N° d'ordre : 357

Année 2002

THÈSE

présentée en vue de
l'obtention du titre de

DOCTEUR
de
L'ÉCOLE NATIONALE SUPÉRIEURE
DE L'AÉRONAUTIQUE ET DE L'ESPACE

SPÉCIALITÉ : Représentation de la connaissance et formalisation du raisonnement

par

Christophe GARION

Apports de la logique mathématique en ingénierie des exigences

Soutenue le 13 décembre 2002 devant la Commission d'Examen :

Mmes.	S.	CERRITO	Présidente - Rapporteur
	L.	CHOLVY	Directrice de thèse
MM.	L.	FARINAS DEL CERRO	
	J.	LANG	
Mme	O.	PAPINI	Rapporteur
M.	L.	VAN DER TORRE	Rapporteur

Thèse préparée au sein du Département Traitement de l'Information et Modélisation de l'ONERA-centre de Toulouse.

Remerciements

Je ne saurais exprimer ici toute ma gratitude envers Laurence Cholvy, qui a été ma directrice de thèse durant ces trois années. Elle a su m'intéresser à la logique mathématique, partager mon enthousiasme dans l'exploration de nouvelles voies dans le processus d'Ingénierie des Exigences et m'apporter toute son expérience pour me recadrer lorsque je partais « dans le décor ». Encadrer un premier thésard n'est sûrement pas un travail facile, mais elle s'en est acquitté parfaitement. J'espère simplement que ce manuscrit sera à la hauteur du travail qu'elle a su me consacrer.

Je remercie également René Jacquart et Jacques Cazin, directeurs successifs du Département de Traitement de l'Information et Modélisation, et Jack Foisseau, responsable de l'équipe Modélisation - Ingénierie des Besoins, pour m'avoir accueilli durant ma thèse.

Je remercie les membres du jury de l'honneur qu'ils m'ont fait en acceptant de s'intéresser à ce travail. Merci aux rapporteurs, Serena Cerrito, Odile Papini et Leon van der Torre, pour leur relecture plus qu'attentive du manuscrit.

Merci à Serena Cerrito pour son travail de rapporteur. Ses remarques sur les problèmes d'« implantation » soulevés par l'utilisation de la logique CO m'ouvrent un nouveau champ de recherche.

Je remercie Odile Papini pour les questions soulevées sur la partie « fusion » de ce travail.

Merci à Leon van der Torre pour toutes ses remarques (en particulier les coquilles dans les preuves...) qui m'ont permis d'améliorer ce manuscrit tant sur le fond que sur la forme, surtout sur la logique déontique.

Merci à Jérôme Lang d'avoir accepté si chaleureusement de faire partie du jury. Les nombreuses discussions que nous avons eues au cours de ces trois années, scientifiques ou autres, ont toujours été très intéressantes.

Je remercie également Luis Fariñas del Cerro d'avoir accepté de faire partie du jury de cette thèse.

Je tiens à remercier l'ensemble du DTIM à l'ONERA Toulouse pour leur accueil, en particulier Noelle, Christiane et Josette pour leur disponibilité. Merci à Robert Demolombe pour toutes ces discussions fructueuses sur la logique. Merci également à Pilar pour m'avoir supporté pendant trois années dans son bureau.

Merci beaucoup à Sébastien Konieczny pour toutes ces discussions (tardives...) sur la logique et les problèmes de fusion de bases de croyances (j'y tiens...). Son aide et sa disponibilité m'ont été profitables lorsque j'ai commencé à m'intéresser à ce domaine.

Je remercie également Pierre Marquis, Sylvie Coste-Marquis et l'équipe de l'IUT de Lens pour leur patience et leur gentillesse.

Merci à Bernard Lécussan pour tout ce qu'il a fait durant ces six dernières années de ma vie d'étudiant (et plus maintenant...).

Je profite de l'occasion qui m'est faite ici pour remercier tous les amis qui m'ont soutenu durant ces trois années (voire plus). Leur apport, très peu scientifique :), m'a permis d'arriver au bout de cette aventure. Que les personnes dont les noms suivent sachent que je ne pourrai jamais assez les remercier pour tout ce qu'elles ont pu faire (liste sans ordre d'importance, bien sûr !) : Camille (même si elle n'est pas arrivée au bout de la thèse), Caro, Sylvain, Arnaud, Manu (Asskronotes...), Émilie, Amel, Dany, Virginie, Laurent, Momo, Bubu, Juju, David, Skal, Flo (Labarthe Crew !), Fred, Tep, Yvan, Jé, Fab, Rémi, Nawel, Erminator, Bruno, Laurent (alias Luis), Thaddée, Fabienne, Séb, Sophie, Cristobal, Johan, Quentin, Mios, Azélia, Vincent, Jérôme, Mike, Maud et Louise, Matthieu, Mathieu, Tanguy, Olivier, Fifi, Pathos, Nat, Cèpe, Antoine, Vanessa, Djidji, Ch'Mini, Thomas, Sylvie. Que ceux que j'aurais pu oublier par inadvertance m'en excuse. Merci simplement d'être vous...

Enfin et surtout, je remercie mes parents et mon frère, pour tout le soutien qu'ils m'ont apporté durant ces vingt six années.

Table des matières

Chapitre 1

Introduction : l'ingénierie des exigences

La complexité actuelle des projets industriels, informatiques ou non, fait que les relations contractuelles entre la maîtrise d'œuvre et la maîtrise d'ouvrage ne sont pas aisées. En particulier, cette complexité conduit souvent à la constitution des deux côtés d'équipes hétérogènes qui ne travaillent souvent que sur une partie infime du projet. Il faut donc pour les équipes de la maîtrise d'œuvre pouvoir vérifier que ce qu'elles développent correspond bien aux besoins initiaux du client et que les équipes de la maîtrise d'ouvrage puissent également vérifier que les besoins initiaux sont respectés. Les exigences sont là pour résoudre ces problèmes : elles sont une traduction plus technique des besoins de la maîtrise d'ouvrage et vont donc être le fil conducteur du développement du produit considéré.

L'ingénierie des exigences caractérise le processus qui doit aboutir à la production d'un ensemble cohérent de spécifications portant sur un objet à concevoir. Ces spécifications porteront non seulement sur les propriétés statiques ou comportementales de l'objet, mais également sur les restrictions à apporter sur le processus de développement de cet objet. Les étapes suivantes de production de l'objet (conception globale et détaillée, développement, vérification et validation) dépendent donc toutes de cette première phase. L'étape d'ingénierie des exigences est donc cruciale, car c'est elle qui induira les erreurs les plus coûteuses.

Les besoins de méthodes formelles pour le processus d'ingénierie des exigences sont donc réels : elles permettent en effet de contrôler le processus dans son ensemble et ainsi de garantir que les exigences correspondent bien aux besoins de la maîtrise d'ouvrage. Nous avons choisi dans ce mémoire d'utiliser un formalisme rigoureux pour représenter ce processus : la logique mathématique. Nous nous efforcerons de justifier ce choix à chaque fois qu'il le sera nécessaire.

On peut voir le processus d'ingénierie des exigences comme étant constitué de trois phases : une phase d'expression d'exigences, une phase d'obtention d'un ensemble cohérent d'exigences, et enfin une phase de distribution des exigences. On peut remarquer que dans la littérature classique sur le sujet, cette troisième phase est occultée.

Lors de la phase d'expression des exigences, plusieurs participants au processus provenant la plupart du temps de la maîtrise d'ouvrage, émettent en premier lieu des besoins concernant l'artefact à construire, le plus souvent en langage naturel. Il faut donc dans cette phase traduire ces besoins initiaux dans un formalisme commun à tous les partici-

pants. On peut en effet avoir besoin de confronter les exigences de plusieurs participants et l'utilisation d'un formalisme commun permet d'effectuer facilement cette comparaison. Nous nous intéresserons également à la modélisation de réglementations (ensemble de règles qui énoncent ce qui est interdit, obligatoire, permis) et de contraintes du domaine (les contraintes du monde physique par exemple), car le développement d'un artefact est toujours contraint par ces deux notions. Par exemple, on peut émettre des exigences qui sont compatibles avec les contraintes du domaine (et donc proposer un artefact physiquement « possible »), mais qui contredisent la réglementation existante. Là encore, on peut remarquer que très peu de travaux s'intéressent à ces deux aspects pourtant essentiels du processus d'ingénierie des exigences. Nous avons choisi d'utiliser une logique de préférences pour représenter ces différentes notions.

La seconde phase du processus a pour but d'obtenir un seul ensemble d'exigences cohérent à partir des ensembles d'exigences émis par les différents participants. Il se peut en effet que plusieurs participants émettent des exigences contradictoires (peut-être sans le savoir). Or, il paraît évident que l'on doit obtenir un seul ensemble cohérent d'exigences, car elles vont représenter des propriétés que doit satisfaire l'objet à concevoir. Il faut donc disposer d'un formalisme permettant de résoudre les conflits possibles entre les différentes exigences. La logique mathématique étant par nature faite pour raisonner sur des problèmes d'incohérence, nous nous attacherons donc à présenter des méthodes formelles basées sur la logique mathématique pour pouvoir résoudre ces conflits éventuels.

Enfin, la troisième phase du processus d'ingénierie des exigences que nous proposons est une phase de distribution des exigences. Cette phase est souvent occultée dans la littérature « classique » sur le processus, mais elle nous paraît pourtant primordiale : il s'agit en effet de vérifier si, étant donné un ensemble d'agents exécutants qui ont pour charge de réaliser l'artefact en question, on peut construire un objet dont les propriétés respectent les exigences émises par les participants au processus. Nous serons donc amenés à nous intéresser à la modélisation d'agents rationnels et à la caractérisation d'un processus de distribution régi par une entité centrale. Là encore, les propriétés de la logique mathématique, en particulier des logiques modales pour la représentation d'agents, nous paraissent parfaitement adaptées à ce problème.

Ce mémoire est structuré comme suit :

Dans une première partie, nous nous intéressons à la modélisation d'exigences, de réglementation et de contraintes du domaine. Le chapitre 2 présente un état de l'art sur la représentation d'exigences, le chapitre 3 dresse un panorama détaillé sur le raisonnement normatif et le chapitre 4 présente un aperçu sur la modélisation en logique de préférences. Enfin, le chapitre 5 présente notre contribution à la modélisation d'exigences, de réglementations et de contraintes du domaine avec un formalisme commun.

La seconde partie du rapport concerne la gestion des incohérences possibles entre différents ensembles d'exigences. Après un bref aperçu sur les techniques existant en ingénierie des exigences dans le chapitre 6, nous nous intéressons plus particulièrement dans le chapitre 7 aux techniques de fusion de bases de croyances. Nous nous appuyons sur ces travaux pour présenter dans le chapitre 8 une logique de fusion majoritaire de bases de croyances que nous avons développée et nous montrons dans le chapitre 9 l'application possible de cette logique à la résolution de conflits entre ensembles d'exigences.

La troisième partie du mémoire concerne la phase de distribution des exigences. Nous présentons rapidement dans le chapitre 10 quelques formalismes logiques développés pour représenter et raisonner avec des agents rationnels. Les chapitres 11 et 12 présentent res-

pectivement la modélisation que nous proposons des agents exécutants et le processus de distribution des exigences à proprement parler.

Enfin, le dernier chapitre présente un résumé de nos travaux et les perspectives pour les travaux futurs.

Toutes les preuves des propriétés et théorèmes présentés dans ce mémoire sont données en annexe A.

Première partie

Modélisation d'exigences et de réglementations

Introduction

Il existe déja beaucoup de formalismes qui permettent de modéliser des exigences de façon semi-formelle ou formelle. La plupart de ces formalismes s'appuient souvent sur une logique du premier ordre, ou sur une logique modale pour pouvoir représenter des notions temporelles. Mais aucun d'entre eux ne permet de raisonner avec des exigences ordonnées suivant un ordre de priorité propre à chaque participant. Nous pensons que permettre à chacun des participants de raffiner la notion d'exigence en introduisant un ordre de préférence va offrir de nouvelles alternatives en cas d'inconsistance entre les exigences des différents participants.

De plus, la plupart des études faites en ingénierie des exigences ne prend pas en compte la réglementation et les contraintes du domaine (les contraintes physiques par exemple). Ce sont pourtant deux points clé du développement d'un produit, puisqu'il existe toujours une réglementation en vigueur et des contraintes du domaine quel que soit le type du produit que l'on veut concevoir. Le respect de la réglementation permet de construire des objets qui sont conformes aux normes et les contraintes du domaine expriment les propriétés que doit satisfaire un objet « possible » (dans le sens où on pourra physiquement le construire). Il faut donc pouvoir exprimer des phrases normatives et des contraintes, afin de vérifier que les exigences produites par chacun des participants ou finales respectent bien la réglementation et les contraintes du domaine.

Nous proposons pour modéliser ces notions d'utiliser une logique de préférences. Cette logique va nous permettre de représenter les exigences sous forme ordonnée (un participant va préférer telle exigence à telle autre), mais également de modéliser des phrases normatives complexes (règles avec exception, Contrary-to-Duties) et des contraintes du domaine.

Dans cette partie, nous présentons tout d'abord un état de l'art sur la représentation d'exigences, puis un état de l'art sur la logique déontique. Dans une troisième section, nous nous intéressons à la représentation sous forme logique de préférences et nous détaillons plus particulièrement la logique modale CO, développée par Craig Boutilier. Enfin, le chapitre 5 présente notre contribution : nous montrons comment utiliser CO pour représenter des exigences sous forme ordonnée, des phrases normatives et des contraintes du domaine. Nous définissons ensuite une notion de cohérence entre un ensemble d'exigences ordonnées et une réglementation et des contraintes du domaine, puis nous montrons que l'on peut extraire de cet ensemble d'exigences ordonnées un ensemble d'exigences « les meilleures possibles » tenant compte de la réglementation et des contraintes du domaine.

Chapitre 2

Représentation d'exigences (état de l'art)

Les exigences sont des spécifications de haut niveau portant sur un objet quelconque à concevoir. On peut les considérer comme étant l'intermédiaire entre les besoins et les spécifications détaillées de l'objet. Les exigences sont exprimées par des participants qui peuvent être des clients, des utilisateurs finaux ou des développeurs par exemple. Elles vont donc être exprimées au départ sous des formes diverses : langage naturel, langage technique (comme par exemple UML en développement orienté objet) etc. Si le fait de garder une certaine hétérogénéité dans les langages est une source de facilité pour les participants, en revanche cela ne facilite pas les opérations de vérification ou de validation des exigences, comme par exemple la vérification de la cohérence entre deux exigences.

De plus, les exigences servent de contrat et de moyen de communication entre les clients et les exécutants. Il faut donc qu'elles soient compréhensibles par les deux parties, mais également non ambigües (car elles peuvent avoir valeur contractuelle). Il convient donc de disposer d'un langage commun pour représenter les exigences.

Le langage naturel étant bien évidemment trop ambigü, nous devons utiliser un cadre formel pour exprimer des exigences. Si l'utilisation de la logique mathématique est assez répandue pour la représentation de spécifications de bas niveau (langage B par exemple), il n'en est pas de même pour la représentation d'exigences. Pourtant, la logique mathématique est un cadre formel précis qui se prête bien à l'expression d'exigences : la sémantique associée à une logique permet à chacun des participants d'avoir les mêmes notions de valeur de vérité d'une formule, les connecteurs logiques permettent de combiner les exigences, une logique du premier ordre permet d'exprimer les liens généraux qui peuvent exister entre les exigences etc. De plus, comme nous le verrons dans la partie II, la gestion de l'incohérence est un problème crucial pour l'ingénierie des exigences. L'incohérence étant une notion logique, il paraît pertinent d'utiliser la logique pour représenter les exigences.

Nous allons donc présenter quelques formalismes qui s'appuient sur la logique mathématique pour pouvoir représenter et raisonner avec des exigences. De nombreuses études s'intéressent aux aspects temporels des exigences. La conception d'un artefact implique souvent l'émission de contraintes temporelles. Les exigences concernant des applications temps-réel critiques doivent évidemment contenir des indications temporelles précises. Dans ce cas, de nombreux formalismes existent et sont communément utilisés : automates, logiques temporelles type CTL*, réseaux de Petri (voir par exemple [15] pour une présentation rapide et exhaustive de ces techniques).

Le cas particulier des exigences sur des applications temps-réel critiques est plus proche du domaine de la spécification (dans le sens où on demande des propriétés très précises sur le système), mais on peut aussi considérer des exigences temporelles de plus haut niveau. Par exemple, une exigence concernant un site web de vente en ligne pourrait spécifier que les réponses à un utilisateur extérieur se fassent en moins de trente secondes. Ces exigences temporelles de haut niveau sont importantes car elles représentent souvent un aspect « qualité » du produit à concevoir.

Néanmoins, nous ne nous intéresserons pas dans cette étude aux aspects temporels des exigences. L'ajout de la dimension temporelle à un formalisme logique n'est pas triviale, en particulier si les contraintes temporelles sont quantifiées comme dans l'exemple précédent (il faut pouvoir en effet raisonner soit sur des moments ordonnés linéairement, soit directement avec les constantes de temps comme « 30s » et donc disposer d'un calcul approprié). On pourra juste remarquer que la plupart des études faites dans ce domaine concernent plutôt l'expression de spécifications à des fins de validation ou de vérification via des procédés comme le model-checking.

Par contre, nous détaillerons dans ce chapitre quatre formalismes permettant de représenter des exigences : les points de vue, une logique quasi-classique, l'utilisation de contextes en logique modale et enfin la notion de position.

2.1 La notion de point de vue

2.1.1 Définition

Généralement, lors de la phase d'expression des exigences, on utilise un modèle de données commun pour des facilités de vérification de cohérence par exemple. Malheureusement, la nature même des exigences les rend relatives : *statiquement* tout d'abord, puisque les exigences sont émises par des participants différents, mais également *dynamiquement*, puisqu'elles sont amenées à évoluer au cours du processus d'ingénierie des exigences.

La notion utilisée depuis ces dix dernières années pour représenter cette relativité des exigences est la notion de *point de vue* [52, 56, 55]. Un point de vue est un couple composé d'un *acteur* et un certain nombre de fragments de spécifications dont il est responsable, les *perspectives*. Le point de vue permet donc non seulement de formaliser la relativité de chaque exigence, puisqu'il est associé à un acteur, mais également de pouvoir exprimer des relations inter-points de vue, qui vont modéliser les relations entre les acteurs, et ainsi détecter les incohérences possibles. Une personne physique peut être l'acteur de plusieurs points de vue. Par exemple une personne peut exprimer des exigences sur une partie d'un projet, et en exprimer d'autres sur une autre partie du projet. Elle sera donc acteur pour deux points de vue différents.

Un point de vue est composé de cinq champs :

- un *style de représentation* qui définit la notation utilisée par le point de vue (réseau de Pétri, logique temporelle, langage naturel etc) ;
- un *domaine* ;
- une *spécification*, i.e. des phrases qui décrivent des domaines particuliers dans le style de représentation du point de vue ;
- un *plan de travail* décrivant le processus de construction de la spécification ;
- un *enregistrement* contenant un historique du développement de la spécification.

Le point de vue permet donc de donner une représentation relative des exigences, tout

d'abord statiquement grâce au style de représentation, au domaine et à la spécification elle même, mais également dynamiquement grâce au plan de travail et au champ d'enregistrement.

La plupart des moyens de représentation des points de vue sont issus du génie logiciel et s'appuient sur des moyens de représentation semi-formels. A. Finkelstein et al. [56, 55] utilisent la notion de point de vue comme élément central d'un processus de développement distribué de logiciels. Easterbrook [52] et Nuseibeh et al. [105, 106] ont continué ces travaux, surtout pour utiliser les points de vue comme outil d'analyse de spécifications incohérentes, comme nous le verrons dans la partie II.

2.1.2 Relations entre points de vue

Comme nous l'avons vu précédemment, les points de vue sont composés de perspectives, qui sont des fragments de spécifications. Or, plusieurs points de vue peuvent concerner la même spécification et avoir des perspectives qui se contredisent, se recouvrent ou se complétent. Il convient donc que chaque fragment de spécification puisse s'assembler correctement avec les autres pour produire un ensemble cohérent de spécifications. Il est donc nécessaire de pouvoir exprimer des relations entre les différents points de vue.

Les relations qui peuvent exister entre deux points de vue sont les suivantes [106] :

- les deux points de vue peuvent être totalement *indépendants*. Par exemple un point de vue peut décrire la décomposition fonctionnelle d'un système et un autre les ressources financières du projet ;
- il peut exister une *relation existentielle* entre les deux points de vue qui implique que l'existence d'un point de vue est dépendante de l'existence du deuxième ;
- les deux points de vue peuvent se *recouvrir partiellement*. Deux points de vue peuvent concerner deux domaines a priori différents, mais qui en fait ont des choses en commun.
- enfin, les deux points de vue peuvent se *recouvrir complètement*. C'est le cas de deux points de vue qui portent sur le même domaine, mais proposent des solutions différentes.

Il ne faut pas oublier non plus que les relations inter-points de vue doivent permettre de transformer l'information d'un point de vue à l'autre si les styles de représentation sont différents. Une relation inter-points de vue va permettre par exemple de traduire une spécification dynamique exprimée grâce à un réseau de Pétri vers une spécification sous forme de diagrammes d'état-transition en UML. Ceci peut permettre de vérifier la cohérence entre les points de vue.

La plupart des approches utilisent la logique du premier ordre ou des formalismes proches pour pouvoir exprimer des relations inter-points de vue. Nous reviendrons sur ces relations dans le cadre de la gestion de l'incohérence (cf. partie II).

Comme nous venons de le voir, la représentation d'exigences est problématique pour plusieurs raisons : hétérogénéité des moyens de représentation, incohérences possibles, exigences non complètes etc. La logique mathématique semble donc être un formalisme intéressant pour représenter des exigences : elle offre un cadre formel identique à tous les participants, elle permet de raisonner sur certaines incohérences et de déduire des informations à partir d'un ensemble réduit d'exigences. Nous présentons dans les sections suivantes des formalismes utilisant la logique mathématique pour représenter les exigences et les points de vue.

2.2 Une modélisation grâce à une logique quasi classique

Dans [72] et [73], Hunter et Nuseibeh présentent un formalisme simple pour pouvoir raisonner avec des exigences et gérer l'incohérence entre différents points de vue. Ces travaux poursuivent l'approche proposée dans [55]. Nous ne détaillerons ici que la représentation des exigences, la gestion de l'incohérence étant détaillée dans la partie II.

Les exigences seront représentées ici grâce à des *points de vue*, notion présentée dans la section précédente. Pour valider et satisfaire une spécification, on a besoin de pouvoir effectuer un raisonnement logique et une analyse logique de cette spécification. Une représentation formelle simple des exigences pourrait utiliser la logique classique, car cette dernière permet de représenter facilement de nombreuses informations et possède des outils et des techniques pour raisonner sur ces informations.

Or, les exigences sont souvent incohérentes : soit un point de vue est intrinséquement incohérent (l'auteur du point de vue ne s'en rend pas forcément compte), i.e. les spécifications contenues dans le point de vue sont incohérentes entre elles, ou incohérent avec un autre point de vue. Si l'on utilise la logique classique pour représenter des exigences, on sera confronté au problème de l'*ex-falso quolibet*. En effet, en logique classique, on peut inférer n'importe quoi à partir d'un ensemble incohérent de formules, i.e. :

$$\{\alpha, \neg\alpha\} \vdash \beta$$

La logique classique ne semble donc pas être adaptée pour représenter des exigences, puisqu'en cas d'incohérence on sera capable de déduire tout et n'importe quoi à partir des ensembles d'exigences. Pour résoudre ce problème, Hunter et Nuseibeh ont utilisé une logique quasi-classique qui autorise un raisonnement non trivial en présence d'un ensemble incohérent. Nous allons la présenter dans ce qui suit.

2.2.1 La logique QC

La logique QC est une logique non classique développée par Besnard et Hunter dans [11], et qui a la propriété de ne pas inférer n'importe quelle formule à partir d'un ensemble de formules incohérent. Une telle logique est dite *para-consistante*. L'affaiblissement des règles de la logique classique dans les logiques para-consistantes rend le comportement des connecteurs «non-classique». Par exemple, le syllogisme disjonctif $((\alpha \vee \beta) \wedge \neg\beta) \rightarrow \alpha$ n'est plus vrai, alors que Modus Ponens est toujours une règle d'inférence. Ainsi, α ne peut pas être déduit de $\{(\alpha \vee \beta), \neg\beta\}$, alors qu'il peut l'être de $\{(\neg\beta \rightarrow \alpha), \neg\beta\}$. La logique QC permet de résoudre ces problèmes.

Le langage $\mathcal{L}$ de QC est défini de façon classique. On ne prendra donc pas la peine de le détailler ici. La sémantique et l'axiomatique de QC sont décrites en détail dans [11]. La sémantique de QC ressemble à celle d'une logique à quatre valeurs de vérité. L'axiomatique de QC permet de conserver le comportement classique des connecteurs logiques, évite l'*ex-falso quolibet* et la relation de conséquence $\vdash_{QC}$ conserve les propriétés classiques de monotonie et de reflexivité. Plutôt que de détailler QC, nous l'illustrons sur un exemple dans le paragraphe suivant.

2.2.2 Exemple de représentation d'exigences

Reprenons l'exemple donné par Hunter et Nuseibeh dans [73]. Il s'agit d'un exemple générique utilisé par d'autres chercheurs. Le problème est de spécifier puis d'éliciter les

exigences concernant le Service Ambulancier de Londres (*London Ambulance Service* noté LAS). Plusieurs participants, parmi lesquel le contrôleur de la salle d'incident, le responsable des opérations et le responsable logistique émettent des exigences par rapport au fonctionnement du LAS. Les exigences émises par chacun des participants sont les suivantes :

Contrôleur de la salle d'incident

- une urgence médicale est la conséquence soit d'une maladie, soit d'un accident ;
- dès que l'on reçoit un coup de fil signalant une urgence médicale, l'ambulance disponible la plus proche doit être dépêchée sur le lieu de l'urgence.

Responsable des opérations

- à la réception d'un coup de fil signalant un incident, si une ambulance n'est pas la plus proche du lieu de l'incident elle ne doit pas être envoyée.

Responsable logistique

- si aucun opérateur (conducteur ou médecin) n'est disponible pour une ambulance, alors cette ambulance ne peut pas être l'ambulance la plus proche ;
- si une ambulance n'est pas la plus proche d'un incident, il ne faut pas lui envoyer un signal d'appel.

Supposons que l'on dispose des prédicats suivants :
- *accident* tel que $accident(p, l)$ signifie que la personne p a eu un accident à l'endroit l ;
- *maladie* tel que $maladie(p, l)$ signifie que la personne p est malade à l'endroit l ;
- *urgence_medicale* tel que $urgence\_medicale(p, l)$ signifie qu'il y a une urgence médicale à l'endroit l concernant la personne p ;
- *appel* tel que $appel(p, l)$ signifie qu'il y a eu un appel concernant la personne p à l'endroit l ;
- *ambulance_proche* tel que $ambulance\_proche(v)$ signifie que v est l'ambulance la plus proche ;
- *envoi_ambulance* tel que $envoi\_ambulance(v, l)$ signifie que l'on envoie l'ambulance v à l'endroit l ;
- *equipe* tel que $equipe(v)$ signifie que l'ambulance v possède une équipe complète (médecin et conducteur) ;
- *appel_ambulance* tel que $appel\_ambulance(v)$ signifie que l'on envoie un signal d'appel à l'ambulance v.

Les exigences émises par chacun des participants vont être représentées respectivement par les ensembles de formules suivants :

$$\begin{aligned} VP_1 &= \{\forall p\ \forall l\ \ accident(p,l) \vee maladie(p,l) \leftrightarrow urgence\_medicale(p,l), \\ &\quad \forall p\ \forall l\ \forall v\ \ appel(p,l) \wedge urgence\_medicale(p,l) \wedge \\ &\quad ambulance\_proche(v) \rightarrow envoi\_ambulance(v,l)\} \\ VP_2 &= \{\forall p\ \forall l\ \forall v\ \ appel(p,l) \wedge \neg ambulance\_proche(v) \rightarrow \neg envoi\_ambulance(v,l)\} \end{aligned}$$

$$VP_3 = \{\forall v \ \neg equipe(v) \rightarrow \neg ambulance\_proche(v) \\ \forall v \ \neg ambulance\_proche(v) \rightarrow \neg appel\_ambulance(v)\}$$

Supposons que l'on souhaite examiner le scénario suivant : John a eu un accident à London Road et on a appelé le LAS pour signaler cet accident. L'ambulance 1 n'a pas d'équipage disponible, mais est l'ambulance la plus proche de l'accident. On modélise ce scénario par l'ensemble F défini ci-après :

$$F = \{accident(John, London\_Road), \ appel(John, London\_Road), \\ \neg equipe(Ambulance1), \ ambulance\_proche(Ambulance1)\}$$

L'ensemble $VP_1 \cup VP_2 \cup VP_3 \cup F$ est incohérent, car on peut en dériver $envoi\_ambulance(Ambulance1, London\_Road)$ et $\neg envoi\_ambulance(Ambulance1, London\_Road)$. Malgré cela, on peut quand même dériver des formules exploitables de $VP_1 \cup VP_2 \cup VP_3 \cup F$, comme par exemple $\neg appel\_ambulance(Ambulance1)$. On peut ainsi commencer à définir la procédure *envoi_ambulance* sans se préoccuper des incohérences révélées par l'analyse du scénario. Nous verrons dans la partie II que l'on peut également trouver et qualifier les sources d'incohérences.

2.2.3 Conclusion

La logique QC a donc été utilisée par Hunter et Nuseibeh pour représenter des exigences et surtout pour pouvoir raisonner avec des exigences provenant de plusieurs points de vue qui peuvent être incohérents localement[1] ou entre eux. Dans le domaine strict de modélisation des exigences, cette logique reste assez pauvre : les exigences sont exprimées sous forme propositionnelle, et on n'a aucune représentation formelle de la relativité des points de vue (seul le langage naturel permet de distinguer les points de vue VP1 et VP2).

2.3 Modélisation d'exigences avec une logique modale contextuelle

Comme nous l'avons vu, les points de vue permettent d'exprimer la relativité statique et dynamique des connaissances portant sur des exigences. Cette relativité est très proche de la notion de *contexte* : chacun des points de vue peut être représenté par un contexte dans lequel les exigences relatives au point de vue sont vraies. Si on considère un contexte englobant tous les contextes « points de vue », le fait qu'une spécification existe dans un contexte n'implique pas forcément que cette spécification existe dans le contexte englobant. L'utilisation des contextes est donc un moyen de représenter les points de vue. Un contexte englobant va permettre de définir les relations inter-points de vue, comme nous le verrons plus tard.

L'utilisation de contextes est un moyen de représenter les problèmes de généralité en Intelligence Artificielle. John Mac Carthy fut un des premiers à défendre l'idée d'une formalisation en logique des contextes [98]. En effet, on peut considérer que toute connaissance n'est jamais vraie dans l'absolu, mais est vraie dans un certain contexte qui représente les conditions d'évaluation des connaissances (conditions de lieu, de temps etc). Pour pouvoir

[1] I.e. les exigences d'un même point de vue sont incohérentes entre elles.

exprimer ces connaissances contextuelles, Mac Carthy introduit un opérateur $ist(c, f)$ où p est une formule et c un contexte. La formule se lit alors : f est vraie dans le contexte c. Historiquement, Guha a été le premier à proposer une formalisation logique des contextes dans [65].

Une première formalisation possible des contextes se base sur une approche syntaxique. Les propositions produites par un contexte sont définies comme étant des objets manipulés par un méta-langage du premier ordre. On peut par exemple citer les travaux de Attardi et Simi [5, 6, 7] qui proposent un formalisme simple se fondant sur la logique des prédicats. Ils étendent la logique du premier ordre en utilisant un prédicat particulier, *in*, qui permet d'exprimer des croyances contextuelles.

Cependant, Montague a montré dans [103] que l'existence de formules auto-référentielles (qui peuvent appartenir à la fois au méta-langage et au langage objet) conduit à un système logique incohérent. Des solutions ont été proposées pour pallier à ce problème : hiérarchisation des méta-langages [5], mais aussi restrictions sur les règles régissant les liens entre méta-langage et langage objet [63]. Toutefois, ces restrictions affaiblissent le pouvoir d'expression du langage.

Reste donc pour pouvoir exprimer cette notion de contexte une approche utilisant les logiques modales (voire multi-modales). Nayak a proposé de formaliser les contextes en utilisant un opérateur modal par contexte. Cette contrainte ne permet donc pas de raisonner sur des relations inter-contextuelles. Buvač et al., dans [21, 19, 20], ont proposé une approche multi-modale de la logique des contextes. Pour eux, les contextes sont des objets du langage, et on peut donc exprimer des relations sur les contextes. Nous allons nous intéresser plus particulièrement aux travaux de Perrussel [111, 109, 110, 112] qui a proposé une logique des contextes multi-modale s'appuyant sur une sémantique de Kripke et étant destinée à raisonner plus spécifiquement sur les exigences.

2.3.1 Raisonnement contextuel propositionnel

Nous présentons dans cette section une logique modale développée par Perrussel qui permet de raisonner sur des contextes. Le langage de la logique s'appuie sur un langage propositionnel $\mathcal{L}_0$. Ce langage est étendu dans un premier temps par une modalité `ist` qui permet de représenter des connaissances contextuelles. $\texttt{ist}(\mathcal{X}, \varphi)$ où $\mathcal{X}$ est un contexte et φ une formule propositionnelle signifie *la formule φ est vraie dans le contexte $\mathcal{X}$*. Le langage ainsi obtenu est étendu aux formules de type $\langle \mathcal{X}_1, \ldots, \mathcal{X}_n \rangle \ : \ \phi$ où $\langle \mathcal{X}_1, \ldots, \mathcal{X}_n \rangle$ est une séquence de contextes non vide et ϕ une formule du langage étendu précédemment.

L'évaluation de la valeur de vérité d'une formule se fera donc toujours dans une séquence de contexte particulière et surtout non vide. On peut ainsi noter qu'on n'a pas de «super-contexte», puisque toutes les expressions seront contextuelles.

L'axiomatique de la logique est définie de la façon suivante : si $\vdash \phi$ est un schéma d'axiome de la logique propositionnelle, alors pour toute séquence non vide de contexte σ, $\vdash \sigma \ : \ \phi$ est un schéma d'axiome. L'opérateur modal `ist` est un opérateur de type (KD) (cf. [25]) et vérifie donc $\vdash \sigma \ : \ \texttt{ist}(\mathcal{X}, \varphi \rightarrow \psi) \rightarrow (\texttt{ist}(\mathcal{X}, \varphi) \rightarrow \texttt{ist}(\mathcal{X}, \psi))$ et $\vdash \sigma \ : \ \texttt{ist}(\mathcal{X}, \varphi) \rightarrow \neg\texttt{ist}(\mathcal{X}, \neg\varphi)$. On peut remarquer que `ist` ne vérifie pas les axiomes d'introspection (notés habituellement 4 et 5). Les trois règles d'inférence sont :

- Modus Ponens ;
- une règle d'entrée dans un contexte :

$$(RIC^{IN}) \quad \frac{\vdash \langle \mathcal{X}_1, \ldots, \mathcal{X}_n \rangle : \ \texttt{ist}(\mathcal{X}, \varphi)}{\vdash \langle \mathcal{X}_1, \ldots, \mathcal{X}_n, \mathcal{X} \rangle : \varphi}$$

– une règle de sortie de contexte :

$$(RIC^{OUT}) \quad \frac{\vdash \langle \mathcal{X}_1, \ldots, \mathcal{X}_n, \mathcal{X} \rangle : \varphi}{\vdash \langle \mathcal{X}_1, \ldots, \mathcal{X}_n \rangle : \ \texttt{ist}(\mathcal{X}, \varphi)}$$

Ces deux dernières règles d'inférence permettent de différencier la logique des contextes de la logique modale de représentation des croyances (qui est de type (KD)). La règle (RIC^{IN}) permet à partir d'une formule de type $\texttt{ist}(\mathcal{X}, \varphi)$ d'obtenir une formule propositionnelle φ en intégrant $\mathcal{X}$ dans le contexte de $\texttt{ist}(\mathcal{X}, \varphi)$. La règle (RIC^{OUT}) permet quant à elle d'obtenir une formule $\texttt{ist}(\mathcal{X}, \varphi)$ à partir d'une formule propositionnelle φ en utilisant le contexte « le moins englobant » de φ.

On peut donc exprimer sans aucun problème la relativité des points de vue dans ce formalisme : $\langle \mathcal{X} \rangle : \neg\varphi$ n'est pas incohérent avec $\langle \mathcal{X} \rangle : \texttt{ist}(\mathcal{X}_1, \varphi)$. On peut également vérifier que des liens entre les différents contextes sont présents.

Supposons par exemple que nous ayons deux contextes $\mathcal{X}_S$ et $\mathcal{X}_D$ qui représentent respectivement la vue statique d'un système et la vue dynamique de ce même système. Une contrainte que l'on pourrait avoir sur ces deux contextes est la suivante : si `classeA` est vraie dans $\mathcal{X}_S$ alors `stateClasseA` est vraie dans $\mathcal{X}_D$. On peut interpréter cela de la façon suivante : si on a une représentation statique de la classe A, alors on doit avoir un diagramme d'états-transitions de cette même classe. La relation qui existe entre les deux contextes est elle-même contextuelle et pourrait être exprimée de la sorte : $\langle \mathcal{R} \rangle : Prob \vee (\texttt{ist}(\mathcal{X}_S, \texttt{classeA}) \rightarrow \texttt{ist}(\mathcal{X}_D, \texttt{stateClasseA}))$. *Prob* est une variable propositionnelle de « marquage » qui va permettre de signaler s'il y a un problème. Si on peut dériver $\langle \mathcal{R} \rangle : Prob$, c'est qu'il y a une représentation statique de A, mais pas de diagrammes d'états-transitions pour A. Si on suppose alors qu'on a $\langle \mathcal{R}, \mathcal{X}_S \rangle : \texttt{classeA}$ et $\langle \mathcal{R}, \mathcal{X}_D \rangle : \neg\texttt{stateClasseA}$, on peut en déduire en appliquant (RIC^{OUT}), (D) et Modus Ponens $\langle \mathcal{R} \rangle : \texttt{ist}(\mathcal{X}_S, \texttt{classeA}) \wedge \neg\texttt{ist}(\mathcal{X}_S, \texttt{stateClasseA})$. Donc par application de Modus Ponens, on peut en déduire $\langle \mathcal{R} \rangle : Prob$. Il y a donc un problème entre les deux points de vue : $\mathcal{X}_D$ est incomplet.

Notons que la sémantique de cette logique modale est basée sur des modèles de Kripke du type $\langle W, S, R, J \rangle$ où W est un ensemble de mondes, S est un ensemble de situations (i.e. des couples consistués d'une séquence de contextes et d'un monde de W), J une fonction de valuation propositionnelle et R une relation d'accessibilité sur $S * S$. La relation de satisfaction va donc dépendre des contextes puisque R est une relation sur $S * S$. On peut se reporter à [111] pour plus de détails sur la sémantique de cette logique.

2.3.2 Extension au premier ordre

La logique présentée précédemment a ses limites : on ne peut utiliser que des formules propositionnelles dans un contexte. Perrussel étend donc cette logique au premier ordre pour deux raisons :

– raisonner sur des formules quelconques et non plus des propositions. On pourra ainsi mieux spécifier les exigences ;
– quantifier les contextes.

On notera qu'il se base toujours sur un ensemble de prédicats communs à tous les contextes (Mac Carthy avait par exemple suggéré que les prédicats aient une arité contextuelle). Nous ne présentons pas cette logique ici, Perrussel étend la logique définie en section 2.3.1 en utilisant les axiomes habituelles de la logique des prédicats avec égalité,

puis étend l'axiomatique de `ist` avec le schéma de Barcan. Deux règles de généralisation sont ajoutées :

– une première qui permet de quantifier «classiquement» :

$$(G)\ \frac{\vdash \sigma\ :\ \varphi}{\vdash \sigma\ :\ (\forall x)\varphi}$$

– et une seconde qui permet de quantifier les séquences de contextes et ainsi de régler le problème des variables libres dans ces séquences :

$$(G')\ \frac{\vdash \sigma\ :\ \varphi}{\vdash \forall x(\sigma\ :\ \varphi)}$$

Grâce à cette extension, on peut maintenant exprimer beaucoup plus finement les relations entre les contextes. Reprenons l'exemple de la section 2.3.1. Supposons maintenant que nous disposions d'un prédicat $\texttt{classe}(-)$ d'arité un qui exprime le fait qu'il existe une description sous la forme d'un diagramme de classe d'un certain concept. Supposons également que nous ayons à disposition un deuxième prédicat, $\texttt{stateClasse}(-)$, qui exprime le fait qu'il existe un diagramme d'états-transitions associé à une classe particulière. On peut donc raffiner la contrainte de l'exemple par $\langle\mathcal{R}\rangle\ :\ \forall\mathcal{X}_0\ \forall x\ Prob \vee (\texttt{ist}(\mathcal{X}, \texttt{classe}(x)) \rightarrow \exists\mathcal{X}_1\ \texttt{ist}(\mathcal{X}_1, \texttt{stateClasse}(x)))$ ($\mathcal{X}_0$ et $\mathcal{X}_1$ sont deux variables contextuelles et x est une variable des objets du discours). Cette contrainte peut s'expliquer de la façon suivante : s'il existe un contexte (donc un point de vue) $\mathcal{X}_0$ dans lequel on a écrit un diagramme de classe pour le concept x, alors il existe un contexte $\mathcal{X}_1$ dans lequel on a écrit un diagramme d'états-transitions pour la classe correspondante. Cela permet de vérifier par exemple qu'une conception orientée objet exprime bien toutes les propriétés dynamiques des concepts qu'elle modélise.

De la même manière que précédemment, on peut vérifier par une preuve que si on a $\langle\mathcal{R}, \mathcal{X}_S\rangle\ :\ \texttt{classe}(a)$ par exemple et que dans toutes les séquences $\langle\mathcal{R}, \mathcal{X}_0\rangle$ on a $\neg\texttt{stateClasse}(a)$, alors on peut en déduire $\langle\mathcal{R}\rangle\ :\ Prob$.

2.3.3 Conclusion

La logique développée par Perrussel dans [111] permet donc d'exprimer assez finement des exigences dans un certain point de vue (représenté ici par des contextes) et surtout des relations inter-points de vue. Le pouvoir d'expression de son formalisme est assez grand, surtout avec l'extension de la logique au premier ordre. On peut ainsi quantifier les contextes et raisonner sur l'existence possible d'un contexte dans lequel une certaine formule est vraie. Le fait de pouvoir plonger dans ou sortir d'un contexte permet également de vérifier que les relations inter-points de vue sont bien satisfaites par les exigences exprimées dans les différents points de vue.

2.4 Modélisation d'exigences ordonnées

Les formalismes décrits jusqu'à présent considèrent que les exigences exprimées par un point de vue ont toutes la même importance. L'acteur d'un point de vue ne peut pas par exemple exprimer le fait qu'une exigence est pour lui prioritaire et qu'une autre est optionnelle. Cholvy et Hunter ont développé dans [44] un formalisme qui permet à chacun des participants au processus d'ingénierie des exigences de « classer » ses exigences selon un ordre de préférence qui lui est propre.

2.4.1 Présentation du formalisme

Cholvy et Hunter représentent les exigences émises par un agent a par un ensemble fini et cohérent de formules propositionnelles noté Δ_a. À partir de cet ensemble Δ_a, chaque agent va construire une *position* qui sera un ensemble ordonné de ses propres exigences. Ils considérent également deux ensembles de formules propositionnelles notés *Dom* et *Reg* qui représentent respectivement les contraintes du domaine et la réglementation en vigueur ($\varphi \in Dom$ signifie « il est obligatoire que φ soit vraie »). Ce sont les premiers travaux qui prennent en compte la réglementation et les contraintes du domaine. Or, quel que soit l'artefact à construire, il existe toujours une réglementation et des lois du domaine le concernant. Il est donc important de pouvoir représenter ces deux notions dans un formalisme qui permet de vérifier si, par exemple, des exigences sont conformes à la réglementation ou aux contraintes du domaine. Cholvy et Hunter supposent évidemment que $Dom \cup Reg$ est cohérent (en particulier on n'oblige pas quelque chose d'impossible). Un ensemble d'exigences est cohérent avec la réglementation et les contraintes du domaine ssi $\Delta_a \cup Dom \cup Reg$ est cohérent.

Si on considère Δ_a comme étant l'ensemble des exigences de l'agent a, on peut écrire Δ_a de la façon suivante : $\Delta_a = \{\alpha_1, \ldots, \alpha_n\}$. Dans cette représentation, chaque α_i représente une conjonction d'exigences de a qui sont d'égale importance pour lui. Un ordre sur ces conjonctions d'exigences est défini par le tuple $\Gamma_a = [\alpha_1, \ldots, \alpha_n]$ appelé position de l'agent a. Les formules α_1, …, α_n sont telles que α_i est préféré par a à α_{i+1} pour tout $i \in \{1, \ldots, n-1\}$. La position de l'agent a construit donc une relation d'ordre linéaire sur Δ_a.

La position d'un agent induit un ordre sur les mondes possibles. Prenons un exemple : supposons que $\Gamma_a = [\alpha, \beta]$. Dans ce cas, l'agent a pour exigence sur l'objet à concevoir $\alpha \wedge \beta$. Mais si ce n'est pas possible (à cause de la réglementation ou des contraintes de domaine), l'agent préfère que l'objet à concevoir satisfasse $\alpha \wedge \neg\beta$. Si ce n'est toujours pas possible, il accepte que l'objet satisfasse $\neg\alpha \wedge \beta$. Enfin, dans le pire des cas, si ce n'est toujours pas possible, il accepte que l'objet satisfasse $\neg\alpha \wedge \neg\beta$.

Au support Γ_a de l'agent correspond l'ordre sur les mondes possibles suivant[2] :

$$\|\alpha \wedge \beta\| < \|\alpha \wedge \neg\beta\| < \|\neg\alpha \wedge \beta\| < \|\neg\alpha \wedge \neg\beta\|$$

Cholvy et Hunter caractérisent ensuite formellement le lien qui existe entre toute position d'un agent a et un ordre sur les mondes possibles. Tout d'abord, une fonction *Compromises* associe la position $\Gamma_a = [\alpha_1, \ldots, \alpha_n]$ avec un ensemble des mondes possibles tel que $Compromises([\alpha_1, \ldots, \alpha_n]) = \{\|\beta_1 \wedge \ldots \wedge \beta_n\| \ : \ \beta_1 \in \{\alpha_1, \neg\alpha_1\}$ et $\beta_n \in \{\alpha_n, \neg\alpha_n\}\}$. *Compromises* permet donc de construire les formules représentant toutes les alternatives possibles quant à la satisfaction ou la non satisfaction des exigences de l'agent. Par exemple, $Compromises([\alpha, \wedge\beta]) = \{\|\alpha \wedge \beta\|, \|\alpha \wedge \neg\beta\|, \|\neg\alpha \wedge \beta\|, \|\neg\alpha \wedge \neg\beta\|\}$. Une relation d'ordre $<_a$ est ensuite définie sur $Compromises([\alpha_1, \ldots, \alpha_n]) - \{\phi\}$ (pour ne pas prendre en compte les éléments de type $\|\alpha \wedge \neg\alpha \wedge \beta\|$ par exemple) en utilisant un ordre lexicographique.

Ces définitions leur permettent de pouvoir vérifier la compatibilité des exigences avec la réglementation et les lois du domaine sémantiquement mais aussi syntaxiquement. Dans la sémantique, $\{D \ : \ D \in Compromises(\Gamma_a)$ et $\exists M \in D$ tel que $M \models Dom \cup Reg\}$ n'est pas vide, donc il existe au moins un élément minimal de cet ensemble pour $<_a$. Cet élément est même unique, car $<_a$ est un ordre strict sur $Compromises(\Gamma_a)$. Il existe

[2] $||\varphi||$ est appelé ensemble de modèles de φ (cf. [25]) et contient les mondes w tels que $w \in val(\varphi)$.

donc une formule γ_a telle que $\|\gamma_a\| = min_{<_a}\{D \ : \ D \in Compromises(\Gamma_a)$ et $\exists M \in D$ tel que $M \models Dom \cup Reg\}$. γ_a est un ensemble d'exigences compatible avec Dom et Reg. C'est même la «meilleure» alternative pour la position de l'agent a qui est cohérente avec la réglementation et les lois du domaine.

Syntaxiquement, on peut construire γ_a en construisant un ensemble E de la façon suivante. Il suffit de vérifier que α_1 soit cohérent avec $Dom \cup Reg$. Si c'est le cas, alors $E = \{\alpha_1\}$ sinon $E = \{\neg\alpha_1\}$. On continue en vérifiant si α_2 est cohérent avec $E \cup Dom \cup Reg$. Si c'est le cas, $E = E \cup \{\alpha_2\}$ sinon $E = E \cup \{\neg\alpha_2\}$. Lorsque l'on a fini avec α_n, $E = \gamma_a$.

2.4.2 Exemple

Prenons un exemple tiré de [44]. On se place dans la phase d'expression des exigences pour la construction d'une maison. Supposons qu'un participant ait la position suivante : $\Gamma = [centre\_ville \vee proche\_metro, murs\_blancs, calme]$. Il souhaite que la maison soit en centre ville ou près d'une station de métro, qu'elle ait des murs blancs et qu'elle soit dans un quartier tranquille. Les exigences sont ordonnées par la position de l'agent. Supposons de plus que $Dom = \{proche\_metro \rightarrow \neg calme\}$ (i.e. quand on est près d'une station de métro, le quartier n'est pas très tranquille) et que $Reg = \{centre\_ville \rightarrow \neg murs\_blancs\}$ (i.e. en centre ville on n'a pas le droit d'avoir des murs blancs). Dans ce cas, $\gamma = \{(centre\_ville \vee proche\_metro) \wedge murs\_blancs \wedge \neg calme\}$.

En tenant compte des contraintes du domaine et de la réglementation, on peut dire qu'une maison qui satisfait $\{proche\_metro, murs\_blancs, \neg calme\}$ sera une maison qui satisfait au mieux les exigences du participant. Les deux exigences prioritaires sont satisfaites (la maison est soit au centre ville, soit près d'une station de métro et a des murs blancs), mais la troisième exigence n'est pas satisfaite. Alors que les exigences du participant n'étaient pas globalement cohérentes avec les lois du domaine et la réglementation, l'ordonnancement des exigences grâce à une position a permis de trouver un compromis dans les choix possibles de l'agent.

2.4.3 Conclusion

Le formalisme développé par Cholvy et Hunter permet donc de raffiner les exigences de chaque participant. Chacun d'entre eux peut ordonner ses exigences pour désigner quelles sont celles qui sont à son avis prioritaires et quelles sont celles qui le sont moins. En cas de conflit, on essayera donc toujours de maximiser la satisfaction des participants. De plus, leur formalisme permet d'exprimer une réglementation et des lois du domaine et de vérifier que les exigences de chaque participant sont cohérentes avec ces notions. Dans le cas contraire, l'utilisation de la position exprimée par le participant permet de trouver une alternative qui le satisfera au mieux.

2.5 Conclusion

Il existe peu de formalismes qui utilisent la logique mathématique comme outil formel d'expression d'exigences. Bien sûr, la plupart des formalismes présentés dans les sections précédentes utilisent la logique propositionnelle ou la logique du premier ordre pour représenter les exigences, mais c'est surtout pour pouvoir gérer les incohérences possibles entre exigences comme nous le verrons dans la partie II. La logique n'est pas, dans la plupart des cas, utilisée comme un moyen de modélisation avancée des exigences, mis à

part dans le formalisme de Cholvy et Hunter qui permet de raffiner la notion d'exigence en utilisant des positions.

On peut revenir sur le formalisme de Perrussel dans [111]. Grâce à la logique modale qu'il a développée pour raisonner sur des contextes, on peut exprimer de façon très précise le contenu des spécifications de chaque point de vue. En particulier, l'extension au premier ordre nous permet de quantifier non seulement des variables qui appartiennent au domaine des objets, mais également des variables contextuelles qui vont permettre d'exprimer des liens inter-points de vue. Par contre, comme nous le verrons dans la partie II, l'existence d'incohérence dans les exigences va se traduire avec ce formalisme par une révision des exigences des participants concernés. On n'a pas comme dans [44] la possibilité d'ordonner les exigences de chaque participant pour pouvoir fournir des alternatives en cas de problème.

Le formalisme développé par Cholvy et Hunter dans [44] permet quant à lui d'affiner les exigences des participants en leur proposant d'exprimer des positions sur ces exigences. Cette opportunité pour chacun des participants nous paraît très intéressante : le raffinement obtenu sur les exigences permet en effet de trouver des solutions adaptées à chaque participant lorsque les exigences prises dans leur ensemble posent problème (incohérence avec la réglementation etc). Nous suivrons leur proposition et nous développerons donc un formalisme d'expression des exigences qui se fondera sur les positions. De plus, Cholvy et Hunter prennent en compte la réglementation et le lois du domaine dans leur formalisme. C'est un point très important, car la plupart des formalismes traitant de l'ingénierie des exigences ne tiennent pas compte de ces deux éléments. Or tout objet à concevoir est contraint par l'existence de lois physiques qui empêchent certaines réalisations et surtout par une réglementation en vigueur qu'il faut respecter. Nous nous attacherons donc à modéliser la réglementation et les lois du domaine dans notre formalisme. On peut juste regretter que le formalisme de Cholvy et Hunter n'ait pas un pouvoir d'expression de phrases normatives très important (ils utilisent la logique propositionnelle). La modélisation de réglementation et le raisonnement normatif est un domaine de recherche vaste et nous allons présenter les différentes propositions qui ont été faites jusqu'à présent dans ce domaine dans le chapitre suivant.

Chapitre 3

Raisonnement normatif (état de l'art)

Le raisonnement normatif désigne le raisonnement que l'on fait avec les notions d'obligation, de permission et d'interdiction. En particulier, dans l'étude de systèmes normatifs (des ensembles d'agents humains ou artificiels régis par des règles précises), on aimerait pouvoir représenter ce que les agents doivent faire, peuvent faire, s'ils sont autorisés à violer une partie de la réglementation etc. Lorsque l'on veut raisonner avec des phrases normatives, de nombreux problèmes apparaissent. On est en effet souvent obligé de raisonner avec des connaissance annexes comme la croyance, la capacité, l'intention pour pouvoir déterminer si un être humain, un système informatique ou une organisation respecte une réglementation en vigueur. Nous verrons dans la partie III comment intégrer ces notions dans un cadre plus détaillé. Dans ce chapitre, nous allons nous intéresser plus en détail à la représentation de phrases normatives et des problèmes qui en découlent.

L'étude du raisonnement normatif conduit à l'étude de la logique déontique. Les origines de la logique déontique se situent dans l'étude de la logique « philosophique », mais son champ d'application s'étend maintenant à l'informatique (contrôle d'accès, politiques de sécurité, contraintes d'intégrité sur des bases de données), la loi (systèmes experts légaux) et les théories organisationnelles (responsabilités, pouvoir). Historiquement, les études contemporaines de la logique déontique sont le fruit d'un fort courant de pensée philosophique nordique. On peut citer par exemple von Wright [138], Åqvist [2], Føllesdan et Hilpinen [57], Hansson [66] ou Chisholm [26]. De nombreuses autres approches sont apparues de nos jours en particulier pour traiter des problèmes spécifiques :

- approches utilisant des logiques temporelles type *branching-time* [68, 69, 17, 18] ;
- approches utilisant la notion d'agent [100, 101, 23] ;
- logiques input/output [95, 96, 97] ;
- approches algébriques [93]...

Nous considérerons ici que le fait de donner une valeur de vérité à une phrase normative comme « il est interdit que α soit vraie » ne pose pas de problème. On pourrait considérer comme Makinson dans [94] que le fait de donner une valeur de vérité à un énoncé normatif n'a pas de sens puisque celui ci ne décrit pas un état du monde, mais un état idéal. La justification philosophique de ce point de vue repose sur la distinction qui doit être faite entre des *normes* et des propositions qui expriment certaines propriétés de ces normes (et qui peuvent donc être valuées). Notre objectif est de pouvoir raisonner sur des phrases normatives, en particulier déterminer quelles normes peuvent être dérivées à partir d'un

ensemble de normes prédéfini, et dans un soucis de simplicité et par abus de langage, nous confondrons dans cette étude normes et propositions représentant des normes.

