preferential semantics. We have :

$$w \in (A \leftrightarrows B)^{\mathcal{M}} \text{ iff } \exists x(x \in A^{\mathcal{M}} \wedge \forall z(z \in B^{\mathcal{M}} \rightarrow x \prec_w z))$$

In minspaces, the right part is equivalent to :

$$\exists u\, u \in A^{\mathcal{M}} \wedge \forall y(y \in B^{\mathcal{M}} \rightarrow \exists x(x \in A^{\mathcal{M}} \wedge x \prec_w y))$$

We introduce a pseudo-modality $\Box_w$ indexed on possible worlds.

$$x \in (\Box_w A)^{\mathcal{M}} \text{ iff } \forall y(y \prec_w x \rightarrow y \in A^{\mathcal{M}})$$

This pseudo-modality is needed for a technical purpose. Its meaning is that $x \in (\Box_w A)^{\mathcal{M}}$ iff A holds in all preferred worlds to x with respect to $\prec_w$. It makes it possible to obtain analytic rules for $A \leftrightarrows B$.

By means of this modality, we obtain the following equivalence :

$$w \in (A \leftrightarrows B)^{\mathcal{M}} \text{ iff } A^{\mathcal{M}} \neq \emptyset \ \wedge \ \forall y(y \notin B^{\mathcal{M}} \vee y \in (\neg\Box_w \neg A)^{\mathcal{M}}$$

This last equivalence yields the tableaux rules $(T \leftrightarrows)$ and $(F \leftrightarrows)$. Figure 2 shows all tableaux rules for our logic.

Let us point out that rule $(F\Box_x)$ introduces the formula $\Box_x \neg A$. This corresponds to the limit assumption and will prevent the calculus to produce infinit descending chains. We can show that no tableau contains infinite descending chains of labels related by the same relation $<_x$, provided it does not contain an infinite number of positive $\leftrightarrows$-formulas, i.e. formulas of the form $x : \phi \leftrightarrows \psi$ labelled with the same label x. A formal proof can be found in (Alenda, 2008)

Definition 5.1 (Closed branch, closed tableau)
A branch $\mathbf{B}$ *of a* $\mathcal{CSL}$*-tableau is* closed *if one of the three following conditions hold : (i)* $x : A \in \mathbf{B}$ *and* $x : \neg A \in \mathbf{B}$*, for any formula* A*, or* $x : \bot \in \mathbf{B}$*. (ii)* $y <_x y \in \mathbf{B}$*. (iii)* $y <_x x \in \mathbf{B}$*.*

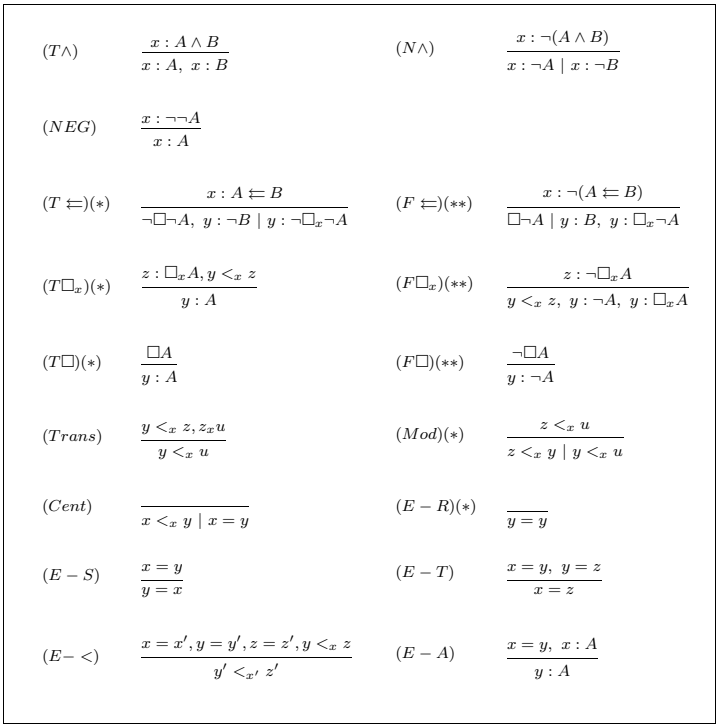
A $\mathcal{CSL}$*-tableau is closed if every branch is closed.*

In order to prove soundness and completeness of the tableaux rules, we introduce the notion of satisfiability of a branch by a model.

Given a branch B, we denote by W_B the set of labels occurring in B.

Definition 5.2 ($\mathcal{CSL}$**-mapping, satisfiable branch)**
Let $\mathcal{M} = (\Delta^{\mathcal{M}}, (\prec_w)_{w \in \Delta^{\mathcal{M}}}, .^{\mathcal{M}})$ *a* $\mathcal{CSL}$*-preferential model, and* $\mathbf{B}$ *a branch of a* $\mathcal{CSL}$*-tableau. A* $\mathcal{CSL}$*-mapping from* $\mathbf{B}$ *to* $\mathcal{M}$ *is a function* $f : W_{\mathbf{B}} \longrightarrow \Delta^{\mathcal{M}}$ *satisfying the two following conditions :*

(i) for every $y <_x z$ *in* $\mathbf{B}$*, we have* $f(y) \prec_{f(x)} f(z)$ *in* $\mathcal{M}$*.*



(*) y is a label occuring in the branch
(**) y is a new label not occuring in the branch

FIG. 2 – Tableau rules for $\mathcal{CSL}$.

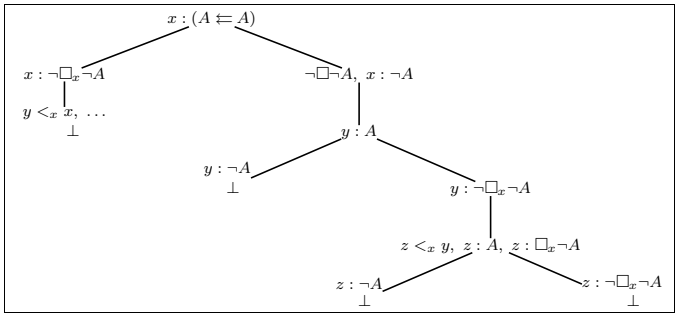


FIG. 3 – An exemple of tableau : provability of $\neg(A \leftleftarrows A)$.

(ii) for every $x = y$ in $\mathbf{B}$, we have $f(x) = f(y)$ in $\mathcal{M}$.

Given a branch $\mathbf{B}$ of a $\mathcal{CSL}$-tableau, a $\mathcal{CSL}$-preferential model $\mathcal{M}$, and a $\mathcal{CSL}$-mapping f from $\mathbf{B}$ to $\mathcal{M}$, we say that $\mathbf{B}$ is satisfiable under f in $\mathcal{M}$ if

$$x : A \in \mathbf{B} \text{ implies } f(x) \in A^{\mathcal{M}}.$$

A branch $\mathbf{B}$ *is satisfiable if it is satisfiable in some* $\mathcal{CSL}$*-preferential model* $\mathcal{M}$ *under some* $\mathcal{CSL}$*-mapping* f*. A* $\mathcal{CSL}$*-tableau is satisfiable if at least one of its branches is satisfiable.*

Theorem 5.3 (Soundness and completeness of the calculus)

(Soundness) If the tableau starting by $x : \neg A$ *is closed, then* A *is* $\mathcal{CSL}$*-valid (wrt. the preferential semantics).*

(Completeness) If a formula is $\mathcal{CSL}$*-valid (wrt. our preferential semantics), then the tableau starting by* $x : \neg A$ *is closed.*

Sketch of the proof

(Soundness) The proof is standard : we show that rule application preserves satisfiability.

(Completeness) The completeness of the tableaux calculus is proved in the standard way : we show by contraposition that if a formula is not provable by the calculus then it is not valid. To this purpose we need a notion of saturated branch (see (Alenda, 2008)). Intuitively a saturated branch is a branch that is closed under the application of every rule of the calculus, that is to say if it contains the premise of the rule, then it contains at least one of its conclusions. Then the proof runs as follows : if a C is not provable, then there exists an open saturated branch B containing $x : \neg C$. We can define a model M_B associated to B which satisfies all formulas occurring in B under a suitable mapping, thus in particular it satisfies $x : \neg C$. Details can be found in (Alenda, 2008).

Corollary 5.4

Our calculus is sound and complete wrt. the $\mathcal{CSL}$*-distance min-models.*

The calculus presented above can lead to non-terminating computations due to the interplay between the rules which generate new labels (the dynamic rules $(F \leftleftarrows)$, $(F\Box)$ and $(F\Box_x)$) and the static rule $(T \leftleftarrows)$ which generates formula $\neg\Box_x A$ to which $(F\Box_x)$ may again be applied. For instance, the tableaux construction for $w : p \leftleftarrows (q \leftleftarrows r)$ can generate an infinite branch containing $x_1 : r$, $x_1 : \Box_x \neg q$, $x_1 : \neg(q \leftleftarrows r)$, $x_2 : r$, $x_2 : \Box_x \neg q$, $x_2 : \neg(q \leftleftarrows r)$, …. Our calculus can be made terminating, without loosing completeness, by defining a systematic procedure for applying the rules and by introducing appropriate blocking conditions.

To this aim, we first define a total ordering $\prec$ on the labels of a branch such that $x \prec y$ for all labels x that are already in the tableau when y is introduced. The systematic procedure for constructing the tableau for formula $x : A$ applies first all static rules as far as possible and then applies one dynamic rules to some formula labelled x only if no dynamic rule is applicable to a formula labelled y, such that $y \prec x$.

In order to stop the infinite expansion of a branch, we consider an equivalence relation on labels : given a branch B, we say that two labels x and y are $B-equivalent$, denoted by $x \equiv_B y$, if they label the same set of formulas in B, ignoring modal-pseudo formulas of type $(\neg)\Box_u C \in B$ (on which they might differ). Since the tableau is initialised by a finite number of formulas, there can be only a finite number of labels that are not $\equiv_B$-equivalent in any branch B. Thus a loop-checking mechanism can be devised to

ensure termination. Moreover, it can be shown, similarly to (Giordano *et al.*, 2009), that idenditying $\equiv_B$ labels preserve the completeness of the method (an open branch will not become close). Details will be given in a full paper.

6 Conclusion

In this paper, we have studied the logic $\mathcal{CSL}$ over minspaces, and we have obtained two main results : first we have provided a direct, sound and complete axiomatisation of this logic. Furthermore, we defined a tableau calculus, which gives a decision procedure for this logic.

There are a number of issues to explore in future research. The decision procedure outlined in the previous section is not guaranteed to have an optimal complexity. To this concern, it is shown in (Sheremet *et al.*, 2005) that $\mathcal{CSL}$ is EXPTIME-complete, so that we can consider how to improve our calculus in order to match this upper bound. Another issue is the extension of our results to symmetric minspaces, and possibly to other classes of models. Finally, since one original motivation of $\mathcal{CSL}$ is to reason about concept similarity in ontologies, and particularly in description logics, we plan to study further its integration with this family of formalisms.

Références

ALENDA R. (2008). Logique et raisonnement automatique pour la comparative concept similarity. Master's thesis, Université de Provence, Aix-Marseille I.

GIORDANO L., GLIOZZI V., OLIVETTI N. & SCHWIND C. (2003). Tableau calculi for preference-based conditional logics. In M. C. MAYER & F. PIRRI, Eds., *TABLEAUX 2003*, volume 2796 of *Lecture Notes in Artificial Intelligence*, p. 81–101 : Springer.

GIORDANO L., GLIOZZI V., OLIVETTI N. & SCHWIND C. (2009). Tableau calculus for preference-based conditional logics : Pcl and its extensions. *ACM Transactions on Computational Logic (TOCL)*. à paraître.

KURUCZ A., WOLTER F. & ZAKHARYASCHEV M. (2005). Modal logics for metric spaces : Open problems. In S. N. ARTËMOV, H. BARRINGER, A. S. D'AVILA GARCEZ, L. C. LAMB & J. WOODS, Eds., *We Will Show Them ! (2)*, p. 193–108 : College Publications.

SHEREMET M., TISHKOVSKY D., WOLTER F. & ZAKHARYASCHEV M. (2005). Comparative similarity, tree automata, and diophantine equations. In G. SUTCLIFFE & A. VORONKOV, Eds., *LPAR 2005*, volume 3835 of *Lecture Notes in Computer Science*, p. 651–665 : Springer.

SHEREMET M., TISHKOVSKY D., WOLTER F. & ZAKHARYASCHEV M. (2007). A logic for concepts and similarity. *J. Log. Comput.*, **17**(3), 415–452.

SHEREMET M., WOLTER F. & ZAKHARYASCHEV M. (2008). A modal logic framework for reasoning about comparative distances and topology. submitted.

Extraire et modéliser des préférences à partir d'un dialogue

Nicholas Asher[1], Elise Bonzon[2]

[1] IRIT, Université Paul Sabatier, Toulouse, asher@irit.fr

[2] CRIP5, Université Paris Descartes, elise.bonzon@mi.parisdescartes.fr

Résumé : Dans de nombreux dialogues, les participants s'engagent sur certaines préférences, puis les développent ou les révisent. Ce travail, encore préliminaire, propose une méthode basée sur la structure discursive du dialogue pour extraire et modéliser ces préférences.

Mots-clés : Représentation de préférences, Dialogues, CP-nets.

1 Introduction

Il est en général bien accepté que les dialogues sont structurés par les actes effectués par les participants. Chaque participant répond aux questions posées par d'autres acteurs du dialogue, puis pose les questions qui en découlent. Ils expliquent leurs annonces, classent les événements temporellement et spatialement. Ils doivent élaborer et défendre leurs revendications. La plupart de ces actes de langage effectués par un agent *A* peuvent affecter la façon dont les autres acteurs du dialogue visualisent les préférences de *A*. Modéliser les préférences d'un agent est important pour pouvoir planifier ses propres coups de conversation, que ce soit dans un cadre coopératif ou stratégique. C'est également important pour de nombreuses applications pratiques qui utilisent des interfaces en langage naturel (le dialogue des systèmes de gestion, etc.). Nous allons modéliser de telles préférences en utilisant des CP-nets, qui permettent d'exploiter les dépendances existant entre les variables au sein d'un dialogue de façon compacte et intuitive. Puis, nous allons élaborer des règles permettant d'utiliser ces coups de dialogue pour créer et transformer ces CP-nets.
Nous illustrons notre approche sur un dialogue similaire à ceux trouvés dans le corpus Verbmobil (Wahlster (2000)) :

(1)
π_1 *A* : Shall we meet sometime in the next week ?
π_2 *A* : What days are good for you ?
π_3 *B* : Well, I have some free time on almost every day except Fridays.
π_4 *B* : Fridays are bad.
π_5 *B* : In fact, I'm busy on Thursday too.
π_6 *A* : Well next week I am out of town Tuesday, Wednesday and Thursday.
π_7 *A* : So perhaps Monday ?

La question posée par A en π_1 lui permet d'exprimer sa préférence pour une rencontre la semaine prochaine. Il pose ensuite en π_2 une question élaborative dont la réponse lui permettra de savoir comment satisfaire au mieux ses préférences. B répond à π_2 en π_3, et explique sa réponse en π_4. En répondant à la question élaborative de A, B montre qu'il souhaite aussi rencontrer A la semaine prochaine. B précise quelles sont ses préférences pour ce rendez-vous – il préfère du lundi au jeudi. En π_5, B corrige les préférences sur lesquelles il s'est engagé, en éliminant le jeudi. A doit donc modifier son modèle des préférences de B. A précise ses préférences en π_6, et propose ensuite une date en π_7.

2 Dialogue

Une théorie du dialogue doit lier l'interprétation du discours à certains principes généraux de rationalité et de coopérativité (Grice (1975)). Les approches mentalistes considèrent les dialogues comme étant fonction des attitudes des agents, habituellement formalisées par des logiques BDI (e.g., Grosz & Sidner (1990)). Les locuteurs expriment alors leurs croyances, à partir desquelles il est possible de déduire leurs intentions et leurs désirs. Comme soulevé par Hamblin (1987), Asher & Lascarides (2003) soutiennent que les locuteurs peuvent mentir, ou ne pas connaitre leurs croyances ou préférences. Lorsqu'un locuteur donne une information sur un état du monde ou sur ses préférences, il s'engage à ce que cette information, ou ces préférences, soient vraies. Nous nous intéressons ici à ces engagements, que nous traitons dans la suite comme étant de simples préférences.

Pour cela, nous partons d'une théorie de la structure et de l'interprétation du discours, la *Segmented Discourse Representation Theory*, ou encore SDRT (Asher (1993); Asher & Lascarides (2003)). Comme de nombreuses autres théories sur l'interprétation du discours (Hobbs *et al.* (1993); Mann & Thompson (1987)), la SDRT construit des structures discursives récursivement, en partant des unités de discours élémentaires (UDE) qui sont liés par des *relations rhétoriques*, comme par exemple *Narration*, *Explication*, *Q-Elab*, etc. La définition formelle d'une SDRS (*Segmented Discourse Representation Structure*) se base sur la description des UDE, sur des prédicats permettant de décrire les relations rhétoriques, ainsi que sur un ensembles d'étiquettes π_i qui correspondent aux éléments des SDRSs.

– Soit ϕ une formule du langage des UDE. ϕ est une formule SDRS.
– Soit $\pi_1, \ldots, \pi_n$ des étiquettes et soit R une relation rhétorique n-aire. $R(\pi_1, \ldots, \pi_n)$ est une formule SDRS.
– Soit ϕ, ϕ' deux formules SDRS. $(\phi \wedge \phi')$, $\neg\phi$ sont des formules SDRS.

Une SDRS est un triplet $\langle \Pi, \mathcal{F}, \text{Last} \rangle$, où Π est un ensemble d'étiquettes ; Last est une étiquette de Π (intuitivement, Last représente la dernière clause du discours) ; et $\mathcal{F} : \Pi \longrightarrow$ formules SDRS.

Une SDRT modélise un dialogue comme étant une suite de paires de SDRSs, chaque paire constituant un tour dans le dialogue (c'est-à-dire un échange minimal entre les participants du dialogue). On appelle une telle structure une DSDRS (*Dialogue Segmented Discourse Representation Structure*). Chaque SDRS dans une DSDRS représente les engagements publics d'un agent. Chaque agent construit une SDRS pour représenter ses propres engagements, et une seconde pour ceux des autres agents.

Tour	SDRS de A	SDRS de B
1	$\pi_{1A} : Q\text{-}Elab(\pi_1, \pi_2)$	$\emptyset$
2	$\pi_{1A} : Q\text{-}Elab(\pi_1, \pi_2)$	$\pi_{2B} :$ $Q\text{-}Elab(\pi_1, \pi_2) \wedge QAP(\pi_2, \pi)$ $\pi : Correction(\pi', \pi_5)$ $\pi' : Explication(\pi_3, \pi_4)$
3	$\pi_{3A} :$ $Q\text{-}Elab(\pi_1, \pi_2) \wedge QAP(\pi_2, \pi) \wedge$ $Elab(\pi_1, \pi_6) \wedge Elab(\pi_1, \pi_7) \wedge$ $Résultat(\pi_6, \pi_7)$	$\pi_{2B} :$ $Q\text{-}Elab(\pi_1, \pi_2) \wedge QAP(\pi_2, \pi)$ $\pi : Correction(\pi', \pi_5)$ $\pi' : Explication(\pi_3, \pi_4)$

TAB. 1 – La DSDRS du dialogue (1).

Dans la suite, nous nous sommes limités à l'étude d'un sous-ensemble de relations dans les SDRT : *QAP*, *Correction*, *Q-Elab*, *Elab*, *Explication intentionnelle*, *Commit*, et *Alternatives*. Par exemple, la DSDRS du dialogue (1) est représentée Table 1[1]. $\pi :_S \phi$, ou plus simplement $\pi : \phi$ lorsqu'il n'y a pas d'ambiguïté sur la SDRS, signifie que $\mathcal{F}_S$ assigne la formule ϕ à l'étiquette π dans S.

La représentation du dialogue (1) montre qu'à la fin du premier tour, A s'est engagé à s'intéresser aux réponses données à π_1 et π_2. Nous interprétons π_1 comme étant un engagement de la part de A à ce qu'il préfère une rencontre cette semaine. La sémantique de la relation *Q-Elab*, selon laquelle toute réponse à la question doit préciser π_1 (Asher & Lascarides (2003)), permet d'interpréter la question π_2 par "Quels jours vous conviennent la semaine prochaine ?". *Q-Elab* ne modifie pas les préférences de A. Au second tour, B répond à π_2 avec la relation *QAP*. Lascarides & Asher (2008) montre que B s'est ainsi engagé sur la contribution illocutoire de π_2 : B s'engage à vouloir une rencontre au cours de la semaine suivante. Il précise son engagement en répondant à π_2, et explique sa réponse : il préfère une rencontre tous les jours sauf le vendredi. Il corrige ensuite cette réponse avec π_5, en précisant qu'il n'est finalement pas disponible le jeudi. Le troisième tour permet à A de proposer une date à partir des réponses de B. La relation *Elab* l'engage publiquement à résoudre la question π_2 (grâce à la sémantique de *QAP*). *Elab* lui permet de préciser ses préférences : il n'est pas disponible du mardi au jeudi. Il suggère ensuite en π_7 une date respectant toutes les contraintes.

3 CP-nets

Nous allons à présent présenter un langage de représentation compacte des préférences : les CP-nets. Ce langage graphique, introduit dans Boutilier *et al.* (2004), utilise l'indépendance préférentielle conditionnelle pour structurer les préférences d'un agent sous l'hypothèse *Ceteris Paribus*.

Nous considérons ici des CP-nets "propositionnels" pour lesquels les variables sont binaires. Nous allons également introduire une relation d'indifférence dans ces CP-nets, qui autorisera un agent à être indifférent entre deux instanciations d'une variable.

[1] Lascarides & Asher (2008) explique comment construire cette DSDRS, ce que nous ne détaillons pas ici par manque de place.

Définition 1
Soit V un ensemble de variables propositionnelles, et $\{X,Y,Z\}$ une partition de V. X est **conditionnellement préférentiellement indépendant** *de Y selon Z ssi $\forall z \in 2^Z$, $\forall x_1,x_2 \in 2^X$ et $\forall y_1,y_2 \in 2^Y$ on a : $x_1y_1z \succeq x_2y_1z$ ssi $x_1y_2z \succeq x_2y_2z$.*

Pour une valeur de Z fixée, la relation de préférence sur les instanciations de X est la même quelles que soient les valeurs des instanciations de Y.
Pour chaque variable X, l'agent spécifie un ensemble de *variables parents* $Pa(X)$ qui influent sur ses préférences entre les différentes valeurs de X. Formellement, X est conditionnellement préférentiellement indépendant de $V \setminus (\{X\} \cup Pa(X))$ selon $Pa(X)$.
Le CP-net est crée à partir de tables de préférences conditionnelles, qui décrivent les préférences de chaque agent sur les valeurs de la variable X, étant données toutes les combinaisons des valeurs des variables parents.

Définition 2
$\mathcal{N} = \langle \mathcal{G}, \mathcal{T} \rangle$ est un **CP-net** *sur V, $\mathcal{G}$ étant un graphe orienté sur V, et $\mathcal{T}$ étant un ensemble de tables de préférences conditionnelles $CPT(X_i)$ pour chaque $X_i \in V$. Chaque $CPT(X_i)$ est associée à un préordre total $\succ^i_p$ sur les valeurs de X_i, selon chaque instanciation $p \in 2^{Pa(X_i)}$.*

Informellement, un CP-net $\mathcal{N}$ est satisfait par $\succ$ si $\succ$ satisfait chacune des préférences conditionnelles exprimées dans les CPTs de $\mathcal{N}$ sous l'interprétation *ceteris paribus*.
Chercher l'unique résultat optimal d'un CP-net $\mathcal{N}$ acyclique, consiste intuitivement à parcourir le graphe de haut en bas (c'est-à-dire des parents aux descendants), en instanciant chaque variable à sa valeur préférée selon l'instanciation de ses parents (*forward sweep procedure*, Boutilier *et al.* (2004)).
Si les CP-nets sont extrêmement intuitifs, ils ne permettent pas de représenter toutes les préférences : il est impossible de représenter des relations de préférences qui comparent 2 à 2 des états n'étant pas identiques "toutes choses étant égales par ailleurs". Pourtant, le pouvoir d'expression de ce langage est suffisant pour les exemples que nous avons étudiés du corpus Verbmobil. La notion de dialogue implique une certaine dépendance entre les variables, qui s'apparente à celle exploitée dans les CP-nets.

4 Du discours aux préférences

Pour raisonner sur les engagements des agents sur leurs préférences, nous devons être capables d'extraire ces préférences à partir de ce que les agents disent, et de ce que leurs actes de langages impliquent. Cette extraction se fait en deux étapes : il faut tout d'abord extraire les préférences à partir des UDEs, puis ensuite appliquer des règles qui modifient ces préférences en fonction de la structure discursive. Les SDRTs utilisent une logique de description, la *glue logic* (GL) pour trouver les relations discursives à partir des unités du discours (Asher & Lascarides (2003)). Les formules de GL décrivent les proprietés des étiquettes, et nous permettent de détecter les UDEs qui dérivent de clauses de la forme *J'aimerais* ϕ, *Nous voulons* ϕ, etc. Pour spécifier de telles préférences, nous définissons une fonction de transfert F qui transforme des étiquettes en formules de la logique propositionnelle. Si une UDE π d'un agent A est de l'une de ces formes, F

retournera une formule propositionnelle atomique X_π, qui représentera une préférence pour ϕ. Cette variable permettra d'introduire les préférences suivantes dans le CP-net de A : $X_\pi \succ_A \overline{X}_\pi$.[2]
Il existe différents autre moyens de traduire les préférences. Il est par exemple possible d'utiliser les verbes qui expriment une opinion ou un sentiment. L'utilisation de question orientées comme[3] *Ne devrions nous pas rentrer maintenant ?* exprime la préférence du locuteur pour rentrer avec ses auditeurs. De même, les formules de politesse peuvent souvent être des moyens détournés pour exprimer des préférences.
Les UDEs peuvent également spécifier des combinaisons booléennes de préférences. La négation d'une préférence pour ϕ est immédiate. Nous considérerons les préférences disjonctives comme étant non exclusives. Dans le cas contraire, nous introduirons des contraintes permettant d'exclure tout état satisfiant les deux préférences. Nous pensons que les conjonctions sont ambiguës lorsqu'il s'agit de préférences, mais qu'il est possible de résoudre cette ambiguïté dans certains cas. Une des interprétations possibles pour *X et Y* est que l'état préféré de l'agent est celui qui satisfait X et Y, mais que cet agent préfère avoir l'une de ces deux variables satisfaite plutôt qu'aucune. On représente ce type de conjonction avec le symbole &. Un agent peut également préférer la "fusion" de X et Y, et ne pas vouloir X ou Y séparément. Nous devons donc spécifier deux formes logiques distinctes pour les UDEs du type *Je veux* ϕ *et* ψ : soit *Je veux* ϕ & *je veux* ψ, ou *Je veux* $\phi \wedge$ *je veux* ψ. Cette dernière forme logique nous amènera à introduire une co-dépendance entre les variables de la conjonction. Un dernier cas à étudier est celui des questions. Si toutes les questions n'engagent pas les préférences de leur auteur, la majorité le font. En effet, si *A* demande *peut-on se rencontrer la semaine prochaine ?*, il exprime ses préférences pour une rencontre. Nous pourrions décrire cette question avec $?(\textit{rencontre moi la semaine prochaine}, \pi)$ dans une SRDT. Cette implication est encore plus forte lorsque les questions sont négatives ou ouvertes.
Les axiomes de GL sont les suivants :

1. $F(\pi) = X_\pi$ pour π atomique
2. $\text{Not}(\pi_1, \pi) \rightarrow F(\pi) = \neg F(\pi_1)$
3. $\text{Or}(\pi_1, \pi_2, \pi) \rightarrow F(\pi) = F(\pi_1) \vee F(\pi_2)$
4. $\&(\pi_1, \pi_2, \pi) \rightarrow F(\pi) = F(\pi_1) \& F(\pi_2)$
5. $\wedge(\pi_1, \pi_2, \pi) \rightarrow F(\pi) = F(\pi_1) \wedge F(\pi_2)$
6. $?(\pi_1, \pi) \rightarrow F(\pi) = F(\pi_1)$
7. $?(\neg\pi_1, \pi) \rightarrow F(\pi) = F(\pi_1)$

5 Les règles

Nous pouvons à présent décrire les règles qui nous permettent de modifier les préférences des agents à partir d'une structure discursive. Nous étudions les relations discursives suivantes, qui sont prépondérantes dans le corpus Verbmobil : *QAP* (paire

[2] GL nous informe également si de telles UDE contiennent une négation qui modifie les préférences.
[3] Pour une étude des questions orientées et de leur implications, voir Reese (in preparation).

de question-réponse), *Correction*, *Q-Elab* (question élaborative), *Explication intentionnelle* (explication permettant de comprendre pour quelles raisons un agent veut quelque chose), *Commit* (engagement à des préférences), *Alternatives* (questions alternatives) et *Elab* (élaboration).

Chacune de ces règles modifie les tables de préférences conditionnelles d'un ou plusieurs agents. Le CP-net associé représente alors l'engagement de l'agent sur ces préférences. Dans la suite, X dénote un littéral, et $Var(X)$ la variable issue de ce littéral.

Commit : Commit(π) représente un engagement de préférences sur les variables de π.

1. $F(\pi) = X$. L'agent préfère que le littéral X soit satisfait. Nous avons $X \succ \overline{X}$.
2. $F(\pi) = X \wedge Y$. L'agent préfère que les littéraux X et Y soient tous deux satisfaits, mais est indifférent s'il ne peut avoir les deux. Nous avons :
   - $Var(X) \in Pa(Var(Y))$ et $X : Y \succ \overline{Y}, \overline{X} : Y \sim \overline{Y}$.
   - $Var(Y) \in Pa(Var(X))$ et $Y : X \succ \overline{X}, \overline{Y} : X \sim \overline{X}$.
3. $F(\pi) = X\&Y$. L'agent préfère avoir les deux littéraux X et Y satisfaits mais est content si au moins l'un des deux l'est. Nous avons : $Y \succ \overline{Y}$ et $X \succ \overline{X}$.
4. $F(\pi) = X \vee Y$. L'agent préfère avoir au moins l'un des deux littéraux X et Y satisfaits. Nous avons :
   - $Var(X) \in Pa(Var(Y))$ et $X : Y \sim \overline{Y}, \overline{X} : Y \succ \overline{Y}$.
   - $Var(Y) \in Pa(Var(X))$ et $Y : X \sim \overline{X}, \overline{Y} : X \succ \overline{X}$.
5. $F(\pi) = \Phi$. Nous pouvons appliquer les règles 1-4 en décomposant Φ.

Par manque de place, nous ne décrirons pas dans la suite les règles pour les cas $\vee$, qui n'apparaissent pas dans notre exemple illustratif.

Notre second ensemble de règles traite les *Explications intentionnelles* ou *Iexplications*.

(2)
π_1 Je veux aller au supermarché
π_2 pour manger quelque chose

Dans cet exemple, l'agent veut manger quelque chose, et c'est pourquoi il veut aller au supermarché. π_1 dépend donc causalement de π_2.

Explication intentionnelle : Dans Iexplication(π_i, π_j), π_j sont les préférences permettant d'expliquer les préférences exprimées dans π_i.

6. $F(\pi_i) = X$, $F(\pi_j) = Y$. Cela signifie que l'agent explique ses préférences sur X par Y. On sait donc que Y est préférée à $\neg Y$, et que si aucune préférence sur X n'est déjà définie, Y est une raison de vouloir X. Dans le cas contraire, l'agent préfère X si Y est vrai, mais ne modifie pas ses préférences sinon.
   - $Y \succ \overline{Y}$, $Var(Y) \in Pa(Var(X))$,
   - si $Pa(Var(X)) = \{Var(Y)\}$ alors $Y : X \succ \overline{X}, \overline{Y} : \overline{X} \sim X$. Sinon, si $\succ_X$ représente la relation de préférences associée à $CPT(X)$, nous avons $\succ_{X,Y} = X \succ \overline{X}$, $\succ_{X,\overline{Y}} = \succ_X$.[4]
7. $F(\pi_i) = X \wedge Z$ et $F(\pi_j) = Y$. L'agent explique ses préférences sur $X \wedge Z$ par Y : il veut satisfaire X et Z si Y est satisfait, mais ne veut pas avoir X ou Z séparément.
   - $Y \succ \overline{Y}$, $Var(Y), Var(Z) \in Pa(Var(X))$, $Var(Y), Var(X) \in Pa(Var(Z))$
   - $\succ_X$ dénote la relation de préférences associée à $CPT(X)$. Si $\succ_X$ n'est pas encore définie, nous avons : $Y \wedge Z : X \succ \overline{X}, \overline{Y} \vee \overline{Z} : \overline{X} \succ X$. Sinon, $\succ_{X,Y,Z} = X \succ \overline{X}$, $\succ_{X,Y,\overline{Z}} = \overline{X} \succ X$, $\succ_{X,\overline{Y},Z} = \succ_{X,\overline{Y},\overline{Z}} = \succ_X$,

[4] Si nous avons $\succ_X$ tel que $Z : \overline{X} \succ X$, $\overline{Z} : X \succ \overline{X}$, $\succ_{X,Y}$ représente les préférences définies par $Z \wedge Y$ et $\overline{Z} \wedge Y$, tandis que $\succ_{X,\overline{Y}}$ représente les préférences définies par $Z \wedge \overline{Y}$ et $\overline{Z} \wedge \overline{Y}$.

- $CPT(Z)$ est défini de la même façon que $CPT(X)$ en inversant X et Z.

8. $F(\pi_i) = X \wedge Z$ et $F(\pi_j) = Y \wedge W$.
   - $Y \succ \overline{Y}, W \succ \overline{W}$,
   - $Var(Y), Var(W), Var(Z) \in Pa(Var(X)), Var(Y), Var(W), Var(X) \in Pa(Var(Z))$
   - Si $\succ_X$ représente la relation de préférences associée à $CPT(X)$ (déjà définie ou pas), alors : $\succ_{X,Y,W,Z} = X \succ \overline{X}$, $\succ_{X,Y,W,\overline{Z}} = \succ_{X,Y,\overline{W},Z} = \succ_{X,Y,\overline{W},\overline{Z}} = \succ_{X,\overline{Y},W,Z} = \succ_{X,\overline{Y},W,\overline{Z}} = \succ_{X,\overline{Y},\overline{W},Z} = \succ_{X,\overline{Y},\overline{W},\overline{Z}} = \overline{X} \succ X$,
   - $CPT(Z)$ est défini de la même façon que $CPT(X)$ en inversant X et Z.
9. $F(\pi_i) = X \wedge Z$ et $F(\pi_j) = Y \& W$.
   - $Y \succ \overline{Y}, W \succ \overline{W}$,
   - $Var(Y), Var(W), Var(Z) \in Pa(Var(X)), Var(Y), Var(W), Var(X) \in Pa(Var(Z))$,
   - $\succ_X$ dénote la relation de préférences associée à $CPT(X)$. Si $\succ_X$ n'est pas déjà définie, nous avons $(Y \vee W) \wedge Z : X \succ \overline{X}$, $(\overline{Y} \wedge \overline{W}) \vee \overline{Z} : \overline{X} \succ X$. Sinon, $\succ_{X,(Y \vee W) \wedge Z} = X \succ \overline{X}$, $\succ_{X,(Y \vee W) \wedge \overline{Z}} = \overline{X} \succ X$, $\succ_{X,(\overline{Y} \wedge \overline{W}),Z} = \succ_X$.
   - $CPT(Z)$ est défini de la même façon que $CPT(X)$ en inversant X et Z.
10. $F(\pi_i) = X \& Z$ et $F(\pi_j) = \Phi$. L'agent explique ses préférences sur $X \& Z$ par Φ. Donc Φ peut expliquer X et peut expliquer Z. Nous appliquons 2 fois la règle 6.
11. $F(\pi_i) = X$ et $F(\pi_j) = Y \wedge Z$. L'agent explique ses préférences sur X par $Y \wedge Z$: Y et Z doivent être satisfaits pour expliquer X.
    - $Y \succ \overline{Y}, Z \succ \overline{Z}, Var(Y), Var(Z) \in Pa(Var(X))$,
    - Si $Pa(Var(X)) = \{Var(Y), Var(Z)\}$ alors $Y \wedge Z : X \succ \overline{X}$, $\overline{Y} \vee \overline{Z} : \overline{X} \succ X$. Sinon, si $\succ_X$ dénote la relation de préférences associée à $CPT(X)$, nous avons $\succ_{X,(Y \wedge Z)} = X \succ \overline{X}$, $\succ_{X,(\overline{Y} \vee \overline{Z})} = \succ_X$.
12. $F(\pi_i) = X$ et $F(\pi_j) = Y \& Z$. Y ou Z doivent être satisfaits pour expliquer X.
    - $Y \succ \overline{Y}, Z \succ \overline{Z}, Var(Y), Var(Z)) \in Pa(Var(X))$,
    - Si $Pa(Var(X)) = \{Var(Y), Var(Z)\}$ alors $Y \& Z : X \succ \overline{X}$, $\overline{Y} \wedge \overline{Z} : \overline{X} \succ X$. Sinon, si $\succ_X$ dénote la relation de préférences associée à $CPT(X)$, nous avons $\succ_{X,(Y \vee Z)} = X \succ \overline{X}$, $\succ_{X,(\overline{Y} \wedge \overline{Z})} = \succ_X$.
13. $F(\pi_i) = \Phi$ et $F(\pi_j) = \Psi$. Nous pouvons appliquer les règles 6-12 en décomposant Φ et Ψ.

La dépendance causale en oeuvre dans Iexplication est très proche de la dépendance logique que l'on trouve dans l'exemple suivant :

(3) π_1 J'aimerais boire du vin
π_2 J'aimerais boire du vin blanc

Les préférences de l'agent pour du vin blanc dépendent de ses préférences pour du vin. Cette relation discursive entre π_1 et π_2 est appelée *Elaboration*. Ses effets sur les préférences sont similaires à ceux d'Iexplication.

Elab Elab(π_i, π_j) $\equiv$ Iexplication(π_j, π_i) en ce qui concerne les modifications de préférences.

On en arrive à présent aux questions. La façon dont ces dernières affectent les préférences dépend des réponses qui leur sont données dans le discours. Etudions tout d'abord les questions fermées.

Q-Elab-ON, QAP QElab-ON$_A$(π_i, π_j), QAP$_B$(π_j, π_k) représente la réponse de B (de la forme "oui" ou "non") à une question élaborative posée par A.

Dans ce cas, nous savons que A s'engage sur $F(\pi_i)$: $\text{Commit}_A(\pi_i)$.
Ensuite, nous devons modifier les préférences de B en fonction de sa réponse :

Oui $\text{QElab-ON}_A(\pi_i, \pi_j)$, $\text{QAP}_B(\pi_j$, oui). Les préférences de B sur π_i et π_j sont définies en appliquant les règles : $\text{Commit}_B(\pi_i)$; $\text{Elab}_B(\pi_i, \pi_j)$.

Non $\text{QElab-ON}_A(\pi_i, \pi_j)$, $\text{QAP}_B(\pi_j$, non). Les préférences de B sur π_i et π_j sont définies en appliquant les règles [5] : $\text{Commit}_B(\pi_i)$; $\text{Elab}_B(\pi_i, \pi_j^*)$, avec $\pi_j^* = \neg(Core(?(\pi_j, \pi)))$.

Etudions à présent les questions ouvertes.
Q-Elab-Wh, QAP : $\text{QElab-Wh}_A(\pi_i, \pi_j)$, $\text{QAP}_B(\pi_j, \pi_k)$ représente la réponse de B à une question élaborative ouverte posée par A. La sémantique de Q-Elab impose que la question ouverte porte sur les variables définies dans π_i : on sait que π_j ne contient aucune variable.
Comme précédemment, A s'engage sur $F(\pi_i)$: $\text{Commit}_A(\pi_i)$.
Les préférences de B sur les variables de π_i et π_k sont identiques à celles définies pour une question Q-Elab-ON à laquelle on aurait répondu oui : les variables dans π_k raffinent les préférences sur les variables de π_i. Les préférences de B sont donc définies en appliquant les règles : $\text{Commit}_B(\pi_i)$; $\text{Elab}_B(\pi_i, \pi_j)$.
La dernière, et plus complexe, catégorie de questions est celle des *alternatives*. Ces questions sont de la forme *préfèreriez vous un restaurant traditionnel ou une pizzeria ?*.
Alternatives L'agent A pose une question contenant n alternatives à un agent B. La réponse de B nous donne des informations sur ses préférences. Soit π la question alternative posée par A, et π^* la réponse de B à la question π, avec $\text{QAP}_B(\pi, \pi^*)$ et $\&(X_i, \ldots, X_n, \pi^*)$. Cette description de la réponse de B prévoit plusieurs réponses toutes les unes aussi bonnes que les autres : pour tout $i \leq n$, B veut satisfaire le littéral X_i. On ajoute donc les préférences suivantes pour chacun des X_i, ou on modifie les préférences existantes le cas échéant, avec $Pa(Var(X_i)) = \emptyset$, et $X_i \succ \overline{X}_i$.
Correction : Les études linguistiques faites sur la Correction (Asher & Lascarides (2003)) ont montré que les révisions impliquées dans une relation discursive sont très restrictives. Leur sémantique nécessite essentiellement une opération "cherche et remplace", l'élément devant être remplacé étant signalé par une marque lexicale.
Au niveau des préférences, $\text{Correction}(\pi_i, \pi_j)$ signifie donc que certains littéraux de π_i sont remplacés par des littéraux de π_j. Nous avons un ensemble de règles de la forme $X \leftarrow (Y_1, \ldots, Y_m)$, signifiant que le littéral X de π_i est remplacé par l'ensemble de littéraux $(Y_1, \ldots, Y_m)$ de π_j. Nous supposons que X ne peut pas dépendre de $(Y_1, \ldots, Y_m)$ avant la correction.

14. Si $Pa(Var(X)) = \emptyset$, on ajoute pour tous $k \in \{1, \ldots, m\}$, $Y_k \succ \overline{Y}_k$. Sinon, si $\succ_X$ dénote la relation de préférences associée à $CPT(X)$, nous avons pour tous $k \in \{1, \ldots, m\}$, $\succ_{Y_k} = \succ_X$.
15. Pour tout W tel que $Var(X) \in Pa(Var(W))$, si $\succ_W$ dénote la relation de préférences associée à $CPT(W)$, nous définissons $\succ'_W$ comme : $\succ'_{\{W \setminus \{X, \overline{X}\}, \bigwedge_{1 \leq i \leq m} Y_i}} = \succ_{\{W \setminus \{X, \overline{X}\}, X}$, et $\succ'_{\{W \setminus \{X, \overline{X}\}, \bigvee_{1 \leq i \leq m} \overline{Y}_i}} = \succ_{\{W \setminus \{l, \overline{X}\}, \overline{X}}$.

Nous supprimons ensuite X du CP-net.

[5] $Core(?(\pi_1, \pi)) = \pi_1$, $Core(?(\neg\pi_1, \pi)) = \pi_1$.

6 Traitement de notre exemple

Appliquons à présent ces règles au dialogue (1).

π_1 *A* : Shall we meet sometime in the next week ?

Seule la règle Commit s'applique sur π_1. Nous avons $\text{Commit}_A(\pi_1)$, $F(\pi_1) = M$, où *M* signifie Meet. On obtient :

$$CPT_A(M) = M \succ \overline{M}$$

π_2 *A* : What days are good for you ?

Q-Elab-(π_1, π_2). *A* continue à s'engager pour *M* en π_2.

π_3 *B* : Well, I have some free time on almost every day except Fridays.

π_4 *B* : Fridays are bad.

π_4 explique π_3, mais n'a aucun impact sur les préférences de *B*. Nous avons Q-Elab-$\text{Wh}_A(\pi_1, \pi_2)$, $\text{QAP}_B(\pi_2, \pi_3)$, avec $\&(J_1, J_2, J_3, J_4, \neg J_5, \pi_3)$ où J_1 signifie Monday, J_2 Tuesday, J_3 Wednesday, J_4 Thursday et J_5 Friday. Nous avons $F(\pi_3) = F(J_1)\&F(J_2)\&F(J_3)\&F(J_4)\&\neg F(J_5) = J_1\&J_2\&J_3\&J_4\&\neg J_5$.

On applique $\text{Commit}_B(\pi_1)$, $\text{Elab}_B(\pi_1, \pi_3)$, *i.e.* $\text{Iexplication}_B(\pi_3, \pi_1)$. On obtient :

$$CPT_B(M) = M \succ \overline{M} \qquad CPT_B(J_1) = \frac{M : J_1 \succ \overline{J}_1}{\overline{M} : \overline{J}_1 \sim J_1} \qquad CPT_B(J_2) = \frac{M : J_2 \succ \overline{J}_2}{\overline{M} : \overline{J}_2 \sim J_2}$$
$$CPT_B(J_3) = \frac{M : J_3 \succ \overline{J}_3}{\overline{M} : \overline{J}_3 \sim J_3} \qquad CPT_B(J_4) = \frac{M : J_4 \succ \overline{J}_4}{\overline{M} : \overline{J}_4 \sim J_4} \qquad CPT_B(J_5) = \frac{M : \overline{J}_5 \succ J_5}{\overline{M} : J_5 \sim \overline{J}_5}$$

π_5 *B* : In fact, I'm busy on Thursday too.

On a $\text{Correction}_B(\pi_3, \pi_5)$, avec $F(\pi_5) = \neg J_4$, et donc $J_4 \leftarrow \neg J_4$. On obtient :

$$CPT_B(M) = M \succ \overline{M} \qquad CPT_B(J_1) = \frac{M : J_1 \succ \overline{J}_1}{\overline{M} : \overline{J}_1 \sim J_1} \qquad CPT_B(J_2) = \frac{M : J_2 \succ \overline{J}_2}{\overline{M} : \overline{J}_2 \sim J_2}$$
$$CPT_B(J_3) = \frac{M : J_3 \succ \overline{J}_3}{\overline{M} : \overline{J}_3 \sim J_3} \qquad CPT_B(J_4) = \frac{M : \overline{J}_4 \succ J_4}{\overline{M} : J_4 \sim \overline{J}_4} \qquad CPT_B(J_5) = \frac{M : \overline{J}_5 \succ J_5}{\overline{M} : J_5 \sim \overline{J}_5}$$

π_6 *A* : Well next week I am out of town Tuesday, Wednesday and Thursday.

On a $\text{Elab}_A(\pi_1, \pi_6)$, et donc $\&(\neg J_2, \neg J_3, \neg J_4, \pi_6)$. $F(\pi_6) = \neg J_2\&\neg J_3\&\neg J_4$. On applique $\text{Iexplication}_A(\pi_6, \pi_1)$, et on obtient :

$$CPT_A(M) = M \succ \overline{M} \qquad CPT_A(J_2) = \frac{M : \overline{J}_2 \succ J_2}{\overline{M} : J_2 \sim \overline{J}_2}$$
$$CPT_A(J_3) = \frac{M : \overline{J}_3 \succ J_3}{\overline{M} : J_3 \sim \overline{J}_3} \qquad CPT_A(J_4) = \frac{M : \overline{J}_4 \succ J_4}{\overline{M} : J_4 \sim \overline{J}_4}$$

π_7 *A* : So perhaps Monday ?

$\text{Elab}_A(\pi_1, \pi_7)$, et $\text{Résultat}_A(\pi_6, \pi_7)$ qui n'affecte pas les préférences. On a $F(\pi_7) = J_1$, c'est-à-dire $\text{Iexplication}_A(\pi_7, \pi_1)$. On obtient :

$$CPT_A(M) = M \succ \overline{M} \qquad CPT_A(J_1) = \frac{M : J_1 \succ \overline{J}_1}{\overline{M} : \overline{J}_1 \sim J_1} \qquad CPT_A(J_2) = \frac{M : \overline{J}_2 \succ J_2}{\overline{M} : J_2 \sim \overline{J}_2}$$
$$CPT_A(J_3) = \frac{M : \overline{J}_3 \succ J_3}{\overline{M} : J_3 \sim \overline{J}_3} \qquad CPT_A(J_4) = \frac{M : \overline{J}_4 \succ J_4}{\overline{M} : J_4 \sim \overline{J}_4}$$

A ce point du dialogue, on peut voir que les issues préférées de *B* sont MJ_1, MJ_2, MJ_3, $M\neg J_4$, $M\neg J_5$; tandis que celles de *A* sont MJ_1, $M\neg J_2$, $M\neg J_3$ et $M\neg J_4$. Une date leur convient à tous deux pour un rendez-vous : lundi.

Cet exemple est assez représentatif des dialogues du corpus Vermobil. Nous avons également analysé quelques dialogues de tourisme. Toutes nos règles ne sont pas appliquées dans cet exemple, mais toutes sont utiles dans d'autres dialogues du corpus Verbmobil ou du tourisme. Nous espérons donner plus d'illustration dans une version plus complète de ce papier, et automatiser l'application de ces règles sur des corpus annotés.

7 Conclusion

Modéliser les préférences exprimées dans des textes est important pour beaucoup d'applications en traitement automatique du langage. Nous avons montré comment utiliser les CP-nets et les modèles des structure discursives pour accomplir cette tâche formellement. Nos règles permettant de modéliser les préférences sont simples et intuitives. Il est en suite possible d'utiliser des techniques de la théorie des jeux pour raisonner sur ces préférences. Bien sûr, tout ceci dépend de l'extraction des structures discursives à partir d'un texte, ce qui est une tâche difficile. Néanmoins, Baldridge & Lascarides (2005) ont montré comment extraire automatiquement cette structure discursive à partir de dialogues du corpus Verbmobil. Notre prochain objectif est d'automatiser l'application de nos règles sur les dialogues de ces corpus.

Références

ASHER N. (1993). *Reference to Abstract Objects in Discourse*. Kluwer Academic Publishers.

ASHER N. & LASCARIDES A. (2003). *Logics of Conversation*. Cambridge University Press.

BALDRIDGE J. & LASCARIDES A. (2005). Probabilistic head-driven parsing for discourse structure. In *CoNLL'05*.

BOUTILIER C., BRAFMAN R. I., DOMSHLAK C., HOOS H. H. & POOLE D. (2004). CP-nets : A Tool for Representing and Reasoning with Conditional *Ceteris Paribus* Preference Statements. *Journal of Artificial Intelligence Research*, **21**, 135–191.

GRICE H. P. (1975). Logic and conversation. In *Sytnax and Semantics Volume 3 : Speech Acts*, p. 41–58. Academic Press.

GROSZ B. & SIDNER C. (1990). Plans for discourse. In *Intentions in Communication*, p. 365–388. MIT Press.

HAMBLIN C. (1987). *Imperatives*. Blackwells.

HOBBS J. R., STICKEL M., APPELT D. & MARTIN P. (1993). Interpretation as abduction. *Artificial Intelligence*, **63**(1–2), 69–142.

LASCARIDES A. & ASHER N. (2008). Grounding and correcting commitments in dialogue. submitted to SIGDIAL.

MANN W. C. & THOMPSON S. A. (1987). Rhetorical structure theory : A framework for the analysis of texts. *International Pragmatics Association Papers in Pragmatics*, **1**, 79–105.

REESE B. (in preparation). *Bias in Questions*. PhD thesis, University of Texas at Austin.

W. WAHLSTER, Ed. (2000). *Verbmobil : Foundations of Speech-to-Speech Translation*. Springer.

Une mise sous forme prénexe préservant les résultats intermédiaires pour les formules booléennes quantifiées

Benoit Da Mota, Igor Stéphan, Pascal Nicolas

LERIA, Université d'Angers
2, boulevard Lavoisier, 49045, Angers Cedex 01
{damota,stephan,pn}@info.univ-angers.fr

Résumé : La plupart des procédures pour résoudre le problème de validité des formules booléennes quantifiées prennent en entrée seulement des formules sous forme normale conjonctive. Mais, il est rarement naturel d'exprimer un problème directement sous cette forme et il est plus courant d'utiliser des variables existentielles pour représenter des résultats intermédiaires. Or, lors de la mise sous forme prénexe, l'équivalence est exprimée en fonction d'autres opérateurs logiques et les variables intermédiaires sont multipliées. Dans ce travail, nous mettons en évidence des équivalences logiques qui permettent aux résultats intermédiaires de traverser les équivalences. Les résultats expérimentaux montrent qu'utiliser ces équivalences logiques avant de transformer la formule sous forme prénexe améliore le temps de résolution par les différentes procédures.

Mots-clés : Formules booléennes quantifiées, validité, forme normale conjonctive, vérification formelle de circuits logiques.

1 Introduction

Le problème de validité pour les formules booléennes quantifiées (QBF) est une généralisation du problème de satisfiabilité pour les formules booléennes. Tandis que décider de la satisfiabilité des formules booléennes est NP-complet, décider de la validité des QBF est PSPACE-complet. C'est le prix à payer pour une représentation plus concise pour de très nombreuses classes de formules. Une multitude d'importants problèmes de décision parmi des champs très divers ont des transformations polynomiales vers le problème de validité des QBF. La plupart des procédures actuelles pour décider des QBF nécessitent d'avoir en entrée une formule sous forme normale conjonctive (Giunchiglia *et al.*, 2004; Biere, 2004; Benedetti, 2005). Or, les problèmes sont rarement exprimés directement sous cette forme. Il est plus naturel de les représenter en utilisant la richesse du langage et donc à l'aide d'opérateurs plus expressifs (l'implication, l'équivalence et le ou exclusif) et avec des quantificateurs à l'intérieur des formules. La mise sous forme normale conjonctive nécessite que la formule soit sous forme prénexe, c'est-à-dire avec

tous les quantificateurs devant la formule. L'ensemble des transformations précédant la résolution est décrit dans la figure 1.

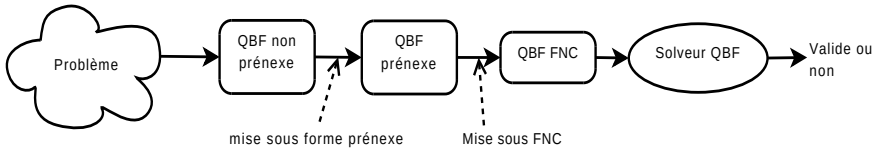


FIG. 1 – Étapes menant à la décision d'un problème via les QBF.

La mise sous forme normale conjonctive a été largement étudiée (Plaisted & Greenbaum, 1986; de la Tour, 1992; Egly, 1996), car cette forme normale est utilisée pour la satisfiabilité des formules booléennes (non quantifiées). Il n'existe pas une unique forme prénexe associée à une QBF et selon la stratégie choisie pour l'ordre d'extraction des quantificateurs, le temps de résolution peut être fortement influencé (Egly *et al.*, 2003; Giunchiglia *et al.*, 2006). Mais aucune règle n'est fournie pour extraire les quantificateurs de formules booléennes quantifiées contenant des équivalences et des ou exclusifs. Le seul choix possible est d'exprimer ces opérateurs en fonction des opérateurs pour lesquels nous connaissons des règles. Nous verrons que ce choix a des conséquences sur la taille de la formule et sur le nombre de variables. Nous nous intéressons dans la mise sous forme prénexe à l'étape qui consiste à ré-écrire une formule en une forme logiquement équivalente sans équivalence ni ou exclusif. Nous nous intéressons tout particulièrement au traitement des variables existentielles qui représentent un résultat intermédiaire et qui sont introduites pour faciliter l'écriture ou factoriser des sous-formules. Pour la partie expérimentale, nous utilisons des QBF codant la vérification formelle de circuits logiques et plus particulièrement l'additionneur n-bits, qui illustre en pratique l'intérêt de notre proposition. Les preuves des théorèmes introduits sont disponibles à l'adresse : `http://forge.info.univ-angers.fr/~damota/iaf08/preuves.pdf`.

2 Préliminaires

L'ensemble des valeurs booléennes **v** et **f** est noté $BOOL$. L'ensemble des variables (propositionnelles ou booléennes) est noté $\mathcal{V}$. Les symboles $\top$ et $\bot$ sont les constantes booléennes. Le symbole $\wedge$ est utilisé pour la conjonction, $\vee$ pour la disjonction, $\neg$ pour la négation, $\rightarrow$ pour l'implication, $\leftrightarrow$ pour l'équivalence et $\oplus$ pour le ou exclusif. L'ensemble des opérateurs $\{\wedge, \vee, \rightarrow, \leftrightarrow, \oplus\}$ est noté $\mathcal{O}$. Le symbole $\exists$ est utilisé pour la quantification existentielle et $\forall$ pour la quantification universelle (q est utilisé pour noter un quantificateur quelconque). Un littéral est une variable ou la négation de celle-ci. L'ensemble QBF des formules booléennes quantifiées est défini inductivement ainsi : tout symbole (constante ou variable) propositionnel est élément de QBF ; si F est élément de QBF alors $\neg F$ est élément de QBF ; si F et G sont éléments de QBF et $\circ$ est élément de $\mathcal{O}$ alors $(F \circ G)$ est élément de QBF ; si F est un élément de QBF et x est une variable alors $(\exists x\ F)$ et $(\forall x\ F)$ sont des éléments de QBF. Par convention, des

quantificateurs différents lient des variables différentes. L'ensemble des variables d'une formule F est noté $\mathcal{V}(F)$. Une variable x est libre si et seulement si elle n'apparait pas sous la portée d'un quantificateur $\exists x$ ou $\forall x$. L'ensemble des variables libres d'une formule F est noté $\mathcal{L}(F)$. Une substitution est une fonction de l'ensemble des variables dans l'ensemble des formules (quantifiées ou non). Nous définissons la substitution de x par F dans G, notée $G[x \leftarrow F]$, comme étant la formule obtenue de G en remplaçant toutes les occurrences de la variable x par la formule F. Un lieur est une chaîne de caractères $q_1x_1 \dots q_nx_n$ avec $x_1, \dots, x_n$ des variables distinctes et $q_1 \dots q_n$ des quantificateurs. Une QBF QF est en forme prénexe si F est une formule booléenne, appelée matrice et Q est un lieur. Elle est sous forme normale conjonctive (FNC) si F est une formule booléenne en forme normale conjonctive (i.e. une conjonction de disjonctions de littéraux).

La sémantique des symboles booléens est définie de manière habituelle. Une valuation est une fonction de l'ensemble des variables dans $BOOL$; elle satisfait une formule si appliquée à celle-ci elle vaut **v** sinon elle la falsifie. La satisfaction propositionnelle est notée $\models$ et l'équivalence logique $\equiv$. La sémantique des quantificateurs est la suivante : pour toute variable x et QBF F,

$$(\exists x\ F) = (F[x \leftarrow \top] \vee F[x \leftarrow \bot]) \quad \text{et} \quad (\forall x\ F) = (F[x \leftarrow \top] \wedge F[x \leftarrow \bot])$$

Une QBF F est valide si $F \equiv \top$.

Enfin, rappelons que le problème (SAT) consistant à décider si une formule booléenne est satisfiable ou non est le problème canonique de la classe NP-complet. De son côté, le problème (QBF) consistant à décider si une formule booléenne quantifiée est valide ou non est le problème canonique de la classe PSPACE-complet.

3 Mise sous forme prénexe et équivalences

3.1 Motivations

Dans (Egly *et al.*, 2003), les auteurs définissent des stratégies pour la mise sous forme prénexe selon l'ordre d'extraction des quantificateurs et montrent expérimentalement que cela peut avoir une forte influence sur le temps de résolution nécessaire aux différentes procédures. Cependant, il n'existe pas de règle pour extraire des quantificateurs de l'équivalence ou du ou exclusif. D'une manière générale, la mise sous forme prénexe se décompose en trois étapes :

1. suppression des équivalences et des ou exclusifs,
2. renommage des variables afin que des quantificateurs différents lient des variables différentes,
3. extraction des quantificateurs.

L'étape 1 se résume à remplacer toutes les occurrences de l'équivalence et du ou exclusif grâce aux équivalences logiques suivantes :

1) $(F \leftrightarrow G) \equiv ((G \rightarrow F) \wedge (F \rightarrow G))$ 2) $(F \oplus G) \equiv ((G \vee F) \wedge (\neg F \vee \neg G))$

Nous remarquons que les formules F et G sont recopiées deux fois. Nous rappelons les équivalences logiques utilisées dans (Egly *et al.*, 2003) qui permettent de déplacer les quantificateurs et nous complétons avec celles pour l'implication (règles 13 à 16) qui sont facilement déduites des précédentes. Soit F, G et H des QBF et x une variable

booléenne ($x \notin \mathcal{L}(H)$), alors :

3) $\exists x(\neg F) \equiv \neg(\forall x\ F)$.
4) $\forall x(\neg F) \equiv \neg(\exists x\ F)$.
5) $\forall x\ F \equiv F$, si $x \notin \mathcal{L}(F)$.
6) $\exists x\ F \equiv F$, si $x \notin \mathcal{L}(F)$.
7) $\forall x(F \wedge H) \equiv (\forall x\ F) \wedge H$,
8) $\forall x(F \vee H) \equiv (\forall x\ F) \vee H$,
9) $\exists x(F \wedge H) \equiv (\exists x\ F) \wedge H$,
10) $\exists x(F \vee H) \equiv (\exists x\ F) \vee H$,
11) $\forall x(F \wedge G) \equiv (\forall x\ F) \wedge (\forall x\ G)$.
12) $\exists x(F \vee G) \equiv (\exists x\ F) \vee (\exists x\ G)$.
13) $\forall x(F \rightarrow H) \equiv (\exists x\ F) \rightarrow H$,
14) $\exists x(F \rightarrow H) \equiv (\forall x\ F) \rightarrow H$,
15) $\forall x(H \rightarrow G) \equiv H \rightarrow (\forall x\ G)$,
16) $\exists x(H \rightarrow G) \equiv H \rightarrow (\exists x\ G)$,

L'extraction des quantificateurs est un processus qui consiste à appliquer ces équivalences de la droite vers la gauche jusqu'à obtenir une QBF sous forme prénexe. La mise sous forme prénexe conserve la validité mais ne garantit ni de garder la taille de la formule ni l'espace de recherche associé. Alors que l'équivalence représente le "si et seulement si" du langage naturel, le ou exclusif représente sa négation. Le pouvoir d'expression de l'équivalence s'étend bien au delà de la simple équivalence logique entre $(F \leftrightarrow G)$ et $((G \rightarrow F) \wedge (F \rightarrow G))$: un quantificateur présent dans F et G, de part les équivalences logiques 13 à 16, sera extrait aussi bien sous sa forme universelle que sous sa forme existentielle, et ce quelle que soit sa forme initiale. Si cela n'a aucun impact vis-à-vis de la validité, il n'en est pas de même vis-à-vis du calcul. Nous montrons dans les exemples suivants qu'à la fin des trois étapes de la mise sous forme prénexe, l'augmentation de la taille de la formule n'est pas le seul problème.

Soit a une variable booléenne, ϕ, ϕ_1 et ϕ_2 des formules booléennes quantifiées telles que $\phi = (\phi_1 \leftrightarrow \exists a(\phi_2))$ et $a \notin \mathcal{L}(\phi_1)$. Dans un premier temps il faut exprimer l'équivalence à l'aide de la conjonction et de l'implication : $\phi \stackrel{1}{\equiv} ((\phi_1 \rightarrow \exists a(\phi_2)) \wedge (\exists a(\phi_2) \rightarrow \phi_1))$. Les formules ϕ_1 et ϕ_2 ont été dupliquées. Ensuite, il faut extraire les quantificateurs des implications : $\phi \stackrel{16,13}{\equiv} (\exists\ a(\phi_1 \rightarrow \phi_2) \wedge \forall\ a(\phi_2 \rightarrow \phi_1))$. Il faut renommer une des variables a afin d'extraire les quantificateurs de la conjonction. Soit b une nouvelle variable booléenne et $\phi'_2 = \phi_2[a \leftarrow b]$ alors : $\phi \stackrel{9,7}{\equiv} \exists a\ \forall b((\phi_1 \rightarrow \phi_2) \wedge (\phi'_2 \rightarrow \phi_1))$. L'ordre d'extraction des quantificateurs aurait pu être inversé, nous aurions obtenu : $\phi \stackrel{7,9}{\equiv} \forall b\ \exists a((\phi_1 \rightarrow \phi_2) \wedge (\phi'_2 \rightarrow \phi_1))$. Dans le cas général, si F est une formule booléenne quantifiée, $\forall y\ \exists x(F) \not\equiv \exists x\ \forall y(F)$. Le fait de savoir que les quantificateurs peuvent s'inverser contrairement au cas général est une information importante qui pourrait être exploitée par les procédures de décision pour le problème QBF. Les variables a et b représentent la même variable de la formule non prénexe mais rien dans la formule prénexe ne semble les mettre en relation.

Nous envisageons le cas où un quantificateur doit traverser plusieurs équivalences. Nous montrons dans l'exemple suivant, que le choix de la position des parenthèses peut influer sur la taille de la formule mise sous forme prénexe. Soit a une variable booléenne, ϕ_d, ϕ_g, ϕ_1, ϕ_2 et ϕ_3 des QBF telles que $\phi_g = ((\phi_1 \leftrightarrow \phi_2) \leftrightarrow (\exists a\ \phi_3))$, $\phi_d = (\phi_1 \leftrightarrow (\phi_2 \leftrightarrow (\exists a\ \phi_3)))$ avec $a \notin \mathcal{L}(\phi_1)$ et $a \notin \mathcal{L}(\phi_2)$ alors $\phi_d \equiv \phi_g$ par associativité de l'équivalence logique. Nous mettons ϕ_g sous forme prénexe :

$$\begin{array}{ll} \phi_g & \stackrel{1}{\equiv} \quad (((\phi_1 \leftrightarrow \phi_2) \rightarrow (\exists a\ \phi_3)) \wedge ((\exists a\ \phi_3) \rightarrow (\phi_1 \leftrightarrow \phi_2))) \\ \phi_g & \stackrel{16,13,9,7}{\equiv} \quad \exists a \forall b\ (((\phi_1 \leftrightarrow \phi_2) \rightarrow \phi_3)\ \wedge (\phi_3[a \leftarrow b] \rightarrow (\phi_1 \leftrightarrow \phi_2))) \end{array}$$

puis ϕ_d :

$$
\begin{array}{lll}
\phi_d & \overset{1}{\equiv} & (\phi_1 \leftrightarrow ((\phi_2 \rightarrow (\exists a\ \phi_3)) \wedge ((\exists a\ \phi_3) \rightarrow \phi_2))) \\
\phi_d & \overset{1}{\equiv} & ((\phi_1 \rightarrow ((\phi_2 \rightarrow (\exists a\ \phi_3)) \wedge ((\exists a\ \phi_3) \rightarrow \phi_2))) \wedge \\
 & & (((\phi_2 \rightarrow (\exists a\ \phi_3)) \wedge ((\exists a\ \phi_3) \rightarrow \phi_2)) \rightarrow \phi_1)) \\
\phi_d & \overset{13,14,15,16,9,7}{\equiv} & \exists a \exists c \forall b \forall d((\phi_1 \rightarrow ((\phi_2 \rightarrow \phi_3) \wedge (\phi_3[a \leftarrow b] \rightarrow \phi_2))) \wedge \\
 & & (((\phi_2 \rightarrow \phi_3[a \leftarrow d]) \wedge (\phi_3[a \leftarrow c] \rightarrow \phi_2)) \rightarrow \phi_1))
\end{array}
$$

Les formules ϕ_g et ϕ_d, bien qu'équivalentes, n'ont pas du tout la même taille ni le même nombre de variables une fois mises sous forme prénexe. Maintenant, si la formule considérée est $(\phi_1 \leftrightarrow (\phi_2 \leftrightarrow ...(\phi_n \leftrightarrow (\exists a\ \phi_{n+1}))))$, alors 2^n variables seront créées dont la moitié sera quantifiée universellement. Les formules ϕ_n et ϕ_{n+1} seront recopiées 2^n fois, la formule ϕ_{n-1} sera recopiée 2^{n-1} fois, et ainsi de suite, jusqu'à ϕ_1 qui sera recopiée 2 fois. La taille de la formule et le nombre de variable croissent exponentiellement avec le nombre d'équivalences traversées. Si chacune des ϕ_k possède un quantificateur à extraire, il sera impossible d'éviter le pire cas.

3.2 Mise sous forme prénexe et résultats intermédiaires

Il est habituel en programmation et en spécification QBF d'utiliser une variable existentiellement quantifiée pour représenter un résultat intermédiaire soit pour ne pas répéter une sous formule, soit pour la lisibilité, soit enfin par construction. Par extension d'un résultat propositionnel classique (Tseitin, 1970), nous nous intéressons au motif $\exists x((x \leftrightarrow F) \wedge G)$, x variable absente de F, qui est équivalent à $G[x \leftarrow F]$. Par la suite nous désignerons une telle variable x, *variable intermédiaire* ou *résultat intermédiaire*. Pour les procédures QBF actuelles, les variables intermédiaires sont traitées comme des variables du problème alors qu'il suffirait pour s'en abstraire de faire un remplacement syntaxique. Nous proposons de garder ces variables intermédiaires car elles peuvent aussi avoir un intérêt opérationnel, tout en évitant les recopies et l'explosion du nombre de variables. Pour ce faire nous exhibons un ensemble d'équivalences logiques permettant d'extraire le quantificateur d'un résultat intermédiaire. L'objectif est de ne créer aucune nouvelle variable et de ne pas recopier les sous formules.

Théorème 1
Soit F et G des QBF et x une variable représentant le résultat intermédiaire F, avec $x \notin \mathcal{V}(F)$, alors les équivalences logiques suivantes sont vraies :

1) $(\exists x((F \leftrightarrow x) \wedge G)) \equiv (\forall x((F \leftrightarrow x) \rightarrow G))$
2) $(\exists x((F \leftrightarrow x) \rightarrow G)) \equiv \top$
3) $(\forall x((F \leftrightarrow x) \wedge G)) \equiv \bot$

et pour une formule booléenne quantifiée $H = (qx((F \leftrightarrow x)\ o\ G))$ avec $q \in \{\exists, \forall\}$ et $o \in \{\wedge, \vee, \rightarrow\}$ il est possible de mettre H sous la forme d'une des trois équivalences.

Notre proposition s'appuie sur le théorème suivant qui permet d'extraire la définition de la variable intermédiaire lorsqu'elle est dans une sous formule d'une équivalence (ou d'un ou exclusif).

Théorème 2
Soit F, G et H des formules booléennes quantifiées et x une variable booléenne qui

représente le résultat intermédiaire F*, avec* $x \notin \mathcal{L}(H)$*,* $x \notin \mathcal{V}(F)$ *et* $o \in \{\oplus, \leftrightarrow\}$*, alors* $(H\ o\ (\exists x((F \leftrightarrow x) \wedge G))) \equiv (\exists x((F \leftrightarrow x) \wedge (H\ o\ G)))$.

L'objectif est atteint : le quantificateur qui porte sur x est extrait de l'équivalence ou du ou exclusif, aucune variable n'est créée, aucune sous formule n'est dupliquée. Il est possible de choisir comment sera quantifié x à l'aide du théorème 1, rien ne nous oblige à garder le même quantificateur. De même, pour extraire un quantificateur universel, il faut appliquer l'équivalence du théorème 1 sur la partie gauche de l'équivalence.

3.3 Vérification formelle de circuits : l'additionneur n-bits

Lors de la conception de circuits logiques, le but est de construire un circuit qui corresponde au comportement souhaité. C'est-à-dire que pour chaque jeu d'entrées possible, la sortie attendue est obtenue. Le plus souvent le modèle booléen du circuit diffère grandement du circuit conçu par l'ingénieur pour des raisons pratiques (encombrement, prix, intégration, etc...). La vérification formelle de circuit est un des problèmes qui s'expriment à l'aide de formules de la logique monadique. La construction de modèles bornés (Bounded Model Construction), est une méthode pour résoudre des problèmes de validité de formules de la logique monadique en les transformant en QBF et en cherchant la validité des formules obtenues. Parmi les circuits, la vérification de l'additionneur n-bits est devenu un problème classique dans sa version QBF et sa description complète peut être trouvée dans (Ayari *et al.*, 1999; Ayari & Basin, 2002). La vérification de l'additionneur consiste à vérifier l'équivalence en logique monadique entre la structure du circuit et le comportement souhaité (n est la taille de l'additionneur, A et B sont les deux n-bits à additionner, S le résultat de l'addition, c_i la retenue en entrée et c_o la retenue en sortie) :

$\phi = \forall n \forall A \forall B \forall S \forall c_i \forall c_o (add_{struc}(n, A, B, S, c_i, c_o) \leftrightarrow add_{comp}(n, A, B, S, c_i, c_o))$

Les formules add_{struc} et add_{comp} contiennent des variables existentielles, qui représentent des résultats intermédiaires, entre autres les retenues. Grâce aux théorèmes 1 et 2, nous avons le choix d'extraire les variables intermédiaires soit en existentielles, soit en universelles. Dans les deux cas nous pouvons obtenir (en sortant d'abord les quantificateurs universels puis existentiels) une alternance des quantificateurs de type $\forall\exists$ et conserver cette alternance après la mise sous FNC.

Nous donnons un exemple pour l'additionneur n-bits avec $n = 1$. La QBF ϕ_1 code la vérification de l'additionneur pour $n = 1$ avec les fonctions booléennes suivantes : $C_1(x) = (c_i \leftrightarrow x)$, $C_2(x) = (c_o \leftrightarrow x)$, $C_3(x) = (x \leftrightarrow (A_0 \oplus B_0))$, $C_4(x) = (x \leftrightarrow (A_0 \wedge B_0))$, $C_5(x, y, z) = (x \leftrightarrow (y \wedge z))$, $F_1(x, y) = (S_0 \leftrightarrow (x \oplus y))$, $F_2(x, y, z) = (x \leftrightarrow (y \vee z))$, $F_3(x, y) = ((((A_0 \wedge B_0) \vee (A_0 \wedge x)) \vee (B_0 \wedge x)) \leftrightarrow y)$, et $F_4(x) = (((A_0 \leftrightarrow B_0) \leftrightarrow S_0) \leftrightarrow x)$. Les fonctions C_k, $1 \leq k \leq 5$ représentent les motifs extraits de la QBF initiale ; les fonctions C_3, C_4 et C_5 représentent les définitions des variables intermédiaires x_3, x_4 et x_5 ; les fonctions F_k, $1 \leq k \leq 4$, représentent le cœur de la spécification de l'additionneur n-bits.

$\phi_1 = \forall c_i \forall c_o \forall A_0 \forall B_0 \forall S_0$
$[\exists x_1 \exists x_2 (((C_1(x_1) \wedge C_2(x_2)) \wedge$
$(\exists x_3 \exists x_4 \exists x_5 (C_3(x_3) \wedge (F_1(x_3, x_1) \wedge (C_4(x_4) \wedge (C_5(x_5, x_1, x_3) \wedge F_2(x_2, x_5, x_4)))))))))]$
$\leftrightarrow [\exists x_6 \exists x_7 ((C_1(x_6) \wedge C_2(x_7)) \wedge (F_3(x_6, x_7) \wedge F_4(x_6)))]$

La QBF ci-dessous ϕ_i est la QBF initiale ϕ_1 mise sous forme prénexe selon l'algorithme classique (les variables y_k sont issues des variables x_k par recopie des sous-formules de l'équivalence).

$\phi_i = \forall c_i \forall c_o \forall A_0 \forall B_0 \forall S_0 \forall y_1 \forall y_2 \forall y_3 \forall y_4 \forall y_5 \forall y_6 \forall y_7 \exists x_1 \exists x_2 \exists x_3 \exists x_4 \exists x_5 \exists x_6 \exists x_7$
$[C_1(y_1) \wedge C_2(y_2) \wedge C_3(y_3) \wedge (F_1(y_3, y_1) \wedge C_4(y_4) \wedge C_5(y_5, y_1, y_3) \wedge F_2(y_2, y_5, y_4)]$
$\rightarrow [C_1(x_6) \wedge C_2(x_7) \wedge F_3(x_6, x_7) \wedge F_4(x_6)]] \wedge$
$[C_1(y_6) \wedge C_2(y_7) \wedge F_3(y_6, y_7) \wedge F_4(y_6)]$
$\rightarrow [C_1(x_1) \wedge C_2(x_2) \wedge C_3(x_3) \wedge F_1(x_3, x_1) \wedge C_4(x_4) \wedge C_5(x_5, x_1, x_3) \wedge F_2(x_2, x_5, x_4)]]$

L'ordre des variables influe grandement sur l'efficacixté des méthodes de résolution (Egly *et al.*, 2003) ; dans la mise sous forme prénexe de ϕ_1 en ϕ_i, l'ordre des variables n'est contraint que par le respect de l'ordre partiel qui nécessite que les quantificateurs universels des variables initiales du problème doivent précéder les quantificateurs existentiels.

La QBF ci-dessous ϕ_t est la QBF initiale ϕ_1 transformée à l'aide de nos équivalences logiques puis mise sous forme prénexe selon l'algorithme classique.

$\phi_t = \forall c_i \forall c_o \forall A_0 \forall B_0 \forall S_0 \exists x_1 \exists x_2 \exists x_3 \exists x_4 \exists x_5 \exists x_6 \exists x_7$
$[C_1(x_1) \wedge C_2(x_2) \wedge C_3(x_3) \wedge C_4(x_4) \wedge C_5(x_5, x_1, x_3) \wedge C_1(x_6) \wedge C_2(x_7)] \wedge$
$[[F_1(x_3, x_1) \wedge F_2(x_2, x_5, x_4)] \leftrightarrow [F_3(x_6, x_7) \wedge F_4(x_6)]]$

La matrice de la QBF obtenue est partagée en deux parties : la définition des variables intermédiaires suivie de l'exploitation de ces variables dans le cœur de l'équivalence.

4 Résultats expérimentaux

Nous avons développé un algorithme en Prolog, qui permet de générer les instances de l'additionneur, de chercher le motif $(H \leftrightarrow (\exists x((F \leftrightarrow x) \wedge G)))$ et de le substituer par $(\exists x((F \leftrightarrow x) \wedge (H \leftrightarrow G)))$, puis d'effectuer la mise sous FNC. La recherche du motif est appliquée jusqu'à l'obtention d'un point fixe. La table 4 montre le nombre de quantificateurs existentiels et universels, pour différentes valeurs de n, de la FNC de l'additionneur n-bits sans et avec notre transformation (NQ_i est le nombre de quantificateurs de la FNC de la QBF initiale, $NQ_i^\forall$ le nombre de quantificateurs universels de la FNC de la QBF initiale, NQ_t le nombre de quantificateurs de la FNC de la QBF transformée, $NQ_t^\forall$ le nombre de quantificateurs universels de la FNC de la QBF transformée et NC_t le nombre de clauses de la FNC de la QBF transformée).

n	4	6	8	10	12	14	16	18	20	22
NQ_i	281	411	541	671	801	931	1061	1191	1321	1451
$NQ_i^\forall$	36	52	68	84	100	116	132	148	164	180
NQ_t	147	215	283	351	419	487	555	623	691	759
$NQ_t^\forall$	14	20	26	32	38	44	50	56	62	68
NC_t	383	561	739	917	1095	1273	1451	1629	1807	1985

TAB. 1 – Tailles des QBF initiales et transformées pour l'additionneur.

Les motifs sont extraits dans la version existentielle. La mise sous forme prénexe appliquée ne cherche pas une forme optimale pour l'ordre des quantificateurs (ce qui

demanderait une étude supplémentaire). La mise sous FNC est faite par introduction de variables intermédiaires quantifiées existentiellement qui ne fait croître que polynômialement la taille de la formule. Le nombre de clauses de la FNC de la QBF initiale est exactement le double du nombre de clauses de la FNC de la QBF transformée.

Les tests ont été effectués sur un Intel(R) Pentium(R) 4 à 2.80GHz avec 600Mo de mémoire vive disponible. Les résultats sont résumés dans la table 4, les temps sont en secondes, un T indique que la procédure n'a pas pu résoudre l'instance en moins de 3600 secondes, un M indique un dépassement de la mémoire disponible. Les temps d'exécution ne prennent pas en compte le temps de recherche et la substitution du motif. Les colonnes où figure la mention "sans" sont les résultats pour une formule sans recherche de motif. Les colonnes où figure un $\exists$ (resp. $\forall$) sont les résultats pour une formule où les variables intermédiaires sont extraites existentiellement (resp. universellement).

Les tests ont été effectués dans un premier temps avec les réglages par défaut de chaque procédure. Nous utilisons différentes procédures : sKizzo v0.8.2-beta (Benedetti, 2005) qui intègre skolémisation symbolique et appel à une procédure SAT, Quantor 3.0 (Biere, 2004) qui combine Q-résolution (Büning *et al.*, 1995) et expansion, QuBE-BJ1.2 (Giunchiglia *et al.*, 2004) qui étend aux QBF la procédure de Davis-Logemann-Loveland (Davis *et al.*, 1962) et QuBE6.3 une version plus récente qui utilise un préprocesseur (Bubeck & Büning, 2007) intégrant la Q-résolution. QuBE n'a pas réussi à résoudre le problème avec les QBF initiales pour $n > 3$.

Les résultats de la table 1 montrent que les procédures testées obtiennent presque toujours de meilleurs résultats avec les QBF transformées. La procédure sKizzo est la seule pour laquelle les résultats sont moins tranchés. Pour $n < 12$ l'amélioration est sensible, puis les performances se dégradent. Le traitement effectué sur les équivalences handicape sKizzo sur les instances plus grandes. En observant la trace de sKizzo, nous remarquons que la Q-résolution supprime des variables représentant des résultats intermédiaires. QuBE6.3 ayant de moins bons résultats que QuBE-BJ1.2, nous nous posons des questions sur l'impact de la Q-résolution sur ce problème. Nous avons effectué des tests en désactivant la Q-résolution de sKizzo. Que ce soit avec les QBF initiales ou les QBF transformées, désactiver la Q-résolution permet d'obtenir de meilleurs temps. Avec les QBF transformées le gain est significatif. Nous avons aussi essayé d'extraire les résultats intermédiaires sous une forme d'universelles, mais la mise sous FNC introduit des variables existentielles dépendant alors de bien plus de variables universelles. Les résultats sont moins bons que pour le codage original. Pour ce type de problème il faudrait utiliser une procédure pouvant raisonner sur des formules non FNC.

Pour la vérification formelle de circuits logiques, la Q-résolution semble travailler à l'encontre des autres méthodes. Une explication possible est que la résolution sur les variables intermédiaires (du codage ou de la mise sous FNC) enlève des possibilités aux procédures d'élaguer l'espace de recherche. Les mécanismes d'apprentissage retiennent des règles moins générales, le parcours de l'espace de recherche est plus long. Avec les QBF transformées, la Q-résolution a un impact très important sur les grandes instances, à tel point que les QBF initiales conviennent mieux à sKizzo. Les QBF transformées semblent être plus denses, augmentant l'importance des variables intermédiaires et la possibilité de réduire l'espace de recherche en raisonnant sur ces variables.