D'autre part, nous ne ferons pas ici la distinction entre l'obligation de faire quelque chose et l'obligation qu'une proposition soit vraie. Par exemple, on peut distinguer l'obligation qu'une lettre soit postée et l'obligation de poster cette lettre. Nous considérerons ici que les énoncés normatifs ne portent que sur des états possibles du monde.

3.1 Une logique déontique : SDL

Intéressons nous de plus près à SDL [25] (Standard Deontic Logic), une logique modale propositionnelle normale qui est habituellement utilisée pour représenter des phrases normatives. Le langage de SDL se compose d'un langage propositionnel, d'un opérateur d'obligation O ($O(\alpha)$ signifie « *il est obligatoire que* α *soit vraie* »), un opérateur de permission défini par $P\alpha \equiv_{def} \neg O\neg\alpha$ et d'un opérateur d'interdiction défini par $F\alpha \equiv_{def} O\neg\alpha$. SDL est une logique modale de type (KD) qui respecte donc les schémas d'axiomes et règles d'inférence suivants :

(PC) les axiomes de la logique propositionnelle
(OK) $O(\alpha \rightarrow \beta) \rightarrow (O\alpha \rightarrow O\beta)$
(OD) $O\alpha \rightarrow \neg O\neg\alpha$
(O-Nec) $\frac{\vdash \alpha}{\vdash O\alpha}$
(MP) $\frac{\alpha \quad \alpha\rightarrow\beta}{\beta}$

On considère qu'une obligation $O\alpha$ est violée ssi $O\alpha \wedge \neg\alpha$ est vraie.

Si SDL permet de modéliser rapidement et simplement des phrases normatives, elle souffre en revanche d'un certain nombre de défauts. Un grand nombre de paradoxes (énoncés intuitivement incorrects) sont des théorèmes de SDL et certaines phrases normatives ne peuvent pas être exprimées correctement dans SDL. Nous allons commenter ces défauts en suivant la présentation de [22].

3.1.1 Les paradoxes dûs à (ORM)

Comme toutes les logiques modales normales, SDL contient la règle (RM) qui est la suivante : si $\vdash \alpha \rightarrow \beta$ alors $\vdash O\alpha \rightarrow O\beta$. De la validité de cette règle découle un certain nombre de paradoxes :

- le paradoxe de Ross : $O\alpha \rightarrow O(\alpha \vee \beta)$ est un théorème de SDL. Une interprétation de ce paradoxe est la suivante : *s'il est obligatoire de poster la lettre, alors il est obligatoire de la poster ou de la brûler.*
  Le paradoxe de Ross peut laisser penser que de chaque obligation on peut dériver une obligation contre-intuitive puisque l'on laisse le choix à l'agent entre deux propositions qui sont intuitivement contradictoires. On peut tout d'abord remarquer que la satisfaction de $O(\alpha \vee \beta)$ n'implique pas la satisfaction de $O\alpha$, ce qui réduit un peu la portée du paradoxe. Le fait de brûler la lettre impliquera donc quand même une violation de l'obligation de la poster. On peut également raisonner en terme de violation : si α est obligatoire et que α est actuellement fausse, combien d'obligations ont été violées ? À cause du paradoxe de Ross, la violation d'une obligation $O\alpha$ va impliquer la violation de toutes les obligations $O(\alpha \vee \beta)$ pour lesquelles β est une proposition fausse dans l'état actuel du monde, ce qui peut paraître étrange.

Pour éviter le paradoxe de Ross, on est donc obligé de limiter les dérivations à partir d'un ensemble de normes. Il ne faut autoriser à dériver que les obligations minimales au sens de l'implication. Par exemple, si on considère un ensemble de normes qui contient $O(poster)$ (il est obligatoire de poster la lettre) mais pas $O(bruler)$ (il est obligatoire de brûler la lettre), il semblerait assez intuitif de pouvoir bloquer la dérivation de $O(poster \vee bruler)$.

- le paradoxe du libre choix : $P(\alpha \vee \beta) \rightarrow (P\alpha \wedge P\beta)$ n'est pas un théorème de SDL. En effet si $P(\alpha \vee \beta) \rightarrow (P\alpha \wedge P\beta)$ était un théorème de SDL, $P\alpha \rightarrow (P\alpha \wedge P\beta)$ serait également un théorème de SDL (d'après la règle (ORM)). On pourrait donc en déduire par exemple que s'il est permis d'aller au cinéma, il est également permis de tuer le président.
  Malgré cela, la signification de $P(\alpha \vee \beta) \rightarrow (P\alpha \wedge P\beta)$ semble assez intuitive : s'il est permis que $\alpha \vee \beta$ soit vraie alors il est évidemment permis que α soit vraie et il est permis que β soit vraie. De nombreux chercheurs comme Carmo et Jones (cf. [22]) pensent que ce paradoxe est en fait un pseudo problème. On peut en effet réfléchir à la signification de $P(\alpha \vee \beta)$. Soit on veut exprimer le fait que α et β sont toutes les deux permises et alors on doit écrire directement $P\alpha \wedge P\beta$. Soit on veut exprimer que α est permise ou que β est permise, et dans ce cas on doit écrire $P\alpha \vee P\beta$.
- le paradoxe du bon samaritain : comme $\alpha \wedge \beta \rightarrow \beta$ est un théorème de la logique propositionnelle, d'après la règle (ORM), on peut en déduire que $O(\alpha \wedge \beta) \rightarrow O\beta$ est un théorème de SDL. Prenons un exemple : si l'on considère la phrase *Marie aide Jean qui a eu un accident* et que l'on la représente par *Marie aide Jean et Jean a eu un accident*, alors on peut en déduire que s'il est obligatoire que Marie aide Jean qui a eu un accident, alors il est obligatoire que Jean ait un accident.
  En ce qui nous concerne, nous pensons que la modélisation de la phrase *Marie aide Jean qui a eu un accident* par *Marie aide Jean et jean a eu un accident* est un peu excessive. Il existe en effet une relation de causalité forte entre le fait que Marie aide Jean et que Jean ait eu un accident. Ce ne sont pas deux faits complétement indépendants ce que laisserait supposer la modélisation précédente. La modélisation de la phrase par une conjonction ne pourrait donc être pas assez suffisante pour exprimer correctement les relations sous-jacentes entre le fait que Marie aide Jean et que Jean ait eu un accident.
- le paradoxe déontique/épistémique : si l'on considère que l'on a également un opérateur modal de connaissance (qui vérifie donc le schéma d'axiome (T)), alors toujours d'après (ORM), on peut en déduire le paradoxe suivant : *s'il est obligatoire qu'un policier sache qu'un crime a été commis, alors il est obligatoire qu'un crime a été commis*. Nous ne revenons pas sur ce paradoxe qui est lui aussi une conséquence du fait que toute conséquence logique de ce qui est obligatoire est également obligatoire.

La règle (ORM) est donc un donc un générateur de paradoxe de par sa nature même. Il est en effet difficilement acceptable de considérer que toute conséquence logique de quelque chose d'obligatoire est également obligatoire (le même problème se pose avec les opérateurs modaux épistémiques, doxastiques ou de type "faire en sorte que" : il paraît en effet difficilement concevable de dire que si un agent fait en sorte que α soit vraie, alors il fait en sorte que toutes les conséquences logiques de α soient vraies). (ORM) est une règle commune à toutes les logiques modales normales, donc le fait de l'enlever de SDL nous amenerait vers une logique modale classique plus compliquée.

3.1.2 Problème dû à la règle (O-Nec)

La règle (O-Nec) ou règle de O-nécessitation pose également un problème : tous les théorèmes de SDL deviennent en effet obligatoires à cause de cette règle. Or il est légitime lorsque l'on parle d'obligation de supposer que toute obligation peut être satisfaite ou violée. Dans le cas des théorèmes de SDL, cette notion de satisfaction et de violation ne peut plus intervenir.

Par exemple, quelle que soit la proposition φ, $O(\varphi \vee \neg\varphi)$ est vraie. En particulier, il est obligatoire de manger ou de ne pas manger, ce qui intuitivement peut paraître étrange.

On peut remarquer que les mêmes problèmes se posent en logique épistémique et doxastique : un agent connaît tous les théorèmes d'une logique modale de connaissance par exemple (voir le problème de « l'omniscience logique » dans [67]), mais aussi dans des logiques modales d'action, puisque ce que l'on peut faire doit pouvoir être également évitable [122].

3.1.3 Problème dû au schéma d'axiome (OD)

Le schéma d'axiome (OD) permet d'éviter que deux obligations ne soient contradictoires : φ ne peut pas être obligatoire en même temps que $\neg\varphi$. Il n'est donc pas possible d'exprimer un conflit entre deux obligations. Or de nombreux systèmes normatifs contiennent des obligations contradictoires. Prenons un exemple connu, le paradoxe du soldat chrétien :

- *pour un soldat, il est obligatoire de tuer* ;
- *pour un chrétien, il est interdit de tuer.*

Si l'on considère un soldat chrétien, quelle est l'obligation qui s'applique à celui-ci ? D'un côté, il est obligé de tuer à cause de sa fonction de soldat, mais il lui est également interdit de tuer de part sa religion. On pourrait penser que l'utilisation d'obligations conditionnelles permettrait de représenter ce genre de conflit d'obligations (appelé également dilemme moral), mais comme nous allons le voir dans la section suivante, les obligations conditionnelles ne règlent pas le problème d'obligations contradictoires. On ne peut donc pas représenter de dilemme moral en utilisant SDL. Par contre, le problème du dilemme moral peut être résolu en indiçant les sources d'émission des normes (cf. [32]). Chacune des sources émet des exigences en utilisant SDL et on ordonne les sources suivant leur degré d'importance. Dans l'exemple du soldat chrétien, on peut supposer que deux sources émettent les deux obligations contradictoires : une « religieuse » et une autre « étatique ». Le soldat chrétien règle son dilemme en considérant une source comme étant prioritaire par rapport à l'autre.

3.1.4 Obligations conditionnelles en SDL

Jusqu'à présent, nous avons utilisé SDL pour représenter des obligations non conditionnelles. Dans le cas du dilemme moral du soldat chrétien, les deux obligations contradictoires sont en fait soumises à condition : si je suis un soldat, alors il est obligatoire que je tue et si je suis un chrétien, alors il m'est interdit de tuer. Considérons un opérateur $O(\beta|\alpha)$ dont la signification est la suivante : *si α est vraie, alors il est obligatoire que β soit vraie.* En utilisant SDL, on a deux solutions pour représenter de telles obligations conditionnelles :

option 1 $O(\beta|\alpha) \equiv_{def} \alpha \rightarrow O(\beta)$

option 2 $O(\beta|\alpha) \equiv_{def} O(\alpha \rightarrow \beta)$

Ces deux formalisations d'obligations conditionnelles ont en commun deux propriétés :

(UN) $\vdash O(\beta) \leftrightarrow O(\beta|\top)$
(SA) $\vdash O(\beta|\alpha) \rightarrow O(\beta|\alpha \wedge \gamma)$

La première de ces propriétés est une bonne propriété : une obligation comportant une tautologie dans sa partie conditionnelle est équivalente à une obligation non conditionnelle. La deuxième propriété, appelée *renforcement de l'antécédent* (Strengthening of the Antecedent) est plus problématique. Elle nous indique que cette modélisation ne permet pas d'exprimer des normes avec exceptions. Or il est très courant d'avoir des obligations, des permissions ou des interdictions qui soient sujettes à des exceptions. Supposons par exemple que l'on souhaite modéliser le dilemme moral du soldat chrétien par les deux formules $O(tuer|soldat)$ et $O(\neg tuer|chretien)$. Dans ce cas, par application de (SA), on va en déduire $O(tuer|soldat \wedge chretien)$ et $O(\neg tuer|soldat \wedge chretien)$, donc l'ensemble $\{O(tuer|soldat), O(\neg tuer|chretien)\}$ est incohérent (car (OD) est un schéma d'axiome valide de SDL). Le même problème se pose si on modélise une des phrases par une obligation non conditionnelle (on peut supposer par exemple qu'il est généralement interdit de tuer, mais qu'un soldat doit tuer), car comme le notent Carmo et Jones, la combinaison des deux théorèmes précédents permet d'inférer $\vdash O(\beta) \rightarrow O(\beta|\alpha)$, donc toute obligation non conditionnelle reste obligatoire sous n'importe quelle condition. Les problèmes rencontrés dans la modélisation d'obligations conditionnelles et en particulier dans le problème des normes régies par des exceptions n'est pas spécifique à la logique déontique. Ils existent également dans d'autres logique modales (cf. [87]).

On peut regarder quelles sont les propriétés vérifiées par chacune de ces deux formalisations. Si l'on modélise une obligation conditionnelle avec l'option 1, on obtient les théorèmes suivants :

$\vdash \neg\alpha \rightarrow O(\beta|\alpha)$
(FD) $\vdash \alpha \wedge O(\beta|\alpha) \rightarrow O(\beta)$

Le premier théorème est problématique : si par exemple il ne pleut pas, alors on peut en déduire que s'il pleut on est obligé de mettre un maillot de bain ou n'importe quelle proposition satisfaisable. Le deuxième théorème, appelé *détachement factuel*, montre comment dériver des obligations non-conditionnelles à partir d'obligations conditionnelles et de faits. Par exemple, s'il est obligatoire de rouler à gauche lorsque l'on est en Grande-Bretagne, si on est effectivement en Grande-Bretagne, alors il est obligatoire de rouler à gauche. Le détachement factuel va donc permettre de dériver les obligations qui s'appliquent dans une situation donnée : on les appellera obligations *effectives*.

Si l'on fait le choix d'utiliser l'option 2 pour modéliser des obligations conditionnelles, on obtient les deux théorèmes suivants :

$\vdash O(\neg\alpha) \rightarrow O(\beta|\alpha)$
(DD) $\vdash O(\alpha) \wedge O(\beta|\alpha) \rightarrow O(\beta)$

Tout comme pour l'option 1, le premier théorème est problématique : n'importe quelle proposition est obligatoire à condition qu'une certaine proposition soit interdite. Par exemple, s'il est interdit de rouler à gauche, alors si on roule à gauche, il est obligatoire de tuer son voisin. Le deuxième théorème, que nous appellerons *détachement déontique*, per-

met de dériver des obligations non-conditionnelles à partir d'obligations conditionnelles et non-conditionnelles. Il va permettre de dériver des obligations *idéales* dans le sens où elles ne considèrent pas la situation actuelle mais simplement l'ensemble des phrases normatives représentant une réglementation.

3.2 Contrary-to-Duties

Les paradoxes ou problèmes de représentation présentés dans la section précédente sont des problèmes qui ne sont pas propres à la logique déontique. Ils apparaissent souvent dans les logiques épistémiques ou doxastiques par exemple. Il existe par contre un problème propre à la logique déontique : les Contrary-to-Duties (CTDs). Un CTD est composé d'une obligation primaire et d'une obligation secondaire qui prend effet lorsque la règle primaire est violée. Un des premiers exemples de CTD a été donné par Chisholm en 1963 dans [26] :

(a) M. X doit aider ses voisins ;
(b) si M. X va aider ses voisins, M. X doit les prévenir ;
(c) si M. X ne va pas aider ses voisins, M. X doit ne pas les prévenir ;
(d) M. X ne va pas aider ses voisins.

Le CTD à proprement parler est formé des deux phrases (a) et (c). Il est couramment admis dans la communauté « déontique » que cet ensemble de quatre phrases est cohérent et que les phrases sont logiquement indépendantes les unes par rapport aux autres. La modélisation de cet énoncé en utilisant SDL est la suivante :

(a) $O(help)$
(b) $O(tell|help)$
(c) $O(\neg tell|\neg help)$
(d) $\neg help$

Les problèmes qui se posent sont les suivants : quelles sont les obligations qui s'appliquent effectivement à M. X ? Quelles sont les obligations qui ont été violées par M. X ? Pour répondre à ces questions, nous allons tout d'abord montrer que SDL n'est pas suffisante pour représenter correctement les CTDs, puis nous présenterons quelques approches différentes. Enfin, nous rappellerons les postulats émis par Carmo et Jones dans [22] pour pouvoir représenter des CTDs.

3.2.1 Modélisation avec SDL

Si l'on veut modéliser le paradoxe de Chisholm avec SDL, on a le choix entre les options 1 et 2 présentés dans la section précédente pour représenter les phrases (b) et (c). Examinons les différents cas possibles :

- choix de l'option 1 pour (b) et (c). La représentation de (b) et de (c) est donc la suivante :
  (b) $help \rightarrow O(tell)$
  (c) $\neg help \rightarrow O(\neg tell)$
  Cette représentation est parfaitement cohérente et elle permet de plus de dériver $O(\neg tell)$. Par contre, (b) est une conséquence logique de (d) et l'indépendance logique entre les phrases de l'énoncé n'est pas assurée.

– choix de l'option 2 pour (b) et (c). La représentation de (b) et de (c) est donc la suivante :
(b) $O(help \rightarrow tell)$
(c) $O(\neg help \rightarrow \neg tell)$
Grâce à la propriété de détachement déontique, on peut dériver $O(tell)$. C'est intuitivement incorrect, car M. X ne va pas aider ses voisins, mais il doit les prévenir qu'il va les aider, obligation que l'on peut dériver.
– choix de l'option 1 pour (b) et choix de l'option (2) pour (c). La représentation de (b) et de (c) est donc :
(b) $help \rightarrow O(tell)$
(c) $O(\neg help \rightarrow \neg tell)$
Dans ce cas, on ne peut rien dériver du tout : en effet, pour pouvoir dériver une obligation non-conditionnelle avec (b) il faut avoir la propriété (DD) et pour pouvoir dériver une obligation non-conditionnelle avec (c), il faut avoir la propriété (FD).
– choix de l'option 2 pour (b) et choix de l'option (1) pour (c). Ce choix semble le plus correct, car l'option (2) nous permet d'utiliser le détachement déontique sur (b) et l'option (1) nous permet d'utiliser le détachement factuel sur (c). On a donc :
(b) $O(help \rightarrow tell)$
(c) $\neg help \rightarrow O(\neg tell)$
Dans ce cas, on obtient en utilisant (FD) avec (c) et (d) $O(\neg tell)$ et en utilisant (DD) avec (a) et (b) $O(tell)$ ce qui est incohérent car (D) est un schéma d'axiomes valide de SDL. La représentation proposée est donc incohérente.

Comme nous venons de le voir exhaustivement, SDL ne permet pas représenter correctement le paradoxe de Chisholm. Les deux cas qui semblaient les plus intuitivement corrects conduisent soit à une incohérence logique, soit à une dépendance logique entre plusieurs phrases de l'énoncé.

3.2.2 Quelques approches pour représenter et raisonner avec des CTDs

Plusieurs approches ont été proposées pour pouvoir résoudre le problème des Contrary-to-Duties. On peut citer par exemple des approches basées sur des logiques temporelles (cf. [3]) qui distinguent différents moments : le moment où une obligation est violée et le moment où on remplit l'obligation secondaire du CTD. Il existe également des approches basées sur des logiques d'action qui font la distinction entre la condition d'une obligation conditionnelle qui est considérée comme un état et l'obligation elle-même qui est considérée comme une action (cf. [24, 102]). Ces deux types de logique permettent très bien de représenter le paradoxe de Chisholm (où on peut inclure facilement une dimension temporelle et faire la distinction entre « il est obligatoire de faire quelque chose » et « il est obligatoire d'être dans un certain état »).

Cependant, considérons le CTD suivant :

(a) il ne doit pas y avoir de chien ;
(b) s'il n'y a pas de chien, il ne doit pas y avoir de panneau[1] ;
(c) s'il y a un chien, il doit y avoir un panneau ;
(d) il y a un chien

[1] Sous-entendu, il ne doit pas y avoir de panneau signalant la présence d'un chien.

Dans cet exemple proposé par Prakken et Sergot dans [114], la notion de temps n'apparaît pas. Les trois phrases normatives s'appliquent au même moment. Les approches temporelles ne sont donc pas adaptées pour représenter ce scénario. De même, les approches basées sur une logique de l'action ne conviennent pas, car on n'exprime pas d'obligation de faire quelque chose dans cet exemple. Nous considérerons ce scénario comme un exemple de référence dans la suite du mémoire et nous nous y référerons par « le scénario du chien ».

Récemment, de nombreuses approches ont affirmé que le raisonnement avec des CTDs n'était qu'un type de raisonnement révisable particulier et que les techniques propres au raisonnement non-monotone pouvaient s'appliquer directement sur ce genre de scénario (cf. [99] par exemple). Si les principes du raisonnement non-monotone peuvent être facilement adaptés pour raisonner avec des dilemmes moraux ou des obligations conflictuelles [113, 133, 135, 68], il n'en est pas de même en ce qui concerne les CTDs. Considérons l'exemple suivant :

(a') il ne doit pas y avoir de chien sauf s'il y a un panneau

Il apparaît clairement que le raisonnement non-monotone convient parfaitement pour représenter cet enoncé. En effet, il est a priori interdit d'avoir un chien, mais si il y a un panneau, alors avoir un chien est autorisé. En particulier, s'il y a un panneau, la première interdiction (on parle alors de *prima facie*) n'est plus en vigueur et c'est la permission d'avoir un chien qui entre en vigueur. Dans le cas du CTD, la phrase (c) n'est pas une exception à l'obligation (a). S'il y a un chien, l'obligation de ne pas en avoir est toujours en vigueur et est violée. Elle n'est pas « supplantée » par l'obligation d'avoir un panneau qui entre alors en vigueur. On ne peut donc pas utiliser directement une approche non-monotone pour raisonner avec des CTDs.

Revenons sur le problème de SDL pour raisonner avec des CTDs. Comme le précisent par exemple [88, 74, 130, 115], SDL ne permet pas de raisonner avec des CTDs car la sémantique de SDL ne peut pas distinguer différents niveaux de non-idéalité. Les mondes sont soit idéaux, soit non-idéaux.

Or nous pensons que dans le scénario du chien, la phrase « s'il y a un chien, alors il doit y avoir un panneau » exprime le fait que certains mondes non-idéaux sont plus idéaux que d'autres mondes non-idéaux : les mondes dans lesquels il y a un chien sont non-idéaux, mais parmi ceux-ci, les mondes dans lesquels il y a un panneau sont « meilleurs » que les mondes dans lesquels il n'y en a pas.

En utilisant cette approche, la phrase « si α est vraie, alors il est obligatoire que β soit vraie » se traduit sémantiquement de la façon suivante : β est vraie dans un sous-ensemble des α-mondes et ces mondes sont les meilleurs α-mondes pour une relation d'ordre représentant la préférence relative qu'il existe entre les mondes possibles. Reprenons l'exemple du chien. Si on considère un langage à deux variables variable propositionnelles *chien* et *panneau*, il existe quatre mondes possibles : $w_1 = \{chien, panneau\}$, $w_2 = \{chien, \neg panneau\}$, $w_3 = \{\neg chien, panneau\}$ et $w_4 = \{\neg chien, \neg panneau\}$. Si l'on suppose que l'on dispose d'une relation de préordre $\leq$ sur les mondes possibles ($w_1 \leq w_2$ signifiant par exemple que w_1 est meilleur que w_2), alors on peut interpréter l'énoncé « s'il y a un chien, il doit y avoir un panneau » par l'assertion $w_1 < w_2$: le monde où il y a un chien et un panneau est strictement meilleur que le monde où il y a un chien et où il n'y a pas de panneau (en utilisant le préordre, on a $w_1 \leq w_2$ et $w_2 \not\leq w_1$).

Hansson fut le premier à proposer l'utilisation d'une telle structure sémantique dans [66] dans sa logique DSDL3 (pour *Dyadic Standard Deontic Logic 3)*. Lewis a poursuivi

ses travaux dans [88] en fournissant une étude logique poussée de ce genre de sémantique. Récemment, de nombreuses approches se fondent sur l'utilisation d'une telle sémantique, dite de Hansson-Lewis. Citons par exemple Prakken et Sergot dans [115] ou encore van der Torre et Tan dans [134]. Notons que Spohn, dans [126], fut le premier à proposer une axiomatisation de DSDL3.

Enfin, parmi les approches hybrides, on peut citer les travaux de van der Torre et Tan [132] qui combinent les principes des logiques des défauts (comme par exemple [119]) et la logique de préférences CDL [131] pour pouvoir raisonner avec des CTDs. Ils proposent de classifier la notion de « revisabilité »[2] en trois types distincts :

- la revisabilité factuelle : une règle par défaut comme « les oiseaux volent » peut être défaite par un fait comme « un certain oiseau ne vole pas » ;
- la revisabilité par surcharge : il existe une règle par défaut comme « les pingouins ne volent pas » qui est plus spécifique que la règle « les oiseaux volent » (on considère bien sûr que les pingouins sont des oiseaux). La révisabilité par surcharge peut elle même être classifiée en deux types en considérant la règle suivante : « les pingouins ne volent pas et vivent en Antarctique ». Supposons que l'on ait un pingouin qui ne vive pas en Antarctique. On peut distinguer deux cas :
  - soit la régle par défaut « les oiseaux volent » est encore surchargée et on parle alors de revisabilité par surcharge forte ;
  - soit la régle par défaut « les oiseaux volent » n'est plus surchargée et on parle de revisabilité par surcharge faible.

Van der Torre et Tan proposent alors d'utiliser cette classification pour pouvoir raisonner avec des CTDs et des normes avec exception. Les CTDs sont modélisé en utilisant la révisabilité factuelle (ce qui correspond intuitivement au théorème (FD) présenté précédemment), alors que les normes avec exceptions sont représentées en utilisant la révisabilité par surcharge (une norme prend le pas sur l'autre dans certains cas plus spécifiques). Une analyse complète de ces problèmes est présentée dans [132].

3.2.3 L'approche de Carmo et Jones

La diversité des approches proposées pour régler le problème de la représentation et du raisonnement avec des CTDs montre bien que ce problème reste encore ouvert. En particulier, il n'existe pas ou peu de consensus sur la représentation formelle d'un scénario comportant des CTDs. Carmo et Jones ont proposé dans [22] huit postulats qui, à leur sens, devraient être vérifiés par toute approche essayant de modéliser des CTDs. Cette initiative permet de disposer d'un consensus « minimum » sur ce qui est demandé aux formalismes logiques traitant des CTDs. En particulier, ils nous permettront de valider notre approche présentée dans la partie III. Nous rappelons ces postulats dans ce qui suit puis nous présentons la logique développée par Carmo et Jones pour raisonner avec des CTDs.

Postulats de Carmo et Jones

Rappelons le scénario du chien :

(a) il ne doit pas y avoir de chien ;
(b) s'il n'y a pas de chien, il ne doit pas y avoir de panneau ;

[2] *Defeasibility* en anglais.

(c) s'il y a un chien, il doit y avoir un panneau ;
(d) il y a un chien.

Les premiers postulats qui viennent à l'esprit sont des postulats consensuels que nous avons déjà présentés : l'ensemble des formules représentant ces quatre phrases doit être cohérent et chacune des formules doit être logiquement indépendante des autres formules. De plus, cet exemple ne fait absolument appel à des notions temporelles, la régle primaire (a) et le CTD (c) peuvent exister en même temps. Un formalisme traitant des CTDs ne devra donc pas être basé sur une approche temporelle.

Un des problèmes qui se pose est la représentation des phrases (b) et (c). Dans le langage naturel, ces deux phrases ont exactement la même structure. Or, elle n'ont pas du tout le même sens : (b) est une obligation conditionnelle « classique », alors que (c) est une obligation qui n'intervient que lorsque la règle (a) est violée. Selon Prakken et Sergot [114], (b) et (c) doivent avoir des représentations logiques différentes, puisque même si (b) est une obligation conditionnelle, elle reste une obligation primaire, alors que (c) représente un cas de violation de la règle (a). Carmo et Jones ont abandonné ce point de vue, car les considérations de Prakken et Sergot sont trop dépendantes du contexte. Supposons par exemple que la loi soit révisée et qu'il ne soit plus interdit d'avoir un chien. On peut donc enlever (a) de l'ensemble des normes. Dans ce cas, on est également obligé de changer la formulation logique de (c). De la même façon, l'ajout de nouveaux CTDs dans le système doit entrainer la vérification de l'écriture des normes existantes.

Carmo et Jones arrivent donc à un premier ensemble de postulats que doit respecter un formalisme logique traitant des CTDs :

(i) cohérence de l'ensemble des formules qui représentent (a), (b), (c) et (d)
(ii) indépendance logique des membres
(iii) applicabilité à des exemples non-temporels
(iv) structure logique identique pour les deux conditions (b) et (c)

Comme on l'a vu précédemment sur le paradoxe de Chisholm, respecter à la fois (DD) et (FD) aboutit à une incohérence logique si on accepte l'axiome (OD). Même si on ne peut pas accepter en même temps (DD) et (FD), il paraît raisonnable de pouvoir dériver à la fois que :

– dans des conditions idéales, il y a une obligation *idéale* de ne pas avoir de chien et de ne pas avoir de panneau ;
– dans certaines circonstances (par exemple la violation de (a)), il y a une obligation *effective* de mettre un panneau.

Un formalisme logique devra également être capable de représenter le fait qu'une obligation a été violée. Carmo et Jones ajoutent donc trois postulats aux précédents :

(v) capacité de dériver des obligations idéales
(vi) capacité de dériver des obligations effectives
(vii) capacité de représenter qu'une règle a été violée

On peut remarquer que la logique proposée par Jones et Pörn dans [74] vérifie ces sept postulats. Or comme le montrent Prakken et Sergot dans [114], l'approche de Jones et Pörn génére ce qu'ils appellent une « idiotie pragmatique »[3]. Dans le scénario du chien, on peut dériver les deux obligations $O(\neg chien)$ et $O(panneau)$: il ne doit pas y avoir de

[3] *Pragmatic oddity* en anglais.

chien, mais il est obligatoire qu'il y ait un panneau, ce qui est intuitivement incorrect. Carmo et Jones rajoutent donc un dernier postulat :

(viii) capacité d'éviter les idioties pragmatiques

Remarquons que ce postulat est totalement informel : on veut juste avoir un formalisme qui evite une situation intuitivement incorrecte.

Nous disposons donc maintenant d'un ensemble de postulats que doit vérifier un formalisme dont le but est de raisonner avec des CTDs. Nous nous y référerons par la suite sous le nom de « postulats de Carmo et Jones ».

La logique développée par Carmo et Jones

Carmo et Jones proposent dans [22] d'utiliser deux niveaux d'obligations pour pouvoir résoudre le problème des CTDs. Ils considérent donc des obligations *idéales* (utilisant l'opérateur monoadique O_i) et *effectives* (utilisant l'opérateur monoadique O_a pour *actual obligation*) dans le sens que nous avons défini précédemment. Les normes s'appliquent à un agent (humain ou système artificiel) et il faut donc être capable de distinguer quelles sont les obligations qui s'appliquent idéalement et quelles sont les obligations qui s'appliquent en tenant compte du contexte actuel. Ils disposent également d'un opérateur dyadique d'obligation conditionnelle $O(-|-)$ dont la signification est plus nuancée que celle employée jusqu'à présent. $O(\beta|\alpha)$ signifie en effet : dans tous les contextes où α est fixée, il est obligatoire que β soit vraie, si c'est possible.

La première remarque que l'on peut faire est la suivante : les propositions qui peuvent être obligatoires sont des propositions possibles (dans un sens que nous définirons après). Il ne peut donc pas être obligatoire d'être dans un état inatteignable. Par exemple, il ne peut pas être obligatoire pour un humain qu'il fasse beau, puisqu'il n'a pas la possibilité de faire apparaître le soleil selon son bon vouloir. La deuxième remarque concerne les contextes dans lesquels une proposition apparaissant en partie conditionnelle d'une obligation est fixée. Pour représenter cette notion, Carmo et Jones utilisent deux niveaux de nécessité, que nous appelons ici *nécessité du fait de l'agent* et *nécessité indépendante de l'agent*.

La nécessité indépendante de l'agent est dénotée ici par $\Box_i$ (son opérateur dual $\Diamond_i$ concerne la possibilité indépendante de l'agent). Intuitivement, $\Box_i\alpha$ signifie que α est fixée dans une certaine situation et ce quelles que soient les décisions de l'agent. $\Box_i\neg panneau$ signifie par exemple qu'il ne peut pas y avoir de panneau et ce indépendamment de la volonté de l'agent (il n'y a pas de panneau et il n'a aucun moyen d'en avoir un). Cet opérateur est utilisé pour dériver des obligations idéales grâce à l'axiome :

$$O(\beta|\alpha) \wedge \Box_i\alpha \wedge \Diamond_i\beta \wedge \Diamond_i\neg\beta \rightarrow O_i(\beta)$$

Un exemple d'utilisation de cet axiome dans le scénario du chien est le suivant :
- s'il y a un chien, alors il doit y avoir un panneau ;
- il y a un chien et il n'y a aucun moyen de l'enlever (le chien est très méchant et si on essaye de le faire partir de force il risque de s'énerver) ;
- on a la possibilité de mettre un panneau ;
- on a la possibilité de ne pas mettre de panneau.

Dans ce cas, on doit idéalement mettre un panneau signalant la présence du chien.

La nécessité du fait de l'agent est notée $\Box_a$. $\Box_a\alpha$ signifie que α est fixée car l'agent en a décidé ainsi. $\Box_a\neg panneau$ signifie donc qu'il ne peut pas y avoir de panneau car l'agent

l'a décidé (en particulier, il avait les moyens de mettre un panneau). $\Diamond_a \neg panneau$ signifie que l'agent n'a pas décidé de mettre un panneau. On utilise cette notion de nécessité dépendant de l'agent à travers l'axiome :

$$O(\beta|\alpha) \wedge \Box_a \alpha \wedge \Diamond_a \beta \wedge \Diamond_a \neg\beta \rightarrow O_a(\beta)$$

Par exemple :
– s'il y a un chien, alors il doit y avoir un panneau ;
– il y a un chien et l'agent a décidé de ne pas l'enlever ;
– l'agent n'a pas décidé de ne pas mettre de panneau ;
– l'agent n'a pas décidé de mettre un panneau.

Alors l'agent a l'obligation effective de mettre un panneau indiquant qu'il y a un chien. On peut remarquer que sur cet exemple, comme on a $\neg\Box_i chien$ (l'agent ayant décidé de garder le chien, *chien* n'est pas fixée indépendamment de la volonté de l'agent), on peut dériver l'obligation idéale de ne pas avoir de chien. Les deux types d'obligations permettent donc de dériver des obligations contradictoires.

Cette approche distinguant les obligations idéales et les obligations effectives est très intéressante car elle permet d'obtenir un comportement de type (DD) dans les cas où celui-ci est intuitivement correct (les obligations idéales) et un comportement de type (FD) pour la dérivation d'obligations effectives. Nous nous appuyerons sur ce travail pour définir les notions d'obligations idéales et effectives dans la partie III.

3.3 Conclusion

La formalisation en logique d'énoncés déontiques est un problème complexe. Elle fait souvent appel à d'autres notions comme l'action, la capacité, l'intention ou le temps. De plus, les paradoxes levés par les logiques modale comme SDL ou le problème des Contrary-to-Duties ne facilitent pas la modélisation de systèmes normatifs.

Cependant, on peut souligner que l'approche de Hansson-Lewis est très intéressante. Ordonner les mondes selon une relation d'idéalité permet de résoudre le problème des CTDs (à condition d'ajouter deux niveaux d'obligations comme Carmo et Jones) et de travailler avec des normes avec exception. Bien sûr, cette approche ne permet pas de résoudre directement les paradoxes dûs à (ORM) par exemple, mais on peut l'adapter pour y parvenir.

Remarquons surtout que cette approche est encore fondée sur la notion de préférences. Nous avons vu dans le chapitre précédent que les travaux de Cholvy et Hunter proposant d'exprimer des positions sur les exigences grâce à une relation de préférences étaient judicieux. Leur formalisme souffrait par contre d'un manque d'expressivité dans la modélisation de réglementation. Nous allons donc essayer de trouver un formalisme logique permettant de raisonner sur des préférences pour pouvoir exprimer à la fois des exigences et des phrases normatives.

Chapitre 4

Représentation de préférences (état de l'art)

Comme nous venons de le voir dans les deux chapitres précédents, la notion de préférence apparaît à la fois dans la représentation d'exigences et de phrases normatives. Nous allons donc nous intéresser dans ce chapitre à différents formalismes qui permettent de représenter des préférences. Le but de ces différents formalismes est de pouvoir exprimer les préférences de façon concise sous la forme d'un ensemble de formules logiques puis de générer un préordre sur les mondes possibles. Nous n'allons nous intéresser qu'à des formalismes fondés sur la logique propositionnelle. Dans un premier temps, nous allons présenter quelques logiques pondérées, puis nous allons faire un rapide état de l'art sur la représentation de préférences contextuelles. De cet état de l'art nous tirerons une logique modale que nous utiliserons dans toute la suite du mémoire.

4.1 Les logiques pondérées : un bref aperçu

Les logiques pondérées sont des logiques qui ont été définies pour représenter des connaissances incertaines. Elles peuvent être également utilisées pour représenter des préférences. Le principe des logiques pondérées est le suivant : on associe à chaque formule du langage un poids (i.e. une valeur réelle) qui a une signification différente selon la logique que l'on utilise. Ces logiques manipulent donc un ensemble de couples (φ, α) où φ est une formule propositionnelle et α le poids associé à cette formule. Nous allons présenter rapidement deux logiques pondérées, la logique possibiliste et la logique des pénalités, et leur utilisation dans le cadre de la représentation de préférences.

4.1.1 La logique possibiliste

La logique possibiliste a été définie par Dubois et Prade (cf. [51] pour plus de détails). Cette logique permet à la base de représenter des connaissances incertaines et de raisonner avec celles-ci. Fariñas del Cerro et Herzig ont donné une caractérisation en logique modale de la logique possibiliste dans [49]. Lang a montré dans [84] que l'on pouvait se servir de la logique possibiliste pour représenter des préférences. Nous n'allons présenter que cette approche dans ce qui suit. On considère donc que l'on dispose d'un ensemble de formules en logique possibiliste qui représente un ensemble de préférences (nous verrons

plus loin comment représenter des préférences). Étant donné un ensemble de préférences, le problème est de déterminer si un monde ou une formule satisfait cet ensemble.

Considérons $PROP$ un langage propositionnel et W l'ensemble des interprétations de ce langage. Une *distribution de possibilité* sur W est une fonction $\pi : W \to [0,1]$. Si on considère un monde $w \in W$, il existe trois cas :

- $\pi(w) = 0$, alors w est inacceptable (au moins une préférence est violée par ce monde) ;
- $\pi(w) = 1$, alors w est acceptable ;
- $0 < \pi(w) < 1$, alors w est quelque peu acceptable.

Une distribution de possibilité sur W induit deux fonctions de l'ensemble des formules de $PROP$ dans $[0,1]$:

- une *mesure de possibilité* notée Π telle que $\forall \varphi \in PROP\ \Pi(\varphi) = sup\{\pi(w) : w \models \varphi\}$.
  Le degré de possibilité d'une formule φ indique à quel point cette formule est cohérente avec l'ensemble des préférences dont on dispose. Si l'on considère une formule φ, trois cas se présentent :
  - $\Pi(\varphi) = 0$ et alors φ est interdite, donc $\neg\varphi$ est une préférence que l'on peut déduire de l'ensemble de départ ;
  - $\Pi(\varphi) = 1$ et alors φ est permise, mais ce n'est pas forcément une préférence. En cas d'indifférence totale, φ et $\neg\varphi$ sont permises et $\Pi(\varphi) = \Pi(\neg\varphi) = 1$;
  - $0 < \Pi(\varphi) < 1$ et alors φ est quelque peu permise.
- une *mesure de nécessité* notée N telle que $\forall \varphi \in PROP\ N(\varphi) = inf\{1 - \pi(w) : w \models \neg\varphi\}$.
  Le degré de nécessité d'une formule traduit l'importance que l'agent accorde à sa satisfaction. Là encore, si l'on considère une formule φ, trois cas se présentent :
  - $N(\varphi) = 0$ et alors on ne peut dire que φ est préférée. Il se peut également que $N(\neg\varphi) = 0$;
  - $N(\varphi) = 1$ et alors φ est une des formules les plus préférées ;
  - $0 < N(\varphi) < 1$ et alors φ est une formule qui n'est pas parmi les plus préférées.

Il est important de remarquer qu'en logique possibiliste, la valeur des poids accordés à chaque formule n'a pas d'importance, c'est l'ordre relatif de ces poids qui est important.

Pour construire un ensemble de formules représentant les préférences d'un agent, on utilise une *base de préférences N-valuée*. Une formule N-valuée est un couple $(\varphi\ \alpha)$ où φ est une formule propositionnelle et α un réel appartenant à $[0,1]$. $(\varphi\ \alpha)$ signifie que $N(\varphi) \geq \alpha$ et donc que le degré de préférence de l'agent vis-à-vis de φ est d'au moins α. Une base de préférences N-valuée est un ensemble fini de formules N-valuées.

Un résultat important est le suivant : soit $\Sigma = \{(\varphi_i\ \alpha_i) : i \in \{1,\ldots,n\}\}$ une base de préférences N-valuées, alors il existe une distribution de possibilité la moins spécifique π_Σ satisfaisant Σ et définie par :

$$\forall w \in W \quad \pi_\Sigma(w) = \begin{cases} 1 \text{ si } w \models \bigwedge_{i \in \{1,\ldots,n\}} \varphi_i \\ \min(1 - \alpha_i : w \models \neg\varphi_i,\ i \in \{1,\ldots,n\}) \text{ sinon} \end{cases}$$

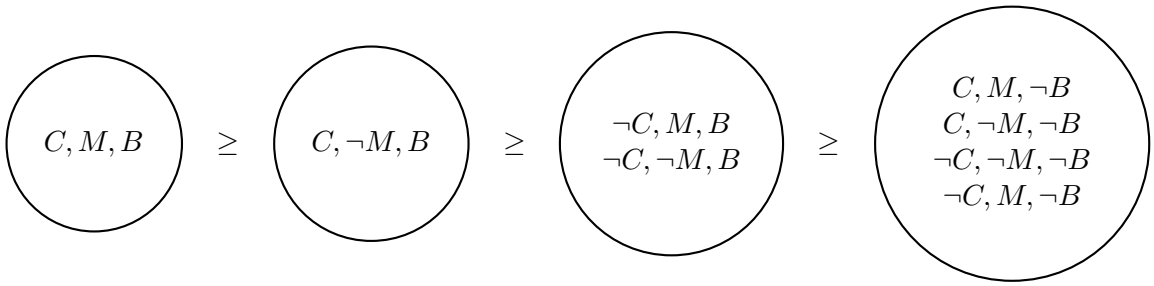
Exemple. *Considérons l'exemple suivant : supposons que M. X veuille construire une maison. Les caractéristiques de la maison sont représentées par trois variables propositionnelles : C représente le fait que la maison est proche du centre ville, M qu'elle est proche d'une station de métro et B que les murs de la maison sont blancs. Les préférences de M. X sont les suivantes :*

– *il préfére que les murs de la maison soient blancs et cette préférence est impérative, donc* $N(B) = 1$ *;*
– *il souhaite fortement d'habiter près du centre ville, par exemple* $N(C) \geq 0.7$ *;*
– *enfin, il souhaite être proche d'une station de métro, mais cette préférence est moins importante,* $N(M) \geq 0.2$.

L'ensemble des préférences de M. X peut être représenté par la base de préférences N-valuée $\Sigma = \{(B, 1), (C, 0.7), (M, 0.2)\}$. *La distribution de possibilité la moins spécifique pour* Σ *va être définie par :*

$$\pi_\Sigma(\{C, M, \neg B\}) = \pi_\Sigma(\{C, \neg M, \neg B\}) = \pi_\Sigma(\{\neg C, \neg M, \neg B\}) = \pi_\Sigma(\{\neg C, M, \neg B\}) = 0$$
$$\pi_\Sigma(\{\neg C, M, B\}) = \pi_\Sigma(\{\neg C, \neg M, B\}) = 0.3$$
$$\pi_\Sigma(\{C, \neg M, B\}) = 0.8$$
$$\pi_\Sigma(\{C, M, B\}) = 1$$

Cette distribution de possibilité induit un préordre total $\geq$ *sur les mondes possibles (*$w \geq w'$ *signifie que* w *est au moins aussi bon que* w'*) :*



Remarquons que le monde $\{C, M, B\}$ *est en fait strictement meilleur que* $\{C, \neg M, B\}$, *car* $\{C, M, B\} \geq \{C, \neg M, B\}$ *et* $\{C, \neg M, B\} \not\geq \{C, M, B\}$.

On peut remarquer que l'utilisation de la logique possibiliste ne permet pas de distinguer le monde $\{\neg C, M, B\}$ du monde $\{\neg C, \neg M, B\}$. Or le deuxième monde viole deux préférences, alors que le premier n'en viole qu'une, il paraît alors intuitif que $\{\neg C, M, B\}$ soit préféré à $\{\neg C, \neg M, B\}$. L'opérateur min n'étant pas discriminatoire, l'utilisation de nouvelles relations de préférences comme la relation discri-min (cf. [50]) permet de pallier ce problème et de raffiner l'ordre obtenu en utilisant π_Σ. Nous ne présenterons pas ces approches ici.

Grâce aux notions de base de préférences N-valuée et de distribution de possibilité la moins spécifique pour cette base, on est capable de construire un préordre sur les mondes possibles correspondant à un ensemble de préférences exprimées de façon compacte.

4.1.2 Logique des pénalités

La logique des pénalités est également une logique pondérée, mais le poids associé à chaque monde est calculé d'une manière différente. Alors que dans le cas de la logique possibiliste la définition d'une distribution de possibilité à partir d'une base de préférences utilise un opérateur mathématique de type min, le poids associé à chaque monde en logique des pénalités est déterminé en utilisant un opérateur de somme. Ainsi, la non satisfaction

de plusieurs formules de faible importance peut être équivalent à la non satisfaction d'une seule formule de grande importance. En logique des pénalités, une base de préférence est un ensemble $\Sigma = \{(\varphi_i, \alpha_i) \ : \ i \in \{1, \ldots, n\}\}$ où chaque φ_i est une formule propositionnelle et chaque α_i un réel appelé penalité représentant le « prix » à payer en cas de non satisfaction de φ_i. Chaque pénalité appartient à $[0, +\infty]$. Une pénalité de $+\infty$ signifie donc que la non satisfaction de la formule associée n'est absolument pas désirée, alors qu'une pénalité de 0 indique que la formule associée n'a aucune importance. Étant donnée une base de préférence Σ, on calcule le coût associé à chaque monde $w \in W$ noté $k_\Sigma(w)$ de la façon suivante :

$$k_\Sigma(w) = \sum_{\{(\varphi_i, \alpha_i) \in \Sigma \ : \ w \models \neg\varphi_i\}} \alpha_i$$

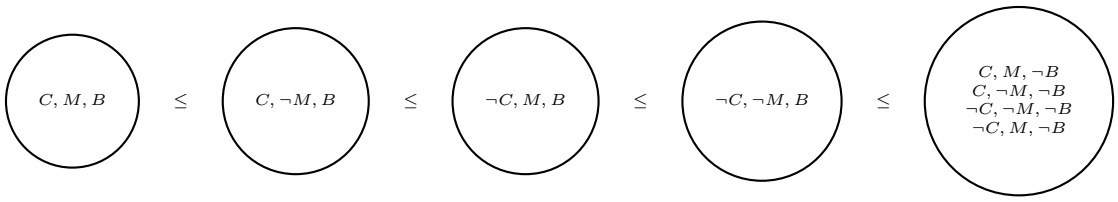
Exemple. *Reprenons l'exemple de la section précédente. On peut représenter les préférences de M. X de la façon suivante :*

- *la préférence absolue concernant les murs de la maison implique d'associer la valeur $+\infty$ à la formule atomique B ;*
- *la proximité du centre ville est assez importante, donc on peut par exemple associer la valeur 50 à la formule atomique C ;*
- *enfin, la proximité d'une station de métro est moins importante, donc on peut associer la valeur 25 à la formule atomique M.*

Les préférences de M. X sont représentées par la base de préférences $\Sigma = \{(B, +\infty), (C, 50), (M, 25)\}$. *Le coût associé à chacun des mondes possibles est le suivant :*

$k_\Sigma(\{C, M, \neg B\}) = k_\Sigma(\{C, \neg M, \neg B\}) = k_\Sigma(\{\neg C, \neg M, \neg B\}) = k_\Sigma(\{\neg C, M, \neg B\}) = +\infty$
$k_\Sigma(\{\neg C, \neg M, B\}) = 75$
$k_\Sigma(\{\neg C, M, B\}) = 50$
$k_\Sigma(\{C, \neg M, B\}) = 25$
$k_\Sigma(\{C, M, B\}) = 0$

Le préordre qui est induit sur les mondes est le suivant[1] *:*



L'utilisation de la logique des pénalités permet de distinguer le monde $\{\neg C, M, B\}$ du monde $\{\neg C, \neg M, B\}$, contrairement à l'approche présentée dans la section précédente. On notera par ailleurs que les mondes les plus préférés sont les mondes minimaux pour le préordre induit par k_Σ, alors qu'en logique possibiliste, ce sont les mondes maximaux pour le préordre induit par π_Σ. Cela s'explique aisément par le fait qu'en logique des

[1] Remarquons que la logique possibiliste ordonne les mondes selon un préordre $\geq$ alors que la logique des pénalités utilise un préordre $\leq$. Dans les deux cas, les mondes les plus préférés sont situés « à gauche » dans les figures.

pénalités, la valuation associée à chaque formule représente la pénalité correspondant à la non satisfaction de la formule alors qu'en logique possibiliste, la valuation représente le degré de préférence minimal pour la formule en question.

4.2 Préférences conditionnelles

Les deux approches présentées dans la section précédente permettent d'exprimer des préférences absolues. Elles ne permettent pas d'exprimer des préférences dépendant du contexte. Par exemple, M. X pourrait exprimer sa préférence concernant la proximité d'une station de métro d'une façon différente : « si je n'habite pas en centre ville, alors je préfère qu'il y ait une station de métro à proximité de ma future maison ». Une formalisation de cette préférence en utilisant les approches précédentes se ferait grâce à une formule comme $\neg C \rightarrow M$. Or $\neg C \rightarrow M$ est vraie lorsque C est fausse et M est vraie, mais également dès que C est vraie. Un monde qui vérifie $\neg C \wedge M$ va donc vérifier la préférence, ce qui est correct. Par contre, la préférence conditionnelle est également satisfaite dans un monde qui vérifie $C \wedge M$, mais intuitivement, la préférence de M. X n'est pas satisfaite dans ce monde. On ne peut donc pas utiliser cette formalisation pour représenter des préférences conditionnelles.

Dans ce qui suit, nous présentons des travaux qui s'intéressent à la modélisation de préférences conditionnelles. Une préférence conditionelle (ou désir conditionnel pour différencier des préférences non conditionnelles) va être dénotée par $D(\beta|\alpha)$, signifiant que dans le contexte où α est vraie, alors l'agent a une préférence pour que β soit vraie. En particulier, dans les α-mondes les plus préférés, β est vraie.

Si $\alpha = \top$, alors on écrira $D(\beta)$ au lieu de $D(\beta|\top)$. Cette préférence n'est pas une préférence absolue, contrairement à ce que nous avons vu dans la section précédente. Cela signifie que l'agent préfére *idéalement* que β soit vraie. En particulier, il peut exister un contexte particulier, par exemple α, dans lequel l'agent préfére que $\neg\beta$ soit vraie. La préférence $D(\beta)$ est défaite (au sens de la logique des défauts, cf. chapitre 3) par la préférence plus spécifique $D(\neg\beta|\alpha)$ si α est vraie.

Exemple. *Supposons que M. X exprime les préférences suivantes à propos de sa future maison :*

- *il ne préfére a priori pas être à proximité d'une station de métro (car il y a trop de passage) ;*
- *par contre, si sa maison est éloignée du centre ville, il préfére être proche d'une station de métro.*

On peut modéliser ces deux préférences par l'ensemble $\{D(\neg M), D(M|\neg C)\}$. $D(\neg M)$ *signifie que généralement, M. X ne préfére pas être à proximité d'une station de métro, mais ce n'est pas une préférence absolue (qui est vraie dans tous les contextes). En particulier, si la maison n'est pas proche du centre ville, M. X préfére être à côté d'une station de métro* ($D(M|\neg C)$)*, et la première préférence sera défaite par cette dernière.*

Les préférences conditionnelles se traduisent dans la sémantique par une relation de préordre partiel ou total sur les mondes possibles d'après leur définition. Dans l'exemple précédent, parmi les mondes vérifiant $\neg C$, les plus préférés vérifieront M. En particulier, $\{M, \neg C\}$ sera préféré à $\{\neg M, \neg C\}$, même si on a exprimé une préférence $D(\neg M)$. Il existe deux grandes approches sémantiques pour représenter des préférences conditionnelles :

– les approches *ceteris paribus* dans lesquelles $D(\beta|\alpha)$ signifie que dans le contexte α, β est préférée à $\neg\beta$, toutes choses étant égales par ailleurs. On ne peut donc comparer deux à deux que les mondes qui s'accordent sur toutes les variables propositionnelles sauf β ;
– les approches en terme d'idéalité dans lesquelles $D(\beta|\alpha)$ signifie que les α-mondes les plus préférés sont ceux qui vérifient β.

Nous allons présenter ces deux types d'approches à travers quelques formalismes.

4.2.1 Approche *ceteris paribus* : formalisme de Tan et Pearl

Nous présentons dans cette section l'approche proposée par Tan et Pearl dans [129]. Les désirs conditionnels $D(\beta|\alpha)$ sont interprétés grâce à une sémantique *ceteris paribus* : β est préféré à $\neg\beta$ toutes autres choses étant égales dans chaque α-monde. Un ensemble de telles formules permet de construire un ordre sur les mondes, puis Tan et Pearl en déduisent un ordre le plus compact possible.

Représentation de désirs conditionnels

Comme dans toutes les approches *ceteris paribus*, la traduction dans la sémantique de $D(\beta|\alpha)$ est une relation de préférence sur les paires de mondes. On compare les mondes qui s'accordent sur les variables propositionnelles qui ne font pas partie de β. Cette notion d'appartenance à une proposition est assez floue : on pourrait en donner une définition purement syntaxique par exemple, mais cela ne suffit pas. Tan et Pearl proposent donc une définition sémantique de la notion de *dépendance* d'une formule à une variable propositionnelle. Une formule β dépend d'une variable propositionnelle a ssi a apparaît dans toutes les formules logiquement équivalentes à β. L'ensemble des variables propositionnelles dont dépend β est noté $S(\beta)$ et l'ensemble des variables propositionnelles dont ne dépend pas β est noté $\overline{S}(\beta) = PROP - S(\beta)$.

Soit S un ensemble de variables propositionnelles. Deux mondes w et w' appartenant à W sont dits *S-équivalents*, noté $w \sim_S w'$, ssi $\forall a_i \in S\ w \models a_i \Leftrightarrow w' \models a_i$. Dans le cas d'un désir conditionnel $D(\beta|\alpha)$, on va donc comparer des mondes qui sont $S(\beta)$-équivalents entre eux. On dit qu'un désir conditionnel est spécifique si la condition de ce désir est un monde (par abus de langage, il s'agit en fait d'une conjonction de littéraux telle que toutes les variables propositionnelles du langage soient représentées dans la conjonction).

Il faut remarquer que dans cette approche, la condition d'un désir ne représente pas les mondes sur lesquels le désir s'applique mais les mondes dans lesquels le désir devrait être satisfait. On peut ainsi exprimer des désirs conditionnels dans lesquels la condition est contradictoire avec le désir. Par exemple, on peut souhaiter que la lumière soit allumée si elle est est éteinte.

Le contexte d'un désir représente l'ensemble des mondes qui vont être contraints par ce désir. Il est défini formellement par :

Définition 4.2.1. *Soit $D(\beta|w)$ un désir conditionnel spécifique. Le contexte de $D(\beta|w)$, noté $C(\beta, w)$ est défini par :*

$$C(\beta, w) = \{w' \ :\ w' \sim_{\overline{S}(\beta)} w\}$$

Le contexte d'un désir conditionnel $D(\beta|\alpha)$, noté $C(\beta, \alpha)$ est défini par :

$$C(\beta, \alpha) = \bigcup_{w \models \alpha} C(\beta, w)$$

Classement admissible

Un classement admissible π est une fonction de W vers l'ensemble des entiers. Un monde w n'est pas moins préféré à un monde w' ssi $\pi(w) \geq \pi(w')$. Un désir conditionnel $D(\beta|\alpha)$ va induire une relation sur les mondes de la façon suivante : chaque β-monde du contexte $C(\beta, \alpha)$ est préféré à tous les $\neg\beta$-mondes du même contexte.

Définition 4.2.2. *Un classement préférentiel π est admissible par rapport à un désir conditionnel $D(\beta|\alpha)$ si pour tous les mondes w et w' appartenant à $C(\beta|\alpha)$, $w \models \beta$ et $w' \models \neg\alpha$ implique $\pi(w) \geq \pi(w')$.*

Un classement préférentiel π est admissible par rapport à un ensemble de désirs conditionnels D ssi il est admissible par rapport à tous les désirs conditionnels de D.

Exemple. *Supposons que M. X exprime les désirs conditionnels suivants :*

- *« je préfére que ma maison ne soit pas proche d'une station de métro », exprimé par la formule $D(\neg M)$;*
- *« si ma maison est loin du centre ville, je préfére qu'elle soit proche d'une station de métro », exprimé par la formule $D(M|\neg C)$.*

Le premier désir va nous permettre de comparer deux à deux les mondes qui ne diffèrent que sur M. Un classement préférentiel admissible pour $D(\neg M)$ π va donc devoir respecter $\pi(\{\neg M, C\}) \geq \pi(\{M, C\})$ et $\pi(\{\neg M, \neg C\}) \geq \pi(\{M, \neg C\})$.

Le second désir donne la contrainte suivante : $\pi(\{M, \neg C\}) \geq \pi(\{\neg M, \neg C\})$. Cette contrainte est contradictoire avec la contrainte $\pi(\{\neg M, \neg C\}) \geq \pi(\{M, \neg C\})$ imposé par le premier désir. À moins de considérer que le second désir comme plus spécifique que le premier, on ne peut pas trouver de classement préférentiel admissible pour $\{D(\neg M), D(M|\neg C)\}$. Tan et Pearl notent d'ailleurs dans [129] que si α et α' sont deux formules propositionnelles, $\alpha \rightarrow \alpha'$ implique que $\{D(\beta|\alpha), D(\neg\beta|\alpha')\}$ est incohérent.

Si l'on considère que le second désir est plus spécifique que le premier, alors un classement préférentiel vérifiant $\pi(\{\neg M, C\}) \geq \pi(\{M, C\})$ et $\pi(\{M, \neg C\}) \geq \pi(\{\neg M, \neg C\})$ est considéré comme admissible pour $\{D(\neg M), D(M|\neg C)\}$. Remarquons tout de même que cette approche est très peu discriminante : si on ajoute une variable propositionnelle B signifiant que les murs de la maison sont blancs, alors les deux mondes $\{M, \neg C, B\}$ et $\{\neg M, \neg C, \neg B\}$ sont incomparables. Or, il paraît intuitivement correct de pouvoir dériver que $\pi(\{M, \neg C, B\}) \geq \pi(\{\neg M, \neg C, \neg B\})$, car si la maison n'est pas proche du centre ville, M. X préfére qu'elle soit proche d'une bouche de métro et le fait que les murs soient blancs n'intervient pas dans cette préférence.

Il ne doit pas y avoir de préférence entre deux mondes si ce n'est pas contraint par les préférences exprimées par l'agent. Tan et Pearl adoptent donc le principe d'*indifférence maximale* qui permet de distinguer parmi tous les classements préférentiels admissibles par rapport à un ensemble D un classement qui minimise les différences dans la relation de préférence.

Définition 4.2.3. *Soit D un ensemble cohérent de désirs conditionnels et Π l'ensemble des classements admissibles par rapport à cet ensemble. Un classement $\pi^+ \in \Pi$ admissible par rapport à D est parmi les* plus compacts *si pour tout $\pi \in \Pi$:*

$$\sum_{(w,w') \in W^2} |\pi^+(w) - \pi^+(w')| \leq \sum_{(w,w') \in W^2} |\pi(w) - \pi(w')|$$

Tan et Pearl définissent ensuite une relation binaire entre ensembles de monde nommée *dominance préférentielle*. Cette relation leur permet d'évaluer des requêtes comme « étant donné que la maison est proche du centre ville, est ce que je préfére qu'elle soit proche d'une station de métro ? ». Enfin, ils raffinent la notion de désir conditionnel en ajoutant un degré ϵ à chaque désir. Sémantiquement, ce degré n'intervient que dans la définition d'un classement admissible : pour un désir $D_\epsilon(\beta|\alpha)$, la condition sur deux mondes w et w' appartenant à $C(\beta, \alpha)$ tels que $w \models \alpha$ et $w' \models \neg\alpha$ devient $\pi(w) \geq \pi(w') + \epsilon$. Nous ne présenterons pas ces travaux ici.

Conclusion

Le formalisme développé par Tan et Pearl permet à partir d'un ensemble de désirs conditionnels de construire un préordre partiel ou total sur les mondes possibles et possède comme avantage de pouvoir exprimer des désirs dans lesquels la condition est contradictoire avec le désir. $D(lampe|\neg lampe)$ qui signifie que si la lampe est eteinte, on préfére qu'elle soit allumée est une formule qui admet un classement préférentiel admissible. Comme nous l'avons déjà précisé, la condition ne spécifie pas les mondes dans lesquels la contrainte du désir s'applique, mais les mondes dans lesquels on souhaite que le désir soit satisfait. Cependant, il convient de noter que ce formalisme n'est pas très discriminant. L'ajout par exemple d'une variable propositionnelle qui n'intervient pas dans un ensemble de désir conduit à rendre des mondes incomparables, alors qu'intuitivement il existe un monde préféré parmi ceux-ci.

4.2.2 Approche *ceteris paribus* : formalisme de Boutilier et al.

Boutilier et al. proposent dans [14] une représentation graphique des préférences conditionnelles *ceteris paribus*. Ils considèrent un ensemble de variables $F = \{F_1, \ldots, F_n\}$ qui peuvent prendre chacune leurs valeurs dans un domaine quelconque. Nous considérerons que pour chaque $F_i \in F$, le domaine de valeur qui lui est associé est $\{F_i, \neg F_i\}$. $\mathcal{F} = F_1 \times \ldots \times F_n$ est l'ensemble des affectations possibles. Une affectation des variables de $X \subseteq F$ est notée x et la concaténation des affectations des variables de deux sous-ensembles X et Y de F est notée xy (si $X \cap Y = \phi$). Si $X \cup Y = F$, alors xy est une affectation complète des variables de F. Un *classement préférentiel* sur $\mathcal{F}$ est noté $\succeq$. $f_1 \succeq f_2$ signifie donc que f_1 est au moins autant préféré que f_2.

Définition 4.2.4. *Un ensemble de variables $X \subseteq F$ est dit* préférentiellement indépendant *de son complémentaire $Y = F - X$ ssi* $\forall x_1 \forall x_2 \forall y_1 \forall y_2 \quad (x_1y_1 \succeq x_2y_1) \Leftrightarrow (x_1y_2 \succeq x_2y_2)$.

Cette condition d'indépendance préférentielle est la traduction de la condition *ceteris paribus* : la structure de la relation de préférence entre x_1 et x_2 est la même quelle que soit l'affectation attribuée aux variables n'appartenant pas à X.

Définition 4.2.5. *Soient X, Y et Z trois sous-ensembles non vides de F qui forment une partition de F. X et Y sont* conditionnellement préférentiellement indépendants *étant donné z ssi* $\forall x_1 \forall x_2 \forall y_1 \forall y_2 \quad (x_1y_1z \succeq x_2y_1z) \Leftrightarrow (x_1y_2z \succeq x_2y_2z)$.

L'indépendance préférentielle entre X et Y ne tient que lorsque Z est assigné à z. Si cette relation est vraie pour tout assignement z de Z, alors on dit que X et Y sont conditionnellement préférentiellement indépendants étant donné Z.

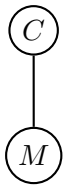
La représentation d'une relation de préférence va être effectuée de façon compacte grâce à un graphe dont chaque nœud est un attribut. Pour chaque nœud F_i du graphe, l'utilisateur doit identifier un ensemble de nœuds parents noté $P(F_i)$ qui peut affecter l'ordre de préférence sur les valeurs possibles de F_i. Étant donné un assignement particulier aux variables de $P(F_i)$, l'utilisateur doit pouvoir déterminer un ordre de préférence sur les assignations possibles de F_i, toutes autres choses étant égales. Formellement, F_i et $(F - (\{F_i\} \cup P(F_i))$ sont conditionnellement préférentiellement indépendants étant donné $P(F_i)$. L'utilisateur spécifie alors ses préférences concernant F_i pour chaque assignation possible de $P(F_i)$.

Boutilier et al. construisent alors un graphe où chaque nœud F_i a les éléments de $P(F_i)$ comme parents. Chaque nœud F_i est également étiqueté par une table de préférence sur les assignations possibles de F_i pour toutes les valeurs possibles des noeuds parents. De telles structures sont appellées réseaux de préférences conditionnelles (*CP-networks*).

Exemple. *Supposons que M. X annonce que le fait que sa maison soit proche d'une station de métro (noté par M) ne dépende que de sa proximité du centre ville (noté par C) et ceci toutes autres choses par ailleurs étant égales. C est donc l'unique parent de M. Supposons de plus que M. X exprime ses préférences de la façon suivante :*

- *il préfére que la maison soit proche du centre ville ;*
- *si la maison n'est pas proche du centre ville, il préfére qu'elle soit proche d'une station de métro ;*
- *si la maison est proche du centre ville, il préfére qu'elle ne soit pas proche d'une station de métro.*

Dans ce cas, on obtient un CP-network *représenté en figure 4.1. On peut en déduire l'ordre suivant sur les valuations de $F = \{M, C\}$: $\{C, \neg M\} \succeq \{C, M\} \succeq \{\neg C, M\} \succeq \{\neg C, \neg M\}$.*



$$C \succeq \neg C$$
$$C \ : \ \neg M \succeq M$$
$$\neg C \ : \ M \succeq \neg M$$

FIG. 4.1 – CP-network de l'exemple 4.2.2

On obtient dans cet exemple un préordre complet sur les mondes possibles (ce n'est pas toujours le cas). On ordonne les assignations possibles en fonction des préférences qu'elles violent. $\{C, \neg M\}$ ne viole aucune préférence, $\{\neg C, M\}$ viole la préférence sur C, $\{C, M\}$ viole la préférence sur M et $\{\neg C, \neg M\}$ viole les deux préférences. Les préférences des nœuds parents ont priorité sur les préférences des nœuds fils (par exemple $\{C, M\} \succeq \{\neg C, M\}$). Il se peut très bien que l'on obtienne non pas un préordre complet, mais simplement un préordre partiel : il existe alors des éléments de $\mathcal{F}$ qui sont incomparables.

L'approche proposée par Boutilier et al. permet de représenter de façon compacte un ensemble de préférences. On peut en déduire un préordre partiel voire complet dans le meilleur des cas sur les mondes possibles. Le principal avantage de cette formalisation est de fournir des algorithmes efficaces pour déterminer ce préordre. En particulier, la question de savoir si deux mondes sont comparables et quel est le monde le plus préféré des deux se ramène à rechercher des chaînes dans le graphe. Ils disposent d'heuristiques efficaces pour cette recherche. Cependant, l'utilisateur est obligé pour chaque nœud du graphe de détailler quelles sont ses préférences en tenant compte de toutes les valuations possibles des variables contenues dans les parents du nœud.

4.2.3 Approche en terme d'idéalité : formalisme de Boutilier

Boutilier a défini dans [12] une logique bimodale appelée CO permettant de raisonner avec des défauts. Il utilise cette logique dans [13] pour représenter et raisonner avec des préférences, en particulier des préférences conditionnelles. Cette logique s'inspire de la logique d'Humberstone développée dans [71]. Nous allons présenter la logique CO dans ce qui suit.

Langage de CO

Boutilier considère un langage propositionnel bimodal L_B constitué d'un langage propositionnel classique $PROP$ muni des connecteurs $\neg$, $\vee$, $\wedge$ et $\rightarrow$ et de deux opérateurs modaux $\Box$ et $\overleftarrow{\Box}$. Les formules bien formées de L_B sont définies de la façon suivante :

- si φ est une formule bien formée de $PROP$ alors φ est une formule bien formée de L_B ;
- si φ est une formule bien formée de L_B, alors $\Box\varphi$ est une formule bien formée de L_B ;
- si φ est une formule bien formée de L_B, alors $\overleftarrow{\Box}\ \varphi$ est une formule bien formée de L_B ;
- si φ est une formule bien formée de L_B, alors $\neg\varphi$ est une formule bien formée de L_B ;
- si φ et ψ sont deux formules bien formées de L_B, alors $\varphi \vee \psi$ est une formule bien formée de L_B.

Les connecteurs $\wedge$, *to* et $\leftrightarrow$ et les opérateurs $\Diamond$, $\overleftarrow{\Diamond}$, $\overleftrightarrow{\Box}$ et $\overleftrightarrow{\Diamond}$ sont définis pour deux formules bien formées de L_B φ et ψ par :

- $\varphi \wedge \psi \equiv_{def} \neg(\neg\varphi \vee \neg\psi)$;
- $\varphi \rightarrow \psi \equiv_{def} \neg\varphi \vee \psi$;
- $\varphi \leftrightarrow \psi \equiv_{def} (\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)$;
- $\Diamond\varphi \equiv_{def} \neg\Box\neg\varphi$;
- $\overleftarrow{\Diamond}\ \varphi \equiv_{def} \neg\ \overleftarrow{\Box}\ \neg\varphi$;
- $\overleftrightarrow{\Box}\ \varphi \equiv_{def} \Box\varphi \wedge \overleftarrow{\Box}\ \varphi$;
- $\overleftrightarrow{\Diamond}\ \varphi \equiv_{def} \neg\ \overleftrightarrow{\Box}\ \neg\varphi$.

Remarquons également que le fragment de CO qui ne contient que des formules du type $I(\beta|\alpha)$ est exactement le système DSDL3 ou système Hansson-Lewis-Spohn (cf. [13]).

Sémantique de *CO*

La sémantique de *CO* est caractérisée par des modèles de Kripke $\langle W, \leq, val\rangle$ où :
- W est un ensemble de mondes possibles ;
- $\leq$ est un préordre total sur W : c'est donc une relation sur W^2 qui est réflexive et transitive. Si w et w' sont deux mondes de W, alors $w \leq w'$ signifie que w est au moins autant préféré que w' ;
- val est une fonction de valuation sur W^2. Pour une formule de L_B, $val(\varphi)$ est donc l'ensemble des mondes de W qui satisfont φ. On notera classiquement cet ensemble $||\varphi||$ (cf. [25]).

Définition 4.2.6. *Soit $\mathcal{M} = \langle W, \leq, val\rangle$ un CO-modèle et $w \in W$. La valeur de vérité d'une formule φ de L_B dans w par rapport à $\mathcal{M}$ est définie par :*
- $\mathcal{M} \models_w \varphi$ *ssi* $w \in val(\varphi)$ *pour φ formule bien formée de PROP ;*
- $\mathcal{M} \models_w \neg\varphi$ *ssi* $\mathcal{M} \not\models_w \varphi$ *;*
- $\mathcal{M} \models_w \varphi \vee \psi$ *ssi* $\mathcal{M} \models_w \varphi$ *ou* $\mathcal{M} \models_w \psi$ *;*
- $\mathcal{M} \models_w \Box\varphi$ *ssi* $\forall w' \in W$ *tel que* $w' \leq w$ *alors* $\mathcal{M} \models_{w'} \varphi$ *;*
- $\mathcal{M} \models_w \overleftarrow{\Box}\varphi$ *ssi* $\forall w' \in W$ *tel que* $w' \not\leq w$ *alors* $\mathcal{M} \models_{w'} \varphi$.

La satisfaisabilité des autres opérateurs modaux est facilement déductible des conditions de satisfaisabilité précédentes :
- $\mathcal{M} \models_w \Diamond\varphi$ ssi $\exists w' \in W$ tel que $w' \leq w$ et $\mathcal{M} \models_{w'} \varphi$;
- $\mathcal{M} \models_w \overleftarrow{\Diamond}\varphi$ ssi $\exists w' \in W$ tel que $w' \not\leq w$ et $\mathcal{M} \models_{w'} \varphi$;
- $\mathcal{M} \models_w \overleftrightarrow{\Box}\varphi$ ssi $\forall w' \in W$ $\mathcal{M} \models_{w'} \varphi$;
- $\mathcal{M} \models_w \overleftrightarrow{\Diamond}\varphi$ ssi $\exists w' \in W$ tel que $\mathcal{M} \models_{w'} \varphi$.

$\Box\varphi$ est vraie dans w signifie « φ est vraie dans tous les mondes au moins autant préférés que w ». $\Diamond\varphi$ est vraie dans w signifie « φ est vraie dans au moins un monde au moins autant préféré que w ». $\overleftarrow{\Box}\varphi$ est vraie dans w ssi φ est vraie dans tous les mondes moins préférés que w et $\overleftarrow{\Diamond}\varphi$ est vraie dans w ssi φ est vraie dans au moins un monde moins préféré que w. $\overleftrightarrow{\Box}$ et $\overleftrightarrow{\Diamond}$ correspondent respectivement aux opérateurs modaux classiques de nécessité et de possibilité (cf. [25]).

La validité d'une formule est définie de la façon suivante :

Définition 4.2.7. *Soit $\mathcal{M} = \langle W, \leq, val\rangle$ un CO-modèle. Une formule φ est valide dans $\mathcal{M}$ (noté $\mathcal{M} \models \varphi$) ssi $\forall w \in W$ $\mathcal{M} \models_w \varphi$. φ est CO-valide (noté $\models_{CO} \varphi$) ssi pour tout CO-modèle $\mathcal{M}$, $\mathcal{M} \models \varphi$. Enfin, φ est satisfaisable ssi $\neg\varphi$ n'est pas valide.*

Par exemple, la figure 4.2 montre un *CO*-modèle $\mathcal{M}$ tel que $\mathcal{M} \models \overleftrightarrow{\Box}\alpha$ (car tous les mondes satisfont α) et $\mathcal{M} \models_{w_2} \Box\beta$ (car tous les mondes au moins autant préférés que w_2 satisfont β).

Enfin, la notion de conséquence est définie par :

Définition 4.2.8. *Soit $\Sigma = \{\varphi_1, \ldots, \varphi_n\}$ un ensemble de formules de CO. Une formule ψ est une conséquence de Σ, noté $\Sigma \models \psi$ ssi pour tout CO-modèle $\mathcal{M}$, $\mathcal{M} \models \bigwedge_{i\in\{1,\ldots,n\}} \varphi_i \Rightarrow \mathcal{M} \models \psi$.*

[2] I.e. $val : PROP \to 2^{\mathcal{W}}$ et val vérifie $val(\neg\varphi) = \mathcal{W} - val(\varphi)$ et $val(\varphi_1 \wedge \varphi_2) = val(\varphi_1) \cap val(\varphi_2)$.

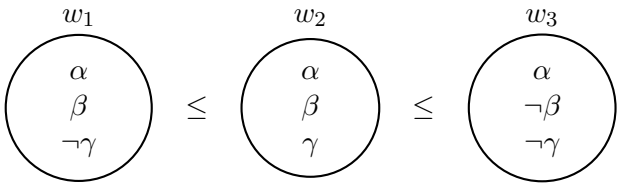


FIG. 4.2 – Un CO-modèle particulier

Axiomatique de CO

La logique CO est le plus petit ensemble $S \subseteq L_B$ tel que S contienne les schémas d'axiomes suivants et soit fermée par les règles d'inférence suivantes :

K $\Box(\alpha \rightarrow \beta) \rightarrow (\Box\alpha \rightarrow \Box\beta)$
K' $\overleftarrow{\Box}\,(\alpha \rightarrow \beta) \rightarrow (\overleftarrow{\Box}\,\alpha \rightarrow \overleftarrow{\Box}\,\beta)$
T $\Box\alpha \rightarrow \alpha$
4 $\Box\alpha \rightarrow \Box\Box\alpha$
S $\alpha \rightarrow \overleftarrow{\Box}\,\Diamond\alpha$
H $\overleftrightarrow{\Diamond}\,(\Box\alpha \wedge \overleftarrow{\Box}\,\beta) \rightarrow \overleftrightarrow{\Box}\,(\alpha \vee \beta)$

Nec $\dfrac{\vdash_{CO}\alpha}{\vdash_{CO}\overleftrightarrow{\Box}\alpha}$

MP $\dfrac{\alpha \rightarrow \beta \quad \alpha}{\beta}$

Théorème 4.2.1. $\forall\alpha \in L_B \quad \vdash_{CO} \alpha$ *ssi* $\models_{CO} \alpha$.

La preuve est donnée dans [12]. Remarquons par contre que nous ne disposons pas de stratégie intéressante de recherche de preuve pour cette logique.

Dans les modèles de CO, l'ensemble de mondes W ne contient pas forcement tous les mondes logiquement possibles étant donné le langage propositionnel $PROP$. Boutilier définit également une logique CO^* qui est la plus petite extension de CO contenant l'axiome suivant :

LP $\overleftrightarrow{\Diamond}\,\alpha$ pour toute formule propositionnelle α satisfaisable

Ainsi, dans CO^* on considère tous les mondes logiquement possible. En particulier, pour tout CO^*-modèle $\langle W, \leq, val\rangle$, $W = \prod_{a \in PROP} \{a, \neg a\}$.

Expression de préférences conditionnelles dans CO

Boutilier a choisi de représenter les préférences conditionnelles en utilisant une approche « idéale » : les préférences seront des formules du type $I(\beta|\alpha)$ qui signifient que « idéalement si α est vraie, alors β est vraie ». Formellement, l'opérateur I est défini par :

$$I(\beta|\alpha) \equiv_{def} \overleftrightarrow{\Box}\,\neg\alpha \vee \overleftrightarrow{\Diamond}\,(\alpha \wedge \Box(\alpha \rightarrow \beta))$$

Ainsi, si l'on considère un CO-modèle $\mathcal{M}$, $I(\beta|\alpha)$ sera valide dans $\mathcal{M}$ ssi :
- soit α n'est vraie dans aucun monde de W ;
- soit il existe un monde w qui satisfait α et tel que tous les mondes au moins autant préférés à W satisfont $\alpha \rightarrow \beta$.

Cette définition traduit bien la signification intuitive en terme d'idéalité donnée à la formule $I(\beta|\alpha)$. $I(\beta)$ sera un raccourci d'écriture pour $I(\beta|\top)$. De même, on notera la notion duale de *tolérance* par $T(\beta|\alpha) \equiv_{def} \neg I(\neg\beta|\alpha)$.

Boutilier définit également un opérateur de préférence $\leq_P$ entre deux propositions par :

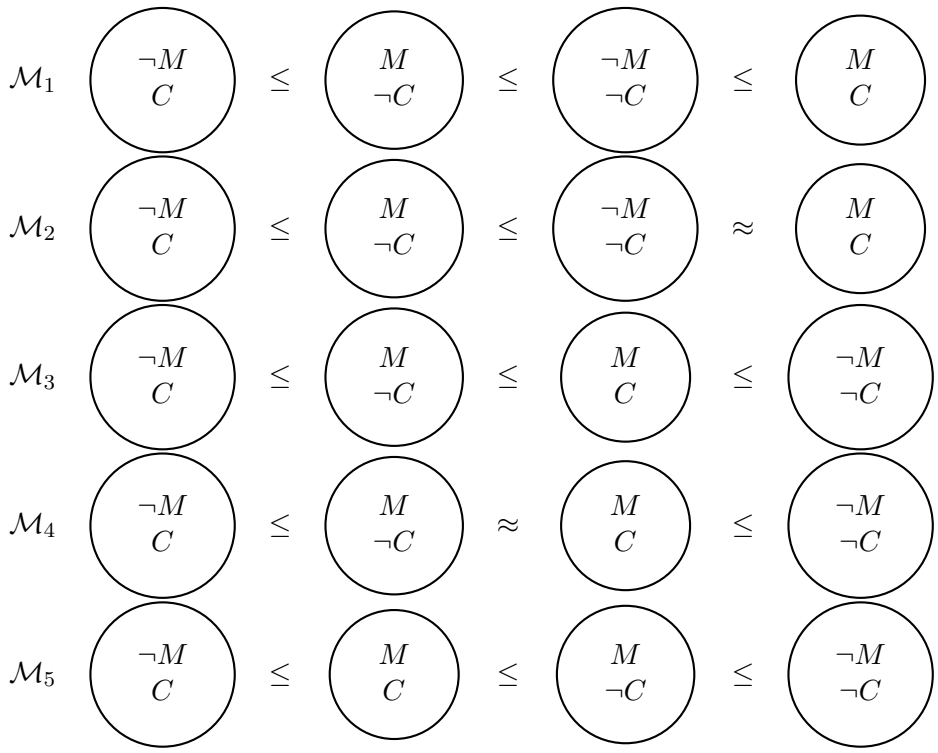
$$\alpha \leq_P \beta \equiv_{def} \overleftrightarrow{\Box}(\beta \rightarrow \Diamond\alpha)$$

Intuitivement, $\alpha \leq_P \beta$ signifie que les meilleurs des α-mondes sont au moins autant préférés que les meilleurs des β-mondes.

Exemple. *Reprenons l'exemple de M. X. Les deux préférences conditionnelles sont ici $I(\neg M)$ et $I(M|\neg C)$. Supposons dans un premier temps que le langage $PROP$ ne contienne que les deux variables M et C. Étant donné un CO^*-modèle $\mathcal{M} = \langle W, \leq, val\rangle$, quelles sont les conditions que doit vérifier $\leq$ pour que $\mathcal{M} \models I(\neg M) \wedge I(M|\neg C)$?*

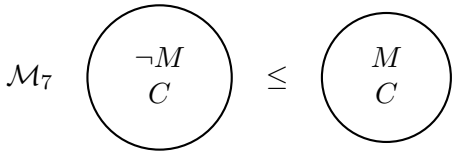
La préférence $I(\neg M)$ se traduit par $\overleftrightarrow{\Diamond}\Box\neg M$. Cela signifie qu'il existe un monde w tel que tous les mondes au moins autant préférés que w satisfont $\neg M$. Donc les mondes minimaux pour le préordre $\leq$ devront vérifier $\neg M$.

La préférence $I(M|\neg C)$ se traduit quant à elle par $\overleftrightarrow{\Box} C \vee \overleftrightarrow{\Diamond}(\neg C \wedge (\neg C \rightarrow M))$. Cela signifie qu'il existe un monde w qui vérifie $\neg C$ et tel que tous les mondes au moins autant préférés à w satisfont $\neg C \rightarrow M$ (on travaille avec des CO^* modèles, donc $\neg C$ est valide dans ces modèles). Donc $\{M, \neg C\} \leq \{\neg M, \neg C\}$. Les CO^*-modèles dans lesquels $I(\neg M) \wedge I(M|\neg C)$ est valide sont au nombre de cinq[3] *:*



Remarquons que si l'on avait travaillé avec des CO-modèles, il se pouvait très bien qu'il n'existe pas de monde dans W qui satisfasse $\neg C$. En particulier, le modèle suivant est un CO-modèle satisfaisant $I(\neg M) \wedge I(M|\neg C)$:

[3] $w \approx w'$ signifie que $w \leq w'$ et $w' \leq w$.



L'utilisation de CO au lieu de CO permet par exemple de prendre en compte des contraintes physiques qui interdisent l'existence de certains mondes.*

Plusieurs remarques sont à faire sur cet exemple : il existe des mondes qui sont incomparables, comme par exemple $\{M, C\}$ et $\{M, \neg C\}$. Bien sûr, comme $\leq$ est un préordre total, pour chaque CO^*-modèle $\mathcal{M}$ satisfaisant $I(\neg M) \wedge I(M|\neg C)$ on aura soit $\{M, C\} \leq \{M, \neg C\}$ soit $\{M, \neg C\} \leq \{M, C\}$, mais aucune contrainte n'est imposée sur les CO^*-modèles de $I(\neg M) \wedge I(M|\neg C)$ concernant ces deux mondes. En particulier, $\{I(\neg M), I(M|\neg C)\} \not\models_{CO^*} (M \wedge C) \leq_P (M \wedge \neg C)$ et $\{I(\neg M), I(M|\neg C)\} \not\models_{CO^*} (M \wedge \neg C) \leq_P (M \wedge C)$.

De plus, contrairement à l'approche de Tan et Pearl, l'ajout d'une nouvelle variable propositionnelle introduit de nouvelles contraintes sur la relation $\leq$. Si on reprend l'exemple précédent, l'ajout de B (les murs sont blancs) dans l'alphabet propositionnel dont on dispose va induire les contraintes suivantes pour tout CO^*-modèle de $I(\neg M) \wedge I(M|\neg C)$: $\{M, \neg C, B\} \leq \{\neg M, \neg C, B\}$, $\{M, \neg C, B\} \leq \{\neg M, \neg C, \neg B\}$, $\{M, \neg C, \neg B\} \leq \{\neg M, \neg C, B\}$ et $\{M, \neg C, \neg B\} \leq \{\neg M, \neg C, \neg B\}$. La relation de préférence entre mondes $\leq$ est donc plus discriminante que celle proposée par Tan et Pearl dans le cadre de [129].

Enfin, pour un ensemble de formules de CO^*, il existe souvent de nombreux modèles (car de nombreux mondes sont incomparables). En particulier, on obtient très rarement un préordre complet sur les mondes. On peut signaler qu'il existe des formalismes, notamment System Z (cf. [107]), qui permettent de restreindre les préordres possibles en formant des « clusters » de formules indifférentes. Nous ne présenterons pas ces travaux ici.

4.2.4 Approche en terme d'idéalité : approche de Lang

Dans [85], Lang étend les travaux de Boutilier [13] en s'inspirant de la logique des pénalités. Il considère qu'un désir conditionnel $D(\beta|\alpha)$ est toujours interprété en terme de situation idéale (β est vraie dans tous les α-mondes les plus préférés), mais ajoute en plus que la violation de $D(\beta|\alpha)$ doit s'accompagner d'une perte d'utilité (au sens de la théorie de la décision, cf. [137]). On considère un ensemble de désirs conditionnels noté $DS = \{D(\beta_1|\alpha_1), \ldots, D(\beta_n|\alpha_n)\}$ que Lang appelle une *spécification de désirs*.

Définition 4.2.9. *Une fonction d'utilité u est une fonction de W dans $\mathbb{R}$. u induit un préordre $\geq_u$ sur W défini par $w \geq_u w'$ ssi $u(w) \geq u(w')$. $\max_{\geq_u}(W)$ représente les mondes les plus préférés de W par rapport à u.*

Définition 4.2.10. *Soit $DS = \{D(\beta_1|\alpha_1), \ldots, D(\beta_n|\alpha_n)\}$ une spécification de désirs et u une fonction d'utilité.*

$u \models DS$ ssi $\forall i \in \{1, \ldots, n\}$ $\max_{\geq_u}(||\alpha_i||) \subseteq ||\beta_i||$

Les fonctions d'utilité sont calculées à partir des fonctions locales associées à chaque désir conditionnel de DS.

Définition 4.2.11. *Soit $D(\beta|\alpha)$ un désir conditionnel, $u_{\beta|\alpha}$ est une* fonction d'utilité locale *associée à $D(\beta|\alpha)$ ssi il existe $x > 0$ tel que*

$$u_{\beta|\alpha}(w) = \begin{cases} -x & \textit{si } w \models \alpha \wedge \neg\beta \\ 0 & \textit{sinon} \end{cases}$$

Définition 4.2.12. *Une fonction d'utilité u est* distinguée *par rapport à $DS = \{D(\beta_1|\alpha_1), \ldots, D(\beta_n|\alpha_n)\}$ ssi* $\begin{array}{ll} (1) & u \models DS \\ (2) & u = u_1 + \ldots + u_n \end{array}$

où $u_1, \ldots, u_n$ sont des fonctions d'utilité locale associée respectivement à $D(\beta_1|\alpha_1), \ldots, D(\beta_n|\alpha_n)$

Les fonctions d'utilité distinguées sont toujours négatives. Si un monde w satisfait l'ensemble des désirs conditionnels contenu dans DS, alors pour toute fonction d'utilité u distinguée par rapport à DS, on a $u(w) = 0$.

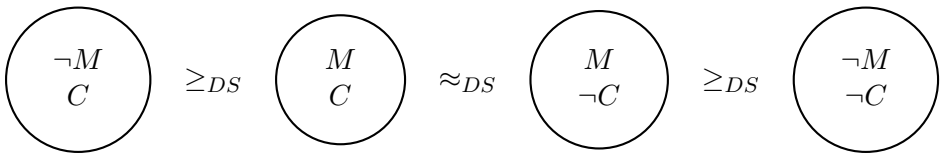
Exemple. *Reprenons l'exemple de M. X. Les deux désirs conditionnels sont $D(\neg M|\top)$ et $D(M|\neg C)$. Les fonctions d'utilité locales associées aux deux désirs conditionnels sont telles que :*

$$u_{\neg M|\top}(w) = \begin{cases} -x & \textit{ssi } w \models M \\ 0 & \textit{sinon} \end{cases}$$

$$u_{M|\neg C}(w) = \begin{cases} -y & \textit{ssi } w \models \neg M \wedge \neg C \\ 0 & \textit{sinon} \end{cases}$$

Une fonction d'utilité u distinguée par rapport à $\{D(\neg M|\top), D(M|\neg C)\}$ associée à ces fonctions d'utilité locales verifie donc : $u(\{M, \neg C\}) = u(\{M, C\}) = -x$, $u(\{\neg M, \neg C\}) = -y$ et $u(\{\neg M, C\}) = 0$.

De plus, $u \models D(\neg M|\top)$ ssi $\max(u(\{\neg M, \neg C\}), u(\{\neg M, C\})) > \max(u(\{M, \neg C\}), u(\{M, C\}))$, donc ssi $max(-y, 0) > max(-x, -x)$, ce qui conduit à $x > 0$. $u \models D(M|\neg C)$ ssi $u(\{M, \neg C\}) > u(\{\neg M, \neg C\})$ donc ssi $-x > -y$. On obtient $y > x > 0$. La fonction d'utilité u induit un préordre $\geq_{DS}$ sur les mondes : $w \geq_{DS} w'$ ssi $u(w) \geq u(w')$. On obtient ici :



$w \approx_{DS} w'$ signifie que $w \geq_{DS} w'$ et $w' \geq_{DS} w$.

Plusieurs remarques sont à faire sur cet exemple. Tout d'abord, on voit que les mondes $\{M, C\}$ et $\{M, \neg C\}$ sont indifférents, ce qui est intuitivement correct. Rien dans la spécification de désirs DS ne nous permet de « classer » ces deux mondes. Par contre, $\{C, M\}$ est toujours préféré à $\{\neg M, \neg C\}$, ce qui nous paraît intuitivement incorrect. En effet, rien dans DS ne nous permet d'affirmer que $\{C, M\}$ est un meilleur monde que $\{\neg M, \neg C\}$. En particulier, $\{\neg M, \neg C\}$ vérifie le premier désir $D(\neg M)$. C'est le problème des approches utilisant directement la logique des pénalités : la non satisfaction d'un désir par un monde conduit à une baisse de l'utilité associée à ce monde , alors que la satisfaction d'une partie des désirs par ce même monde ne conduit pas à une « augmentation » de l'utilité.

Remarquons de plus que le fait d'ajouter un désir conditionnel $D(\neg M|C)$ à $DS = \{D(\neg M|\top), D(M|\neg C)\}$ conduit à introduire une nouvelle fonction d'utilité locale :

$$u_{\neg M|\top}(w) = \begin{cases} -z & \text{ssi } w \models M \wedge C \\ 0 & \text{sinon} \end{cases}$$

et donc l'ensemble des fonctions d'utilité u distinguées par rapport à DS est caractérisé par :

$u(\{\neg M, C\}) = 0$
$u(\{\neg M, \neg C\}) = -y$
$u(\{M, \neg C\}) = -x$
$u(\{M, C\}) = -x - z$

Cette fois ci, les contraintes imposées sur la fonction d'utilité sont les suivantes : $\max(u(\{\neg M, \neg C\}, \{\neg M, C\}) > \max(u(\{M, \neg C\}, \{M, C\})$ donc $x > 0$, $u(\{M, \neg C\}) > u(\{\neg M, \neg C\})$ donc $y > x$ et enfin $u(\{\neg M, C\}) > u(\{M, C\})$ donc $x + z > 0$. Donc $\{M, \neg C\} \geq_{DS} \{M, C\}$, alors que l'ajout de $D(\neg M|C)$ à DS n'ajoutait intuitivement rien, puisque l'on avait déjà $D(\neg M|\top)$.

Enfin, si on considère une spécification de désirs $\{D(M|\neg C)\}$, le seul monde pénalisé est $\{\neg M, \neg C\}$ donc tous les autres mondes sont préférés à $\{\neg M, \neg C\}$ alors qu'intuitivement, la spécification d'un tel désir ne devrait permettre que d'écrire que $\{M, \neg C\} \geq_{DS} \{\neg M, \neg C\}$. La encore, la notion de pénalité ne semble pas appropriée pour représenter ce genre de préférence, car le fait de pénaliser un monde favorise tous les autres.

Nous considérons donc que l'utilisation de la logique des pénalités seule ne permet pas de représenter correctement des préférences conditionnelles. L'ordre obtenu sur les mondes est souvent trop restrictif dans le sens où on ajoute des contraintes sur les mondes alors qu'elles n'ont pas lieu d'être (comme par exemple $\{C, M\} \geq_{DS} \{\neg M, \neg C\}$). De plus, l'ajout d'une préférence conditionnelle « redondante » à une spécification de désirs rajoute des contraintes contre-intuitives sur les mondes. Ceci est principalement dû à la signification de la pénalité : si un monde ne respecte pas un désir conditionnel, alors il est pénalisé, mais s'il en respecte un autre, il n'y a pas d'augmentation de l'utilité associée à ce monde. Nous allons voir dans la section suivante que l'utilisation de polarités permet de résoudre ces problèmes.

4.2.5 Approche en terme d'idéalité : formalisme de van der Torre et Weydert

Comme nous l'avons vu dans la section précédente, l'utilisation de pénalités seules ne permet pas de modéliser correctement des préférences. Les pénalités ne caractérisent que les mondes qui violent un certain désir conditionnel. Or quand on considère un désir conditionnel $D(\beta|\alpha)$ et un monde w, il existe trois cas :

1. soit $w \models \alpha \wedge \beta$ et alors $D(\beta|\alpha)$ est satisfait ;
2. soit $w \models \alpha \wedge \neg\beta$ et alors le désir $D(\beta|\alpha)$ est violé ;
3. soit $w \models \neg\alpha$ et alors le désir $D(\beta|\alpha)$ n'est pas applicable.

L'interprétation des désirs en termes négatifs comme dans la section précédente ne permet pas de distinguer les situations 1 et 3. Le mondes qui satisfont un désir où dans lesquels le désir est innapplicable ont une utilité (locale par rapport au désir) nulle. De la même façon, on pourrait considérer les désirs de façon positive, c'est-à-dire qu'un monde qui satisfait un désir conditionnel aura une utilité associée à ce désir strictement positive. Mais dans ce cas, on ne peut pas distinguer les situations 2 et 3.

Pour pallier à ce problème, van der Torre et Weydert [136] (voir aussi [86]) étendent l'approche de Lang en considérant qu'à chaque désir sont associées une recompense et une pénalité. On peut donc maintenant distinguer les trois cas présentés précédemment. Par rapport à l'approche de Lang, ils introduisent deux notions supplémentaires :

– la *force* d'un désir conditionnel qui représente l'importance du désir à laquelle elle est associée. La force est représentée par un poids associé au désir ;

– la *polarité* d'un désir conditionnel qui représente le rapport entre la perte d'utilité engendrée par la non satisfaction du désir conditionnel et le gain d'utilité associé à la satisfaction du désir conditionnel.

Un désir conditionnel est maintenant représenté sous la forme $D^p_s(\beta|\alpha)$ où $p \in [0,1]$ est la polarité associée au désir et $s \in \{\geq r, > r | r \in [0, +\infty]\}$ la force associée au désir (par exemple, la force associée à un désir peut être > 2 ou $\geq +\infty$). Si l'on considère un désir conditionnel $D^p_s(\beta|\alpha)$, alors la fonction d'utilité locale associée à ce désir est définie par :

$$u_{\beta|\alpha}(w) = \begin{cases} x & \text{si } w \models \alpha \wedge \beta \\ -y & \text{si } w \models \alpha \wedge \neg\beta \\ 0 & \text{sinon} \end{cases}$$

La force associée au désir nous indique la différence minimale entre l'utilité associée à un monde qui respecte le désir et un monde qui ne le respecte pas. Nous considérerons dans la suite de la présentation que la force associée à chaque désir sera nulle et nous l'omettrons dans l'écriture d'un désir (on peut remarquer que la force est > 0 et non pas ≥ 0).

La polarité associée à $D^p(\beta|\alpha)$ est définie par $p = \frac{y}{x+y}$ si $x + y > 0$. Cinq cas peuvent donc se présenter :

– $p = 0$ et on ne considère que des désir positifs car $y = 0$;
– $p = 1$ et on ne considère que des désirs négatifs car $x = 0$. C'est le cas du formalisme de Lang présenté dans la section précédente ;
– $p = \frac{1}{2}$ et alors la violation du désir a autant d'importance que sa satisfaction ;
– $p \in]0, \frac{1}{2}[$ et alors on accorde plus d'importance à la satisfaction du désir qu'à sa violation. On peut alors le voir comme un but (la récompense est plus importante que la pénalité) ;
– $p \in]\frac{1}{2}, 1[$ et alors on accorde plus d'importance à la violation du désir qu'à sa satisfaction. On peut alors le voir comme une contrainte.

Les définitions des fonctions d'utilité locales et des fonctions d'utilité distinguées par rapport à une spécification de désirs sont les mêmes que celles données dans [85] (car nous ne considérons pas la force des désirs), nous ne les rappelerons donc pas.

Exemple. *Reprenons l'exemple précédent : on considére une spécification de désirs* $DS = \{D^{p_1}(\neg M|\top), D^{p_2}(M|\neg C)\}$. *Les fonctions d'utilités locales associées aux deux désirs conditionnels sont donc :*

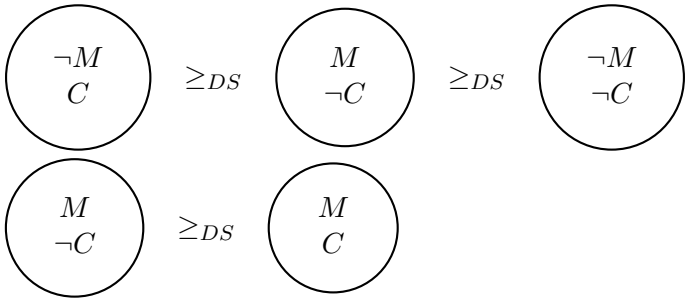
$$u_{\neg M|\top}(w) = \begin{cases} x_1 & \textit{si } w \models \neg M \\ -y_1 & \textit{si } w \models M \\ 0 & \textit{sinon} \end{cases}$$

$$u_{M|\neg C}(w) = \begin{cases} x_2 & \textit{si } w \models \neg C \wedge M \\ -y_2 & \textit{si } w \models \neg C \wedge \neg M \\ 0 & \textit{sinon} \end{cases}$$

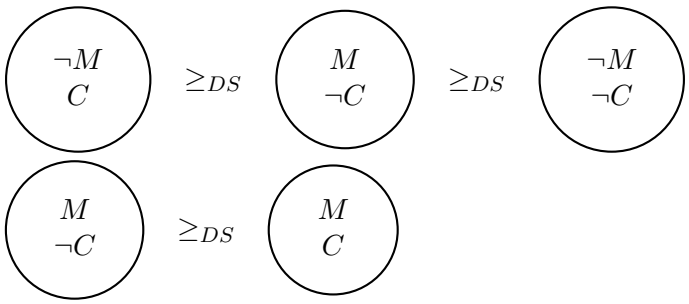
Une fonction d'utilité distinguée par rapport à $DS = \{D^{p_1}(\neg M|\top), D^{p_2}(M|\neg C)\}$ *vérifie donc* $u(\{M, C\}) = -y_1$, $u(\{M, \neg C\}) = x_2 - y_1$, $u(\{\neg M, C\}) = x_1$ *et* $u(\{\neg M, \neg C\}) = x_1 - y_2$.

Les contraintes d'idéalité impliquent de plus que $\max(u(\{\neg M, C\}), u(\{\neg M, \neg C\}) > \max(u(\{M, C\}), u(\{M, \neg C\}))$, *donc* $x_1 > x_2 - y_1$. *Enfin* $u(\{M, \neg C\}) > u(\{\neg M, \neg C\})$, *donc* $x_2 - y_1 > x_1 - y_2$.

Le préordre sur les mondes associé à u *devient donc :*



Si les désirs sont purement négatifs, i.e. $p_1 = p_2 = 1$*, on obtient le même résultat qu'avec l'approche de Lang. Si les polarités associées au désirs ne sont pas extrêmes,* x_1*,* x_2*,* y_1 *et* y_2 *sont tous strictement positifs et le préordre devient alors :*



Cette fois ci, on obtient un préordre sur les mondes qui paraît moins intuitivement incorrect. En particulier, $\{M, C\}$ n'est pas obligatoirement préféré à $\{\neg M, \neg C\}$.

L'utilisation des polarités ajoute un pouvoir expressif important au formalisme développé par Lang. En particulier, on est capable de modéliser précisement des phrases contenant des désirs conditionnels comme (les exemples sont tirés de [86]) :

- des désirs strictement positifs comme par exemple « si j'ai du poisson au diner, je préfére avoir du Bourgogne blanc ». Dans ce cas, seule la satisfaction du désir contraint la fonction d'utilité associée. En particulier, la non satisfaction du désir n'induit pas de perte d'utilité ;
- des désirs strictement négatifs comme par exemple « s'il pleut, je préfére avoir un parapluie ». La violation du désir implique une baisse de l'utilité mais la satisfaction du désir n'implique pas une augmentation de l'utilité. Ce qui est important dans ce genre de désir, c'est d'être pénalisé en cas de violation ;
- des désirs mixtes comme par exemple « si je mange des pommes de terre, je préfére qu'elles soient cuites ». Dans ce cas, l'agent préfére manger une pomme de terre cuite à ne pas manger, et préfére ne pas manger plutôt que de manger une pomme de terre crue (à moins qu'il ne soit affamé).

4.3 Conclusion

Nous venons de voir qu'il existe beaucoup de formalismes permettant de représenter des préférences. Il existe bien sûr d'autres formalismes (cf. par exemple [83]), mais nous ne les présenterons pas ici. Les logiques pondérées, qui se fondent sur l'association entre une formule et un poids, permettent de représenter facilement des bases de connaissance et de calculer l'ordre sur les mondes possibles associé. Mais nous pensons que la pondération des

formules n'est pas forcément évidente : pourquoi associer un poids de 30 à une formule au lieu d'un poids de 40 par exemple ?

Nous avons vu en étudiant les préférences conditionnelles qu'il existe d'autres formalismes intéressants permettant de représenter des préférences. En particulier, les logiques modales CO et CO^* développées par Boutilier ont beaucoup de qualités : compacité de la représentation des préférences, préférences totalement qualitatives et surtout le préordre induit sur les mondes nous semble intuitivement correct. CO et CO^* permettent de garder une certaine souplesse vis-à-vis de la représentation sémantique des préférences. Bien sûr cette souplesse a une contrepartie : on n'obtient pas souvent un préordre complet, mais on peut utiliser d'autres formalismes tels System Z pour réduire le nombre de modèles d'un ensemble de formules. De plus, l'utilisation de l'opérateur $\leq_P$ permet de comparer deux formules deux à deux, ce qui est très expressif. Nous allons donc choisir d'utiliser ces deux logiques dans la suite du mémoire pour représenter des préférences.

Chapitre 5

Contribution à la représentation d'exigences et de réglementations en utilisant une logique de préférences

Nous avons vu dans les chapitres précédents que les exigences et les phrases normatives peuvent être représentées en utilisant des préférences. En étendant l'approche proposée par Cholvy et Hunter dans [44], nous allons utiliser la logique CO pour représenter :

- les exigences d'un agent sous forme d'une *position*, i.e. chaque agent exprimera un ordre de préférence sur les exigences qu'il émet ;
- les contraintes du domaine, i.e. les contraintes induites par les lois physiques par exemple, et qui permettent de considérer que certains mondes logiquement possible sont impossibles à atteindre en réalité ;
- des phrases normatives concernant le domaine de l'objet à concevoir. Nous allons montrer que l'utilisation de CO permet de représenter des obligations conditionnelles, des normes avec exceptions et des Contrary-to-Duties.

L'utilisation d'un formalisme commun pour représenter à la fois des exigences, une réglementation et des lois du domaine va nous permettre de vérifier qu'un ensemble d'exigences est cohérent avec une réglementation donnée et des contraintes du domaine.

5.1 Représentation d'exigences

5.1.1 Ensemble d'exigences

Nous allons considérer que les exigences sont exprimées grâce à des formules de la logique propositionnelle. Cette hypothèse est restrictive dans le sens où on ne pourra pas par exemple quantifier des variables ou utiliser des prédicats pour affiner la représentation des exigences, mais elle va nous permettre de fournir dans la suite du mémoire un formalisme unique traitant des exigences et des positions. On considérera un ensemble fini $\{a_1, \ldots, a_m\}$ d'agents qui émettent des exigences.

Définition 5.1.1. *L'ensemble des exigences émises par un agent a est noté Δ_a et est défini comme étant un ensemble cohérent de formules de $PROP$.*

On remarquera que l'ensemble des exigences émises par un agent doit être cohérent. En effet, les exigences étant des propriétés que doit satisfaire l'objet à concevoir, il paraît raisonnable de supposer que l'agent n'exige pas que l'objet satisfasse une proposition et sa négation.

Exemple. *On considère un agent X qui désire construire une maison. Les exigences qu'il exprime vis-à-vis de la maison sont les suivantes :*

- *X exige que la maison ne soit pas proche d'une station de métro (à cause de la gêne provoquée par le bruit) ou qu'elle soit bien insonorisée ;*
- *X exige que si la maison n'est pas proche du centre ville, alors elle devra être proche d'une station de métro.*

Si l'on dispose d'un langage propositionnel contenant les variables I (la maison est insonorisée), M (la maison est proche d'une station de métro) et C (la maison est proche du centre ville), alors on peut exprimer l'ensemble des exigences de l'agent par $\Delta_X = \{I \vee \neg M, \neg C \rightarrow M\}$. Cet ensemble d'exigences est cohérent.

5.1.2 Préférences sur les exigences

Chaque agent a dispose d'un ensemble de formules propositionnelles qui représente les exigences qu'il émet vis-à-vis de l'objet à concevoir. Nous allons maintenant raffiner cette représentation en utilisant un préordre sur les exigences de l'agent. Par exemple, l'exigence « si la maison n'est pas proche du centre ville, alors elle devra être proche d'une bouche de métro » émise par X peut lui sembler prioritaire par rapport à l'exigence « la maison n'est pas proche d'une station de métro ». En particulier, la non satisfaction de la deuxième exigence peut être moins importante pour X que la non satisfaction de la première exigence.

Les exigences d'un agent a vont maintenant être représentées par un ensemble d'exigences et un préordre sur ces exigences (remarquons que l'on peut considérer ce préordre comme un ordre, car on regroupe les exigences qui ont la même importance dans une même formule grâce à une conjonction). Ces deux éléments vont être représentés par un tuple.

Définition 5.1.2. *Soit a un agent et $\Delta_a = \{\alpha_1, \ldots, \alpha_n\}$ l'ensemble d'exigences associé à a. On appelle* position *de l'agent a le tuple $\Gamma_a = [\alpha_1, \ldots, \alpha_n]$. Γ_a contient les mêmes formules que Δ_a et de plus α_i précéde α_j dans le tuple ssi l'agent a considère α_i comme étant prioritaire par rapport à α_j.*

Une position pour un agent définit donc une relation de préordre sur les exigences de l'agent. Chaque élément du tuple Γ_a est une formule de Δ_a ou une conjonction de formules de Δ_a (plusieurs formules de Δ_a peuvent avoir la même importance pour l'agent a).

Exemple. *Supposons que l'agent X ait les préférences suivantes sur ses exigences : l'exigence prioritaire pour X est que la maison ne soit pas proche d'une station de métro ou qu'elle soit correctement insonorisée. Dans ce cas, la position de X est $\Gamma_X = [\neg M \vee I, \neg C \rightarrow M]$.*

La signification intuitive de cette position est la suivante : dans le meilleur des cas, X veut que sa maison vérifie $\neg M \vee I$ et $\neg C \rightarrow M$. Mais si ce n'est pas possible (à cause des contraintes du domaine ou de la réglementation), il préfére que la maison vérifie $\neg M \vee I$ et ne vérifie pas $\neg C \rightarrow M$. Si ce n'est toujours pas possible, il accepte que la maison

satisfasse $\neg C \rightarrow M$ et ne satisfasse pas $\neg M \vee I$. Enfin, dans le pire des cas, il accepte que la maison ne satisfasse ni $\neg C \rightarrow M$ ni $\neg M \vee I$.

On peut donner tout comme Cholvy et Hunter une interprétation en terme de préordre sur les mondes à cette position. Si $\leq$ est un préordre sur les mondes associé à la position de l'agent (les mondes minimaux étant les plus préférés), on obtient alors :

$$||(\neg M \vee I) \wedge (C \vee M)|| \leq ||(\neg M \vee I) \wedge (\neg C \wedge \neg M)|| \leq ||(M \wedge \neg I) \wedge (C \vee M)|| \leq ||(M \wedge \neg I) \wedge (\neg C \wedge \neg M)||$$

Remarquons que l'on n'obtient pas forcément un ordre sur les mondes, car plusieurs mondes peuvent être indifférents.

Notre objectif est d'obtenir un moyen de traduire formellement une position d'un agent en un préordre sur les mondes possibles correspondant à l'idée intuitive donnée dans [44]. Pour cela, nous allons traduire la position de l'agent a en un ensemble de formules de CO dont les modèles vérifieront le préordre présenté ci-dessus. Nous allons dans un premier temps construire les formules associées à chaque « cluster » de mondes.

Définition 5.1.3. *Soit $\{\alpha_1, \ldots, \alpha_n\}$ un ensemble cohérent de formules propositionnelles. Soit $f_{[\alpha_1,\ldots,\alpha_n]} : N \rightarrow fl(PROP)$[1] la fonction qui associe à un entier $i \in \{0, \ldots, 2^n - 1\}$ la formule propositionnelle $\alpha_1(i) \wedge \ldots \wedge \alpha_n(i)$ de la façon suivante :*

- *i est decomposé en $i = \sum_{k=0}^{n-1} c_k(i) * 2^{n-1-k}$ avec $\forall k \in \{0, \ldots, n-1\}$ $c_k(i) = 0$ ou $c_k(i) = 1$;*
- *$\forall k \in \{1, \ldots, n\}$ $\alpha_k(i) = \begin{cases} \alpha_k & si\ c_{k-1}(i) = 0 \\ \neg\alpha_k & si\ c_{k-1}(i) = 1 \end{cases}$*

Dans l'exemple traité précedemment, on obtient :

$$\begin{array}{lcl} f_{\Gamma_X}(0) & = & (I \vee \neg M) \wedge (M \vee C) \\ f_{\Gamma_X}(1) & = & (I \vee \neg M) \wedge \neg(M \vee C) \equiv \neg M \wedge \neg C \wedge I \\ f_{\Gamma_X}(2) & = & \neg(I \vee \neg M) \wedge (M \vee C) \equiv \neg I \wedge M \wedge C \\ f_{\Gamma_X}(3) & = & \neg(I \vee \neg M) \wedge \neg(M \vee C) \equiv \bot \end{array}$$

On restreint maintenant la fonction f pour n'obtenir que des propositions satisfaisables :

Définition 5.1.4. *Soit $\{i_0, \ldots, i_{m_a}\} \subset \{0, \ldots, 2^n - 1\}$ l'ensemble des entiers tels que $\forall j \in \{i_0, \ldots, i_{m_a}\}$ $\not\vdash f_{\Gamma_a}(j) \leftrightarrow \bot$. f'_{Γ_a} est construite de la façon suivante :*

$$\forall j \in \{0, \ldots, m_a\}\ f'_{\Gamma_a}(j) = f_{\Gamma_a}(i_j)$$

Avec l'exemple précédent, on obtient maintenant :

$$\begin{array}{lcl} f'_{\Gamma_X}(0) & = & (I \vee \neg M) \wedge (M \vee C) \\ f'_{\Gamma_X}(1) & = & (I \vee \neg M) \wedge \neg(M \vee C) \equiv \neg M \wedge \neg C \wedge I \\ f'_{\Gamma_X}(2) & = & \neg(I \vee \neg M) \wedge (M \vee C) \equiv \neg I \wedge M \wedge C \end{array}$$

On voit que la fonction f' permet de trouver les « clusters » de mondes ordonnés par rapport à l'idée intuitive que nous avions présentée ci-dessus. En effet, le préordre sur les mondes correspondants à la position de l'agent est $||f'_{\Gamma_X}(0)|| \leq ||f'_{\Gamma_X}(1)|| \leq ||f'_{\Gamma_X}(2)||$. Nous allons maintenant construire à partir des formules données par f' un ensemble de formules de CO dont les modèles correspondront au préordre donné précédemment.