ϕ	sKizzo avec Q-résolution		sKizzo sans Q-résolution			QuBE 6.3	QuBE BJ1.2	Quantor	
n	sans	∃	sans	∃	∀	∃	∃	sans	∃
4	0,3	**0,1**	0,5	**0,1**	1,0	2,98	0,26	8,53	0,14
6	1,0	**0,1**	1,3	**0,1**	11,5	T	17,90	M	3,04
8	5,5	0,4	13,3	**0,2**	23,1	T	1199.67	M	53,43
10	23,7	6,0	11,5	**0,4**	53,2	T	T	M	M
12	170,9	143,7	18,0	**0,5**	128,1	T	T	M	M
14	915,6	T	19,7	**0,8**	1710,4	T	T	M	M
16	T	T	24,4	**0,9**	T	T	T	M	M
18	T	T	68,4	**1,6**	T	T	T	M	M
20	T	T	130,9	**1,3**	T	T	T	M	M
22	T	T	101,4	**2,2**	T	T	T	M	M

TAB. 2 – Résultats pour différentes procédures avec les QBF initiales et les QBF transformées.

5 Conclusion

Dans cet article, nous proposons différentes équivalences qui permettent de traiter les résultats intermédiaires contenus dans des équivalences ou des ou exclusifs pour les formules booléennes quantifiées. Ces équivalences permettent de conserver sa taille et le nombre de ses variables, tout en sortant les quantificateurs afin de pouvoir appliquer une mise sous forme prénexe. Il est possible de choisir le type de quantification de la variable intermédiaire et ainsi réduire le nombre d'alternances de quantificateurs. Les QBF peuvent être vues comme un jeu à deux joueurs : le joueur existentiel essaye de rendre la formule valide alors que le joueur universel essaye de l'invalider. Un résultat intermédiaire est une conséquence des choix précédents, il n'y a aucun choix à faire, peu importe quel joueur joue ce coup. Notre proposition va dans ce sens, en permettant de faire jouer un seul des deux joueur, au choix, lors d'un résultat intermédiaire dans une équivalence ou sa négation, sans augmenter ni la taille de la formule ni le nombre de variables.

Pour notre partie expérimentale, nous avons choisi la vérification formelle de circuits logiques, où une spécification doit être équivalente à la structure du circuit. Nos résultats montrent que notre codage permet d'avoir de meilleurs résultats sans sur-coût. Toutefois, il est possible de rechercher les motifs de résultats intermédiaires afin d'effectuer un remplacement. Nos tests montrent que pour le problème choisi, la Q-résolution est néfaste au temps de résolution. Il serait intéressant de voir, si celle-ci pourrait être de nouveau intéressante si elle ne s'appliquait pas sur des résultats intermédiaires.

Pour utiliser nos équivalences, soit il faut coder le problème en les utilisant directement, soit il faut posséder le codage original du problème et effectuer des remplacements. Il serait donc intéressant d'avoir des procédures qui travaillent directement sur la formulation non prénexe. Elles pourraient ensuite en interne mettre sous FNC si nécessaire et classer les variables en deux catégories : les variables du problème et les résultats intermédiaires. Il serait alors possible de connaître l'impact des heuristiques telle la Q-résolution et de traiter différemment les résultats intermédiaires.

Dans nos travaux futurs, nous nous intéresserons à :

– étendre nos résultats au cas général, afin d'être capable de traiter les quantificateurs dans des équivalences et des ou exclusifs, pas seulement pour les résultats intermédiaires ;
– comparer les nouvelles stratégies possibles pour la mise sous forme prénexe car nos résultats semblent montrer que le choix du type de quantificateur pour les résultats intermédiaires peut influencer grandement le temps de résolution ;
– développer une procédure qui prend en entrée une formule quelconque (non prénexe et non FNC), en effet nos résultats montrent qu'il est difficile de trouver une bonne stratégie pour mettre sous FNC.

Références

AYARI A. & BASIN D. (2002). Qubos : Deciding Quantified Boolean Logic using Propositional Satisfiability Solvers. In *Formal Methods in Computer-Aided Design, Fourth International Conference, FMCAD 2002* : Springer-Verlag.

AYARI A., BASIN D. A. & FRIEDRICH S. (1999). Structural and Behavioral Modeling with Monadic Logics. In *ISMVL*, p. 142–151.

BENEDETTI M. (2005). sKizzo : A Suite to Evaluate and Certify QBFs. In R. NIEUWENHUIS, Ed., *CADE*, volume 3632 of *LNCS*, p. 369–376 : Springer.

BIERE A. (2004). Resolve and Expand. In *7th Intl. Conf. on Theory and Applications of Satisfiability Testing (SAT'04)*.

BUBECK U. & BÜNING H. K. (2007). Bounded Universal Expansion for Preprocessing QBF. In J. MARQUES-SILVA & K. A. SAKALLAH, Eds., *SAT*, volume 4501 of *LNCS*, p. 244–257 : Springer.

BÜNING H. K., KARPINSKI M. & FLÖGEL A. (1995). Resolution for Quantified Boolean Formulas. *Inf. Comput.*, **117**(1), 12–18.

DAVIS M., LOGEMANN G. & LOVELAND D. W. (1962). A machine program for theorem-proving. *Commun. ACM*, **5**(7), 394–397.

DE LA TOUR T. B. (1992). An Optimality Result for Clause Form Translation. *J. Symb. Comput.*, **14**(4), 283–302.

EGLY U. (1996). On Different Structure-Preserving Translations to Normal Form. *J. Symb. Comput.*, **22**(2), 121–142.

EGLY U., SEIDL M., TOMPITS H., WOLTRAN S. & ZOLDA M. (2003). Comparing Different Prenexing Strategies for Quantified Boolean Formulas. In E. GIUNCHIGLIA & A. TACCHELLA, Eds., *SAT*, volume 2919 of *LNCS*, p. 214–228 : Springer.

GIUNCHIGLIA E., NARIZZANO M. & TACCHELLA A. (2004). QuBE++ : An Efficient QBF Solver. In A. J. HU & A. K. MARTIN, Eds., *FMCAD*, volume 3312 of *LNCS*, p. 201–213 : Springer.

GIUNCHIGLIA E., NARIZZANO M. & TACCHELLA A. (2006). Quantifier structure in search based procedures for qbfs. In *Proceedings of the conference on Design, automation and test in Europe (DATE'06)*, p. 812–817.

PLAISTED D. A. & GREENBAUM S. (1986). A Structure-Preserving Clause Form Translation. *Journal of Symbolic Computation*, **2**(3), 293–304.

TSEITIN G. S. (1970). On the complexity of derivation in propositional calculus. In A. O. SLISENKO, Ed., *Studies in Constructive Mathematics and Mathematical Logic, Part 2*, p. 115–125. New York : Consultants Bureau.

Redondance dans les CNF : laissons le solveur agir

Cédric Piette

Université Lille-Nord de France, Artois
CRIL-CNRS UMR 8188
F-62307 Lens
piette@cril.fr

Résumé : La redondance d'information en logique propositionnelle est une piste de recherche très active. La complexité de certains problèmes liés à la redondance a par exemple récemment été établie pour les CNF, ainsi que pour ses fragments 2-SAT et Horn par Liberatore (2005, 2008). Toutefois, ce phénomène de redondance n'est pas considéré par les solveurs SAT modernes, et est la plupart du temps tout simplement ignoré. Gérer la redondance au sein des CNF afin d'améliorer l'efficacité des solveurs est pourtant un challenge important. Dans ce papier est étudiée une solution adaptative, permettant de discrimer l'information redondante et de ne conserver que les éléments non fondamentaux qui se montrent *utiles* pendant la recherche.

1 Introduction

SAT est le problème de décision NP-complet qui consiste à vérifier si un ensemble de clauses booléennes (appelé CNF) admet au moins une affectation logique qui le safistasse. Il s'agit d'un problème central pour la discipline informatique, qui possède de nombreuses applications dans des domaines variés tels que la bioinformatique, la vérification formelle, l'intelligence artificielle ou la conception assistée par ordinateur (CAO). Une grande communauté de chercheurs s'intéresse de près à l'étude théorique et pratique de ce problème (cf. `http://www.SATlive.org`) et de ses applications.

Les problèmes liés à la redondance au sein des CNF ont été l'objet de nombreuses études, mais celles-ci se concentrent la plupart du temps sur les aspects théoriques de ces problèmes. En effet, le problème de minimisation d'une formule propositionnelle est connu depuis la première formalisation de la hiérarchie polynomiale (Meyer & Stockmeyer (1972)). Des résultats de complexité à propos de problèmes liés à la redondance ont été établis par Liberatore (2005). D'autres résultats ont également été proposés pour des cas restreints, par exemple quand la formule est exclusivement composée de clauses de Horn (cf. Ausiello *et al.* (1986); Hammer & Kogan (1993)).

L'importance de la redondance pour la résolution pratique de SAT a ensuite été soulignée par plusieurs études empiriques. La première, conduite par Boufkhad & Roussel

(2000) sur des instances générées aléatoirement, a par exemple permis d'observer que les formules irredondantes sont en général plus difficiles à résoudre qu'avec l'ajout d'information redondante. Ce travail a été ensuite étendu par Zeng & McIlraith (2005), en montrant en particulier que la « difficulté » de résolution des CNF peut souvent être mise en lien avec la taille de ses sous-formules irredondantes.

En dépit de ces travaux qui montrent une relation forte entre redondance et coût calculatoire de la satisfaisabilité, il s'agit un problème qui est ignoré en pratique par les solveurs, car il nécessiterait tout d'abord d'extraire, ou détecter, de l'information redondante. Il s'agit malheureusement d'un problème d'une forte complexité dans le pire cas. En effet, les problèmes liés à la redondance sont situés au moins au premier niveau de la hiérarchie polynomiale (cf. Liberatore (2005)), ce qui les rend aussi difficiles que le problème SAT lui-même. Dans ce papier, une première approche pratique pour gérer la redondance de clauses pendant une recherche pour la satisfaisabilité est présentée et expérimentalement évaluée. Tout d'abord, pour circonvenir à la trop grande complexité du test de redondance, une forme d'inférence incomplète, mais linéaire en temps est utilisée. Ensuite, comme l'ajout ou le retrait d'information redondante peut, suivant les cas, améliorer ou dégrader les performances d'un solveur, une stratégie adaptative est utilisée pour manipuler l'ensemble de clauses redondantes extrait polynomialement.

Ce papier est organisé comme suit. Dans la section suivante, les notations basiques concernant la logique propositionnelle ainsi que des définitions liées à la redondance sont données. Un nouveau schéma de résolution de SAT permettant de gérer la redondance est ensuite présenté en section 3. Dans la section 4, une étude empirique montre l'intérêt de l'approche. Enfin, nous concluons par quelques perspectives de recherche.

2 Sur la redondance des CNF

Soit $\mathcal{L}$ le langage booléen standard, construit sur un ensemble fini de variables. Une formule CNF est un ensemble (interprété comme une conjonction) de clauses, une clause étant un ensemble (interprété comme une disjonction) de littéraux. Un littéral est une variable booléenne x ou sa négation $\neg x$. Deux littéraux x et $\neg x$ sont dits *complémentaires*. On note $\bar{l}$ le complémentaire d'un littéral l. Pour un ensemble de littéraux L, $\bar{L}$ est défini comme $\{\bar{l} \mid l \in L\}$. Une clause *unitaire* est une clause qui ne contient qu'un seul littéral, tandis qu'une clause *vide*, notée $\bot$, est interprétée comme $faux$. L'ensemble des variables apparaissant dans une formule CNF Σ est notée V_Σ. Une *interprétation* ρ d'une formule Σ associe une valeur $\rho(x)$ à chaque variable $x \in V_\Sigma$ et est appelée *modèle* si elle la satisfait (noté $\rho \vDash \Sigma$).

Soit Σ une formule CNF. On note $\Sigma|_x$ la formule obtenue en affectant le littéral x à la valeur $vrai$. Formellement, $\Sigma|_x = \{c \mid c \in \Sigma, \{x, \neg x\} \cap c = \emptyset\} \cup \{c \backslash \{\neg x\} \mid c \in \Sigma, \neg x \in c\}$. Cette notation est étendue aux ensembles de littéraux : étant donné $\rho = \{x_1, \ldots, x_n\}$, on définit $\Sigma|_\rho = (\ldots((\Sigma|_{x_1})|_{x_2})\ldots|_{x_n})$. Par ailleurs, on note Σ^* la formule CNF Σ close pour la propagation unitaire, définie récursivement comme suit :

1. $\Sigma^* = \Sigma$ si Σ ne contient pas de clause unitaire ;
2. $\bot \in \Sigma^*$ si Σ contient deux clauses unitaires $\{x\}$ et $\{\neg x\}$;
3. sinon, $\Sigma^* = (\Sigma|_x)^*$ où x est un littéral inclus dans une clause unitaire de Σ.

De plus, une clause c est impliquée par propagation unitaire à partir de Σ, noté $\Sigma \models^* c$, si $\bot \in (\Sigma|_{\bar{c}})^*$.

La présence d'information redondante au sein d'une CNF peut jouer un rôle important pour décider de sa satisfaisabilité. En effet, rendre explicite un grand nombre de clauses induites mais non exprimées peut aider les solveurs. Néanmoins, la présence de ce type d'information trouve ses limites dans les problèmes de stockage et de mise-à-jour. Ainsi, la présence d'un grand nombre d'information redondante ne fait que ralentir le processus de résolution. Définissons maintenant formellement une clause redondante.

Définition 1
Soit Σ une formule CNF, et $c \in \Sigma$ une clause. c est dite redondante *(par rapport à Σ) si et seulement si $\Sigma \backslash \{c\} \models \{c\}$. Toute clause non redondante est dite* irredondante.

À travers cette définition, il semble clair que toute clause redondante peut être ôtée d'une formule CNF, en préservant non seulement sa satisfaibilité mais également l'ensemble de ses modèles.

Exemple 1
Soit $\Sigma = \{\neg a \vee b \vee c, \neg b \vee c, \neg c, a \vee \neg b, c \vee d, \neg a \vee c \vee \neg d\}$ une formule CNF. La clause $\phi_1 = \neg b \vee c$ est redondante par rapport à Σ, puisque $(\Sigma \backslash \{\phi_1\})|_{\bar{\phi_1}}$ est clairement insatisfaisable. Les clauses $\phi_2 = \neg a \vee b \vee c$, $\phi_3 = a \vee \neg b$ et $\phi_4 = \neg a \vee c \vee \neg d$ sont également redondantes par rapport à Σ.

Décider si une clause est redondante est clairement CoNP-complet, et en itérant un tel test sur toutes les clauses d'une CNF et en retirant chaque clause prouvée redondante, il est possible d'obtenir une *formule irredondante*.

Définition 2
Soit Σ une CNF. Σ est appelée formule irredondante *ssi $\forall c \in \Sigma$, c est irredondante.*

Cette notion de formule irredondante est utilisée depuis longtemps, bien qu'elle ait été baptisée de différentes façons dans la littérature. Par exemple, elle est définie sous le nom de *irredundant equivalent subset* par Liberatore (2005) et *satisfiable core* par Boufkhad & Roussel (2000). De plus, de nombreuses études se sont récemment concentrées sur le problème d'expliquer *pourquoi* une CNF ne possède aucune solution, à travers le concept de MUS (*Minimally Unsatisfiable Subformula*) (cf. Grégoire *et al.* (2007)). Un MUS est en fait une formule irredondante dans le cas particulier de l'insatisfaisabilité.

Une formule CNF peut posséder plusieurs formules irredondantes ; en fait, une CNF contenant m clauses peut posséder dans le pire cas $C_m^{m/2}$ formules irredondantes.

Exemple 2
Soit Σ la formule CNF de l'exemple précédent. Σ possède 3 sous-formules irredondantes, qui sont $\Sigma_1 = \Sigma \backslash \{\phi_1, \phi_2\}$, $\Sigma_2 = \Sigma \backslash \{\phi_2, \phi_3\}$ et $\Sigma_3 = \Sigma \backslash \{\phi_3, \phi_4\}$. Ces 3 formules sont illustrées dans la figure 1.

Le rôle des clauses redondantes au sein des CNF reste flou. Boufkhad & Roussel (2000) ont montré que les formules irredondantes sont typiquement plus difficiles à résoudre qu'avec la présence de clauses redondantes. Il est en effet connu que ces clauses

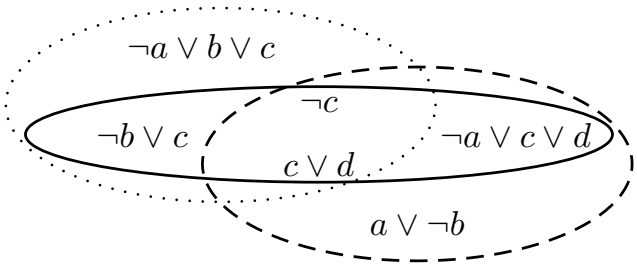


FIG. 1 – Les 3 formules irredondantes de l'exemple 2

peuvent en pratique aider les solveurs ; à titre d'exemple, les mécanismes d'apprentissage (Beame *et al.* (2003)), qui produisent une résolvante particulière après chaque conflit peuvent être vu comme un ajout dynamique de clauses redondantes pendant la recherche. Cette stratégie d'apprentissage est l'un des éléments clés de l'efficacité des solveurs modernes, ce qui montre l'intérêt de l'information redondante par rapport à la résolution pratique de SAT. Cependant, il est également connu qu'ajouter trop d'information redondante peut au contraire ralentir la recherche. D'ailleurs, une simple expérience consistant à ajouter à une CNF toutes les clauses apprises après sa résolution, montre que cette nouvelle formule est typiquement plus difficile à résoudre.

Ainsi, il paraît difficile de prédire si une clause redondante va se montrer utile pour la résolution d'une formule CNF donnée, en particulier parce que cela dépend de la façon dont l'espace de recherche est exploré. Dans la section suivante, une approche pour gérer l'information redondante est proposée.

3 Un nouveau schéma pour la gestion de la redondance

3.1 Extraire des clauses redondantes en temps polynomial

Comme présenté dans la section précédente, l'information redondante peut s'avérer utile pour la résolution de SAT, ou la ralentir considérablement. Nous proposons ici une technique originale permettant de gérer ce type d'information. Malheureusement, cela induit de pouvoir extraire des clauses redondantes au sein des CNF. S'agissant d'un problème aussi difficile que SAT lui-même, effectuer un tel test pour la résolution pratique de ce problème n'a donc aucun sens. Pourtant, une approche récente de Fourdrinoy *et al.* (2007) vise à détecter de manière incomplète des clauses redondantes en temps polynomial. L'idée générale de ce test est de n'effectuer ce test de redondance qu'à travers la propagation unitaire.

Définition 3
Soit Σ une formule CNF et $c \in \Sigma$ une clause. c est appelée U-redondante *si et seulement si $\Sigma \backslash \{c\} \models^* c$, i.e. $\bot \in (\Sigma \backslash \{c\})|_{\bar{c}})^*$.*

Données: Σ : une formule CNF
Résultat: $vrai$ si Σ est satisfaisable, $faux$ sinon

début
  $\Sigma_{ur} \leftarrow \emptyset$;
  $\Sigma_{uir} \leftarrow \Sigma$;
  pour chaque $c \in \Sigma$ **faire**
    $\Sigma_{uir} \leftarrow \Sigma_{uir} \backslash \{c\}$;
    Soit $c = \{l_1, \ldots, l_k\}$;
    $i \leftarrow 1$; *conflit* $\leftarrow$ *faux* ;
    tant que $i \leq k$ **et** *conflit* $=$ *faux* **faire**
      si $(\Sigma_{uir}|_{\{\bar{l_1},\ldots,\bar{l_i}\}})^*$ = $\bot$ **alors** $\Sigma_{ur} \leftarrow \Sigma_{ur} \cup \{l_1, \ldots, l_i\}$;
      $i \leftarrow i + 1$;
    fin
    si *conflit* $=$ *faux* **alors** $\Sigma_{uir} \leftarrow \Sigma_{uir} \cup \{c\}$;
  fin
  retourner `solve`$(\Sigma_{uir},\Sigma_{ur})$;
fin

Algorithm 1: Solveur U-redSAT

Considérer les clauses U-redondantes permet d'éviter toute explosion combinatoire tout en extrayant un grand nombre de clauses redondantes en pratique. En fait, toute formule CNF Σ peut être partitionnée en temps polynomial en $\Sigma_{uir} \cup \Sigma_{ur}$, où Σ_{uir} est une formule U-irredondante de Σ et Σ_{ur} est un ensemble de clauses redondantes par rapport à Σ. Assez clairement, l'ordre dans lequel les tests de U-redondance sont effectués peut mener à différentes partitions de Σ.

3.2 Gérer les clauses extraites de manière spécifique

L'intuition de la technique présentée ici est que l'un des mécanismes implémenté dans les solveurs modernes pourrait être en charge de la sélection des clauses redondantes pertinentes par rapport à la recherche en cours. En effet, les solveurs actuels produisent de l'information redondante après chaque conflit, à travers leur fonction d'apprentissage. Ils doivent donc faire face aux problèmes de stockage et de mise-à-jour de cette information additionnelle. Cet aspect est contrôlé via différentes techniques dont le but est d'assurer un bon compromis entre la quantité de clauses à mettre-à-jour et la capacité à propager de l'algorithme. Actuellement, la stratégie la plus largement utilisée est inspirée de l'heuristique de branchement de variables VSIDS, et vise à ne conserver que les clauses les plus actives, c'est-à-dire celles permettant de filtrer le plus l'espace de recherche (Goldberg & Novikov (2002)). Plus précisément, un compteur (initialisé à 0) appelé *activité* est associé à chaque clause apprise, et est incrémenté chaque fois que la clause correspondante est « utilisée » pour déclencher une propagation unitaire. Périodiquement, la base de clauses apprises est purgée des éléments possédant les scores les plus bas, suivant le principe que les clauses les plus utiles jusqu'à présent le seront également pendant le reste de la recherche.

Instance			Test de Redondance			Temps de résolution	
Name	#cla	S/U	#cla$_{red}$	$\%_{red}$	temps	`minisat`	`minisat`$_{red}$
hanoi5u	59 718	U	2 139	3,58%	1,21	214,87	164,15
bqwh.33.381	9 040	S	494	5,46%	0,06	456,5	38,32
5col100_15_6	3 473	U	397	11,43%	0,03	56,93	30,88
lksat-n2000-m6840-k3-l4	6 598	U	242	3,66%	0,02	28,72	29,66
Composite-024BitPrimes-0	9 689	S	793	8,18%	0,05	206,25	163,71
shuffling-1-s1025511904	42 818	U	4 818	11,3%	2,17	284,04	332,98
manol-pipe-c6id	238 142	U	3 899	1,64%	3,38	150,58	147,97
ferry12	23 285	S	7 285	31,3%	0,43	0,85	0,94
avg-checker-5-34	33 206	U	2 700	8,13%	1,09	55,09	28,77
2dlx_cc_mc_ex_bp_f2_bug015	149 693	S	37 664	25,2%	40,35	2,31	2,35
9vliw_bp_mc	156 921	U	22 542	14,4%	59,99	248,77	898,7
k2fix_gr_2pinvar_w8	270 136	U	0	0%	30,04	67,52	67,48
hanoi6	22 633	S	11 120	49,1%	0,21	132,56	87,69
shuffling-2-s1182968979	58 408	U	4 487	7,68%	5,05	405,65	348,83
frg2mul.miter	58 477	U	4 418	7,56%	2,02	540,28	305,04
CompositeRSA640	1 929 086	?	105 148	5,45%	26,04	*time out*	*time out*
lksat-n900-m6174-k4-l4-s...	6 084	U	90	1,48%	0,02	57,59	109,26
4pipe_1_ooo	60 564	U	13 956	23,04%	4,19	184,56	82,56
rand_net70-30-5	12 071	U	390	3,23%	0,33	266,33	344,21
gripper10	14 126	S	3 782	26,77%	0,21	*time out*	192,95

TAB. 1 – Taux de redondance et temps de résolution d'un échantillon de problèmes

Nous proposons donc d'utiliser la méthode de Fourdrinoy *et al.* (2007) pour capturer efficacement un ensemble de clauses redondantes au sein d'une CNF, et d'informer le solveur de la nature de ces clauses. Dans ce but, les clauses détectées sont éliminées de la CNF et placées dans la base de clauses apprises du solveur. Grâce à cette approche, les clauses redondantes les plus utiles (i.e. avec la plus haute *activité*) seront conservées, tandis que les clauses non pertinentes seront progressivement supprimées.

L'approche proposée est décrite dans l'algorithme 1. Tout d'abord, les clauses de Σ sont testées pour la U-redondance. En pratique, les opposés des littéraux de la clause courante c sont affectés un par un (littéral l_i avec $1 \leq i \leq k$), et dès qu'un conflit survient, la clause composée des littéraux l_1 à l_i est enregistrée dans la partie redondante Σ_{ur} (lignes 7 à 11). Si au contraire c n'est pas U-redondante, alors elle est replacée dans la formule courante Σ_{uir} (ligne 12).

À la fin de cette étape de prétraitement, deux CNF Σ_{uir} et Σ_{ur} sont obtenues. Supposons maintenant qu'un solveur SAT appelé `solve` soit modifié de sorte que son premier paramètre soit le problème donné en entrée sous forme CNF, et le second un ensemble de clauses apprises « initiales » : un appel classique à un tel solveur serait donc « `solve`(Σ,$\emptyset$) ». Au lieu de cet appel, après l'étape préliminaire le solveur modifié est invoqué avec les formules Σ_{uir} et Σ_{ur} (ligne 14).

Ce schema de résolution de SAT, du test de redondance à l'utilisation particulière des clauses U-redondantes, peut être très facilement greffé à la plupart des solveurs actuels. Dans la section 4, les idées proposées sont évaluées d'un point de vue empirique.

4 Résultats expérimentaux

Dans le but de valider la viabilité pratique des idées présentées, nous avons implémenté le test de U-redondance pour calculer (de manière incomplète) un ensemble de clauses redondantes au sein d'une CNF. Comme l'ordre dans lequel les tests de U-redondance sont effectués a un impact sur la sous-formule U-irredondante obtenue, nous avons choisi de trier les clauses dans l'ordre décroissant de leur taille, afin d'obtenir une sous-formule close pour la subsumption. La procédure retourne donc une partition clausale, la première partie étant une sous-formule U-irredondante et la seconde une base de clauses redondantes.

Nous avons ensuite modifié une méthode complète dans le but de tenir compte de cette information en considérant les clauses redondantes comme des clauses apprises initiales. Nous avons ensuite comparé le comportement de notre solveur modifié avec la version originale, qui considère toutes les clauses de la CNF comme fondamentales (aucune ne peut être supprimée). Comme cas d'étude, `minisat` (Eén & Sorensson (2008)) a été choisi comme implémentation de solveur moderne. Précisons en outre que nous avons volontairement conservé tel quel les paramètres de `minisat` dans le but d'obtenir une comparaison juste entre les 2 solveurs. En particulier, le nombre de clauses apprises conservées est initialement $|\Sigma|/3$ et est incrémenté régulièrement. Ceci signifie qu'à tout moment de la recherche, les 2 solveurs possèdent la même borne sur le nombre de clauses apprises, ce qui contribue à les exécuter dans les même conditions. Néanmoins, notre version commence son calcul avec un ensemble de clauses apprises non vide, et quand cet ensemble dépasse le tiers du nombre total de clauses, certaines sont supprimées dès qu'un conflit survient. Fort heureusement, comme le montrent nos résultats expérimentaux, ce cas ne se produit que de très rares fois.

Nos expérimentations ont été conduites sur une sélection de 940 problèmes réels (académiques et industriels) venant des dernières Compétitions SAT (2008). Tout d'abord, un échantillon des résultats obtenus est proposé dans la table 1, qui est divisé en trois parties. La première partie fournit des informations sur l'instance testée (nom, nombre de clauses et satisfaisabilité) La deuxième partie de la table se concentre sur le test de U-redondance, en reportant le nombre de clauses redondantes, le pourcentage de clauses qu'elles représentent par rapport à l'instance originale, et le temps en secondes nécessaire à cette détection. Enfin, dans la troisième partie de la table présente le temps (en secondes) requis pour résoudre la formule CNF avec le solveur original et modifié (noté $\texttt{minisat}_{red}$). Nos études expérimentales ont été conduites sur des processeurs Intel Xeon 3GHz sous Linux CentOS 4.1. (noyau 2.6.9) avec une limite mémoire de 2Go. Pour toutes ces expérimentations, une limite de temps de 900 secondes a été respectée. Si un calcul dépasse cette limite, alors la mention *time out* est reportée.

Tout d'abord, concentrons-nous sur le test de U-redondance. Les résultat obtenus montrent qu'il existe des modélisations générant un très grand nombre de clauses redondantes. À titre d'exemple, les benchmarks `2dlx...bug015` et `ferry12` possèdent respectivement au moins 25,2% et 31,3% de leurs clauses qui sont redondantes. Limiter ce test à la propagation unitaire permet donc de capturer un nombre important de clauses redondantes grâce à cet algorithme calculatoirement léger : la plupart du temps, ce test peut être effectué en moins de 5 secondes, bien que pour les très grandes instances,

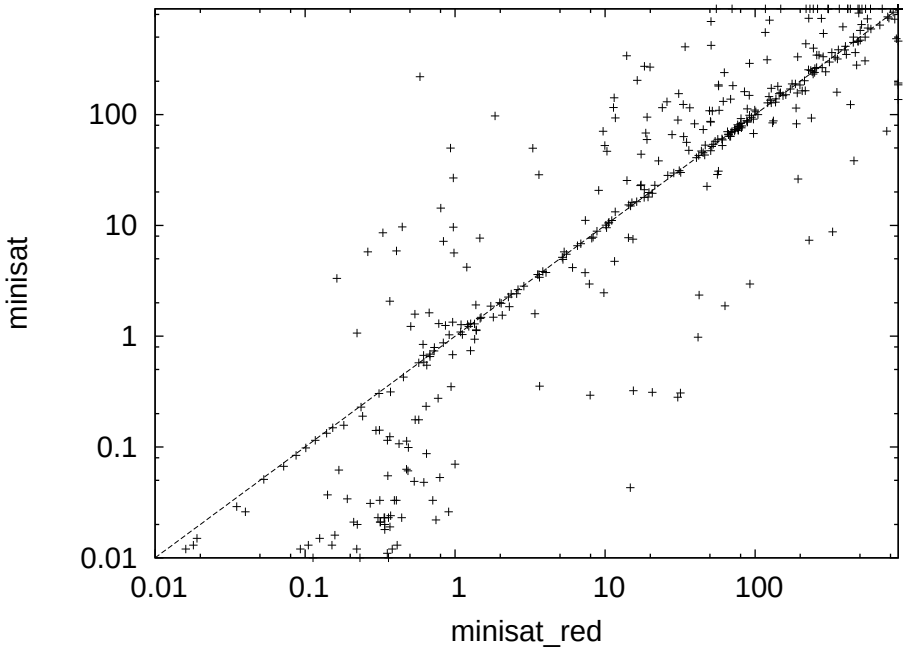


FIG. 2 – Minisat tenant (ou non) compte des clauses redondantes détectées

plusieurs dizaines de secondes peuvent s'avèrer nécessaires (`CompositeRSA640`). De tous les benchmarks testés, la formule CNF le plus redondante est l'encodage clausal propositionnel du problème des tours de Hanoi de taille 6. Ce problème possède pas moins de 22633 clauses, mais près de la moitié d'entre elles peut être inférée par l'autre moitié, et ne sont pas nécessaires pour coder le problème. Heureusement, pour la plupart des instances testées durant nos expérimentations, le « taux de redondance » n'excède pas 15%. En fait, dans certains cas, la procédure n'a été capable de détecter aucune clause redondante (`k2fix_gr_2pinvar_w8`).

Considérons maintenant les conséquences de cette information obtenue polynomialement. Ces quelques résultats montrent que discriminer les clauses U-redondantes accélère souvent la résolution d'instances. En outre, il a été observé que certaines clauses sont parfois conservées un long moment pendant la recherche ou peuvent éliminées assez rapidement, quand des clauses plus *utiles* sont apprises. Notons que même quand la CNF possède un taux de redondance relativement faible (< 15%), notre approche améliore souvent le comportement du solveur (`Composite-024BitPrimes-0, avg-checker-5-34`). Bien sûr, prendre en considération la redondance des clauses peut dans certains cas se révèler moins efficace qu'ignorer leur état. Par exemple, en dépit de la découverte de 14,3% de clauses U-redondantes, la version originale du solveur `minisat` prouve l'insatisfaisabilité de la formule `9vliw_bp_mc` en près de 4 minutes, alors que considérer ces clauses comme optionelles pour le même solveur le mène à un calcul de presque 15 minutes.

De manière plus générale, nous avons comparé le temps nécessaire à ces versions du

même solveur sur l'ensemble des tests réalisés. Les résultats obtenus sont fournis dans la figure 2 sous la forme d'un nuage de points. Chaque point de cette figure représente le temps de résolution d'un problème : le temps de $\texttt{minisat}_{red}$ est donné par la projection de ce point sur l'axe des abscisses, tandis que le temps de la version originale est donnée par sa projection sur l'axe des ordonnées. De cette manière, un point situé au dessus (dessous) de la première diagonale signifie que la version modifiée a nécessité moins (resp. plus) de temps pour résoudre le problème que la version originale. Notons en outre que les deux axes sont reportés à une échelle logarithmique.

Premièrement, on peut remarquer que pour les problèmes les plus faciles à résoudre (temps de résolution inférieur à 1 secondes), le solveur « classique » est en général la version la plus efficace. Ce phénomène est expliqué par le surcoût dû au test de redondance préliminaire. Celui-ci n'est en général pas très lourd, mais avec des problèmes si faciles à résoudre, ce temps passé a son importance. De plus, comme mentionné plus haut, pour de très grandes CNF, ce test peut nécessiter plusieurs secondes. Quant un tel benchmark est en réalité très simple pour une implémentation DPLL, le calcul inutile des clauses U-redondantes détériore les résultats. Heureusement, cette perte de temps ne représente dans les pires cas observés que quelques dizaines de secondes. Considérons maintenant les problèmes plus difficiles (temps de résolution supérieur à 10 secondes). Sur de tels problèmes, choisir parmi les clauses prouvées redondantes celles qui déclenchent le plus de propagations se montre très souvent viable. En effet, la plupart des points sont situés au dessus de la diagonale, ce qui montre l'amélioration de `minisat` par notre nouvelle approche. Enfin, notons que de nombreux points sont situés autour de la diagonale. Ceci est expliqué par la fait qu'il existe certaines CNF qui ne possèdent qu'un faible taux de U-redondance, voire aucune clause de ce type. Évidemment, dans de tels cas le comportement des deux solveurs est très similaire, voire identique dans le cas d'échec de l'étape préliminaire.

Néanmoins, de manière générale, ces premiers résultats plaident pour plus d'attention à propos de la redondance au sein des CNF pour la résolution pratique de SAT. Notre première implémentation fournit clairement des résultats prometteurs, et utiliser les mécanismes existants créés pour l'apprentissage semble adéquat pour la gestion de clauses redondantes. Assez clairement, de meilleurs résultats peuvent être attendus en réglant précisément les différents paramètres du solveur, et tout spécialement le nombre initial de clauses apprises.

5 Conclusion

Dans ce papier, une nouvelle stratégie pour la gestion de clauses redondantes au sein des CNF est présentée. Cette technique a l'avantage d'être facilement implémentable dans la plupart des solveurs actuels, grâce à une utilisation originale de leurs caractéristiques. Plus précisément, l'idée est d'extraire polynomialement un ensemble de clauses redondantes. La stratégie de type VSIDS des solveurs modernes est ensuite appliquée à cet ensemble de clauses, et si certaines d'entre elles ne permet pas de propager efficacement pendant la recherche, le solveur est alors libre de les supprimer. Cette technique est empiriquement validée par des expérimentations intensives qui montrent son intérêt pratique.

Ce travail ouvre de nombreuses pistes de recherche très intéressantes. D'abord, comme mentionné dans ce papier, l'ordre dans lequel les clauses sont testées pour la U-redondance peut être d'une grande importance. Le choix que nous avons fait, basé sur la subsumption, s'est montré pertinent, mais de nouveaux doivent être testés. De plus, les solveurs SAT proposent souvent un grand nombre de paramètres qui sont cruciaux pour l'efficacité de la procédure. Il a été choisi de ne pas modifier ces paramètres dans notre version test, mais certains d'entre eux, comme le nombre de clauses apprises autorisées à être conservées, pourraient être redéfinis pour tenir compte de la quantité d'information initialement fournie. Enfin, cette étude se concentre sur la façon de gérer l'information redondante déjà présente au sein des CNF. Il serait également intéressant de produire des clauses redondantes en utilisant des formes de résolution limitée, comme certains préprocesseurs tels que `HyPre` de Bacchus & Winter (2003) le font. Ces clauses produites pourraient être considérées comme apprises, dans le but d'aider le solveur quand elles se montrent effectivement utiles. Si certaines d'entre elles ne sont au contraire pas pertinentes pendant la recherche, le solveur serait alors capable de s'en débarasser. Nous prévoyons d'explorer ces différentes pistes de recherche.

Références

AUSIELLO G., D'ATRI A. & SACCÁ D. (1986). Minimal representation of directed hypergraphs. *SIAM Journal on Computing*, **15**(2), 418–431.

BACCHUS F. & WINTER J. (2003). Effective preprocessing with hyper-resolution and equality reduction. In *SAT'03*, volume 2919, p. 341–355.

BEAME P., KAUTZ H. & SABHARWAL A. (2003). Understanding the power of clause learning. In *IJCAI'03*, p. 1194–1201.

BOUFKHAD Y. & ROUSSEL O. (2000). Redundancy in random SAT formulas. In *AAAI'00*, p. 273–278.

Compétitions SAT (2008). http ://www.satcompetition.org.

EÉN N. & SORENSSON N. (2008). Minisat home page http ://www.cs.chalmers.se/cs/research/formalmethods/minisat.

FOURDRINOY O., GRÉGOIRE E., MAZURE B. & SAIS L. (2007). Eliminating redundant clauses in SAT instances. In *CP-AI-OR'07*, p. 71–83.

GOLDBERG E. P. & NOVIKOV Y. (2002). Berkmin : a fast and robust SAT-solver. In *DATE'02*, p. 142–149.

GRÉGOIRE E., MAZURE B. & PIETTE C. (2007). Local-search extraction of MUSes. *Constraints Journal*, **12**(3), 325–344.

HAMMER P. L. & KOGAN A. (1993). Optimal compression of propositional horn knowledge bases : Complexity and approximation. *Artificial Intelligence*, **64**(1), 131–145.

LIBERATORE P. (2005). Redundancy in logic I : CNF propositional formulae. *Artificial Intelligence*, **163**(2), 203–232.

LIBERATORE P. (2008). Redundancy in logic II : 2CNF and horn propositional formulae. *Artificial Intelligence*, **172**(2–3), 265–299.

MEYER A. R. & STOCKMEYER L. J. (1972). The equivalence problem for regular expressions with squaring requires exponential space. In *FOCS'72*, p. 125–129.

ZENG H. & MCILRAITH S. A. (2005). The role of redundant clauses in solving satisfiability problems. In *CP'05*, p. 873.

Fusion par R-ensembles : une méthode complète de fusion

Julien Hué[1], Odile Papini[2], Eric Würbel[1]

[1] LSIS UMR CNRS 6168 Université du Sud-Toulon Var, Av. de l'université BP20132, 83957 La Garde Cedex {hue,wurbel}@univ-tln.fr

[2] LSIS UMR CNRS 6168 Université de la méditerranée, 163 avenue de Luminy, 13288 Marseille Cedex 9. odile.papini@esil.univmed.fr

Résumé : La fusion d'informations provenant de sources multiples est un problème important en intelligence artificielle. La plupart des propositions ont été faites dans le cadre sémantique qui présente une très grande complexité. De plus, peu d'implantations sont disponibles. Ce papier propose un cadre pour réaliser la fusion de bases de croyances propositionnelles. Puis, il décrit une implantation basée sur la programmation logique avec sémantique des modèles stables qui peut prendre en compte n'importe quelle formule bien formée et s'exécuter indépendamment du solveur. Finalement, il présente une étude expérimentale. [1]

1 Introduction

La disponibilité des sources d'informations distribuées ne cesse d'augmenter. Mais celles-ci manquent souvent de structure, elles peuvent être conflictuelles ou redondantes, ce qui rend leur exploitation difficile. La fusion a pour objectif d'obtenir un point de vue global, exploitant la complémentarité des sources, résolvant les conflits existants et supprimant les répétitions. Parmi les nombreuses approches de fusion, les approches logiques ont obtenu un intérêt croissant, spécifiquement dans le cadre propositionnel (Cholvy, 1998), (Revesz, 1997), (Baral *et al.*, 1991). Plusieurs postulats permettant de caractériser le comportement d'un opérateur de fusion rationnel ont été proposés ainsi que de nombreux opérateurs (Konieczny & PinoPérez, 1999). Plus récemment, des approches sémantiques se basant sur la distance de Hamming (Konieczny, 2000) ou sur des propriétés morphologiques comme la dilatation ou l'érosion ont été définies (Bloch & Lang, 2002). Une implantation basée sur les BDD a récemment été proposée et une expérimentation sur des opérateurs sémantiques a été conduite (Gorogiannis & Hunter, 2008).

[1]Ce travail a été réalisé avec le soutien de la Communauté européenne au sein du projet VENUS (IST-034924) du 6ème programme cadre pour la recherche et le développement (FP6) (Société, Information et technologie)(IST). Les auteurs sont seuls responsables du contenu de ce papier qui ne reflète pas l'opinion de la Communauté européenne. De plus, la Communauté européenne n'est pas responsable de l'utilisation des données apparaissant dans l'article.