[1] $fl(PROP)$ représente l'ensemble des formules bien formées de $PROP$.

Définition 5.1.5. *Soit $\Gamma_a = [\alpha_1, \ldots, \alpha_n]$ la position de l'agent a. Cette position est représentée par l'ensemble de formules de CO noté Γ_a^{CO} et défini par :*

$$\Gamma_a^{CO} = \bigcup_{i \in \{0,\ldots,m_a-1\}} \{\overleftrightarrow{\Diamond} (f'_{\Gamma_a}(i) \wedge \overleftarrow{\Box} \neg f'_{\Gamma_a}(i) \wedge \Box \neg f'_{\Gamma_a}(i+1))\}$$

On notera :

$$\alpha <_E \beta \equiv_{def} \overleftrightarrow{\Diamond} (\alpha \wedge \overleftarrow{\Box} \neg\alpha \wedge \Box\neg\beta)$$

$$\alpha <_{\Gamma_a^{CO}} \beta \text{ ssi } \Gamma_a^{CO^*} \models \alpha <_E \beta$$

Revenons sur la formule $\overleftrightarrow{\Diamond} (f'_{\Gamma_a}(i) \wedge \overleftarrow{\Box} \neg f'_{\Gamma_a}(i) \wedge \Box\neg f'_{\Gamma_a}(i+1))$. Les CO-modèles de cette formule sont tels que :

- il existe un monde w_0 qui vérifie $f'_{\Gamma_a}(i)$;
- tous les mondes au moins autant préférés que w_0 ne vérifient pas $f'_{\Gamma_a}(i+1)$;
- tous les mondes moins préférés que w_0 ne vérifient pas $f'_{\Gamma_a}(i)$.

Les modèles de $f'_{\Gamma_a}(i) <_E f'_{\Gamma_a}(i+1)$ sont donc tels que le « moins bon » des $f'_{\Gamma_a}(i)$ mondes sera toujours meilleur que les $f'_{\Gamma_a}(i+1)$ mondes. Nous montrons également que $<_{\Gamma_a^{CO}}$ est bien une relation d'ordre.

Propriété 5.1.1. *$<_{\Gamma_a^{CO}}$ est une relation d'ordre sur les formules de PROP.*

Enfin, nous prouvons que les CO-modèles de Γ_a^{CO} respectent bien l'ordre intuitif donné sur les mondes :

Propriété 5.1.2. *Soit $\Gamma_a = [\alpha_1, \ldots, \alpha_n]$ la position de l'agent a. Les CO-modèles $\langle W, \leq, val\rangle$ de Γ_a^{CO} sont tels que : $\|f'_{\Gamma_a}(0)\| \leq \ldots \leq \|f'_{\Gamma_a}(m_a)\|$.*

Exemple. *Reprenons l'exemple précédent. On a $\Gamma_X = [\neg M \vee I, \neg C \rightarrow M]$ et :*

$$\begin{array}{lcl} f'_{\Gamma_X}(0) & = & (I \vee \neg M) \wedge (M \vee C) \\ f'_{\Gamma_X}(1) & = & (I \vee \neg M) \wedge \neg(M \vee C) \equiv \neg M \wedge \neg C \wedge I \\ f'_{\Gamma_X}(2) & = & \neg(I \vee \neg M) \wedge (M \vee C) \equiv \neg I \wedge M \wedge C \end{array}$$

On a donc $\Gamma_X^{CO} = \{(I \vee \neg M) \wedge (M \vee C) <_E \neg M \wedge \neg C \wedge I, \neg M \wedge \neg C \wedge I <_E \neg I \wedge M \wedge C\}$. On peut construire un modèle de Γ_X^{CO} et ainsi vérifier que le préordre sur les mondes est correct. Soit $M = \langle W, \leq, val\rangle$ un modèle de Γ_X^{CO}.

1. *$M \models \neg M \wedge \neg C \wedge I <_E \neg I \wedge M \wedge C$ signifie qu'il existe un monde w_1 tel que $M, w_1 \models (\neg M \wedge \neg C \wedge I) \wedge \Box(I \vee \neg M \vee \neg C) \wedge \overleftarrow{\Box} (M \vee C \vee \neg I)$.*

$$\|I \vee \neg M \vee \neg C\| \quad \overset{w_1}{\leq} \quad \|\neg M \wedge \neg C \wedge I\| \quad \leq \quad \|M \vee C \vee \neg I\|$$

2. *$M \models (I \vee \neg M) \wedge (M \vee C) <_E \neg M \wedge \neg C \wedge I$ signifie qu'il existe un monde w_0 tel que $M, w_0 \models (I \vee \neg M) \wedge (M \vee C) \wedge \Box(M \vee C \vee \neg I) \wedge \overleftarrow{\Box} ((\neg I \wedge M) \vee (M \wedge \neg C)$.*
   On peut en déduire facilement que $w_0 \leq w_1$ (car $M, w_1 \models (\neg M \wedge \neg C \wedge I)$ et que $M, w_0 \models \Box(M \vee C \vee \neg I)$). Les modèles de Γ_X^{CO} devront donc vérifer :

$$\overset{w_1}{\|I \vee \neg M \vee \neg C \wedge \|} \quad \leq \quad \|\neg M \wedge \neg C \wedge I\| \quad \leq \quad \|\neg I \wedge M \wedge C\|$$

$$\|M \vee C \vee \neg I\| \quad \leq \quad \overset{w_0}{\|(I \vee \neg M) \wedge (M \vee C)\|} \quad \leq \quad \|(\neg I \wedge M) \vee (\neg M \wedge \neg C)\|$$

soit :

$$\|(I \vee \neg M) \wedge (M \vee C)\| \quad \leq \quad \|\neg M \wedge \neg C \wedge I\| \quad \leq \quad \|\neg I \wedge M \wedge C\|$$

Grâce à la méthode de traduction des positions en formules de CO, on est capable d'exprimer les exigences d'un agent sous forme ordonnée en utilisant la logique CO. Nous allons maintenant voir comment exprimer des phrases normatives avec CO.

5.2 Représentation de phrases normatives avec CO

Nous allons étudier dans cette section la représentation de phrases normatives, i.e. de phrases portant sur des obligations, des permissions, des interdictions. Nous allons considérer que les obligations, permissions et interdictions ne porteront que sur des états du monde et non pas sur des actions par exemple. Ainsi, on pourra écrire « il est obligatoire que α soit vraie » si α représente une description du monde. On ne pourra pas exprimer par exemple qu'« il est interdit de faire a » où a représente une action.

Nous avons vu dans le chapitre 3 que l'on pouvait représenter des phrases normatives, en particulier les Contrary-to-Duties, en utilisant une sémantique fondée sur un ordre sur les mondes possibles. Nous allons donc utiliser CO pour modéliser les phrases normatives, puisque nous disposons grâce à CO d'un préordre sur les mondes. Dans cette étude, nous allons distinguer trois types de phrases normatives : tout d'abord les phrases normatives non conditionnelles sans exception qui sont les plus simples, puis les phrases normatives avec exception et enfin les Contrary-to-Duties. Ces travaux ont été présentés dans [34], [36] et [38].

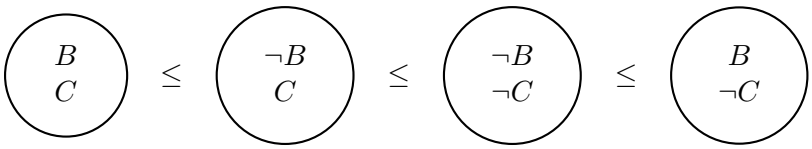
5.2.1 Phrases normatives non conditionnelles sans exception

Les phrases normatives non conditionnelles sans exception sont des phrases qui peuvent être de trois types :

- une obligation, comme par exemple « il est obligatoire que α soit vraie ». Nous considérons que l'interprétation de cette phrase est la suivante : les mondes les plus préférés vérifient α ;
- une permission, comme par exemple « il est permis que α soit vraie ». La sémantique de cette phrase est la suivante : il existe un monde parmi les mondes les plus préférés qui vérifie α ;
- une interdiction, comme par exemple « il est interdit que α soit vraie ». Dans ce cas, nous considérons que le sens donné à cette phrase est le suivant : les mondes les plus préférés vérifient $\neg\alpha$.

La sémantique que nous donnons à la notion d'obligation est assez proche de celle de SDL. En effet, dans SDL, la seule contrainte imposée à la relation d'accessibilité entre les mondes est la sérialité. On peut donc voir notre interprétation comme utilisant une relation d'accessibilité qui associe à un monde le monde le plus préféré du modèle (elle est évidemment sérielle).

Dans cette optique, on peut supposer que l'on utilise l'opérateur d'idéalité $I(-|\top)$ pour représenter une obligation du type précédent. Par exemple, modélisons la phrase « il est obligatoire que les murs de la maison soient blancs » par $I(B)$. Les modèles de $I(B) \equiv \overrightarrow{\Diamond}\ \Box B$ sont tels que qu'il existe un monde w_0 qui vérifie B et tels que tous les mondes au moins autant préférés que w_0 vérifient B. Si on considère une autre variable propositionnelle C, alors le modèle suivant est un modèle de $I(B)$:



On peut remarquer que le moins préféré des mondes est un monde qui vérifie B et donc l'obligation. On pourrait penser que tous les mondes qui vérifient B devraient être préférés aux mondes qui vérifient $\neg B$. Dans ce cas, on exprime une préférence *stricte* (cf. [13]). Or, l'obligation doit être vue en terme d'idéalité : dans tous les modèles de $I(B)$, B doit être vraie dans le monde le plus préféré. On peut remarquer comme nous l'avons précisé auparavant que la sémantique de SDL est basée sur une relation d'accessibilité sérielle : la seule contrainte qu'il existe sur la relation est qu'à partir d'un monde il y a toujours au moins un monde accessible. En particulier, il existe des modèles dans lesquels il n'y a qu'un seul monde vérifiant B accessible et non pas tous les mondes vérifiant B. On peut enfin remarquer que I vérifie plusieurs propriétés identiques à l'opérateur O de SDL :

Propriété 5.2.1. *L'opérateur I vérifie les schémas d'axiomes et la régle suivants :*

OC $I(\alpha) \wedge I(\beta) \rightarrow I(\alpha \wedge \beta)$

ON $I(\top)$

OD $\neg I(\bot)$

ROM $\frac{\vdash \alpha \rightarrow \beta}{\vdash I(\alpha) \rightarrow I(\beta)}$

L'utilisation de l'opérateur $I(-)$ comme opérateur d'obligation ne nous permettra donc pas d'échapper aux contradictions générées par SDL. Comme nous l'avons précisé dans le chapitre 3, trouver un opérateur qui ne respecte pas les schémas d'axiome et les règles d'inférences engendrant les paradoxes de SDL n'est pas un travail évident. Nous allons donc choisir $I(-)$ comme opérateur d'obligation, tout en sachant que nous pouvons générer des paradoxes avec cette approche.

Définition 5.2.1. *Les opérateurs d'obligation, de permission et d'interdiction que la proposition α soit vraie sont définis respectivement par $I(\alpha)$, $\neg I(\neg\alpha)$ et $I(\neg\alpha)$.*

5.2.2 Phrases normatives avec exception

Nous allons maintenant nous intéresser à la modélisation de phrases normatives avec exception. Par phrases normatives avec exception, nous entendons une des deux phrases suivantes :

- « α est interdite, mais si β est vraie, alors α est permise » ;
- « α est interdite, mais si β est vraie, alors α est obligatoire »

Remarquons que dans ces deux phrases, la première interdiction n'est plus en effet dès que β est vraie. En particulier, si β est vraie, on ne viole pas la première interdiction si

α est vraie. Nous allons profiter des propriétés de $I(-|-)$ qui est à la base un opérateur permettant d'exprimer des préférences révisables pour modéliser ce genre de phrases normatives.

Définition 5.2.2. *Une phrase « il est interdit que α soit vraie, mais si β est vraie, alors α est autorisée » sera modélisée par la formule $I(\neg\alpha) \wedge \neg I(\neg\alpha|\beta)$.*

Une phrase « il est interdit que α soit vraie, mais si β est vraie, alors α est obligatoire » sera modélisée par la formule $I(\neg\alpha) \wedge I(\alpha|\beta)$.

Intéressons nous à la première phrase. Soit $\mathcal{M} = \langle W, \leq, val\rangle$ un CO-modèle de $I(\neg\alpha) \wedge \neg I(\neg\alpha|\beta)$. Les mondes les plus préférés pour $\leq$ seront des $\neg\alpha$-mondes, car $\mathcal{M}$ est un modèle de $I(\neg\alpha)$. Par contre, comme $\mathcal{M}$ est un modèle de $\neg I(\neg\alpha|\beta)$, quelque soit le monde que l'on considère, soit il vérifie $\neg\beta$, soit il existe un monde plus préféré que lui qui vérifie $\beta \wedge \neg\alpha$. Donc, si on ne considère que les situations où β est vraie, les mondes les plus préférés ne verifieront pas obligatoirement $\neg\alpha$. La modélisation proposée est donc judicieuse (et c'est normal, car le comportement de $I(-|-)$ est fait pour raisonner avec des exceptions).

5.2.3 Contrary-to-Duties

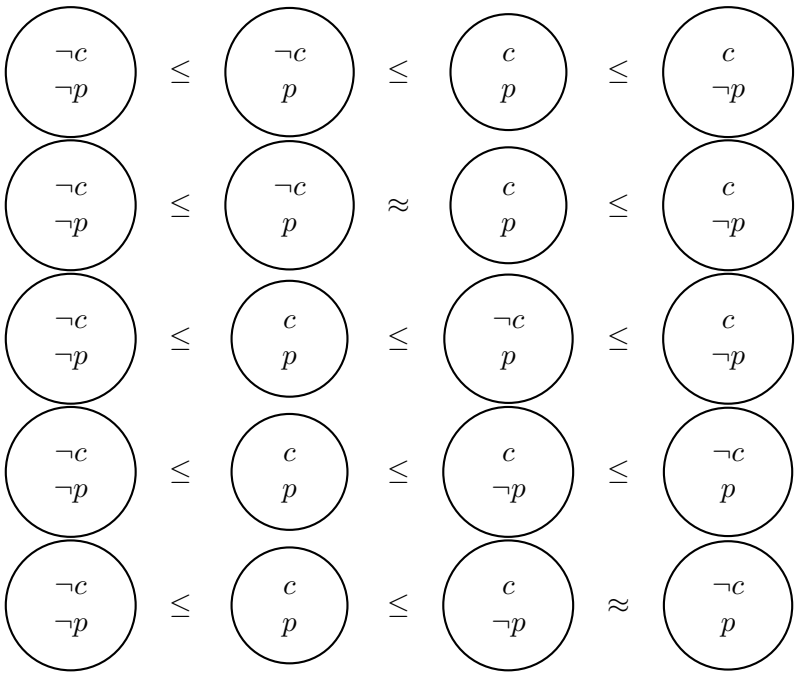
Nous nous intéressons maintenant à la modélisation de Contrary-to-Duties. Nous avons vu dans le chapitre 3 que SDL ne permettait pas de modéliser correctement les CTDs car elle ne propose qu'un niveau d'idéalité : les mondes sont soit idéaux, soit non idéaux. Or dans le cas du CTD, il existe des mondes parmi les mondes moins idéaux qui sont meilleurs que les autres (celui ou il y a un chien et un panneau par exemple). Une approche en terme de mondes ordonnés permet de représenter ces différents niveaux d'idéalité. Rappelons le scénario du chien :

(a) il ne doit pas y avoir de chien ;
(b) s'il n'y a pas de chien, il ne doit pas y avoir de panneau ;
(c) s'il y a un chien, il doit y avoir un panneau ;
(d) il y a un chien

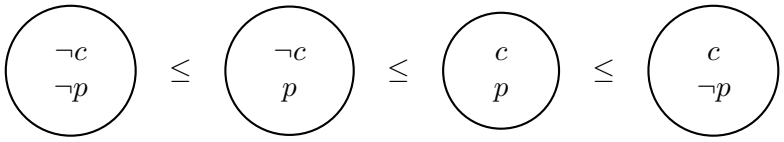
Supposons que l'on dispose d'un langage propositionnel à deux variables c et p et d'un préordre sur les mondes possibles $\leq$ ($w_0 \leq w_1$ signifie que w_0 est au moins autant préféré que w_1). Examinons la traduction sémantique des différentes phrases :

(a) il ne doit pas y avoir de chien signifie que le monde idéalement préféré est un monde qui falsifie c. Il existe donc deux candidats qui peuvent être le monde idéal : $\{\neg c, p\}$ et $\{\neg c, \neg p\}$;
(b) s'il n'y a pas de chien, il ne doit pas y avoir de panneau. Donc le monde $\{\neg c, \neg p\}$ est un monde plus idéal que $\{\neg c, p\}$. On en déduit en particulier que $\{\neg c, \neg p\}$ est le monde le plus préféré ;
(c) s'il y a un chien, il doit y avoir un panneau. On en déduit donc que le monde $\{c, p\}$ est préféré au monde $\{c, \neg p\}$.

On obtient plusieurs préordres possibles :



On pourrait là encore se demander pourquoi ne pas imposer au monde $\{\neg c, p\}$ d'être préféré à $\{c, p\}$. Ce serait alors considérer que la violation du fait de ne pas avoir de chien est plus importante que la violation de ne pas mettre de panneau s'il y a un chien. Dans [36, 38], nous supposions que le scénario du chien n'induisait qu'un seul préordre sur les mondes qui est le suivant :



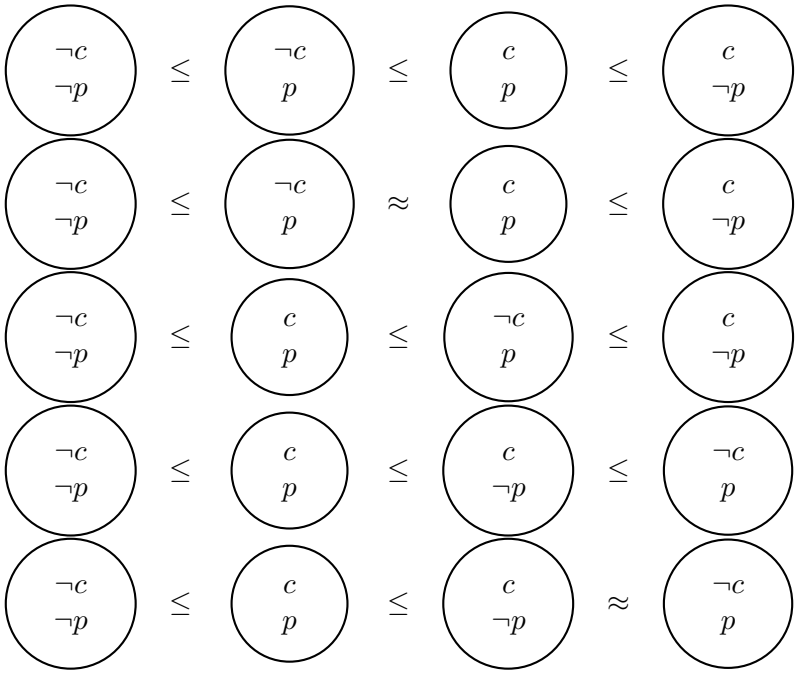
Dans ce cas, n'importe quel $\neg c$-monde sera toujours préféré à tous les c-mondes. Nous pensons que la contrainte que nous avions imposé sur les mondes (i.e. que les $\neg c$-mondes sont toujours préférés à tous les c-mondes) est trop forte. L'interdiction d'avoir un chien est une interdiction sans exception et doit être modélisée comme telle. Si l'on reprend la modélisation que nous avons faite d'obligations sans exceptions, nous avons utilisé l'opérateur $I(-)$ et il se pouvait donc très bien qu'un des mondes vérifiant l'interdiction soit un des mondes les moins préférés. Le fait de rajouter une deuxième règle particulière, le CTD, ne doit pas modifier ce point de vue.

Définition 5.2.3. *Soit le CTD exprimé par les phrases suivantes :*
- *il est interdit que α soit vraie ;*
- *si α est vraie alors il est obligatoire que β soit vraie.*

Ce CTD est modélisé par l'ensemble de formules de CO suivant : $\{I(\neg\alpha), I(\beta|\alpha)\}$.

Le scénario du chien va donc être modélisé par les formules suivantes :
(a) $I(\neg c)$
(b) $I(p|c)$
(c) $I(\neg p|\neg c)$
(d) c

Les CO-modèles[2] des trois premières formules sont les suivants :



On obtient bien le préordre intuitif que l'on cherchait.

Propriété 5.2.2. *La modélisation proposée du scénario du chien respecte les postulats (i) à (iv) de Carmo et Jones.*

La démonstration est évidente. Nous montrerons dans la partie III que cette modélisation accompagnée du modèle d'agent que nous proposons respecte les autres postulats de Carmo et Jones.

Revenons sur l'approche proposée dans [36]. Dans cet article, nous avions choisi de représenter le scénario du chien par l'ensemble de formules suivant :

(a) $I(\neg c)$
(b) $I(p|c) \wedge (p \wedge \neg c) \leq_P (p \wedge c)$[3]
(c) $I(\neg p|\neg c)$
(d) c

Cette fois ci, il n'existe qu'un seul modèle de cet ensemble de formules. L'ajout de $(p \wedge \neg c) \leq_P (p \wedge c)$ permet de restreindre le préordre et de spécifier que les $\neg c$-mondes doivent être quoiqu'il arrive meilleurs que les c-mondes. Supposons maintenant que l'on utilise cette approche pour modéliser le même scénario, mais en supprimant la phrase (c). On obtient alors :

(a) $I(\neg c)$
(b) $I(p|c) \wedge (p \wedge \neg c) \leq_P (p \wedge c)$
(c) c

Dans ce cas, le monde $\{p, \neg c)$ sera toujours préféré au monde $\{p, c\}$. Mais il existe des modèles de cet ensemble de formules où le monde le moins préféré est $\{\neg p, \neg c\}$. Il existe alors des $\neg c$-mondes qui font partie des mondes les moins préférés. L'utilisation de

[2]Ce sont en fait des CO^*-modèles car on considère que pour chaque formule satisfaisable il existe un monde qui la satisfait.

[3]Rappelons que $\alpha \leq_P \beta \equiv_{def} \overset{\rightrightarrows}{\Box} (\beta \rightarrow \Diamond\alpha)$.

$(p \wedge \neg c) \leq_P (p \wedge c)$ ne fonctionne que dans le cas précis du scénario du chien. Nous pensons donc que l'ajout de $(p \wedge \neg c) \leq_P (p \wedge c)$ est purement artificiel et que le fait que $\{p, \neg c\}$ et $\{p, c\}$ soient deux mondes indifférents est la traduction correcte du scénario du chien. Remarquons de plus que l'ajout de $(p \wedge \neg c) \leq_P (p \wedge c)$ empêche la modélisation de respecter le postulat (iv) de Carmo et Jones.

5.2.4 Conclusion

Rappelons la modélisation des différentes phrases normatives possibles :

il est obligatoire que α soit vraie	$I(\alpha)$
il est autorisé que α soit vraie	$T(\alpha) \equiv \neg I(\neg\alpha)$
il est interdit que α soit vraie	$I(\neg\alpha)$
il est normalement interdit que α mais si β est vraie, alors α est autorisé	$I(\neg\alpha) \wedge \neg I(\neg\alpha\|\beta)$
il est normalement interdit que α mais si β est vraie alors α est obligatoire	$I(\neg\alpha) \wedge I(\alpha\|\beta)$
Contrary-To-Duty : (RP) Il est interdit que α soit vraie (CTD) Mais si α est vraie, alors il est obligatoire que β soit vraie	 $I(\neg\alpha)$ $I(\beta\|\alpha)$

L'approche que nous proposons permet de modéliser un grand nombre de phrases normatives de nature différente. La sémantique de CO en terme de mondes ordonnés permet de représenter des obligations, permissions et interdictions « classiques », mais aussi des normes avec exceptions et des Contrary-to-Duties. Enfin, l'utilisation de l'opérateur $I(-)$ pour modéliser une obligation classique dans tous les cas apporte un avantage certain : on peut facilement rajouter un CTD par exemple, sans être obligé de réécrire les normes déjà existantes.

Remarquons tout de même que cette formalisation ne nous permet d'échapper aux paradoxes classiques de SDL. De plus, certains théorèmes déduits sont contre-intuitifs. Ainsi, $\vdash I(\neg\alpha) \wedge I(\alpha|\beta) \rightarrow I(\neg\beta)$. Notre souci était d'obtenir un premier formalisme permettant de modéliser toutes les notions présentées dans ce chapitre, mais il reste bien sûr à affiner.

5.3 Représentation de contraintes du domaine avec CO

Les contraintes du domaine représentent les contraintes physiques ou les contraintes inhérentes au domaine de l'objet à concevoir. On peut les voir comme des contraintes d'intégrité sur une base de données. Elles représentent ce qui est nécessairement vrai dans le monde réel. Si on travaille avec la logique CO, on peut restreindre les mondes possibles à des mondes qui vérifient les contraintes du domaine. La logique CO^* ne le permet pas, car toute proposition satisfaisable (au sens de la logique propositionnelle) doit être vraie dans au moins un monde.

Définition 5.3.1. *Une contrainte du domaine du type « la proposition α est toujours vraie » est représentée par la formule* $\overline{\Box}\,\alpha$.

Ainsi, si on a une contrainte du type « les grandes maisons coûtent plus de 100 000 euros », on peut la modéliser par la formule $\overleftrightarrow{\Box} G \rightarrow P\_100000$ (G représente le fait qu'une maison est grande et P_100000 qu'elle coûte plus de 100 000 euros). Les CO-modèles de cette formule sont tels que le monde $\{G, \neg P\_100000\}$ n'existe pas. Il n'existe bien sûr pas de CO^*-modèle de cette formule, car dans CO^*, on a $\overleftrightarrow{\Diamond} (G \wedge P\_100000)$.

Définition 5.3.2. *Un ensemble de contraintes du domaine $\mathcal{C}$ est un ensemble cohérent de formules du type $\overleftrightarrow{\Box} \alpha$.*

On considérera ainsi qu'une proposition et sa négation ne peuvent pas être impossible physiquement en même temps. Cette hypothèse est raisonnable, dans le sens où on ne trouve pas « dans la nature » de proposition α telle que α et $\neg\alpha$ soient toujours vraies.

5.4 Cohérence d'exigences avec une réglementation et des contraintes du domaine

Nous allons maintenant nous intéresser à la cohérence d'exigences avec une réglementation et des contraintes du domaine. Pour vérifier la cohérence d'un ensemble d'exigences avec une réglementation et des contraintes du domaine, il convient tout d'abord de vérifier que les contraintes du domaine et la réglementation sont également cohérentes. Il nous paraît en effet correct de supposer qu'on ne peut pas interdire quelque chose qui est vrai de par une contrainte physique du monde réel.

Définition 5.4.1. *Soit $\mathcal{R}$ un ensemble de phrases normatives. $\mathcal{R}$ est cohérent avec la contrainte du domaine $\overleftrightarrow{\Box} \varphi$ ssi $\forall \psi$ telle que $\psi \wedge \varphi$ est satisfaisable, alors $\mathcal{R} \not\models I(\neg\varphi|\psi)$. $\mathcal{R}$ est cohérent avec l'ensemble de contraintes du domaine $\mathcal{C} = \{\overleftrightarrow{\Box} \varphi_1, \ldots, \overleftrightarrow{\Box} \varphi_l\}$ ssi $\mathcal{R}$ est cohérent avec $\overleftrightarrow{\Box} (\varphi_1 \wedge \ldots \wedge \varphi_l)$.*

Il ne suffit pas de vérifier que les formules idéales de la réglementation représentées par les formules du type $I(\varphi)$ sont compatibles avec les contraintes du domaine, il faut également vérifier que toutes les règles avec exception et les CTDs le sont aussi (c'est le sens de $\mathcal{R} \not\models I(\neg\varphi|\psi)$, qui permet de vérifier que les règles conditionnelles sont bien cohérentes avec φ). On ne doit jamais pouvoir déduire que quelque chose d'impossible (au sens des contraintes du domaine) est obligatoire. Les contraintes du domaine sont donc plus importantes que la réglementation. Remarquons également qu'il serait intéressant de trouver un algorithme performant permettant de tester si une réglementation est cohérente avec des contraintes du domaine.

Exemple. *Considérons la réglementation suivante : il est obligatoire d'avoir une ligne électrique dans une maison, mais si on n'en a pas, alors il est obligatoire d'avoir un groupe électrogène. On modélise cette réglementation par l'ensemble $\{I(le), I(ge|\neg le)\}$. Supposons que les contraintes du domaine nous indique qu'il est impossible d'avoir un groupe électrogène (par exemple parce que la maison considérée est trop grosse, un groupe électrogène ne suffirait pas à lui fournir de l'électricité). Dans ce cas, $\{I(le), I(ge|\neg le)\}$ et $\overleftrightarrow{\Box} \neg ge$ sont incohérents, car il existe des cas prévus par la réglementation (lorsque l'on n'a pas de ligne électrique) où il est obligatoire d'avoir un groupe électrogène.*

Comme nous disposons d'une position Γ_a pour chaque agent a, on dispose de plusieurs propositions qui sont ordonnées suivant l'ordre de préférence fourni par l'agent : ce sont les

propositions $f'_a(i)$ pour $i \in \{0, \ldots, m_a\}$. Nous allons maintenant déterminer quelles sont les propositions parmi celles-ci qui vérifient les contraintes du domaine.

Définition 5.4.2. *Soit Γ_a la position de l'agent a. Soit $\mathcal{C}$ un ensemble de contraintes du domaine. L'ensemble $\mathcal{F}'_{\mathcal{C}}(a)$ est défini par :*

$$\mathcal{F}'_{\mathcal{C}}(a) = \{f'_a(i) \ : \ i \in \{0, \ldots, m_a\} \text{ et } \forall \varphi \text{ telle que } \mathcal{C} \models^{\leftrightarrows}_{\Box} \varphi \ , \ \varphi \wedge f'_a(i) \text{ est satisfaisable}\}$$

L'ensemble $\mathcal{F}'(a)$ représente donc l'ensemble des formules du type $f'_a(i)$ qui sont compatibles avec les contraintes du domaine. Remarquons que l'ordre $\leq_{\Gamma_a^{CO}}$ représente toujours la préférence de l'agent sur ce sous-ensemble de formules.

Propriété 5.4.1. *Soit $\mathcal{C}$ un ensemble de contraintes du domaine. $\mathcal{F}'_{\mathcal{C}}(a)$ est non vide.*

On est donc toujours sûr de trouver un ensemble de formules du type $f'_a(i)$ qui soient compatibles avec les contraintes du domaine correspondant à l'objet à concevoir. Il faut maintenant trouver quelles vont être les formules qui sont compatibles avec une réglementation donnée.

Définition 5.4.3. *Soit Γ_a la position de l'agent a. Soient $\mathcal{C}$ un ensemble de contraintes du domaine et $\mathcal{R}$ une réglementation cohérente avec $\mathcal{C}$. L'ensemble $\mathcal{F}'_{\mathcal{C},\mathcal{R}}(a)$ est défini par :*

$$\mathcal{F}'_{\mathcal{C},\mathcal{R}}(a) = \{f'_a(i) \in \mathcal{F}'_{\mathcal{C}}(a) \ : \ \mathcal{R} \models T(f'_a(i))\}$$

Rappelons que par définition $T(\alpha) \equiv_{def} \neg I(\neg\alpha)$. L'ensemble $\mathcal{F}'_{\mathcal{C},\mathcal{R}}(a)$ va donc contenir les formules de $\mathcal{F}'_{\mathcal{C}}(a)$ (donc cohérentes avec les contraintes du domaine) qui sont tolérées par la réglementation (elles peuvent également être obligatoires). On obtient donc un ensemble de formules compatibles avec les contraintes du domaine et la réglementation.

Propriété 5.4.2. *Soient $\mathcal{C}$ un ensemble de contraintes du domaine et $\mathcal{R}$ une réglementation cohérente avec $\mathcal{C}$. $\mathcal{F}'_{\mathcal{C},\mathcal{R}}(a)$ est un ensemble non vide.*

Nous disposons donc, grâce à la position de l'agent a d'un ensemble de formules non vide qui sont cohérentes avec la réglementation et les contraintes du domaine. Cet ensemble de formules est ordonné par $\leq_{\Gamma_a^{CO}}$ qui représente le degré de préférence de l'agent par rapport à ces formules. Nous allons donc choisir les formules les plus préférées qui vont représenter les meilleures exigences possibles pour l'agent étant données une réglementation et des contraintes du domaine. Bien sûr, si l'agent a introduit des disjonctions dans ses exigences, il se peut qu'il soit obligé de vérifier que les choix possibles sont cohérents avec les contraintes du domaine et la réglementation.

Définition 5.4.4. *Soient Γ_a position de l'agent a, $\mathcal{C}$ un ensemble de contraintes du domaine, $\mathcal{R}$ une réglementation cohérente avec $\mathcal{C}$. On appelle* meilleures exigences de a compatibles avec $\mathcal{C}$ et $\mathcal{R}$ *la formule $\gamma_a(\mathcal{C},\mathcal{R}) = \min_{\leq_{\Gamma_a^{CO}}} \mathcal{F}'_{\mathcal{C},\mathcal{R}}(a)$.*

Exemple. *Supposons que M. X émette les exigences suivantes à propos de sa future maison. Il souhaiterait que :*

- *la maison soit au centre ville ou proche d'une station de métro ;*
- *la maison ait des murs blancs ;*
- *la maison coûte moins de 100 000 euros et soit dans un quartier peu bruyant.*

La position de M. X respecte l'ordre des exigences ci-dessus. Les contraintes du domaine sont les suivantes :

- *si une maison est proche d'une station de métro, alors elle n'est pas dans un quartier peu bruyant ;*
- *si une maison est au centre ville, alors elle coûte plus de 100 000 euros.*

Enfin la réglementation stipule que si une maison est au centre ville, alors ses murs ne doivent pas être peints en blanc et qu'il est interdit de construire une maison sous une ligne haute tension.

Si l'on dispose d'un langage propositionnel dont les variables sont C (la maison est au centre ville), M (la maison est proche d'une station de métro), B (les murs de la maison sont blancs), L (la maison coûte moins de 100 000 euros), T (la maison est dans un quartier peu bruyant) et H (la maison est sous une ligne haute tension), la position de M. X, l'ensemble des contraintes du domaine et la réglementation s'écrivent respectivement :

$$\Gamma_X = [C \vee M, B, L \wedge T]$$

$$\mathcal{C} = \{\overleftrightarrow{\Box}\ (M \rightarrow \neg T), \overleftrightarrow{\Box}\ (C \rightarrow \neg L)\}$$

$$\mathcal{R} = \{I(\neg B|C), I(\neg H)\}$$

Remarquons tout de suite que $\mathcal{C}$ est bien un ensemble cohérent de formules de CO et que $\mathcal{R}$ est bien cohérente avec C.

L'ensemble des formules f'_{Γ_X} est défini par :

$$\begin{array}{lcl}
f'_{\Gamma_X}(0) & = & (C \vee M) \wedge B \wedge (L \wedge T) \\
f'_{\Gamma_X}(1) & = & (C \vee M) \wedge B \wedge (\neg L \vee \neg T) \\
f'_{\Gamma_X}(2) & = & (C \vee M) \wedge \neg B \wedge (L \wedge T) \\
f'_{\Gamma_X}(3) & = & (C \vee M) \wedge \neg B \wedge (\neg L \vee \neg T) \\
f'_{\Gamma_X}(4) & = & (\neg C \wedge \neg M) \wedge B \wedge (L \wedge T) \\
f'_{\Gamma_X}(5) & = & (\neg C \wedge \neg M) \wedge B \wedge (\neg L \vee \neg T) \\
f'_{\Gamma_X}(6) & = & (\neg C \wedge \neg M) \wedge \neg B \wedge (L \wedge T) \\
f'_{\Gamma_X}(7) & = & (\neg C \wedge \neg M) \wedge \neg B \wedge (\neg L \vee \neg T)
\end{array}$$

$\mathcal{F}'_C(X) = \{f'_{\Gamma_X}(1), f'_{\Gamma_X}(3), \ldots, f'_{\Gamma_X}(7)\}$ car $f'_{\Gamma_X}(0)$ et $f'_{\Gamma_X}(2)$ sont incompatibles avec les contraintes de domaine : $((C \vee M) \wedge (L \wedge T)) \wedge ((M \rightarrow \neg T) \wedge (C \rightarrow \neg L))$ est incohérente.

Les formules restantes sont quant à elles toujours cohérentes avec la réglementation (en particulier parce que l'on a $(C \vee M)$). On va donc choisir comme meilleure exigence de X compatible avec $\mathcal{R}$ et $\mathcal{C}$ la formule préférée pour $\leq_{\Gamma_X^{CO}}$, soit $f'_{\Gamma_X}(1)$. Les exigences de l'agent sont donc représentées par la formule $(C \vee M) \wedge B \wedge (\neg L \vee \neg T)$.

On remarque que l'agent a le choix en particulier entre C et M. Or B est une exigence de l'agent, donc pour respecter la réglementation, l'agent ne peut pas choisir que sa maison soit en centre ville. Elle sera donc proche d'une station de métro et sera bruyante. L'agent peut tout de même choisir une maison qui coûte moins de 100 000 euros. Les exigences de M. X sur la maison vont donc être :

- *la maison est proche d'une bouche de métro ;*
- *la maison a des murs blancs ;*
- *la maison sera dans un quartier bruyant.*

La maison sera bruyante, mais elle respecte les deux premieres exigences par ordre de priorité : elle est proche d'une station de métro ou en centre ville et a des murs blancs. Une maison située en centre ville (donc sans murs blancs) mais tranquille convient moins d'après la position de M. X car elle respecte la première exigence et la troisième, mais pas la deuxième.

5.5 Conclusion

Ce chapitre a décrit notre contribution en ce qui concerne la modélisation des exigences, des contraintes du domaine et des réglementations. L'apport principal de cette contribution est le choix d'un formalisme unique (la logique de préférences CO) pour représenter ces différentes notions.

En effet, nous avons montré que toute position (notion introduite par Cholvy et Hunter pour exprimer des exigences ordonnées) correspond à un ensemble de formules de CO. Nous avons également montré quelles sont les formules de CO qui expriment les contraintes du domaine. Et enfin, nous avons montré comment représenter par la logique CO des phrases normatives complexes, comme des règles avec exception ou des Contrary-to-Duties.

Traduire toutes ces notions dans une logique unique nous a permis de définir la notion de cohérence entre une réglementation et les contraintes du domaine, puis la notion de meilleures exigences d'un agent, qui sont, rappelons-le, les exigences préférées par l'agent parmi toutes celles qu'il a exprimées et qui sont compatibles avec les contraintes et la réglementation.

Dans la partie suivante, nous allons étudier le cas où plusieurs agents émettent leurs propres exigences, ce qui peut soulever un problème de conflits entre exigences.

Deuxième partie

Gérer l'incohérence

Introduction

Lors de la phase d'expression des exigences, plusieurs agents émettent des exigences sur l'objet à concevoir. Il se peut très bien que deux agents aient des exigences cohérentes avec la réglementation et les lois du domaine, mais qui soient contradictoires. On peut supposer par exemple qu'un ingénieur réclame une solution technique pointue mais coûteuse, alors que le service financier de l'entreprise ait des limitations de budget. Il faut donc être d'abord capable de détecter ce genre de conflit, puis de le traiter, soit par une nouvelle phase d'expression des exigences par exemple, ou automatiquement.

Dans cette partie, nous allons tout d'abord présenter quelques formalismes qui permettent de gérer les problèmes d'incohérence en ingénierie des exigences, surtout dans des spécifications représentées par des points de vue. Nous nous intéresserons également à l'approche de Cholvy et Hunter qui proposent l'utilisation d'une technique de fusion prioritaire de bases de croyances pour pouvoir éliminer les incohérences possibles entre ensembles d'exigences : les agents sont ordonnés suivant un ordre d'importance et on considère que les exigences des agents les plus prioritaires doivent être satisfaites en priorité. Nous pensons que cette approche peut être étendue en développant un autre système de fusion, dit majoritaire, dans lequel les agents ont tous la même importance et qui simule un vote majoritaire. En effet, il existe des participants au processus d'émission des exigences qui n'ont aucune relation hiérarchique entre eux. Nous proposons donc d'utiliser un système de fusion majoritaire dans ce cas pour résoudre les éventuels problèmes. La combinaison de ces deux systèmes de fusion permet de représenter assez finement la structure hiérarchique existant dans une organisation responsable de l'émission d'exigences sur un artefact à concevoir.

Nous nous intéresserons plus particulièrement au problème de la fusion de bases de croyances dans un second chapitre. Nous présenterons ensuite une logique que nous avons développée qui permet de raisonner sur le résultat de la fusion majoritaire de plusieurs bases de croyances constituées de formules propositionnelles. Nous présenterons également un démonstrateur de théorèmes associé à cette logique et nous montrerons que notre approche peut être étendue partiellement au premier ordre. Enfin, nous montrerons que nous pouvons utiliser les résultats obtenus dans le cadre de la gestion de l'incohérence entre différents émetteurs d'exigences.

Chapitre 6

Gestion de l'incohérence en ingénierie des exigences (état de l'art)

La gestion de l'incohérence dans le processus d'ingénierie des exigences est un problème crucial. Les incohérences possibles peuvent apparaître à différents niveaux. Un agent qui émet des exigences peut exprimer des exigences contradictoires sans s'en rendre compte. Il peut également exprimer des exigences qui ne sont pas cohérentes avec une réglementation en vigueur ou des contraintes du domaine. Enfin, il peut exprimer des exigences cohérentes avec une réglementation et des contraintes du domaine mais qui sont incohérentes avec les exigences émises par un autre agent. Nous avons vu dans la partie précédente comment gérer les incohérences possibles entre les exigences d'un agent et une réglementation ou des contraintes du domaine grâce à la position fournie par un agent. Nous allons maintenant nous intéresser aux incohérences possibles entre les exigences de différents agents.

Dans le processus de développement d'un artefact un peu complexe, il n'est pas envisageable qu'un seul agent émette les exigences sur cet artefact. En effet, la conception d'un système quel qu'il soit requiert des compétences multiples (techniques, mais aussi financières ou ayant rapport avec la qualité etc). Ce sont plusieurs agents qui vont émettre un ensemble d'exigences qui leur paraissent judicieuses quant à la fonction finale de l'objet et des contraintes qu'ils imposent. Il se peut donc très bien que deux ou plusieurs agents émettent des exigences sur l'objet qui soient contradictoires. Or, on doit à la fin du processus d'ingénierie des exigences produire un seul ensemble cohérent d'exigences. Il faut donc régler le problème de l'incohérence pour pouvoir fournir cet ensemble cohérent d'exigences.

Nous allons nous intéresser dans ce chapitre à la gestion des incohérences possibles entre les exigences émises par plusieurs agents. Dans un premier temps, nous revenons sur la notion de point de vue et nous présentons un formalisme permettant de gérer les incohérences possibles entre points de vue, puis nous présentons la suite des travaux de Hunter et Nuseibeh (cf. section 2.2) dans le cadre de la gestion de l'incohérence. Enfin, nous présentons la solution proposée par Cholvy et Hunter qui repose sur l'utilisation d'une méthode de fusion.

6.1 Gestion de l'incohérence entre points de vue

La prise en compte de différents points de vue pendant le processus d'ingénierie des exigences pose un problème : elle remet en cause un principe qui était jusque là admis par tous, i.e. la conservation de la cohérence pendant tout le processus. En effet, la thèse habituellement admise était la suivante : puisque nous devons produire un ensemble cohérent d'exigences (ou dans le meilleur des cas de spécifications), exigeons une cohérence totale pendant tout le processus. Or, cette affirmation n'est pas forcement correcte pour plusieurs raisons [54, 53] :

- la vérification de cohérence à tout moment du processus d'ingénierie des exigences est quasiment impossible. Un nombre important d'agents participe à l'expression des exigences et on devrait vérifier la cohérence de l'ensemble des exigences à chaque fois qu'un participant modifie ou ajoute une exigence ;
- la non tolérance des incohérences *provisoires* pendant le processus laisse de côté certaines alternatives qu'il aurait été judicieux d'explorer ;
- enfin, les agents ont souvent des vues contradictoires sur les exigences et ces conflits peuvent être résolus en permettant aux différents agents d'exprimer différentes alternatives pour leurs exigences.

Bien sûr, on doit obtenir un ensemble cohérent d'exigences à la fin du processus, mais l'approche proposée par Easterbrook et al. permet de retarder les tests de cohérence jusqu'au moment opportun. En particulier, on n'a pas besoin de résoudre les éventuels conflits quand ils sont découverts, la phase de vérification de cohérence étant séparée de celle de résolution.

Nuseibeh a listé dans [104] les principales causes d'incohérence en ingénierie des exigences (avec des points de vue) :

- les participants n'ont pas la même vue de l'objet ;
- les participants n'utilisent pas le même langage ;
- les participants n'utilisent pas les mêmes stratégies ;
- les participants ne sont pas au même stade de développement ;
- leurs centres d'intérêts peuvent se recouvrir partiellement ou totalement ;
- les participants n'ont pas les mêmes buts techniques, économiques et/ou politiques.

Pour pouvoir déterminer si une incohérence existe, il faut disposer de règles de cohérence qui vont permettre de décrire les relations entre les différents points de vue, mais aussi les relations existant à l'intérieur de chaque point de vue. À partir de ces règles, on peut alors vérifier si un point de vue est incohérent ou si deux ou plusieurs points de vue sont incohérents entre eux.

6.1.1 Règles de cohérence

Finkelstein et al. partent du principe que l'incohérence peut être tolérée entre deux points de vue. Les règles de cohérence que nous allons écrire sont donc des règles qui *devraient* être vraies, et non des règles qui *sont* vraies. Il est bien sûr évident qu'à l'intérieur d'un même point de vue, la cohérence devra être respectée. Par contre, le développement de chaque point de vue ne devra pas être freiné par ces règles de cohérence. Les vérifications de cohérence devront être invoquées par chaque point de vue quand il « sent » qu'il en a besoin. Finkelstein et al. considérent donc trois grands types de cohérence dans les points de vue [53] :

- la cohérence locale d'un point de vue ;

– la cohérence inter-point de vue ;
– la cohérence globale du système.

Les deux premiers types de cohérence peuvent être utilisés dans le formalisme des points de vue. Au niveau local, chaque règle définit une propriété qui devrait être vérifié par un point de vue, et au niveau inter-point de vue, chaque règle définit une relation qui devrait être vérifiée entre plusieurs points de vue. La cohérence globale du système peut être quant à elle assurée en créant un point de vue qui englobe tous les autres. Les règles locales à ce point de vue permettent de vérifier une cohérence globale.

Il est également intéressant de pouvoir distinguer les règles qui vont vérifier l'*existence* (ou l'absence) d'une information et celle qui vérifient un *accord* (cohérence des informations entre deux points de vue, vérification des noms utilisés etc).

Easterbrook et al. [53] rappellent que Nuseibeh et al. ont défini dans [106] une notation pour exprimer des règles de cohérence inter-point de vue. Le principe est simple : on exprime une relation entre un point de vue *source* $\mathsf{VP_S}$ et un point de vue *cible* $\mathsf{VP_C}$. Les relations sont alors de la forme :

Définition 6.1.1.

$$\forall\ \mathsf{VP_S}\ \exists\ \mathsf{VP_C}\ tq\ (ps_1\ \mathcal{R}\ \mathsf{VP}(t,d)\ :\ ps_2)$$

où $\mathsf{VP}(t,d)$ donne le point de vue cible avec le template t[1] et le domaine d, et où ps_1 et ps_2 sont des spécifications partielles. Comme tout point de vue doit confronter une partie de ses spécifications avec d'autres points de vue, il est normal que cette relation s'applique à tous les points de vue. Perrussel dans [111] propose une notation plus "logique" pour cette relation :

Définition 6.1.2.

$$\forall \mathsf{P_i}\ \exists \mathsf{P_j}\ ((i \neq j)\ et\ (\mathsf{P_i}\ \mathcal{R}\ \mathsf{P_j}))$$

où $\mathsf{P_i}$ et $\mathsf{P_j}$ sont deux perspectives.

En réalité, la vérification d'une règle se fait en deux étapes : on vérifie d'abord l'existence du point de vue cible (il faut donc son template et son domaine) puis la relation qui doit lier les deux points de vue. Easterbrook et al. raffinent donc la définition 6.1.1 de la façon suivante :

Définition 6.1.3.

$$\exists\ \mathsf{VP_C}(t,d)\ tq\ (ps_S\ \mathcal{R}\ \mathsf{VP_C}(t,d)\ :\ ps_C)$$

Enfin, on peut noter que Perrussel a référencé quatre types de liens entre deux perspectives $\mathsf{P_1}$ et $\mathsf{P_2}$:

– Si la perspective $\mathsf{P_1}$ existe, alors la perspective $\mathsf{P_2}$ doit exister ;
– Si un élément o est présent dans $\mathsf{P_1}$, alors $\mathsf{P_2}$ doit exister ;
– Si un élément o est présent dans $\mathsf{P_1}$, alors un élément o' doit être présent dans $\mathsf{P_2}$;
– Si un élément o est présent dans $\mathsf{P_1}$, alors o doit aussi être présent dans $\mathsf{P_2}$;

On peut maintenant se demander comment traiter les incohérences qui risquent d'apparaître au cours du processus. Il est clair pour Finkelstein et al. dans [55] que le passage à la logique est nécessaire et que le traitement des incohérences se fera au méta-niveau (cf. figure 6.1). Les actions qui vont en résulter peuvent être de différents types : demande de renseignements complémentaires au propriétaire du point de vue, traitement automatique etc.

[1] Les points de vue sont construits par template, i.e. des points de vue où seuls le style de représentation et le plan de travail sont définis.

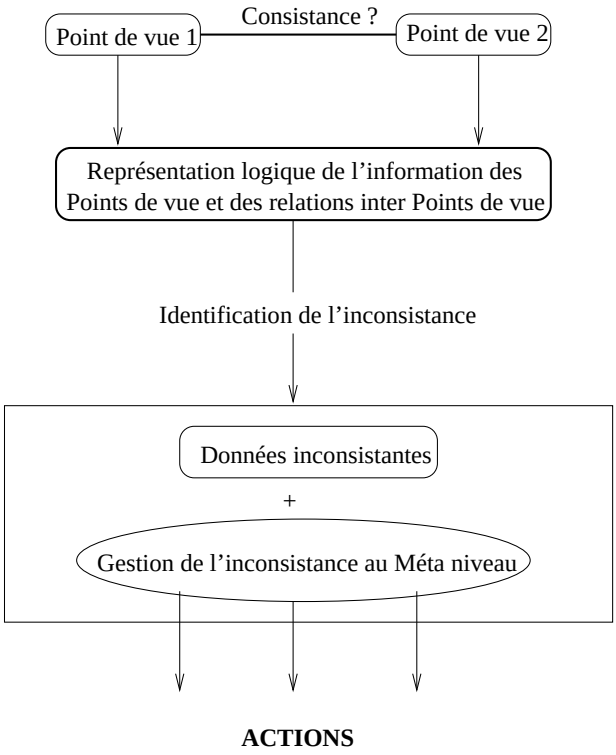


FIG. 6.1 – Gestion de l'incohérence entre les points de vue

6.1.2 Étude d'un exemple

Nous allons maintenant étudier un exemple simple tiré de [55] qui concerne un système de gestion de bibliothèque. Finkelstein et al. utilisent deux formalismes :

- une hiérarchie sur les agents (cf. figure 6.2) qui représente les flux d'informations entre les différents agents ;
- des diagrammes de flots de données pour chaque agent représentés par des tables d'action. On a représenté ici les tables d'actions pour les agents *Borrower* (cf. figure 6.3), *Clerk* (cf. figure 6.4) et *Librarian* (cf. figure 6.5).

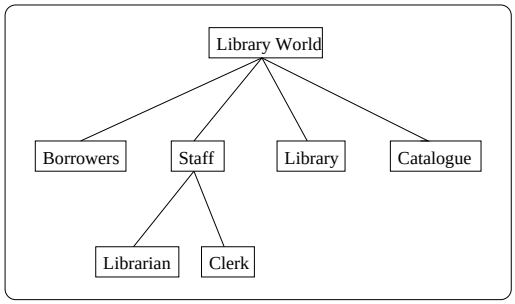


FIG. 6.2 – La hiérarchie sur les agents

SOURCE	INPUT	ACTION	OUTPUT	CIBLE
Borrower	book	check-in	book	Clerk
	card		card	
Librarian	book	check-out	book	Borrower

FIG. 6.3 – La table d'actions de l'agent *Borrower*

SOURCE	INPUT	ACTION	OUTPUT	CIBLE
			database update	Catalogue
Borrower	book	check-in	book	Librarian
			card	
	card	check-out	book	Borrower
			database update	Catalogue

FIG. 6.4 – La table d'actions de l'agent *Clerk*

6.1.3 Principes

Pour pouvoir utiliser la notion de point de vue, l'incohérence dans des points de vue va être définie comme l'incohérence dans des bases de données logiques distribuées. On a donc besoin de réécrire les spécifications et les relations inter-points de vue comme un ensemble de formules logiques. Dans l'approche proposée par Finkelstein et al., l'incohérence va être gérée dans l'environnement le plus large possible (i.e. pas seulement au niveau des données). On doit donc considérer à la fois les données de la base de données, mais également l'utilisation de cette dernière dans l'environnement.

L'analyse de l'incohérence se fait en termes de couples (D, E), où D est une base de données (un ensemble de formules logiques du premier ordre) représentant une partie de l'information contenue dans un ou plusieurs point de vue et E est une représentation logique de certaines assertions implicites et de certaines contraintes d'integrité utilisées pour contrôler et coordonner un ensemble de points de vue (E est l'environnement de la base de données). On pose pour hypothèse le fait que E *est cohérent.*

Les incohérences dans (D, E) sont gérées en adoptant un méta-langage qui va permettre de choisir les actions requises lors de la découverte d'une incohérence. Ces actions pourront agir directement sur les bases de données (appeler un TMS par exemple pour rétablir la cohérence de la base) ou demander des actions externes (demander plus d'informations à l'utilisateur etc).

SOURCE	INPUT	ACTION	OUTPUT	CIBLE
Clerk	book	shelve	book	Library

FIG. 6.5 – La table d'actions de l'agent *Librarian*

6.1.4 Traduction des tables d'actions

Il faut maintenant traduire les points de vue, donc les tables d'actions, en une base de données logique. Finkelstein et al. utilisent un formalisme qui se fonde sur les préconditions et les postconditions d'une action. On note $pre(X, Y)$ pour noter que X est une source et Y est une « conjonction » [2] d'entrées. De même, on note $post(X, Y)$ pour noter que X est la destination de Y.