Dans (Meyer *et al.*, 2001), des approches syntaxiques ont été proposées pour la logique propositionnelle. Nous avons récemment proposé une approche syntaxique appelée fusion par R-ensembles (ou RSF) pour la fusion de bases de croyances propositionnelles (Hue *et al.*, 2007). Suite à une expérimentation précédente (Bennaim *et al.*, 2004), nous avons choisi de mettre en œuvre cette approche dans le cadre de la programmation logique avec sémantique des modèles stables (ou ASP) et l'algorithme Smodels a été adapté afin de fournir une implantation.

Nous proposons ici de généraliser RSF afin que la méthode puisse gérer n'importe quelle formule bien formée. Nous montrons que les opérateurs classiques de la fusion peuvent être capturés. Nous proposons également une implantation indépendante du solveur. Ainsi, notre principale contribution est la suivante :

- La notion de R-ensembles, pour simplifier les sous-ensembles de clauses causant l'incohérence, est étendue aux sous-ensembles de formules. Les opérateurs de fusion classiques sont capturés en les traduisant en préférences entre R-ensembles.
- Une implantation de RSF en programmation logique avec sémantique des modèles stables (ASP). Le problème de fusion est traduit en ASP et l'équivalence entre les modèles stables préférés du programme et les R-ensembles est démontrée.
- Une étude expérimentale qui illustre le comportement de RSF pour les stratégies Σ et Max est menée et compare RSF à la méthode de Gorogiannis et Hunter.

Le papier s'articule comme suit. La section 2 fixe les notations. La section 3 présente la généralisation de RSF à toute formule bien formée. L'implantation de la méthode en ASP est décrite dans la section 4 ainsi que les propriétés essentielles. La section 5 montre comment RSF peut être mise en œuvre indépendamment du solveur. Enfin, les résultats de l'étude expérimentale sont présentés dans la section 6.

2 Préliminaires

2.1 Notations

Nous considérons un langage $\mathcal{L}$ sur un alphabet fini $\mathcal{P}$ constitué d'atomes propositionnels. Une interprétation est une fonction de $\mathcal{P}$ vers $\{0, 1\}$ et l'ensemble de toutes les interprétations est noté $\mathcal{W}$. Pour toute interprétation I et toute formule A, on dit que I implique A ($I \models A$) ssi A est vraie pour I. Si A et B sont deux formules, alors $A \models B$ ssi $I \models A$ implique $I \models B$. $mod(A)$ représente l'ensemble des modèles de A.

Une base de croyances K est un ensemble fini de formules propositionnelles. Elle peut être vue comme une formule unique qui est la conjonction de ses formules. Soient $K_1, \ldots, K_n$ n bases de croyances qui ne sont pas forcément différentes deux à deux. Nous appelons *profil de croyances* le multi-ensemble E constitué de tous les $K_{i\ (1 \leq i \leq n)}$. L'ensemble des atomes apparaissant dans E est noté $Atome(E)$. L'union sur les multi-ensemble est représentée par $\sqcup$ et $\sqsubseteq$ dénote l'inclusion multi-ensembliste. On note E^n le multi-ensemble constitué de n répétitions de E. De plus, la conjonction (resp. disjonction) est notée $\bigwedge$ (resp. $\bigvee$). Pour simplifier, nous notons K le profil de croyances $E = \{K\}$. Pour le reste de cette section, nous supposerons que $\forall i$, la base de croyances K_i est cohérente. Nous considérons qu'un profil de croyances $E = \{K_1, \ldots, K_n\}$ est cohérent ssi il existe au moins une interprétation I telle que $\forall i, I \models K_i$. Soient

$E_1 = \{K_{1,1}, \ldots, K_{1,n_1}\}$ et $E_2 = \{K_{2,1}, \ldots, K_{2,n_2}\}$ deux profils de croyances, E_1 et E_2 sont dits équivalents, noté $E_1 \leftrightarrow E_2$, ssi il existe une bijection f de E_1 vers E_2 telle que $\models f(K_{1,i}) \leftrightarrow K_{2,j}$.

Un pré-ordre sur $\mathcal{W}$ est une relation transitive et réflexive sur les éléments de $\mathcal{W}$. Un pré-ordre est total ssi $\forall I, \forall J$ on a soit $I \leq J$, soit $J \leq I$. On définit $<$ comme suit : $I < J$ ssi $I \leq J$ et $J \not\leq I$. On définit également $=$ par : $I = J$ ssi $I \leq J$ et $J \leq I$. Une interprétation I appartient aux modèles minimaux de K par rapport à $\leq$ (noté $I \in min(mod(K), \leq)$) ssi $I \models K$ et $\forall J \in mod(K), I \leq J$.

Une opération de fusion syntaxique Δ est définie comme une fonction qui associe à tout profil de croyances E une base de croyances cohérente notée $\Delta(E)$.

2.2 La programmation logique avec sémantique des modèles stables

Un *programme logique normal* est un ensemble de règles de la forme $c \leftarrow a_1, \ldots, a_n, not\ b_1, \ldots, not\ b_m$ où $c, a_i (1 \leq i \leq n), b_j (1 \leq j \leq m)$ sont des atomes de la logique propositionnelle et le symbole *not* représente la négation par échec. Soit r une règle, on introduit $head(r) = c$ et $body(r) = \{a_1, \cdots, a_n, b_1, \cdots, b_m\}$. De plus, $body^+(r) = \{a_1, \cdots, a_n\}$ représente l'ensemble des éléments positifs du corps de la règle et $body^-(r) = \{b_1, \cdots, b_m\}$ l'ensemble de ses éléments négatifs. r^+ représente la règle $head(r) \leftarrow body^+(r)$ obtenue à partir de r. Un *programme basique* est un programme logique ne comportant pas de négation par échec.

Un ensemble d'atomes X est *clos sous* un programme basique Π ssi pour n'importe quelle règle $r \in \Pi$, $head(r) \in X$ lorsque $body(r) \subseteq X$. Le plus petit ensemble d'atomes qui est clos sous un programme basique Π est noté $CN(\Pi)$. La *réduction* (Gelfond & Lifschitz, 1988), Π^X d'un programme Π relativement à un ensemble X d'atomes est définie par $\Pi^X = \{r^+ \mid r \in \Pi \text{ et } body^-(r) \cap X = \emptyset\}$. Un ensemble d'atomes X est un *modèle stable* de Π ssi $CN(\Pi^X) = X$.

Ces dix dernières années, la programmation logique avec sémantique des modèles stables (ASP) a été considérée comme étant un outil pratique pour le raisonnement non-monotone et certains solveurs ASP se sont montrés efficaces : Smodels (Niemelä & Simons, 1997), NoMore (Anger *et al.*, 2002), CLASP (Baral *et al.*, 2007).

Afin d'améliorer son expressivité, l'ASP a été étendue avec de nouvelles règles et instructions : les *définitions de domaines* permettent de coder de manière compacte les valeurs possibles d'un domaine. Par exemple, les règles $\#domain\ possible(X).\ possible(1 \ldots n)$ remplacent chaque apparition de la variable X par n règles où X est remplacé par l'ensemble des valeurs de 1 à n. Les *restrictions de domaines* : par exemple, la règle $short(X) \leftarrow size(Y), X < Y$ spécifie que la règle ne peut s'appliquer (et donc n'est instancié) que pour les valeurs de X et Y tel que $X < Y$. Les *contraintes de cardinalité* permettent d'exprimer qu'au moins (resp. au plus) un certain nombre d'atomes d'un ensemble doivent apparaître. Par exemple, $h \leftarrow k\{a_1, \ldots, a_n\}l$ exprime qu'au moins k atomes et au plus l atomes parmi les $a_{i(1 \leq i \leq n)}$ doivent être vrais pour que h soit vrai. Les *instructions d'optimisation* permettent de sélectionner parmi les modèles stables seulement ceux qui contiennent le moins possible ou le plus possible d'atomes parmi un ensemble donné. Par exemple, l'instruction $minimize\{a_1, \ldots, a_n\}$

permet de ne sélectionner parmi les modèles stables que ceux qui contiennent un nombre minimaux d'atomes parmi les $a_{i(1\leq i\leq n)}$. L'inverse s'effectue au travers de l'instruction $maximize\{\}$.

3 La fusion par R-ensembles

L'idée principale consiste à identifier, puis retirer, un sous-ensemble adéquat de formules afin de restaurer la cohérence dans l'union des bases de croyances. Dans un premier temps, l'ensemble des sous-ensembles de formules est considéré avant d'y effectuer une sélection grâce à l'une des stratégies que nous décrirons *infra*.

Définition 1
Soit $E = \{K_1, \ldots, K_n\}$ un profil de croyances tel que $K_1 \sqcup \ldots \sqcup K_n$ est incohérent. Soit X un sous-ensemble de formules de $K_1 \sqcup \ldots \sqcup K_n$. X est un R-ensemble potentiel *de E ssi $(K_1 \sqcup \ldots \sqcup K_n)\backslash X$ est cohérent.*

Le nombre de R-ensembles potentiels est exponentiel par rapport au nombre de formules. Il faut donc faire un choix parmi ceux-ci et ne garder que les plus pertinents. Cependant la notion de minimalité dans le cadre de la fusion n'est pas unique et plusieurs stratégies existent qui permettent d'obtenir de bonnes solutions selon le critère de minimalité choisi. Soient X et Y deux R-ensembles potentiels. Pour chaque stratégie P, un pré-ordre total $\leq_P$ sur les R-ensembles potentiels est défini. $X \leq_P Y$ signifie que X est préféré à Y selon la stratégie P. On définit $<_P$ comme étant le pré-ordre strict associé à $\leq_P$ (i.e. $X <_P Y$ ssi $X \leq_P Y$ et $Y \not\leq_P X$).

Définition 2
Soit $E = \{K_1, \ldots, K_n\}$ un profil de croyances tel que $K_1 \sqcup \ldots \sqcup K_n$ est incohérent. Soit P une stratégie de fusion. $X \subseteq K_1 \sqcup \ldots \sqcup K_n$ est un R-ensemble *de E selon P ssi (1) X est un R-ensemble potentiel de E ; (2) Il n'existe pas de Y qui soit un R-ensemble potentiel de E tel que $Y <_P X$.*

La collection des R-ensembles de E selon la stratégie P est notée par $\mathcal{F}_P\mathcal{R}(E)$. La fusion par R-ensembles est définie par :

Définition 3
Soit $E = \{K_1, \ldots, K_n\}$ un profil de croyances. L'opération de fusion $\Delta^P(E)$ est définie par : $\Delta^P(E) = \bigvee_{X \in \mathcal{F}_P\mathcal{R}(E)} \{(K_1 \sqcup \ldots \sqcup K_n)\backslash X\}$

L'opportunité de l'union des bases pourrait être discutée. Ce choix peut permettre de définir, dans un second temps, une sélection (partial meet, full meet, ...) sur l'union. Les opérateurs de fusion classiques (Σ, Max, $Gmax$) sont capturés dans notre cadre.

3.1 Différents opérateurs capturés par RSF

Dans un premier temps, nous allons donner les définitions formelles des pré-ordres associés à ces opérateurs avant, dans un second temps, de donner un exemple permettant d'illustrer leur comportement.

Définition 4
Soient X et Y deux R-ensembles potentiels de E. Les pré-ordres selon les différentes stratégies sont :

Σ *:* $X \leq_\Sigma Y$ *ssi* $\sum_{1\leq i\leq n} |X \cap K_i| \leq \sum_{1\leq i\leq n} |Y \cap K_i|$.

$Card$ *:* $X \leq_{Card} Y$ *ssi* $|X \cap s(E)| \leq |Y \cap s(E)|$. [2]

Max *:* $X \leq_{Max} Y$ *ssi* $\max_{1\leq i\leq n} |X \cap K_i| \leq \max_{1\leq i\leq n} |Y \cap K_i|$.

$Gmax$ *: Pour tout R-ensemble potentiel X et chaque base de croyances K_i, on définit $p_X^{K_i} = |X \cap K_i|$. Soit L_X^E la séquence $(p_X^{K_1}, \ldots, p_X^{K_n})$ rangée dans l'ordre décroissant.* $X \leq_{Gmax} Y$ *ssi* $L_X^E \leq_{lex} L_Y^E$.

La stratégie Σ minimise le nombre de formules à retirer de E. Elle correspond à l'opérateur *intersection* défini dans (Konieczny & PinoPérez, 1999) . La stratégie $Card$ tente, tout comme Σ, de minimiser le nombre de formules retirées. Cependant, elle ne tient pas compte des formules exprimées plusieurs fois. La stratégie Max essaye de répartir au mieux les formules à retirer parmi les bases de croyances. Elle tente cela en retirant le moins possible de formules dans la base de croyances où elle en retire le plus. La stratégie $Gmax$ est un raffinement lexicographique de la stratégie Max.

Exemple 1
On considère le profil de croyances suivant $E = \{K_1, K_2, K_3\}$ avec $K_1 = \{\neg d, s, s \vee o\}$, $K_2 = \{\neg s, d \vee o, \neg(d \wedge o)\}$ et $K_3 = \{s, d, o\}$. Les R-ensembles préférés de E stratégie par stratégie, sont :

$\mathcal{W}$	K_1		K_2		K_3		Σ	$Card$	Max	$Gmax$
(000)	$s \vee o, s$	2	$d \vee o$	1	s, d, o	3	6	5	3	321
(001)	s	1		0	s, d	2	3	2	2	210
(010)	$s \vee o, \neg d, s$	3		0	s, o	2	5	4	3	320
(011)	$\neg d, s$	2	$\neg(d \wedge o)$	1	s	1	4	3	2	211
(100)		0	$d \vee o, \neg s$	2	d, o	2	4	4	2	220
(101)		0	$\neg s$	1	d	1	2	2	1	110
(110)	$\neg d$	1	$\neg s$	1	o	1	3	3	1	111
(111)	$\neg d$	1	$\neg(d \wedge o), \neg s$	2		0	3	3	2	210

Ainsi, les R-ensembles et le résultat de la fusion des bases de E, pour chaque stratégie, seront :

Σ : $\mathcal{F}_\Sigma\mathcal{R}(E) = \{\{\neg s, d\}\}$; $\Delta^\Sigma(E) = \{\{\neg d, s \vee o, s, d \vee o, \neg(d \wedge o), o\}\}$.

$Card$: $\mathcal{F}_{Card}\mathcal{R}(E) = \{\{\neg s, d\}, \{s, s, d\}\}$;$\Delta^{Card}(E) = \{\{\neg d, s \vee o, s, d \vee o, \neg(d \wedge o), o\}, \{\neg d, s \vee o, \neg s, d \vee o, \neg(d \wedge o), o\}\}$.

Max : $\mathcal{F}_{Max}\mathcal{R}(E) = \{\{\neg d, \neg s, o\}, \{\neg s, d\}\}$; $\Delta^{Max}(E) = \{\{s \vee o, s, d \vee o, \neg(d \wedge o), d\}, \{\neg d, s \vee o, s, d \vee o, \neg(d \wedge o), o\}\}$.

$Gmax$: $\mathcal{F}_{Gmax}\mathcal{R}(E) = \{\{\neg s, d\}\}$; $\Delta^{Gmax}(E) = \{\{\neg d, s \vee o, s, d \vee o, \neg(d \wedge o), o)\}\}$.

Nous decrivons maintenant comment effectuer un calcul efficace des R-ensembles.

4 Mise en œuvre de la méthode RSF

Cette section présente une mise en œuvre de l'opération de fusion syntaxique. Soit $E = \{K_1, \ldots, K_n\}$ un profil de croyances et P une stratégie de fusion, ce problème est traduit en un programme logique Π_E^P dont les solutions permettent de determiner

[2] Soit E un profil de croyances, on note $s(E)$ la base de croyances créée à partir de E où toutes les formules apparaissant plusieurs fois sont réduites à un singleton.

les R-ensembles. Dans un premier temps, nous construisons un programme Π_E dont les solutions correspondent aux R-ensembles potentiels. Puis, nous introduisons des atomes représentant chaque règle permettant de comparer les R-ensembles potentiels. Enfin, nous ajoutons des instructions permettant de sélectionner les R-ensembles.

4.1 Générer les R-ensembles potentiels

Soient $f, f^1, \ldots, f^n$ des formules de K_i. L'ensemble des littéraux positifs (resp. négatifs) de Π_E est noté V^+ (resp. V^-). Soit $f \in K_i$ une formule, on note r_f^i l'atome de règle qui correspond à f. L'ensemble de tous les atomes représentant les formules, appelés atomes de règle, est défini par $R^+ = \{r_f^i \mid f \in K_i\}$. La traduction du problème nécessite la création d'atomes intermédiaires représentant les sous-formules de f. On note ρ_f^j l'atome intermédiaire représentant f^j qui est une sous-formule de $f \in K_i$. De plus, on note $F_O(r_f^i)$ la formule de K_i correspondant à r_f^i dans Π_E. En d'autres termes $\forall r_f^i \in R^+, F_O(r_f^i) = f$. La création de Π_E se déroule en deux étapes :

(i) Pour tout atome $a \in V^+$, la première étape introduit les règles $a \leftarrow not\ a'$ et $a' \leftarrow not\ a$ où a' représente la négation de a. On construit ainsi une correspondance entre les interprétations de V^+ et les modèles stables de Π_E. **(ii)** La seconde étape introduit les atomes de règle qui permettent de déterminer les R-ensembles potentiels associés aux interprétations. La présence de l'atome de règle r_f^i dans le modèle stable S signifie que f est dans le R-ensemble potentiel correspondant. Quelle que soit la formule f, les règles suivantes sont introduites selon la syntaxe de f : **(1)** Si $f \equiv a$, la règle correpondante est $r_f^i \leftarrow$ not a ; **(2)** Si $f \equiv \neg f^1$, la règle correpondante est $r_f^i \leftarrow$ not ρ_{f^1} ; **(3)** Si $f \equiv f^1 \vee \ldots \vee f^m$, la règle correpondante est $r_f^i \leftarrow \rho_{f^1}, \ldots, \rho_{f^m}$; **(4)** Si $f \equiv f^1 \wedge \ldots \wedge f^m$, il est alors nécessaire de rajouter plusieurs règles au programme. Les règles correpondantes sont $\forall 1 \leq j \leq m$, $r_f^i \leftarrow \rho_{f^j}$.

Exemple 2
Soit $E = \{K_1, K_2, K_3\}$ le profil de croyances constitué des bases $K_1 = \{a \vee b, b\}$, $K_2 = \{a \leftrightarrow b, b\}$ et $K_3 = \{\neg a \wedge \neg b, \neg a \vee \neg b\}$. Le programme logique correspondant à E, Π_E, est le suivant :
$\Pi_E = \{a \leftarrow not\ a'.\ \ a' \leftarrow not\ a.\ \ b \leftarrow not\ b'.\ \ b' \leftarrow not\ b.\ \ r^1_{a \vee b} \leftarrow not\ a, not\ b.\ \ r^1_b \leftarrow not\ b.\ r^2_{a \leftrightarrow b} \leftarrow \rho^2_1.\ \ r^2_{a \leftrightarrow b} \leftarrow \rho^2_2.\ \ \rho^2_1 \leftarrow a, not\ b.\ \rho^2_2 \leftarrow not\ a, b.\ \ r^2_b \leftarrow not\ b.\ \ r^3_{\neg a \wedge \neg b} \leftarrow \rho^3_1.\ \ r^3_{\neg a \wedge \neg b} \leftarrow \rho^3_2.\ \ \rho^3_1 \leftarrow a.\ \ \rho^3_2 \leftarrow b.\ \ r^3_{\neg a \vee \neg b} \leftarrow a, b.\}$

4.2 Sélectionner les R-ensembles

4.2.1 Sélection *off the shelf*

Afin de choisir parmi ces modèles stables ceux qui correspondent aux R-ensembles selon la stratégie P, nous introduisons la notion de modèles stables préférés selon P. On note I_X comme étant : $I_X = \{a \mid a \in X\} \cup \{\neg a \mid a' \in X\}$.

Définition 5
Soient Π_E un programme logique et X et Y deux ensembles d'atomes de Π_E. X est un modèle stable préféré de Π_E selon la stratégie P ssi (1) X est un modèle stable de Π_E ; (2) $\forall Y$ modèle stable de Π_E, Y n'est pas préféré à X selon P.

Proposition 1
Soit ρ un atome de règle ou un atome intermédiaire. $\rho \in CN(\Pi_E^X)$ ssi $I_X \not\models F_O(\rho)$.

Proposition 2
Soit $E = \{K_1, \ldots, K_n\}$ un profil de croyances. Soit X un ensemble d'atomes. X est un modèle stable de Π_E ssi X correspond à une interprétation I_X de V^+ qui satisfait $(K_1 \sqcup \ldots \sqcup K_n) \backslash F_O(X \cap R^+)$.

Les résultats suivants établissent une correspondance entre les modèles stables préférés de Π_E et les R-ensembles de E selon la stratégie employée.

Proposition 3
Soit $X \subseteq (K_1 \sqcup \ldots \sqcup K_n)$ tel que $(K_1 \sqcup \ldots \sqcup K_n) \backslash X$ est cohérent. Alors, il existe un ensemble d'atomes S tel que S est un modèle stable de Π_E et $F_O(S \cap R^+) \subseteq X$

Proposition 4
Soit $X \subseteq (K_1 \sqcup \ldots \sqcup K_n)$ un ensemble d'atomes. X est un R-ensemble de E selon Σ, $Card$ et Max. ssi il existe un modèle stable préféré de Π_E tel que $F_O(S \cap R^+) = X$.

Nous détaillons maintenant l'implantation qui nous permet de sélectionner les R-ensembles.

4.2.2 L'implantation

Une méthode pour réaliser la fusion par R-ensembles (RSF) à l'aide de n'importe quel solveur ASP consiste à rajouter au programme Π_E, des règles et des instructions dont le but est de compter les atomes de règle présents dans chaque modèle stable.

Pour la stratégie Σ, le principe est relativement simple puisqu'il suffit de compter le nombre total d'atomes de règle et de sélectionner les R-ensembles en contenant le moins. Cette sélection est réalisée par l'intermédiaire de l'instruction $minimize$. Ainsi, $\Pi_E^\Sigma = minimize\ \{r_f^i \mid r_f^i \in R^+\} \cup \Pi_E$. La méthode employée pour la stratégie $Card$ est simlaire au détail près que les doublons sont préalablement enlevés du profil. Ainsi $\Pi_E^{Card} = minimize\ \{r_f^i \mid r_f^i \in R^+\} \cup \Pi_{s(E)}$ La proposition suivante est vérifiée :

Proposition 5
L'ensemble de modèles stables de Π_E^Σ (resp. Π_E^Σ) est l'ensemble des modèles stables préférés de Π_E selon la stratégie Σ (resp. $Card$).

Exemple 3
Afin d'illustrer les définitions, nous reprenons l'exemple 2 et, stratégie par stratégie, montrons la traduction du problème. Pour la stratégie Σ (resp. $Card$) $\Pi_E^\Sigma = \Pi_E \cup$ (resp. $\Pi_{s(E)} \cup$) $minimize\{r_b^1, r^2_{a \leftrightarrow b}, r_b^2, r^3_{\neg a \wedge \neg b}, r^3_{\neg a \vee \neg b}\}$. La formule b qui était présente deux fois dans E n'apparaît qu'une seule fois dans $s(E)$.

La stratégie Max requiert une autre utilisation de l'instruction $minimize$. Soit X un modèle stable. Une première étape calcule $max_{1 \leq i \leq n}(|\{r_f^i \mid r_f^i \in X\}|)$ pour chaque X. La seconde étape utilise ce calcul et l'instruction $minimize$ pour conserver les modèles stables minimaux au sens de Max. Pour savoir combien de formules sont retirées dans la base où il y a le plus de retraits, il est nécessaire de commencer par

calculer combien de formules sont retirées dans chaque base de croyances. Ce calcul est effectué par :

$\Pi_E^{max,size} = \{\delta_0 : \#domain\ possible(U).\ \ \delta_1 : \#domain\ base(V).\ \ \delta_2 : possible(1..m).\ \delta_3 : base(1..n).\ \ \alpha : size(U) \leftarrow U\{r_f^V | F_0(f) \in \varphi_V\}U.\}$

où m est la taille de la plus grande base de croyances. Les règles δ_i sont les domaines de définition pour la règle α. S'il existe une base φ_V où le nombre d'atomes r_f^V vrais dans X est égal à U, alors $size(U)$ sera vrai dans X. La plus grande valeur U pour laquelle $size(U)$ est vrai est calculée grâce à $\Pi_E^{max,bound}$:

$\Pi_E^{max,bound} = \{\delta_4 : \#domain\ possible(W).\ \ \beta_1 : negmax(W) \leftarrow size(U), U > W.\ \beta_2 : max(U) \leftarrow size(U), not\ negmax(U).\ \ \}$

β_1 determine tous les entiers W pour lesquelles il existe $U > W$ tel que $size(U)$ est vrai. Alors, $max(U)$ est vrai pour la plus grande valeur de U telle que $size(U)$ est vrai. Pour la stratégie Max , soit $\Pi_E^{Max} = \Pi_E^{max,size} \cup \Pi_E^{max,bound} \cup \Pi_E \cup minimize[max(1) = 1, \ldots, max(m) = m]$. La proposition suivante est vérifiée :

Proposition 6
L'ensemble des R-ensembles de Π_Ψ^{Max} *est l'ensemble des modèles stables préférés de* Π_E *selon* Max.

Exemple 4
$\Pi_E^{Max} = \Pi_E \cup \{\#domain\ possible(U).\ \ negmax(W) \leftarrow size(U), U > W.\ \ \#domain\ base(V).\ size(U) \leftarrow U\ \{r_b^1\}\ U.\ \ \#domain\ possible(W).\ \ size(U) \leftarrow U\ \{r_{a \leftrightarrow b}^2, r_b^2\}\ U.\ \ base(1..3).\ size(U) \leftarrow U\ \{r_{\neg a \wedge \neg b}^3, r_{\neg a \vee \neg b}^3\}\ U.\ \ possible(1..2).\ \ max(U) \leftarrow size(U), not\ negmax(U).\ minimize[max(1) = 1, max(2) = 2].\}$

La stratégie $Gmax$ compare les R-ensembles potentiels d'après la séquence des $|X \cap K_i|$ rangée dans l'ordre décroissant. Comme pour Max, il faut connaître le nombre de formules retirées dans chacune des bases. Ce calcul est représenté par la règle suivante où m représente la taille de la base φ_V.

Si $size(V, U)$ est vrai, dans la base φ_V, le modèle stable en cours contient U atomes de règle. Une fois calculées pour toutes les bases K_V, ces valeurs sont ordonnées par $\Pi_E^{gmax,bound}$ où X_1 est la plus grande valeur de $size()$ et X_n la plus petite. Finalement, le polynôme $X_n + X_{n-1} \times m + \ldots + X_1 \times m^{n-1}$ est optimisé.

$\Pi_E^{gmax,size} = \{\gamma_1 : size(V, U) \leftarrow U\ \{rem(V, 1), \ldots, rem(V, m)\}\ U.\}\ \ \Pi_E^{gmax,bound} = \{\alpha_0 : max(X_1, \ldots, X_n) \leftarrow size(Y_1, X_1), \ldots, size(Y_n, X_n), X_1 \geq X_2, \ldots, X_{n-1} \geq X_n, neq(Y_1, \ldots, Y_n).\ \alpha_1 : max_1(X_1) \leftarrow max(X_1, \ldots, X_n), X_1 \geq X_2, \ldots, X_{n-1} \geq X_n.\ \ \alpha_i : \ldots\ \ \alpha_n : max_3(X_3) \leftarrow max(X_1, \ldots, X_n), X_1 \geq X_2, \ldots, X_{n-1} \geq X_n.\}$

Pour la stratégie $Gmax$, soit $\Pi_E^{Gmax} = \Pi_E^{gmax,size} \cup \Pi_E^{gmax,bound} \cup \Pi_E \cup minimize[\ max_n(1) = 1, max_n(2) = 2, \ldots, max_i(1) = m^{n-i}, max_i(2) = 2 \times m^{n-i}, \ldots, max_1(n) = n \times m^{n-1}\ ]$. La proposition suivante est vérifiée :

Proposition 7
L'ensemble des R-ensembles de Π_E^{Gmax} *est l'ensemble des modèles stables préférés de* Π_E *pour la stratégie* $Gmax$.

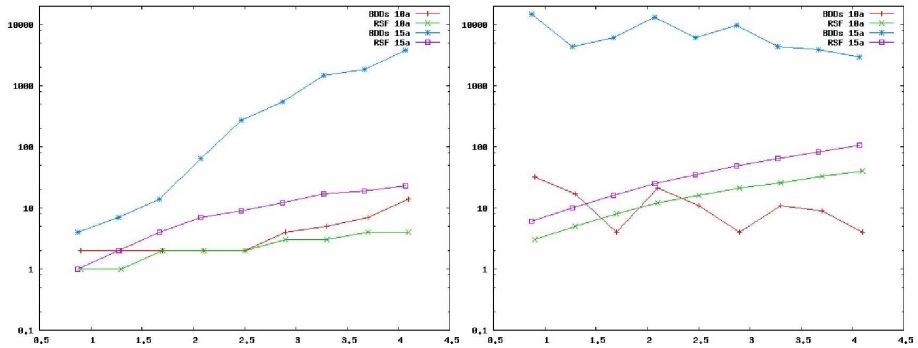
Exemple 5
$\Pi_E^{Gmax} = \Pi_E \cup \{\#domain\ possible(U).\ \ possible(1..2).\ \ \#domain\ base(V).\ \ base(1..3).\ \#domain\ possible(X_1).\ \ size(1, U) \leftarrow U\ \{r_b^1\}\ U.\ \ \#domain\ possible(X_2).\ \ size(2, U) \leftarrow$

$U\ \{r^2_{a\leftrightarrow b}, r^2_b\}\ U.\quad \#domain\ possible(X_3).\quad size(3,U) \leftarrow U\ \{r^3_{\neg a\wedge\neg b}, r^3_{\neg a\vee\neg b}\}\ U.$
$\#domain\ base(Y_1).\quad max_1(X_1) \leftarrow max(X_1,X_2,X_3).\quad \#domain\ base(Y_2).\quad max_2(X_2) \leftarrow$
$max(X_1,X_2,X_3).\quad \#domain\ base(Y_3).\quad max_3(X_3) \leftarrow max(X_1,X_2,X_3).\quad max(X_1,X_2,X_3) \leftarrow$
$size(Y_1,X_1), size(Y_2,X_2), size(Y_3,X_3), Y_1 != Y_2, Y_1 != Y_3, Y_2 != Y_3, X_1 >= X_2, X_2 >= X_3.\}$

5 Etude expérimentale et comparaison

Nous présentons maintenant les résultats de l'étude expérimentale pour les stratégies Max et Σ et nous comparons les approches implantées en ASP avec les approches sémantiques avec BDD présentées dans (Gorogiannis & Hunter, 2008). Les tests ont été menés sur un Centrino Duo cadencé à 1.73GHz et equipé de 2Go de RAM.

FIG. 1: Tests pour les stratégies Max et Σ



(a) La stratégie Σ (b) La stratégie Max

Nous utilisons un protocole proche de celui décrit dans (Gorogiannis & Hunter, 2008). Chaque profil de croyances posséde 7 bases de croyances. Les bases de croyances sont générées aléatoirement, selon le nombre d'atomes (na) et le nombre de formules (nf) souhaitées. Les bases contiennent des formules de tout type dont la profondeur ne dépasse pas 3. Les tests ont été effectués pour les stratégies Max et Σ.

La figure 1 montre le temps d'exécution en fonction du ratio $(nf)/(na)$. Les temps d'exécution sont donnés en centièmes de seconde. Les courbes ont été tracées pour 10 et 15 atomes pour les deux stratégies. Sur les deux stratégies, les temps d'execution pour les BDD sont très instables, ce qui explique la forme du graphe.

Ce qui ressort de cette étude est que RSF obtient de meilleurs résultats lorsque le nombre d'atomes augmente. D'un autre coté, les BDD obtiennent de meilleurs résultats lorsque le nombre de formules augmente. Néanmoins, la différence fondamentale entre ces deux approches réduit la valeur de cette comparaison. Malgré cette différence, cette étude a le mérite de pointer les difficultés propres aux deux approches.

6 Conclusion

Ce papier généralise l'approche RSF aux formules quelconques et montre que les opérateurs classiques de fusion peuvent être exprimés dans ce cadre. Ce papier propose une implantation en ASP indépendante du solveur. Enfin, une expérimentation a été conduite et les résultats ont été comparés avec l'approche (Gorogiannis & Hunter, 2008).

Une expérimentation plus large doit être menée sur une application réelle afin de donner une meilleur idée des performances de RSF. Cette expérimentation sera fait dans le cadre du projet européen VENUS dans le contexte de l'information archéologique. Un travail futur détaillera la caractérisation sémantique de RSF.

Références

ANGER C., KONCZAK K. & LINKE T. (2002). Nomore non-monotonic reasoning with logic programs. In *JELIA*, p. 521–524.

C. BARAL, G. BREWKA & J. SCHLIPF, Eds. (2007). *Proc. of the Ninth International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR07)*, volume 4483 of *Lecture Notes in Artificial Intelligence*.

BARAL C., KRAUS S., MINKER J. & SUBRAHMANIAN V. S. (1991). Combining knowledge bases consisting of first order theories. In *ISMIS*, p. 92–101.

BENNAIM J., BENFERHAT S., PAPINI O. & WÜRBEL E. (2004). An answer set programming encoding of prioritized removed sets revision : application to gis. In *JELIA*, p. 604–616.

BLOCH I. & LANG J. (2002). Towards mathematical morpho-logics. p. 367–380.

CHOLVY L. (1998). Reasoning about merging information. *Handbook of DRUMS*, **3**, 233–263.

GELFOND M. & LIFSCHITZ V. (1988). The stable model semantics for logic programming. In *Proc. of the 5th Int. Conf. on Log. Prog.*, p. 1070–1080.

GOROGIANNIS N. & HUNTER A. (2008). Implementing semantic merging operators using binary decision diagrams. *International Journal of Approximate Reasoning*, p. 234–251.

HUE J., PAPINI O. & WÜRBEL E. (2007). Syntactic propositional belief bases fusion with removed sets. *Proceedings of ECSQARU*, p. 66–77.

KONIECZNY S. (2000). On the difference between merging knowledge bases and combining them. In *KR*, p. 135–144.

KONIECZNY S. & PINOPÉREZ R. (1999). Merging with integrity constraints. *LNCS*, **1638**, 233–244.

MEYER T., GHOSE A. & CHOPRA S. (2001). Syntactic representations of semantic merging operations. In *PRICAI '02 : Proceedings of the 7th Pacific Rim International Conference on Artificial Intelligence*, p. 620.

NIEMELÄ I. & SIMONS P. (1997). An implementation of stable model and well-founded semantics for normal logic programs. In S. VERLAG, Ed., *Proc. of LPNMR'97*, volume 1265 of *LNAI*, p. 420–429.

REVESZ P. Z. (1997). On the semantics of arbitration. *Journal of Algebra and Computation*, **7**(2), 133–160.

Traiter les exceptions en ASP à partir d'une représentation compacte des informations

Stéphane NGOMA, Laurent GARCIA et Pascal NICOLAS

LERIA, UFR Sciences, Université d'Angers
2 boulevard Lavoisier, 49045 ANGERS CEDEX 01
{ngoma,garcia,pn}@info.univ-angers.fr

Résumé : L'ASP, utilisé dans la forme des programmes logiques normaux, est un cadre adéquat pour le raisonnement non-monotone dans la mesure où il offre un modèle formel valide ainsi que des systèmes opérationnels. Le problème est que la représentation des informations dans ce cadre n'est pas une opération aisée. C'est pourquoi nous proposons de générer automatiquement ces programmes logiques normaux à partir d'une représentation compacte des informations à base de programmes logiques définis. Pour cela, nous appliquons une méthode définie dans le cadre de la logique des défauts dont nous proposons une adaptation.

Mots-clés : ASP, exceptions, spécificité

1 Introduction

Delgrande et Schaub (Delgrande & Schaub, 1997) ont présenté une approche générale et automatique pour introduire les informations de spécificité dans les théories non monotones. Cette approche est illustrée, entre autres, pour la logique des défauts (Reiter, 1980) ; pour autant, aucun système opérationnel n'a été développé.

Pour le travail présenté ici, nous avons choisi un cadre où le modèle formel est correct et pour lequel il existe des systèmes opérationnels performants : l'*Answer Set Programming* (ASP) (qui peut être vu comme un sous-domaine de la logique des défauts). Les informations sont codées sous forme de règles, un processus d'inférence permet de faire du raisonnement ; certaines règles peuvent être bloquées (d'où la propriété de non monotonie) et permettent d'exprimer les exceptions.

Certaines définitions ont été proprosées par Delgrande et Schaub de manière indépendante du formalisme. Néanmoins, pour être mené à terme, le processus est lié à une représentation particulière. C'est pourquoi nous redéfinissons l'intégralité des définitions dans le cadre de l'ASP, ce qui permet d'associer le travail théorique à une mise en œuvre pratique.

D'un point de vue formel, on distingue deux aspects dans l'ASP : les programmes logiques normaux, qui cadrent bien à la problématique mais sont difficiles à écrire, et les programmes logiques définis, plus simples mais moins intéressants (pas de gestion des exceptions). Il convient donc d'écrire les informations sous forme de programme

logique défini, de le transformer automatiquement en un programme logique normal adéquat puis d'utiliser le moteur d'inférence des programmes logiques normaux pour raisonner.

Dans la suite du document, nous présentons formellement l'ASP dans la section 2, puis nous exposons les travaux sur lesquels nous nous appuyons dans la section 3, à savoir l'approche de Delgrande et Schaub. Ensuite, dans la section 4, nous abordons l'aspect théorique du travail effectué.

2 Answer Set Programming (ASP)

Cette section présente le formalisme de programmation logique non monotone que nous utilisons dans ce travail. Il existe plusieurs variantes autour de ce formalisme que l'on regroupe sous le paradigme de l'*Answer Set Programming* (ASP) et nous présentons celles qui nous sont nécessaires.

Un *programme logique défini* est un ensemble de règles de la forme :

$$b \leftarrow a_1, \ \ldots \ , a_n. \qquad (n \geq 0) \qquad .$$

Pour une telle règle r, $tête(r) = b$ est un atome appelé *tête* et $corps(r) = \{a_1, \ldots, a_n\}$ est un ensemble d'atomes appelé *corps*. La règle $b \leftarrow a.$ peut se lire : « si l'on peut prouver a, alors on conclut b ». Étant donné une règle r et un ensemble d'atomes A, on dit que r est *applicable* (ou *déclenchable*) dans A si $corps(r) \subseteq A$. Un ensemble d'atomes A est dit *clos* par rapport à un programme P si et seulement si pour toute règle r de P, si $corps(r) \subseteq A$, alors $tête(r) \in A$. On appelle $Cn(P)$, ou *modèle de Herbrand*, le plus petit ensemble d'atomes clos par rapport au programme P. Pour un programme P et un ensemble d'atomes A, l'opérateur T_P défini par :

$$T_P(A) = \{tête(r) \,|r \in P, corps(r) \subseteq A\}$$

calcule l'ensemble des atomes déductibles à partir de P et A. À partir de cet opérateur, on définit la suite : $T_P^0 = T_P(\emptyset)$ et $T_P^{k+1} = T_P(T_P^k), \forall k \geq 0$. $Cn(P)$ est le plus petit point fixe de T_P et $Cn(P) = \bigcup_{k \geq 0} T_P^k$. Ainsi, $Cn(P)$ contient tous les atomes que l'on peut déduire du programme P. On dit qu'un programme logique défini est *enraciné* s'il peut être ordonné selon la suite $\langle r_1, \ \ldots \ , r_n \rangle$ telle que $\forall i$, $1 \leq i \leq n$, r_i est applicable dans $\{tête(r_j) \mid 1 \leq j < i\}$.

Un *programme logique normal* est un ensemble de règles de la forme :

$$c \leftarrow a_1, \ \ldots \ , \ a_n, \ not \ b_1, \ \ldots \ , \ not \ b_m. \qquad (n \geq 0, \ m \geq 0) \qquad .$$

Comme précédemment, les a_i, b_j et c sont des atomes et pour une telle règle r on note $corps^+(r) = \{a_1, \ldots, a_n\}$ (respectivement $corps^-(r) = \{b_1, \ldots, b_m\}$) son corps *positif* (respectivement *négatif*). En outre, $r^+ = c \leftarrow a_1, \ \ldots \ , a_n.$ désigne la *projection positive* de la règle r. Intuitivement, la règle $c \leftarrow a, \ not \ b.$ peut se lire : « si l'on peut prouver a et si l'on ne peut pas prouver b, alors on peut conclure c ». Les « not » du corps négatif s'interprètent donc comme des *négations par défaut*. Soit une règle r et un ensemble d'atomes A. On dit que r est *applicable* (ou *déclenchable*) dans A si $corps^+(r) \subseteq A$ et $corps^-(r) \cap A = \emptyset$.

Définition 1
Le réduit d'un programme logique normal P par rapport à un ensemble d'atomes A est le programme logique défini : $P^A = \{r^+ \,|r \in P, corps^-(r) \cap A = \emptyset\}$.

La sémantique originelle (Gelfond & Lifschitz, 1988) des programmes logiques normaux est celle des modèles stables rappelée ci-après.