La table d'action associée à B va être notée de la façon suivante :

$$table(A, P, B, Q)$$

où :
– A est un agent (le *domaine* du point de vue)
– P est une « conjonction » de préconditions pour B
– Q est une « conjonction » de postconditions pour B

Les informations contenues dans la table d'action de l'agent *Clerk* (cf.) vont donc être représentées par les formules logiques suivantes :

$$table(borrower, pre(borrower, book), check\_in, post(book, clerk)) \quad (6.1)$$

$$table(borrower, pre(borrower, card)\&pre(library, book), check\_out, post(book\&card, borrower)) \quad (6.2)$$

$$table(clerk, pre(borrower, book), check\_in, post(database\_update, catalogue) \&post(book, library)) \quad (6.3)$$

$$table(clerk, pre(borrower, book\&card), check\_out, post(book\&card, borrower) \&post(database\_update, catalogue)) \quad (6.4)$$

$$table(library, pre(clerk, book), shelve, post(book, library)) \quad (6.5)$$

Si l'on veut s'intéresser à la cohérence des informations représentées dans ce point de vue, ces formules logiques vont être contenues dans la base de données D. On peut également modéliser la hiérarchie des agents grâce à un prédicat *tree* :

$$tree(library\_world, borrower) \quad (6.6)$$

$$tree(library\_world, staf) \quad (6.7)$$

$$tree(library\_world, library) \quad (6.8)$$

$$tree(library\_world, catalogue) \quad (6.9)$$

$$tree(staff, librarian) \quad (6.10)$$

$$tree(staff, clerk) \quad (6.11)$$

[2] « conjonction » est notée par « & » et est un symbole de fonction dans la logique du premier ordre

Ces formules sont également des éléments de D. On rajoute alors des axiomes dans le méta-langage pour pouvoir exprimer les propriétés de l'arbre[3] :

$$\text{for all } X,\ tree(X,X) \tag{6.12}$$
$$\text{for all } X,\ Y,\ Z,\ [tree(X,Y)\ and\ tree(Y,Z) \rightarrow tree(X,Z)] \tag{6.13}$$
$$\text{for all } X,\ Y,\ X \neq Y \rightarrow [tree(X,Y) \rightarrow \neg tree(Y,X)] \tag{6.14}$$
$$\text{for all } X,\ Y,\ Z,\ [tree(X,Z)\ and\ tree(Y,Z)$$
$$\rightarrow tree(X,Y)\ or\ tree(Y,X)] \tag{6.15}$$
$$\text{for all } X,\ Y,\ [tree(X,Y)\ and\ \neg(\text{there exists a}$$
$$Z \text{ such that } tree(Y,Z)) \rightarrow leaf\,(Y)] \tag{6.16}$$

Ces formules vont quant à elles être des éléments de E car elles permettent de vérifier la cohérence de la hiérarchie. Ce sont des règles de cohérences locales pour le point de vue qui exprime cette hiérarchie. La traduction mathématique de ces règles est la suivante :

Règle	Signification
(6.12)	Réfléxivité
(6.13)	Transitivité
(6.14)	Antisymétrie
(6.15)	Up-linéarity
(6.16)	Leaf

Leaf permet de « reconnaitre » une feuille de l'arbre et Up-linéarity exprime le fait qu'il n'y a pas d'« héritage » multiple pour les flots de données.

6.1.5 Règles inter-points de vue

Les règles inter-points de vue vont être modélisées grâce au méta-langage, puisqu'elles vont concerner les élements des spécifications. Supposons que les deux règles à modéliser soient :

(Rule 1) (entre une hierarchie d'agents et les tables d'actions) :
toute « source » ou « destination » doit être une feuille de la hierarchie des agents

(Rule 2) (entre les tables d'actions) :
la sortie Z d'une table pour un agent X vers une destination Y doit être une entrée Z provenant d'une source X pour la table d'action de Y

La règle (Rule1) va être modélisée de la façon suivante[4] :

$$\text{for all } A_j\ [[table(\_, pre(A_1,\_)\ \&..\&\ pre(A_m,\_),\_,\_)]$$
$$\rightarrow\ \text{there exists } X \text{ such that } [leaf\,(X)\ and\ X = A_j]] \tag{6.17}$$
$$\text{for all } B_j\ [[table(\_,\_,\_, post(B_1,\_)\ \&..\&\ post(B_m,\_))]$$
$$\rightarrow\ \text{there exists } X \text{ such that } [leaf\,(X)\ and\ X = B_j]] \tag{6.18}$$

[3] Le prédicat *leaf* exprime le fait qu'un élément est une feuille de l'arbre.
[4] _ signifie qu'on ne s'intéresse pas au champ concerné

La règle (Rule 2) va être modélisée de la façon suivante :

$$\begin{aligned}&\text{for all } A,\ B_i,\ C_i\ [[table(A, \_, \_, post(C_1, B_1)\&..\&post(C_m, B_m))]\\&\rightarrow [table(B_1, X_1, \_, \_)\ and\ ...\ and\ table(B_m, X_m, \_, \_)]]\\&\text{où } i \in \{1 \dots m\} \text{ et } pre(A, C_i) \text{ est une conjonction dans } X_i. \end{aligned} \tag{6.19}$$

Ces trois formules vont bien sûr faire partie de E. On peut remarquer qu'elles respectent la forme donnée par la définition 6.1.3.

6.1.6 Vérification de cohérence

On peut maintenant vérifier si il existe des incohérences dans un point de vue ou entre deux points de vue. Comme on utilise une logique des prédicats, on va se servir de l'*hypothèse du monde clos* (souvent notée CWA pour *Closed World Assumption*) pour déduire certains faits dans D. Cette hypothèse nous permet de considérer que les formules non représentées dans les bases sont fausses. Ces faits déduits, combinés aux formules déja présentes dans D et à certaines instanciations des règles de E, vont nous permettre de trouver les incohérences.

Par exemple, étudions les relations entre *Borrower* et *Clerk*. Supposons que la formule (6.3) ne soit pas représentée dans D. En utilisant CWA, on en déduit :

$$\begin{aligned}&\neg table(clerk, pre(borrower, book), check\_in,\\&post(database\_update, catalogue)\&post(book, library))\end{aligned}$$

En utilisant un démonstrateur automatique avec CWA, on montre facilement que (D, E) n'est pas cohérent et que les formules (6.1), (6.19) et (6.20) sont les sources de cette incohérence, car (6.1) et (6.19) impliquent :

$$table(clerk, pre(borrower, book), check\_in, X, Y)$$

où X et Y peuvent être instanciées avec n'importe quel terme du langage, alors que CWA donne :

$$\neg table(clerk, pre(borrower, book), check\_in, X, Y)$$

On peut également raisonner sur le fait que trop d'informations sont présentes. Supposons qu'il existe deux points de vue qui expriment :

$$\begin{aligned}VP1 \quad & reference\_book(childrens\_dictionnary) && (6.20)\\ & \text{for all X, } reference\_book(X) \rightarrow \neg lendable(X) && (6.21)\\ VP2 \quad & childrens\_book(childrens\_dictionnary) && (6.22)\\ & \text{for all X, } childrens\_book(X) \rightarrow lendable(X) && (6.23)\end{aligned}$$

De la même façon, si ces formules sont introduites dans D, on peut détecter l'incohérence dans (D, E) grâce à un démonstrateur automatique.

6.1.7 Conclusion

Le formalisme proposé dans [55] par Finkelstein et al. permet de vérifier la cohérence locale à un point de vue ou entre plusieurs points de vue. L'utilisation de la logique des prédicats comme outil de modélisation des spécifications de chaque point de vue et des relations inter-points de vue permet de ramener le problème de cohérence en Ingénierie des Exigences à un problème de cohérence de bases de données munies de contraintes d'intégrité.

Pour pouvoir utiliser ce formalisme, il faut par contre traduire toutes les spécifications des points de vue et les relations inter-points de vue dans un méta langage du permier ordre. Cette traduction peut évidemment s'avérer fastidieuse. De plus, comme nous allons le voir dans la prochaine section, l'identification d'une incohérence possible conduit à une action qui est souvent informelle. Le formalisme ne propose pas de résolution « automatique » de l'incohérence.

6.2 Utilisation de la logique QC étiquetée

Nous avons vu dans la section 2.2 que Hunter et Nusebeih ont proposé l'utilisation d'une logique quasi classique pour pouvoir représenter les exigences émises par différents points de vue. Nuseibeh et Hunter présentent de plus deux types d'analyse, indépendantes du langage QC, mais qui ont plus de sens dans le contexte des spécifications : l'*étiquetage* (ou *labelling*) et la *qualification d'inférences*. L'étiquetage va servir à identifier les sources d'incohérences, alors que la qualification d'inférences va permettre d'ordonner suivant un degré de crédibilité les différentes spécifications.

6.2.1 Identifier les sources d'incohérence

Après avoir identifié une incohérence dans une spécification, il est intéressant de pouvoir trouver les sources de cette incohérence avant de décider quelles actions réaliser. C'est l'utilisation du raisonnement QC étiqueté (*labelled QC reasoning*) qui va permettre de trouver ces sources. Hunter et Nuseibeh introduisent des étiquettes dans les formules pour pouvoir tracer les exigences.

Définition 6.2.1. *Soit S un ensemble de symboles atomiques et L une logique (par exemple QC). Si $i \subseteq S$ et $\alpha \in L$, alors $i : \alpha$ est une formule étiquetée.*

Par exemple, Résolution va s'écrire :

$$\frac{i : \neg\alpha \vee \beta \quad j : \alpha \vee \gamma}{i \cup j : \beta \vee \gamma}$$

Remarquons que $\beta \vee \gamma$ est étiquetée avec l'union des étiquettes de $\neg\alpha \vee \beta$ et $\alpha \vee \gamma$ qui est notée $i \cup j$. En effet, par définition, une étiquette est un sous-ensemble de S.

Pour illustrer l'utilisation des étiquettes, on prend l'exemple suivant :

(1) `{a}: patient_attend(London_Road)`

(2) `{b}: ¬patient_attend(London_Road)∨ambulance_disponible(Hopital)`

(3) `{c}:` ¬`ambulance_disponible(Hopital)`

Supposons de plus que (1) et (2) soient des spécifications stables et acceptées et que (3) est une spécification nouvelle que l'on essaye d'introduire, alors on peut considérer (3) comme la source d'incohérence dans `{a, b, c}:` $\perp$.

Définition 6.2.2. *Soit* Δ *un ensemble de formules étiquetées représentant des spécifications, et soit* i *une étiquette d'une conséquence de* Δ*. L'ensemble des assertions de* Δ *correspondant à cette étiquette est défini par :*

$$Formulae(\Delta, i) = \{j : \alpha \in \Delta | j \subseteq i\}$$

Pour une incohérence `i:` $\perp$, $Formulae(\Delta, i)$ est une source possible d'incohérence si $j \subseteq i$ et $Formulae(\Delta, i - j)$ est un sous ensemble cohérent de Δ.

Comme il peut y avoir plusieurs sources d'incohérence, on est obligé d'introduire un ordre sur les informations. Si i est plus grand (pour l'ordre sur les informations) que j, alors `i:` α a moins de chance de générer des incohérences que `j:` β. Cet ordre est transitif, mais pas nécessairement linéaire.

6.2.2 Qualification des inférences d'informations incohérentes

Lorsque l'on considère des informations incohérentes, on a plus confiance dans certaines inférences que d'autres. Par exemple, on peut avoir plus confiance en α dérivée d'un ensemble cohérent si on ne peut pas dériver $\neg\alpha$ d'un autre ensemble cohérent.

Définition 6.2.3. *Soit* Δ *un ensemble de formules étiquetées représentant des spécifications. On forme les ensembles suivants de formules :*

$$CON(\Delta) = \{\Gamma \subseteq \Delta | \Gamma \nvdash_Q i : \perp\}$$

$$INC(\Delta) = \{\Gamma \subseteq \Delta | \Gamma \vdash_Q i : \perp\}$$

Hunter et Nusebeih définissent alors $MI(\Delta)$ comme étant un ensemble d'ensembles d'étiquettes, où chaque ensemble d'étiquettes correspond à un ensemble minimal de formules incohérentes. Un ensemble de formules incohérentes est dit *minimal* ssi chacun de ses sous-ensembles propres est cohérent. Enfin, ils définissent $MC(\Delta)$ comme un ensemble d'ensembles d'étiquettes, où chaque ensemble d'étiquettes correspond à un ensemble maximal de formules cohérentes. Un ensemble de formules cohérentes est dit *maximal* ssi l'ensemble est cohérent et si l'ajout de n'importe quelle formule à l'ensemble le rend incohérent.

On peut considérer un sous-ensemble maximal cohérent d'une spécification pour capturer une vue plausible de la spécification. $FREE(\Delta) = \cap MC(\Delta)$ est défini comme l'ensemble des informations non controversées de Δ. A l'inverse, $MI(\Delta)$ va représenter l'ensemble des informations problèmatiques de Δ.

Définition 6.2.4. *Soit* Δ *un ensemble de formules étiquetées représentant une spécification et soit* $\Delta \vdash_Q i : \alpha$*. On peut qualifier les inférences de trois façons :*

- α *est une inférence* existentielle *si* $\exists k \in MC(\Delta)$ *tel que* $i \subseteq k$*.*
- α *est une inférence* universelle *si* $\forall k \in MC(\Delta)$*,* $\exists j$ *tel que* $\jmath \subseteq k$ *and* $\Delta \vdash_Q j : \alpha$*.*
- α *est une inférence* libre *si* $i \subseteq FREE(\Delta)$*.*

Si α est une inférence existentielle, on a moins confiance en elle que si c'était une inférence universelle. Si α est une inférence libre, elle n'est associée avec aucune information incohérente. On peut remarquer que cette approche est proche du diagnostic, car on peut considérer que $MC(\Delta)$ est un ensemble de diagnostics et $MI(\Delta)$ un ensemble de conflits.

Par exemple, si l'on considère les assertions suivantes :

{a}: accident_arrive(London_Road)∨accident_rapporte(London_Road)
{b}: accident_arrive(London_Road)∨ ¬accident_rapporte(London-Road)
{c}: ambulance_disponible(London_Road)

On peut en déduire deux sous-ensembles maximaux cohérents :

Ensemble 1
{a}: accident_arrive(London_Road)∨accident_rapporte(London_Road)
{c}: ambulance_disponible(London_Road)

Ensemble 2
{b}: accident_arrive(London_Road)∨ ¬accident_rapporte(London_Road)
{c}: ambulance_disponible(London_Road)

On peut donc écrire que accident_rapporte(London_Road) et ¬accident_rapporte (London_Road) sont seulement des inférences existentielles, que ambulance_disponible (London_Road) est une inférence libre et que accident_arrive(London_Road) est une inférence universelle.

Ce genre de qualification est utile pour raisonner avec des informations incohérentes, parce qu'il permet de lier de façon claire les inférences et les données problématiques. On peut dire par exemple que accident_rapporte(London_Road) et ¬accident_rapporte (London_Road) sont des inférences dans lesquelles on peut avoir moins confiance que dans accident_arrive(London_Road), même si accident_arrive(London_Road) fait également partie des disjonctions qui posent problème.

Nous allons maintenant exposer l'exemple réel utilisé par Hunter et Nusebeih pour montrer l'utilité du formalisme que nous avons déjà présenté dans 2.2.

6.2.3 Exemple : Service Ambulancier de Londres

Exposé du problème

Le but de cette étude est de spécifier un système informatique de deploiement d'ambulances. Les exigences sont les suivantes :

Contrôleur de la salle d'incident
- une urgence médicale est la conséquence soit d'une maladie soit d'un accident ;
- à la reception d'un coup de fil signalant une urgence médicale, une ambulance doit être dépêchée sur les lieux ;
- à la reception d'un coup de fil, si l'incident n'est pas une urgence médicale, on doit transférer le coup de fil à une autorité compétente (la police, les pompiers, ...).

Responsable des opérations
- à la reception d'un coup de fil signalant un incident, si une ambulance est disponible, elle doit être dépêchée sur les lieux ;

– à la reception d'un coup de fil signalant un incident, si une ambulance n'est pas disponible, elle ne doit pas être dépêchée sur les lieux.

Responsable logistique

– s'il n'y a pas d'opérateurs (conducteurs, médecins) disponibles, il n'y a pas d'ambulance disponible ;
– S'il n'y a pas d'ambulance disponible, alors il faut chercher une ambulance libre ;
– si une année s'est écoulée depuis la dernière révision d'une ambulance, on doit réviser cette ambulance.

Représentation avec QC labellée

Hunter et Nusebeih utilisent QC labellée pour représenter ces informations :

Contrôleur de la salle d'incident

$\{a\}$: $\forall X,Y,\ accident(X,Y) \vee maladie(X,Y) \leftrightarrow urgence\_medicale(X,Y)$
$\{b\}$: $\forall X,Y,\ appel(X,Y) \wedge urgence\_medicale(X,Y) \rightarrow depecher\_ambulance(X,Y)$
$\{c\}$: $\forall X,Y,\ appel(X,Y) \wedge \neg urgence\_medicale(X,Y) \rightarrow transferer\_service(X,Y)$

Responsable des opérations

$\{d\}$: $\forall X,Y,\ call(X,Y) \wedge ambulance\_disponible(X) \rightarrow depecher\_ambulance(X,Y)$
$\{e\}$: $\forall X,Y,\ call(X,Y) \wedge \neg ambulance\_disponible(X) \rightarrow \neg depecher\_ambulance(X,Y)$

Responsable logistique

$\{f\}$: $\forall X,Y,\ \neg a\_un\_ou\_plus(X,Y) \rightarrow \neg ambulance\_disponible(X)$
$\{g\}$: $\forall X,Y,\ \neg ambulance\_disponible(X) \rightarrow commencer\_recherche\_nouvelle\_ambulance$
$\{m\}$: $\forall X,\ un\_an\_depuis\_derniere\_maintenance(X) \rightarrow commencer\_verification\_ambulance(X)$

6.2.4 Gestion de l'incohérence

Raisonnement

On rajoute les faits suivants :

$\{h\}$: $accident(Anthony, London\_Road)$
$\{i\}$: $appel(Anthony, London\_Road)$
$\{j\}$: $\neg a\_un\_ou\_plus(Ambulance1, Operateur)$
$\{k\}$: $\neg maladie(Anthony, London\_Road)$
$\{n\}$: $un\_an\_depuis\_derniere\_maintenance(Ambulance2)$

On peut générer les inférences (contradictoires) suivantes :

$\{a, b, h, i\}$: $depecher(Ambulance1, London\_Road)$
$\{e, f, i, j\}$: $\neg depecher(Ambulance1, London\_Road)$

En utilisant QC, on peut continuer à raisonner avec les faits précédents pour générer les inférences (utiles) :

```
{f, g, j}:   commencer_recherche_ambulance_libre
{m, n}:      commencer_verification_ambulance(Ambulance2)
```

On peut donc continuer à dériver des inférences utiles.

Analyse : qualification des inférences

On a :

```
{a, b, h, i}:   depecher(Ambulance1, London_Road)
{e, f, i, j}:   ¬depecher(Ambulance1, London_Road)
```

qui sont des inférences existentielles. On doit donc faire attention à ces inférences lorsque l'on va réviser les spécifications par exemple.

Au contraire, `{f, g, j}: commencer_recherche_ambulance_libre` est une inférence universelle. On sera donc moins enclin à la retirer lors du processus de révision.

Enfin, `{m, n}: commencer_la_verification_ambulance(Ambulance2)` est une inférence libre, ce qui semble normal.

Analyse : identification des sources d'incohérences

Il y a deux ensembles d'étiquettes qui sont liés avec des données à problème. Ce sont les étiquettes attachées aux deux inférences conflictuelles `{a, b, h, i}` et `{e, f, i ,j}`. Si l'on considère que les faits ajoutés ne peuvent pas être sources d'incohérence, i.e. l'ensemble `{h, i, j ,k, n}` ne cause pas de problème, on peut les ordonner comme meilleur que l'ensemble `{a, b, c, d, e, f, m}`.

On obtient donc un sous ensemble qui contient les sources probables de l'incohérence, i.e. `{a, b, e, f}`. On doit donc reconsulter les parties prenantes pour pouvoir résoudre ce problème.

Agir sur l'incohérence

Dans [73], Hunter et Nuseibeh proposent un formalisme permettant d'agir sur les incohérences (c'est d'ailleurs le même que dans [55]). Le travail précédent permettait de trouver et de qualifier les sources d'incohérences, donc de construire un « rapport » sur le système. En utilisant ce rapport, on doit être capable de trouver les actions qui doivent être exécutées pour résoudre les problèmes d'incohérence.

L'approche de Hunter et Nusebeih est la suivante : si l'on considère que la gestion de l'incohérence va nécessiter l'intervention d'un agent extérieur (un humain par exemple), on doit mettre en place un formalisme au méta-niveau qui contienne des règles du type `Incohérence Implique Action`. Le formalisme utilisé est une logique des actions temporelle qui va permettre d'exprimer le contexte passé et les sources d'incohérences pour pouvoir en déduire les actions futures.

Un exemple de règle est le suivant :

$$
\begin{aligned}
&[data(\Delta_1) \wedge data(\Delta_2)\\
&\wedge union(\Delta_1, \Delta_2) \vdash \bot\\
&\wedge source\_incoherence(union(\Delta_1, \Delta_2), S)\\
&\wedge probleme\_syntaxe(S)\\
&\wedge \neg LAST^1\ probleme\_syntaxe(S)\\
&\wedge \neg LAST^2\ probleme\_syntaxe(S)]\\
&\quad \rightarrow NEXT\ dire\_utilisateur(\text{« Y-a-t'il un probleme de syntaxe ? »}, S)
\end{aligned}
$$

Cette règle spécifie que l'utilisateur doit être consulté pour vérifier des erreurs de syntaxe. Hunter et Nusebeih utilisent des opérateurs temporels $LAST^n$ et $NEXT^n$ pour se situer à n unités de temps dans le passé ou le futur. De plus :

- $data(\Delta_1)$ et $data(\Delta_2)$ sont vraies si les bases de données Δ_1 et Δ_2 sont les traductions en logique classique de certaines spécifications ;
- $union(\Delta_1, \Delta_2) \vdash \bot$ est vraie si l'union de Δ_1 et de Δ_2 est incohérente ;
- $source\_incoherence(union(\Delta_1, \Delta_2), S)$ est vraie si S est un ensemble minimal incohérent de $union(\Delta_1, \Delta_2)$;
- $probleme\_syntaxe(S)$ est vraie si la cause de l'incohérence est un problème de syntaxe dans S ;
- $dire\_utilisateur(\text{« Y-a-t'il un probleme de syntaxe ? »}, S)$ est l'action à entreprendre.

On peut remarquer que les formules $LAST$ permettent d'éviter que des problèmes passés n'influent sur des actions futures. Nuseibeh dans [104] a identifié quatre types d'actions qui peuvent être générées pour réduire les problèmes d'incohérence :

- *ignorer* l'incohérence complétement et continuer le développement ;
- *isoler* les parties incohérentes des spécifications et continuer le développement ;
- *enlever* l'incohérence en corrigeant des erreurs ou en résolvant des conflits ;
- *améliorer* les situations incohérentes en augmentant la possibilité de résolution dans le futur.

Décider de l'action à choisir pour un type donné d'incohérence est donc le premier pas vers une gestion « propre » de l'incohérence.

6.2.5 Conclusion

L'utilisation d'étiquettes avec la logique QC permet non seulement de dériver des informations utiles à partir de spécifications incohérentes comme nous l'avions vu dans la section 2.2, mais également de trouver et de qualifier les sources d'incohérence. La construction d'ensembles maximaux cohérents conduit à trouver des ensembles d'étiquettes tels que les formules associées sont cohérentes. La qualification des sources d'incohérences à travers les notions d'inférence universelle ou existentielle fournit à l'utilisateur un moyen de savoir quelles sont les exigences les plus « sûres ».

Comme dans le formalisme présenté par Finkelstein et al., la gestion des conflits après la découverte d'incohérences se ramène à la génération d'actions. Ces actions sont souvent des envois d'informations aux émetteurs d'exigences, qui doivent alors agir en conséquence (réviser leurs exigences ou ne rien changer). En particulier, ces formalismes ne fournissent pas un ensemble d'exigences cohérent qui représenterait un consensus (calculé d'une manière particulière) entre les participants. Nous allons voir dans la section

suivante que Cholvy et Hunter utilisent une logique de fusion pour pouvoir dériver ce genre d'information.

6.3 Utilisation d'une méthode de fusion

Nous avons vu dans la section 2.4 que Cholvy et Hunter utilisent des positions pour représenter les exigences d'un agent sous forme ordonnée. Ils considèrent également une réglementation et un ensemble de contraintes du domaine et peuvent choisir la meilleure alternative possible au sens de l'agent en cas d'incohérence entre les exigences et la réglementation ou les contraintes du domaine.

Dans [44], ils proposent également de résoudre les incohérences existant entre les exigences émises par des agents différents en fusionnant ces exigences en suivant un ordre entre les différents agents. Prenons un exemple tiré de [44]. Considérons un centre de recherche qui doit acheter un nouvel ordinateur. Cet ordinateur doit être capable de gérer des données confidentielles, car la réglementation interne au centre l'impose. De plus, le centre dispose d'une quantité d'argent limitée pour l'achat de l'ordinateur. Le directeur demande donc à plusieurs membres de son personnel d'exprimer leurs exigences vis-à-vis du nouvel ordinateur ; il s'agit de l'ingénieur système, d'une secrétaire et d'un chercheur. La secrétaire et le chercheur seront les utilisateurs de l'ordinateur, alors que l'ingénieur système devra l'installer et le maintenir.

Comme ces trois personnes vont utiliser la machine d'une façon très différente, l'ensemble total des exigences peut être incohérent. La secrétaire par exemple peut exiger des propriétés qui ne sont disponibles sur aucune machine. Le chercheur peut vouloir une machine très puissante que l'ingénieur système ne veut peut être pas installer et maintenir ou qui est trop chère pour le budget disponible. De même, l'ingénieur système peut vouloir une machine qu'il connaît très bien, mais qui ne convient pas à la secrétaire ou au chercheur ou qui ne permet pas de gérer des données confidentielles.

Dans de telles situations, si une renégotiation des exigences des différents participants n'est pas possible, le directeur du centre va choisir en exprimant un ordre de priorité entre les trois agents. Par exemple, il peut considérer que comme il travaille dans un centre de recherche, les exigences les plus importantes sont celles exprimées par le chercheur. Les exigences de la secrétaire seront plus importantes que celles de l'ingénieur système, car elle travaille avec le chercheur et l'aide dans son travail (elle tape ses rapports etc). Bien sûr, le directeur peut choisir un autre ordre de priorité sur les agents. Il peut par exemple considérer que l'ingénieur système est prioritaire car c'est lui qui connait le mieux les machines. Cholvy et Hunter proposent donc deux stratégies pour fusionner les exigences de différents agents en utilisant un ordre sur ces agents.

6.3.1 Stratégie de fusion M1

Définition 6.3.1. *Soient n agents $a_1, \ldots, a_n$ et leurs positions associées $\Gamma_{a_1}, \ldots, \Gamma_{a_n}$. Soient $\gamma_{a_1}, \ldots, \gamma_{a_n}$ les ensembles d'exigences associés à Γ_{a_1}, ..., Γ_{a_n} compatibles avec les contraintes du domaine et la réglementation. Si l'on considère un ordre total $\succ$ sur les agents tel que $a_1 \succ \ldots \succ a_n$ (sans perte de généralité), alors une nouvelle position notée Γ est définie par $\Gamma = [\gamma_{a_1}, \ldots, \gamma_{a_n}]$.*

L'ensemble d'exigences obtenu en fusionnant $\Gamma_{a_1}, \ldots, \Gamma_{a_n}$ et en tenant compte de l'ordre $a_1 \succ \ldots \succ a_n$ est défini par γ associé à Γ.

Dans cette première stratégie, on calcule d'abord les ensembles d'exigences compatibles avec la réglementation et les contraintes du domaine associés à chaque position, puis on forme une nouvelle position avec ces ensembles en tenant compte de l'ordre sur les agents. Enfin, l'ensemble d'exigences obtenu en fusionnant les positions des agents est calculé avec cette nouvelle position.

Exemple. *Supposons que deux agents émettent des exigences à propos d'une maison à construire. Le premier agent, a_1, veut que la maison soit grande et ait des murs blancs. Le second agent, a_2, veut que la maison coûte moins de 100 000 euros, soit en centre ville et ait un jardin et des murs en brique. Les contraintes du domaine imposent que les grandes maisons coûtent plus de 100 000 euros et la réglementation que les maison en centre ville ne doivent pas avoir de murs blancs. On obtient :*

$$\Gamma_{a_1} = [big\_house, white\_walls]$$
$$\Gamma_{a_2} = [less\_100000, downtown, garden, brick\_walls]$$
$$Dom = \{big\_house \rightarrow \neg less\_100000\} \quad Reg = \{downtown \rightarrow \neg white\_walls\}$$

Si on considère que l'ordre sur les agents est $1 \succ 2$, alors l'application de la stratégie M1 conduit à l'ensemble d'exigences suivant : $\{big\_house, white\_walls, \neg(less\_100000 \wedge downtown \wedge garden \wedge brick\_walls)\}$. Si l'on considère maintenant que $2 \succ 1$, alors on obtient l'ensemble d'exigences suivant : $\{less\_100000, downtown, garden, brick\_walls, \neg(big\_house \wedge white\_walls)\}$.

6.3.2 Stratégie de fusion M2

Définition 6.3.2. *Soient n agents $a_1, \ldots, a_n$ et leurs positions associées $\Gamma_{a_1} = [\alpha_1^1, \ldots, \alpha_{k_1}^1]$, $\ldots, \Gamma_{a_n} = [\alpha_1^n, \ldots, \alpha_{k_n}^n]$. Si l'on considère un ordre total $\succ$ sur les agents tel que $a_1 \succ \ldots \succ a_n$ (sans perte de généralité), alors une nouvelle position notée Γ est définie par $\Gamma = [\alpha_1^1, \ldots, \alpha_{k_1}^1, \ldots, \alpha_1^n, \ldots, \alpha_{k_n}^n]$.*

L'ensemble d'exigences obtenu en fusionnant $\Gamma_{a_1}, \ldots, \Gamma_{a_n}$ et en tenant compte de l'ordre $a_1 \succ \ldots \succ a_n$ est défini par γ associé à Γ.

Dans cette seconde stratégie, on ne calcule pas l'ensemble d'exigences compatibles associé à chaque position et compatible avec les contraintes du domaine et la réglementation, mais on construit une nouvelle position à partir des positions de chaque agent.

Exemple. *Reprenons l'exemple précédent. Si l'on considère que $1 \succ 2$, alors on obtient une nouvelle position $\Gamma = [big\_house, white\_walls,$ $less\_100000, downtown, garden,$ $brick\_walls]$. Dans ce cas, on obtient un ensemble d'exigences $\{big\_house, white\_walls, \neg less\_100000,$ $\neg downtown, garden, brick\_walls\}$.*

Si on considère maintenant que $2 \succ 1$, on obtient $\{less\_100000, downtown, garden,$ $brick\_walls, \neg(big\_house \wedge white\_walls)\}$

Cette deuxième stratégie de fusion est plus fine que la première : dans le cas où $1 \succ 2$, avec M1 on éliminait toutes les exigences de a_2, alors qu'avec M2, on n'élimine que les exigences de a_2 qui sont incompatibles avec celles de a_1.

6.3.3 Conclusion

Cholvy et Hunter ont donc utilisé une méthode de fusion pour résoudre les conflits possibles entre les exigences de plusieurs agents. Cette méthode est basée sur un ordre d'importance entre les différents agents. Ils caractérisent deux stratégies de fusion différentes, une première qui ne tient compte que des ensembles d'exigences des agents, une deuxième qui tient compte des positions entières des agents. M2 permet d'obtenir un résultat plus « fin » que M1, mais est plus complexe au point de vue calculatoire. L'utilisation de cette technique permet d'obtenir un seul ensemble d'exigences cohérent, sans à avoir à faire appel a des actions interactives avec les agents comme dans les formalismes de Finkelstein et al. et de Hunter et Nusebeih.

6.4 Conclusion

La gestion des incohérences possibles entre les exigences des différents agent est un problème important du processus d'ingénierie des exigences. Puisque l'on doit obtenir un seul ensemble cohérent d'exigences à la fin du processus, on doit trouver un moyen d'éliminer ces incohérences.

Le formalisme développé par Finkelstein et al. permet, grâce à l'utilisation de la logique des prédicats, de se ramener à un problème d'incohérence dans des bases de données munies de contraintes d'intégrité. Hunter et Nusebeih proposent quant à eux un formalisme plus fin, qui permet de trouver et de qualifier les sources d'incohérences. On peut donc déduire des informations et savoir le degré de confiance que l'on peut accorder à ces informations. Ces approches reposent ensuite sur des logiques d'actions qui indiquent quelles actions réaliser en cas d'incohérence.

Le formalisme de Cholvy et Hunter fournit quant à lui un ensemble d'exigences cohérent « automatiquement ». L'utilisation d'une méthode de fusion permet d'obtenir un seul ensemble d'exigences. Bien sûr, cette fusion est contrainte par un ordre total sur les agents : en cas d'incohérence, il faut pouvoir choisir quel est l'agent prioritaire par rapport aux autres. Nous pensons donc que l'approche de Cholvy et Hunter est très intéressante dans le sens où l'on obtient tout de suite un ensemble d'exigences cohérent. Bien sûr, cet ensemble peut ne pas satisfaire certains agents, mais dans ce cas, on peut imaginer recommencer une phase d'expression des exigences pour que les agents puissent modifier leurs points de vue.

L'utilisation d'une méthode de fusion permet de trouver l'ensemble d'exigences à partir des positions de chacun des agents en tenant compte de critères particuliers. Le critère utilisé par Cholvy et Hunter (un ordre total sur les agents) permet de refléter la structure hiérarchique d'une organisation. Or, on ne dispose pas toujours de cet ordre total sur les agents. L'organisation d'une entreprise est certes souvent hiérarchique, mais il peut exister des personnes qui n'ont aucune relation de dominance entre elles. On ne peut donc pas classer « objectivement » ces personnes et l'utilisation d'une méthode de fusion ordonnée ne convient pas. Nous pensons donc que l'utilisation de techniques de fusion majoritaire (qui n'accordent pas plus d'importance à un agent qu'à un autre) permet de raffiner cette approche.

Chapitre 7

Fusion de bases de croyances (état de l'art)

Le problème de la fusion de bases de croyances[1] est un problème important dans les domaines des bases de données et de l'intelligence artificielle. En bases de données par exemple, on dispose de plus en plus souvent de plusieurs bases de données réparties sur plusieurs sites et souvent hétérogènes. Si le problème de l'hétérogénéité des bases de données a été souvent traité (cf. par exemple [128]), celui de l'incohérence entre données contradictoires a été très peu étudié. Il se peut en effet que deux bases de données contiennent l'une une information a et l'autre une information $\neg a$. La simple réunion de ces deux bases aboutit à une incohérence logique. Que peut-on alors déduire de ces deux bases ?

Le but de la fusion de bases de croyances est d'obtenir une base de croyances cohérente à partir de plusieurs bases de croyances. Ces bases peuvent bien entendu contenir des données contradictoires. Lorsque l'on veut fusionner plusieurs bases de croyances, il existe plusieurs méthodes :

- en autorisant l'incohérence, par exemple en utilisant des logiques paraconsistantes [45] ;
- en utilisant la technique des ensembles maximaux cohérents pour obtenir une base cohérente [8]. Malheureusement, cette approche ne traite pas de la rationalité de la fusion ;
- en disposant d'une méta information qui permet d'ordonner les bases de croyances (crédibilité, importance etc). Des travaux ont déjà été réalisés dans ce domaine (cf. [27, 28, 30, 31]) ;
- en privilégiant le groupe de bases de croyances, i.e. en essayant de satisfaire la majorité des bases. C'est ce qu'on appelle la fusion *majoritaire* (cf. [91, 92, 79]) ;
- en privilégiant chaque base de croyances, i.e. en essayant de satisfaire au mieux chacune des bases de croyances. C'est de la fusion par *arbitrage* (cf. [120, 89, 90, 79]) ;
- en utilisant un oracle, élément extérieur qui tranchera en cas de conflit [128, 127]. L'existence d'un tel oracle dans un système étant peu probable, il est intéressant de définir des méthodes permettant de traiter ces contradictions.

On peut également remarquer que Kaci [75] et Benferhat et al. [10] définissent des méthodes de fusion en logique possibiliste. Nous avons choisi dans la partie précédente de

[1] Dans la littérature, on parle souvent de bases de connaissances : à notre avis, ce terme est inexact, car une connaissance est toujours vraie dans le monde réel. Il ne peut donc pas y avoir de données contradictoires entre bases de connaissances.

ne pas utiliser la logique possibiliste pour représenter des préférences. Nous n'allons donc pas détailler ici l'approche proposée par Kaci. L'utilisation de préférences bipolaires dans ces travaux est à rapprocher de [86] : on peut exprimer dans une base ce qui est préféré, mais aussi ce qui est à rejeter.

Nous allons présenter trois méthodes de fusion : les méthodes utilisant une méta information, les méthodes de fusion majoritaire et les méthodes de fusion par arbitrage. Dans le premier cas, nous allons nous appuyer sur les travaux de Cholvy qui a développé une logique de fusion par priorité (utilisant un ordre sur les différentes sources) et dans les deux derniers cas, nous allons nous appuyer sur les travaux de Konieczny et Pino-Pérez qui fournissent des postulats pour la fusion majoritaire ou par arbitrage de bases de croyances.

7.1 Fusion prioritaire

Lorsque l'on veut fusionner des bases de croyances, on peut disposer d'une méta information importante : un ordre sur les différentes bases de croyances. Cet ordre peut représenter la plausibilité de chaque base par exemple, mais aussi être totalement artificiel. Ainsi, dans l'exemple présenté en section 6.3, l'ordre sur les trois personnes du centre de recherche était défini par le directeur du centre. Il considérait que le chercheur était plus important que l'ingénieur système par exemple.

Nous allons présenter dans cette section $FUSION$ une logique modale de fusion prioritaire développée par Cholvy [27, 28, 30, 31]. L'approche est la suivante : on dispose d'un ordre total sur les bases de croyances qui représente la crédibilité de chacune des bases. Plus une base est crédible, plus on va la croire. On va donc retrouver dans le résultat de la fusion le contenu de la base la plus crédible par exemple. De plus, dans [28] et [33], Cholvy et Demolombe ont montré qu'il était plus réaliste de considérer plusieurs ordres sur les bases plutôt qu'un seul. Il paraît en effet difficile de pouvoir dire qu'une base est plus crédible qu'une autre sur toutes les informations qu'elle contient. Par contre, elle peut être plus crédible sur un certain sujet. Il faut donc utiliser plusieurs ordres, suivant le sujet de l'information que l'on cherche à déduire, pour pouvoir raisonner correctement avec plusieurs bases de croyances. Enfin, Cholvy propose d'utiliser deux approches dans le cas de la fusion prioritaire : une approche « suspicieuse », dans laquelle toute information fournie par une base de croyances contredisant une base de croyances plus crédible est rejetée, et une approche « confiante », dans laquelle on ne rejette qu'un ensemble minimal d'informations provenant d'une base contredisant une base plus crédible.

Dans cette approche, Cholvy considère que les bases de croyances sont des ensembles cohérents de littéraux. Elle suppose de plus l'existence d'un ensemble de règles IDB commun à toutes les bases de croyances. IDB est telle que pour toute base de croyance db, $db \cup IDB$ est équivalent à un ensemble de littéraux. On ne considère donc pas ici des données disjonctives dans les bases de croyances.

7.1.1 Langage de $FUSION$

Tout d'abord, notons que si O dénote l'ordre total $db_1 > \ldots > db_n$ sur des bases de croyances $db_1, \ldots, db_n$ et si db est une autre base de croyances, alors $O > db$ dénote l'ordre $db_1 > \ldots > db_n > db$.

Soit $PROP$ un langage propositionnel. Le langage de $FUSION$, noté L', est obtenu en ajoutant à $PROP$ un nombre fini de modalités de la forme :

- $B_{exp_{db}}$ où db est une base de croyances. Cette modalité va être utilisée pour représenter les croyances de db ;
- B_O où O est un ordre total sur des bases de croyance. Dans le cas d'une seule base de croyances db, on obtient la modalité B_{db}. Dans le cas de n bases de croyances $db_1, \ldots, db_n$ ordonnées par $db_1 > \ldots > db_n$, on obtient la modalité $B_{db_1>\ldots>db_n}$.

$B_{exp_{db}}\varphi$ va donc signifier que la formule φ est crue explicitement dans la base db, donc qu'elle est déductible des données stockées dans db. $B_{db}\varphi$ signifie que φ est crue explicitement par la base db, donc que φ est dérivable des données stockées dans db et des règles. Enfin, $B_{db_1>\ldots>db_n}\varphi$ signifie que φ est déductible de la base obtenue en fusionnant $db_1, \ldots, db_n$ en supposant que db_1 est plus crédible que db_2, elle même plus crédible que db_3 etc.

Définition 7.1.1. *Si φ est une formule de $PROP$ et si $B_{exp_{db}}$ et B_O sont des modalités, alors $B_{exp_{db}}\varphi$ et $B_O\varphi$ sont des formules de L'. Si φ et ψ sont des formules de L', alors $\neg\varphi$ et $\varphi \wedge \psi$ sont des formules de L'. $\varphi \vee \psi$ et $\varphi \rightarrow \psi$ sont définies comme d'habitude.*

Exemple. *Si db_1 et db_2 sont les deux seules bases de croyances à fusionner, les modalités de L' sont : $B_{exp_{db_1}}$, $B_{exp_{db_2}}$, B_{db_1}, B_{db_2}, $B_{db_1>db_2}$ et $B_{db_2>db_1}$. Par exemple :*

- *$B_{exp_{db_1}}a$ signifie que le littéral a appartient à db_1 ;*
- *$B_{exp_{db_1}}a \vee b$ signifie que $a \vee b$ est déductible de db_1 ;*

7.1.2 Axiomatique de $FUSION$

Les schémas d'axiomes de $FUSION$ sont :

(A0) les schémas d'axiomes de la logique propositionnelle
(A1) $B_m\neg\varphi \rightarrow \neg B_m\varphi$ où B_m est une modalité B_{exp} ou B_O
(A2) $B_m\varphi \wedge B_m(\varphi \rightarrow \psi) \rightarrow B_m\psi$ où B_m est une modalité B_{exp} ou B_O
(A3) $B_O\varphi \rightarrow B_{O>db}\varphi$
(A4) $B_{exp_{db}}l \wedge \neg B_O\neg l \rightarrow B_{O>db}l$ si l est un littéral de $PROP$
(A5) $B_{O>db}l \rightarrow B_Ol \vee B_{exp_{db}}l$ si l est un littéral de $PROP$
(A6) $B_{exp_{db}}\varphi \rightarrow B_{db}\varphi$
(A7) $B_m(l_1 \vee \ldots \vee l_p) \rightarrow B_ml_1 \vee \ldots \vee B_ml_p$ où $l_1 \vee \ldots \vee l_p$ n'est pas une tautologie et où B_m est une modalité B_{exp} ou B_O

Si on appelle $\vdash_F$ la relation d'inférence de $FUSION$, les régles d'inférence de $FUSION$ sont :

(Nec) $\frac{\vdash_F\varphi}{\vdash_F B_m\varphi}$ où φ est une formule propositionnelle sans modalités et B_m est une modalité B_{exp} ou B_O
(MP) $\frac{\vdash_F\varphi\rightarrow\psi \quad \vdash_F\varphi}{\vdash_F B_m\psi}$

(A0), (A1) et (A2) sont les schémas d'axiomes des logiques modales de type (KD) (cf. [25]). (A3) signifie que si φ est déductible d'une base obtenue en fusionnant plusieurs bases ordonnées selon l'ordre O, alors φ est encore déductible si on fusionne une nouvelle base moins crédible. (A4) signifie que si un littéral l est explicitement présent dans db et que sa négation n'est pas déductible de la base obtenue en fusionnant plusieurs bases selon l'ordre O et les règles, alors l reste déductible si l'on fusionne db avec les autres bases en la considérant comme moins crédible. (A5) signifie que si l est un littéral déductible de la base résultant de la fusion de plusieurs bases et de db suivant un ordre $O > db$,

alors soit l est déductible des bases suivant O ou l apparaît explicitement dans db. (A6) signifie que l'information explicite est également déductible et (A7) signifie que les bases à fusionner sont des ensembles de littéraux et que le résultat de la fusion de plusieurs bases est également un littéral ou une conjonction de littéraux.

7.1.3 Sémantique de $FUSION$

La sémantique de $FUSION$ est donnée en terme de modèles de Kripke. Tout d'abord, on dit qu'un ensemble de mondes W représente une base de croyances ssi W est l'ensemble de tous les modèles d'une conjonction des littéraux de la base. Par exemple, $W_1 = \{\{a, b\}, \{a, \neg b\}\}$ représente une base de croyances, car c'est l'ensemble des modèles de $\{a\}$. $W_2 = \{\{a, b\}, \{\neg a, \neg b\}\}$ ne représente pas une base.

Définition 7.1.2. *Les modèles de $FUSION$ sont des tuples contenant :*
- *W, l'ensemble des mondes de $PROP$;*
- *autant de relations sur $W \times W$ notées $R_{exp_{db}}$ que de modalités $B_{exp_{db}}$;*
- *autant de relations sur $W \times W$ notées R_O que de modalités B_O ;*
- *une fonction de valuation val*

Cholvy impose ensuite des contraintes sur les relations $R_{exp_{db}}$ et $B_{exp_{db}}$ pour que la sémantique de $FUSION$ corresponde à l'axiomatique donnée précédemment. Nous ne détaillerons pas ici ces contraintes, ni la notion de satisfaction des formules dans $FUSION$. La relation de conséquence de $FUSION$ est notée $\models_F$.

7.1.4 Validité et complétude

La validité et la complétude de $FUSION$ est obtenue pour un ensemble de formules particuliers. Soient $db_1, \ldots, db_n$ n bases de croyances à fusionner. On définit la formule ψ de la façon suivante :

$$\psi = \bigwedge_{i=1}^{n} \Big(\bigwedge_{l \in db_i} B_{exp_{db_i}} l \wedge \bigwedge_{db_i \cup IDB \not\models c} \neg B_{db_i} c \Big) \bigwedge_{B_O \text{ est une modalité}} \Big(\bigwedge_{c \in IDB} B_O c \Big)$$

ψ va donc lister les informations contenues dans les bases $db_1, \ldots, db_n$. Elle liste :
- les informations stockées dans les bases de croyances : $\bigwedge_{i=1}^{n} \big(\bigwedge_{l \in db_i} B_{exp_{db_i}} l \big)$;
- les informations qui ne sont pas déductibles des bases et des règles : $\bigwedge_{i=1}^{n} \big(\bigwedge_{db_i \cup IDB \not\models c} \neg B_{db_i} c \big)$;
- les règles communes à toutes les bases : $\bigwedge_{B_O \text{ est une modalité}} \big(\bigwedge_{c \in IDB} B_O c \big)$

Le théorème de « validité-complétude » est le suivant :

Théorème 7.1.1. *Soient $db_1, \ldots, db_n$ n bases à fusionner, ψ la formule définie précédemment et O un ordre total sur un sous-ensemble de $db_1, \ldots, db_n$. Alors :*

$$\models_F \psi \rightarrow B_O \varphi \quad \Leftrightarrow \quad \vdash_F \psi \rightarrow B_O \varphi$$

Ce théorème montre l'équivalence entre l'axiomatique de $FUSION$ et sa sémantique sous certaines conditions. En particulier, pour toutes les bases db, $db \cup IDB$ doit être équivalent à un ensemble de littéraux.

7.1.5 Exemple

Considérons un langage propositionnel constitué de quatre variables a, b, c et d. Soient trois bases définies par $db_1 = \{a\}$, $db_2 = \{c\}$ et $db_3 = \{\neg b, d\}$ et un ensemble de règles $IDB = \{a \rightarrow b, c \rightarrow d, \neg b \vee \neg d\}$.

On peut alors prouver $\vdash_F \psi \rightarrow B_{db_1 > db_2 > db_3} a$. En effet, par définition de ψ, on a $\vdash_F \psi \rightarrow B_{exp_{db_1}} a$. D'après (A6) et (MP), on a $\vdash_F \psi \rightarrow B_{db_1} a$, puis par (A3) et (MP) (deux fois), on en déduit $\vdash_F \psi \rightarrow B_{db_1 > db_2 > db_3} a$. Cela signifie intuitivement que a est déductible du résultat de la fusion car il apparait explicitement dans la base considérée comme la plus crédible.

De la même façon, on peut déduire $\vdash_F \psi \rightarrow B_{db_1 > db_2 > db_3} \neg c$ et $\vdash_F \psi \rightarrow B_{db_3 > db_2 > db_1} c$.

7.1.6 Conclusion

Tout d'abord, notons que Cholvy étend son travail dans plusieurs directions (cf. [31]) :

- tout d'abord, elle montre que le travail réalisé pour des bases de croyances contenant des littéraux peut être étendu à des bases de données du premier ordre. Il faut pour cela faire plusieurs hypothèses : tout d'abord les bases ne contiennent que des littéraux comme $P(a)$ si P est un prédicat (mais non nécessairement monadiques), les clauses de IDB sont des clauses du premier ordre et on suppose que l'hypothèse des domaines clos (DMA pour *Domain Closure Assumption* cf. [118]) est vraie. Dans ce cas, les seuls individus du monde sont ceux contenus dans les bases de données. On peut alors utiliser la méthode de fusion par priorité présentée précédemment pour des bases de données du premier ordre ;
- puis, elle étend la logique à la notion de topique, qui correspond intuitivement à la notion de sujet. Les topiques sont des ensembles de littéraux tels que n'importe quel littéral du langage appartient à un topique, les topiques peuvent avoir des littéraux en commun et si un littéral appartient à un topique, sa négation appartient également à ce topique. Grâce à la notion de topique, elle peut raffiner l'ordre total sur les bases de croyances en exprimant un ordre par topique. Ceci permet ainsi de supposer qu'une base est plus crédible qu'une autre sur un sujet précis, mais pas sur tous les sujets par exemple ;
- enfin, Cholvy développe un démonstrateur de théorème pour la logique $FUSION$ (cf. [29] pour un démonstrateur pour $FUSION$ étendue aux topiques). Ce démonstrateur est implanté en PROLOG et permet d'automatiser des requètes à plusieurs bases de données que l'on veut fusionner grâce à un ordre total sur ces bases.

Cholvy a donc développé une logique de fusion prioritaire, comportant une sémantique et une axiomatique. Sa logique est valide et complète pour certaines formules (ce sont en fait les formules « intéressantes » car elles utilisent le contenu des bases de croyances) et elle a également développé un démonstrateur automatique permettant de faire des requètes sur le résultat de la fusion de plusieurs bases. On peut donc utiliser cette logique pour fusionner des bases de croyances si on a un ordre total sur les bases indiquant par exemple la crédibilité relative des bases.

7.2 Fusion majoritaire et par arbitrage

Nous allons maintenant nous intéresser à deux méthodes de fusion qui ne demandent pas de connaître une méta information concernant les bases de croyances.

Les premiers travaux traitant de la rationalité de la fusion de bases de croyances ont été effectués par Revesz [121, 120]. Il a défini dans ces articles des opérateurs de fusion dans le cadre de la révision AGM (cf. [1, 58]) qui propose un ensemble de propriétés qu'un « bon » opérateur de révision doit satisfaire. La révision est le processus qui consiste à introduire une nouvelle information dans une base de croyances. Les postulats AGM permettent de caractériser une classe d'opérateurs de révision qui ont un comportement intuitivement et logiquement correct. Les opérateurs définis par Revesz ont donc une caractérisation logique et un théorème de représentation dans la sémantique. Lin et Mendelzon ont quant à eux défini des opérateurs de fusion majoritaire [91, 92] qui permettent de satisfaire le groupe constitué des différentes bases dans son ensemble.

Plus récemment, Konieczny et Pino-Pérez ont proposé une caractérisation logique des opérateurs de fusion [78, 79, 76, 77]. En s'inspirant comme Revesz du cadre AGM, ils énoncent un certain nombre de postulats que doit respecter un opérateur de fusion pour être un « bon » opérateur. De plus, ils caractérisent deux classes d'opérateurs : les opérateurs d'arbitrage et les opérateurs majoritaires. La caractérisation donnée par Konieczny et Pino-Pérez est plus générale que celles données par Revesz ou Lin et Mendelzon et nous allons la présenter dans ce qui suit.

7.2.1 Préliminaires

Un multi-ensemble est un ensemble d'éléments dans lequel un élément peut être représenté plusieurs fois. Il est alors aisé de représenter la multiplicité d'un élément particulier.

Un multi-ensemble E contenant les éléments a une fois et b deux fois sera noté $E = [a, b, b]$. L'opérateur d'union $\sqcup$ est défini classiquement : par exemple, $[a, b] \sqcup [a, c] = [a, a, b, c]$.

Nous allons maintenant définir les notions de bases de croyances et d'ensemble de croyances.

Définition 7.2.1. *Une base de croyances est un ensemble cohérent de formules propositionnelles. Un ensemble de croyances est un multi-ensemble de bases de croyances.*

Soit Ψ un ensemble de croyances tel que $Psi = \{\varphi_1, \ldots, \varphi_n\}$, on note alors $\bigwedge \Psi$ la conjonction $\varphi_1 \wedge \ldots \wedge \varphi_n$. Si φ est une base de croyances et Ψ un ensemble de croyances, alors $\Psi \wedge \varphi \equiv_{def} \bigwedge \Psi \wedge \varphi$. De la même façon, $\bigvee \Psi \equiv_{def} \varphi_1 \vee \ldots \vee \varphi_n$. On note $\mathcal{B}$ l'ensemble des bases de croyances et $\mathcal{S}$ l'ensemble des ensembles de croyances.

Un ensemble de croyances Ψ est cohérent ssi $\bigwedge \Psi$ est cohérent. Si φ est une base de croyances, Ψ est cohérent avec φ ssi $\bigwedge \Psi \wedge \varphi$ est cohérent. Les modèles d'un ensemble de croyances Ψ sont définis par $Mod(\Psi) = Mod(\bigwedge \Psi)$. On note $I \models \Psi$ pour tout $I \in Mod(\Psi)$.

Définition 7.2.2. *Soient Ψ et Ψ' deux ensembles de croyances. On dit que Ψ et Ψ' sont équivalents, noté $\Psi \leftrightarrow \Psi'$ ssi il existe une bijection f de $\Psi = \{\varphi_1, \ldots, \varphi_n\}$ vers $\Psi' = \{\varphi'_1, \ldots, \varphi'_n\}$ telle que $f(\varphi) \leftrightarrow \varphi$.*

7.2.2 Caractérisation des opérateurs de fusion contrainte

Nous allons nous intéresser aux opérateurs de fusion contrainte. Ces opérateurs prennent en compte une base particulière qui représente les contraintes d'intégrité que doit respecter le résultat de la fusion des bases de croyances. Ainsi, il se peut très bien qu'une des bases de croyances ne respecte pas les contraintes d'intégrité. Il est à noter qu'une notion

très importante dans ces travaux est la notion d'*état épistémique*. On ne considère pas en effet simplement les propositions représentant la base de croyances d'un agent, mais également une stratégie de fusion, représentée par un préordre sur les interprétations (cf. section 7.2.3).

Les opérateurs de fusion contrainte sont des fonctions qui associent à un ensemble de croyances et une base de croyances une base de croyances. L'ensemble de croyances représente les bases à fusionner, la base de croyances représente les contraintes d'intégrité de la fusion. Si Δ est un opérateur de fusion contrainte, Ψ est l'ensemble de croyances à fusionner, μ les contraintes d'intégrité et φ le résultat de la fusion, alors on note $\Delta_\mu(\Psi) = \varphi$.

Nous allons maintenant énoncer les postulats que doivent respecter les opérateurs de fusion contrainte [77] :

Définition 7.2.3. *Soit Ψ un ensemble de croyances et μ une base de croyances. Δ est un opérateur de fusion contrainte ssi il satisfait les propriétés suivantes :*

(IC0) $\Delta_\mu(\Psi) \vdash \mu$
(IC1) *Si μ est cohérent, alors $\Delta_\mu(\Psi)$ est cohérent*
(IC2) *Si Ψ est cohérent avec μ, alors $\Delta_\mu(\Psi) = \bigwedge \Psi \wedge \mu$*
(IC3) *Si $\Psi_1 \leftrightarrow \Psi_2$ et $\mu_1 \leftrightarrow \mu_2$ alors $\Delta_{\mu_1}(\Psi_1) \leftrightarrow \Delta_{\mu_2}(\Psi_2)$*
(IC4) *si $\varphi \vdash \mu$ et $\varphi' \vdash \mu$, alors $\Delta_\mu(\{\varphi\} \sqcup \{\varphi'\}) \wedge \varphi \nvdash \bot \Rightarrow \Delta_\mu(\{\varphi\} \sqcup \{\varphi'\}) \wedge \varphi' \nvdash \bot$*
(IC5) $\Delta_\mu(\Psi_1) \wedge \Delta_\mu(\Psi_2) \vdash \Delta_\mu(\Psi_1 \sqcup \Psi_2)$
(IC6) *Si $\Delta_\mu(\Psi_1) \wedge \Delta_\mu(\Psi_2)$ est cohérent, alors $\Delta_\mu(\Psi_1 \sqcup \Psi_2) \vdash \Delta_\mu(\Psi_1) \wedge \Delta_\mu(\Psi_2)$*
(IC7) $\Delta_{\mu_1}(\Psi) \wedge \mu_2 \vdash \Delta_{\mu_1 \wedge \mu_2}(\Psi)$
(IC8) *Si $\Delta_{\mu_1}(\Psi) \wedge \mu_2$ est cohérent alors $\Delta_{\mu_1 \wedge \mu_2}(\Psi) \vdash \Delta_{\mu_1}(\Psi) \wedge \mu_2$*

La contrainte (IC0) stipule que le résultat de la fusion respecte les contraintes d'intégrité. (IC1) dit que si les contraintes d'intégrité sont cohérentes, alors le résultat de la fusion l'est aussi. (IC2) dit que si un ensemble de croyances Ψ est cohérent avec les contraintes d'intégrité μ, alors le résultat de la fusion est $\bigwedge \Psi \wedge \mu$. (IC3) stipule que si deux ensembles de croyances sont syntaxiquement équivalents et que deux bases de croyances sont équivalentes, les résultats des deux fusions sont équivalents. (IC4) assure que si l'on fusionne deux bases de croyances on ne donne pas priorité à l'une d'entre elles. (IC5) exprime l'idée suivante : si un groupe Ψ_1 se met d'accord sur un ensemble d'alternatives qui contient φ et si Ψ_2 se met également d'accord sur un ensemble d'alternatives qui contient φ, alors le résultat de la fusion de $\Psi_1 \sqcup \Psi_2$ contiendra également φ. (IC5) et (IC6) exprime le fait que si deux groupes sont d'accord sur au moins une alternative alors le résultat de la fusion de ces deux sous groupes sera exactement les alternatives sur lesquelles les sous groupes s'accordent. (IC7) et (IC8) sont des traductions des postulats (R5) et (R6) de la révision (cf. [1]) et permettent de s'assurer que les conditons sur les contraintes d'intégrité sont bien fondées.

On peut remarquer que la fusion contrainte est fortement lié à la théorie du choix social (cf. [4]). On peut se reporter à [76] pour une comparaison entre les postulats précédents et les propriétés énoncées par Arrow sur les règles de choix social. La fusion contrainte « échappe » au théorème d'impossibilité d'Arrow, car elle ne respecte pas la propriété d'indépendance des alternatives non-disponibles.

Konieczny et Pino-Pérez définissent ensuite deux sous classes d'opérateurs de fusion, les opérateurs de fusion majoritaires et les opérateurs d'arbitrage.

Définition 7.2.4. *Un opérateur de* fusion majoritaire *est un opérateur de fusion contrainte qui satisfait la propriété :*

(Maj) $\exists n\ \Delta_\mu(\Psi_1 \sqcup \Psi_2^n) \vdash \Delta_\mu(\Psi_2)$

*Un opérateur d'*arbitrage *est un opérateur de fusion contrainte qui respecte la propriété suivante :*

$$\textbf{(Arb)}\quad \left.\begin{array}{l} \Delta_{\mu_1}(\varphi_1) \leftrightarrow \Delta_{\mu_2}(\varphi_2) \\ \Delta_{\mu_1 \leftrightarrow \neg\mu_2}(\varphi_1 \sqcup \varphi_2) \leftrightarrow (\mu_1 \leftrightarrow \neg\mu_2) \\ \mu_1 \not\models \mu_2 \\ \mu_2 \not\models \mu_1 \end{array}\right\} \Rightarrow \Delta_{\mu_1 \vee \mu_2}(\varphi_1 \sqcup \varphi_2) \leftrightarrow \Delta_{\mu_1}(\varphi_1)$$

Les opérateurs de fusion majoritaire sont donc tels que si une opinion a une large audience, alors ce sera l'opinion du groupe. La propriété caractérisant les opérateurs d'arbitrage est difficile à expliquer intuitivement, mais nous allons voir qu'avec le théorème de représentation dans la sémantique elle est plus facile à saisir. Notons que beaucoup de personnes prennent une autre définition des opérateurs d'arbitrage : ils utilisent la propriété

(MI) $\forall n\ \Delta_\mu(\Psi_1 \sqcup \Psi_2^n) \leftrightarrow \Delta_\mu(\Psi_1 \sqcup \Psi_2)$

Cette propriété n'est pas compatible avec la définition d'un opérateur de fusion contrainte et ne correspond pas intuitivement à ce que l'on attend d'un opérateur d'arbitrage. Elle est appelée propriété d'indépendance de la majorité.

7.2.3 Théorèmes de représentation

Konieczny et Pino-Pérez ont également donné des théorèmes de représentation qui permettent de caractériser sémantiquement les opérateurs de fusion contrainte. À chaque opérateur de fusion contrainte correspond une famille de préordres sur les mondes possibles.

Définition 7.2.5. *Un* assignement syncrétique *est une fonction qui associe à chaque ensemble de croyances Ψ un préordre $\leq_\Psi$ sur les mondes possibles telle que pour tous les ensembles de croyances Ψ, Ψ_1, Ψ_2 et pour toutes les bases de croyances φ et φ' les conditions suivantes sont satisfaites*[2] *:*

1. *si $I \models \Psi$ et $J \models \Psi$, alors $I \simeq_\Psi J$*
2. *si $I \models \Psi$ et $J \not\models \Psi$, alors $I <_\Psi J$*
3. *si $\Psi_1 \leftrightarrow \Psi_2$, alors $\leq_{\Psi_1} = \leq_{\Psi_2}$*
4. *$\forall I$ telle que $I \models \varphi$ $\exists J$ $J \models \varphi'$ et $J \leq_{\varphi \sqcup \varphi'} I$*
5. *si $I \leq_{\Psi_1} J$ et $I \leq_{\Psi_2} J$ alors $I \leq_{\Psi_1 \sqcup \Psi_2} J$*
6. *si $I <_{\Psi_1} J$ et $I \leq_{\Psi_2} J$ alors $I \leq_{\Psi_1 \sqcup \Psi_2} J$*

Ils définissent ensuite les notions d'assignements syncrétiques majoritaire et juste qui correspondent aux opérateurs de fusion majoritaire et d'arbitrage.

Définition 7.2.6. *Un* assignement syncrétique majoritaire *est un assignement syncrétique qui satisfait la condition suivante :*

[2] $I \simeq_\Psi J$ signifie $I \leq_\Psi J$ et $J \leq_\Psi I$.

7. si $I <_{\Psi_2} J$ *alors* $\exists n\ I <_{\Psi_1 \sqcup \Psi_2^n} J$

Un assignement syncrétique juste *est un assignement syncrétique qui satisfait la condition suivante :*

$$8.\quad \left.\begin{array}{l} I <_{\varphi_1} J \\ I <_{\varphi_2} J' \\ J \simeq_{\varphi_1 \sqcup \varphi_2} J' \end{array}\right\} \Rightarrow I <_{\varphi_1 \sqcup \varphi_2} J$$

Les assignements syncrétiques justes sont donc des assignements qui « élisent » les choix médians des bases. Ils correspondent donc aux opérateurs d'arbitrage. Konieczny et Pino-Pérez prouvent ensuite la correspondance entre la caractérisation en logique des opérateurs de fusion contrainte et les assignements syncrétiques.

Théorème 7.2.1. *Un opérateur* Δ *est un opérateur de fusion majoritaire ssi il existe un assignement syncrétique majoritaire qui associe à chaque ensemble de croyances* Ψ *un préordre total* $\leq_\Psi$ *tel que*

$$Mod(\Delta_\mu(\Psi)) = min_{\leq_\Psi}(Mod(\mu))$$

Un opérateur Δ *est un opérateur d'arbitrage ssi il existe un assignement syncrétique juste qui associe à chaque ensemble de croyances* Ψ *un préordre total* $\leq_\Psi$ *tel que*

$$Mod(\Delta_\mu(\Psi)) = min_{\leq_\Psi}(Mod(\mu))$$

La caractérisation sémantique des opérateurs de fusion contrainte correspond à la définition logique donnée dans la section précédente. Nous allons maintenant présenter quelques opérateurs de fusion contrainte.

7.2.4 Les opérateurs Σ et GMax

Dans un premier temps, Konieczny et Pino-Pérez définissent la famille d'opérateur Σ qui est une famille d'opérateurs majoritaires.

Définition 7.2.7. *Soit un ensemble de croyances* Ψ *et une interprétation* I*. On définit la distance* Σ *entre une interprétation et un ensemble de croyances comme*

$$d_\Sigma(I, \Psi) = \sum_{\varphi \in \Psi} d(I, \varphi)$$

Cette distance induit le préordre suivant :

$$I \leq_\Psi^\Sigma J \text{ ssi } d_\Sigma(I, \Psi) \leq d_\Sigma(J, \Psi)$$

L'opérateur Δ^Σ *est défini par*

$$Mod(\Delta_\mu^\Sigma(\Psi)) = min_{\leq_\Psi^\Sigma}(Mod(\mu))$$

Pour calculer la distance d'une interprétation à un ensemble de croyances, on calcule donc la somme des distances de cette interprétation aux bases de croyances de l'ensemble.

Théorème 7.2.2. Δ^Σ *est un opérateur de fusion majoritaire.*

Ils définissent ensuite la famille d'opérateurs GMax qui sont des opérateurs d'arbitrage.

Définition 7.2.8. *Soit un ensemble de croyances $\Psi = \{\varphi_1, \ldots, \varphi_n\}$. Pour chaque interprétation I on construit la liste $(d_1^I, \ldots, d_n^I)$ des distances entre cette interprétation et les n bases de croyances de Ψ ($\forall j \in \{1, \ldots, n\}$ $d_j^I = d(I, \varphi_j)$). Soit L_I^Ψ la liste obtenue à partir de $(d_1^I, \ldots, d_n^I)$ en la triant dans l'ordre décroissant. On définit $d_{GMax}(I, \Psi) = L_I^\Psi$. Soit $\leq_{lex}$ l'ordre lexicographique entre les séquences d'entiers.*

On définit le préordre suivant :

$$I \leq_\Psi^{GMax} \text{ ssi } d_{GMax}(I, \Psi) \leq_{lex} d_{GMax}(J, \Psi)$$

On définit l'opérateur Δ^{GMax} par :

$$Mod(\Delta_\mu^{GMax}(\Psi)) = min_{\leq_\Psi^{GMax}}(Mod(\mu))$$

Théorème 7.2.3. *L'opérateur Δ^{GMax} est un opérateur d'arbitrage.*

7.2.5 Exemple

Nous allons illustrer le comportement de ces deux opérateurs en utilisant l'exemple donné dans [77]. Remarquons tout de suite qu'il faut choisir une distance entre les interprétations. Nous choisissons tout comme Konieczny et Pino-Pérez la distance de Hamming, reprise par Dalal [47]. Cette distance est basée sur le nombre de littéraux qui différent d'une interprétation à l'autre.

Définition 7.2.9. *Soit I et J deux interprétations. La distance de Hamming entre I et J est définie par le nombre de littéraux propositionnels l tels que $I \models l$ et $J \models \neg l$.*

L'exemple que nous allons présenter a été traité dans [120] et [77]. Un cours de bases de données est suivi par trois étudiants. Le professeur peut enseigner SQL, Datalog et O_2. Il demande l'avis de ses étudiants pour satisfaire la classe du mieux possible :

- le premier étudiant veut étudier SQL ou O_2 : $\varphi_1 = (S \vee O) \wedge \neg D$;
- le second étudiant veut étudier Datalog ou O_2 mais pas les deux : $\varphi_2 = (\neg S \wedge D \wedge \neg O) \vee (\neg S \wedge \neg D \wedge O)$;
- le troisième étudiant veut étudier les trois langages : $\varphi_3 = (S \wedge D \wedge O)$.

Si on ne considère que les trois variables propositionnelles S, D et O dans cet ordre, on a $Mod(\varphi_1) = \{(1,0,0), (0,0,1), (1,0,1)\}$, $Mod(\varphi_2) = \{(0,1,0), (0,0,1)\}$ et $Mod(\varphi_3) = \{(1,1,1)\}$. Le professeur n'impose aucune contrainte donc $\mu = \top$. Les distances nécessaires aux calculs pour $\Delta^\Sigma(\Psi)$ et $\Delta^{GMax}(\Psi)$ sont récapitulées dans le tableau ci dessous.

	φ_1	φ_2	φ_3	d_Σ	d_{GMax}
$(0,0,0)$	1	1	3	5	(3,1,1)
$(0,0,1)$	0	0	2	**2**	(2,0,0)
$(0,1,0)$	2	0	2	4	(2,2,2)
$(0,1,1)$	1	1	1	3	(1,1,1)
$(1,0,0)$	0	2	2	4	(2,2,0)
$(1,0,1)$	0	1	1	**2**	**(1,1,0)**
$(1,1,0)$	1	1	1	3	(1,1,1)
$(1,1,1)$	1	2	0	3	(2,1,0)

Le minimum de d_Σ est 2, on a donc $Mod(\Delta^\Sigma(\Psi)) = \{(0,0,1),(1,0,1)\}$. Le professeur doit donc enseigner SQL et O_2 ou O_2 seul pour satisfaire la classe dans son ensemble.

Le minimum de d_{GMax} est $(1,1,0)$ et on donc $Mod(\Delta^{GMax}(\Psi)) = \{(1,0,1)\}$. Le professeur doit donc enseigner SQL et O_2 pour satisfaire au mieux chacun des élèves.

7.2.6 Conclusion

Konieczny et Pino-Pérez ont donc proposé une caractérisation logique et sémantique des opérateurs de fusion contrainte. Cette caractérisation se rapproche des postulats AGM pour la révision. Ils dégagent ensuite deux familles d'opérateurs de fusion contrainte : les opérateurs de fusion majoritaire qui permettent de satisfaire les bases dans leur ensemble et les opérateurs d'arbitrage qui tendent à satisfaire au mieux chacune des bases. Il est à remarquer que dans d'autres articles comme [80], Konieczny et Pino-Pérez montrent qu'il existe une famille d'opérateurs de fusion dont le comportement tend à être à la fois celui d'un opérateur de fusion majoritaire et celui d'un opérateur d'arbitrage.

On peut toutefois noter que même si Konieczny et Pino-Pérez intitulent leur article [78] « On the logic of merging », on n'a qu'une méthode de calcul sur les modèles. En particulier, on est obligé de calculer toutes les distances pour pouvoir obtenir le résultat de la fusion, même si par exemple, savoir seulement si une proposition est dans le résultat de la fusion nous intéresse. Une axiomatique associée à la sémantique donnée précédemment permettrait de faciliter le calcul du résultat de la fusion de bases de croyances.

7.3 Conclusion

Dans le cadre de la gestion de l'incohérence en ingénierie des exigences, les méthodes de fusion permettent d'obtenir un seul ensemble d'exigences cohérent qui correspond à un consensus entre les différents participants. Bien sûr, les agents qui émettent les exigences ne sont pas consultés, puisque la fusion se fait sans renégociation des exigences, mais cette méthode permet d'avoir une idée de l'ensemble d'exigences final. Cet ensemble peut être présenté aux différents agents qui peuvent alors renégocier leurs exigences si besoin est.

La logique $FUSION$ développée par Cholvy a plusieurs avantages. Tout d'abord, elle dispose d'une axiomatique et d'un démonstrateur automatique associé qui permettent d'automatiser le calcul de la fusion. De plus, le principe de la fusion prioritaire permet de représenter la structure hiérarchique d'un ensemble d'agents émetteurs d'exigences.

On ne dispose pas toujours d'un ordre total sur les agents. En effet, même dans une structure comme une entreprise, certains agents sont hiérarchiquement au même niveau que d'autres. On ne peut donc pas faire entièrement appel à une méthode de fusion prioritaire pour obtenir l'ensemble d'exigences final. Il faut donc utiliser une méthode type majoritaire ou arbitrage. Là encore, la méthode majoritaire nous semble la plus appropriée : dans une structure décisionnelle comme celle d'une organisation émettant des exigences, on choisit les exigences à la majorité. On n'essaye pas de satisfaire au mieux chacun des agents. Nous allons donc utiliser les opérateurs de fusion majoritaires caractérisés par Konieczny et Pino-Pérez.

Chapitre 8

Une logique de fusion majoritaire

Dans ce chapitre, nous allons présenter MF, une logique (langage, sémantique et axiomatique) permettant de fusionner des bases de croyances de façon majoritaire. Comme nous l'avons vu dans le chapitre précédent, Konieczny et Pino-Pérez ne disposent pas d'un calcul pour leurs opérateurs de fusion. Notre objectif est donc de fournir ce calcul automatique. Nous montrons que la logique que nous avons développée correspond sémantiquement à un opérateur de fusion majoritaire. Nous implantons également un démonstrateur automatique en Prolog et nous prouvons que cette approche peut être étendue dans le cas de certaines bases de croyances du premier ordre. Ces travaux ont été présentés dans [60], [37] et [40].

8.1 La logique propositionnelle MF

8.1.1 Notations

Nous allons tout d'abord définir plusieurs notations que nous utiliserons dans le reste du chapitre.

Notation 8.1.1. *Soient un multi-ensemble E et e un élément de E.*
$e \in^i E \Leftrightarrow$ il y a i occurences de e dans E.
$e \in^0 E \Leftrightarrow e \notin E$

On notera $a \in E$ ssi $\exists i > 0$ tq $a \in^i E$.

Par exemple, on a $a \in^2 [a, a]$, mais on n'a pas $a \in^1 [a, a]$. Le nombre i dans $\in^i$ représente exactement le nombre d'occurences de l'élément dans le multi-ensemble.

Notation 8.1.2. *Si db et db' sont deux bases de croyances, alors $db * db'$ dénote le résultat de la fusion majoritaire de db et de db'. Par base de croyances nous entendons soit une base de croyances qui doit être fusionnée (dans ce cas, nous parlerons de bases de croyances* primitive*), soit d'une base résultant de la fusion de plusieurs autres bases de croyances.*

Exemple. *Si l'on considère trois bases de croyances primitives db_1, db_2 et db_3, alors $db_1 * db_2$ et $(db_1 * db_2) * db_3$ sont des bases de croyances mais ne sont pas primitives. $db_1 * db_2$ est le résultat de la fusion majoritaire de db_1 et de db_2. $(db_1 * db_2) * db_3$ est le résultat de la fusion majoritaire de $db_1 * db_2$ avec db_3.*

8.1.2 Langage de MF

Soit A un ensemble de variables propositionnelles. Soit $PROP$ le langage propositionnel construit à partir de A et utilisé pour représenter le contenu des bases de croyances. Soit DB un ensemble fini de bases de croyances primitives notées db_i avec $i \in \{1, \ldots, n\}$.

Nous construisons tout d'abord $DB*$ qui représente l'ensemble des bases de croyances *constructibles* à partir de DB.

Définition 8.1.1.

$$DB* = \{db_{k_1} * \ldots * db_{k_m} \ : \ \{k_1, \ldots, k_m\} \in 2^{\{1,\ldots,n\}}\}$$

Par exemple, si $DB = \{db_1, db_2\}$, alors $DB* = \{db_1, db_2, db_1 * db_2, db_2 * db_1\}$.

L'alphabet du langage $\mathcal{L}$ de MF est obtenu en étendant $PROP$ avec des opérateurs modaux de la forme B^i_{db} et B_{db}, où i est un entier positif et db est une base de croyances appartenant à $DB*$. On a donc une modalité B et une modalité B^i *par base de croyances constructible.*

Définition 8.1.2. *L'alphabet du langage $\mathcal{L}$ contient les ensembles suivants :*
- *l'alphabet propositionnel A ;*
- *un ensemble fini de modalités B_{db} où $db \in DB*$;*
- *un ensemble fini de modalités B^i_{db} où $db \in DB*$ et $i \in \{0, \ldots, n\}$;*
- *un ensemble de connecteurs logiques : $\{\neg, \vee, \wedge, \rightarrow, \leftrightarrow\}$.*

Le sens donné à une formule $B^i_{db}l$ est le suivant : il y a exactement i occurences du littéral l dans db. On remarquera que l'on a un nombre fini de modalités de type B^i_{db}, car nous imposons à i d'appartenir à $\{1, \ldots, n\}$. Nous verrons par suite que, comme nous considérons que les bases de données primitives représentent des ensembles finis de littéraux, nous n'avons pas besoin de considérer des modalités B^i_{db} avec $i > n$. $B_{db}\varphi$ signifie que la formule φ est crue par db.

Nous introduisons les modalités B^i_{db} pour pouvoir compter les occurences du littéral dans une base de croyances. L'idée intuitive est la suivante : quand on fusionne deux bases de croyances, le nombre d'occurences d'un littéral est la somme des occurences de ce littéral dans les deux bases. Puis, un littéral est cru par une base de croyances si le nombre d'occurences de ce littéral est strictement plus grand que le nombre d'occurences de sa négation.

La syntaxe de $\mathcal{L}$ est définie de la façon suivante :

Définition 8.1.3. *Soient φ une formule de $PROP$, B^i_{db} et B_{db} deux opérateurs modaux appartenant à l'alphabet de $\mathcal{L}$, alors :*
- *φ est une formule de $\mathcal{L}$;*
- *$B^i_{db}\varphi$ est une formule de $\mathcal{L}$;*
- *$B_{db}\varphi$ est une formule de $\mathcal{L}$.*

Si φ_1 et φ_2 sont deux formules de $\mathcal{L}$, alors $\neg\varphi_1$ et $\varphi_1 \wedge \varphi_2$ sont des formules de $\mathcal{L}$. $\varphi_1 \vee \varphi_2$, $\varphi_1 \rightarrow \varphi_2$ et $\varphi_1 \leftrightarrow \varphi_2$ sont définies commme habituellement.

On peut remarquer que les opérateurs modaux ne prennent en argument que des formules propositionnelles (sans modalités). On ne peut donc pas exprimer le fait par exemple qu'une base db_1 croit qu'une base db_2 croit que a est vrai.

Exemple. *Supposons que db_1 et db_2 soient les deux bases de croyances primitives qui doivent être fusionnées. Dans ce cas, les opérateurs modaux dont on a besoin sont de la forme $B^i_{db_1}$, B_{db_1}, $B^i_{db_2}$, B_{db_2}, $B^i_{db_1 * db_2}$, $B_{db_1 * db_2}$, $B^i_{db_2 * db_1}$ et $B_{db_2 * db_1}$ où $i \in \{0, 1, 2\}$.*

$B^1_{db_1} a$ signifie que db_1 contient une occurence de a. $B^0_{db_2} a$ signifie que db_2 ne contient pas d'occurence de a. $B^1_{db_1 * db_2} a$ signifie que la base de croyances obtenue en fusionnant db_1 et db_2 contient une occurence de a. Enfin, $B_{db_1 * db_2} a$ signifie que la base de croyances obtenue en fusionnant db_1 et db_2 croit a.

8.1.3 Sémantique de MF

La sémantique de MF est une sémantique de Kripke (cf. [25]). Les modèles sont définis dans ce qui suit.

Définition 8.1.4. *Un modèle de MF est un tuple $\langle W, val, R, B\rangle$ tel que*

- *W est un ensemble de mondes qui est l'ensemble des valuations propositionnelles ;*
- *val est une fonction de valuation qui associe chaque proposition de $PROP$ à un ensemble de mondes de W. Elle vérifie $val(\neg\varphi) = W - val(\varphi)$ et $val(\varphi_1 \wedge \varphi_2) = val(\varphi_1) \cap val(\varphi_2)$;*
- *R est un ensemble de fonctions notées f_{db} où db est une base de croyances appartenant à $DB*$. Chaque fonction f_{db} associe chaque monde de W à un multi-ensemble d'ensembles de mondes de W ;*
- *B est un ensemble de fonctions notées g_{db} où db est une base de croyances appartenant à $DB*$. Chaque fonction g_{db} associe chaque monde de W avec un ensemble de mondes de W.*

La relation R va être utilisée pour énoncer la relation de satisfaction d'une formule de type $B^i_{db}\varphi$. En effet, comme chaque f_{db} associe à chaque monde de W un multi-ensemble d'ensembles de mondes de W, on va pouvoir « compter » le nombre d'occurences d'un ensemble de mondes dans le multi-ensemble. La relation B va être utilisée pour énoncer la relation de satisfaction d'une formule de type $B_{db}\varphi$ (comme classiquement en logique modale).

Les ensembles de fonctions R et B sont contraints par deux contraintes que nous allons énoncer après avoir rappelé quelques définitions.

Définition 8.1.5. *Soient w et w' deux mondes de W. La distance $d(w, w')$ entre w et w' est définie comme le nombre de variables propositionnelles p telles que $w \in val(p)$ et $w' \notin val(p)$ (cette distance est habituellement appelée distance de Hamming).*

Soit $MS = [S_1, \ldots, S_n]$ un multi-ensemble d'ensembles de mondes tels que $\forall i \in \{1, \ldots, n\}$ $S_1 \neq \emptyset$. La distance $dsum(w, MS)$ entre un monde w et MS est définie par :

$$dsum(w, MS) = \sum_{i=1}^{n} \min_{w' \in S_i} d(w, w')$$

Enfin tout multi-ensemble d'ensembles de mondes de W MS est associé à un préordre $\leq_{MS}$ sur W défini par :

$$w \leq_{MS} w' \text{ ssi } dsum(w, MS) \leq dsum(w', MS)$$

Le préordre $\leq_{MS}$ va nous permettre de déterminer si un monde de W est plus proche (au sens de la distance de Hamming) du multi-ensemble MS qu'un autre monde. Dans le cas de la fusion, il va nous permettre de déterminer les mondes les plus proches de la « concaténation » des différentes bases à fusionner.

Nous allons maintenant énoncer les contraintes sur les modèles de MF.

Définition 8.1.6. *Soit $\langle W, val, R, B\rangle$ un modèle de MF. Alors ce modèle est contraint par :*

(C1) *si db et db' sont deux bases de croyances alors :*

$$\forall w \in W \quad f_{db*db'}(w) = f_{db}(w) \bigsqcup f_{db'}(w)$$

(C2) *si db est une base de croyances, alors $\forall w \in W \quad g_{db}(w) = \min_{\leq_{f_{db}(w)}} W$*

Si l'on considère que les bases de croyances primitives « sont » des ensembles finis et cohérents de littéraux, la contrainte **(C1)** reflète le fait que les occurences d'un littéral dans une base de croyances fusionnée $db * db'$ est l'union de ses occurences dans db et de ses occurences dans db'. Donc, le nombre d'occurences d'un littéral dans $db * db'$ est la sommme du nombre de ses occurences dans db et du nombre de ses occurences dans db'. Remarquons également que cette contrainte nous assure de la commutativité et de l'associativité de notre « opérateur » de fusion d'après la définition de $\sqcup$ (i.e. $f_{db*db'} = f_{db'*db}$ et $f_{(db*db')*db''} = f_{db*(db'*db'')}$).

La contrainte **(C2)** exprime le fait que les modèles d'une base de croyance qui est obtenue par cet opérateur de fusion majoritaire sont les mondes minimaux de W par rapport au préordre $\leq_{f_{db}(w)}$. Nous prouverons que cela correspond à un opérateur de fusion majoritaire défini dans [78].

Exemple. *Soient deux variables propositionnelles a et b et soient deux bases primitives db_1 et db_2. Supposons que $f_{db_1} = [\|a\|, \|\neg b\|]$ et $f_{db_2} = [\|a\|, \|b\|]$ (on considère des bases de croyances primitives qui ne contiennent que des littéraux pour simplifier l'exemple).*

Dans ce cas, $f_{db_1*db_2} = [\|a\|, \|a\|, \|b\|, \|\neg b\|]$, d'après la contrainte (C1). On peut considérer que le littéral a apparaît deux fois dans le résultat de la fusion de db_1 et de db_2 et que les littéraux b et $\neg b$ y apparaissent chacun une fois.

Pour calculer $g_{db_1*db_2}$, il nous faut déjà calculer le préordre $\leq_{f_{db_1*db_2}}$.

$$\begin{aligned} dsum(\{a,b\}, f_{db_1*db_2}) &= \min_{w\in\|a\|}(\{a,b\},w) + \min_{w\in\|a\|}(\{a,b\},w) + \min_{w\in\|b\|}(\{a,b\},w) + \\ &\quad \min_{w\in\|\neg b\|}(\{a,b\},w) \\ &= 0+0+0+1 \\ &= 1 \end{aligned}$$

De la même façon, on a :

$$\begin{aligned} dsum(\{a,\neg b\}, f_{db_1*db_2}) &= 1 \\ dsum(\{\neg a,b\}, f_{db_1*db_2}) &= 3 \\ dsum(\{\neg a,\neg b\}, f_{db_1*db_2}) &= 3 \end{aligned}$$

Donc $g_{db_1*db_2} = \{\{a,b\},\{a,\neg b\}\}$. Intuitivement, on obtient bien le résultat escompté lors de la fusion : la fusion des bases de croyances db_1 et db_2 « croit » a.

Nous définissons maintenant la notion de satisfaction d'une formule.

Définition 8.1.7. *Soit $M = \langle W, val, R, B\rangle$ un modèle de MF et soit $w \in W$. Soit p une variable propositionnelle de $PROP$. Soient φ, φ_1 et φ_2 des formules de $\mathcal{L}$ et db une base de croyances (primitive ou pas).*

$$
\begin{array}{lll}
M, w \models_{MF} p & ssi & w \in val(p) \\
M, w \models_{MF} \neg\varphi & ssi & M, w \not\models_{MF} \varphi \\
M, w \models_{MF} \varphi_1 \wedge \varphi_2 & ssi & M, w \models_{MF} \varphi_1 \quad et\ M, w \models_{MF} \varphi_2 \\
M, w \models_{MF} B^i_{db}\varphi & ssi & val(\varphi) \in^i f_{db}(w) \\
M, w \models_{MF} B_{db}\varphi & ssi & g_{db}(w) \subseteq val(\varphi)
\end{array}
$$

On remarquera que la satisfaction des formules de type B^i_{db} et B_{db} correspond bien à l'idée intuitive que l'on se fait de la fusion (cf. exemple précédent).

La notion de formule valide est définie classiquement :

Définition 8.1.8. *Soit φ une formule de $\mathcal{L}$. φ est une formule valide de MF ssi $\forall M$ modèle de MF, $\forall w \in W$, $M, w \models_{MF} \varphi$. On note alors $\models_{MF} \varphi$.*

8.1.4 Théorie de la preuve

Dans la suite de la section, db et db' sont des bases de croyances (primitives ou non), φ et φ' sont des formules de $PROP$, l, $l_1, \ldots, l_n$ sont des littéraux propositionnels et i, j, k sont des entiers.

Les schémas d'axiomes de MF sont définis ci après.

Schémas d'axiomes

(**A0**) Schémas d'axiomes de la logique propositionnelle
(**A1**) $B_{db}\neg\varphi \rightarrow \neg B_{db}\varphi$
(**A2**) $B_{db}\varphi \wedge B_{db}(\varphi \rightarrow \varphi') \rightarrow B_{db}\varphi'$
(**A3**) $B^i_{db}l \leftrightarrow B^i_{db}\neg\neg l$
(**A4**) $B^i_{db}l \rightarrow \neg B^j_{db}l$ si $i \neq j$
(**A5**) $B^i_{db}l \wedge B^j_{db'}l \rightarrow B^k_{db*db'}l$ si $k = i + j$
(**A6**) $B^i_{db}l \wedge B^j_{db}\neg l \rightarrow B_{db}l$ si $i > j$
(**A7**) $B^i_{db}l \wedge B^i_{db}\neg l \rightarrow \neg B_{db}l$
(**A8**) $B_{db}(l_1 \vee ... \vee l_n) \rightarrow B_{db}l_1 \vee ... \vee B_{db}l_n$ avec $\forall i \in \{1...n\}, \forall j \in \{1...n\}\ \ l_i \neq \neg l_j$

Les opérateurs modaux B_{db} sont des opérateurs modaux de croyances et sont donc gouvernés par des axiomes de type (KD), i.e. (A_0), (A_1) et (A_2).

(A_3) permet de retrouver l'équivalence entre l et $\neg\neg l$ dans une modalité B^i_{db}.

(A_4) dit que le nombre d'occurences d'un littéral dans une base de croyances est unique.

(A_5) exprime le fait que le nombre d'ocurrences d'un littéral dans la base de croyances fusionnée $db * db'$ est la somme du nombre de ses occurences dans db et du nombre de ses occurences dans db'.

(A_6) et (A_7) expriment l'aspect majoritaire de l'opérateur de fusion sous jacent. Tout d'abord, un littéral l est cru dans une base de croyances db si le nombre de ses occurences est strictement plus grand que le nombre des occurences de sa négation. Si le nombre

d'occurences de l est égal au nombre des occurences de sa négation, alors le littéral et sa négation ne sont pas crus par la base de croyances.

(A_8) restreint les bases de croyances que nous considérons à des ensembles de littéraux.

Les régles d'inférence de MF sont :

Règles d'inférences

(**MP**) $\frac{\vdash_{MF}\varphi \quad \vdash_{MF}(\varphi\rightarrow\varphi')}{\vdash_{MF}\varphi'}$

(**Nec**) $\frac{\vdash_{MF}\varphi}{\vdash_{MF}B_{db}\varphi}$ pour toute modalité B_{db}

$\vdash_{MF} \varphi$ signifie que φ est un théorème de MF, i.e. une formule qui est soit une instance d'un schéma d'axiome ou qui peut être déduite en utilisant les schémas d'axiomes et les règles d'inférence.

8.1.5 Validité et complétude

Dans un premier temps, nous allons nous intéresser à la validité et la complétude d'un fragment de la logique MF. En effet, nous allons considérer que les bases de croyances primitives sont des ensembles finis et cohérents de littéraux et nous allons nous intéresser aux formules que l'on peut déduire d'une représentation des bases primitives. Nous reviendrons sur cette restriction lors de la conclusion générale.

Définition 8.1.9. *Soient $DB_1, \ldots, DB_n$ n ensembles finis de littéraux à fusionner tels que chaque DB_i est cohérent. Soient $db_1, \ldots, db_n$ les bases primitives du langage de MF qui les représentent. On définit la formule ψ par :*

$$\psi = \bigwedge_{i=1}^{n}(\bigwedge_{l\in DB_i} B^1_{db_i} l \wedge \bigwedge_{l\notin DB_i} B^0_{db_i} l)$$

ψ liste les informations que nous avons à propos du contenu des bases de croyances à fusionner. Plus précisemment, pour une base de croyance db, ψ exprime que chaque littéral de DB a une et une seule occurence et que chaque littéral qui n'est pas contenu dans DB n'a pas d'occurences. Par abus de langage, on dira également que db est cohérente.

Le résultat suivant prouve que la théorie des modèles et la théorie de la preuve présentées précédemment sont equivalentes pour des formules du type $\psi \rightarrow B_{db}\varphi$ où db est une base de croyances.

Théorème 8.1.1. *Soit ψ la formule définie en définition 8.1.9. Soit φ une formule de $PROP$ et db une base de croyances (primitive ou pas). On a alors :*

$$\models_{MF} \psi \rightarrow B_{db}\varphi \iff \vdash_{MF} \psi \rightarrow B_{db}\varphi$$

$$\models_{MF} \psi \rightarrow \neg B_{db}\varphi \iff \vdash_{MF} \psi \rightarrow \neg B_{db}\varphi$$

Théorème 8.1.2. *Soit ψ la formule définie en définition 8.1.9. Soit φ une formule de $PROP$ et db une base de croyances (primitive ou pas). On a alors :*

$$\not\models_{MF} \psi \rightarrow B_{db}\varphi \iff \vdash_{MF} \psi \rightarrow \neg B_{db}\varphi$$

Ce théorème est une sorte de « complètude » syntaxique pour ψ. Il va nous être utile lorsque nous chercherons à implanter un évaluateur de requète pour MF.

8.1.6 Exemple

Nous donnons dans cette section quelques exemple de preuves dans MF.

Nous considérons trois bases de croyances : $db_1 = \{a, b\}$, $db_2 = \{a, \neg c\}$, $db_3 = \{\neg a, c\}$. D'après la définition 8.1.9, ψ vaut :

$$B^1_{db_1} a \wedge B^1_{db_1} b \wedge B^0_{db_1} c \wedge B^0_{db_1} \neg c \wedge B^0_{db_1} \neg a \wedge B^0_{db_1} \neg b \wedge B^1_{db_2} a \wedge B^1_{db_2} \neg c \wedge B^0_{db_2} b \wedge B^0_{db_2} \neg b \wedge B^0_{db_2} \neg a \wedge B^0_{db_2} c \wedge B^1_{db_3} \neg a \wedge B^1_{db_3} c \wedge B^0_{db_3} b \wedge B^0_{db_3} \neg b \wedge B^0_{db_3} a \wedge B^0_{db_3} \neg c$$

Voici quelques théorèmes de MF que l'on peut dériver :

(α) $\vdash \psi \rightarrow B^2_{db_1 * db_2} a$ (d'après (A5))
(β) $\vdash \psi \rightarrow B^2_{(db_1 * db_2) * db_3} a$ (d'après (α) et (A5))
(γ) $\vdash \psi \rightarrow B^0_{db_1 * db_2} \neg a$ (d'après (A5))
(δ) $\vdash \psi \rightarrow B^1_{(db_1 * db_2) * db_3} \neg a$ (d'après (γ) et (A5))

Enfin, d'après (β), (δ) et (A6), on peut prouver :

(ζ) $\vdash \psi \rightarrow B_{(db_1 * db_2) * db_3} a$

Ce théorème signifie que la base de croyances obtenue en fusionnant db_1, db_2 et db_3 croit a. Remarquons que cela illustre une attitude majoritaire, car deux bases de croyances primitives croient a alors qu'une seule croit $\neg a$.

De la même façon, on prouve :

(η) $\vdash \psi \rightarrow B_{(db_1 * db_2) * db_3} b$

Donc, d'après (ζ), (η), (A0) et (A2) on prouve :

$\vdash \psi \rightarrow B_{(db_1 * db_2) * db_3} (a \wedge b)$

Ce théorème signifie que la base obtenue en fusionnant db_1, db_2 et db_3 croit $(a \wedge b)$.

On a aussi :

$\theta : \vdash_{MF} \psi \rightarrow B^0_{db_1 * db_2} c$ (d'après (A5))
$\iota : \vdash_{MF} \psi \rightarrow B^1_{(db_1 * db_2) * db_3} c$ (d'après (θ) et (A5))
$\kappa : \vdash_{MF} \psi \rightarrow B^1_{db_1 * db_2} \neg c$ (d'après (A5))
$\lambda : \vdash_{MF} \psi \rightarrow B^1_{(db_1 * db_2) * db_3} \neg c$ (d'après (θ) et (A5))
$\mu : \vdash_{MF} \psi \rightarrow \neg B_{(db_1 * db_2) * db_3} c$ (d'après (ι), (λ) et (A7))
$\nu : \vdash_{MF} \psi \rightarrow \neg B_{(db_1 * db_2) * db_3} \neg c$ (d'après (ι), (λ) et (A7))

Enfin, d'après (μ), (ν) et (A0) on peut prouver :

$\vdash_{MF} \psi \rightarrow \neg B_{(db_1 * db_2) * db_3} c \wedge \neg B_{(db_1 * db_2) * db_3} \neg c$

Ce théorème signifie que la base obtenue en fusionnant db_1, db_2 et db_3 ne croit ni c ni $\neg c$.

8.2 Relation avec les travaux de Konieczny et Pino-Pérez

Dans cette section, nous prouvons formellement que la logique MF permet de raisonner avec des données fusionnées obtenues par un opérateur de fusion majoritaire. Plus précisement, nous nous concentrons sur un opérateur de fusion défini par Konieczny et Pino-Pérez et nous établissons une relation entre certains théorèmes de MF et la base de croyances obtenues par cet opérateur.

Tout d'abord, nous rappelons la définition introduire par Konieczny et Pino-Pérez [1] (cf. section 7.2).

Soient $db_1, \ldots, db_n$ n bases de croyances à fusionner.

Konieczny et Pino-Pérez définissent un opérateur de fusion majoritaire, noté Δ_Σ, tel que les modèles de la base de croyances obtenue en fusionnant $db_1, \ldots, db_n$ avec cet opérateur sont caractériés sémantiquement par :

$$Mod(\Delta_\Sigma([db_1, ..., db_n])) = \min_{\leq^\Sigma_{[db_1 \ldots db_n]}} (W)$$

où W représente l'ensemble de toutes les interprétations du langage $PROP$ (le langage propositionnel utilisé pour décrire le contenu des bases de croyances). $\leq^\Sigma_{[db_1 \ldots db_n]}$ est un préordre total sur W défini par :

$$w \leq^\Sigma_{[db_1 \ldots db_n]} w' \text{ ssi } d_\Sigma(w, [db_1 ... db_n]) \leq d_\Sigma(w', [db_1 ... db_n])$$

avec :

$$d_\Sigma(w, [db_1 ... db_n]) = \sum_{i=1}^{n} \min_{w' \in Mod(db_i)} d(w, w')$$

où $Mod(db_i)$ est l'ensemble des modèles de db_i et $d(w, w')$ est la distance de Hamming.

En d'autres termes, quand on fusionne $db_1, \ldots, db_n$ avec l'opérateur Δ_Σ, le résultat est caractérisé sémantiquement par les interprétations qui sont minimales par rapport au préordre $\leq^\Sigma_{[db_1, \ldots, db_n]}$.

Le théorème suivant établit la relation entre certains théorèmes de la logique MF et le résultat de la fusion effectuée par cet opérateur.

Théorème 8.2.1. *Soient $db_1, \ldots, db_n$ n ensembles de littéraux finis et cohérents à fusionner et φ une formule de PROP. Avec les notations introduites précédemment, on a :*

$$\vdash \psi \rightarrow B_{(\ldots(db_1 * db_2) * \ldots db_n)} \varphi \Longleftrightarrow \Delta_\Sigma([db_1, \ldots, db_n]) \models \varphi$$

La base dont les croyances sont caractérisées par les théorèmes $\vdash \psi \rightarrow B_{(\ldots(db_1 * db_2) * \ldots db_n)} \varphi$ est équivalente à $\Delta_\Sigma([db_1, \ldots, db_n])$.

Cela prouve que la logique MF est une logique pour raisonner avec des croyances fusionnées obtenues par un opérateur de fusion majoritaire quand les bases de croyances sont des ensembles de littéraux.

[1] Nous changeons les notations et la présentation des définitions pour rester cohérent avec ce qui a été présenté.

8.3 Déduction automatique dans MF

Dans cette section, nous nous intéressons à des aspects d'implantation. Nous présentons un démonstrateur de théorème pour la logique MF. Il nous permet de poser des questions de la forme : « étant donnée la description du contenu des bases de croyances, est ce que la formule φ est déductible du résultat de la fusion de ces bases ? ». Il nous permet donc de prouver des théorèmes du type $\psi \rightarrow B_{db}\varphi$. Remarquons que ce démonstrateur de théorème a été implanté dans un PROLOG écrit en LISP.

On remarquera également que dans ce démonstrateur, la formule ψ introduite dans la définition 8.1.9 n'est pas utilisée. En fait, ψ a été introduite pour des raisons théoriques. Sa raison d'être était de décrire ce qui est cru et ce qui n'est pas cru dans les bases de croyances primitives. Dans le démonstrateur, nous avons juste besoin de lister les croyances explicites des bases. Les propositions qui ne sont pas crues seront dérivées grâce à la négation par échec.

8.3.1 Le méta-langage

Considérons un méta-langage ML, basé sur le langage $PROP$ et défini par :
- les symboles de constantes de ML sont les lettres propositionnelles de $PROP$, les noms des bases de croyances et un symbole de constante noté nil et des constantes pour les entiers : 1, 2, etc ;
- une fonction binaire notée $*$. Par convention, $(db_{i_1} * \ldots * db_{i_k})$ représente le terme $db_{i_1} * (db_{i_2} \ldots * (db_{i_k} * nil) \ldots)$. Cette fonction va être utilisée pour nommer les bases de croyances obtenues en fusionnant les bases de croyances $db_{i_1}, \ldots, db_{i_k}$. Remarquons que toutes les bases se « terminent » par nil ;
- une fonction binaire notée + qui est la fonction qui calcule la somme d'entiers ;
- cinq méta-prédicats binaires : B_{exp}, B, neg, $=$ et $>$;
- un méta-prédicat ternaire R ;
- un méta-prédicat unaire NIL.

La sémantique intuitive des prédicats est la suivante :

- $neg(l, l')$ est vrai si le littéral l' est la négation du littéral l. Ce prédicat va être utilisé pour représenter la négation au niveau des objets ;
- $B_{exp}(db, l)$ est vrai si le littéral l est explicitement cru par la base de croyances primitive db ;
- $R(db, l, i)$ est vrai si l apparaît i fois dans la base de croyances db ;
- $B(db, l)$ est vrai si la base de croyances db croit l ;
- $NIL(db)$ est vrai si db est nil;
- $i = j$ (respectivement $(i > j)$) est vrai si les entiers i et j sont égaux (respectivement si l'entier i est strictement plus grand que l'entier j). Ces deux prédicats seront définis en extension dans le méta-programme par un nombre fini de faits.

8.3.2 Le méta-programme

Si il y a n bases de croyances à fusionner $db_1, \ldots, db_n$, alors on définit META comme étant l'ensemble suivant de formules de ML :

(1) $B_{exp}(db, l)$ si le littéral l appartient à la base de croyances primitive db

(2) $\neg NIL(db_2) \wedge R(db_1, l, i) \wedge R(db_2, l, j) \wedge (k = i + j) \rightarrow R(db_1 * db_2, l, k)$
(3) $NIL(db_2) \wedge B_{exp}(db_1, l) \rightarrow R(db_1 * db_2, l, 1)$
(4) $NIL(db_2) \wedge \neg B_{exp}(db_1, l) \rightarrow R(db_1 * db_2, l, 0)$
(5) $R(db, l, i) \wedge neg(l, l') \wedge R(db, l', j) \wedge (i > j) \rightarrow B(db, l)$
(6) $NIL(nil)$
(7) $k = (r + l)$ et $(r + l) = k$ pour tout $k \in \{1, \ldots, n\}$ pour tout $r \in \{1, \ldots, k\}$ et pour tout l tel que $l = k - r$
(8) $k > r$ pour tout $k \in \{1, \ldots, n\}$ et pour tout $r \in \{1, \ldots, k - 1\}$
(9) $neg(l, l')$ où l est un littéral positif ou négatif du sous-langage de ML utilisé pour décrire le contenu des bases et l' sa négation

L'axiome (1) permet de lister les littéraux contenus explicitement dans une base de croyances primitive. L'axiome (2) dit que si db_2 et db_1 sont deux bases telles que db_2 n'est pas la base vide (nil), alors si l apparaît i fois dans db_1 et j fois dans db_2, il apparaît exactement $i + j$ fois dans le résultat de la fusion de db_1 et db_2. Si la base db_2 était nil, alors l'utilisation de cet axiome n'aurait aucun sens, car on ajoute le nombre d'occurences du littéral dans chacune des bases. L'axiome (3) dit que si db_2 est la base nulle, alors si l est contenu explicitement dans la base primitive db_1, il apparaît une seule fois dans le résultat de la fusion de db_1 et de db_2[2] . L'axiome (4) dit que si db_2 est la base nulle, alors si l n'est pas contenu explicitement dans la base primitive db_1, il n'apparaît pas dans le résultat de la fusion de db_1 et de db_2. L'axiome (5) montre le caractère majoritaire de la fusion : si l apparaît i fois dans db, $\neg l$ apparaît j fois dans db et que $i > j$ alors l est cru par la base db. L'axiome (6) dit que la base nil est une base de croyances nulle.

Remarquons qu'il y a un nombre fini d'axiomes (7) et (8). On ne considère le calcul sur les entiers que dans les cas qui nous intéressent : en particulier, le nombre maximum d'occurences dans une base de croyances d'un littéral est égal au nombre de bases de croyances primitives.

De la même façon, il y a un nombre fini d'axiome (9).

Le résultat suivant assure que le méta-programme est correct.

Propriété 8.3.1. *Soit l un littéral, soit db une base de croyances primitive ou pas. Dans ce cas, en utilisant la négation par échec sur le méta-programme META,*
(1) PROLOG réussit à prouver $B(db, l)$ ssi $\models_{MF} (\psi \rightarrow B_{db} l)$
(2) PROLOG échoue à prouver $B(db, l)$ ssi $\models_{MF} (\psi \rightarrow \neg B_{db}\ l)$

8.4 Spécification d'un évaluateur de requêtes

Dans cette section, nous appliquons les résultats précédents pour spécifier un évaluateur de requête qui répond à des requêtes adressées à plusieurs bases de données. Pour pouvoir utiliser les résultats précédents, nous ne considérons que les bases de données « équivalentes à un ensemble de littéraux de base[3] ». Nous verrons dans le chapitre suivant que cette approche va nous permettre de fusionner des positions émises par des agents.

[2]Les bases de croyances primitives représentent des ensembles de littéraux, donc chaque littéral qui « appartient » à une base a exactement une occurrence dans cette base.

[3]Un littéral de base est un littéral du type $P(a)$ où P est un prédicat et a un symbole de constante. Remarquons que l'on peut également travailler avec des prédicats d'arité supérieure à 1.

8.4.1 Bases de données équivalentes à un ensemble de littéraux de base

Soit *LO* un langage du premier ordre.

Définition 8.4.1. *Une base de données est une paire $DB = \langle EDB, IDB\rangle$ telle que EDB est un ensemble non vide et fini de littéraux de base positifs ou négatifs de LO et IDB est un ensemble fini et cohérent de clauses de LO sans symbole de fonction.*

On peut noter que les littéraux de EDB peuvent être positifs ou négatifs. IDB représente les contraintes d'intégrité de la base. Chaque base peut donc avoir ses propres contraintes d'intégrité.

Définition 8.4.2. *Soit $DB = \langle EDB, IDB\rangle$ une base de données. Soient $a_1, \ldots, a_n$ (respectivement $P_1, \ldots, P_k$) les symboles de constantes (respectivement les symboles de prédicats) qui apparaissent dans les formules de $EDB \cup IDB$. La base de Herbrand est l'ensemble de littéraux positifs écrits avec les P_i et les a_j.*

Une interprétation de Herbrand de DB est une interprétation dont le domaine est $\{a_1, ..., a_n\}$.

Un modèle de Herbrand de DB est une interprétation de Herbrand qui statisfait $EDB \cup IDB$.

Définition 8.4.3. *Soient $HM_1, \ldots, HM_n$ les modèles de Herbrand de $EDB \cup IDB$.*

Soit $L = \{l : l$ est un littéral de la base de Herbrand tel que $\exists\ HM_i\ \exists\ HM_j$ $HM_i \models l$ et $HM_j \models \neg l\}$

La base de données $DB = \langle EDB, IDB\rangle$ est équivalente à un ensemble de littéraux de base ssi pour toute conjonction satisfaisable $l_1 \wedge ... \wedge l_m$ où $\forall i \in \{1, \ldots, m\}$ tel que $l_i \in L$ ou $\neg l_i \in L$, il existe $i_0 \in \{1, \ldots, m\}$ tel que $HM_{i_0} \models l_1 \wedge ... \wedge l_m$.

Exemple. *Considérons $DB_1 = \langle EDB_1, IDB_1\rangle$ avec $EDB_1 = \{p(a)\}$ et $IDB_1 = \{\neg p(x) \vee \neg q(x), p(x) \vee r(x)\}$. Les modèles de Herbrand de DB_1 sont[4] : $\{p(a)\}$ et $\{p(a), r(a)\}$. On a $L = \{r(a)\}$. On peut vérifier que $\neg r(a)$ est satisfait dans le premier modèle de Herbrand et que $r(a)$ est satisfait dans le second. Donc DB_1 est équivalente à un ensemble de littéraux.*

Considérons maintenant $DB_2 = \langle EDB_2, IDB_2\rangle$ avec $EDB_2 = \{p(a)\}$ et $IDB_2 = \{\neg p(x) \vee q(x) \vee r(x)\}$. Les modèles de Herbrand de DB_2 sont $\{p(a), q(a)\}$, $\{p(a), r(a)\}$ et $\{p(a), q(a), r(a)\}$. On a $L = \{r(a), q(a)\}$. On peut vérifier qu'aucun de ces modèles ne satisfont $\neg q(a) \wedge \neg r(a)$. Donc DB_2 n'est pas équivalente à un ensemble de littéraux de base.

Propriété 8.4.1. *Soit $DB = \langle EDB, IDB\rangle$ une base de données qui est équivalente à un ensemble de littéraux de base. Soient $l_1, \ldots, l_n$ des littéraux de base de LO tels que $l_1 \vee ... \vee l_n$ n'est pas une tautologie. Alors,*

$EDB \cup IDB \models l_1 \ \vee \ ... \ \vee \ l_n$ ssi $\exists i_0 \in \{1...n\}\ \ EDB \cup IDB \models l_{i_0}$

Ce résultat assure que, dans une base de données équivalente à un ensemble de littéraux de base, une disjonction de littéraux de base qui n'est pas une tautologie est dérivable de la base de données ssi un de ces littéraux est dérivable de la base de données. Cela implique qu'il n'y a pas de vraies données disjonctives dérivables de ces bases de données.

[4]Un modèle est dénoté par l'ensemble de ses faits positifs.

8.4.2 Spécification d'un évaluateur de requêtes

Dans cette section, nous utilisons le méta-programme défini dans la section précédente pour spécifier un évaluateur de requête qui répond à des requêtes adressées à plusieurs bases de données. Les réponses calculées par l'évaluateur sont les mêmes que celles qui seraient calculées par un évaluateur classique quand la requête est adressée à une base de données obtenue en fusionnant plusieurs base de données avec une attitude majoritaire. Cependant, on remarquera que la fusion des bases de données n'est jamais calculée.

Le méta-programme défini dans la section précédente suppose que les bases de croyances sont des ensembles de littéraux propositionnels positifs ou négatifs. Considérer que les bases de données qui sont équivalentes à un ensemble de littéraux de base nous permet de réutiliser le méta-programme : chaque littéral de base sera considéré comme un littéral propositionnel. Cependant, nous devons étendre le méta-programme pour prendre en compte les clauses de IDB.

Extension du méta-progamme pour prendre IDB en compte

Nous notons h la fonction qui associe chaque clause de IDB avec un ensemble de formules de la façon suivante :
$h(l_1 \vee \ldots \vee l_n) = \{(\neg l_1 \wedge \ldots \neg l_{i-1} \wedge \neg l_{i+1} \ldots \wedge \neg l_n) \rightarrow l_i, \quad i \in \{1, \ldots, n\}\}$

Dans ce cas, l'axiome (1) du méta-programme est remplacé par les axiomes suivants :

(1.1) $EDB(db, l)$ si le littéral de base l appartient à la partie EDB de la base de donnée db
(1.2) $IDB(db, f)$ si la formule f est dans $h(c)$, où c est une clause de la partie IDB de db
(1.3) $EDB(db, l) \rightarrow B_{exp}(db, l)$
(1.4) $IDB(db, (r \rightarrow l)) \wedge B_{conj}(db, r) \rightarrow B_{exp}(db, l)$
(1.5) $B_{conj}(db, nil)$
(1.6) $B_{exp}(db, l_1) \wedge B_{conj}(db, r_1) \rightarrow B_{conj}(db, l_1 \wedge r_1)$

Ces axiomes permettent de dériver l'ensemble des conséquences de chaque base en tenant compte des contraintes d'intégrité.

Propriété 8.4.2. *Soit $db = \langle EDB, IDB \rangle$ une base de données telle que IDB n'est pas récursive. Soit l un littéral de base. Alors PROLOG prouve $B_{exp}(db, l)$ ssi $EDB \cup IDB \models l$.*

Ce résultat assure que, si IDB n'est pas récursive, l'axiome (1) peut être remplacé par les axiomes (1.1), ..., (1.6). Donc, en utilisant le théorème 8.1.2, si IDB n'est pas récursive, le méta-programme défini pour les bases de croyances qui sont des ensembles de littéraux propositionnels peut être utilisé dans le cas de bases de données du premier ordre qui sont équivalentes à des ensembles de littéraux de base.

Définition des réponses

Nous allons utiliser deux types de questions. Nous définissons tout d'abord les questions fermées.