Définition 2
Soit P un programme logique normal et S un ensemble d'atomes. S est un modèle stable *de P si et seulement si $S = Cn(P^S)$.*

Pour des besoins de représentation des connaissances on peut être amené à utiliser conjointement la « vraie » négation (appelée aussi *négation forte*) et la négation par défaut au sein d'un même programme. Ceci est rendu possible par les *programmes logiques étendus* (Gelfond & Lifschitz, 1991), où l'on autorise les littéraux à la place des atomes dans l'écriture des règles. Mais si, comme c'est notre cas, on refuse les modèles inconsistants alors la sémantique associée à ces programmes est réductible à celle des modèles stables en prenant en compte les considérations suivantes :

- tout littéral $\neg x$ est codé par l'atome nx,
- on ajoute une règle $\bot \leftarrow x,\ nx.$ pour tout atome x,
- un modèle stable ne doit pas contenir le symbole $\bot$.

Les règles de type $\bot \leftarrow x,\ nx.$, parfois simplement notées $\leftarrow x,\ nx.$ (sans tête), sont appelées des *contraintes*. L'ensemble des contraintes d'un programme P est noté P_K. L'ajout des contraintes induit le comportement attendu, à savoir qu'il est impossible d'obtenir un modèle stable contenant à la fois x et nx. Ainsi, seuls les modèles stables consistants sont conservés. Cette manière de faire est celle implantée dans tous les solveurs disponibles qui sont donc capables de traiter les programmes logiques étendus tout en restant dans le cadre des programmes logiques normaux.

Exemple 2.1
L'exemple habituel des oiseaux (où O désigne « oiseau », A « avoir des ailes », M « manchot » et V « vole ») peut donc être codé de cette manière : $\{V \leftarrow O,\ not\ M.,\ A \leftarrow O.,\ O \leftarrow M.,\ nV \leftarrow M.,\ \bot \leftarrow V,\ nV.\}$. En présence d'un manchot (codé par l'ajout de $M \leftarrow .$), on obtient un seul modèle stable : $\{A,\ M,\ O,\ nV\}$, ce qui correspond à l'intuition.

Il est reconnu que les programmes logiques normaux sont adaptés à notre problématique de représentation de règles avec exceptions. Mais ils sont difficiles à écrire dans la mesure où il faut expliciter les exceptions ou tout au moins décrire les anormalités. Or on veut pouvoir donner une représentation simple des informations, ce que permettent les programmes logiques définis qui, par contre, peuvent mener à des incohérences car ils ne prennent pas en compte la notion d'exception. Notre travail consiste donc à résoudre ce problème en déterminant de manière automatique des classes et sous-classes d'information. La notion de *spécificité* (Pearl, 1990) prend alors tout son sens. Cet aspect ayant déjà été traité formellement pour la logique des défauts (Delgrande & Schaub, 1997), il nous reste à l'appliquer à l'ASP et à définir un processus automatique gérant correctement les exceptions. Ainsi, à partir d'informations présentées sous la forme simple suivante : $\{O \rightarrow V,\ O \rightarrow A,\ M \rightarrow O,\ M \rightarrow \neg V\}$, on veut obtenir automatiquement le programme logique normal adéquat : $\{V \leftarrow O,\ not\ nV,\ not\ M.,\ A \leftarrow O.,\ O \leftarrow M.,\ nV \leftarrow M,\ not\ V.,\ \bot \leftarrow V,\ nV.\}$.

3 Spécificité en logique des défauts

Nous présentons l'approche de Delgrande et Schaub (Delgrande & Schaub, 1997) proposée dans le cadre de la logique des défauts (Reiter, 1980). Elle se compose de deux étapes majeures. Premièrement, à partir d'une théorie de défauts, il faut localiser les règles qui sont en conflit et qui n'ont pas la même spécificité ; ceci est accompli en utilisant une partie du mécanisme du Système Z (Pearl, 1990). Ainsi pour notre exemple, il est clair que les règles $O \rightarrow V$ et $M \rightarrow \neg V$ sont en conflit (une contradition est déduite, à savoir V et $\neg V$, à cause de $M \rightarrow O$) et que la seconde est plus spécifique que la première (M est une sous-classe de O). Deuxièmement, il faut modifier certaines règles de sorte que si les deux règles en contradiction sont potentiellement applicables alors seule la plus spécifique est appliquée.

Dans le Système Z, un ensemble de règles R est partitionné (stratifié) en sous-ensembles $R_0, \ldots, R_n$ où les règles d'une strate inférieure sont moins spécifiques que celles d'une strate supérieure. La partition résultante est appelée un *ordre Z* ; cet ordre fournit donc des informations de spécificité. Plutôt que calculer l'ordre Z complet, Delgrande et Schaub déterminent d'abord des ensembles minimaux de règles en conflit, qu'ils stratifient séparément ; ainsi ils classent les informations selon leur spécificité, relativement au(x) conflit(s) dans le(s)quel(s) elles interviennent. Dans notre exemple, $C = \{O \rightarrow V,\ M \rightarrow O,\ M \rightarrow \neg V\}$ est un conflit (en présence de M).

Delgrande et Schaub ont montré que l'ordre Z d'un tel ensemble C est une partition binaire (C_0, C_1) des règles ; les règles de C_0 sont moins spécifiques que celles de C_1. Par exemple, pour C nous obtenons : $C_0 = \{O \rightarrow V\}$ et $C_1 = \{M \rightarrow O,\ M \rightarrow \neg V\}$. Par conséquent, si les règles de C_1 sont applicables, on voudrait être sûr que certaines règles de C_0 sont bloquées.

Maintenant que nous avons localisé les règles en conflit, il nous faut déterminer lesquelles sont candidates pour être bloquées, ainsi que le moyen de bloquer leur application dans le formalisme utilisé.

Intuitivement, O est une sur-classe de M. Si O et M sont vrais, alors dans la traduction en logique des défauts, on voudrait que le défaut correspondant à $M \rightarrow \neg V$ soit applicable de préférence à $O \rightarrow V$. Donc on veut que les règles les plus spécifiques soient applicables de préférence aux règles moins spécifiques en conflit, indépendamment des autres règles de l'ensemble. Ceci est fait en localisant les règles dont l'application conjointe crée une inconsistance. Dans notre exemple, il s'agit de $O \rightarrow V$ et $M \rightarrow \neg V$. $O \rightarrow V \in C_0$ et $M \rightarrow \neg V \in C_1$. Les règles sélectionnées de cette manière dans C_0 constituent les candidates pour être bloquées. Ce critère de sélection a la propriété importante d'être *indépendant du contexte*. Pour des théories de défauts R et R' telles que $R \subseteq R'$, si $r \in R$ est choisie, alors elle devrait également être choisie dans R'. De plus, si l'on souhaite bloquer une règle dans un théorie de défauts R, alors on voudra aussi la bloquer dans tout sur-ensemble R'.

Le deuxième problème (*« Comment bloquer l'application d'une règle ? »*) dépend du formalisme utilisé. Pour la logique des défauts par exemple, la théorie de défauts correspondant à notre ensemble de règles est composée de défauts normaux, excepté pour ceux représentant les règles sélectionnées de la manière décrite ci-dessus, qui deviennent semi-normaux ; pour ces derniers, la justification se compose du conséquent

avec une assertion assurant que les règles de C_1 sélectionnées comme précédemment ne sont pas applicables (on ajoute ces règles, transformées en implications matérielles, puis on simplifie si possible).

Considérons l'ensemble C. Les règles $M \rightarrow O$ et $M \rightarrow \neg V$ sont transformées respectivement en : $\frac{M:O}{O}$ et $\frac{M:\neg V}{\neg V}$. La règle $O \rightarrow V$ est quant à elle transformée en : $\frac{O:V \wedge (M \rightarrow \neg V)}{V}$, qui peut être simplifié en : $\frac{O:V \wedge \neg M}{V}$.

Pour résumer, on peut décomposer l'approche décrite ci-dessus de la manière suivante, pour un ensemble de règles R :

Algorithme 3.1 Procédure `transformer_ensemble`

Entrées : Un ensemble R de règles avec exceptions
Sorties : L'ensemble R modifié de manière à gérer la spécificité

```
1 conf ← conflits_minimaux(R)
2 pour chaque C ∈ conf faire
3     stratifier(C)
4     sélectionner_règles(C)
5 fin pour
6 transformer_règles(R, conf)
```

Nous allons procéder de la même manière dans le cadre de l'ASP : nous commencerons par chercher les conflits, nous les stratifierons puis les partitionnerons et, enfin, nous modifierons les règles candidates.

4 Spécificité en ASP

Pour notre travail, nous avons adapté pour l'ASP toutes les notions définies dans les travaux sur lesquels nous nous basons ainsi que les algorithmes liés.

Définition 3
Pour une règle r d'un programme logique défini P telle que $r = b \leftarrow a_1, \ldots, a_n.(n \geq 0)$, un ensemble d'atomes A :

- *satisfait r si $\{a_1, \ldots, a_n\} \not\subseteq A$ ou $b \in A$;*
- *vérifie r si $\{a_1, \ldots, a_n\} \subseteq A$ et $b \in A$;*
- *falsifie r si $\{a_1, \ldots, a_n\} \subseteq A$ et $b \notin A$.*

Définition 4
Une règle r d'un programme P est dite tolérée *par P si et seulement s'il existe un ensemble d'atomes A, clos par rapport à P et consistant, qui vérifie r.*

La tolérance d'une règle caractérise le fait que son application ne génère pas de contradiction. À partir de cette notion de tolérance on va pouvoir obtenir une stratification du programme appelée *ordre Z*.

Nous développons par la suite les quatre étapes de l'algorithme concernant les conflits (à savoir les lignes 1, 3, 4 et 6 de la procédure $transformer\_ensemble$). Un conflit étant un ensemble minimal, les règles qui le composent ne peuvent introduire qu'une et une seule inconsistance ; ainsi, un conflit comporte une et une seule contrainte (voir la section 2 pour la définition d'une contrainte). Pour un conflit C, on définit :

- $contrainte(C)$ la contrainte de C ;
- $règles(C) = C \setminus \{contrainte(C)\}$;
- $contr(C) = corps(contrainte(C))$ (atomes en conflit).

4.1 Calcul des conflits *(ligne 1 de l'algorithme)*

Si déterminer les conflits pour un nombre restreint de règles peut sembler simple, ce n'est pas forcément vrai dans le cas général. La définition d'un conflit (adaptée dans notre cadre) donnée dans (Delgrande & Schaub, 1997) est la suivante, où un ordre Z trivial possède une seule strate :

Définition 5
Soit P un programme logique défini. Un ensemble de règles $C \subseteq P$ est un conflit *dans P si et seulement si C a un ordre Z non trivial et si chaque $C' \subset C$ a un ordre Z trivial.*

Dans le pire des cas, pour un programme P de n règles, il faudrait donc isoler et stratifier les 2^n sous-ensembles de P. Heureusement, des travaux similaires ont été menés. En logique classique, et en particulier dans le cadre du problème SAT, un MUS (*Minimally Unsatisfiable Subformula*) est un ensemble insatisfiable de clauses tel que tous ses sous-ensembles propres sont satisfiables. Un tel ensemble fournit donc une « explication » de l'incohérence qui ne peut être plus petite en termes de nombre de clauses impliquées, ce qui correspond à notre idée de conflits. Des travaux ont montré que le calcul de MUS n'est pas réalisable en pratique dans le pire dans cas ; en effet, décider si un ensemble de clauses est un MUS est DP-complet (Papadimitriou & Wolfe, 1988), et tester si une formule appartient à l'ensemble des MUS est Σ_2^p-difficile (Eiter & Gottlob, 1992). Mais Bruni (Bruni, 2005) a montré que pour certaines classes de clauses (dont les clauses de Horn), l'extraction d'un MUS pouvait être réalisée en un temps polynomial.

Nous avons donc décidé d'utiliser l'algorithme HYCAM[1] (Grégoire *et al.*, 2007) qui permet de déterminer tous les MUS d'une instance SAT en un temps raisonnable. La fonction $conflits\_minimaux$ détermine les conflits grâce à des appels à HYCAM.

On veut connaître les règles qui posent *potentiellement* problème (lorsqu'elles sont applicables conjointement), il faut donc ajouter des faits au programme avant de le passer à HYCAM (ligne 7). Choisir successivement les atomes du corps de chaque règle (lignes 3 à 7) plutôt que chaque atome (présent dans le programme) séparément assure que chaque règle soit déclenchée au moins une fois ; l'algorithme trouve ainsi tous les MUS dans lesquels elle intervient. Nous cherchons des ensembles minimaux ; nous devons donc nous assurer qu'aucun conflit généré n'est un sur-ensemble d'un autre conflit (lignes 8 à 10).

Exemple 4.1
Soit le programme : P = {$Co \leftarrow Mo.(r_1)$, $Mo \leftarrow Ce.(r_2)$, $nCo \leftarrow Ce.(r_3)$, $Ce \leftarrow Na.(r_4)$, $Co \leftarrow Na.(r_5)$, $\bot \leftarrow Co,\ nCo.(c_1)$} où généralement, les mollusques (Mo)

[1] Il s'agit en réalité d'une amélioration d'un algorithme existant, CAMUS (*Computing All MUS*) (Liffiton & Sakallah, 2008).

Algorithme 4.1 Fonction `conflits_minimaux`

Entrées : Un programme logique P
Sorties : L'ensemble des ensembles minimaux de conflit de P

```
1  conflits ← ∅
2  traités ← ∅
3  pour chaque r ∈ P \ P_K faire
4      A ← corps(r)
5      si A ∉ traités alors
6          traités ← traités ∪ {A}
7          conf ← HYCAM(P ∪ A)
8          pour chaque C ∈ conf faire
9              si ∀C' ∈ conflits, C' ⊈ C alors
10                 conflits ← (conflits ∪ {C}) \ {C' ∈ conflits | C ⊂ C'}
11             fin si
12         fin pour
13     fin si
14 fin pour
15 retourner conflits
```

sont des coquillages (Co), les céphalopodes (Ce) sont des mollusques, les céphalopodes ne sont pas des coquillages, les nautiles (Na) sont des cépalopodes et les nautiles sont des coquillages. Les ensembles de faits qu'il faut ajouter sont $\{Mo\}$, $\{Ce\}$ et $\{Na\}$. Pour $\{Mo\}$, aucun MUS (conflit) n'est détecté. Pour $\{Ce\}$, on obtient le conflit $C = \{r_1,\ r_2,\ r_3,\ c_1\}$. Pour $\{Na\}$, on obtient $C' = \{r_1,\ r_2,\ r_3,\ r_4,\ c_1\}$ et $C'' = \{r_3,\ r_4,\ r_5,\ c_1\}$. $C \subset C'$, donc C' n'est pas minimal. Les deux seuls conflits de P sont donc : C = {$Co \leftarrow Mo.$, $Mo \leftarrow Ce.$, $nCo \leftarrow Ce.$, $\bot \leftarrow Co,\ nCo.$} et C'' = {$nCo \leftarrow Ce.$, $Ce \leftarrow Na.$, $Co \leftarrow Na.$, $\bot \leftarrow Co,\ nCo.$}

4.2 Stratification des conflits *(ligne 3 de l'algorithme)*

Delgrande et Schaub ont montré qu'un conflit possède un ordre Z avec exactement deux strates.

Algorithme 4.2 Fonction `stratifier`

Entrées : Un ensemble minimal de conflit C
Sorties : La stratification (C_0, C_1) de C

```
1  C_0 ← ∅
2  C_1 ← ∅
3  pour chaque r ∈ règles(C) faire
4      si r est tolérée par (règles(C) \ {r}) ∪ {contrainte(C)} alors
5          C_0 ← C_0 ∪ {r}
6      sinon
7          C_1 ← C_1 ∪ {r}
8      fin si
9  fin pour
10 retourner (C_0, C_1)
```

On sait qu'il n'y aura que deux strates, donc soit une règle est tolérée dans le conflit (ligne 4) et elle appartient à la strate la plus générale (ligne 5), soit elle ne l'est pas (ligne 6) et elle appartient à la strate la plus spécifique (ligne 7).

Exemple 4.2
Considérons un conflit de l'exemple précédent : C = {$Co \leftarrow Mo.(r_1)$, $Mo \leftarrow Ce.(r_2)$, $nCo \leftarrow Ce.(r_3)$, $\perp \leftarrow Co,\ nCo.(c_1)$}. $\{Mo,\ Co\}$ *vérifie* r_1, *est clos par rapport à* C *et est consistant. Donc* r_1 *est tolérée dans* C, *et* $r_1 \in C_0$. *Le seul ensemble clos qui vérifie* r_2 *et ne falsifie ni* r_1 *ni* r_3 *est* $A = \{Ce,\ Mo, Co,\ nCo\}$; *mais* A *est inconsistant. Alors* $r_2 \in C_1$. *De même, on trouve que* $r_3 \in C_1$. *Finalement, l'ordre Z associé à* C *est le suivant :* C_0 = {$Co \leftarrow Mo.$} *et* C_1 = {$Mo \leftarrow Ce.$, $nCo \leftarrow Ce.$}.

4.3 Sélection des règles candidates *(ligne 4 de l'algorithme)*

Une fois un conflit C stratifié, on peut faire ressortir trois sous-ensembles : les règles candidates pour être modifiées (règles générales ayant des exceptions), $min(C)$; les règles indiquant comment modifier les règles précédentes (les exceptions), $max(C)$; et les règles restantes faisant le lien entre ces deux ensembles, $inf(C)$. Pour cela, il faut d'abord déterminer le plus petit sous-ensemble de règles dont l'application conjointe provoque une inconsistance.

Définition 6
Soit (C_0, C_1) *la stratification des règles du conflit* C. *Un* noyau *de* C *est une paire d'ensembles minimaux* $\big(min(C), max(C)\big)$ *tels que* $min(C) \subseteq C_0$, $max(C) \subseteq C_1$ *et* {$a \leftarrow .|a \in corps(r) \cup \{tête(r)\}$ *tq* $r \in min(C) \cup max(C)$} $\cup$ {$contrainte(C)$} *possède un modèle inconsistant.*

Algorithme 4.3 Procédure `sélectionner_règles`

Entrées : Un ensemble minimal de conflit stratifié C
Sorties : Les règles candidates ($min(C)$), les exceptions ($max(C)$) et les autres ($inf(C)$)
1 $noyau \leftarrow \{r \in$ règles(C) | tête(r) $\in$ contr(C)$\}$
2 min(C) $\leftarrow noyau \cap C_0$
3 max(C) $\leftarrow noyau \cap C_1$
4 inf(C) $\leftarrow$ règles(C) $\setminus noyau$

Le noyau est constitué des règles qui concluent sur des atomes en conflit (ligne 1) ; ainsi, nous sommes assurés qu'un tel noyau est unique puisque les têtes des règles ne contiennent qu'un atome. Il est alors facile de partitionner le conflit (lignes 2 à 4).

Exemple 4.3
Reprenons le conflit stratifié précédemment : C_0 = {$Co \leftarrow Mo.$}, C_1 = {$Mo \leftarrow Ce.$, $nCo \leftarrow Ce.$}, $contrainte(C) = \perp \leftarrow Co,\ nCo.$, $contr(C)$ = {$Co,\ nCo$}, *alors le noyau est :* $\big(\{Co \leftarrow Mo.\}, \{nCo \leftarrow Ce.\}\big)$ *et donc :* $min(C)$ = {$Co \leftarrow Mo.$}, $max(C)$ = {$nCo \leftarrow Ce.$} *et* $inf(C)$ = {$Mo \leftarrow Ce.$}.

4.4 Modification des règles *(ligne 6 de l'algorithme)*

Une fois les noyaux des conflits déterminés, les règles sont transformées de telle sorte que les plus générales sont bloquées si au moins une de leurs exceptions est applicable,

mais restent déclenchables sinon. Pour cela, nous transformons les règles les plus générales en explicitant les exceptions dans leur corps négatif. Le corps négatif des autres règles non sujettes à exceptions est laissé vide. L'algorithme suivant réalise cette tâche en transformant un programme logique défini P en le programme logique normal correspondant à partir de l'ensemble de ses conflits $conflits$:

Algorithme 4.4 Procédure `transformer_règles`

Entrées : Un programme logique défini P, un ensemble $conflits$ d'ensembles minimaux de conflit
Sorties : P transformé en un programme logique normal tenant compte de la spécificité

```
1  pour chaque r ∈ P faire
2      si r est une contrainte alors
3          pour chaque r' ∈ P tel que tête(r') ∈ corps+(r) faire
4              corps−(r') ← corps−(r') ∪ (corps+(r) \ tête(r'))
5          fin pour
6      sinon
7          conf ← {C ∈ conflits | r ∈ min(C)}
8          si conf ≠ ∅ alors
9              pour chaque C ∈ conf faire
10                 pour chaque r' ∈ max(C) faire
11                     si |corps+(r')| = 1 alors
12                         corps−(r) ← corps−(r) ∪ corps+(r')
13                     sinon
14                         P ← P ∪ {σ_r'}
15                         corps−(r) ← corps−(r) ∪ {tête(σ_r')}
16                     fin si
17                 fin pour
18             fin pour
19         fin si
20     fin si
21 fin pour
```

Si r est une contrainte, on transforme toutes les règles pouvant déclencher la contrainte c'est-à-dire pouvant être impliquées dans une contradiction (lignes 3 à 5). Si ce n'est pas une contrainte, on cherche les conflits pour lesquels $r \in min(C)$ (candidate pour être modifiée) (ligne 7) puis on modifie la règle en explicitant ses exceptions (lignes 9 à 18), avec un traitement différent selon le nombre d'atomes dans le corps positif de la règle r' définissant l'exception (ligne 12 ou lignes 14 et 15). Pour une telle règle r' (lignes 14 et 15), on définit $\sigma_{r'}$, une nouvelle règle telle que :

- $tête(\sigma_{r'})$ est un nouvel atome n'apparaissant pas déjà dans le programme ;
- $corps^+(\sigma_{r'}) = corps^+(r')$;
- $corps^-(\sigma_{r'}) = \emptyset$.

Par construction, $\sigma_{r'}$ est une règle qui sera déclenchée chaque fois que r sera applicable.

Ainsi, pour chacune de ses exceptions r', r est bloquée si tous les atomes du corps positif de r' sont prouvés (i.e. r' est applicable) ou si la tête de r' a été obtenue (ou si on a déjà conclu l'« opposé » de la tête de r).

Exemple 4.4
Pour notre exemple, le programme logique normal correspondant est : $P = \{Co \leftarrow Mo,\ not\ nCo,\ not\ Ce.,\ Mo \leftarrow Ce.,\ nCo \leftarrow Ce,\ not\ Co,\ not\ Na.,\ Ce \leftarrow Na.,\ Co \leftarrow Na,\ not\ nCo.,\ \bot \leftarrow Co,\ nCo.\}$.

5 Conclusion et perspectives

A partir d'une représentation compacte des informations, nous pouvons maintenant utiliser les ASP pour représenter automatiquement les informations tolérant les exceptions et pouvoir raisonner avec via la sémantique des modèles stables.

Ce travail, développé lors d'un stage de Master Recherche, comporte aussi le développement du système pratique opérationnel implantant les algorithmes décrits ici. Toutes les informations (rapport de stage et programmes) sont disponibles à l'adresse `http://www.info.univ-angers.fr/pub/pn/Softwares/aspSpecif.html`.

Nous nous intéressons maintenant à resituer notre travail par rapport aux autres travaux existants en particulier aux propriétés définies pour les systèmes de raisonnement non-monotone telles que celles du système P et la monotonie rationnelle.

Références

BRUNI R. (2005). On Exact Selection of Minimally Unsatisfiable Subformulae. *Annals of Mathematics and Artificial Intelligence*, **43**(1-4), 35–50.

DELGRANDE J. P. & SCHAUB T. (1997). Compiling specificity into approaches to nonmonotonic reasoning. *Artificial Intelligence*, **90**(1-2), 301–348.

EITER T. & GOTTLOB G. (1992). On the Complexity of Propositional Knowledge Base Revision, Updates and Counterfactuals. *Artificial Intelligence*, **57**(2-3), 227–270.

GELFOND M. & LIFSCHITZ V. (1988). The stable model semantics for logic programming. In R. A. KOWALSKI & K. BOWEN, Eds., *International Conference on Logic Programming*, p. 1070–1080, Cambridge, Massachusetts : The MIT Press.

GELFOND M. & LIFSCHITZ V. (1991). Classical negation in logic programs and disjunctive databases. *New Generation Computing*, **9(3-4)**, 363–385.

GRÉGOIRE E., MAZURE B. & PIETTE C. (2007). Boosting a Complete Technique to Find MSS and MUS thanks to a Local Search Oracle. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI'07)*, volume 2, p. 2300–2305, Hyderabad (Inde).

LIFFITON M. H. & SAKALLAH K. A. (2008). Algorithms for Computing Minimal Unsatisfiable Subsets of Constraints. *J. Autom. Reasoning*, **40**(1), 1–33.

PAPADIMITRIOU C. H. & WOLFE D. (1988). The Complexity of Facets Resolved. *Journal of Computer and System Sciences*, **37**(1), 2–13.

PEARL J. (1990). System Z : A natural ordering of defaults with tractable applications to nonmonotonic reasoning. In R. PARIKH, Ed., *Proceedings of Theoritical Aspects of Reasonning about Knowledge*, p. 121–135, San Mateo : Morgan Kaufmann Publishers.

REITER R. (1980). A logic for default reasoning. *Artificial Intelligence*, **13**, 81–132.

Contrepartie sémantique de la révision locale de croyances par le modèle C-structure

Omar Doukari

UMR CNRS 6168 LSIS, 13397 MARSEILLE CEDEX 20
omar.doukari@lsis.org

Résumé : Dans ce papier, nous définissons un nouveau modèle pour la représentation et la révision locale de croyances que nous appelons le modèle C-structure. D'autre part, en utilisant les systèmes de sphères de Grove, nous considérons des contraintes additionnelles sur le calcul de la distance entre les interprétations et nous prouvons que ces contraintes caractérisent précisément la contrepartie sémantique de la révision par le modèle C-structure.

Mots-clés : modèle C-structure, systèmes de sphères de Grove, révision de croyances.

1 Introduction

Agents faced with incomplete, uncertain, and inaccurate information must employ a rational belief revision operation in order to manage belief changes. The agent's epistemic state represents its reasoning process ; belief revision consists of modifying its initial epistemic state in order to maintain consistency, while keeping new information and modifying previous information *as less as possible* (principle of minimal change).

To introduce relevance-sensitivity into belief revision, Parikh (Parikh 99) defined the language splitting model which says that any set of beliefs may be represented as a family of letter-disjoint sets and that revision may be restricted to local portions of the belief corpus (those intersecting with the language of the new epistemic input). In practice, since beliefs do have some overlap, the partition of the main set of beliefs cannot be actually strict. In view of this gap, Parikh's original model for belief revision (and others based on it) (Chopra 01a; Kourousias 07; Parikh 99; Peppas 04) has been extended, by allowing for such overlap, in the B-structure model of (Chopra 00). However, this model is incapable to guarantee a global revision by only a local one, i.e., after revising our beliefs we are not sure if they are globally consistent or not. This fact interferes with the soundness of belief revision.

A new model for belief representation and local belief revision called the C-structure model and also based on such overlap was defined (Doukari 07b). This model based on the containment property defined in (Doukari 07a), allows some overlap between the different belief subsets and preserves all the desirable properties of the language splitting model, in particular soundness of revision operation. Furthermore, this model

allows to respect the principle of minimal change when the language splitting (henceforth LS) model fails to do that.

Using Grove's system of spheres construction (Grove 88), we provide a semantics for local revision using the C-structure model, by providing additional constraints based on a distance measurement between interpretations. We prove these constraints characterize the containment property.

The structure of the paper is as follows. In section 2, we provide some preliminaries on the AGM paradigm. In section 3, we define the C-structure model. In section 4, we provide system of spheres semantics for belief revision by the C-structure model.

2 Preliminaries

Throughout this paper, $\mathcal{L}$ is a propositional language defined on some finite set of propositional variables (atoms) $\mathcal{V}$ and the usual connectors ($\neg$, $\vee$, $\wedge$, $\rightarrow$, $\leftrightarrow$). $|X|$ denotes the cardinality of the set X. If $\alpha \in \mathcal{L}$ is a sentence, then $\mathcal{V}(\alpha)$ represents the set of variables appearing in α, and similarly for a set of sentences. If V is a subset of $\mathcal{V}$ then $\mathcal{L}(V)$ represents the propositional sublanguage defined over V. $\vdash$ represents the classical inference relation. A literal is a propositional variable or its negation. A clause is a disjunction of literals. A clause c is an implicate of a sentence α iff $\alpha \vdash c$. A clause c is a prime implicate of α iff for all implicates c' of α such that $c' \vdash c$, it is the case that $c \vdash c'$. We denote by $Cove_\alpha$ an arbitrary covering of α, which is a set of prime implicates of α such that for every clause c where $\alpha \vdash c$, there exists $c' \in Cove_\alpha$ such that $c' \vdash c$. $\mathcal{V}(Cove_\alpha)$ is the minimal set of atoms needed to express (a sentence logically equivalent to) α (Herzig 99). This set is unique (Parikh 99).

If X is a set of sentences then $Cn(X)$ is the logical closure of X. In particular, X is a theory iff $X = Cn(X)$. For a theory T, B_T denotes a belief base of T, which is a finite set of sentences such that $T = Cn(B_T)$. B_T is a minimal belief base of T iff (i) B_T is a belief base of T, (ii) T is axiomatized by B_T (i.e., $\forall \alpha \in B_T, B_T \setminus \{\alpha\} \nvdash \alpha$), and (iii) $B_T \subseteq Cove_{\bigwedge_{\alpha \in B_T}}$.

In particular, if B_T is an inconsistent belief base, $M \subseteq B_T$ is a minimal inconsistent subset (MIS) of B_T iff M is inconsistent, and $\forall M' \subset M, M'$ is consistent. We denote the set of all consistent theories of $\mathcal{L}$ by $\mathcal{K}_\mathcal{L}$. A theory T of $\mathcal{K}_\mathcal{L}$ is complete iff $\forall \alpha \in \mathcal{L}, \alpha \in T$ or $\neg\alpha \in T$. We denote by $\mathcal{M}_\mathcal{L}$ the set of all consistent complete theories of $\mathcal{L}$. In the context of system of spheres, consistent complete theories play the role of interpretations (possible worlds). For a set of sentences X of $\mathcal{L}$, $[X]$ represents the set of all interpretations of $\mathcal{L}$ that contain X. For a theory T and a set of sentences X of $\mathcal{L}$, $T + X$ represents the set $Cn(T \cup X)$.

For a sublanguage $\mathcal{L}'$ of $\mathcal{L}$ defined over a subset V' of $\mathcal{V}$, $\overline{\mathcal{L}'} = \mathcal{L}(\mathcal{V} \setminus V')$. $Cn_{\mathcal{L}'}(X)$ for $X \subset \mathcal{L}'$, represents the logical closure of X in $\mathcal{L}'$. When no subscript is present, it is understood that the operation is relevant to the original language $\mathcal{L}$.

In belief revision, much work takes as its starting point the AGM postulates, which appear to capture much of what characterizes rational belief revision (Alchourrón 85). In this framework belief states are represented as theories of $\mathcal{L}$, and the process of belief revision is modeled on a revision function $*$ which is any function from $\mathcal{K}_\mathcal{L} \times \mathcal{L}$ to $\mathcal{K}_\mathcal{L}$,

mapping $\langle T, \alpha \rangle$ to $T * \alpha$ that satisfies the AGM postulates (see them in (Alchourrón 85)).

Apart from this axiomatic approach to belief revision, Grove(Grove 88) introduced another construction of revision functions based on a special structure on consistent complete theories, called a system of spheres. Let T be a theory of $\mathcal{L}$, and S_T a collection of sets of interpretations i.e., $S_T \subseteq 2^{\mathcal{M}_\mathcal{L}}$. S_T is a system of spheres centered on $[T]$ iff the following conditions are satisfied :

($\mathbf{S1}$). S_T is totally ordered ; if $U, U' \in S_T$ then $U' \subseteq U$ or $U \subseteq U'$.

($\mathbf{S2}$). The smallest sphere in S_T is $[T]$; $[T] \in S_T$ and if $U \in S_T$ then $[T] \subseteq U$.

($\mathbf{S3}$). $\mathcal{M}_\mathcal{L} \in S_T$.

($\mathbf{S4}$). $\forall \alpha \in \mathcal{L}$, if there is any sphere in S_T intersecting $[\alpha]$ then there is also a smallest sphere in S_T intersecting $[\alpha]$.

For a system of spheres S_T and a sentence $\alpha \in \mathcal{L}$, the smallest sphere in S_T intersecting α is denoted $C_T(\alpha)$. With any system of spheres S_T, Grove associates a function $f_T : \mathcal{L} \mapsto 2^{\mathcal{M}_\mathcal{L}}$ defined as follows : $f_T(\alpha) = [\alpha] \cap C_T(\alpha)$. Consider now a theory T of $\mathcal{L}$ and let S_T be a system of spheres centered on $[T]$. Grove uses S_T to define constructively the process of revising T, by means of the following condition : $(S*) : T * \alpha = \bigcap f_T(\alpha)$.

Grove showed the class of functions generated from systems of spheres by means of $(S*)$, is precisely the family of the functions satisfying the AGM postulates.

We now recall the main definitions and results of the C-structure model.

3 The C-structure model (Doukari 07b)

The C-structure model uses disjoint sublanguages to define a set of *cores* of a given language, each surrounded by a *covering* of atoms. The concept of covering allows some degree of overlap between the sublanguages defined over the coverings.

Definition 1
$\{V_1, ..., V_n\}$ is a set of cores of $\mathcal{L}$ iff it is a partition of $\mathcal{V}$.

Example 1
Let the language $\mathcal{L}$ be built from the propositional variables a, b, c, d. Let T be an arbitrary theory of $\mathcal{L}$, axiomatized by the minimal belief base $B_T = \{\neg a \vee b, \neg b \vee c, \neg c \vee b, \neg c \vee d\}$. The set $\{\{a\}, \{b\}, \{c\}, \{d\}\}$ is a set of cores of $\mathcal{L}$.

To order the atoms of $\mathcal{L}$, we use the following relevance relation from (Chopra 01b).

Definition 2
Let T be a theory of $\mathcal{L}$. We say that two atoms, p and q, are directly relevant wrt B_T, denoted by $R(p, q, B_T)$ (or by $R_0(p, q, B_T)$), iff $\exists \alpha \in B_T$ s.t., $p, q \in \mathcal{V}(\alpha)$. Two atoms p, q are k-relevant wrt B_T, denoted by $R_k(p, q, B_T)$, if $\exists p_0, p_1, ..., p_{k+1} \in \mathcal{V}$ s.t. : $p_0 = p$; $p_{k+1} = q$; and $\forall i \in \{0, ..., k\}, R(p_i, p_{i+1}, B_T)$.

In Example 1, we find : $R(a, b, B_T)$, $R_1(a, c, B_T)$, $R_2(a, d, B_T)$, etc.

To define clearly the extent of overlapping between the various sublanguages, we define a *distance* between variables.

Definition 3
Suppose two atoms $p, q \in \mathcal{V}$, T is a theory of $\mathcal{L}$. The distance between p, q wrt B_T, denoted by $dist(p, q, B_T)$, is defined as follows :

$$dist(p, q, B_T) = \begin{cases} 0 & \text{if } p = q \\ min\{k : R_k(p, q, B_T)\} + 1 & \text{if } k \text{ exists} \\ \infty & \text{otherwise.} \end{cases}$$

In Example 1 : $dist(a, b, B_T) = 1$, $dist(a, c, B_T) = 2$, $dist(a, d, B_T) = 3$, etc.

We now define the notion of a covering, parametrized by its *thickness* :

Definition 4
Let $\{V_1, ..., V_n\}$ be a set of cores of $\mathcal{L}$ and T be a theory of $\mathcal{L}$. $Cov_k(V_i, B_T)$ is a covering whose thickness is equal to k of V_i wrt B_T iff : $Cov_k(V_i, B_T) \subseteq \mathcal{V}$; and $\forall p \in \mathcal{V}, if \exists q \in V_i$ s.t., $dist(p, q, B_T) \leq k$ then $p \in Cov_k(V_i, B_T)$.

For example, the set of coverings with thickness 1 corresponding to the set of cores $\{\{a\}, \{b\}, \{c\}, \{d\}\}$ wrt B_T (Example 1) is : $\{\{a, b\}, \{a, b, c\}, \{b, c, d\}, \{c, d\}\}$.

In order to parametrize a C-structure by a particular thickness, we require a definition of the size of a MIS :

Definition 5
Let B_T and $B'_{T'}$ be two belief bases such that $\mathcal{V}(B'_{T'}) \subseteq \mathcal{V}(B_T)$ and $B'_{T'}$ is inconsistent. The size of the MIS M of $B'_{T'}$ wrt B_T, $Size(M, B_T) = max\{dist(a, b, B_T) : a, b \in \mathcal{V}(M)\}$.

In Example 1, let $M = \{a \wedge \neg b, a \rightarrow b\}$ be a MIS of $B'_{T'} = B_T \cup \{a \wedge \neg b\}$, so $Size(M, B_T) = 1$.

When we want to construct a C-structure C on a belief base B_T, we only require that the thickness of coverings of cores (value of k) should be (at least) equal to the maximal size of MISs which may exist in B_T.

Definition 6
Let T be a theory defined in $\mathcal{L}$ and B_T an arbitrary belief base of T. The set $C = \{(V_1, Cov_k(V_1, B_T), T_1), ..., (V_n, Cov_k(V_n, B_T), T_n)\}$ is a C-structure of T iff : (i) *$\{V_1, ..., V_n\}$ is a set of cores of $\mathcal{L}$,* (ii) *$Cov(C) = \{Cov_k(V_1, B_T), ..., Cov_k(V_n, B_T)\}$ is a corresponding set of coverings wrt B_T s.t., $\forall i \in \{1, ..., n\} \forall \alpha \in \mathcal{L}(Cov_k(V_i, B_T))$, if $B_T \cup \{\alpha\}$ is inconsistent, then $\forall M$ a MIS of $B_T \cup \{\alpha\}$, $Size(M, B_T) \leq k$, and* (iii) *$\forall T_i, T_i = Cn_{\mathcal{L}(Cov_k(V_i, B_T))}(\mathcal{L}(Cov_k(V_i, B_T)) \cap T)$. C is called an atomic C-structure of T iff $\forall i \in \{1, ..., n\}$, $|V_i| = 1$*

We obtain the following C-structure corresponding to Example 1 by assuming that the maximal size of eventual exiting MISs in B_T is 1 (condition (ii) of Definition 6) : $\{(\{a\}, \{a, b\}, Cn_{\mathcal{L}(\{a,b\})}(\{\neg a \vee b\}))$, $(\{b\}, \{a, b, c\}, Cn_{\mathcal{L}(\{a,b,c\})}(\{\neg a \vee b, \neg b \vee c, \neg c \vee b\}))$, $(\{c\}, \{b, c, d\}, Cn_{\mathcal{L}(\{b,c,d\})}(\{\neg b \vee c, \neg c \vee b, \neg c \vee d\}))$, $(\{d\}, \{c, d\}, Cn_{\mathcal{L}(\{c,d\})}(\{\neg c \vee d\}))\}$.

Now, we can formalize the local revision property called containment property.

(Containment Property) : Let T be a theory of $\mathcal{L}$, B_T an arbitrary belief base of T, and $C = \{(V_1, Cov_k(V_1, B_T), T_1), ..., (V_n, Cov_k(V_n, B_T), T_n)\}$ a C-structure of T. If $\alpha \in \mathcal{L}(Cov_k(V_i, B_T))$ and $\mathcal{V}(Cove_\alpha) \cap V_i \neq \emptyset$ for some i, then : $T * \alpha = (T_i \circ \alpha) + ((\bigcup_{j=0}^{n} T_j) \setminus T_i)$ where $\circ$ is a revision operator of the sublanguage $\mathcal{L}(Cov_k(V_i, B_T))$.

4 Semantics for the Containment Property

Consider a C-structure $C = \{(V_1, Cov_k(V_1, B_T), T_1), ..., (V_n, Cov_k(V_n, B_T), T_n)\}$ of the theory T. Moreover, let α be any sentence in $\mathcal{L}(Cov_k(V_1, B_T))$ such that $\mathcal{V}(Cove_\alpha) \cap V_1 \neq \emptyset$. According to the containment property, anything outside T_1 in T will not be affected during the revision of the theory T by α. More formally, this leads to the following condition, which is equivalent to the containment property.

(**C**). If $C = \{(V_1, Cov_k(V_1, B_T), T_1), ..., (V_n, Cov_k(V_n, B_T), T_n)\}$ is a C-structure of T, $\alpha \in \mathcal{L}(Cov_k(V_1, B_T))$ and $\mathcal{V}(Cove_\alpha) \cap V_1 \neq \emptyset$, then : (i) if $\forall i \in \{2, ..., n\}$, $\mathcal{L}(Cov_k(V_1, B_T)) \cap \mathcal{L}(Cov_k(V_i, B_T)) = \emptyset$, then : $T \cap \mathcal{L}(Cov_k(V_1, B_T)) = (T * \alpha) \cap \mathcal{L}(Cov_k(V_1, B_T))$; (ii) otherwise : $((\bigcup_{i=0}^{n} T_i) \setminus T_1) \subseteq T * \alpha$.

Condition (C) is straightforward : when revising a theory T by a sentence α, the part of T that is not related to α is not affected by the revision ; we do not remove any information from it since the size of MISs is limited by k. However, we can deduce more consequences because the existence of the overlap between the parts related and unrelated to α. The following result shows (C) is equivalent to the containment property.

Theorem 1
Let $*$ be a revision function satisfying the AGM postulates $(T * 1)$–$(T * 8)$. Then $*$ satisfies the containment property iff $*$ satisfies (C).

4.1 The special case of Complete Theories

Let T be a consistent complete theory, and let S_T be a system of spheres centered on $[T]$. The intended meaning of S_T is that it represents comparative similarity between possible worlds i.e., the further away a world is from the center of S_T, the less similar it is to $[T]$. However, none of the conditions (S1)–(S4) indicate how similarity between worlds should be measured. In (Peppas 00), a specific criterion of similarity is considered, originally introduced in the context of reasoning about action with Winslett's Possible Models Approach (PMA) (Winslett 88).

This criterion, called PMA's criterion of similarity, measures "distance" between worlds based on propositional variables. In particular, let w, w' be any two interpretations of $\mathcal{L}$. By $Diff(w, w')$ we denote the set of propositional variables that have different truth values in the two interpretations i.e., $Diff(w, w') = \{v_i \in \mathcal{V} : v_i \in w$ and $v_i \notin w'\} \cup \{v_j \in \mathcal{V} : v_j \notin w$ and $v_j \in w'\}$. A system of spheres S_T is a PMA system of spheres iff it satisfies the following condition (Peppas 00) :

(PS) For any two consistent complete theories w and w', if $Diff\ (T, w) \subset Diff\ (T, w')$ then there is a sphere $U \in S_T$ that contains w but not w'.

4.1.1 Condition (C) and Systems of Spheres

In our case, condition (PS) is the counterpart of (C) in the realm of systems of spheres.

Theorem 2
Let $*$ be a revision function satisfying the AGM postulates $(T*1)$–$(T*8)$, T a consistent complete theory of $\mathcal{L}$, and S_T the system of spheres centered on $[T]$, corresponding to $*$ by means of $(S*)$. Then $*$ satisfies (C) at T iff S_T satisfies (PS).

What is appealing about Theorem 2 is that it characterizes (C), not in terms of some technical non-intuitive condition, but rather by a natural constraint on similarity between interpretations, that in fact predates (C) and was motivated independently in a different context (Winslett 88). Moreover, as we will show in the next section, the essence of this characterization of (C) in terms of constraints on similarity, carries over into the general case of incomplete theories (albeit with some modifications).

4.2 The General Case

To elevate Theorem 2 to the general case, we need to extend the definition of $Diff$ to cover comparisons between an interpretation w and an arbitrary, possibly incomplete, theory T. Our generalization of $Diff$ is based on the comparison between an interpretation w and a C-structure C which gives us the minimal subsets of atoms, wrt C, such that w does not satisfy the subtheories of T defined over these subsets (w is not a possible world of them), and it satisfies the subtheories of T defined over the complements of these subsets (w is a possible world of them). The usual distance definition between w and T (Winslett 88) wrt set inclusion fails to compute that.

Definition 7
Let C be a consistent C-structure constructed on the belief base B_T, and w an interpretation. $Diff(C, w) = min\{|\bigcup_{r\in R}| : R$ is a subset of $Cov(C)$ s.t., for some $\alpha \in \mathcal{L}\left(\bigcup_{r\in R}\right), Cn(B_T) \vdash \alpha$, and $w \vdash \neg\alpha$; and $\forall\alpha \in Cn(B_T \backslash (B_T \cap \mathcal{L}(\bigcup_{r\in R}))), w \vdash \alpha\}$.

Minimality is required for the elements of $Diff(C, w)$ as the coverings of C overlap and are not disjoint.

To guarantee the satisfaction of the minimal change principle, we use, to compute the difference between a (possibly incomplete) theory T of $\mathcal{L}$ and an interpretation w, only atomic C-structures constructed on minimal belief bases of T :

Definition 8
Let T be a consistent theory of $\mathcal{L}$ (possibly incomplete) and w an interpretation. $Diff$ $(T, w) = \bigcap Diff\ (C, w)$, s.t., C is an atomic C-structure of T constructed on B_T a minimal belief base of T.

It is not hard to verify that in the special case of a consistent complete theory, the above definition of $Diff$ collapses to the one given in Section 4.1.

From Example 1, Table 1 illustrates the computation of $Diff(T, w_i)$ for all $w_i \in \mathcal{M}_\mathcal{L} \setminus [T]$. In this table we are representing interpretations as sequences of literals rather

w_i	$Diff(C, w_i)$	$Diff(T, w_i)$
$w_1 = abc\overline{d}$	{{c,d}}	{c,d}
$w_2 = ab\overline{c}d$	{{a,b,c},{b,c,d}}	{b,c}
$w_3 = ab\overline{c}\overline{d}$	{{a,b,c},{b,c,d}}	{b,c}
$w_4 = a\overline{b}cd$	{{a,b,c}}	{a,b,c}
$w_5 = a\overline{b}c\overline{d}$	{{a,b,c,d}}	{a,b,c,d}
$w_6 = a\overline{b}\overline{c}d$	{{a,b}}	{a,b}
$w_7 = a\overline{b}\overline{c}\overline{d}$	{{a,b}}	{a,b}
$w_8 = \overline{a}bc\overline{d}$	{{c,d}}	{c,d}
$w_9 = \overline{a}b\overline{c}d$	{{a,b,c},{b,c,d}}	{b,c}
$w_{10} = \overline{a}b\overline{c}\overline{d}$	{{a,b,c},{b,c,d}}	{b,c}
$w_{11} = \overline{a}\overline{b}cd$	{{a,b,c},{b,c,d}}	{b,c}
$w_{12} = \overline{a}\overline{b}c\overline{d}$	{{b,c,d}}	{b,c,d}

TAB. 1 – $Diff(T, w_i)$ computation of Example1

than theories ; moreover the negation of a propositional variable p is denoted $\overline{p}$. $[T] = \{abcd, \overline{a}bcd, \overline{a}\overline{b}\overline{c}d, \overline{a}\overline{b}\overline{c}\overline{d}\}$.

If T is incomplete, then for any interpretation w such that $w \in [T]$, $Diff(T, w) = \emptyset$. Moreover, for any interpretation w', $Diff(T, w') \subseteq \bigcup_{w \in [T]} Diff(w, w')$.

4.2.1 Condition (C) and Systems of Spheres

(Peppas 04) showed that condition (PS) does not correspond to the revision by the LS model in the general case of arbitrary theories. Indeed, (PS) does not correspond to (C) for a system of spheres S_T related to a theory T which is not necessarily complete, since the LS model is a special case of the C-structure model. To see this, refer to the counter-example given in (Peppas 04).

Despite its failure to generalize, (PS) should not be disregarded altogether. It can still serve as a guide in formulating the appropriate counterpart(s) of (C) for the general case ; as we prove later in this section, the two general conditions (Q1) and (Q2) that correspond to (C) are both in the spirit of (PS) (and surprisingly, they collapse to (PS) in the special case of complete theories).

To formulate the conditions (Q1) and (Q2), we need to define concepts related to the notion of distance between an interpretation and an incomplete theory (Peppas 04).

Definition 9
Let w, w' be interpretations, and let T be a theory of $\mathcal{L}$. The interpretations w and w' are external T-duals iff $Diff(T, w) = Diff(T, w')$ and $w \cap (\mathcal{V} \setminus Diff(T, w)) = w' \cap (\mathcal{V} \setminus Diff(T, w'))$.

Multiple T-duals (external and internal ones as we will see later) add more structure to a system of spheres, and render condition (PS) too strong for the general case. The possibility of placing external T-duals in different spheres, opens up new ways of ordering interpretations that still induce containment property revision functions without

fulfilling entirely the demands of (PS). Let us elaborate on this point and define the notion of w'-cover :

Definition 10
Let T be a theory of $\mathcal{L}$, let w, w' be two interpretations s.t., $Diff\ (T, w) \subset Diff\ (T, w')$, and let w'' be an external T-dual of w. The interpretation w'' is the w'-cover for w at T denoted by $\vartheta_T(w, w')$, iff $w'' \cap Diff(T, w) = w' \cap Diff(T, w)$.

In Table 1, w_{11} is the w_4-cover for w_9 at T (T is the theory of Example 1).

The notion of "cover" will be used to weaken (PS). In particular, consider the condition (Q1) below :

($\mathbf{Q1}$). If $Diff(T, w) \subset Diff(T, w')$ then there is a sphere $V \in S_T$ that contains $\vartheta_T(w, w')$ but not w'.

Condition (Q1) formalizes the intuition mentioned earlier about weakening (PS) with the aid of external T-duals. It is not hard to show that (PS) entails (Q1), and that (Q1) collapses to (PS) when the initial theory T is complete. Moreover, (Q1) is strictly weaker than (PS).

Now, condition (Q1) alone does not suffice to guarantee the satisfaction of the containment property ; from something too strong for (C) (condition (PS)), we have now moved to something too weak. Consider the following counter-example given in (Peppas 04) : the language $\mathcal{L}$ is built over three propositional variables a, b, c, the initial theory T is $T = Cn(\{a \leftrightarrow b\})$, and the system of spheres S_T centered on $[T]$ is the following :

$$\begin{array}{ccccccc} abc & & & & & & \\ ab\bar{c} & & & & & & a\bar{b}\bar{c} \\ \bar{a}\bar{b}c & \leq & a\bar{b}c & \leq & \bar{a}bc & \leq & \bar{a}b\bar{c} \\ \bar{a}\bar{b}\bar{c} & & & & & & \end{array}$$

In this example all the interpretations outside $[T]$ (i.e. in $\mathcal{M}_\mathcal{L} \setminus [T]$) differ from $[T]$ on precisely the same propositional variables, namely on $\{a, b\}$. Consequently S_T satisfies (Q1) since its antecedent $Diff(T, w) \subset Diff(T, w')$ never holds for $w, w' \notin [T]$. Yet despite the compliance with (Q1), the revision function $*$ induced from S_T violates (C) at T (simply consider the revision of T by $a \wedge \neg b$).

To secure the correspondence with (C), condition (Q1) needs to be complimented with a second condition, (Q2). This second condition uses the notion of internal T-dual.

Definition 11
Let w, w' be interpretations, and let T be a theory of $\mathcal{L}$. The interpretations w and w' are internal T-duals iff $Diff(T, w) = Diff(T, w')$, and $w \cap Diff(T, w) = w' \cap Diff(T, w')$.

In Table 1, w_2, w_3, w_9, and w_{10} are internal T-duals.

Clearly, for any theory T and any two interpretations w, w', if w and w' are both internal and external T-duals, then they are identical.

We now proceed with the presentation of condition (Q2), which together with (Q1), brings about the correspondence with (C1). In the following condition T is an arbitrary consistent theory of $\mathcal{L}$, S_T is a system of spheres centered on $[T]$, and w, w' are interpretations.

($\mathbf{Q2}$). If w and w' are internal T-duals, then they belongs to the same spheres in S_T ; i.e., for any sphere $V \in S_T, w \in V$ iff $w' \in V$.

In the special case that T is complete, no interpretation w has internal or external T-duals (other than itself). Consequently, in that case (Q1) reduces to (PS), while (Q2) degenerates to a vacuous condition.

The promised correspondence between (C) and the two conditions (Q1) and (Q2) is given by the theorem below :

Theorem 3
Let $*$ be a revision function satisfying $(T*1)$–$(T*8)$. Let T be a consistent theory of $\mathcal{L}$, and S_T a system of spheres centered on $[T]$, that corresponds to $*$ by means of $(S*)$. Then $*$ satisfies (C) at T iff S_T satisfies (Q1)–(Q2).

It should be noted that in the case of overlapping theories T, the LS model cannot avoid the counter-intuitive effect of throwing away all non-tautological beliefs in T whenever the new information is inconsistent with T, regardless of whether these beliefs can be kept or not. For example, the system of spheres S_T centered on $[T]$ (T is the theory of Example 1), which is based on the LS model (see in (Peppas 04) for all details on S_T construction) is composed only of two spheres : the sphere $[T]$ and the sphere $\mathcal{M}_\mathcal{L}$, the set of all consistent complete theories of $\mathcal{L}$. However, systems of spheres based on the C-structure model (satisfying the two conditions (Q1) and (Q2)) allows us to avoid such undesirable systems of spheres. To see this, consider one system of spheres S'_T corresponding to the theory T of Example 1, and satisfying the two conditions (Q1) and (Q2) as given below. Then, consider the revision of T by $a \wedge \neg b$ using S_T and S'_T. By S_T, the result is $Cn(a \wedge \neg b)$.

$$
\begin{array}{ccccccc}
 & & w_1 : abc\overline{d} & & & & \\
 & & w_2 : ab\overline{c}d & & & & \\
abcd & & w_3 : ab\overline{c}\,\overline{d} & & & & \\
\overline{a}bcd & & w_6 : a\overline{b}\overline{c}d & & w_4 : a\overline{b}cd & & \\
\overline{a}\overline{b}\overline{c}d & \leq & w_7 : a\overline{b}\overline{c}\,\overline{d} & \leq & w_{12} : \overline{a}\overline{b}c\overline{d} & \leq & w_5 : a\overline{b}c\overline{d} \\
\overline{a}\overline{b}\overline{c}\,\overline{d} & & w_8 : \overline{a}bc\overline{d} & & & & \\
 & & w_9 : \overline{a}b\overline{c}d & & & & \\
 & & w_{10} : \overline{a}b\overline{c}\,\overline{d} & & & & \\
 & & w_{11} : \overline{a}\overline{b}cd & & & &
\end{array}
$$

Theorems 1 and 3 provide immediately the following theorem that provides semantics for the containment property.

Theorem 4
Let $*$ be a revision function satisfying the AGM postulates $(T*1)$–$(T*8)$. Let T be a consistent theory of $\mathcal{L}$, and S_T a system of spheres centered on $[T]$, that corresponds to $*$ by means of $(S*)$. Then $*$ satisfies the containment property iff S_T satisfies (Q1)–(Q2).

5 Conclusion

In this paper, based on Grove's system of spheres construction, we provided semantics for local revision using the C-structure model by considering additional constraints

on measuring distance between interpretations. We also proved these constraints characterize precisely the containment property.

In future work we intend to carry out a thorough study of this property by generalizing our results to belief merging.

Références

[Alchourrón 85] C. E. Alchourrón, P. Gärdenfors & D. Makinson. *On the Logic of Theory Change : Partial Meet Contraction and Revision Functions.* J. Symb. Logic, vol. 50, no. 2, pages 510–530, 1985.

[Chopra 00] S. Chopra & R. Parikh. *Relevance sensitive belief structures*. Annals of Mathematics and Artificial Intelligence, vol. 28, no. 1-4, pages 259–285, 2000.

[Chopra 01a] S. Chopra, K. Georgatos & R. Parikh. *Relevance Sensitive Non-Monotonic Inference on Belief Sequences*. J. of Applied Non-Classical Logics, vol. 11, no. 1-2, pages 131–150, 2001.

[Chopra 01b] S. Chopra, R. Parikh & R. Wassermann. *Approximate belief revision*. Logic J. IGPL, vol. 9, no. 6, pages 755–768, 2001.

[Doukari 07a] O. Doukari & R. Jeansoulin. *Space-contained conflict revision, for geographic information*. In 10th AGILE International Conference on Geographic Information Science, Aalborg (Danmark), march 2007.

[Doukari 07b] O. Doukari, E. Würbel & R. Jeansoulin. *A New Model for Belief Representation and Belief Revision Based on Inconsistencies Locality*. In 19th IEEE International Conference on Tools with Artificial Intelligence, ICTAI'07., pages 262–269, Patras, Greece, October 29-31 2007. IEEE Computer Society.

[Grove 88] A. Grove. *Two modelings for theory change. Journal of Philosophical Logic*. Philosophical Logic, vol. 17, pages 157–170, 1988.

[Herzig 99] Andreas Herzig & Omar Rifi. *Propositional Belief Base Update and Minimal Change*. Artificial Intelligence, vol. 115, no. 1, pages 107–138, 1999.

[Kourousias 07] G. Kourousias & D. Makinson. *Parallel interpolation, splitting, and relevance in belief change* . J. Symb. Logic, vol. 72, no. 3, pages 994–1002, 2007.

[Parikh 99] R. Parikh. *Beliefs, belief revision, and splitting languages*. Logic, language and computation, vol. 2, pages 266–278, 1999.

[Peppas 00] P. Peppas, N. Y. Foo & A. C. Nayak. *Measuring similarity in belief revision.* J. of Logic and Computation, vol. 10, no. 4, pages 603 – 619, 2000.

[Peppas 04] P. Peppas, S. Chopra & N. Y. Foo. *Distance Semantics for Relevance-Sensitive Belief Revision*. In 9th International Conference on Principles of Knowledge Representation and Reasoning, KR'2004., pages 319–328, Canada, June 2004.

[Winslett 88] M. A. Winslett. *Reasoning about action using a possible models approach*. In AAAI'88 : In Proc. of the 7th AAAI conference, pages 89–93, San Francisco, CA, USA, 1988. Morgan Kaufmann Publishers.

Preuve Dialectique dans les Systèmes d'Argumentation à Contrainte

Caroline Devred[1], Sylvie Doutre[2]

[1] LERIA - Université d'Angers - devred@info.univ-angers.fr

[2] IRIT - Université Toulouse 1 - doutre@irit.fr

Résumé : Le système d'argumentation à contrainte généralise le cadre du système d'argumentation classique de Dung, et permet de capturer les sémantiques d'acceptabilité existantes dans ce cadre et dans ses extensions. Nous cherchons dans cet article à déterminer si un argument donné appartient à au moins un ensemble d'arguments acceptable sous la sémantique dite préférée, étendue au cadre du système d'argumentation à contrainte. La réponse que nous apportons prend la forme d'un dialogue qui explique pourquoi l'argument est ou n'est pas acceptable. Ce dialogue, pour les systèmes d'argumentation à contrainte, s'inscrit dans une généralisation d'un cadre dialectique défini pour les systèmes d'argumentation classique.

Mots-clés : Raisonnement, argumentation, dialogue

1 Introduction

L'argumentation est une approche générale pour le raisonnement non monotone, dont la caractéristique principale est l'utilisation d'arguments pour dériver des conclusions basées sur la façon dont les arguments interagissent. De nombreuses théories de l'argumentation ont été proposées (voir Besnard & Hunter (2008) pour une synthèse), parmi lesquelles le système d'argumentation de Dung (1995). Ce système est basé sur une définition abstraite des arguments et sur une relation binaire qui exprime une notion de contrariété entre arguments. Ce niveau d'abstraction permet au système de Dung de capturer de nombreuses approches pour l'inférence non-monotone et la programmation logique (Bondarenko *et al.* (1997)). Plusieurs travaux ont étendu le cadre de Dung, par exemple par l'ajout d'une relation de support entre arguments, ou de relations de préférences (voir par exemple Cayrol & Lagasquie-Schiex (2005); Baroni *et al.* (2000)).

L'une des questions les plus importantes en argumentation concerne la définition de l'acceptabilité d'ensembles d'arguments, appelés extensions. Dans le cadre de Dung, toutes les définitions des extensions sont basées uniquement sur les interactions entre arguments. Coste-Marquis *et al.* (2006) et Devred (2006) ont proposé une variante du système de Dung qui prend en compte une information supplémentaire : une contrainte. Cette contrainte permet d'exprimer, par exemple, le fait que si un argument appartient à une extension, alors un autre argument donné ne doit pas lui appartenir, alors même

que la relation de contrariété n'exprime aucun conflit entre ces deux arguments. La contrainte est exprimée par une formule propositionnelle sur l'alphabet des arguments. Coste-Marquis *et al.* (2006) et Devred (2006) ont montré que l'ajout de cette contrainte permet à leur système de capturer non seulement le système de Dung, mais également de nombreux systèmes dérivés de celui-ci. Ce nouveau formalisme s'avère donc très puissant. Il a par ailleurs récemment été appliqué dans le cadre du *practical reasoning* par Amgoud *et al.* (2008).

Une question fréquemment posée en complément de la définition des extensions, est celle de l'appartenance d'un argument donné, ou plus généralement de l'inclusion d'un ensemble d'arguments donné, dans une extension (on parle alors d'acceptabilité crédule), voire dans toute extension d'un système d'argumentation (on parle d'acceptabilité sceptique). Une réponse au problème qui concerne l'appartenance d'un argument à au moins une extension préférée a été proposée dans le cadre de Dung par Cayrol *et al.* (2003). Cette réponse prend la forme d'un dialogue entre deux participants, l'un cherchant à montrer l'existence d'une extension qui contient l'argument, l'autre cherchant à montrer que l'ensemble que tente de construire le premier ne peut être une extension. Le cadre dialectique dans lequel sont définis ces types de dialogues a été étendu par Doutre & Mengin (2004) et Devred & Doutre (2007) à l'acceptabilité d'un ensemble d'arguments. C'est une généralisation de ce cadre dialectique aux systèmes d'argumentation à contrainte, et à la notion d'extension préférée étendue à ces systèmes, que nous proposons dans cet article. A noter que d'autres travaux se sont attachés à définir une réponse sous forme d'un dialogue, tels que Dunne & Bench-Capon (2003) ; cependant, les dialogues de ces travaux cherchent à résoudre un conflit d'opinion entre participants, alors que le cadre dialectique de Cayrol *et al.* (2003) ne cherche qu'à prouver l'acceptabilité d'arguments.

L'article est organisé comme suit : en section 2, nous présentons le cadre de Dung et les systèmes d'argumentation à contrainte. En section 3, nous présentons une extension du cadre dialectique de Cayrol *et al.* (2003) aux systèmes d'argumentation à contrainte. Cette extension est utilisée dans la section 4 pour répondre au problème d'acceptabilité crédule d'un argument sous la sémantique $\mathcal{C}$-préférée. Nous concluons en section 5.

2 Les systèmes d'argumentation

2.1 Le cadre de Dung (1995)

Nous présentons ici brièvement le cadre de Dung pour l'argumentation.

Définition 1
Dung (1995) Un **système d'argumentation fini** *est une paire* $AF = \langle A, R \rangle$ *où :*
- A *est un ensemble fini d'objets, les* **arguments**,
- $R \subseteq A \times A$ *est une relation binaire sur* A*, la* **relation d'attaque**.

Pour deux arguments b et c, b attaque c suivant la relation R s'écrit bRc ou encore $(b, c) \in R$. On dit qu'un ensemble S d'arguments attaque un argument a suivant la relation R, s'il existe un élément de S qui attaque a suivant la relation R. Un argument

a est dit attaqué s'il existe un argument qui l'attaque suivant la relation R. Un ensemble d'arguments est attaqué s'il existe un argument de cet ensemble qui est attaqué.

Un système d'argumentation $AF = \langle A, R\rangle$ peut se représenter sous la forme d'un graphe orienté. Les sommets représentent les arguments et les arcs figurent les attaques. **Exemple :** $AF = \langle A, R\rangle$ avec $A = \{a, b, c, d, e, f, g, h, i\}$ et $R = \{(a, b), (b, d), (d, i), (i, h), (a, c), (c, e), (e, f), (f, e), (f, g), (g, h)\}$. Le graphe de AF est représenté figure 1.

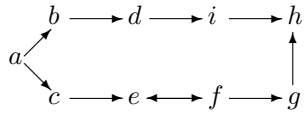


FIG. 1 – Représentation graphique de AF

Définition 2
Dung (1995) Soit $AF = \langle A, R\rangle$ un système d'argumentation. $S \subseteq A$ est un **ensemble sans conflit** *si et seulement s'il n'existe pas $a \in S$ et $b \in S$ tels que $(a, b) \in R$; $a \in A$ est* **acceptable par rapport à** *$S \subseteq A$ si et seulement si $\forall b \in A$: si $(b, a) \in R$ alors il existe $c \in S$ tel que $(c, b) \in R$; $S \subseteq A$ est* **admissible** *si et seulement si S est sans conflit et $\forall a \in S$, a est acceptable par rapport à S*

Exemple : (suite) $\{a, b\}$ n'est pas sans conflit, $\{a, d\}$ l'est. d est acceptable par rapport à $\{a\}$. $\{a, d, e\}$ est admissible.

2.2 Les systèmes d'argumentation à contrainte

Devred (2006) et Coste-Marquis *et al.* (2006) ont introduit les systèmes d'argumentation à contrainte, i.e. des systèmes d'argumentation « à la Dung » dans lesquels on introduit une contrainte que les ensembles acceptables d'arguments (les extensions) doivent respecter.

Définition 3
Coste-Marquis et al. *(2006) Soit $PROP_{PS}$ un langage propositionnel défini de la manière inductive habituelle sur un ensemble PS de symboles propositionnels, les constantes booléennes $\top$, $\bot$, et les connecteurs $\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$.*

Un **système d'argumentation à contrainte** *est un triplet $CAF = \langle A, R, \mathcal{C}\rangle$ où :*
- *A est un ensemble fini d'objets, les* **arguments** *;*
- *$R \subseteq A \times A$ est une relation binaire sur A, la* **relation d'attaque** *;*
- *$\mathcal{C}$ est une formule propositionnelle de $PROP_A$, la* **contrainte**[1].

[1]Par abus de langage, A, l'ensemble des arguments, est considéré comme un ensemble de symboles propositionnels dans $PROP_A$.

Nous faisons l'hypothèse que la contrainte $\mathcal{C}$ est sous forme normale conjonctive (CNF). Etant donné qu'il est établi que toute formule admet une formule sous forme CNF qui lui est logiquement équivalente, cette hypothèse est non réductrice.

Etant donné une contrainte $\mathcal{C}$ définie sur $PROP_A$, $\mathcal{C}^+$ (resp. $\mathcal{C}^-$) dénote l'ensemble des littéraux positifs (resp. négatifs) de $\mathcal{C}$. Il est possible que $\mathcal{C}^+ \cap \mathcal{C}^- \neq \emptyset$.

Dans toute la suite, par abus de langage, nous dirons que la conjonction d'une contrainte $\mathcal{C}$ et d'autres littéraux **ne conduit pas à l'inconsistance**, si la propagation unitaire des autres littéraux sur $\mathcal{C}$ n'amène pas la clause vide.

Exemple : (suite) Nous étendons le système d'argumentation AF de la figure 1 avec une contrainte, $\mathcal{C} = \neg a \vee \neg d \vee \neg e$.

Chaque sous-ensemble S de A correspond à une interprétation sur A donnée par la complétion de S.

Définition 4
Coste-Marquis et al. *(2006) Soit $CAF = \langle A, R, \mathcal{C}\rangle$ un système d'argumentation à contrainte et $S \subseteq A$. S* **satisfait** *$\mathcal{C}$ si et seulement si la* **complétion**

$$\widehat{S} = \{a \mid a \in S\} \cup \{\neg a \mid a \in A \setminus S\}$$

de S est un modèle de $\mathcal{C}$ (noté $\widehat{S} \models \mathcal{C}$).

Exemple : (suite) L'ensemble $\{a, d\}$ satisfait $\mathcal{C}$, car $\widehat{\{a, d\}} = \{a, \neg b, \neg c, d, \neg e, \neg f, \neg g, \neg h, \neg i\}$ et $\widehat{\{a, d\}} \models \mathcal{C}$.

L'idée est de restreindre les ensembles d'arguments qui pourraient constituer des ensembles admissibles en ne gardant que ceux qui satisfont la contrainte $\mathcal{C}$.

Définition 5
Coste-Marquis et al. *(2006) Soit $CAF = \langle A, R, \mathcal{C}\rangle$ un système d'argumentation à contrainte. Un sous-ensemble S de A est* **$\mathcal{C}$-admissible** *pour CAF si et seulement si S est admissible pour $\langle A, R\rangle$ et S satisfait $\mathcal{C}$. Un ensemble $\mathcal{C}$-admissible $S \subseteq A$ de CAF est une* **$\mathcal{C}$-extension préférée** *de CAF si et seulement si $\nexists S' \subseteq A$ tel que $S \subset S'$ et S' est $\mathcal{C}$-admissible pour CAF.*

Exemple : (suite) L'ensemble $\{a, d, e\}$ est admissible, mais il n'est pas $\mathcal{C}$-admissible, car il ne satisfait pas la contrainte $\mathcal{C}$. $\{a, e\}$ est $\mathcal{C}$-admissible. $\{a, d, f, h\}$ et $\{a, e, g\}$ sont les $\mathcal{C}$-extensions préférées de CAF.

Un problème important en argumentation consiste à déterminer si un argument (ou un ensemble d'arguments) donné est acceptable étant donnée une certaine définition des extensions (autrement dit, une sémantique). Pour les $\mathcal{C}$-extensions préférées, ce problème est défini de la manière suivante :

Définition 6
Soit $CAF = \langle A, R, \mathcal{C}\rangle$ un système d'argumentation à contrainte. Soit $S \subseteq A$ un ensemble d'arguments.

- S *est* crédulement accepté *sous la sémantique $\mathcal{C}$-préférée ssi S est inclus dans au moins une $\mathcal{C}$-extension préférée de $\langle A, R, \mathcal{C}\rangle$.*
- S *est* sceptiquement accepté *sous la sémantique $\mathcal{C}$-préférée ssi S est inclus dans toute $\mathcal{C}$-extension préférée $\langle A, R, \mathcal{C}\rangle$.*

Un argument $a \in A$ est dit crédulement (resp. sceptiquement) accepté sous la sémantique $\mathcal{C}$-préférée ssi l'ensemble $\{a\}$ l'est.

Coste-Marquis *et al.* (2006) a montré que, comme pour la sémantique préférée dans le cadre classique de Dung, déterminer si un ensemble d'arguments S est crédulement accepté, est un problème NP-complet ; déterminer si S est sceptiquement accepté est un problème Π_2^p-complet.

Nous nous concentrons dans cet article sur le problème qui consiste à déterminer si un argument est crédulement accepté sous la sémantique $\mathcal{C}$-préférée. Nous allons pour cela utiliser le résultat suivant :

Proposition 1
Coste-Marquis et al. *(2006) Soit $CAF = \langle A, R, \mathcal{C}\rangle$ un système d'argumentation à contrainte. Pour chaque ensemble $\mathcal{C}$-admissible S de CAF, il existe une $\mathcal{C}$-extension préférée E de CAF telle que $S \subseteq E$.*

Déterminer si un argument appartient à une $\mathcal{C}$-extension préférée revient donc à déterminer s'il appartient à au moins un ensemble $\mathcal{C}$-admissible.

Pour répondre à ce problème, Coste-Marquis *et al.* (2006) suggère de s'appuyer sur une traduction du système d'argumentation à contrainte en logique propositionnelle. Cette méthode a l'avantage de permettre l'utilisation de prouveurs SAT. L'inconvénient majeur est que la réponse retournée par ces prouveurs ne fera pas apparaître les raisons qui font que l'argument est ou n'est pas acceptable. Or c'est là l'essence même du processus d'argumentation : montrer la façon dont l'argument est contrarié, se défend, comment il respecte la contrainte.

Ce dernier aspect a été pris en compte par Cayrol *et al.* (2003); Doutre & Mengin (2004); Devred & Doutre (2007) dans un cadre dialectique pour la sémantique préférée de Dung. Nous allons étendre ce cadre aux systèmes d'argumentation à contrainte et à la sémantique $\mathcal{C}$-préférée.

3 Cadre dialectique

Une généralisation des cadres dialectiques de Cayrol *et al.* (2003); Doutre & Mengin (2004) a été proposée dans Devred & Doutre (2007). C'est une extension de ce dernier cadre dialectique que nous proposons ici.

Une preuve dialectique est formalisée par un dialogue entre deux joueurs, PRO et OPP. Un dialogue se déroule dans un système d'argumentation à contrainte donné, et est régi par des règles exprimées dans une fonction appelée coup-légal.

Etant donné un ensemble A, A^* dénote l'ensemble des séquences finies d'éléments de A. Nous étendons l'ensemble des arguments avec un argument "vide" (qui n'a de valeur que purement syntaxique), que nous dénotons _. Cet argument va pouvoir être

utilisé par un joueur pour exprimer le fait qu'il ne peut avancer aucun argument de l'ensemble A pour répondre à l'autre joueur. L'ensemble $A \cup \{\_\}$ est dénoté A^-.

Définition 7
Un type de dialogue à contrainte *est un tuple* $(A, R, \mathcal{C}, \phi)$*, où* $\langle A, R, \mathcal{C}\rangle$ *est un système d'argumentation à contrainte et* $\phi : A^{-*} \rightarrow 2^{A^-}$ *est une fonction appelée* fonction coup-légal*. Un* coup *dans* A *est une paire* $[P, x]$ *où* $P \in \{\mathrm{PRO}, \mathrm{OPP}\}$ *et* $x \in A$*. Pour un coup* $\mu = [P, x]$*, on utilise* $\mathrm{pl}(\mu)$ *pour dénoter* P *et* $\arg(\mu)$ *pour dénoter* x*.*

Un dialogue d *pour un argument* $x \in A$ *dans* $(A, R, \mathcal{C}, \phi)$ *(ou* ϕ*-dialogue) est une séquence dénombrable* $\mu_0\mu_1 \ldots$ *de coups dans* A *telle que :*

1. *le premier coup est joué par* PRO *pour avancer* x
2. *les coups suivants sont joués alternativement par* OPP *et* PRO
3. $\forall i \geq 0, \arg(\mu_{i+1}) \in \phi(\arg(\mu_0) \ldots \arg(\mu_i))$

On dit que d porte sur x *(i.e.* $\arg(\mu_0)$*).*

Lorsque l'ensemble des arguments retourné par ϕ est vide, le dialogue ne peut être continué.

Soit $d = \mu_0\mu_1 \ldots \mu_i$ un ϕ-dialogue fini :
– μ_i est dénoté par $\mathrm{last}(d)$;
– $\phi(\arg(\mu_0) \ldots \arg(\mu_i))$ est dénoté par $\phi(d)$;
– $\mathrm{PRO}(d)$ (resp. $\mathrm{OPP}(d)$) dénote l'ensemble des arguments avancés par PRO (resp. OPP) dans d ;

Définition 8
Etant donné un type de dialogue à contrainte $(A, R, \mathcal{C}, \phi)$*, un* ϕ*-dialogue fini* d *est* gagné par PRO *ssi le dernier coup est joué par* OPP *et contient l'argument vide, i.e.* $\mathrm{last}(d) = [\mathrm{OPP}, \_]$.

Dans la section suivante, nous allons montrer comment utiliser ce cadre dialectique pour répondre au problème d'acceptabilité crédule sous la sémantique $\mathcal{C}$-préférée.

4 Acceptabilité crédule

Le problème d'acceptabilité crédule auquel nous cherchons à répondre est celui de l'appartenance d'un argument x à au moins une $\mathcal{C}$-extension préférée. Nous avons montré en section 2 que cela revient à trouver un ensemble $\mathcal{C}$-admissible qui contient x.

Tout d'abord, notons que le problème est trivial si x s'attaque lui-même ou si $x \wedge \mathcal{C}$ conduit à l'inconsistance ; un tel argument ne peut appartenir à aucun ensemble $\mathcal{C}$-admissible. Si x n'est pas dans ce cas trivial, nous allons construire une preuve sous forme de dialogue. Si x est crédulement accepté, la preuve va mettre en évidence un ensemble $\mathcal{C}$-admissible qui contient x, ainsi que les attaquants de x et de l'ensemble $\mathcal{C}$-admissible, la façon dont l'ensemble se défend contre ces attaquants, ainsi que la façon dont la contrainte $\mathcal{C}$ est respectée.

Les dialogues que nous avons définis dans la section précédente nous permettent de distinguer les arguments qui défendent l'argument x contre ses attaquants et qui permettent à l'ensemble construit autour de x de satisfaire la contrainte (ces arguments seront joués par PRO), de ceux qui attaquent l'ensemble (ils seront joués par OPP). Dans le dialogue, chaque coup joué par OPP va faire suite à l'un des coups précédemment joué par PRO tel que l'argument de OPP attaque l'argument de PRO. Si un tel argument n'existe pas, OPP va jouer l'argument vide. Les arguments joués par PRO répondent quant à eux au tout dernier argument avancé par OPP. Tant que l'argument de OPP n'est pas l'argument vide, l'argument de PRO l'attaque et doit respecter un certain nombre de contraintes que nous allons définir un peu plus loin dans l'ensemble POSS. Si l'argument de OPP est l'argument vide, l'ensemble des arguments joués par PRO sera, nous le verrons, admissible. Il ne reste alors qu'à vérifier si cet ensemble satisfait la contrainte pour être $\mathcal{C}$-admissible. Si tel n'est pas le cas, PRO joue un des arguments positifs de la contrainte qui satisfait POSS.

Le dialogue s'arrête lorsque le dernier argument de OPP est l'argument vide et l'ensemble des arguments joués par PRO satisfait la contrainte ; l'ensemble des arguments joués par PRO est alors $\mathcal{C}$-admissible.

Pour présenter l'ensemble POSS et la fonction coup-légal, nous introduisons plusieurs notations : étant donné un système d'argumentation à contrainte $\langle A, R, \mathcal{C}\rangle$, soit $x \in A$ et $S \subseteq A$.

- $\text{Refl} = \{x \in A \mid (x, x) \in R\}$
- $R^+(S) = \{y \in A \mid \exists x \in S \text{ tel que } (x, y) \in R\}$
- $R^-(S) = \{y \in A \mid \exists x \in S \text{ tel que } (y, x) \in R\}$
- $R^\pm(S) = R^+(S) \cup R^-(S)$

De plus, $R^+(\{\_\}) = R^-(\{\_\}) = \emptyset$.

Soit d un ϕ-dialogue fini. Si ce dialogue doit mener à la construction d'un ensemble $\mathcal{C}$-admissible, et donc à un ensemble admissible, alors, comme dans Cayrol *et al.* (2003), les arguments joués par PRO ne doivent pas contenir les arguments qui s'auto-attaquent (i.e. Refl), ni ceux qui attaquent ou sont attaqués par des arguments joués par PRO (i.e. $R^\pm(\text{PRO}(d))$). De plus, PRO ne doit pas avoir à répéter un argument qu'il a déjà avancé ($\text{PRO}(d)$) puisque cet argument doit déjà avoir été montré acceptable. Pour résumer, les arguments joués par PRO doivent appartenir à l'ensemble $A \setminus (\text{PRO}(d) \cup \text{Refl} \cup R^\pm(\text{PRO}(d)))$ (condition γ).

En outre, parce que l'ensemble des arguments joués par PRO doit satisfaire la contrainte, les arguments joués par PRO ne doivent pas mener à l'inconsistance, c'est-à-dire, étant donné un argument x satisfaisant la condition γ, $\forall x' \in (\text{PRO}(d) \cup \{x\})$, $\forall x'' \in (\text{OPP}(d) \cup R^\pm(\text{PRO}(d) \cup \{x\}))$, $\bigwedge x' \wedge \bigwedge \neg x'' \wedge \mathcal{C}$ ne conduit pas à l'inconsistance.

Tous ces élements sont exprimés dans l'ensemble suivant :

$$\begin{aligned}\text{POSS}(d) = \{x \mid\ & x \in A \setminus (\text{PRO}(d) \cup \text{Refl} \cup R^\pm(\text{PRO}(d))) \text{ et} \\ & \forall x' \in (\text{PRO}(d) \cup \{x\}), \\ & \forall x'' \in (\text{OPP}(d) \cup R^\pm(\text{PRO}(d) \cup \{x\})), \\ & \textstyle\bigwedge x' \wedge \bigwedge \neg x'' \wedge \mathcal{C} \text{ ne conduit pas à l'inconsistance}\}\end{aligned}$$

La fonction coup légal ϕ suivante va nous permettre de définir les dialogues de preuve pour répondre au problème d'acceptabilité crédule :

Définition 9
Etant donné un système d'argumentation à contrainte $\langle A, R, \mathcal{C}\rangle$, soit $\phi : A^{-*} \rightarrow 2^{A^-}$ défini par :

– *si d est un dialogue de longueur impaire (c'est au tour de OPP de jouer),*

$$\phi(d) = \begin{cases} \{\_\} & \text{si } R^-(\mathrm{PRO}(d)) \setminus R^+(\mathrm{PRO}(d)) = \emptyset \\ R^-(\mathrm{PRO}(d)) \setminus R^+(\mathrm{PRO}(d)) & \text{sinon} \end{cases}$$

– *si d est un dialogue de longueur paire (c'est au tour de PRO de jouer),*

$$\phi(d) = \begin{cases} \mathrm{POSS}(d) \cap \mathcal{C}^+ & \text{si } \arg(\mathrm{last}(d)) = \_ \text{ et } \widehat{\mathrm{PRO}(d)} \not\models \mathcal{C} \\ \mathrm{POSS}(d) \cap R^-(\{\arg(\mathrm{last}(d))\}) & \text{sinon} \end{cases}$$

A noter qu'un dialogue fini, défini avec cette fonction coup-légal, aura toujours un dernier coup joué par OPP.

Définition 10
Soit un système d'argumentation à contrainte $\langle A, R, \mathcal{C}\rangle$. Soit $x \in A$ un argument tel que $x \notin \mathrm{Refl}$ et $x \wedge \mathcal{C}$ ne conduit pas à l'inconsistance. Une ϕ-preuve pour x est un ϕ-dialogue pour x gagné par PRO *tel que $\mathrm{PRO}(d)$ satisfait la contrainte $\mathcal{C}$.*

Les résultats suivants établissent la correction et la complétude des ϕ-preuves. L'ensemble de ces résultats est démontré ; pour des raisons de place, nous ne présentons qu'une ébauche de ces démonstrations.

Proposition 2 (Correction des ϕ-preuves)
Si d est une ϕ-preuve pour un argument x, alors $\mathrm{PRO}(d)$ est un ensemble $\mathcal{C}$-admissible contenant x.

Lemme 1
Étant donné un type de dialogue $(A, R, \mathcal{C}, \phi)$ et un ϕ-dialogue d, $\mathrm{PRO}(d)$ est sans conflit et ne conduit pas à l'inconsistance.