Définition 8.4.4. *Soient $db_1, \ldots, db_n$ n bases de données, chacune d'entre elles étant équivalente à un ensemble de littéraux. Soit F un littéral de base. La réponse à la question*

« F est-il vrai dans la base de données obtenue en fusionnant $db_1, \ldots, db_n$ » est définie par :

- $answer((db_1 * \ldots * db_n), F) = OUI$ *ssi* *PROLOG prouve* $B((db_1 * \ldots * db_n), F)$
- $answer((db_1 * \ldots * db_n), F) = NON$ *ssi* *PROLOG prouve* $B((db_1 * \ldots * db_n, \neg F)$
- $answer((db_1 * \ldots * db_n), F) =?$ *sinon*

On peut remarquer que nous utilisons une approche de type « monde ouvert ». Dans toute base de données (primitive ou obtenue par fusion d'autres bases), il peut exister une formule atomique F telle que ni F ni $\neg F$ ne soient dérivables. Ceci explique pourquoi la réponse ? peut parfois être retournée. D'un point de vue opérationnel, cela correspond au cas pour lequel les deux buts $B((db_1 * \ldots * db_n), F)$ et $B((db_1 * \ldots * db_n), \neg F)$ échouent en temps fini dans PROLOG.

Nous définissons maintenant les questions ouvertes.

Définition 8.4.5. *Soient $db_1, \ldots, db_n$ n bases de données, chacune d'entre elles étant équivalente à un ensemble de littéraux. Soit $F(X)$ un littéral ouvert. La réponse à la question « quels sont les X qui satisfont F dans la base de données obtenue en fusionnant $db_1, \ldots, db_n$ » est définie par :*

$answer((db_1 * ... * db_n), F(X)) = \{A$: tuple de symboles de constantes tel que PROLOG prouve $B((db_1 * ... * db_n), F(A))\}$

8.4.3 Exemple

Considérons les trois bases de données suivantes :

$$db_1 = \langle EDB_1, IDB\rangle, db_2 = \langle EDB_2, IDB\rangle, db_3 = \langle EDB_3, IDB\rangle$$

avec :

$EDB_1 = \{etudiant(Jean), employe(Louis), self(Philippe), self(Donald), restaurant(Louis)\}$

$EDB_2 = \{employe(Philippe), employe(Louis), restaurant(Jean), restaurant(Henri)\}$

$EDB_3 = \{etudiant(Jean), employe(Philippe)\}$

$IDB = \{\forall x \ \ etudiant(x) \rightarrow self(x), \forall x \ \ employe(x) \rightarrow restaurant(x), \forall x \ \ \neg self(x) \vee \neg restaurant(x)\}$

Les contraintes de IDB sont communes à toutes les bases. IDB contient les contraintes suivantes : tous les étudiants mangent au self service, tous les employés mangent au restaurant et soit on ne mange pas au self, soit on ne mange pas au restaurant.

On peut remarquer que chaque base de données est équivalente à un ensemble de littéraux de base (ici tous les littéraux sont positifs) et que IDB n'est pas récursive. Voila quelques requêtes et les réponses générées par l'évaluateur de requêtes :

Est-ce que Jean est un étudiant dans la base obtenue en fusionnant db_1, db_2 et db_3 ?

$answer((db_1 * db_2 * db_3), etudiant(Jean)) = OUI$

Comme $etudiant(Jean)$ est cru par la majorité des bases (db_1 et db_3), il est cru par le résultat de la fusion de db_1, db_2 et db_3.

Est ce que Jean n'est pas un employé dans la base obtenue en fusionnant db_1, db_2 et db_3 ?
$answer((db_1 * db_2 * db_3), \neg employe(Jean)) = OUI$

Est ce que Donald est un étudiant dans la base obtenue en fusionnant db_1, db_2 et db_3 ?
$answer((db_1 * db_2 * db_3), etudiant(Donald)) =?$

La seule information que l'on a sur Donald est qu'il mange au self dans db_1. On peut donc juste dire que Donald n'est pas un employé, mais on ne sait pas s'il est un étudiant.

Qui est étudiant dans la base obtenue en fusionnant db_1 et db_2 ?
$answer((db_1 * db_2), etudiant(x)) = \emptyset$

Qui est étudiant dans la base obtenue en fusionnant db_1, db_2 et db_3 ?
$answer((db_1 * db_2 * db_3), etudiant(x)) = \{Jean\}$

Qui est un employé dans la base obtenue en fusionnant db_1, db_2 et db_3 ?
$answer((db_1 * db_2 * db_3), employe(x)) = \{Philippe, Louis\}$

Qui mange au self service dans la base obtenue en fusionnant db_1, db_2 et db_3 ?
$answer((db_1 * db_2 * db_3), self(x)) = \{Jean, Donald\}$

Qui mange au restaurant dans la base obtenue en fusionnant db_1, db_2 et db_3 ?
$answer((db_1 * db_2 * db_3), restaurant(x)) = \{Louis, Philippe, Henri\}$

Est-ce que est un étudiant dans la base obtenue en fusionnant db_1 et db_2 ?
$answer((db_1 * db_2), etudiant(Jean)) =?$

8.5 Conclusion

Nous avons développé une logique modale qui permet de raisonner avec le résultat de la fusion de plusieurs bases de croyances. La logique MF possède une axiomatique et est complète et valide pour certaines formules intéressantes. Nous nous sommes également intéressés à des aspects d'implantation et nous avons développé en démonstrateur automatique pour MF.

L'extension du méta-programme servant de démonstrateur automatique pour MF à des bases de données du premier ordre particulières nous a également permis de spécifier un évaluateur de requête. Remarquons que même si les bases présentées en exemple n'utilisaient que des littéraux monadiques (i.e. on n'avait que des symboles de prédicats d'arité 1)), on peut utiliser cette approche avec des littéraux d'arité quelconque et pour des bases de données relationnelles quelconques. Nous allons voir dans le chapitre suivant que nous pouvons également utiliser cet évaluateur de requête pour fusionner les exigences émises par plusieurs agents.

Chapitre 9

Fusion d'exigences

Nous allons nous intéresser dans ce chapitre à la fusion d'exigences. Nous avons vu qu'un moyen de résoudre les conflits entre les différents participants est de fusionner les exigences émises par ces participants pour proposer un seul ensemble d'exigences cohérent. Bien sûr, cette méthode peut sembler « radicale », mais on peut la voir comme un moyen de proposer un premier ensemble d'exigences aux participants. Libre à eux ensuite de reformuler leurs exigences si cela ne leur convient pas.

Dans le chapitre précédent, nous avons développé un évaluateur de requêtes pour des bases de données qui implante une méthode de fusion majoritaire. Nous allons utiliser cet évaluateur pour fusionner des ensembles d'exigences, exprimées sans position ou avec position. Nous montrerons comment utiliser également l'évaluateur développé par Cholvy dans [31] pour pouvoir affiner cette approche. Ces travaux ont été présentés dans [42] et [43].

9.1 Fusion simple d'exigences

Dans cette section, nous allons nous intéresser à la fusion d'exigences exprimées de façon non ordonnée. Chaque ensemble d'exigences est modélisé grâce à une base de données. Nous allons voir que nous pouvons modéliser les contraintes du domaine en utilisant les contraintes d'intégrité de chaque base de données.

9.1.1 Modélisation des exigences et des contraintes du domaine

Nous supposons que les exigences et les contraintes du domaine vont être modélisées en utilisant un langage du premier ordre que nous appelons LO. L'ensemble fini et cohérent des clauses sans fonction de LO qui modélisent les contraintes du domaine va être noté C. Nous considérons plusieurs ensembles d'exigences, notés $Req_1, \ldots, Req_n$, chacun d'entre eux étant un ensemble de clauses de LO sans fonction et cohérent avec C.

Exemple. *Cet exemple est tiré de [55]. Deux personnes émettent leurs exigences à propos de l'organisation d'une bibliothèque. La première personne exige que les dictionnaires pour enfants soient des livres de référence et que les livres de référence ne soient consultables que sur place. La seconde personne exige que les dictionnaires pour enfants soient des livres pour enfants et que les livres pour enfants soient empruntables. Les contraintes du domaine imposent qu'un livre est soit empruntable soit consultable sur place. On a les ensembles suivants de formules :*

$$Req_1 = \{livre\_reference\ (dico\_enfant), \neg livre\_reference(x) \vee consulte(x)\}$$

$$Req_2 = \{livre\_enfant\ (dico\_enfant), \neg livre\_enfant(x) \vee emprunt(x)\}$$

$$C = \{\neg consulte(x) \vee \neg emprunt(x)\}$$

Les agents peuvent donc émettre des exigences qui ne sont pas des littéraux de base. Les variables contenues dans de telles clauses sont quantifiées universellement.

9.1.2 Modification du méta-programme

Nous allons modifier le méta-programme META pour pouvoir prendre en compte le fait que les bases de données peuvent elles même contenir des clauses. Pour cela, nous remplaçons les axiomes (1.1) à (1.6) définis dans la section 8.4.2 par :

(1.1) $DATA(x, f)$ pour tout $x \in \{Req_1, \ldots, Req_n\}$ et pour toute $f \in h(y)$ où $y \in x$
(1.2) $DATA(x, f)$ pour tout $f \in h(x)$ où $x \in C$
(1.3) $DATA(rq, l) \rightarrow B_{exp}(rq, l)$ si l est un littéral
(1.4) $DATA(c, (conj \rightarrow l)) \wedge CONJ(rq, conj) \rightarrow B_{exp}(rq, l)$ où l est un littéral et $conj$ une conjonction de littéraux
(1.5) $CONJ(rq, nil)$ où nil est une constante
(1.6) $B_{exp}(rq, l) \wedge CONJ(rq, conj) \rightarrow CONJ(rq, l \wedge conj)$ pour tout littéral l et toute conjonction de littéraux $conj$

Nous appelons METAEX le nouveau programme ainsi défini.

Pour des facilités de notation, nous allons modifier la définition des questions ouvertes et fermées dans l'évaluateur de requêtes.

Définition 9.1.1. *Une question fermée est une question du type « étant donné l'ensemble des exigences obtenu en fusionnant $Req_1, \ldots, Req_n$, est-ce que F est réquis ? » (F est un littéral de base). On note cette question* question-fermee($Req_1 * \ldots * Req_n$, F) *et sa réponse est :*

- *OUI ssi PROLOG prouve $B((Req_1 * \ldots * Req_n), F)$ dans METAEX ;*
- *NON ssi PROLOG prouve $B((Req_1 * \ldots * Req_n), \neg F)$ dans METAEX ;*
- *? sinon.*

Une question ouverte est une question du type « étant donné l'ensemble des exigences obtenu en fusionnant $Req_1, \ldots, Req_n$, quels sont les X tels que $F(X)$ est requis ? ». On la note question-fermee($Req_1 * \ldots * Req_n$, $F(X)$) *et sa réponse est définie par :*

$\{A$: *tuple de symboles de constantes tel que PROLOG prouve* $B((db_1 * ... * db_n), F(A))\}$

9.1.3 Exemple

Nous montrons dans cette section comment l'évaluateur présenté dans le chapitre précédent peut être utilisé pour la fusion d'exigences. Le système que nous considérons ici est une plateforme maritime (par exemple une frégate) dont les fonctions sont de deux types : une fonction de défense anti-aérienne (AA) et une fonction de défense anti-maritime (AM). Nous considérons également qu'il y a trois personnes qui émettent des exigences à propos des capteurs que doit embarquer la plateforme. Le scénario est le suivant :

– le première personne, appelée *un*, exprime le fait que la fonction AA exige un radar multi-fonction (RMF) et un radar de recherche et de poursuite longue portée (RRPLP) ;
– la seconde personne, appelée *deux*, exprime le fait que la fonction AA exige un RMF et que la fonction AM exige un radar de surface haute fréquence (RSHF) ;
– la troisième personne, appelée *trois*, exprime le fait que la fonction AA exige un capteur de recherche et de poursuite infrarouge (RPI).

Chaque type de capteur requis devra être embarqué sur la plateforme.

Premier scénario

Supposons que pour certaines raisons (par exemple le coût) il n'est pas possible d'embarquer à la fois sur la plateforme un RMF et un capteur RPI. Cela signifie que si l'on choisit d'embarquer un capteur RPI, il n'est pas possible d'embarquer un RMF et réciproquement.

Voici les formules du méta-langage ML qui modélisent cet exemple :

$DATA(un, Requiert(AA, RMF))$
$DATA(un, Requiert(AA, RRPLP))$
$DATA(deux, Requiert(AA, RMF))$
$DATA(deux, Requiert(AM, RSHF))$
$DATA(trois, Requiert(AA, RPI))$
$DATA(c, Requiert(AA, x) \rightarrow embarque(x))$
$DATA(c, not(embarque(x)) \rightarrow not(Requiert(AA, x)))$
$DATA(c, Requiert(AM, x) \rightarrow embarque(x))$
$DATA(c, not(embarque(x)) \rightarrow not(Requiert(AM, x)))$
$DATA(c, embarque(RPI) \rightarrow not(embarque(RMF)))$
$DATA(c, embarque(RMF) \rightarrow not(embarque(RPI)))$

Voici quelques requêtes et leurs réponses générés par PROLOG.

1. **UTILISATEUR :** (question-ouverte (un * deux * trois) (embarque x))
   REPONSE : (RMF), (RRPLP), (RSHF)
2. **UTILISATEUR :** (question-fermee (un * deux * trois) (embarque RPI))
   REPONSE : NON

   La réponse à la première requête signifie qu'étant donnés les trois ensembles d'exigences, on doit embarquer un RMF, un RRPLP et un RSHF.

   Remarquons que bien que la troisième personne exprime le fait que la fonction AA requiert un capteur RPI, on ne pas conclure que l'on doit embarquer un capteur RPI. Plus précisement (cf. la deuxième requête), on peut conclure que l'on doit ne pas embarquer de capteur RPI. C'est une conséquence des contraintes et l'attitude majoritaire implantée pour gérer les contradictions : on ne peut pas embarquer à la fois un capteur RPI et un RMF, mais deux personnes sur trois exigent d'embarquer un RMF.
3. **UTILISATEUR :** (question-ouverte (un * deux * trois) (Requiert x RMF))
   REPONSE : (AA)

4. **UTILISATEUR :** (question-fermee (deux * trois) (embarque RMF))
   REPONSE : ?
5. **UTILISATEUR(20) :** (question-fermee (deux * trois) (embarque RPI))
   REPONSE : ?

   Ces deux dernières requêtes illustrent le fait que, quand on fait des requêtes avec l'évaluateur, la fusion des exigences est seulement virtuelle : les ensembles d'exigences initiaux ne sont pas modifiés et ainsi on peut faire une requête sur deux des ensembles d'exigences sur les trois, même si précédemment on a fait une requête sur les trois ensembles.

   De plus, quand on ne prend en compte que *deux* et *trois*, nous ne pouvons pas conclure si l'on doit ou pas embarquer en capteur RPI ou un RMF. Cela est dû à l'attitude majoritaire : de *trois* on peut conclure que l'on doit embarquer un RMF, mais de *deux* on doit embarquer un capteur RPI. Ce n'est pas possible à cause des contraintes. Évidemment, quand on ne considère que deux des émetteurs d'exigences, l'attitude majoritaire ne nous permet pas de conclure.
6. **UTILISATEUR :** (question-ouverte (deux * trois) (embarque x))
   REPONSE : (RSHF)

   Même si nous ne pouvons pas conclure si l'on doit embarquer un RMF ou un capteur RPI (cf. les deux requêtes précédentes), on peut conclure que l'on doit embarquer un RSHF : en fait *deux* l'exige et *trois* ne le contredit pas.
7. **UTILISATEUR :** (question-ouverte (un * deux * trois) (Requiert AA x))
   REPONSE : (RMF), (RRPLP), (RPI)

   Cette réponse signifie que, quand on considère les trois ensembles d'exigences, on peut conclure que la fonction AA requiert trois types de capteurs : un RMF, un RRPLP et un capteur RPI. On peut remarquer que cela n'est pas contradictoire avec les contraintes : en fait AA requiert un RMF et un capteur RPI, mais à cause des contraintes, on ne peut pas embarquer les deux à la fois (plus précisement (cf. la première requête), on doit embarquer un RMF).

9.1.4 Deuxième scénario

Supposons maintenant que les contraintes imposées sur les capteurs sont les suivantes :
- si on embarque un RMF, alors on ne peut pas embarquer un $RSHF$, un capteur RPI et un $RRPLP$;
- si on embarque un $RSHF$, un capteur RPI ou un $RRPLP$, alors on ne peut pas embarquer un RMF

Le résultat obtenu en fusionnant les exigences des trois agents est :
UTILISATEUR : (question-ouverte (un * deux * trois) (embarque x))
REPONSE : (nil)

Dans ce cas, on conclut qu'il n'y a pas de capteurs embarqués sur la frégate. Cela est du au fait que les contraintes empèchent l'élicitation de RMF, $LSTR$, $RSHF$ ou de RPI, comme cela est prouvé par :
UTILISATEUR : (question-ouverte (un * deux * trois) (Requiert AA x))
REPONSE : (RMF), (RRPLP), (RPI)
UTILISATEUR : (question-ouverte (un * deux * trois) (Requiert AM x))
REPONSE : (RSHF)

Ainsi, tous les capteurs sont requis par le résultat de la fusion. Mais à cause des contraintes du domaine, l'évaluateur de requête ne peut pas choisir quel est le type de capteur à embarquer sur la frégate.

9.2 Fusion d'exigences ordonnées

Dans la section précédente, les exigences des agents ont la même importance : par exemple, l'agent *un* exige que la frégate embarque un RMF et un RRPLP. Mais il voulait peut être exprimer que s'il avait à choisir entre un RMF et un RRPLP, alors il préférerait embarquer un RMF. Nous allons donc utiliser des positions pour chaque agent (cf. partie I) pour que les agents puissent ordonner leurs exigences. Les exigences des agents ne seront exprimées qu'avec des formules propositionnelles.

Supposons que la position de l'agent *un* soit $\Gamma_{un} = [RMF, RRPLP]$. Supposons également que les seules variables propositionnelles soient RMF et $RRPLP$, il y a alors quatre mondes possibles :

- w_1 dans lequel RMF et $RRPLP$ sont vraies ;
- w_2 dans lequel RMF est vraie et $RRPLP$ est fausse ;
- w_3 dans lequel RMF est fausse et $RRPLP$ est vraie ;
- w_4 dans lequel RMF et $RRPLP$ sont fausses ;

La position $[RMF, RRPLP]$ peut être vue comme un préordre de préférence sur les mondes :

$$pref(w_1, w_2)$$
$$pref(w_2, w_3)$$
$$pref(w_3, w_4)$$

où $pref$ est une relation binaire sur les mondes qui est contrainte par les propriétés de transitivité, de reflexivité et d'anti-symétrie. $pref(w_1, w_2)$ signifie que le monde w_1 est préféré au monde w_2.

Maintenant, on peut exprimer les positions des agents dans notre évaluateur de requête. Le programme PROLOG précédent peut être utilisé pour raisonner avec cet ordre de préférence. Nous utilisons un prédicat binaire $pref$ qui exprime un ordre de préférence sur les mondes. Les contraintes habituelles sur les ordres (transitivité, reflexivité et anti-symétrie) sont traduites en contraintes sur $pref$.

Les contraintes sur les exigences qui peuvent être exprimées par un seul agent émetteur d'exigences ou qui peuvent être communes à tous les agents (contraintes de domaine par exemple) sont traduites en mondes « impossibles », i.e. les mondes qui ne respectent pas les contraintes sont éliminés de la relation. Par exemple, si l'on suppose que la contrainte imposée sur la frégate est qu'il est impossible d'embarquer à la fois un RMF et un RRPLP, alors la position $[RMF, RRPLP]$ sera traduite en :

$$pref(w_2, w_3)$$
$$pref(w_3, w_4)$$

Le résultat du processus de fusion est un préordre sur les mondes. Bien sûr, ce préordre peut être partiel, i.e. des mondes peuvent être incomparables. Comme toutes les contraintes

sur les exigences sont exprimées avant le processus de fusion, nous considérons que seuls les mondes les plus préférés sont intéressants : ils caractérisent les exigences idéales obtenues en fusionnant les positions des agents.

Prenons un exemple qui traite d'exigences ordonnées. Comme dans la section 9.1.3, nous considérons trois agents qui expriment leurs exigences sur les capteurs qui doivent être embarqués sur la frégate. Les trois agents émettent les exigences suivantes :

- l'agent *un* requiert que la frégate embarque un MFR et un RRPLP ;
- l'agent *deux* requiert que la frégate embarque un MFR et un RSHF ;
- l'agent *trois* requiert que la frégate embarque un capteur RPI.

Examinons quelques scénarios à propos des positions des différents agents.

9.2.1 Premier scénario

Supposons que les positions des agents soient les suivantes :

- $\Gamma_{un} = [RMF, RRLPL]$, ce qui signifie que l'agent *un* préfère embarquer RMF plutôt que RRLPL ;
- $\Gamma_{deux} = [RSHF, RMF]$, ce qui signifie que l'agent *deux* préfère embarquer RSHF plutôt que RMF ;
- $\Gamma_{trois} = [RPI]$, ce qui signifie que l'agent *trois* requiert seulement que la frégate embarque un capteur RPI.

Supposons également que l'on ne puisse pas embarquer à la fois un RMF et un capteur RPI sur la frégate. Dans ce cas, on élimine tous les mondes qui satisfont $RMF \wedge RPI$. Le monde le plus préféré dans ce cas est le monde qui satisfait RMF, $RRLPL$, $RSHF$ et $\neg RPI$.

C'est le même cas que le premier scénario de la section précédente. Nous avons les mêmes résultats. En fait, l'ordre sur les exigences ne nous apporte aucune information nouvelle, car nous sommes certains que l'exigence RPI sera éliminée dans le processus de fusion. Toutes les exigences de *un* et de *deux* sont satisfaites.

9.2.2 Deuxième scénario

Supposons que les positions des agents soient les mêmes que celle du premier scénario. Supposons que si un RMF est embarqué, alors un RRLPL, un RSHF et un capteur RPI ne peuvent pas être embarqués. Enfin, supposons que si un RRLPL, un RSHF ou un capteur RPI est embarqué, alors un RMF ne peut pas être embarqué.

En utilisant les positions des agents, le monde le plus préféré dans ce cas est celui qui satisfait $RRLPL$, $RSHF$, RPI et $\neg RMF$. Comme les agents ont ordonné leurs exigences, les contraintes n'empêchent pas l'évaluateur de requête de déduire quel type de capteur doit être embarqué contrairement à la section 9.1.4. En particulier, l'agent *deux* préfère embarquer un RSHF plutôt qu'un RMF, ce qui n'est pas contredit par la position de *trois*, donc les mondes les plus préférés après la fusion ne satisfont RMF.

Notons que les agents *un* et *deux* expriment des exigences qui ne satisfont pas les contraintes. Cela signifie que les exigences idéales émises par *un* et *deux* ne seront pas choisies. Mais comme ils expriment une position sur leurs exigences, on peut choisir une situation moins préférée qu'ils ont émise.

9.2.3 Troisième scénario

Supposons que les positions des agents sont :
- $\Gamma_{un} = [RMF, RRLPL]$, ce qui signifie que l'agent *un* préfére que l'on embarque un RMF plutôt qu'un RRLPL ;
- $\Gamma_{deux} = [RMF, RSHF]$, ce qui signifie que l'agent *deux* préfére embarquer un RMF plutôt qu'un RSHF. Notons que l'agent *deux* a révisé sa position : il préfére maintenant embarquer un RMF plutôt qu'un RSHF ;
- $\Gamma_{trois} = [RPI]$, ce qui signifie que l'agent *trois* requiert seulement que la frégate embarque un capteur RPI.

Supposons que les contraintes imposées sur les exigences soient les mêmes que dans le scénario précédent. Comme dans le scénario de la section 9.1.4, le processus de fusion sans ordre sur les exigences ne peut rien déduire à propos des capteurs à embarquer.

En utilisant les positions des agents, l'évaluateur de requête conclut que le monde le plus préféré dans ce cas est le monde qui satisfait RMF, $\neg RRLPL$, $\neg RSHF$ et $\neg RPI$. Comme l'agent *deux* a révisé sa position sur la frégate, on choisit un RMF comme le capteur à embarquer sur la frégate. Notons que si on embarque un RMF, alors aucun autre capteur ne peut être embarqué. Comme les agents *un* et *deux* ont choisi RMF comme étant leur exigence prioritaire sur la frégate, un RMF est le seul capteur à être embarqué.

Remarquons également que dans tous ces exemples, l'évaluateur de requêtes conclut qu'il n'y a qu'un seul monde le plus préféré, mais il peut très bien en avoir plusieurs.

9.3 Conclusion

L'évaluateur de requêtes présenté précédemment, basé sur la logique MF, permet donc de fusionner selon une approche majoritaire des ensembles d'exigences éventuellement contradictoires. Nous avons montré que ces ensembles peuvent être des ensembles d'exigences sans priorité, ou bien des ensembles d'exigences avec priorité. Nous pouvons donc grâce aux techniques de fusion majoritaire proposer un ensemble cohérent d'exigences aux différents participants. Cette méthode permet d'éliminer les éventuelles incohérences entre exigences. On peut voir cette technique comme un moyen de proposer un consensus entre les différents agents. Elle permet de fournir rapidement un ensemble d'exigences cohérent qui peut éventuellement servir de point de départ à une renégociation entre les agents.

L'utilisation d'une logique de fusion majoritaire permet de traiter les agents sur un même pied d'égalité. Or dans une structure décisionnelle, les agents n'ont pas forcément le même poids : certains peuvent avoir plus d'importance que d'autres. Par exemple, les exigences émises par un service financier sont souvent prioritaires par rapport aux exigences techniques. Il peut donc être interessant d'utiliser l'évaluateur de requête développé par Cholvy dans [31] pour raisonner sur le résultat de la fusion prioritaire de plusieurs bases.

Le principe est simple : on sépare les différents agents en clusters de même importance hiérarchique. À l'intérieur de ces clusters, on fusionne les exigences des agents de façon majoritaire pour obtenir les exigences de chaque groupe. Ensuite, on fusionne les exigences de chaque cluster de façon prioritaire pour respecter la structure hiérarchique de l'organisation.

Troisième partie

Distribution d'exigences

Introduction

La plupart des études faites sur l'ingénierie des exigences s'arrêtent à la production d'un ensemble cohérent d'exigences ou à la construction de méthodes permettant de gérer les éventuels problèmes d'incohérence dans le processus. Or ces exigences définissent un objet qui va être construit par des agents exécutants. Nous pensons que nous intéresser à la phase de « construction » de l'objet peut permettre de détecter des impossibilités dans les exigences : il se peut très bien que l'on exige une propriété sur l'objet mais que l'on n'ait pas les moyens de la garantir (i.e. on ne dispose pas d'agents exécutants capables de construire un objet respectant cette propriété).

Pour cela, nous allons supposer que nous disposons d'un ensemble d'agents exécutants qui sont chargés de construire l'artefact à partir d'un ensemble cohérent d'exigences (celui obtenu après la phase de fusion). Les exigences vont donc devenir des buts pour ces agents exécutants. Le problème qui se pose est de pouvoir distribuer cet ensemble de buts communs sur le groupe d'agents en prenant en compte certaines caractéristiques des agents et plus particulièrement leurs compétences (un agent ne pourra réaliser un but du groupe que si celui-ci entre dans son domaine de compétence). Nous proposons également de modéliser ce processus de distribution comme étant centralisé : chaque agent s'engage à réaliser une ou plusieurs tâches et communique ses engagements à une entité centrale. Cette entité répartit ensuite les tâches de chaque agent en suivant les engagements des autres agents. On montre que l'on peut introduire à ce niveau différentes stratégies de distribution suivant les engagements de agents.

Après un rapide état de l'art sur la modélisation en logique d'agents rationnels, nous allons définir comment modéliser chaque agent exécutant. Nous reviendrons alors sur les notions déontiques que nous avons présentées dans la partie I. Enfin, nous présenterons un processus de distribution des buts sur le groupe d'agents exécutants et nous montrerons que l'on peut définir des stratégies de distribution diverses.

Chapitre 10

Logiques pour la modélisation d'agents

Dans ce chapitre, nous allons présenter quelques formalismes logiques qui permettent de modéliser des agents rationnels. Par agents rationnels, on entend des agents qui ont des états mentaux propres, comme des croyances à propos de l'état du monde, des buts ou des intentions, qui peuvent raisonner sur ces états mentaux et qui agissent en fonction de ceux-ci. Le but de ces formalismes est donc de pouvoir modéliser correctement de telles notions, en particulier de définir les liens formels qui existent entre elles. Nous nous intéresserons plus particulièrement à un aspect « théorie de la décision », puisque nous voulons pouvoir déterminer les buts d'un agent à partir de ses préférences et de ses croyances. Par exemple, si un agent croit qu'il pleut et que s'il pleut il préfére avoir un parapluie, on devrait pouvoir en déduire qu'il va prendre un parapluie (si celui ci est disponible).

Là encore, la logique nous parait être un formalisme parfaitement adaptée à ce type de modélisation : les différents agents disposent d'un formalisme commun qui permet d'éviter tout ambiguité sur la signification des termes et on peut facilement raisonner sur les concepts décrits.

Nous allons présenter deux approches qui ont émergé au cours de ces dix dernières années : les approches de type BDI qui se fondent sur une modélisation des croyances, des désirs et des intentions de l'agent et une approche proposée par Boutilier qui étend de façon qualitative la théorie de la décision classique.

10.1 Approches BDI (Belief / Desire / Intention)

Les formalismes qui attachent une importance particulière à la notion d'intention sont appelés architectures BDI. L'intention joue un rôle prépondérant dans la détermination du comportement d'agents rationnels cherchant à atteindre leurs buts. Les formalismes de type BDI cherchent donc à relier les notions de croyance, de désir (ou de but) et d'intention. En particulier, les intentions antérieures que l'agent a adoptées vont contraindre les actions futures de l'agent. Des principes sont créés pour pouvoir gouverner l'interaction entre les intentions antérieures de l'agent et la formation de nouvelles intentions. Par exemple, on peut considérer qu'un agent abandonne une de ses intentions si le but concerné a été atteint ou qu'il est devenu inatteignable pour une raison quelconque.

Les principaux travaux sur les approches BDI sont ceux effectués par Rao et Georgeff [116, 117], Jennings et Wooldrige [139, 62], Singh [123, 124, 125, 70] ou Cohen et Levesque

[46]. Nous allons nous intéresser ici à l'approche proposée par Rao et Georgeff dans [116].

La logique définie par Rao et Georgeff est une logique multi-modale constituée de trois opérateurs modaux : B pour les croyances, G pour les buts et I pour l'intention. On peut remarquer que l'intention est pour eux un concept de base, alors que chez Cohen et Levesque [46], l'intention est dérivée des autres attitudes mentales. Rao et Georgeff rejoignent dans ce sens la théorie de l'intention de Bratman [16].

Ces modalités sont reliées entre elles : ainsi, si une formule φ est un but pour un agent, celui-ci doit croire que ce but est possible ($G\varphi \rightarrow B\varphi$). Cette propriété est appelée propriété de *réalisme faible*. De même, si un agent à l'intention de réaliser φ, alors φ doit être un but pour l'agent ($I\varphi \rightarrow G\varphi$). Les trois opérateurs modaux sont ainsi liés par des relations qui expriment des propriétés que l'on voudrait avoir intuitivement pour un agent rationnel.

La sémantique de la logique BDI se fonde sur des modèles de type *branching-time* (cf. par exemple [69] ou la logique CTL). Dans la sémantique de la logique BDI, la notion de satisfaction des formules peut être de deux types : une satisfaction par rapport à un état donné ou par rapport à un chemin donné. La satisfaction par rapport à un état donné permet de représenter les effets des actions et la satisfaction par rapport à un chemin donné (on ordonne des états linéairement pour pouvoir représenter des notions temporelles) permet de représenter des plans ou des combinaisons d'actions pour l'agent.

Pour pouvoir déterminer le rôle des intentions premières des agents, Rao et Georgeff définissent trois grands types d'agents qui permettent de définir les relations entre les mondes accessibles par l'opérateur d'intention et les mondes accessibles par l'opérateur de croyance et de but. Ces types sont :

- les agents *blind-committed* pour lesquels les intentions premières peuvent ne pas changer même si les croyances ou les buts de l'agent changent ;
- les agents *single-minded* pour lesquels les intentions de l'agent peuvent changer seulement si les croyances ou les buts de l'agent changent ;
- les agents *open-minded* pour lesquels les intentions de l'agent changent si les croyances ou les buts de l'agent changent.

Leur approche permet donc de caractériser différents types d'agents. Ces différents types d'agents correspondent à différentes stratégies d'engagement des agents vis-à-vis de leurs intentions futures. Nous ne développerons pas ce sujet ici.

10.2 Approche de Boutilier

Dans [13], Boutilier propose une approche logique et qualitative de la théorie de la décision. La théorie de la décision cherche à déterminer pour un agent rationnel quels sont ses buts connaissant ses croyances sur l'état du monde et ses préférences. En théorie de la décision « classique » (cf. [137]), l'état du monde est représenté par une distribution de probabilité et les préférences de l'agent par une fonction d'utilité sur les sorties possibles. Boutilier utilise quant à lui la logique CO pour pouvoir représenter à la fois les préférences de l'agent et un ordre sur les mondes possible représentant la normalité des différents mondes. Un monde w_1 est plus normal qu'un monde w_2 si, étant donnée une connaissance incomplète du monde réel, on peut dire qu'il y a plus de chances que w_1 soit le monde réel que w_2.

Nous avons vu dans le chapitre 4.2.3 que CO et CO^* permettent de représenter des préférences conditionnelles. Boutilier utilise donc CO pour pouvoir représenter les

préférences de l'agent sous forme de formules du type $I(\alpha|\beta)$. Nous allons maintenant nous intéresser à son modèle de représentation du monde.

10.2.1 Représentation du monde

L'agent dispose d'une certaine représentation du monde. Cette représentation peut bien évidemment être incomplète, car l'agent n'est peut être pas capable de faire les observations nécessaires pour pouvoir déterminer la valeur de vérité de tous les atomes représentant le monde réel. Dans toute la suite de la section, on considère donc un ensemble fini et cohérents de formules propositionnelles qui représente les faits vrais dans le monde connus par l'agent.

Définition 10.2.1. *On appelle* base de croyances *un ensemble fini et cohérent de formules propositionnelles. On le note KB.*

Boutilier considère que les buts d'un agent ne doivent pas être déterminés en fonction d'un état certain du monde, mais à partir d'un raisonnement par défaut qui nous permet de tirer des conclusions vraisemblables étant données des observations partielles du monde réel. Par exemple, si l'agent sait qu'il y a des nuages, il peut en déduire que la situation où il pleut est plus vraisemblable que la situation où il y a du soleil. Pour représenter cela, Boutilier utilise de nouveau un préordre sur les mondes, ce préordre représentant la normalité relative des différents mondes. Il définit alors une logique QDT dont les modèles sont définis ci-après.

Définition 10.2.2. *Un modèle M de QDT est un tuple :*

$$M = \langle W, \leq_P, \leq_N, val \rangle$$

où :
- W est un ensemble de mondes possibles ;
- $\leq_P$ représente un préordre de préférence sur les mondes possibles. Il correspond au préordre présenté dans le chapitre 4.2.3 ;
- $\leq_N$ est un nouveau préordre sur les mondes possibles qui possède les mêmes propriétés que $\leq_P$, mais qui représente le degré de normalité des mondes ;
- *val* est une fonction de valuation sur W.

Ainsi, si l'on considère deux mondes w_1 et w_2, $w_1 \leq_N w_2$ signifie que le monde w_1 est considéré comme étant plus vraisemblable que le monde w_2. Si on prend deux variables propositionnelles p (il pleut) et n (il y a des nuages), on peut par exemple écrire que $\{p, n\} \leq_N \{\neg p, n\}$.

Boutilier définit ensuite les opérateurs $\Box_P$, $\overleftarrow{\Box}_P$, $\Box_N$ et $\overleftarrow{\Box}_N$ comme il a défini $\Box$ et $\overleftarrow{\Box}$ dans la section 4.2.3 et note :

$$\alpha \Rightarrow \beta \quad \equiv_{def} \quad \overleftrightarrow{\Box}_N \neg\alpha \vee \overleftrightarrow{\Diamond}_N (\alpha \wedge \Box_N(\alpha \rightarrow \beta))$$

$\alpha \Rightarrow \beta$ signifie donc que β est vraie dans les plus normaux des α-mondes. L'opérateur $I(-|-)$ est défini comme dans la section 4.2.3. La sémantique de $\Rightarrow$ est donc la même que celle de l'opérateur d'idéalité.

Lorsque $\Box$ et $\Diamond$ n'ont pas d'indice, c'est que l'on considère indifféremment les opérateurs comme étant indicés par N ou P. L'axiomatique des opérateurs de préférence $\Box_P$, $\overleftarrow{\Box}_P$ et des opérateurs de normalité $\Box_N$, $\overleftarrow{\Box}_N$ est donnée par :

Schémas d'axiomes

K $\Box(\alpha \to \beta) \to (\Box\alpha \to \Box\beta)$
K' $\overleftarrow{\Box}(\alpha \to \beta) \to (\overleftarrow{\Box}\alpha \to \overleftarrow{\Box}\beta)$
T $\Box\alpha \to \alpha$
4 $\Box\alpha \to \Box\Box\alpha$
S $\alpha \to \overleftarrow{\Box}\Diamond\alpha$
H $\overrightarrow{\Diamond}(\Box\alpha \wedge \overleftarrow{\Box}\beta) \to \overrightarrow{\Box}(\alpha \vee \beta)$
PN $\overrightarrow{\Box}_N \alpha \leftrightarrow \overrightarrow{\Box}_P \alpha$

Règles d'inférence

Nec $\frac{\vdash \alpha}{\vdash \overrightarrow{\Box}\alpha}$

MP $\frac{\vdash \alpha \quad \vdash \alpha \to \beta}{\vdash \beta}$

Le schéma d'axiome **PN** permet de lier les deux types d'opérateurs : une formule nécessairement vraie en terme de normalité le sera aussi en terme de préférences. Notons également que l'on peut définir QDT^* à partir de QDT de la même façon que l'on a défini CO^* à partir de CO.

Boutilier note ensuite :

Définition 10.2.3. *Soit $\mathcal{N}$ un ensemble fini et cohérent de formules du type $\alpha \Rightarrow \beta$.*

$$Cl(KB) = \{\varphi \ : \ \mathcal{N} \models KB \Rightarrow \varphi\}$$

$Cl(KB)$ représente donc les croyances par défaut qu'a l'agent du monde réel. Bien sûr, ces croyances peuvent être fausses : il se peut très bien qu'il y ait des nuages qu'il ne pleuve pas par exemple. Dans toute la suite, on supposera que $Cl(KB)$ est fini et qu'il est équivalent à une proposition.

10.2.2 Buts d'un agent

Boutilier peut maintenant déterminer quels sont les buts de l'agent. Intuitivement, les buts d'un agent dans une situation donnée vont correspondre à des propositions qui sont vraies dans les mondes les plus préférés. Il faut donc calculer les buts de l'agent après avoir calculer la fermeture par $\Rightarrow$ de KB (un agent agit en fonction de ce qu'il croit le plus normal).

Buts idéaux

Les buts idéaux vont représenter les propositions vraies dans les meilleures situations où $Cl(KB)$ est vraie. Ces propositions ne vont pas permettre de déterminer ce que l'agent va faire, puisque l'on n'a pas encore modélisé la capacité de l'agent à faire quelque chose. Les buts idéaux portent donc bien leur nom : ils considèrent que l'agent peut changer la valeur de vérité de n'importe quel atome du langage.

Définition 10.2.4. *Soit M un modèle. Un but idéal est une proposition φ telle que :*

$$M \models I(\varphi | Cl(KB))$$

L'ensemble des buts idéaux de l'agent est noté Id.

Exemple. *Considérons un langage dont les variables propositionnelles sont l (la porte est laquée) et p (la porte est poncée). Soit l'ensemble de préférence $\mathcal{P} = \{I(l), I(\neg l|\neg p)\}$ signifiant que l'agent préfére que la porte soit laquée et si elle n'est pas poncée, l'agent préfére que la porte ne soit pas laquée. On suppose également que $\mathcal{N} = \{c \Rightarrow p\}$, i.e. que normalement, s'il y a des copeaux par terre, la porte est poncée.*

Comme les préférences ne concernent pas la variable c, pour simplifier le problème, nous ne considérerons que les mondes qui concernent l et p. Les mondes possibles sont donc $w_1 = \{l, p\}$, $w_2 = \{\neg l, \neg p\}$, $w_3 = \{l, \neg p\}$, $w_4 = \{\neg l, p\}$.

Du fait de $I(l)$, les situations w_1 et w_3 peuvent être les mondes préférés. Mais, à cause de $I(\neg l|\neg p)$, w_3 ne peut pas être le monde le plus préféré. Donc w_1 est le monde le plus préféré et on a $w_1 <_P w_2$, $w_1 <_P w_3$ et $w_1 <_P w_4$. De plus, on ne peut pas avoir $w_3 \leq_P w_2$, car on a $I(\neg l|\neg p)$. On a donc $w_2 <_P w_3$. $I(l)$ et $I(\neg l|\neg p)$ ne sont donc satisfaites que par les modèles de QDT[1] suivants :

$$\begin{array}{llllllll}
M_1 & w_1 & \leq_P & w_2 & \leq_P & w_3 & \leq_P & w_4 \\
M_2 & w_1 & \leq_P & w_2 & \leq_P & w_4 & \leq_P & w_3 \\
M_3 & w_1 & \leq_P & w_2 & \leq_P & w_4 & \approx_P & w_3 \\
M_4 & w_1 & \leq_P & w_2 & \leq_P & w_4 & \leq_P & w_3 \\
M_5 & w_1 & \leq_P & w_2 & \approx_P & w_4 & \leq_P & w_3 \\
M_6 & w_1 & \leq_P & w_4 & \leq_P & w_2 & \leq_P & w_3
\end{array}$$

Supposons que $KB_1 = \{c\}$ (il y des copeaux par terre et c'est une information connue par l'agent). Alors $Cl(KB_1) = \{c, p\}$ car $c \Rightarrow p$. Les buts idéaux sont les α tels que $\forall M\ M \models I(\alpha|p)$. l est donc le but idéal de l'agent. Comme la porte est poncée, il préfére qu'elle soit laquée.

Si maintenant on suppose que $KB_2 = \{\neg p\}$ (la porte n'est pas poncée), on peut montrer que le but idéal est $\neg l$: comme la porte n'est pas poncée, l'agent préfére qu'elle ne soit pas laquée.

Boutilier introduit une notion élégante pour déterminer quelles sont les conditions suffisantes pour satisfaire tous les buts idéaux :

Définition 10.2.5. *Soit φ une proposition. ψ est une condition suffisante étant donnés φ et un modèle $\mathcal{M}$ de QDT ssi :*
- *$\psi \wedge \varphi$ est satisfaisable ;*
- *$\mathcal{M} \models \overleftrightarrow{\Box}_P (\varphi \rightarrow \overleftarrow{\Box}_P (\varphi \rightarrow \neg\psi))$*

Intuitivement, une condition suffisante ψ nous garantit que l'agent est dans un des meilleurs φ-mondes possibles.

Propriété 10.2.1. *Si ψ est une condition suffisante pour $Cl(KB)$ étant donné un QDT modèle $\mathcal{M}$ alors pour tout but idéal φ on a $M \models \psi \wedge Cl(KB) \rightarrow \varphi$.*

CK-buts

Les buts idéaux sont censés représenter le fait qu'un agent doit atteindre la meilleure situation possible en fonction de sa connaissance du monde. Or, cette définition est trop restrictive pour deux raisons :

[1] En fait, on ne considère que des modèles de QDT^* pour des raisons de simplicité.

- on ne devrait considérer que les éléments de KB qui sont fixés : si un agent peut changer la valeur de vérité d'un atome vrai dans KB, alors celui-ci ne doit pas entrer en compte dans le calcul des buts de l'agent. En particulier, comme pour toute formule φ de $PROP$ et pour tout modèle M de QDT $M \models I(\varphi|\varphi)$, tous les atomes vrais dans KB deviennent des buts idéaux ;
- les buts idéaux représentent des situations préférées que l'agent doit atteindre. Ils ne représentent pas ce que l'agent doit *faire*. En effet, un agent peut préférer qu'il pleuve par exemple. Il est tout a fait juste de penser que l'agent n'a aucun contrôle sur le fait qu'il pleuve. Dans ce cas, il paraît légitime de considérer « il pleut » comme étant une situation idéale, mais pas comme étant un but pour l'agent.

Pour raffiner la notion de but, Boutilier introduit un modèle simple des capacités de l'agent. Il partitionne l'ensemble des variables propositionnelles en deux classes disjointes C et $\overline{C}$ qui représentent respectivement les variables S-contrôlables[2] par l'agent (dont il peut changer la valeur de vérité) et les variables S-incontrôlables par l'agent (dont il ne peut pas changer la valeur de vérité). Ainsi la variable représentant le fait « l'agent ponce la porte » peut être considérée comme S-contrôlable, mais la variable représentant le fait « il fait beau » est S-incontrôlable par l'agent. Boutilier étend ensuite ces notions aux propositions :

Définition 10.2.6. *Soit $PROP$ un ensemble de variables propositionnelles. $V(P)$ est l'ensemble des distributions de valeurs de vérité de cet ensemble. Si P et Q sont deux ensembles disjoints et que $v \in V(P)$ et $w \in V(Q)$, alors $v;w \in V(P \cup Q)$ représente les distributions correspondant aux variables de P et de Q. L'élément neutre pour ; est la distribution vide.*

Définition 10.2.7. *Une proposition α est* S-contrôlable *ssi pour tout $u \in V(\overline{C})$, il existe $v \in V(C)$ et $w \in V(C)$ tels que $v;u \models \alpha$ et $w;u \models \neg\alpha$.*

Une proposition α est S-influençable *ssi il existe $u \in V(\overline{C})$, il existe $v \in V(C)$ et $w \in V(C)$ tels que $v;u \models \alpha$ et $w;u \models \neg\alpha$.*

α est non-S-influençable *ssi elle n'est pas influençable.*

Ainsi, si a est S-contrôlable et b est S-incontrôlable, $a \wedge b$ est S-influençable, mais pas S-contrôlable. En effet, l'agent ne peut pas changer la valeur de vérité de $a \wedge b$ si b est faux, mais par contre, si b est vrai, alors l'agent peut rendre $a \wedge b$ vraie ou fausse suivant la valeur qu'il donne à a.

Définition 10.2.8. *Les croyances non S-influençables d'un agent sont définies par $UI(KB) = \{\alpha \in Cl(KB) \ : \ \alpha$ est non-S-influençable$\}$.*

Boutilier détermine les CK-buts à partir des propositions non-S-influençables qui sont vraies dans $Cl(KB)$. Comme le note Boutilier, lors du calcul de $Cl(KB)$, on considère bien évidemment toutes les propositions. Ils se peut en effet que l'on ait une règle $c \Rightarrow p$ et que l'agent puisse changer la valeur de vérité de c mais ne puisse pas changer la valeur de vérité de p. Si $c \in KB$, on calculera quand même $Cl(KB)$ en utilisant la règle précédente. Les règles utilisant le préordre de normalité des mondes ne sont pas des règles de causalité (i.e. le fait d'avoir des copeaux n'est pas une cause du ponçage de la porte), mais des

[2]Nous avons changé la notation de Boutilier qui appelle les variables de C variables *contrôlables*. Nous les appelons S-contrôlables (pour Symétriques-contrôlables, car si une variable est contrôlable, sa négation l'est aussi) pour les distinguer des littéraux contrôlables que nous introduirons dans le chapître 11.

règles de preuve (i.e. le fait d'avoir des copeaux est normalement une preuve que la porte a été poncée). En particulier, enlever les copeaux ne va pas changer le fait que la porte soit poncée. Il faut donc bien utiliser les atomes S-contrôlables pour calculer $Cl(KB)$.

Dans un premier temps, Boutilier considère que $UI(KB)$ est complet[3], i.e. que la valeur de vérité de tous les atomes S-incontrôlables est connue. Sous cette hypothèse, il définit la notion de *CK-but*.

Définition 10.2.9. *Soient $\mathcal{P}$ un ensemble de préférences conditionnelles, $\mathcal{N}$ un ensemble de règles par défaut et KB une base de connaissances telle que $UI(KB)$ est complet. Une proposition α est un CK-but dérivé de $\mathcal{P}$ ssi $\mathcal{P} \models I(\alpha|UI(KB))$ et α est S-contrôlable.*

Boutilier définit l'ensemble des actions « atomiques » nécessaires à la réalisation des CK-buts.

Définition 10.2.10. *Soit M un modèle de QDT. Un ensemble de buts atomiques est un ensemble de littéraux $\mathcal{A} = \{l_1, \ldots, l_n\}$ S-contrôlables tels que pour tout CK-but φ de M, $M \models UI(KB) \wedge \mathcal{A} \rightarrow \varphi$.*

Exemple. *Reprenons l'exemple précédent. Supposons que $KB = \{c\}$, il y a donc des copeaux par terre. En utilisant la règle par défaut $c \Rightarrow p$, on peut donc en déduire que $Cl(KB) = \{c, p\}$. Supposons que l soit la seule variable S-contrôlable par l'agent. Dans ce cas, $UI(KB) = \{c, p\}$. On peut donc en déduire que $\{l\}$ est le seul ensemble d'actions atomiques de l'agent. L'agent a pour but de laquer la porte.*

Si l'on suppose maintenant que $KB = \{\neg p\}$, mais que l'agent contrôle les deux variables l et p (l'agent peut poncer la porte et la laquer). Dans ce cas, $UI(KB) = \phi$, donc l'ensemble d'actions atomiques de l'agent est $\{l, p\}$. Comme l'agent peut poncer la porte, on ne considère plus $\neg p$ pour déterminer ses buts.

10.2.3 Priorité des défauts sur les préférences

Dans le processus de dérivations des buts proposé par Boutilier, on calcule d'abord la fermeture de KB par les règles de défauts avant de calculer les buts de l'agent. Implicitement,une certaine priorité est donnée aux défauts. En particulier, on ne peut pas mesurer l'impact sur les buts de l'agent du fait que la règle par défaut utilisée soit fausse. La probabilité d'avoir une représentation du monde fausse n'est pas utilisée.

Reprenons l'exemple précédent. Supposons que $KB = \{c\}$ et que p soit S-incontrôlable. Dans ce cas, $UI(KB) = \{c, p\}$ et donc l'agent va avoir pour but de laquer la porte. Mais si la règle de défaut ne s'applique pas ici (i.e. il y a des copeaux par terre, mais la porte n'est pas poncée), l'agent va laquer la porte alors que celle-ci n'est pas poncée.

Pour pouvoir raisonner correctement avec ce formalisme, il faut que les echelles de préférence et de normalité soient calibrées correctement. En regroupant les différents mondes en clusters d'utilité proche, Boutilier présente une propriété simple que doivent vérifier les préordres de défaut et de préférence pour pouvoir dériver des buts acceptables.

10.3 Conclusion

Nous avons présenté dans cette section deux grandes approches pour pouvoir raisonner avec des agents rationnels : les approches de types BDI et l'approche de Boutilier qui est

[3]Dans un second temps, Boutilier considère que $UI(KB)$ est incomplet, mais nous n'étudierons pas ce cas ici.

une extension qualitative de la théorie de la décision. Notons qu'il existe d'autres approches qualitative de la théorie de la décision, comme celle proposée par Pearl dans [108]. Notons également que Dastani et al. comparent dans [48] ces deux approches du point de vue de la théorie de la décision classique.

Notre objectif est de pouvoir modéliser des agents rationnels qui sont censés construire un artefact défini par un ensemble d'exigences. Nous avons vu dans les parties précédentes que l'utilisation de la logique CO nous permet de représenter les exigences sous forme de positions, mais également une réglementation ou des contraintes du domaine. Nous allons donc étendre l'approche de Boutilier pour pouvoir, à partir d'un ensemble d'exigences exprimées grâce à CO, déterminer les buts de chacun des agents exécutants en fonction de leurs croyances et de leurs préférences initiales.

Chapitre 11

Modélisation d'agents exécutants

Notre premier objectif est de pouvoir modéliser des agents exécutants. Ces agents sont censés être des agents rationnels, qui ont une certaine connaissance du monde, des capacités à réaliser certaines actions et des préférences sur les états du monde possibles. À partir de ces notions, nous devons être capable de déterminer quels sont leurs buts ou leurs intentions. Nous avons vu que Boutilier propose un modèle logique de la théorie de la décision classique qui permet de déterminer les buts d'un agent en fonction de ses croyances, de ses capacités et de ses préférences.

Le formalisme de Boutilier souffre d'un défaut important (Boutilier l'avait également remarqué dans [13]) : le modèle de capacité formalisé via la notion de S-contrôlabilité n'est pas très réaliste. Ainsi, si un littéral l est S-contrôlable, alors $\neg l$ l'est également. Un agent ayant la capacité de poncer une porte a également la capacité de la « déponcer », ce qui peut sembler intuitivement incorrect. Nous allons donc nous attacher à étendre ce modèle de capacité pour pouvoir éviter de tels cas.

De plus, puisque nous avons montré que nous pouvons modéliser des phrases normatives dans ce formalisme, il nous semble pertinent de pouvoir déterminer si un agent exécutant viole une obligation exprimée par une réglementation. Nous pourrons en particulier modéliser différents types d'agents, suivant qu'ils accordent une importance plus ou moins grande à la réglementation.

Nous allons donc dans un premier temps proposer un nouveau modèle de capacité pour les agents, puis nous redéfinirons la notion de but et nous nous intéresserons ensuite aux aspects normatifs. Dans toute la suite de la section, $PROP$ représentera un langage propositionnel et W un ensemble de mondes possibles construit à partir de $PROP$.

11.1 Actions et capacités

Puisque nous utilisons l'approche de Boutilier pour modéliser les agents exécutants, nous ne disposons pas d'un ensemble de symboles particuliers qui représentent les actions disponibles pour l'agent. On ne travaille que sur les mondes possibles. Les actions disponibles pour l'agent sont donc les suivantes :

- soit il change la valeur de vérité d'une formule φ en utilisant l'action $\texttt{do}(\varphi)$;
- soit il conserve la valeur de vérité d'une formule φ en utilisant l'action $\texttt{keep}\ (\varphi)$;
- soit il ne fait rien.

Nous ne nous intéressons pas à la définition même des actions (séquentialité, parallélisme etc) ni aux problèmes de causalité. On suppose juste que l'agent peut changer

(ou conserver) par une de ses actions la valeur de vérité d'une formule propositionnelle. Notons que nous ne voulons pas raisonner sur les actions possibles de l'agent, mais simplement pouvoir déterminer quels sont les actions disponibles pour l'agent. L'utilisation de formalismes plus compliqués (cf. [69] par exemple) est donc inutile.

Dans [13], Boutilier propose un modèle de contrôlabilité qui se fonde sur une partition entre les variables propositionnelles du langage. Dans ce modèle, si un atome l est S-contrôlable par l'agent, cela signifie que l'agent peut réaliser l'action $\texttt{do}(l)$. Comme la partition est effectuée sur les variables propositionnelles, si l est contrôlable, alors l'agent peut également réaliser l'action $\texttt{do}(\neg l)$. Comme nous l'avons précisé, cela peut paraitre contre-intuitif. Si l'on considère que p qui représente le fait qu'une certaine porte est poncée est S-contrôlable par l'agent, alors l'agent a la capacité de poncer la porte, mais également de la déponcer. Nous allons donc étendre le modèle de S-contrôlabilité proposé par Boutilier afin d'obtenir un modèle de contrôlabilité qui corresponde aux trois types d'actions que nous avons présentés ci-dessus. Pour cela, nous partitionnons non pas les variables propositionnelles du langage en deux classes, mais les littéraux du langage en deux classes.

Définition 11.1.1. *On note C l'ensemble des littéraux contrôlables par l'agent et $\overline{C}$ l'ensemble des littéraux incontrôlables par l'agent.*

On appelle $lit(PROP)$ l'ensemble des littéraux de $PROP$.

Soit $\mathcal{L}$ un ensemble de littéraux. On appelle $C(\mathcal{L})$ l'ensemble $C \cap \mathcal{L}$ et $\overline{C}(\mathcal{L})$ l'ensemble $\overline{C} \cap \mathcal{L}$.