Démonstration (Lemme 1) Il s'agit comme dans Cayrol *et al.* (2003) de faire une preuve par récurrence sur le nombre d'élément de $\mathrm{PRO}(d)$. La partie concernant l'admissibilité est similaire à celle de la preuve initiale, on vérifie juste à chaque fois que l'ensemble ne conduit pas à l'inconsistance. ■

Démonstration (Proposition 2) Le lemme 1 permet de prouver que l'on travaille sur un ensemble admissible et ne conduisant pas à l'inconsistance. De plus on sait que l'on arrête le dialogue lorsque la complétion de $\mathrm{PRO}(d)$ satisfait la contrainte. On peut donc prouver que $\mathrm{PRO}(d)$ est un ensemble $\mathcal{C}$-admissible. ■

Proposition 3 (Complétude des ϕ-preuves)
Si un argument x appartient à une $\mathcal{C}$-extension préférée d'un système d'argumentation à contrainte $CAF = \langle A, R, C\rangle$ tel que A est fini, alors il existe une ϕ-preuve pour x.

Lemme 2
Soit $(A, R, \mathcal{C}, \phi)$ un type de dialogue et un ϕ-dialogue dont le dernier coup est joué par PRO. *Soit $S \subseteq A$ un ensemble minimal tel que S est $\mathcal{C}$-admissible et contient* $\mathrm{PRO}(d)$. *Si $S \neq \mathrm{PRO}(d)$ et* $\mathrm{PRO}(d)$ *non admissible, alors il existe $x, y \in A$ tels que le dialogue $d' = d.[\mathrm{OPP}, x].[\mathrm{PRO}, y]$ est un ϕ-dialogue et S est minimal $\mathcal{C}$-admissible contenant* $\mathrm{PRO}(d')$.

Démonstration (Lemme 2) Il s'agit d'une adaptation de la preuve proposée dans Cayrol *et al.* (2003) avec, pour chaque ajout d'argument dans le dialogue, la vérification de la contrainte, conformément à la fonction coup-légal. ■

Démonstration (Proposition 3) Le lemme 2 nous permet de construire des dialogues dont l'ensemble des arguments joués par PRO est admissible et ne conduit pas à l'inconsistance à partir d'un ensemble $\mathcal{C}$-admissible. Nous partons comme dans Cayrol *et al.* (2003) d'un ensemble $\mathcal{C}$-admissible S contenant x. Pour chaque élément de l'ensemble, nous pouvons donc construire un dialogue dont l'ensemble des arguments joués par PRO est admissible et ne conduit pas à l'inconsistance. C'est à partir de la réunion de ces dialogues avec l'élimination des redondances et en se basant sur un ordre aléatoire des dialogues (avec comme contrainte que le dialogue pour x est le premier) que nous construisons la ϕ-preuve pour x. ■

Exemple : (suite) Construisons une preuve dialectique pour d (d ne s'auto-attaque pas et ne mène pas à l'inconsistance de $\mathcal{C}$).

$\mu_0 = [\mathrm{PRO}, d]$, $\quad \phi(\mu_0) = \{b\}$;
$\mu_1 = [\mathrm{OPP}, b]$, $\quad \phi(\mu_0\mu_1) = \{a\}$;
$\mu_2 = [\mathrm{PRO}, a]$, $\quad \phi(\mu_0\mu_1\mu_2) = \{\_\}$;
$\mu_3 = [\mathrm{OPP}, \_]$, $\quad \mathrm{PRO}(\widehat{\mu_0\mu_1\mu_2\mu_3}) \models \mathcal{C}, \phi(\mu_0\mu_1\mu_2\mu_3) = \emptyset$

Le dialogue $\delta = \mu_0\mu_1\mu_2\mu_3$ est gagné par PRO, $\mathrm{PRO}(\delta) = \{d, a\}$ est un ensemble $\mathcal{C}$-admissible. L'argument d appartient donc à une $\mathcal{C}$-extension préférée.

Exemple : Nous étendons le système d'argumentation de la figure 1 avec la contrainte $\mathcal{C} = (\neg a \vee \neg d \vee \neg e) \wedge (h \vee b)$. Construisons une preuve dialectique pour d (d ne s'auto-attaque pas et ne mène pas à l'inconsistance de $\mathcal{C}$).

$\mu_0 = [\mathrm{PRO}, d]$, $\quad \phi(\mu_0) = \{b\}$;
$\mu_1 = [\mathrm{OPP}, b]$, $\quad \phi(\mu_0\mu_1) = \{a\}$;
$\mu_2 = [\mathrm{PRO}, a]$, $\quad \phi(\mu_0\mu_1\mu_2) = \{\_\}$;
$\mu_3 = [\mathrm{OPP}, \_]$, $\quad \mathrm{PRO}(\widehat{\mu_0\mu_1\mu_2\mu_3}) \not\models \mathcal{C}, \phi(\mu_0\mu_1\mu_2\mu_3) = \{h\}$;
$\mu_4 = [\mathrm{PRO}, h]$ $\quad \phi(\mu_0\mu_1\mu_2\mu_3\mu_4) = \{g\}$;
$\mu_5 = [\mathrm{OPP}, g]$ $\quad \phi(\mu_0\mu_1\mu_2\mu_3\mu_4\mu_5) = \{f\}$;
$\mu_6 = [\mathrm{PRO}, f]$ $\quad \phi(\mu_0\mu_1\mu_2\mu_3\mu_4\mu_5\mu_6) = \{\_\}$;
$\mu_7 = [\mathrm{OPP}, \_]$ $\quad \mathrm{PRO}(\widehat{\mu_0\mu_1\mu_2\mu_3\mu_4\mu_5\mu_6\mu_7}) \models \mathcal{C}, \phi(\mu_0\mu_1\mu_2\mu_3\mu_4\mu_5\mu_6\mu_7) = \emptyset$

Le dialogue $\delta = \mu_0\mu_1\mu_2\mu_3\mu_4\mu_5\mu_6\mu_7$ est gagné par PRO, $\mathrm{PRO}(\delta) = \{d, a, h, f\}$ est un ensemble $\mathcal{C}$-admissible. L'argument d appartient donc à une $\mathcal{C}$-extension préférée.

5 Conclusion et perspectives

Le cadre dialectique et la théorie de la preuve présentés dans cet article pour les systèmes d'argumentation à contrainte généralisent les travaux de Cayrol *et al.* (2003), Doutre & Mengin (2004) et Devred & Doutre (2007).

Comme cela a pu être fait dans Devred & Doutre (2007), la théorie de la preuve peut aisément être étendue au problème qui concerne l'acceptabilité crédule d'un ensemble d'arguments, c'est-à-dire déterminer si un ensemble d'arguments donné est inclus dans au moins une $\mathcal{C}$-extension préférée. En perspective, nous envisageons également d'étendre les algorithmes de Cayrol *et al.* (2003) pour calculer les théories de la preuve pour les systèmes d'argumentation à contrainte.

Dans la continuité des travaux de Amgoud *et al.* (2008), nous comptons étudier comment cette théorie de la preuve peut s'appliquer dans le cadre du *practical reasoning*, en particulier pour le problème de la recherche des intentions.

Références

AMGOUD L., DEVRED C. & LAGASQUIE-SCHIEX M.-C. (2008). A constrained argumentation system for practical reasoning. In *AAMAS 2008*, p. 429–436.

BARONI P., GIACOMIN M. & GUIDA G. (2000). Extending abstract argumentation systems theory. *Artificial Intelligence*, **120**(2), 251–270.

BESNARD P. & HUNTER A. (2008). *Elements of Argumentation*. The MIT Press.

BONDARENKO A., DUNG P. M., KOWALSKI R. A. & TONI F. (1997). An abstract, argumentation-theoretic approach to default reasoning. *Artificial Intelligence*, **93**, 63–101.

CAYROL C., DOUTRE S. & MENGIN J. (2003). On decision problems related to the preferred semantics for argumentation frameworks. *Journal of Logic Computation*, **13**(3), 377–403.

CAYROL C. & LAGASQUIE-SCHIEX M.-C. (2005). On the acceptability of arguments in bipolar argumentation framework. In *Proceedings of* 8^{th} *European Conference on Symbolic and Quantitative Approache to Reasoning with Uncertainty (ECSQARU 2005)*, p. 378–389, Barcelona, Spain.

COSTE-MARQUIS S., DEVRED C. & MARQUIS P. (2006). Constrained argumentation frameworks. In *KR 2006*, p. 112–122.

DEVRED C. (2006). *Étude des inférences argumentatives : spécialisations et généralisations du cadre de Dung*. Thèse de doctorat, Université d'Artois, Lens.

DEVRED C. & DOUTRE S. (2007). Dialectical proof theories for the credulous prudent preferred semantics of argumentation. In *ECSQARU 2007*, p. 271–282.

DOUTRE S. & MENGIN J. (2004). On sceptical vs credulous acceptance for abstract argument systems. In *NMR 2004*, p. 134–139.

DUNG P. (1995). On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games. *Artificial Intelligence*, **77**(2), 321–358.

DUNNE P. & BENCH-CAPON T. (2003). Two party immediate response disputes : Properties and efficiency. *Artificial Intelligence*, **149**, 221–250.

Une inférence lexicographique à partir de bases de croyances partiellement pré-ordonnées

Safa Yahi[1], Salem Benferhat[1], Sylvain Lagrue[1],
Mariette Sérayet[2], Odile Papini[2]

[1] Université Lille-Nord de France,
Artois, F-62307 Lens, CRIL, F-62307 Lens
CNRS UMR 8188, F-62307 Lens
{yahi,benferhat,lagrue}@cril.univ-artois.fr

[2] LSIS-CNRS UMR 6168,
Université de la Méditerranée, ESIL,
163 av. de Luminy - 13288 Marseille Cedex 09. France
{serayet,papini}@esil.univmed.fr

Résumé : L'inférence lexicographique à partir de bases de croyances totalement pré-ordonnées, est intéressante d'un point de vue théorique, pratique et psychologique. Par ailleurs, les bases de croyances partiellement pré-ordonnées offrent plus de flexibilité afin de représenter efficacement des connaissances incomplètes. Dans cet article, nous proposons une inférence lexicographique à partir de bases de croyances partiellement pré-ordonnées qui étend l'inférence lexicographique classique. Nous proposons deux définitions équivalentes. La première, assez naturelle, consiste à appliquer l'inférence lexicographique classique sur toutes les bases totalement pré-ordonnées compatibles. La seconde revient à appliquer l'inférence classique sur les sous-bases cohérentes qui sont préférées par rapport à une nouvelle relation de préférence lexicographique. De plus, nous présentons une caractérisation sémantique de l'inférence proposée, nous montrons qu'elle satisfait les propriétés du système P et qu'elle est plus productive que les extensions possibiliste et celle basée sur l'inclusion.

Mots-clés : Raisonnement en présence d'incohérence, pré-ordre partiel, Inférence lexicographique.

1 Introduction

En Intelligence Artificielle, le raisonnement en présence d'incohérence est un problème important que l'on rencontre dans diverses situations telles que le raisonnement tolérant les exceptions, la révision de croyances, la fusion d'informations provenant de sources multiples, éventuellement conflictuelles, le raisonnement incertain ou en présence d'informations incomplètes, etc. Dans ce cas, l'inférence classique ne peut être

directement appliquée puisqu'à partir d'une base incohérente, on peut déduire n'importe quoi. Différentes approches ont été proposées pour raisonner en présence d'incohérence. Certaines consistent à affaiblir la relation d'inférence à l'instar des logiques paraconsistantes da Costa (1974), d'autres affaiblissent les croyances disponibles telles que les approches basées sur la restauration de la cohérence qui sont assez populaires. La plupart des approches basées sur la restauration de la cohérence Resher & Manor (1970) sont définies à partir de bases de croyances totalement pré-ordonnées comme l'inférence possibiliste Dubois *et al.* (1994), l'inférence linéaire Nebel (1994), l'inférence basée sur l'inclusion Brewka (1989) et l'inférence lexicographique Benferhat *et al.* (1993); Lehmann (1995). Par ailleurs, les bases de croyances partiellement pré-ordonnées offrent plus de flexibilité pour représenter efficacement des connaissances incomplètes, et permettent d'éviter de comparer des informations qui sont incomparables. En effet, dans de nombreuses applications, la relation de priorité associée aux croyances disponibles n'est que partiellement définie, et astreindre l'utilisateur à rajouter des priorités afin de les pré-ordonner d'une manière totale pourrait conduire à inférer des conclusions indésirables.

Cette flexibilité a motivé la définition de nouvelles approches basées sur la restauration de la cohérence qui sont dédiées aux bases de croyances partiellement pré-ordonnées, et ce, en général, par extension de celles qui sont définies dans le cadre totalement pré-ordonné. Par exemple, on peut citer l'extension de l'inférence possibiliste proposée dans Benferhat *et al.* (2004b) et l'extension de l'inférence basée sur l'inclusion donnée par Junker & Brewka (1989). Néanmoins, aucune extension n'a été proposée pour l'inférence lexicographique bien qu'elle soit dotée de propriétés intéressantes d'un point de vue théorique et pratique. En fait, elle est plus productive que l'inférences possibiliste et l'inférence basée sur l'inclusion. En outre, dans une étude psychologique réalisée dans Benferhat *et al.* (2004a) dans le contexte du raisonnement par défaut, il a été prouvé qu'elle est la plus intéressante par rapport aux autres relations d'inférence considérées dans la même étude.

Dans cet article, nous proposons une inférence lexicographique à partir de bases de croyances partiellement pré-ordonnées qui étend l'inférence lexicographique classique. Deux définitions équivalentes sont proposées. La première, assez naturelle, consiste à appliquer l'inférence lexicographique classique sur toutes les bases totalement pré-ordonnées compatibles. Quand à la seconde, elle revient à appliquer l'inférence classique sur les sous-bases cohérentes qui sont préférées par rapport à une nouvelle relation de préférence lexicographique. De plus, nous présentons une caractérisation sémantique de l'inférence proposée et nous montrons qu'elle satisfait les propriétés du système P.

L'article s'articule comme suit. Après une section préliminaire qui fixe les notations et rappelle l'inférence lexicographique classique, nous présentons en Section 3 la première définition de l'inférence lexicographique à partir de bases de croyances partiellement pré-ordonnées. Celle-ci consiste à appliquer l'inférence lexicographique classique sur toutes les bases totalement pré-ordonnées compatibles. La Section 4 présente une nouvelle relation de préférence lexicographique entre les sous-bases cohérentes d'une base de croyances partiellement pré-ordonnées. Nous proposons ensuite, en Section 5, la deuxième définition de l'inférence lexicographique qui revient à appliquer l'inférence classique sur les sous-bases cohérentes qui sont préférées, puis nous montrons

l'équivalence des deux définitions. La caractérisation sémantique de l'inférence lexicographique est présentée en Section 6 ainsi que quelques propriétés, en particulier, nous montrons qu'elle satisfait les propriétés du système P avant de conclure en Section 7.

2 Préléminaires

2.1 Notations

Soit V un ensemble fini de variables propositionnelles, notées par les lettres romaines miniscules $a, b, \ldots$ Le langage propositionnel, noté PL_V, est construit à partir de V, de $\{\top, \bot\}$ pour la tautologie et la contradiction respectivement, et des connecteurs usuels $\wedge, \vee, \neg, \rightarrow, \leftrightarrow$. Les formules de PL_V sont notées par les lettres grecques $\phi, \varphi, \psi, \ldots$ La relation d'inférence classique est notée $\vdash$. Soit Σ un ensemble fini de formules, l'ensemble de toutes les bases cohérentes de Σ est noté $CONS(\Sigma)$ et l'ensemble des toutes ses bases maximales (par rapport à l'inclusion ensembliste) cohérentes est noté $MCONS(\Sigma)$. L'ensemble des interprétations est noté $\mathcal{W}$. Soit ϕ une formule propositionnelle, $Mod(\phi)$ représente l'ensemble des modèles de ϕ.

Un pré-ordre partiel $\preceq$ sur un ensemble fini A est une relation binaire réflexive et transitive. $a \preceq b$ exprime que a est au moins aussi préféré que b. Un ordre strict $\prec$ sur A est une relation binaire irreflexive et transitive. $a \prec b$ signifie que a est strictement préféré à b. Il est défini comme suit : $a \prec b$ si et seulement si $a \preceq b$ est vérifié alors que $b \preceq a$ ne l'est pas. L'équivalence, notée par $\approx$, est définie par $a \approx b$ si et seulement si $a \preceq b$ et $b \preceq a$. De plus, nous définissons l'incomparabilité, notée $\sim$, par $a \sim b$ si et seulement si ni $a \preceq b$ ni $b \preceq a$ ne sont vérifiés. L'ensemble des éléments minimaux de A par rapport à $\prec$, noté $Min(A, \prec)$, est défini par : $Min(A, \prec) = \{a \in A, \nexists b \in A : b \prec a\}$. Un pré-ordre total $\leq$ sur un ensemble fini A est une relation binaire réflexive et transitive telle que $\forall a, b \in A$, $a \leq b$ ou $b \leq a$.

2.2 Un bref rappel sur l'inférence lexicographique classique

Une base de croyances totalement pré-ordonnées $(\Sigma, \leq)$ est un ensemble Σ de formules logiques classiques muni d'un pré-ordre total $\leq$ reflétant la relation de priorité existant entre les formules. $(\Sigma, \leq)$ peut être donnée sous forme d'une base stratifiée $(\Sigma, \leq) = S_1 \cup \cdots \cup S_m$ telle que les formules dans la strate S_i sont plus prioritaires que celles de la strate S_j pour $j > i$, avec $1 \leq i \leq m$, et $1 \leq j \leq m$.

De nombreuses approches basées sur la restauration de la cohérence ont été développées afin de raisonner à partir de bases de croyances totalement pré-ordonnées. Selon l'analyse de Pinkas & Loui (1992), l'inférence basée sur la restauration de cohérence est un processus à deux étapes qui consiste à générer dans un premier temps les sous-bases cohérentes préférées, puis à appliquer l'inférence classique sur certaines d'entre elles. Parmi ces approches, on peut citer l'inférence possibiliste Dubois *et al.* (1994), l'inférence basée sur l'inclusion Brewka (1989) et l'inférence lexicographique Benferhat *et al.* (1993); Lehmann (1995) autour de laquelle s'articule cet article.

La préférence lexicographique entre les sous-bases cohérentes d'une base de croyances totalement pré-ordonnées est définie comme suit :

Définition 1
Soit une base de croyances totalement pré-ordonnées $(\Sigma, \leq) = S_1 \cup \ldots \cup S_m$. Soient A et B deux sous-bases cohérentes de Σ. Alors, (i) $A <_{lex} B$ ssi $\exists i$, $1 \leq i \leq m$ tel que $|S_i \cap A| > |S_i \cap B|$ [1] et $\forall j$, $j < i$, $|S_j \cap B| = |S_j \cap A|$, (ii) $A =_{lex} B$ ssi $\forall i$, $1 \leq i \leq m : |S_i \cap A| = |S_i \cap B|$.

Dans la suite, l'ensemble de toutes les sous-bases cohérentes lexicographiquement préférées de Σ est noté $Lex(\Sigma, \leq)$ et $Lex(\Sigma, \leq) = Min(CONS(\Sigma), <_{lex})$. On peut facilement vérifier que chaque élément de $Lex(\Sigma, \leq)$ est une sous-base maximale cohérente de $(\Sigma, \leq)$. L'inférence lexicographique à partir de $(\Sigma, \leq)$ est donnée par la définition suivante :

Définition 2
Soit $(\Sigma, \leq)$ une base de croyances totalement pré-ordonnées et soit ψ une formule propositionnelle. ψ est une conséquence lexicographique de $(\Sigma, \leq)$, notée $\Sigma \vdash_{lex} \psi$, si et seulement si ψ est une conséquence classique de toute sous-base cohérente lexicographiquement préférée de $(\Sigma, \leq)$. Plus formellement : $(\Sigma, \leq) \vdash_{lex} \psi$ ssi $\forall B \in Lex(\Sigma, \leq) : B \vdash \psi$.

L'inférence basée sur l'inclusion, notée par $\vdash_{incl}$, est définie de façon analogue en remplaçant la comparaison basée sur la cardinalité par celle basée sur l'inclusion : $A <_{incl} B$ ssi $\exists i$, $1 \leq i \leq m$ tel que $(S_i \cap B) \subset (S_i \cap A)$ et $\forall j$, $j < i$, $(S_j \cap B) = (S_j \cap A)$.

Enfin, l'inférence possibiliste, notée par $\vdash_\pi$, revient à appliquer l'inférence classique à la base classique $\{\bigcup_{i=1}^{s-1} S_i\}$, où s est le plus petit indice tel que $\bigcup_{i=1}^{s} S_i$ est incohérente.

3 Une première définition de l'inférence lexicographique

Nous rappelons, dans un premier temps, la notion de bases totalement pré-ordonnées compatibles avec une base de croyances partiellement pré-ordonnées. Intuitivement, une base de croyances totalement pré-ordonnées $(\Sigma, \leq)$ est dite compatible avec une base de croyances partiellement pré-ordonnées $(\Sigma, \preceq)$ si et seulement si le pré-ordre total $\leq$ étend le pré-ordre partiel $\preceq$. Plus formellement : (i) $\forall \varphi, \phi \in \Sigma$: si $\varphi \preceq \phi$ alors $\varphi \leq \phi$, (ii) $\forall \varphi, \phi \in \Sigma$: si $\varphi \prec \phi$ alors $\varphi < \phi$.

Par la suite, l'ensemble de toutes les bases de croyances totalement pré-ordonnées compatibles avec $(\Sigma, \preceq)$ est noté $\mathcal{C}(\Sigma, \preceq)$.

Nous définissons maintenant une inférence lexicographique à partir de bases de croyances partiellement pré-ordonnées, et ce en s'appuyant sur la notion de bases de croyances totalement pré-ordonnées compatibles comme suit :

Définition 3
Soit $(\Sigma, \preceq)$ une base de croyances partiellement pré-ordonnées et soit ψ une formule propositionnelle. ψ est une conséquence $\mathcal{C}$-lexicographique de $(\Sigma, \preceq)$, notée $(\Sigma, \preceq$

[1] $|A|$ dénote le nombre de formules de A.

$) \vdash_{lex}^{\mathcal{C}} \psi$, si et seulement si ψ est une conséquence lexicographique classique de chaque base de croyances totalement pré-ordonnées compatible avec $(\Sigma, \preceq)$. Plus formellement : $(\Sigma, \preceq) \vdash_{lex}^{\mathcal{C}} \psi$ ssi $\forall (\Sigma, \leq) \in \mathcal{C}(\Sigma, \preceq) : (\Sigma, \leq) \vdash_{lex} \psi$.

Cette relation d'inférence est assez naturelle. En effet, la notion de bases de croyances totalement pré-ordonnées compatibles est très intuitive et a été considérée par d'autres auteurs (mais pas par rapport à l'inférence lexicographique) comme l'extension possibiliste donnée par Benferhat *et al.* (2003) et l'extension basée sur l'inclusion proposée par Junker & Brewka (1989). Nous illustrons l'inférence $\mathcal{C}$-lexicographique avec l'exemple suivant que utilisons tout au long de l'article.

Exemple 1
Cet exemple est issu d'une application du projet européen VENUS qui traite de la gestion d'informations archéologiques sous-marines. L'une des tâches concerne la mesure fondée sur la connaissance archéologique et nécessite la fusion d'informations provenant de plusieurs sources. Par exemple, plusieurs agents, munis d'appareils de mesure variés observent des amphores sur un site archéologique. Selon le premier agent, il s'agit d'une amphore Dressel 7 (d) dont la hauteur est 70 cm (h). Selon le second, l'amphore a une lèvre d'un diamètre de 15 cm (r) et une hauteur de 75 cm ($\neg h$). Selon le troisième, le diamètre de l'amphore est de 18 cm ($\neg r$). Le premier agent est moins fiable que les autres et on ne sait pas si le second agent est plus fiable que le troisième. Une autre source d'informations est l'information archéologique sur les amphores de type Dressel 7 qui ont été consignées dans un fichier XML par un quatrième agent qui est moins fiable que les trois autres. Selon le fichier XML, une Dressel 7 a une lèvre d'un diamètre compris entre 9.00 et 15.00 cm et une hauteur comprise entre 50.00 et 70.00 cm ($d \rightarrow r \wedge h$). L'ensemble de formules est $\Sigma = \{d \rightarrow r \wedge h, d, r, \neg h, \neg r, h\}$ et le pre-ordre partiel $\preceq$ sur Σ est donné par la figure 1, où l'arc "$a \rightarrow b$" représente $b \prec a$.

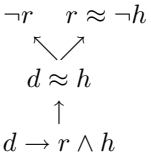


FIG. 1 – Le pré-ordre partiel $\preceq$ sur Σ

Les 3 pré-ordres totaux compatibles avec $\preceq$ sont $\leq^1 : \{\neg r =^1 r,\ \neg r =^1 \neg h,\ r =^1 \neg h,\ \neg r <^1 d,\ r <^1 d,\ \neg h <^1 d,\ d =^1 h,\ d <^1 d \rightarrow r \wedge h\}$, $\leq^2 : \{\neg r <^2 r,\ \neg r <^2 \neg h,\ r =^2 \neg h,\ r <^2 d,\ \neg h <^2 d,\ d =^2 h,\ d <^2 d \rightarrow r \wedge h\}$, $\leq^3 : \{r =^3 \neg h,\ r <^3 \neg r,\ \neg h <^3 \neg r,\ \neg r <^3 d,\ d =^3 h,\ d <^3 d \rightarrow r \wedge h\}$. Par manque de place nous ne donnons que les relations entre les sous-bases maximales cohérentes. Cela n'a aucune incidence sur la définition de la conséquence $\mathcal{C}$-lexicographique. L'ensemble des sous-bases maximales cohérentes de Σ est $MCONS(\Sigma) = \{A_1, A_2, A_3, A_4, A_5, A_6, A_7\}$ avec $A_1 = \{d \rightarrow r \wedge h, d, r, h\}$, $A_2 = \{d \rightarrow r \wedge h, \neg r, h\}$, $A_3 = \{d \rightarrow r \wedge h, \neg h, \neg r\}$, $A_4 = \{d \rightarrow r \wedge h, r, \neg h\}$, $A_5 = \{d, \neg r, h\}$, $A_6 = \{d, \neg h, \neg r\}$ et $A_7 = \{d, r, \neg h\}$. Les préferences $\leq_{lex}^1$, $\leq_{lex}^2$ et $\leq_{lex}^3$ sur $MCONS(\Sigma)$ par rapport aux bases de croyances compatibles $(\Sigma, \leq^1), (\Sigma, \leq^2)$ et $(\Sigma, \leq^3)$ respectivement sont $\leq_{lex}^1 : \{A_6 =_{lex}^1 A_7,\ A_6 <_{lex}^1 A_3,\ A_3 =_{lex}^1 A_4,\ A_3 <_{lex}^1 A_1,\ A_1 <_{lex}^1 A_5,\ A_5 <_{lex}^1 A_2\}$, $\leq_{lex}^2 : \{A_6 <_{lex}^2 A_3,\ A_3 <_{lex}^2 A_5,\ A_5 <_{lex}^2 A_2,\ A_7 <_{lex}^2 A_4,\ A_4 <_{lex}^2 A_1\}$ et $\leq_{lex}^3 : \{A_7 <_{lex}^2 A_4,\ A_4 <_{lex}^2 A_6,\ A_6 <_{lex}^2 A_3,\ A_3 <_{lex}^2 A_1,\ A_1 <_{lex}^2 A_5,\ A_5 <_{lex}^2 A_2\}$. Finalement, nous déduisons $\bigcup_{(\Sigma, \leq^i) \in \mathcal{C}(\Sigma, \preceq)} Lex(\Sigma, \leq^i) = \{A_6, A_7\} \cup \{A_6\} \cup \{A_7\} = \{A_6, A_7\}$, et par exemple, d est une consequence $\mathcal{C}$-lexicographique de Σ, puisque $A_6 \vdash d$ et $A_7 \vdash d$.

4 Une nouvelle préférence lexicographique

Nous proposons une nouvelle relation de préférence lexicographique entre les sous-bases cohérentes d'une base de croyances partiellement pré-ordonnées. Soit $(\Sigma, \preceq)$ une

base de croyances partiellement pré-ordonnées.

Dans un premier temps, nous partitionnons Σ de la façon suivante : $\Sigma = E_1 \cup \ldots \cup E_n$ $(n \geq 1)$ tel que : (i) $\forall i,\ 1 \leq i \leq n$, nous avons $\forall \varphi, \varphi' \in E_i : \varphi \approx \varphi'$, ; (ii) $\forall i,\ 1 \leq i \leq n, \forall j,\ 1 \leq j \leq n$ avec $i \neq j$, nous avons $\forall \varphi \in E_i, \forall \varphi' \in E_j : \varphi \not\approx \varphi'$.

En d'autres termes, chaque sous ensemble E_i représente une classe d'équivalence de Σ par rapport à la relation d'équivalence $\approx$. Nous définissons, ensuite, une relation de préférence entre les classes d'équivalence E_i's, notée par $\prec_s$, comme suit :

Définition 4
Soient E_i et E_j deux classes d'équivalence de Σ par rapport à $\approx$. Alors, $E_i \prec_s E_j$ ssi $\exists \varphi \in E_i, \exists \varphi' \in E_j$ tel que $\varphi \prec \varphi'$.

La relation $\prec_s$ peut être définie d'une manière équivalente par : $E_i \prec_s E_j$ ssi $\forall \varphi \in E_i, \forall \varphi' \in E_j : \varphi \prec \varphi'$. De plus, on peut facilement vérifier que la relation de préférence $\prec_s$ sur l'ensemble des classes d'équivalence de E_i est un ordre partiel strict. Enfin, $E_i \sim_s E_j$ si et seulement si ni $E_i \prec_s E_j$ ni $E_j \prec_s E_i$ ne sont vérifiés.

Exemple 2
Soit la base de croyances pré-ordonnées $(\Sigma, \preceq)$ de l'Exemple 1. Σ est partitionné comme suit : $\Sigma = E_1 \cup E_2 \cup E_3 \cup E_4$ avec $E_1 = \{\neg r\}$, $E_2 = \{r, \neg h\}$, $E_3 = \{d, h\}$ et $E_4 = \{d \rightarrow r \wedge h\}$. Selon la Définition 4, nous avons : $E_1 \sim_s E_2$, $E_1 \prec_s E_3$, $E_1 \prec_s E_4$, $E_2 \prec_s E_3$, $E_2 \prec_s E_4$ et $E_3 \prec_s E_4$. L'ordre partiel strict $\prec_s$ est illustré dans la Figure 2.

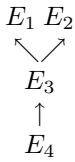


FIG. 2 – L'ordre partiel stict $\prec_s$ sur Σ

Nous proposons maintenant une relation de préférence lexicographique entre les sous-bases cohérentes de $(\Sigma, \preceq)$, notée par $\preceq_{\triangle}$, comme suit :

Définition 5
Soient $(\Sigma, \preceq)$ une base de croyances partiellement pré-ordonnées et A et B deux sous-bases cohérentes de Σ. Alors, A est dite lexicographiquement préférée à B, notée $A \preceq_{\triangle} B$, si et seulement si $\forall i,\ 1 \leq i \leq n$: si $|E_i \cap B| > |E_i \cap A|$ alors $\exists j,\ 1 \leq j \leq n$ tel que $|E_j \cap A| > |E_j \cap B|$ et $E_j \prec_s E_i$.

La proposition suivante décrit quelques propriétés de la relation de préférence $\preceq_{\triangle}$:

Proposition 1
Soit $(\Sigma, \preceq)$ une base de croyances partiellement pré-ordonnées. Alors, (1) $\preceq_{\triangle}$ est un pré-ordre partiel sur l'ensemble des sous-bases cohérentes de Σ. (2) $\preceq_{\triangle}$ vérifie la propriété de monotonie, c'est à dire : $\forall A, B \subseteq \Sigma$, si $B \subseteq A$ alors $A \preceq_{\triangle} B$.

L'ordre partiel strict associé à $\preceq_{\triangle}$ est noté $\prec_{\triangle}$ et est défini par : $A \prec_{\triangle} B$, si et seulement si $A \preceq_{\triangle} B$ et $B \not\preceq_{\triangle} A$. L'égalité correspondante, notée $\approx_{\triangle}$, est définie par $A \approx_{\triangle} B$ ssi $A \preceq_{\triangle} B$ et $B \preceq_{\triangle} A$.

La proposition suivante fournit une définition équivalente à l'équivalence $\approx_{\triangle}$:

Proposition 2
Soit$(\Sigma, \preceq)$ une base de croyances partiellement pré-ordonnées et soient A et B deux sous-bases cohérentes de Σ. Alors, $A \approx_{\triangle} B$ ssi $\forall i, 1 \leq i \leq n : |E_i \cap B| = |E_i \cap A|$.

La préférence lexicographique entre les sous-bases cohérentes d'une base de croyances partiellement pré-ordonnées (Définition 5) recouvre la préférence lexicographique classique entre les sous-bases cohérentes d'une base de croyances totalement pré-ordonnées (Définition 1) comme le montre la proposition suivante :

Proposition 3
Soit une base de croyances totalement pré-ordonnées $(\Sigma, \leq) = S_1 \cup \ldots \cup S_m$, soient A et B deux sous-bases cohérentes de Σ. Alors, (1) $A <_{lex} B$ ssi $A \prec_{\triangle} B$, (2) $A =_{lex} B$ ssi $A \approx_{\triangle} B$.

Exemple 3
Nous illustrons la préférence lexicographique à partir des exemples 1 et 2. Le pré-ordre partiel sur $MCONS(\Sigma)$, obtenu à partir de la Definition 5 est $\preceq_{\triangle} : \{A_7 \prec_{\triangle} A_4,\ A_4 \prec_{\triangle} A_1,\ A_3 \prec_{\triangle} A_1,\ A_3 \sim_{\triangle} A_4,\ A_1 \sim_{\triangle} A_5,\ A_1 \sim_{\triangle} A_2,\ A_7 \sim_{\triangle} A_6,\ A_6 \prec_{\triangle} A_3,\ A_3 \prec_{\triangle} A_5,\ A_5 \prec_{\triangle} A_2\}$. Par exemple, $A_3 \prec_{\triangle} A_1$ car nous avons $i = 3$ tel que $|E_3 \cap A_1| > |E_3 \cap A_3|$ et $\exists j, j = 1$ tel que $|E_1 \cap A_3| > |E_1 \cap A_1|$ et $E_1 <_s E_3$. $A_4 \prec_{\triangle} A_1$ car nous avons $i = 3$ tel que $|E_3 \cap A_1| > |E_3 \cap A_4|$ et $\exists j, j = 2$ tel que $|E_2 \cap A_4| > |E_2 \cap A_1|$ et $E_2 <_s E_3$. $A_3 \sim_{\triangle} A_4$ puisque $A_3 \not\prec_{\triangle} A_4$ car nous avons $i = 2$ tel que $|E_2 \cap A_4| > |E_2 \cap A_3|$ mais $\nexists j$ tel que $|E_j \cap A_3| > |E_j \cap A_4|$ et $E_j <_s E_2$. $A_4 \not\prec_{\triangle} A_3$ car nous avons $i = 1$ tel que $|E_1 \cap A_3| > |E_1 \cap A_4|$ mais $\nexists j$ tel que $|E_j \cap A_4| > |E_j \cap A_3|$ et $E_j <_s E_1$.

5 Une deuxième définition de l'inférence lexicographique

Soit $(\Sigma, \preceq)$ une base de croyances partiellement pré-ordonnées. L'ensemble de toutes les sous-bases cohérentes de $(\Sigma, \preceq)$ qui sont lexicographiquement préférées est noté $\mathfrak{L}ex(\Sigma, \preceq)$, et $\mathfrak{L}ex(\Sigma, \preceq) = Min(CONS(\Sigma), \prec_{\triangle})$. On peut facilement vérifier que chaque élément de $\mathfrak{L}ex(\Sigma, \preceq)$ est une sous-base maximale (par rapport à l'inclusion ensembliste) cohérente de Σ puisque $\preceq_{\triangle}$ satisfait la propriété de monotonie (Proposition 1). Nous définissons maintenant une inférence lexicographique basée sur le pré-ordre $\preceq_{\triangle}$ entre les sous-bases cohérentes :

Définition 6
Soit $(\Sigma, \preceq)$ une base de croyances partiellement pré-ordonnées et soit ψ une formule. ψ est conséquence $\mathcal{P}$-lexicographique de $(\Sigma, \preceq)$, notée $(\Sigma, \preceq) \vdash^{\mathcal{P}}_{lex} \psi$, ssi ψ est conséquence classique de chaque sous-base cohérente préférée de $(\Sigma, \preceq)$ par rapport à $\preceq_{\triangle}$, c'est à dire : $(\Sigma, \preceq) \vdash^{\mathcal{P}}_{lex} \psi$ ssi $\forall B \in \mathfrak{L}ex(\Sigma, \preceq) : B \vdash \psi$.

Le lemme suivant, nécessaire pour prouver un de nos résultats les plus importants stipule que si A est lexicographiquement préférée à B alors A est lexicographiquement préférée à B pour toute base totalement pré-ordonnée compatible.

Lemma 1
Soit $(\Sigma, \preceq)$ une base de croyances partiellement pré-ordonnées et soient A et B deux sous-bases cohérentes de Σ. Soit $(\Sigma, \leq)$ une base de croyances totalement pré-ordonnées compatible avec $(\Sigma, \preceq)$. Alors, (1) Si $A \prec_{\triangle} B$ alors $A <_{lex} B$. (2) Si $A \approx_{\triangle} B$ alors $A =_{lex} B$.

Cette inférence lexicographique est équivalente à l'inférence lexicographique basée sur les compatibles donnée par la Définition 3 comme le montre la proposition suivante :

Proposition 4
Soit $(\Sigma, \preceq)$ une base de croyances partiellement pré-ordonnées et soit ψ une formule propositionnelle. Alors $(\Sigma, \preceq) \vdash_{lex}^{\mathcal{P}} \psi$ ssi $(\Sigma, \preceq) \vdash_{lex}^{\mathcal{C}} \psi$.

Exemple 4
Nous continuons l'exemple 3. Nous avons $\mathfrak{Lex}(\Sigma, \preceq) = \{A_6, A_7\}$.De plus, on peut voir que $\mathfrak{Lex}(\Sigma, \preceq) = \bigcup_{(\Sigma, \leq^i) \in \mathcal{C}(\Sigma, \preceq)} Lex(\Sigma, \leq^i)$ où $\bigcup_{(\Sigma, \leq^i) \in \mathcal{C}(\Sigma, \preceq)} Lex(\Sigma, \leq^i)$ est calculé dans l'exemple 1.

6 Contrepartie sémantique et propriétés

Nous commençons par donner une caractérisation sémantique de l'inférence lexicographique proposée. Une relation de préférence entre deux interprétations est donnée comme suit :

Définition 7
Soient ω et ω' deux interprétations. ω est dite lexicographiquement préférée à ω', notée $\omega \preceq_{\mathcal{W}}^{\Delta} \omega'$, ssi $[\omega] \preceq_{\Delta} [\omega']$ où $[\omega]$ est l'ensemble de formules des Σ satisfaites par ω.

Soit $Min(\mathcal{W}, \prec_{\mathcal{W}}^{\Delta})$ l'ensemble des interprétations préférées de $\mathcal{W}$ par rapport à $\prec_{\mathcal{W}}^{\Delta}$ l'inférence sémantique est définie comme suit :

Définition 8
Soit $(\Sigma, \preceq)$ une base de croyances partiellement pré-ordonnées et soit ψ une formule propositionnelle. Alors ψ est une conséquence $\mathcal{S}$-lexicographique de $(\Sigma, \preceq)$, noté par $(\Sigma, \preceq) \models_{lex}^{\mathcal{S}} \psi$, si et seulement si ψ est satisfaite par chaque interprétation préférée, c'est à dire : $(\Sigma, \preceq) \models_{lex}^{\mathcal{S}} \psi$ ssi $\forall \omega \in Min(\mathcal{W}, \prec_{\mathcal{W}}^{\Delta}), \omega \models \psi$.

Nous montrons que :

Proposition 5
Soient $(\Sigma, \preceq)$ une base de croyances partiellement pré-ordonnées et ψ une formule propositionnelle. Alors, $(\Sigma, \preceq) \vdash_{lex}^{\mathcal{S}} \psi$ ssi $(\Sigma, \preceq) \vdash_{lex}^{\mathcal{P}} \psi$.

Nous illustrons l'approche sémantique avec l'exemple suivant.

Exemple 5
Nous reprenons l'exemple 1. Soit $\mathcal{W}$ l'ensemble des interpretations et $[\omega]$ l'ensemble des formules de Σ satisfaites par ω :

$\mathcal{W}$	d	h	r	$[\omega]$	A_i
ω_0	$\neg d$	$\neg h$	$\neg r$	$\{d \rightarrow r \wedge h, \neg h, \neg r\}$	A_3
ω_1	$\neg d$	$\neg h$	r	$\{d \rightarrow r \wedge h, r, \neg h\}$	A_4
ω_2	$\neg d$	h	$\neg r$	$\{d \rightarrow r \wedge h, h, \neg r\}$	A_2
ω_3	$\neg d$	h	r	$\{d \rightarrow r \wedge h, h, r\}$	
ω_4	d	$\neg h$	$\neg r$	$\{d, \neg h, \neg r\}$	A_6
ω_5	d	$\neg h$	r	$\{d, \neg h, r\}$	A_7
ω_6	d	h	$\neg r$	$\{d, \neg r, h\}$	A_5
ω_7	d	h	r	$\{d \rightarrow r \wedge h, d, r, h\}$	A_1

Les préférences entre les A_i sont données dans l'exemple 3 le pré-ordre entre les interpretations est $\preceq^{\Delta}_{\mathcal{W}}$: $\{\omega_4 \prec^{\Delta}_{\mathcal{W}} \omega_0,\ \omega_0 \prec^{\Delta}_{\mathcal{W}} \omega_6,\ \omega_6 \prec^{\Delta}_{\mathcal{W}} \omega_2,\ \omega_0 \prec^{\Delta}_{\mathcal{W}} \omega_7,\ \omega_4 \sim^{\Delta}_{\mathcal{W}} \omega_5,\ \omega_0 \sim^{\Delta}_{\mathcal{W}} \omega_1,\ \omega_6 \sim^{\Delta}_{\mathcal{W}} \omega_7,\ \omega_2 \sim^{\Delta}_{\mathcal{W}} \omega_3,\ \omega_5 \prec^{\Delta}_{\mathcal{W}} \omega_1,\ \omega_1 \prec^{\Delta}_{\mathcal{W}} \omega_7,\ \omega_7 \prec^{\Delta}_{\mathcal{W}} \omega_3,\}$. Nous pouvons facilement vérifier que $\bigcup_{B \in \mathfrak{L}ex(\Sigma,\preceq)} Mod(B) = Min(\mathcal{W}, \prec^{\Delta}_{\mathcal{W}})$ En effet, $\mathfrak{L}ex(\Sigma,\preceq) = \{A_6, A_7\}$ et puisque $A_6 = \{d, \neg h, \neg r\}$ et $A_7 = \{d, r, \neg h\}$, $Mod(A_6) = \{\omega_4\}$ et $Mod(A_7) = \{\omega_5\}$. Donc,

$$\bigcup_{B \in \mathfrak{L}ex(\Sigma,\preceq)} Mod(B) = \{\omega_4, \omega_5\} = Min(\mathcal{W}, \prec^{\Delta}_{\mathcal{W}})$$

Par conséquent, $(\Sigma, \preceq) \vdash^{\mathcal{P}}_{lex} \psi$ iff $(\Sigma, \preceq) \models^{\mathcal{S}}_{lex} \psi$.

Nous analysons maintenant les propriétés non monotones de cette relation d'inférence. Dans un premier temps, nous l'étendons de telle sorte qu'elle soit définie entre deux formules par rapport à une base de croyances partiellement pré-ordonnées :

Définition 9
Soit $(\Sigma, \preceq)$ une base de croyances partiellement pré-ordonnées et soient ϕ et ψ deux formules. Soit $(\Sigma', \preceq')$ une nouvelle base de croyances partiellement pré-ordonnées telle que : (1) $\Sigma' = \Sigma \cup \{\phi\}$ (2) $\forall \alpha, \beta \in \Sigma, \alpha \preceq \beta$ ssi $\alpha \preceq' \beta$ (3) $\forall \alpha \in \Sigma, \phi \preceq' \alpha$, c'est à dire ϕ est la formule la plus prioritaire dans Σ'. Alors ψ est une Lex-conséquence de ϕ par rapport à Σ, notée $\phi \mid\sim_{lex} \psi$, ssi $(\Sigma', \preceq') \vdash^{\mathcal{P}}_{lex} \psi$.

Ainsi, une d'inférence préférentielle (voir Shoham (1988)) de notre inférence lexicograhique peut être définie comme suit :

Proposition 6
Soient ϕ et ψ deux formules. Alors, $\phi \mid\sim_{lex} \psi$ ssi $\forall \omega \in Min(Mod(\phi), \prec^{\Delta}_{\mathcal{W}}), \omega \models \psi$.

Par conséquent, l'inférence lexicographique proposée satisfait le Système P. Néanmoins, et contrairement à l'inférence lexicographique classique, elle ne satisfait pas la monotonie rationnelle.

Proposition 7
L'inférence $\mid\sim_{lex}$ satisfait les postulats rationnels du Système P contrairement à la monotonie rationnelle.

Enfin, comparée à l'extension possibiliste de Benferhat *et al.* (2004b) et à l'extension de l'inférence basée sur l'inclusion de Junker & Brewka (1989), notre inférence s'avère plus productive. Ce résultat peut être aisément vérifié et ce, en utilisant par exemple la définition basée sur les compatibles et en considérant le résultat démontré par Benferhat *et al.* (1993). En fait, ce résultat stipule qu'étant donnée une base de croyances totalement pré-ordonnées, toute conséquence possibiliste est une conséquence basée sur l'inclusion, et que toute conséquence basée sur l'inclusion est une conséquence lexicographique.

7 Conclusion

Dans cet article, nous avons combiné les avantages de l'inférence lexicographique et ceux des bases de croyances partiellement pré-ordonnées pour raisonner en présence

d'incohérence. Nous avons présenté une inférence lexicographique à partir de bases de croyances partiellement pré-ordonnées qui étend l'inférence lexicographique classique.

Plus précisément, nous avons proposé deux relations d'inférence dont nous avons prouvé l'équivalence. La première, assez naturelle, consiste à appliquer l'inférence lexicographique classique sur toutes les bases totalement pré-ordonnées compatibles. Quant à la seconde, elle s'articule autour de la définition d'un nouveau pré-ordre partiel basé sur la cardinalité entre sous-bases cohérentes. Il s'agit ensuite d'appliquer l'inférence classique sur les sous-bases cohérentes préférées selon ce pré-ordre.

Ce travail ouvre de nombreuses perspectives, entre autres son application à la correlation d'alerte ainsi qu'au traitement d'informations archéologiques.

Remerciements

Ce travail a été réalisé avec le soutien du projet PLACID (ANR SETIN 2006). Ce travail a été réalisé avec le soutien de la Communauté Européenne au sein du projet VENUS (IST-034924) du 6ième programme cadre pour la recherche et le développement (FP6) (Société, Information et Technologie) (IST). Les auteurs sont seuls responsables du contenu de ce papier qui ne reflète pas l'opinion de Communauté Européenne. De plus, la Communauté Européenne n'est pas responsable de l'utilisation des données apparaissant dans l'article.

Références

BENFERHAT S., BONNEFON J.-F. & DA SILVA NEVES R. (2004a). An experimental analysis of possibilistic default reasoning. In *KR'04*, p. 130–140.

BENFERHAT S., DUBOIS D., CAYROL C., LANG J. & PRADE H. (1993). Inconsistency management and prioritized syntax-based entailment. In *IJCAI'03*, p. 640–647.

BENFERHAT S., LAGRUE S. & PAPINI O. (2003). A possibilistic handling of partially ordered information. In *UAI'03*, p. 29–36.

BENFERHAT S., LAGRUE S. & PAPINI O. (2004b). Reasoning with partially ordered information in a possibilistic framework. *Fuzzy Sets and Systems*, **144**, 25–41.

BREWKA G. (1989). Preferred sutheories : an extende logical framework for default reasoning. In *IJCAI'89*, p. 1043–1048.

DA COSTA N. C. A. (1974). Theory of inconsistent formal systems. *Notre Dame Journal of Formal Logic.*, **15**, 497–510.

DUBOIS D., LANG J. & PRADE H. (1994). Possibilistic logic. *Handbook of Logic in Articial Intelligence and Logic Programming.*, **3**, 439–513.

JUNKER U. & BREWKA G. (1989). Handling partially ordered defaults in TMS. In *IJCAI'89*, p. 1043–1048.

LEHMANN D. J. (1995). Another perspective on default reasoning. *Annals of Mathematics and Artificial Intelligence*, **15**(1), 61–82.

NEBEL B. (1994). Base revision operations and schemes : semantics, representation and complexity. In *ECAI'94*, p. 341–345.

PINKAS G. & LOUI R. P. (1992). Reasoning from inconsistency : A taxonomy of principles for resolving conflict. In *KR'92*, p. 709–719.

RESHER N. & MANOR R. (1970). On inference from inconsistent premises. *Theory and Decision.*, **1**, 179–219.

SHOHAM Y. (1988). *Reasoning about change : Time and Causation from the standpoint of Artificial Intelligence*. MIT press.

Fouille de méta-données pour la découverte de mappings entre taxonomies : une approche combinant logique et probabilités

Rémi Tournaire[1,2], Marie-Christine Rousset[1]

[1] Laboratoire d'Informatique de Grenoble UMR5217, équipe HADAS
Bâtiment IMAG D - 681, rue de la Passerelle - 38400 Saint Martin d'Hères

[2] LIRIS UMR5205, équipe Base de données,
Bât. Blaise Pascal - INSA de Lyon - 69621 Villeurbanne Cedex

Résumé : Dans le cadre de la découverte automatique de mappings entre ontologies dans un réseau P2P, nous fournissons un formalisme et une méthode d'estimation pour la probabilité d'un mapping, grâce à une démarche bayésienne exploitant les méta-données associées aux ressources des classes les annotant. Nous présentons un algorithme pour fournir de façon efficace, à partir d'un ensemble de mappings candidats, l'ensemble des mappings dont la probabilité dépasse un certain seuil grâce à un ordre sur les mappings fondé sur leur sémantique logique.
Mots-clés : P2P, mappings, ontologies, probabilité, méta-données

1 Contexte et position du problème

Cet article se situe dans le contexte de réseaux pair à pair de partage sémantique de ressources (documents, données, services). Dans les réseaux pair à pair que nous considérons dont SomeWhere (Adjiman *et al.* (2005)) est un exemple, chaque pair annote sémantiquement les ressources qu'il stocke localement par les noms de classes d'une taxonomie qui lui est propre. Des mappings sont des correspondances entre classes de taxonomies de plusieurs pairs qui permettent des reformulations de requêtes dans les vocabulaires adéquats afin de trouver dans l'ensemble du réseau les ressources satisfaisant une requête exprimée en fonction du vocabulaire d'un pair particulier. De nombreux travaux ont porté sur la découverte automatique de mappings entre deux ontologies (e.g. Shvaiko & Euzenat (2005); Doan *et al.* (2002)). La plupart des méthodes d'alignement sont semi-automatiques : elles renvoient comme résultat un ensemble de mappings probables qu'il faut ensuite valider manuellement.

Dans cet article nous établissons un cadre formel pour le calcul de la probabilité d'un mapping à partir des observations sur les méta-données associées aux ressources des pairs concernés. Nous prouvons sa monotonie par rapport à la relation d'implication logique entre mappings. Puis nous présentons la méthode qui en découle pour explorer

de façon efficace un ensemble de mappings candidats dont on veut sélectionner ceux dont la probabilité dépasse un certain seuil.

1.1 Préliminaires

Une **ressource** est un document identifié par une URI[1] qui est unique.
Nous supposons qu'il est possible d'extraire des **méta-données** de chacun des documents. C'est le cas de fichiers de musique en MP3 pour lesquels la norme ID3 propose un langage attribut-valeur de méta-données qui contient de l'ordre de 50 noms d'attributs (e.g., "title", "genre", "artist", "year", ...) dont les valeurs associées sont soit énumérées (comme les valeurs de l'attribut "genre"), soit du texte libre (comme les valeurs des attributs "title" ou artiste"), soit numériques comme les valeurs de l'attribut "year".
L'exemple suivant illustre un extrait de méta-données que l'on peut trouver dans deux fichiers de musique MP3 identifiés respectivement par id_1 et id_2.

	title	artist	genre	year
$md(id_1)$	"It's raining again"	"Supertramp"	"Rock"	"1982"
$md(id_2)$	"Le lundi au soleil"	"Claude François"	"Pop"	"1972"

Les méta-données que nous considérons dans notre cadre formel sont exprimées dans un langage propositionnel relativement à m attributs booléens $A_1, \ldots, A_m$: chaque ressource id_i disponible dans le réseau est associée à un vecteur booléen de méta-données $mdb(id_i)$= $[b_{i_1}, \ldots, b_{i_m}]$ de taille m où $b_{i_j} = mdb(id_i)[j] = 1$ ssi l'attribut A_j est présent dans la description booléenne des méta-données associées au document d'identifiant id_i.
Ces méta-données peuvent être obtenues par pré-traitement et encodage des méta-données réelles que l'on extrait des fichiers. Ainsi, les méta-données exprimées dans la norme ID3 dans l'exemple précédent peuvent être codées de façon booléenne de la façon suivante :

- Attributs booléens : $A_1 = Rock\_genre$, $A_2 = Pop\_genre$,
  $A_3 = Supertramp\_artist$, $A_4 = ClaudeFrancois\_artist$,
  $A_5 = RainingAgain\_title$, $A_6 = LundiSoleil\_title$, $A_7 = 1982\_year$,
  $A_8 = 1972\_year$,
- Méta-données booléennes :
  - $mdb(id_1) = [1, 0, 1, 0, 1, 0, 1, 0]$
  - $mdb(id_2) = [0, 1, 0, 1, 0, 1, 0, 1]$

Le prétraitement et l'encodage propositionnel sont des problèmes importants que nous n'aborderons pas dans cet article.
Chaque **pair** P_i est une entité (logicielle ou matérielle) autonome qui stocke localement des ressources et les catégorise à l'aide d'un ensemble de classes définies et structurées dans son ontologie.

[1] Unified Resource Identifier

L'**ontologie** O_i d'un pair P_i est un ensemble d'axiomes d'inclusions entre classes ou complémentaires de classes : $E_i^k \sqsubseteq E_i^l$ (où E_i^k et E_i^l sont les noms de classes du vocabulaire de P_i ou des complémentaires de noms de classes de ce même pair).
Par exemple l'axiome $Classique_1 \sqsubseteq Musique_1$ exprime que dans l'ontologie O_1 du pair P_1, la classe appelée $Classique$ est une sous-classe de la classe de nom $Musique$, alors que l'axiome $Classique_1 \sqsubseteq \overline{Rock_1}$ exprime que dans O_1, les classes $Classique$ et $Rock$ sont disjointes.
On fait l'hypothèse que les vocabulaires (i.e. les noms de classe) de pairs différents sont distincts. On convient de la notation C_i pour dénoter la classe appelée C par le pair P_i.
Un **mapping** est une *expression d'inclusion* entre classes ou complémentaires de classes de pairs différents. Par exemple, $Mouv_2 \sqsubseteq Rock_1$ exprime que la classe appelée $Mouv$ par le pair P_2 est une sous-classe de la classe appelée $Rock$ par le pair P_1.
Chaque ressource est stockée dans un (ou plusieurs) pair(s) qui déclare(nt) des **axiomes d'appartenance à une classe** de la forme $C_i(id)$ pour exprimer que la ressource d'identifiant id est classée dans C_i. Par exemple, si id_1 et id_2 sont les identifiants des fichiers $SupertrampItsraining.mp3$ et $lelundiausoleil.mp3$ respectivement, les déclarations $Rock_1(id_1)$, $Pop\_Rock_3(id_1)$, $Nostalgie_2(id_2)$ expriment que le premier fichier est catégorisé par le pair P_1 comme du $Rock$ et par le pair P_3 comme du Pop_Rock, alors que le second fichier est catalogué par le pair P_2 dans la classe $Nostalgie$.
Les axiomes d'inclusion, les axiomes d'appartenance ainsi que les mappings ont une **sémantique logique standard** à base d'interprétation ensemblistes des classes. Une interprétation $\mathcal{I}$ est un couple $(\Delta^{\mathcal{I}}, .^{\mathcal{I}})$, où $\Delta^{\mathcal{I}}$ est le domaine d'interprétation et $.^{\mathcal{I}}$ est une fonction qui interprète chaque nom de classe comme un sous-ensemble de $\Delta^{\mathcal{I}}$, et chaque URI comme un élément de $\Delta^{\mathcal{I}}$. L'hypothèse d'URI unique pour une ressource se traduit formellement par : $a \neq b \Rightarrow a^{\mathcal{I}} \neq b^{\mathcal{I}}$.
L'interprétation du complémentaire d'une classe est l'ensemble complémentaire de son interprétation dans $\Delta^{\mathcal{I}}$.
Etant donnée $\mathcal{O}$ une ontologie (ou une union d'ontologies et de mappings) et $\mathcal{F}$ un ensemble d'axiomes d'appartenance, $\mathcal{I}$ est un modèle de $\mathcal{O} \cup \mathcal{F}$ si :
– pour tout axiome d'inclusion $E \sqsubseteq F$ de $\mathcal{O}$: $E^{\mathcal{I}} \subseteq F^{\mathcal{I}}$
– pour tout axiome d'appartenance $C(a)$ de $\mathcal{F}$: $a^{\mathcal{I}} \in C^{\mathcal{I}}$
Une expression d'inclusion $G \sqsubseteq H$ est vraie dans $\mathcal{O}$ ssi dans tout modèle $\mathcal{I}$ de $\mathcal{O}$, $G^{\mathcal{I}} \subseteq H^{\mathcal{I}}$. On note alors : $\mathcal{O} \models G \sqsubseteq H$.
Une expression d'appartenance $D(b)$ est vraie dans $\mathcal{O} \cup \mathcal{F}$ ssi dans tout modèle $\mathcal{I}$ de $\mathcal{O} \cup \mathcal{F}$, $b^{\mathcal{I}} \in D^{\mathcal{I}}$. On note alors : $\mathcal{O} \cup \mathcal{F} \models D(b)$.
Soit $\mathcal{P}$ un pair ou un ensemble de pairs, $\mathcal{O}(\mathcal{P})$ l'ontologie (ou l'union d'ontologies et de mappings) associée, et $\mathcal{F}(\mathcal{P})$ l'ensemble des axiomes d'appartenance déclarés dans $\mathcal{P}$. Soit $\mathcal{D}(\mathcal{P})$ l'ensemble des identifiants de ressources stockées dans $\mathcal{P}$.
L'**extension** d'une classe C dans $\mathcal{P}$ est l'ensemble des identifiants dont on peut déduire l'appartenance à cette classe :

$$ext(C, \mathcal{P}) = \{d \in \mathcal{D}(\mathcal{P}) \ / \ \mathcal{O}(\mathcal{P}) \cup \mathcal{F}(\mathcal{P}) \models C(d)\}$$

L'extension du complémentaire de $\mathcal{C}$ dans $\mathcal{P}$ est l'ensemble des identifiants de $\mathcal{P}$ dont on ne peut pas déduire l'appartenance à $\mathcal{C}$: $ext(\overline{C}, \mathcal{P}) = \mathcal{D}(\mathcal{P}) \setminus ext(C, \mathcal{P})$.

Les identifiants appartenant à l'extension d'une classe dans $\mathcal{P}$ sont appelées ses **instances**.

1.2 Position du problème

Dans notre cadre, les relations d'inclusion entre classes d'une ontologie ainsi que les relations d'appartenance d'instances à une classe sont posées comme des axiomes ou déduites logiquement, et sont donc vraies ou fausses. Pour les mappings, nous voulons pouvoir modéliser que certains, bien que ne se déduisant pas logiquement d'autres mappings posés comme axiomes, ont une forte vraisemblance en fonction des méta-données associées aux instances stockées dans les pairs mis en correspondance par ces mappings. Par exemple, l'observation que les caractéristiques communes des méta-données extraites des fichiers MP3 classés par P_2 dans $Nostalgie_2$ se retrouvent pour la plupart dans les méta-données extraites des fichiers mp3 classés par P_1 dans Pop_1 (par exemple, 95% et 99% des instances de ces deux classes ont le genre Pop) est un indice fort en faveur de l'ajout du mapping $Nostalgie_2 \sqsubseteq Pop_1$ comme un nouvel axiome.
Nous explicitons dans la section 2 la sémantique probabiliste que nous définissons pour les mappings et son lien avec leur sémantique logique posée dans la section 1.1. Puis, nous montrons comment calculer la probabilité d'un mapping par une estimation bayésienne en considérons les méta-données associées aux classes des pairs impliqués dans le mapping comme des observations.
Le problème central considéré dans cet article est de déterminer parmi un ensemble de mappings candidats ceux dont la probabilité dépasse un certain seuil, en faisant le moins de calcul de probabilités possibles. Nous exhibons dans la section 3 un algorithme d'énumération et de test de mappings candidats qui exploite la propriété de monotonie de la fonction de probabilité de mappings par rapport à l'implication logique.

2 Modélisation et calcul de la probabilité d'un mapping

Considérons un mapping $E_1 \sqsubseteq F_2$ entre deux pairs P_1 et P_2, où E_1 est une classe (ou le complémentaire d'une classe) de l'ontologie O_1 de P_1, et F_2 est une classe (ou le complémentaire d'une classe) de l'ontologie O_2 de P_2.
A partir des méta-données booléennes associées aux instances des extensions de E_1 et $\overline{E_1}$ dans P_1 et de F_2 et $\overline{F_2}$ dans P_2 (cf. Section 1.1), nous calculons l'ensemble des observations noté $Ob(E_1, \overline{E_1}, F_2, \overline{F_2})$ et défini de la façon suivante :

Définition 2.1

$$Ob(E_1, \overline{E_1}, F_2, \overline{F_2}) = (Card(E_1, \overline{E_1}, F_2, \overline{F_2}), MD(E_1, \overline{E_1}, F_2, \overline{F_2}))$$

où $Card(E_1, \overline{E_1}, F_2, \overline{F_2})$ est le quadruplet des nombres d'instances de E_1 et $\overline{E_1}$ dans P_1, et de F_2 et $\overline{F_2}$ dans P_2 : $(|ext(E_1, P_1)|, |ext(\overline{E_1}, P_1)|, |ext(F_2, P_2)|, |ext(\overline{F_2}, P_2)|)$, et $MD(E_1, \overline{E_1}, F_2, \overline{F_2})$ est une matrice à 4 colonnes et m lignes où m est le nombre d'attributs booléens décrivant les méta-données (cf. Section 1.1). La ligne k notée

$MD[k]$ est le quadruplet $(e_1^k, \overline{e_1}^k, f_2^k, \overline{f_2}^k)$ des nombres d'instances de E_1, $\overline{E_1}$, F_2 et $\overline{F_2}$ (dans P_1 et P_2 respectivement) ayant l'attribut numéro k valant 1 dans leur vecteur de méta-données booléennes.

2.1 Modélisation de la probabilité d'un mapping

Nous définissons la probabilité d'un mapping $E_1 \sqsubseteq F_2$ comme la probabilité d'appartenir à $\overline{E_1} \cup F_2$ sachant les observations $Ob(E_1, \overline{E_1}, F_2, \overline{F_2})$

Définition 2.2
Soit X_{E_1} (respectivement X_{F_2}) la variable aléatoire binaire définie sur l'ensemble union des ressources de P_1 et P_2 par $X_{E_1}(r)$ = 1 ssi $r \in ext(E_1, P_1)$ (respectivement $X_{F_2}(r)$ = 1 ssi $r \in ext(B_2, P_2)$). La probabilité du mapping $E_1 \sqsubseteq F_2$ est notée $P(E_1 \sqsubseteq F_2)$ et est définie par :

$$P(E_1 \sqsubseteq F_2) = P(X_{E_1} = 0 \; ou \; X_{F_2} = 1 | Ob(E_1, \overline{E_1}, F_2, \overline{F_2}))$$

On peut définir de la même façon la probabilité d'une inclusion entre classes d'une même ontologie.
Le théorème suivant établit le lien entre la sémantique logique et la sémantique probabiliste des mappings. Il montre que la fonction de probabilité entre mappings est monotone par rapport à l'implication logique entre mappings. Il repose sur l'hypothèse de cohérence des méta-données observées par rapport aux ontologies de chaque pair.

Définition 2.3
Soit O_i l'ontologie d'un pair P_i, les méta-données observées sont cohérentes par rapport à O_i si pour tout axiome d'inclusion $E_i \sqsubseteq E_i'$ de O_i :
$P(E_i \sqsubseteq E_i' | Ob(E_i, \overline{E_i}, E_i', \overline{E_i'})) = 1$

Théorème 2.1
Soient 2 mappings $E_1 \sqsubseteq F_2$ et $E_1' \sqsubseteq F_2'$ entre 2 ontologies O_1 et O_2 de 2 pairs P_1 et P_2. A condition que les méta-données observées dans chaque pair P_1 et P_2 soient cohérentes par rapport aux ontologies respectives O_1 et O_2, on a :

$$Si \; O_1 \cup O_2, E_1 \sqsubseteq F_2 \models E_1' \sqsubseteq F_2' \; alors \; P(E_1 \sqsubseteq F_2) \leq P(E_1' \sqsubseteq F_2')$$

2.2 Estimation bayésienne de la probabilité d'un mapping

Nous présentons d'abord un théorème donnant une formule exprimant la probabilité d'un mapping par rapport aux observations, puis nous donnons la façon dont on peut estimer chacun de ses membres par une méthode statistique bayésienne. Nous abrégeons désormais $Ob(E_1, \overline{E_1}, F_2, \overline{F_2})$ par $Ob_{1,2}$, l'égalité $X_{E_1} = 1$ par E_1, et $X_{F_2} = 0$ par $\overline{F_2}$.
Soit X_i pour $1 \leq i \leq m$ la variable aléatoire binaire définie sur l'ensemble union des ressources de P_1 et P_2, par $X_i(r) = 1$ ssi l'attribut booléen i vaut 1 dans les méta-données booléennes de r. On fait les hypothèses suivantes :

– **(h1)** X_{E_1} et X_{F_2} sont indépendantes conditionnellement à $O_{1,2}$ et aux X_i : i.e. dès qu'on connaît $Ob_{1,2}$ et les X_i, le fait de savoir qu'une instance appartient à E_1 ou non n'apporte pas plus d'information sur la probabilité qu'elle appartienne à F_2
– **(h2)** les X_i sont indépendantes conditionnellement aux classes (hypothèse classique en apprentissage naïve bayes) et indépendantes

Théorème 2.2
*Sous les hypothèses **(h1)** et **(h2)** :* $P(E_1 \sqsubseteq F_2) =$

$$1 - P(E_1|Ob_{1,2})P(\overline{F_2}|Ob_{1,2}) \prod_{i=1}^{m} \sum_{v_i=0}^{1} \frac{P(X_i = v_i|E_1, Ob_{1,2})P(X_i = v_i|(\overline{F_2}, Ob_{1,2})}{P(X_i = v_i|Ob_{1,2})} \quad \textbf{(1)}$$

Malgré la complexité apparente de cette formule, son calcul ne nécessite qu'un nombre d'opération en $O(m)$ à partir de ses opérandes. Le théorème 2.3 fournit un moyen d'estimer la valeur des paramètres de ces opérations. Ci-dessous figurent ces $2 + 3m$ paramètres à estimer :
– $P(E_1|Ob_{1,2})$ et $P(\overline{F_2}|Ob_{1,2})$: probabilités des classes E_1 et $\overline{F_2}$ sachant les observations
– $P(X_i = 1|Ob_{1,2})$ pour $1 \leq i \leq m$: probabilité qu'une instance ait l'attribut i à 1 dans ses méta-données booléennes sachant les observations
– $P(X_i = 1|E_1, Ob_{1,2}), P(X_i|\overline{F_2}, Ob_{1,2})$ pour $1 \leq i \leq m$: probabilité qu'une instance ait la valeur 1 pour l'attribut i dans ses méta-données booléennes *sachant qu'elle appartient à E_1, ou à F_2, connaissant les observations sur P_1 et P_2.*

Nous montrons maintenant comment estimer ces paramètres nécessaires au calcul de $P(E_1 \sqsubseteq F_2)$, par une approche bayésienne de la statistique (Schervish, 2002). Afin d'alléger les formules dans cette partie, on notera :
– $n_{E_1} = |ext(E_1, P_1)|$ et $n_{F_2} = |ext(F_2, P_2)|$
– n_{P1} nombre d'instances du pair P_1, n_{P2} celui du pair P_2.
– $n = n_{P1} + n_{P2}$ le nombre d'instances sur les pairs P_1 et P_2
– $n_i = a_i + \overline{a_i} + b_i + \overline{b_i}$ le nombre d'instances u de P_1 et P_2 avec $mdb(u)[i] = 1$
On s'intéresse en premier lieu à l'estimation de $P(E_1|Ob_{1,2})$. Le résultat de l'expérience consistant à prendre au hasard une instance et à regarder si elle appartient à E_1 suit une loi de Bernouilli de paramètre $p = P(A_1|Ob_{1,2})$. C'est la loi probabiliste la plus simple, correspondant à 2 états, de probabilités respectives p et $1 - p$. L'approche bayésienne consiste à modéliser le paramètre recherché p comme une variable aléatoire.

Théorème 2.3
Si $X_{E_1}|Ob_{1,2}$ suit une loi de Bernouilli de paramètre $p = P(X_{E_1}|Ob_{1,2})$, on peut estimer p de façon bayésienne par la formule suivante (Schervish, 2002; Gia-Hien Nguyen, 2008) :

$$P(E_1|Ob_{1,2}) = p \approx \frac{1 + n_{E_1}}{2 + n_{E_1} + (n_{P1} - n_{E_1})} = \frac{1 + n_{E_1}}{2 + n_{P1}}$$

En transposant ce théorème aux autres paramètres, on obtient les formules d'estimation suivantes :
– $P(\overline{F_2}|Ob_{1,2}) \approx \frac{1+n_{P2}-n_{F_2}}{2+n_{P2}}$

– $P(X_i = 1|Ob_{1,2}) \approx \frac{1+n_i}{2+n}$, et $P(X_i = 0|Ob_{1,2}) \approx= 1 - \frac{1+n_i}{2+n}$
– $P(X_i = 1|E_1, Ob_{1,2}) \approx \frac{1+n_i+a_i}{2+n+n_{E_1}}$, et idem que ci-dessus pour $X_i = 0$
– $P(X_i = 1|\overline{F_2}, Ob) \approx \frac{1+n_i+\overline{b_i}}{2+n+n_{P2}-n_{F_2}}$ (idem pour $X_i = 0$)

Le théorème pour l'estimation de $P(E_1 \sqsubseteq F_2)$ est un corollaire des théorèmes 2.2 et 2.3 :

Théorème 2.4
Sous les hypothèses du théorème 2.2, en utilisant l'estimation bayésienne du théorème 2.3, on estime $P(E_1 \sqsubseteq F_2)$ sachant les observations $Ob_{1,2}$ issues des méta-données par :

$$P(E_1 \sqsubseteq F_2) \approx 1 - \frac{1+n_{E_1}}{2+n_{P1}} \frac{1+n_{P2}-n_{F_2}}{2+n_{P2}} \prod_{i=1}^{m} \left(\frac{t_E^i t_F^i}{t_X^i} + \frac{(1-t_E^i)(1-t_F^i)}{1-t_X^i} \right)$$

avec $t_E^i = \frac{1+n_i+a_i}{2+n+n_{E_1}}$, $t_F^i = \frac{1+n_i+\overline{b_i}}{2+n+n_{P2}-n_{F_2}}$, $t_X^i = \frac{1+n_i}{2+n}$

Cette estimation sera notée $P_{estim}(E_1 \sqsubseteq F_2, Ob_{1,2})$.

Malgré l'apparence complexe de cette formule, chacun de ses membres se calculent facilement à partir des n_{P1}, n_{P2}, n_{E_1} et n_{F_2}, a_i et b_i, $\overline{a_i}$ et $\overline{b_i}$ contenus dans $Ob_{1,2}$, eux-mêmes obtenus en comptant les instances de P_1 et P_2 respectant les critères adéquats. On remarque que si E_1 n'a pas d'instance (dans P_1), le calcul se réduit à l'estimation suivante : $P(E_1 \sqsubseteq F_2) \approx \frac{1+n_{E_1}}{2+n_{P1}} \frac{1+n_{P2}-n_{F_2}}{2+n_{P2}}$ qui est celle de $1-P(E_1|Ob_{1,2})P(\overline{F_2}|Ob_{1,2})$, le produit faisant 1, indépendants des statistiques sur les valeurs des attributs. L'absence d'instances dans E_1 se traduit donc de façon cohérente par la non-prise en compte de la corrélation entre valeurs des attributs dans les classes concernées par le mapping.

3 Enumération et test d'un ensemble de mappings candidats

Le problème abordé dans cette section consiste à énumérer et tester (i.e. savoir si $P(m) > s$, en réalité si $P_{estim}(m, Ob) > s$) de façon efficace un grand ensemble de mappings candidats $\mathcal{M}$ entre deux ontologies O_1 et O_2. Pour éviter de tester successivement tous les mappings de $\mathcal{M}$ par la méthode décrite à la section 2, on structure $\mathcal{M}$ par une relation d'ordre fondée sur la sémantique, puis on exploite la propriété de monotonie de la probabilité par rapport à cet ordre, qui découle du théorème 2.1.

3.1 Ordre sur $\mathcal{M}$ et monotonie de la probabilité des mappings

Nous introduisons tout d'abord la relation d'équivalence logique $\equiv_{O_1,O_2}$ suivante :
m et m' sont équivalents ssi $O_1, O_2 \models m \Leftrightarrow m'$.
La relation $\preceq$ est la relation d'implication logique entre mappings compte tenu des ontologies :

$$m \preceq m' \quad \Leftrightarrow \quad O_1, O_2, m \models m'$$

$m \preceq m'$ est une relation d'ordre sur l'ensemble $\mathcal{M}' = \mathcal{M}/\equiv_{O_1,O_2}$, à savoir l'ensemble des mappings candidats quotienté par l'équivalence.
Le théorème 2.1 implique directement la propriété de monotonie suivante :

Théorème 3.1
Si les méta-données observées dans chaque pair P_1 et P_2 sont cohérentes par rapport aux ontologies respectives O_1 et O_2 alors : $m \preceq m' \Rightarrow P(m) \leq P(m')$.

Ce théorème entraîne deux conséquences (équivalentes) sur lesquelles est basé l'algorithme 1 : Sous l'hypothèse de la cohérence des méta-données par rapport aux ontologies O_1 et O_2, étant donnés deux mappings m et m' entre O_1 et O_2, avec $m \preceq m'$, et un seuil quelconque $0 \leq s \leq 1 : P(m) > s \Rightarrow P(m') > s \;\; et \;\; P(m') < s \Rightarrow P(m) < s$

Influence de la modélisation de la probabilité d'un mapping sur la monotonie
On peut penser à modéliser $P(E_1 \sqsubseteq F_2)$ par une probabilité conditionnelle :
$P(E_1 \sqsubseteq F_2) = P(X_{F_2} = 1 | X_{E_1} = 1, Ob)$
Dans ce cas, il n'y a pas de monotonie. De plus, cette modélisation manque de cohérence avec la logique car l'équivalence de la contraposition n'est pas préservée au niveau probabiliste : $P(E_1 \sqsubseteq F_2) \neq P(\overline{F_2} \sqsubseteq \overline{E_1})$

3.2 Algorithme d'énumération et de test

L'entrée de l'algorithme 1 est constituée :
– des observations Obs : $Obs(E_1 \sqsubseteq F_2) = Ob(E_1, \overline{E_1}, F_2, \overline{F_2})$ pour chaque mapping $E_1 \sqsubseteq F_2$
– G le graphe de la réduction transitive de la relation $\preceq$ sur $\mathcal{M}'$. On trouve un algorithme à cet effet dans (Schwarz, 2007).
– un seuil s
La sortie de cet algorithme est l'ensemble des minimaux de l'ensemble des mappings de $\mathcal{M}$ dont la probabilité (estimée) dépasse s.
On suppose qu'on a déjà vérifié la cohérence des méta-données par rapport aux ontologies, à partir de l'estimation des probabilités des liens de subsomption avec la méthode de la section 2. L'algorithme utilise les primitives suivantes :
– INF(m, G) et SUP(m, G) : retournent respectivement les ensembles de mappings inférieurs et supérieurs de m par rapport à $\preceq$ dans $\mathcal{M}'$
– MAX(G) : maximaux des mappings de $\mathcal{M}'$
– PRED(m, G) et SUCC(m, G) : prédécesseurs et successeurs de m dans les mappings candidats, par rapport à $\preceq$. m' est prédécesseur de m ssi $m' \preceq m$ et s'il n'existe pas $m'' \in \mathcal{M}'$ tel que $m' \preceq m'' \preceq m$. Successeur est la relation inverse de prédécesseur.
Le principe général de l'algorithme est le parcours d'un ensemble de mappings courants $Current \subseteq \mathcal{M}'$, et la construction de la liste suivante $Next$ en fonction des résultats du parcours de $Current$. S'ils dépassent le seuil, les mappings sont stockés dans M_{Val}, sinon dans M_{NVal}.
$Current$ est initialisé avec les maximaux de $\mathcal{M}'$. Chaque mapping de $Current$ non encore testé est soumis au test $P_{estim}(m, Obs(m)) > s$ (ligne 7) :

– Si $P_{estim}(m, Ob) > s$, alors on l'ajoute ainsi que ses mappings supérieurs de $\mathcal{M}'$ dans M_{Val} (lignes 11 et 12). On ajoute aussi tous les prédécesseurs de m dans $Next$ (ligne 13), car on ne sait rien sur eux (sauf si on en a déjà vu avant, auquel cas on épure $Next$ ligne 18).
– Sinon, on ajoute m et tous les mappings inférieurs à m dans M_{NVal} (l. 15).

Algorithm 1 Enumération et test d'un ensemble de mappings candidats

Require: Obs, G, s
Ensure: Retourne les minimaux de l'ensemble des mappings de $\mathcal{M}/\equiv_{O_1,O_2}$ dont la probabilité dépasse s

1: $M_{Val} \leftarrow \emptyset$, $M_{NVal} \leftarrow \emptyset$
2: $M_{NotMin} \leftarrow \emptyset$
3: $Current \leftarrow \text{MAX}(G)$
4: **while** $Current \neq \emptyset$ **do**
5: $Next \leftarrow \emptyset$
6: **for** each $m \in Current$ **do**
7: **if** $m \notin M_{Val}$ **and then** $P_{estim}(m, Obs(m)) > s$ **then**
8: $E \leftarrow \text{SUP}(m, G)$
9: $M_{Val} \leftarrow M_{Val} \cup E$
10: $M_{NotMin} \leftarrow M_{NotMin} \cup E \setminus \{m\}$
11: **end if**
12: **if** $m \in M_{Val}$ **then**
13: $Next \leftarrow Next \cup \text{PRED}(m, G)$
14: **else**
15: $M_{NVal} \leftarrow M_{NVal} \cup \text{INF}(m, G)$
16: **end if**
17: **end for**
18: $Current \leftarrow Next \setminus M_{NVal}$
19: **end while**
20: Return($M_{Val} \setminus M_{NotMin}$)

La monotonie est doublement exploitée : elle sert d'une part à construire le niveau suivant, et d'autre part à propager vers les supérieurs ou inférieurs d'un mapping le fait que sa probabilité dépasse le seuil s ou non. Nous avons montré la correction de cet algorithme qu'exprime le théorème suivant :

Théorème 3.2
L'algorithme 1 termine, et soit $Result = M_{Val} \setminus M_{NotMin}$ l'ensemble qu'il retourne $Result$ est l'ensemble des mappings minimaux par rapport à $\preceq$ de l'ensemble des mappings de $\mathcal{M}'$ dont la probabilité dépasse le seuil s.

4 Conclusion

Par rapport aux travaux existants sur l'alignement d'ontologies, nous nous positionnons au sein des méthodes d'alignement d'ontologies à bases d'instances (e.g. Doan

et al. (2002), fondé sur une méthode de classification). Nous ne faisons pas l'hypothèse de connaissance centralisée. L'originalité de notre approche est la prise en compte de méta-données normalisées, transformées (sans l'avoir détaillé ici) en logique propositionnelle, qui permettent d'estimer de façon bayésienne la sémantique probabiliste de mappings. Celle-ci est différente de celle de (Dong *et al.*, 2007) qui définit des sémantiques pour des mappings probabilistes (p-mappings) dans le cadre d'intégration de schémas relationnels à base de correspondances entre attributs, ainsi que la consistance des données par rapport aux p-mappings. Les p-mappings et les probabilités sont supposés fournis par une méthode en amont.
En outre, nous avons fourni une méthode et un algorithme pour structurer un ensemble de mappings candidats afin de fournir en sortie les mappings dont la probabilité d'un mapping dépasse un certain seuil, de manière efficace. Pour celà, nous exploitons la monotonie de la probabilité sur un ordre fondé sur la sémantique logique.
Nous avons comme perspective la mise en application sur des jeux de tests générés et réels, afin de mesurer les performances en terme de complexité et de robustesse, en variant les méthodes d'estimation. Nous étudierons le prétraitement des méta-données. Le langage de mapping pourra être étendu, notamment avec des expressions disjonctives et conjonctives.

Références

ADJIMAN P., CHATALIC P., GOASDOU F., ROUSSET M.-C. & SIMON L. (2005). SomeWhere in the Semantic Web. In *International Workshop on Principles and Practice of Semantic Web Reasoning*.

DOAN A., MADHAVAN J., DOMINGOS P. & HALEVY A. (2002). Learning to map between ontologies on the semantic web. In *WWW '02 : Proceedings of the 11th international conference on World Wide Web*, p. 662–673, New York, NY, USA : ACM.

DONG X. L., HALEVY A. Y. & YU C. (2007). Data integration with uncertainty. In *VLDB*, p. 687–698.

GIA-HIEN NGUYEN, PHILIPPE CHATALIC M.-C. R. (2008). A Probabilistic Trust Model for Semantic Peer to Peer systems.

RUSSELL S., NORVIG P., MICLET L. & POPINEAU F. (2006). *Intelligence Artificielle. 2e édition*. PEARSON EDUCATION.

SCHERVISH D. G. (2002). *Probability and Statistics, p. 336*. Addison Wesley.

SCHWARZ U. M. (2007). Transitive reduction and Union Find.

SHVAIKO P. & EUZENAT J. (2005). A survey of schema-based matching approaches. p. 146–171.

TOM M. MITCHELL M.-H. (1997). *Machine Learning, by Tom M. Mitchell, McGraw-Hill*.