Soit $\mathcal{L}$ un ensemble de littéraux. On appelle $\neg\mathcal{L}$ l'ensemble de littéraux tel que $\forall l$ littéral de $PROP$ $l \in \mathcal{L} \Leftrightarrow \neg l \in \neg\mathcal{L}$.

Un littéral l est contrôlable ssi :
- à partir d'un monde où l est faux, l'agent peut par une de ses actions arriver dans un monde où l est vrai et ou seul l a changé ;
- à partir d'un monde où l est vrai, l'agent peut rester dans ce monde, soit en ne faisant rien, soit en gardant l vrai.

Si l est contrôlable, alors l'agent peut réaliser les actions $\texttt{do}(l)$ et $\texttt{keep}(l)$. On peut noter le fait qu'un agent puisse rendre un littéral vrai ne suffit pas à rendre ce littéral contrôlable. Il faut également que l'agent puisse conserver une situation dans laquelle le littéral est vrai.

Nous étendons maintenant cette définition aux propositions.

Définition 11.1.2. *Soient $w \in W$ et $w' \in W$. Notons $w' - w = \{l \ : \ w' \models l, \ w \models \neg l$ et l est un littéral$\}$. Une proposition φ est :*
- *contrôlable ssi $\forall w \in W \ w \models \neg\varphi \exists w' \in W \ w' \models \varphi$ et $(w' - w) \subseteq C$;*
- *influençable ssi $\exists w \in W \ w \models \neg\varphi \exists w' \in W \ w' \models \varphi$ et $(w' - w) \subseteq C$.*
  On dit alors que φ est influençable dans w.
- *non influençable ssi elle n'est pas influençable.*

Un monde $w \in W$ est un contexte pour une proposition influençable φ ssi φ est influençable dans w ou si $w \models \varphi$.

Les contextes d'une proposition influençable φ vont représenter les mondes qui sont soit un monde qui falsifie φ mais « à partir » duquel l'agent va pouvoir arriver à un monde qui vérifie φ, soit un monde qui vérifie φ.

Une proposition contrôlable est donc une proposition que l'agent est capable de rendre vraie à partir d'un monde où elle est fausse en ne changeant que la valeur de vérité de

littéraux contrôlables. Si l'agent est dans un monde où une proposition contrôlable est vraie, l'agent peut rester dans ce monde d'après la définition d'un littéral contrôlable. Notons également que si l'on considère une proposition contrôlable, tous les mondes sont des contextes pour celle-ci, car elle est également influençable.

Une proposition influençable est une proposition que l'agent peut rendre vraie à partir de certaines situations. Par exemple, si $a \in C$ et si $b \in \overline{C}$ alors $a \wedge b$ est influençable, mais pas contrôlable. Comme b n'est pas contrôlable, si b est faux, alors l'agent ne peut pas rendre $a \wedge b$ vraie, donc $a \wedge b$ n'est pas contrôlable. Par contre, si b est vrai et que a est faux, l'agent peut rendre $a \wedge b$ vraie donc $a \wedge b$ est influençable. Les contextes de $a \wedge b$ sont donc dans ce cas $\{a, b\}$ et $\{\neg a, b\}$.

On peut remarquer que nous avons respecté la classification que Boutilier avait donné aux propositions dans [13].

Cette notion de contrôlabilité va nous permettre de déterminer les actions disponibles pour un agent. Si φ est contrôlable et vraie dans un monde qui est un contexte pour φ, alors l'agent peut réaliser l'action $\texttt{keep}(\varphi)$. Si φ est contrôlable et fausse dans un monde qui est un contexte pour φ, alors l'agent peut réaliser l'action $\texttt{do}(\varphi)$. On peut donc s'affranchir de la définition d'opérateurs d'action pour l'agent.

Il est à noter que d'après la définition d'un contexte, on considère que l'agent pourra toujours conserver la valeur de vérité d'une proposition influençable. On considère alors qu'il peut le faire soit en ne faisant rien, soit en agissant avec `keep`. Il n'y a donc pas d'actions extérieures ni d'évolution extérieure du monde.

En écrivant une formule propositionnelle φ en DNF minimale, on peut facilement déterminer les implicants premiers de la formule. Soient $PI(\varphi)$ l'ensemble des implicants premiers de φ[1] et $PI(\neg\varphi)$ l'ensemble des implicants premiers de $\neg\varphi$.

Propriété 11.1.1. – *φ est contrôlable par a_i ssi $\forall \mathcal{U}_{\neg\varphi} = \bigcup_{p \in PI(\neg\varphi)} p$ tel que $\forall l \in lit(PROP)$ $l \in \mathcal{U}_{\neg\varphi} \Rightarrow \neg l \notin \mathcal{U}_{\neg\varphi}$ alors $\exists \mathcal{U}_\varphi \subseteq PI(\varphi)$ tel que*

1. *$\mathcal{U}_\varphi \neq \phi$;*
2. *$\forall \mathcal{C} \subseteq 2^{lit(PROP)}$ tel que $l \in \mathcal{C} \Rightarrow \neg l \notin \mathcal{C}$ et $\mathcal{U}_{\neg\varphi} \subseteq \mathcal{C}$, alors $\exists p' \in \mathcal{U}_\varphi$ tel que $\overline{C}_{a_i}(p') \subseteq \mathcal{C}$.*

– *φ est influençable par a_i ssi :*

1. *$\exists \mathcal{U}_{\neg\varphi} = \bigcup_{p \in PI(\neg\varphi)} p$ tel que $\forall l \in lit(PROP)$ $l \in \mathcal{U}_{\neg\varphi} \Rightarrow \neg l \notin \mathcal{U}_{\neg\varphi}$;*
2. *$\exists \mathcal{C} \subseteq 2^{lit(PROP)}$ tel que $l \in \mathcal{C} \Rightarrow \neg l \notin \mathcal{C}$ et $\mathcal{U}_{\neg\varphi} \subseteq \mathcal{C}$;*
3. *$\exists p' \in PI(\neg\varphi)$ tel que $\overline{C}_{a_i}(p') \subseteq \mathcal{C}$.*

11.2 CK-buts d'un agent

La définition d'un CK-but (pour Complete Knowledge but) proposée par Boutilier est la suivante : un CK-but est une proposition contrôlable et idéale (au sens de I) étant fixées les propositions ininfluençables par l'agent. Avec le nouveau modèle proposé dans la section précédente, nous allons raffiner cette définition. Les CK-buts vont être déterminés à partir des propositions qui sont vraies dans $Cl(KB)$ et pour lesquelles $Cl(KB)$ n'est pas

[1] On suppose qu'un implicant premier de φ est un ensemble de littéraux P tel $P \models \varphi$ et $\forall P' \subset P \;\; P' \not\models \varphi$. On notera $\bigwedge P = \bigwedge_{l \in P} l$.

un contexte. Cette condition nous garantit que ces propositions resteront vraies quelles que soient les actions que peut faire l'agent. De plus, un CK-but ne pourra être qu'une proposition pour laquelle $Cl(KB)$ est un contexte. En effet, dans ce cas :

- soit la proposition est fausse dans $Cl(KB)$ et l'agent peut changer la valeur de vérité de la proposition par une de ses actions ;
- soit la proposition est vraie dans $Cl(KB)$ et l'agent peut conserver la valeur de vérité de la proposition.

On définit l'ensemble des propositions non contextuelles de KB de la façon suivante :

Définition 11.2.1. *L'ensemble des propositions non contextuelles de KB est défini par :*

$$NC(KB) = \{\varphi \in Cl(KB) \ : \ Cl(KB) \text{ n'est pas un contexte pour } \neg\varphi\}$$

Les propositions non contextuelles de KB sont bien les propositions telles que $Cl(KB)$ n'est pas un contexte pour $\neg\varphi$. En effet, si KB n'est pas un contexte pour $\neg\varphi$, alors l'agent ne pourra pas par une de ses actions rendre φ fausse. φ restera donc vraie quoique fasse l'agent. On suppose ici que $NC(KB)$ est complet, i.e. que l'agent connaît la valeur de vérité de tous les littéraux pour lesquels $Cl(KB)$ n'est pas un contexte. Remarquons que Boutilier a étudié le cas où $UI(KB)$[2] n'est pas complet, et qu'il se ramène à des méthodes type *maximax* ou *maximin* de la théorie de la décision.

Définition 11.2.2. *Soit M un modèle de QDT. φ est un CK-but ssi $M \models I(\varphi|NC(KB))$ et $Cl(KB)$ est un contexte pour φ.*

Comme pour les buts idéaux, on peut définir l'ensemble des actions « atomiques » nécessaires à la réalisation des CK-buts.

Définition 11.2.3. *Soit M un modèle de QDT. Un ensemble de buts atomiques est un ensemble de littéraux $\mathcal{A} = \{l_1, \ldots, l_n\}$ contrôlables tels que :*

- *$\forall i \in \{1, \ldots, n\}$ $Cl(KB)$ est un contexte pour l_i ;*
- *pour tout CK-but φ dans M, $M \models NC(KB) \wedge \mathcal{A} \rightarrow \varphi$.*

On peut bien sûr définir les ensembles de buts atomiques grâce à une condition de suffisance :

Propriété 11.2.1. *Soit $\mathcal{A}$ un ensemble de littéraux tels que $Cl(KB)$ est un contexte pour tous ces littéraux. Alors $\mathcal{A}$ est un ensemble de buts atomiques ssi $\mathcal{A}$ est une condition suffisante pour $NC(KB)$.*

Exemple. *Reprenons l'exemple utilisé dans la section 10.2. On a un ensemble de préférences $\mathcal{P} = \{I(l), I(\neg l|\neg p)\}$ et une règle de normalité $\mathcal{N} = \{c \Rightarrow p\}$.*

Supposons que $KB = \{\neg p\}$ (la porte n'est pas poncée) et que l'agent contrôle les deux littéraux l et p (l'agent peut laquer la porte et la poncer). $\neg p \in Cl(KB)$ et p est contrôlable donc $Cl(KB)$ est un contexte pour p. On peut donc en déduire que $NC(KB) = \phi$. Dans ce cas, les CK-buts potentiels pour l'agent sont l et p. Comme $Cl(KB)$ est un contexte pour p, p est un CK-but pour l'agent. $Cl(KB)$ est également un contexte pour l, car l est contrôlable, donc :

- *soit la porte n'est pas laquée et alors l'agent peut la laquer et $Cl(KB)$ est un contexte pour l ;*

[2] I.e. l'ensemble des variables non-S-influençables et vraies dans KB.

– *soit la porte est laquée et alors l'agent peut ne rien faire et la laisser laquée, et* $Cl(KB)$ *est un contexte pour* l.

l est donc un CK-but pour l'agent. L'ensemble d'actions atomiques pour l'agent est donc $\{l, p\}$. *On peut remarquer que l'on retrouve le même résultat qu'avec l'approche de Boutilier.*

Supposons maintenant que l'ensemble de préférences soit $\mathcal{P} = \{I(l|p), I(\neg l|\neg p)\}$ *(le monde idéal n'est donc plus forcément un monde vérifiant l). Supposons que l'agent contrôle toujours l et p, et que* $KB = \{p\}$. *Dans ce cas, comme l'agent ne contrôle pas* $\neg p$, $Cl(KB)$ *n'est pas un contexte pour* $\neg p$, *donc* $NC(KB) = \{p\}$. *On peut donc en déduire que l'ensemble des buts atomiques pour l'agent est* $\{l, p\}$ *: il « doit » conserver la porte poncée et la laquer. Remarquons qu'en utilisant le formalisme de Boutilier, dans ce cas* $UI(KB) = \phi$ *car p est contrôlable, donc les seuls CK-buts que l'on peut déduire sont* $p \rightarrow l$ *et* $\neg p \rightarrow \neg l$. *Notre formalisme permet donc de représenter plus précisement le processus de décision de l'agent.*

11.3 Raisonnement normatif

Dans cette section, nous allons tout d'abord nous intéresser à un aspect normatif particulier : les Contrary-to-Duties (cf. chapitre 3). Nous avons en effet vu que Jones et Carmo proposent dans [22] un formalisme permettant de raisonner avec des CTDs utilisant un modèle des capacités et des décisions d'un agent. Nous montrons dans cette section que nous pouvons faire de même en utilisant le formalisme présenté dans les sections précédentes et que notre adaptation respecte les postulats de Carmo et Jones[3].

Le formalisme de Carmo et Jones repose sur la capacité de dériver deux types d'obligations : des obligations idéales et des obligations effectives. Les obligations idéales ne dépendent pas des décisions de l'agent alors que les obligations effectives sont calculées en fonction de ce que l'agent a décidé de faire ou de ne pas faire. On peut remarquer tout de suite qu'il y a un parallèle intuitif entre ces notions et les notions de but idéal et de CK-but. Nous allons nous attacher dans la suite à montrer que nous pouvons utiliser les notions de but idéal et de CK-but pour dériver des obligations idéales et effectives.

Nous montrerons ensuite que nous pouvons utiliser ces notions avec les autres types de normes définis dans le chapitre 5, puis nous définirons la notion de responsabilité d'un agent.

11.3.1 Dérivation d'obligations idéales et effectives

Pour pouvoir dériver des obligations idéales et effectives, nous avions proposé dans [34, 36, 38] d'étendre le modèle de contrôlabilité proposé par Boutilier en introduisant les notions suivantes :

– les propositions vraies et ininfluençables qui correspondent aux propositions nécessairement vraies indépendamment de l'agent dans le formalisme de Carmo et Jones ;
– les propositions contrôlables et fixées qui correspondent aux propositions nécessairement vraies du fait de l'agent dans le formalisme de Carmo et Jones ;
– les propositions contrôlables et non fixées qui correspondent aux propositions possiblement fausses du fait de l'agent dans le formalisme de Carmo et Jones.

[3] Nous discuterons lors de la conclusion de la validité de ces postulats.

Munis de ces trois définitions, nous pouvions alors déterminer des obligations idéales et effectives. Il nous parait plus judicieux maintenant que nous possédons un modèle de l'agent nous permettant de dériver quels sont ses buts d'utiliser ce modèle pour déterminer les décisions de l'agent plutôt que d'utiliser une distinction syntaxique comme nous l'avions fait précédemment. Pour cela, nous considérons trois ensembles :

- $\mathcal{R}$ un ensemble de formules de CO qui représente un ensemble de positions normatives.
  Par exemple, on pourra avoir $\mathcal{R} = \{I(\neg c), I(p|c), I(\neg p|\neg c)\}$ qui représente les positions normatives suivantes :
  - il ne doit pas y avoir de chien ;
  - s'il y a un chien, il doit y avoir un panneau ;
  - s'il n'y a pas de chien, il ne doit pas y avoir de panneau.
- $\mathcal{P}$ un ensemble de formules de CO qui représente les préférences de l'agent considéré. Par exemple, on pourra avoir $\mathcal{P} = \{I(c)\}$ qui représentent le fait que l'agent préfére qu'il y ait un chien ;
- $\mathcal{N}$ un ensemble de formules de QDT qui représente les règles de défauts qu'a l'agent à propos du monde.
  Par exemple, on pourra avoir $\mathcal{N} = \{c \Rightarrow p\}$ qui représente le fait que normalement, s'il y a un chien, il y a un panneau.

A partir de ces trois ensembles, nous allons pouvoir déterminer dans un premier temps les ensembles d'actions atomiques de l'agent. Ces actions vont représenter les « décisions » de l'agent et vont donc remplacer la notion de proposition influençable et fixée.

Dans un second temps, nous allons pouvoir dériver les obligations idéales et effectives de l'agent en utilisant le procédé décrit dans [36].

Pour cela, il nous faut spécifier dans un premier temps quels sont les ensembles représentant l'état du monde qui nous intéresse. On considère bien sûr que l'agent peut avoir une représentation incomplète du monde, en particulier des propositions dont il ne peut pas changer la valeur de vérité.

Par contre, nous considérons que pour pouvoir déterminer les obligations idéales et effectives qui s'imposent à l'agent, nous avons besoin de connaître la valeur de vérité de tous les atomes du langage. On introduit donc deux ensembles :

Définition 11.3.1. *Soit KB_N une valuation complète des atomes de $PROP$. KB_N est représenté sous la forme $\{a_1, a_2, \ldots, a_n\}$. L'état du monde pour l'agent est un ensemble KB tel que $KB \subseteq KB_N$.*

L'ensemble des actions atomiques de l'agent va représenter les décisions que l'agent a pris en fonction de ses préférences et de sa connaissance (peut-être partielle) du monde.

Définition 11.3.2. *On appelle Dec l'ensemble des actions atomiques de l'agent dérivé à partir de $\mathcal{P}$, de $\mathcal{N}$ et d'un état KB du monde suivant le procédé expliqué en section 11.2.*

On remarquera que comme précédemment, on considère que l'agent a une connaissance complète (au sens donné dans la section 11.2) du monde.

Définition 11.3.3. $NC_a(KB_N) = Dec \cup NC(KB_N)$

$NC_a(KB_N)$ (l'indice a correspond à l'agent) va donc contenir les littéraux qui sont vrais dans Dec et les littéraux vrais dans KB_N que l'agent ne peut pas modifier. Remarquons que $NC_a(KB_N)$ est consistant car les littéraux contenus dans $NC(KB_N)$ ne peuvent

pas appartenir à Dec. On peut également remarquer que cette consistance est également obtenue dans le cas où $KB \not\subseteq KB_N$ (i.e. l'agent a une représentation du monde qui est différente de la réalité). En effet, si on suppose qu'il existe un littéral l tel que $l \in Dec$, alors cela signifie que l est contrôlable et dans ce cas $\neg l \notin NC(KB)$ d'après la définition 11.2.1.

Nous définissons ensuite les obligations idéales, effectives et les violations :

Définition 11.3.4. – *les obligations idéales de l'agent sont définies par :*

$$O_i \equiv_{def} \{O_i(\varphi) \ : \ \mathcal{R} \models I(\varphi|NC(KB_N)) \text{ et } KB_N \text{ est un contexte pour } \varphi\}$$

– *les obligations effectives de l'agent sont définies par :*

$$O_a \equiv_{def} \{O_a(\varphi) \ : \ \mathcal{R} \models I(\varphi|NC_a(KB_N)), \ KB_N \text{ est un contexte pour } \varphi\}$$

– *les violations sont définies par :*

$$viol(\varphi) \equiv_{def} O_i(\varphi) \text{ et } \vdash KB_N \rightarrow \neg\varphi$$

Contrairement à ce qui a été fait dans [36], les obligations effectives portent également sur des propositions qui peuvent être contrôlables et fixées. On considère donc ici que l'agent est obligé de respecter ses décisions. Ceci est normal, puisqu'on se sert de ses décisions (i.e. de Dec) pour calculer ses obligations effectives. On peut remarquer que dans [22], Carmo et Jones ne peuvent pas dériver ce genre d'obligation : une obligation effective porte sur une proposition que l'agent n'a pas « fixée ».

Propriété 11.3.1. *Le formalisme proposé ici respecte les postulats de Carmo et Jones.*

Il suffit juste de vérifier qu'on ne peut pas dériver d'idiotie pragmatique avec ce formalisme. Dans le scénario du chien, l'idiotie pragmatique revient à dériver qu'il est obligatoire de ne pas avoir de chien et obligatoire d'avoir un panneau. Si l'on essaye de dériver une obligation en utilisant $\neg c$, alors comme le monde $\{\neg c, \neg p\}$ est préféré au monde $\{\neg c, p\}$, on dérivera une obligation de ne pas avoir de panneau. Si l'on essaye de dériver une obligation en utilisant p, on ne pourra pas deriver d'obligation de ne pas avoir de chien, car les mondes $\{c, p\}$ et $\{\neg c, p\}$ sont incomparables. On évite donc l'idiotie pragmatique.

Exemple. *Reprenons l'exemple du chien donné dans [36]. On a donc* $\mathcal{R} = \{I(\neg c), I(p|c), I(\neg p|\neg c)\}$. *On peut également supposer que* $\mathcal{N} = \{c \Rightarrow p\}$, *i.e. l'agent suppose que normalement, s'il y a un chien, il y a un panneau.*

Supposons que $KB_N = \{c, p\}$, *i.e. il y a un chien et un panneau. Supposons également que* $KB = \{p\}$ *: l'agent sait qu'il y a un panneau, mais ne sait pas qu'il y a un chien. Supposons enfin que* c, $\neg c$, p *et* $\neg p$ *soient contrôlables.*

Si $\mathcal{P} = \{I(c)\}$, *alors* $Dec = \{c\}$ *: l'agent a décidé de garder son chien. On peut remarquer que l'agent n'a rien décidé à propos du panneau (p est alors contrôlable et non fixée en utilisant le formalisme de [36]).*

Dans ce cas, on dérive facilement $O_i(\neg c)$ *et* $O_i(\neg p)$. *De plus, on a* $viol(\neg c)$ *et* $viol(\neg p)$ *: les obligations de ne pas avoir de chien et de panneau sont toutes les deux violées.*

Par contre, on a $NC_a(KB_N) = \{c\}$, *car* $Dec \models c$. *Donc on peut en déduire deux obligations effectives qui sont* $O_a(p)$ *et* $O_a(c)$ *: l'agent est obligé de garder son chien (puisqu'il l'a décidé) et le panneau.*

Si $\mathcal{P} = \{I(\neg c)\}$, alors $Dec = \{\neg c\}$: l'agent a décidé de se débarrasser de son chien. Dans ce cas, on a toujours les mêmes obligations idéales et elles sont toutes les deux violées. $NC_a(KB_N) = \{\neg c\}$, donc l'agent a l'obligation effective de se débarrasser du chien et d'enlever le panneau.

On peut voir maintenant les obligations idéales comme étant les obligations imposées par une autorité extérieure et les obligations effectives étant les obligations déterminées suivant les décisions de l'agent. Ces obligations peuvent en particulier correspondre à des cas dégradés de réglementation (cf. les CTDs). Plus précisement, les obligations effectives correspondent à des obligations que l'agent s'impose à lui-même (de par ses décisions). En particulier, l'agent aura pour obligation effective de « réaliser » toutes ses décisions.

11.3.2 Application aux autres normes

Le formalisme présenté dans la section précédente pour raisonner avec des CTDs peut aussi s'appliquer aux autres types de normes que nous avions définis dans le chapitre 5. Rappelons que nous avons présenté deux autres types de normes : les normes non conditionnelles et les normes avec exception. Pour ces deux types de normes, il n'y a pas besoin de définir des obligations effectives, puisqu'il n'existe qu'un seul niveau d'obligation, contrairement aux CTDs.

Normes non conditionnelles

Les normes non conditionnelles sont des formules du type $I(\alpha|\top)$ signifiant que α est obligatoire quelque soit le contexte. Dans ce cas, toute formule de ce type peut être interprétée comme étant une obligation idéale.

Définition 11.3.5. *Soit $\mathcal{R}$ un ensemble de normes non conditionnelles. Les obligations idéales de l'agent par rapport à $\mathcal{R}$ sont définies par :*

$$O_i \equiv_{def} \{O_i(\varphi) \ : \ \mathcal{R} \models I(\varphi) \text{ et } KB_N \text{ est un contexte pour } \varphi\}$$

On remarquera que l'on considère encore les formules dont l'agent peut changer la valeur de vérité (i.e. pour lesquelles KB_N est un contexte). En effet, seules les obligations concernant des formules dont l'agent peut changer la valeur de vérité nous intéressent, pour pouvoir déterminer s'il est « responsable » d'une violation.

Remarquons également que l'opérateur I n'est pas clos par la règle de normalité $\Rightarrow$: si on a $I(\alpha)$ et $\alpha \Rightarrow \beta$, alors on n'a pas $I(\beta)$ (en effet, les deux opérateurs sont définis à partir de préordres indépendants). C'est intuitivement correct.

Normes avec exception

Les normes avec exception sont des cas plus difficiles à gérer : elles dépendent en effet du contexte. Par exemple, on peut avoir $I(\alpha)$ qui précise que α est obligatoire, mais également $\neg I(\alpha|\beta)$ qui signifie que si β est vraie, alors α n'est plus obligatoire. On est donc obligé avec ces types de normes d'utiliser KB_N pour déterminer si une obligation est en vigueur ou pas.

Définition 11.3.6. *Soit $\mathcal{R}$ un ensemble de normes avec exception. Les obligations idéales de l'agent par rapport à $\mathcal{R}$ sont définies par :*

$$O_i \equiv_{def} \{O_i(\varphi) \ : \ \mathcal{R} \models I(\varphi|NC_a(KB_N)) \text{ et } KB_N \text{ est un contexte pour } \varphi\}$$

Pour dériver des obligations idéales à partir d'un ensemble de normes avec exception, on utilise le même principe que pour les CTDs. Cela est normal, car on doit tenir compte du contexte, en particulier des formules dont l'agent ne peut pas changer la valeur de vérité, pour dériver des obligations à partir d'obligations avec exception.

Le principal problème qui se pose avec cette approche, c'est que nous ne dérivons pas les obligations idéales de la même façon suivant le type de norme considéré. Il faut partitionner les phrases normatives en trois ensembles : les obligations non conditionnelles, les obligations avec exception et les Contrary-to-Duties.

11.3.3 Responsabilités

Pour l'instant, nous ne sommes capables que de déterminer les obligations idéales et effectives de l'agent et quelles sont les obligations qui ont été violées. Il serait également intéressant de pouvoir déterminer si l'agent est responsable de la violation d'une obligation. Par détermination des responsabilités, nous n'entendons pas pouvoir déterminer qui est la cause de la violation d'une obligation, mais si l'agent, de par ses décisions, fera perdurer une violation d'une obligation idéale. On peut voir cette notion comme une responsabilité « future ». Par exemple, dans l'exemple précédent, comme l'agent a décidé de garder le chien, on peut considérer qu'il sera responsable de la violation de $O_i(\neg c)$ si le chien reste.

Définition 11.3.7. *L'agent est responsable de la violation d'une obligation idéale $O_i(\varphi)$ (noté $Resp(\varphi)$) ssi $viol(\varphi)$ est vraie et $Dec \models \neg\varphi$).*

On suppose bien sûr que les agents vont respecter leurs obligations effectives. Par exemple, dans l'exemple précédent, si l'agent a décidé de garder son chien, il est tenu pour responsable de la violation de $O_i(\neg c)$.

Un problème peut se poser dans ce procédé de détermination des responsabilités lorsque l'agent n'a pas une connaissance complète du monde, en particulier des propositions qu'il ne contrôle pas. Il se peut en effet dans ce cas qu'un agent prenne des décisions en fonction d'un état du monde qui est faux. Mais même dans ce cas, nous considérons que l'agent est responsable de la violation de l'obligation idéale.

11.3.4 Position de l'agent vis-à-vis de la réglementation

Nous avons supposé jusqu'à présent que l'agent avait une attitude neutre par rapport à la réglementation : il ne calculait ses buts qu'en fonction de ses préférences. On peut très bien imaginer un agent qui calcule ses buts en tenant compte de la réglementation, en la privilégiant par exemple. Si l'on considère une réglementation $\mathcal{R}$ et un ensemble de préférences $\mathcal{P}'$ de l'agent, on peut distinguer trois attitudes :

- *égoiste* qui correspond au cas étudié précédemment, i.e. $\mathcal{P} = \mathcal{P}'$. L'agent prend ses décisions indépendamment de la réglementation ;
- *vertueux* qui correspond à un agent qui prendra ses décisions en privilégiant la réglementation. Pour obtenir $\mathcal{P}$, on fusionne[4] $\mathcal{R}$ et $\mathcal{P}'$ en donnant la priorité à $\mathcal{R}$;

[4]Par fusion de deux ensembles de formules de CO, on entend les modèles obtenus par fusion des modèles de chacun des ensembles.

– *neutre* qui correspond à un agent qui prendra ses décisions en privilégiant ses préférences, mais qui respectera la réglementation si c'est possible. Dans ce cas, $\mathcal{P}$ est obtenu en fusionnant $\mathcal{R}$ et $\mathcal{P}'$ en donnant priorité à $\mathcal{P}'$.

Le fait d'utiliser un formalisme commun (CO) pour exprimer à la fois une réglementation et des préférences nous permet donc de qualifier l'attitude de l'agent vis-à-vis de la réglementation dans le processus de détermination de ses buts.

11.4 Conclusion

Nous avons développé un modèle d'agent exécutant simple, qui permet de déterminer quels sont les CK-buts d'un agent à partir de ses préférences, de sa connaissance du monde et de ses compétences. Nous nous sommes également intéressés aux aspects normatifs en nous avons en particulier montré que notre formalisme répond aux critères émis par Carmo et Jones pour raisonner avec des normes de types Contrary-to-Duty. Nous allons maintenant présenter un formalisme permettant de distribuer des exigences sur un ensemble d'agents exécutants.

Chapitre 12

Distribution d'exigences

Notre objectif dans ce chapitre est de formaliser la notion de distribution d'exigences. Nous considérons que nous disposons d'un ensemble d'exigences cohérent exprimé avec CO qui représente les propriétés que doit satisfaire l'artefact à construire et d'un ensemble d'agents exécutants possédant leurs propres capacités et leurs propres préférences. Le processus de distribution doit être vu comme un processus géré par une autorité centrale qui dispose de toutes ces données. Cette autorité doit alors affecter à chaque agent exécutant un certain nombre de tâches qui sont censées valider une partie des exigences. Ces tâches sont déterminées en fonction des compétences de chacun des agents, mais également de ses engagements et de celui des autres agents.

Intuitivement, l'autorité centrale dispose de l'ensemble d'exigences, des compétences et des engagements des agents. À partir de ces notions et de stratégies que nous définirons dans la suite du chapitre, elle peut affecter à chaque agent une ou plusieurs tâches dépendant des engagements des autres agents et de ses propres engagements. Par exemple, si l'on est dans une optique où l'on cherche à minimiser le nombre d'agents effectuant la même tâche, si un agent s'engage à réaliser un tâche, on peut considérer que les autres agents ayant également la capacité de réaliser cette tâche peuvent en être dispensés.

Remarquons que nous nous affranchissons des problèmes de communication pouvant exister entre les différents agents exécutants. En particulier, nous n'étudions pas les cas de négociations entre des agents exécutants qui partagent des ressources communes (cf. [81, 82, 64]). Ces problèmes sont occultés par la présence de l'entité centrale qui régit le processus de distribution.

Dans ce chapitre, nous allons tout d'abord étendre le modèle d'agent exécutant proposé dans le chapitre précédent à un groupe d'agents, puis nous nous intéresserons à la détermination des buts effectifs de chaque agent et enfin nous détaillerons quelques stratégies et la manière de les combiner. Une partie de ces travaux ont été présentés dans [59], [61], [35] et [39].

Nous considérons que nous disposons d'un ensemble fini d'agents exécutants noté $\mathcal{A} = \{a_1, \ldots, a_n\}$. Pour chacun des agents a_i du groupe, on connait l'ensemble C_{a_i} des littéraux contrôlables par l'agent. Dans toute la suite du chapitre, nous considérons également que nous disposons d'un ensemble Σ de formules de CO représentant les préférences imposées au groupe d'agents traduisant les exigences finales concernant l'objet à construire.

12.1 Extension au cas multi-agents

Supposons que nous ayons le scenario suivant qui concerne l'état d'une porte. On dispose d'un groupe de deux agents $\{a_1, a_2\}$. Les préférences que nous imposons au groupe sont les suivantes :

- si la porte est poncée, on préfére que la porte soit laquée mais pas recouverte de papier ;
- si la porte n'est pas poncée, on préfére qu'elle soit recouverte de papier, mais pas laquée.

La modélisation en formules de CO de ce scénario est la suivante : $\Sigma = \{I(l \wedge \neg r|p), I(r \wedge \neg l|\neg p)\}$.

Si l'on reprend le formalisme présenté dans le chapitre précédent, pour pouvoir déterminer les CK-buts du groupe $\{a_1, a_2\}$, il faut connaitre les propositions influençables par le groupe d'agents. Nous devons donc en premier lieu étendre la notion de contrôlabilité d'un littéral à un groupe d'agents à partir de la contrôlabilité individuelle de chacun des agents. Nous déterminerons alors quels sont les CK-buts du groupe d'agents.

12.1.1 Extension de la contrôlabilité

Pour chacun des agents, nous avons partitionné les littéraux de $PROP$ en deux classes : les littéraux contrôlables par l'agent et les littéraux incontrôlables par l'agent. Nous allons étendre cette proposition au cas multi-agents en émettant deux hypothèses à propos des domaines de contrôlabilité des agents. La première hypothèse que nous faisons exprime le fait que tout agent contrôle au moins un littéral (sinon cet agent n'est pas à proprement parler un agent exécutant) :

Hypothèse 12.1.1. $\forall a_i \in \mathcal{A} \quad C_{a_i} \neq \phi$

La seconde hypothèse que nous faisons porte sur les domaines de contrôlabilité des agents :

Hypothèse 12.1.2. *Les domaines de contrôlabilité des agents ne sont pas forcément disjoints.*

Il se peut très bien que deux agents différents soient compétents pour réaliser la même action. Ils entrent alors en « concurrence » pour réaliser cette action.

On peut alors chercher à relier les notions de propositions contrôlables, influençables ou non-influençables par un agent à la notion de proposition contrôlable ou non-contrôlable par un groupe d'agents.

Définition 12.1.1. *Soit $lit(PROP)$ l'ensemble des littéraux du langage. L'ensemble des littéraux contrôlables par le groupe d'agents est $C = \bigcup_{a_i \in \mathcal{A}} C_{a_i}$ et l'ensemble des littéraux non-contrôlables par le groupe d'agents est $\overline{C} = lit(PROP) - C$.*

L'extension aux propositions est la même que dans la définition 11.1.2.

Prenons un exemple : un groupe d'agents $\{a_1, a_2\}$ est tel que p soit contrôlable par a_1 et que r soit contrôlable par a_2. Quelle est la « contrôlabilité » de la proposition $(p \vee q) \wedge (r \vee s)$?

Dans ce cas, comme p est contrôlable par a_1 et r est contrôlable par a_2, $(p \vee q) \wedge (r \vee s)$ est contrôlable par $\{a_1, a_2\}$. En effet, les mondes qui vont falsifier a_2, $(p \vee q) \wedge (r \vee s)$ sont

des mondes qui vont vérifier $\neg p$ ou qui vont vérifier $\neg r$. Comme p et r sont contrôlables par le groupe $\{a_1, a_2\}$, le groupe pourra toujours rendre ou garder p et q vraies.

Par contre, $(p \wedge q) \vee (r \wedge s)$ est seulement influençable : les mondes qui vérifient $\neg q \wedge \neg r$ ne sont pas des contextes pour $(p \wedge q) \vee (r \wedge s)$. Les contextes de $(p \wedge q) \vee (r \wedge s)$ sont les mondes qui vérifient $(q \vee r) \wedge (\neg p \vee \neg s)$. En effet, à partir des mondes qui vérifient $(q \vee r) \wedge (\neg p \vee \neg s)$, on va pouvoir rendre $(p \wedge q) \vee (r \wedge s)$ vraie en rendant p vraie et r vraie.

12.1.2 CK-buts du groupe d'agents

Comme nous avons défini la notion d'influençabilité d'une proposition par le groupe d'agents $\mathcal{A}$, nous pouvons introduire la notion de CK-buts pour $\mathcal{A}$. Il nous faut préciser deux éléments : la définition de KB et celle de $NC(KB)$.

La définition de KB dans le cas multi-agents est délicate. Pour un agent, KB représente les croyances qu'il a à propos du monde réel. Il se peut donc très bien que deux agents aient des croyances contradictoires et donc des KB différents. Dans ce cas, il parait difficile de déterminer un ensemble de croyances cohérent pour l'ensemble des agents du groupe[1]. Nous allons donc supposer que tous les agents de $\mathcal{A}$ ont la même représentation du monde.

Hypothèse 12.1.3. *Les agents de $\mathcal{A}$ ont la même représentation du monde. Cette représentation du monde est caractérisée par un ensemble fini et cohérent de formules propositionnelles noté KB.*

Le groupe d'agent a une représentation du monde qui est également KB.

Nous pouvons maintenant définir l'ensemble des propositions non contextuelles de KB de la façon suivante (la définition est identique au cas mono-agent) :

Définition 12.1.2. *L'ensemble des propositions non contextuelles de KB est défini par :*

$$NC(KB) = \{\varphi \in Cl(KB) \ : \ KB \text{ n'est pas un contexte pour } \neg\varphi\}$$

Comme dans le cas mono-agent, on supposera ici que l'agent connaît la valeur de vérité de tous les littéraux pour lesquels $Cl(KB)$ n'est pas un contexte. La définition d'un CK-but pour $\mathcal{A}$ est donnée ci-après :

Définition 12.1.3. *φ est un CK-but pour $\mathcal{A}$ ssi $\Sigma \models I(\varphi | NC(KB))$ et KB est un contexte pour φ par rapport à $\mathcal{A}$.*

Il faut pouvoir maintenant déterminer quelles sont les tâches que chaque agent doit accomplir pour que les buts du groupe soient effectivement réalisés. Reprenons l'exemple donné en début de section : dans le cas où a_1 peut laquer la porte ou la recouvrir de papier et que a_2 peut la poncer, on voit que la tâche de a_1 peut dépendre des *engagements* de a_2 à propos du ponçage de la porte. Si a_2 s'engage à poncer la porte et qu'il peut le faire, a_1 doit la laquer.

12.1.3 Les engagements des agents

Étant donné un littéral contrôlable par un agent, celui ci peut exprimer trois positions vis-à-vis de ce littéral :

[1] On notera que l'on peut alors faire appel à des méthodes de fusion de bases de croyances pour déterminer les croyances du groupe d'agents.

- l'agent peut exprimer qu'il va faire une action qui va rendre ou garder ce littéral vrai. Nous dirons que l'agent s'engage à réaliser le littéral ;
- l'agent peut exprimer qu'il ne va faire aucune action pouvant rendre ce littéral vrai. Nous dirons que l'agent s'engage à ne pas réaliser le littéral ;
- enfin, il se peut que l'agent n'exprime rien à propos du littéral en question. Nous dirons alors que l'agent ne s'engage ni à réaliser le littéral, ni à ne pas le réaliser.

Pour représenter les engagements de chaque agent a_i, nous allons utiliser trois sous-ensembles de C_{a_i} : Com_{+,a_i}, Com_{-,a_i} et P_{a_i}. On les définit de la façon suivante :

- si l est un littéral, si l est contrôlable par a_i et que $l \in Com_{+,a_i}$ cela signifie que « l'agent a_i s'engage à réaliser l » ;
- si l est un littéral, si l est contrôlable par a_i et que $l \in Com_{-,a_i}$ cela signifie que « l'agent a_i s'engage à ne pas réaliser l ».
- $P_{a_i} = C_{a_i} - (Com_{+,a_i} \cup Com_{-,a_i})$ est l'ensemble des littéraux contrôlables par a_i et pour lesquels a_i ne s'engage à rien (i.e. a_i ne s'engage ni à les réaliser, ni à ne pas les réaliser).

Le tableau suivant résume la modélisation que nous proposons des phrases du type « l'agent a_i s'engage à réaliser l » :

Phrase à modéliser	Modélisation
a_i s'engage à réaliser l	$l \in Com_{+,a_i}$
a_i s'engage à ne pas réaliser l	$l \in Com_{-,a_i}$
a_i ne s'engage ni à réaliser l ni à ne pas réaliser l	$l \in P_{a_i}$

Nous imposons tout d'abord deux contraintes sur ces deux ensembles.

Contraintes 12.1.1. $\forall a_i \in \mathcal{A}$ Com_{+,a_i} *est cohérent.*

Contraintes 12.1.2. $\forall a_i \in \mathcal{A}$ $Com_{+,a_i} \cap Com_{-,a_i} = \phi$

Ces deux contraintes expriment une sorte de cohérence du modèle de l'agent. La contrainte 1 exprime le fait qu'un agent ne s'engage pas à réaliser l et $\neg l$ à la fois. La contrainte 2 exprime le fait qu'un agent ne peut pas s'engager à la fois à réaliser l et à ne pas réaliser l.

Notons que cette notion d'engagement pour un agent a_i peut être reliée à l'ensemble d'actions atomiques Dec_{a_i} introduit dans le chapitre précédent. Dec_{a_i} contient les actions atomiques de l'agent a_i dérivées à partir de $\mathcal{P}_{a_i}$, de C_{a_i} et de KB. Ce sont les buts de l'agent étant données ses propres préférences. On peut considérer en simplifiant le processus de détermination des engagements (cf. formalismes de types BDI présentés en section 10.1) que ce sont les actions que l'agent va s'engager à réaliser. On a donc $Com_{+,a_i} = Dec_{a_i}$. La cohérence de Com_{+,a_i} est donc verifiée implicitement. Par contre, la définition de Com_{-,a_i} à partir de Dec_{a_i} est impossible. En effet, Com_{-,a_i} représente les littéraux que l'agent s'engage à ne pas rendre ou garder vrai. Il faudrait pour cela disposer d'un deuxième ensemble de décision dérivé à partir d'un deuxième ensemble de préférences $\mathcal{P}'_{a_i}$ représentant les préférences de l'agent par rapport à des buts qu'il voudrait ne pas voir réalisés. Nous ne poursuivrons pas la discussion plus loin dans ce mémoire.

Remarque. *Les notions précédentes ont été modélisées en logique modale dans [59] en utilisant deux familles d'opérateurs modaux : C_i et E_i, $i \in \{1...n\}$. L'opérateur E_i est l'opérateur stit ([9], [69]). $E_i\varphi$ exprime le fait que l'agent i fait en sorte que φ soit vraie. Il est défini par l'axiomatique suivante :*

$$
\begin{array}{rl}
(C) & E_i\varphi \wedge E_i\psi \rightarrow E_i(\varphi \wedge \psi) \\
(T) & E_i\varphi \rightarrow \varphi \\
(4) & E_i\varphi \rightarrow E_iE_i\varphi \\
(RE) & \vdash (\varphi \leftrightarrow \psi) \implies \vdash (E_i\varphi \leftrightarrow E_i\psi)
\end{array}
$$

L'opérateur C_i est un opérateur de type KD (cf. [25]) et $C_i\varphi$ dsignifie que l'agent a_i s'engage à rendre φ vraie. Il est défini par l'axiomatique suivante :

$$
\begin{array}{rl}
(K) & C_i\varphi \wedge C_i(\varphi \rightarrow \psi) \rightarrow C_i\psi \\
(D) & C_i\neg\varphi \rightarrow \neg C_i\varphi \\
(Nec) & \dfrac{\vdash \varphi}{\vdash C_i\varphi} \\
(MP) & \dfrac{\vdash \varphi \quad \vdash \varphi \rightarrow \psi}{\vdash \psi}
\end{array}
$$

Etant donné un atome l et étant donné ces opérateurs, un agent a_i est confronté à trois positions :

- *C_iE_il : l'agent s'engage à faire en sorte que l soit vraie ;*
- *$C_i\neg E_il$: l'agent s'engage à ne pas faire en sorte que l soit vraie ;*
- *$\neg C_iE_il \wedge \neg C_i\neg E_il$: l'agent ne s'engage ni à faire en sorte que l soit vraie, ni à ne pas faire en sorte que l soit vraie.*

Dans ce mémoire, nous oublions l'axiomatique et nous ne considérons que les trois ensembles d'atomes suivants :

- *Com_{+,a_i} qui correspond à $\{l \ : \ C_iE_il\}$;*
- *Com_{-,a_i} qui correspond à $\{l \ : \ C_i\neg E_il\}$;*
- *P_i qui correspond à $\{l \ : \ \neg C_iE_il \wedge \neg C_i\neg E_il\}$.*

On peut vérifier d'après l'axiomatique précédente que l'on peut dériver le théorème $\vdash \neg(C_iE_il \wedge C_iE_i\neg l)$. Ceci explique la contrainte 12.1.1. De plus, on peut également dériver $\vdash \neg(C_i\neg E_il \wedge C_iE_il)$. Ceci explique la contrainte 12.1.2.

Définition 12.1.4. *Nous appellerons $Com_{+,\mathcal{A}}$ l'ensemble des engagements positifs de tous les agents, soit :*

$$Com_{+,\mathcal{A}} = \bigcup_{a_i \in \mathcal{A}} Com_{+,a_i}$$

Nous appellerons $Com_{-,\mathcal{A}}$ l'ensemble des engagements « négatifs » des agents :

$$Com_{-,\mathcal{A}} = \{l \in KB \ : \ \forall a_i \in \mathcal{A} \ \neg l \text{ contrôlable par } a_i \Rightarrow \neg l \in Com_{-,a_i}\}$$

La signification de $Com_{-,\mathcal{A}}$ est donc la suivante : si tous les agents qui peuvent contrôler un littéral l s'engagent à ne pas réaliser l et que $\neg l \in KB$, on considérera que $\neg l$ restera vrai. Cela revient à supposer qu'il n'y a pas d'intervention extérieure.

Une hypothèse que nous allons poser sur les engagements des agents est la suivante : *tout CK-but du groupe $\mathcal{A}$ est cohérent avec l'union de $Com_{+,\mathcal{A}}$ et de $Com_{-,\mathcal{A}}$.*

Hypothèse 12.1.4. *Pour toute formule φ telle que $\Sigma \models I(\varphi|NC(KB))$ et KB est un contexte pour φ par rapport à $\mathcal{A}$, alors :*

$$Com_{-,\mathcal{A}} \cup Com_{+,\mathcal{A}} \cup \{\varphi\} \text{ est cohérent.}$$

Poser cette hypothèse permet d'éliminer des cas problématiques comme par exemple :
- le cas où deux agents qui contrôlent l'un l et l'autre $\neg l$ s'engagent l'un à réaliser l et l'autre à réaliser $\neg l$ (i.e. $Com_{+,\mathcal{A}}$ incohérent) ;
- le cas où un littéral est vrai dans KB et le restera car les agents du groupe qui pouvaient le rendre faux ne s'y sont pas engagés alors que ce littéral contredit les CK-buts du groupe ;
- le cas où les engagements positifs et négatifs des agents contredisent un CK-but du groupe.

Concrétement, si on veut « distribuer » des buts à un groupe d'agents, il faut d'abord vérifier la cohérence des engagements des agents avec les CK-buts du groupe. Si la cohérence n'est pas vérifiée, les agents doivent revoir leurs engagements. La gestion de conflits éventuels entre les engagements des agents exécutants et les buts du groupe est donc occultée dans ce travail.

12.2 Buts effectifs d'un agent exécutant

Si l'hypothèse 12.1.3 est vérifiée, on est sûr que les engagements des agents ne vont pas à l'encontre des CK-buts du groupe. Il reste maintenant à déterminer quels seront les buts effectifs de chaque agent. On ne parle pas ici de CK-but pour chaque agent, car comme on l'a vu précedemment, les buts de chaque agent ne doivent pas dépendre que de $NC(KB)$ (les faits « non-contrôlables » par $\mathcal{A}$) mais également des engagements des autres agents et de ses propres engagements. Il paraît donc légitime pour un agent $a_i \in \mathcal{A}$ de dériver ses buts effectifs à partir :
- des propositions de KB pour lesquelles KB n'est pas un contexte par rapport à $\mathcal{A}$, soit $NC(KB)$;
- de l'ensemble des engagements « positifs » des agents, soit $Com_{+,\mathcal{A}}$;
- de l'ensemble des engagements « négatifs » des agents , soit $Com_{-,\mathcal{A}}$.

Nous allons appeler cet ensemble *ensemble de données* et le noter $D(KB)$. Cet ensemble servira en partie *conditionnelle* des préférences pour déterminer les buts effectifs des agents.

Définition 12.2.1. *On définit :*

$$D(KB) = NC(KB) \cup Com_{+,\mathcal{A}} \cup Com_{-,\mathcal{A}}$$

Propriété 12.2.1. *$D(KB)$ est un ensemble cohérent.*

Notons que cette définition est tout à fait arbitraire : on aurait très bien pu dériver les buts effectifs de chaque agent en fonction de $NC(KB)$ seulement. Mais dans l'optique de distribution des exigences et pour qualifier notre processus de distribution dépendant d'une entité centrale connaissant les engagements des agents, il nous parait plus judicieux d'utiliser les engagements des agents pour déterminer les buts effectifs de chaque agent.

Nous introduisons maintenant la notion de *but effectif* pour un agent :

Définition 12.2.2. *φ est un but effectif pour a_i, noté $EGoal_{a_i}(\varphi)$ ssi :*

$$\Sigma \models I(\varphi|D(KB)) \text{ et } Cl(KB) \text{ est un contexte pour } \varphi \text{ par rapport à } a_i$$

Comme on utilise l'opérateur $I(-|-)$, on est sûr qu'un agent n'aura pas de buts contradictoires. Comme dans le cas mono-agent, on peut définir un ensemble d'actions atomiques effectives en étendant trivialement la définition 10.2.10.

Il est également intéressant de pouvoir définir une notions de *non satisfaction* d'un CK-but φ.

Définition 12.2.3. *Soit φ un CK-but de $\mathcal{A}$. On dit que φ n'est pas satisfaite (noté $Nonsat(\varphi)$) ssi :*

$$\bigcup_{a_i \in \mathcal{A}} \{\varphi' \ : \ EGoal_{a_i}(\varphi')\} \nvdash \varphi$$

Nous pouvons énoncer maintenant quelques propriétés du formalisme.

Propriété 12.2.2. *Soit l un littéral de $PROP$. Si l est un CK-but de $\mathcal{A}$, alors $\exists a_i \in \mathcal{A}$ tel que l'on a $EGoal_{a_i}(l)$.*

Cette propriété signifie que si l est un littéral et est un CK-but du groupe d'agent, alors il existe au moins un agent qui aura pour but effectif de réaliser l. Un corollaire immédiat de cette propriété est le suivant[2] :

Propriété 12.2.3. *Soit $l_1, \ldots, l_n$ n littéraux de $PROP$ tels que $l_1 \wedge \ldots \wedge l_n$ soit un CK-but de $\mathcal{A}$. Alors $\forall i \in \{1, \ldots, n\}$ $\exists a_{l_i} \in \mathcal{A}$ tel que l'on ait $EGoal_{a_{l_i}}(l_i)$.*

Pour tous les CK-buts qui sont des littéraux ou des conjonctions de littéraux, le processus défini précédemment assigne aux agents des buts effectifs tels que ces CK-buts soient atteints.

12.3 Exemple

Reprenons l'exemple précédent. On dispose d'un groupe de deux agents $\{a_1, a_2\}$. Les préférences du groupe sont :

- si la porte est poncée, on préfére que la porte soit laquée mais pas recouverte de papier ;
- si la porte n'est pas poncée, on préfére qu'elle soit recouverte de papier mais pas laquée.

La modélisation grâce à CO de ce scénario est la suivante : $\Sigma = \{I(l \wedge \neg r|p), I(r \wedge \neg l|\neg p)\}$. Pour chaque modèle de Σ :

- $I(l \wedge \neg r|p)$ signifie qu'il existe un monde vérifiant p et pour lequel tous les mondes préférés vérifient $p \rightarrow l \wedge \neg r$;
- $I(r \wedge \neg l|\neg p)$ signifie qu'il existe un monde vérifiant $\neg p$ et pour lequel tous les mondes préférés vérifient $\neg p \rightarrow r \wedge \neg l$;

Nous supposerons qu'il n'y a pas de règles par défaut, donc $Cl(KB) = KB$. Etudions différents scénarios :

1. Supposons que $KB = \{p, \neg l, \neg r\}$ i.e. les agents savent que la porte est poncée, qu'elle n'est ni laquée ni recouverte de papier. Supposons que $\neg p$ soit incontrôlable (i.e. les agents n'ont pas les moyens de faire en sorte que la porte ne soit plus poncée), que $C_{a_1} = \{l\}$ et que $C_{a_2} = \{r, \neg r\}$ (i.e. l'agent a_1 peut laquer la porte, l'agent a_2 peut la recouvrir de papier ou enlever le papier si elle est recouverte).

[2] Car si $l_1 \wedge \ldots \wedge l_m$ est un CK-but de $\mathcal{A}$, $\forall i \in \{1, \ldots, l\}$ l_i est un CK-but de $\mathcal{A}$.

Dans ce cas, $NC(KB) = \{p\}$, puisque KB est un contexte pour l et pour r. $l \wedge \neg r$ est un CK-but du groupe[3]. Si les agents ne s'engagent à rien, $D(KB) = \{p\}$, on peut dériver $EGoal_{a_1}(l)$ et $EGoal_2(\neg r)$. L'agent a_1 a donc pour ensemble de but effectifs atomiques $\{l\}$ (i.e. il a pour but de laquer la porte) et l'agent 2 a pour ensemble de buts effectifs atomiques $\{\neg r\}$ (i.e. il a pour but de ne pas la recouvrir). C'est intuitivement correct.

2. Supposons que $KB = \{\neg p, \neg l, \neg r\}$, que $C_{a_1} = \{l, \neg l\}$ et que $C_{a_2} = \{p, r, \neg r\}$. Dans ce cas, $NC(KB) = \phi$ et $(l \wedge \neg r) \vee (\neg l \wedge r)$ est un CK-but du groupe. Si $D(KB) = \phi$ (i.e. les agents ne s'engagent à rien), on ne peut pas dériver de buts effectifs pour les agents, car a_2 contrôle p et pourrait donc rendre p vrai. On a donc $Nonsat((l \wedge \neg r) \vee (\neg l \wedge r))$.

   Par contre, si a_2 s'engage à ne pas réaliser p (i.e. à ne pas poncer la porte), on a alors $Com_-(\{a_1, a_2\} = \{\neg p\}$ et on peut alors dériver $EGoal_{a_2}(r)$ et $EGoal_{a_1}(\neg l)$: l'agent a_1 a pour but effectif de recouvrir la porte de papier et l'agent a_1 a pour but effectif de conserver la porte non laquée.

3. Supposons que $KB = \{\neg p, \neg l, \neg r\}$, que $C_{a_1} = \{l\}$ et que $C_{a_2} = \{p, r\}$. Dans ce cas, $NC(KB) = \phi$ et $l \vee r$ est un CK-but du groupe. Supposons de plus que a_2 s'engage à réaliser p, soit $Com_+(a_2) = \{p\}$. Dans ce cas, $D(KB) = \{p\}$, et on dérive $EGoal_{a_1}(l)$, $EGoal_{a_2}(p)$: l'agent a_1 doit laquer la porte, l'agent a_2 doit la poncer. L'agent a_2 n'a pas à ne pas recouvrir la porte de papier car il ne contrôle pas $\neg r$.

4. Supposons que $KB = \{\neg p, \neg l, \neg r\}$, que $C_{a_1} = \{l\}$ et que $C_{a_2} = \{p, r\}$. a_2 s'engage à réaliser p, donc $Com_+(a_2) = \{p\}$ et a_1 s'engage à ne pas réaliser l, soit $Com_-(a_1) = \{l\}$. Dans ce cas, $NC(KB) = \phi$, $Com_+(\{a_1, a_2\}) = \{p\}$ et $Com_-(\{a_1, a_2\}) = \{\neg l\}$. $Com_+(\{a_1, a_2\}) \cup Com_-(\{a_1, a_2\}) \cup \{p \rightarrow l\}$ est incohérent. Or $p \rightarrow l$ est un CK-but de $\{a_1, a_2\}$, donc l'hypothèse 12.1.4 n'est pas vérifiée. a_1 et a_2 doivent revoir leurs engagements.

On remarquera que nous n'utilisons pas ici les ensembles de buts atomiques des agents, car les buts effectifs des agents sont assez simples.

12.4 Stratégies de distribution

Il se peut qu'en utilisant les définitions précédentes, plusieurs agents distincts $a_i, \ldots, a_j$ appartenant à $\mathcal{A}$ aient tous pour but effectif de réaliser φ. Pour éviter cela, deux solutions se proposent à nous :

- soit on peut considérer que les agents $a_i, \ldots, a_j$ n'ont pas à réaliser φ mais que c'est le groupe $\{a_i, \ldots, a_j\}$ (ou un de ses sous-groupes) qui réalisera φ (stratégie coopérative) ;
- soit on peut éliminer tous les agents sauf un grâce à une stratégie particulière.

Dans toute la suite, on appelera $Ag(\varphi)$ l'ensemble des agents qui ont pour but effectif de réaliser φ avant utilisation d'une stratégie.

Définition 12.4.1. *Soit φ une formule propositionnelle.*
$Ag(\varphi) = \{a_i \in \mathcal{A} : \Sigma \models I(\varphi | D(KB))$ *et KB est un contexte pour φ par rapport à $a_i\}$.*

[3] C'est en fait le seul qui nous interesse. On peut bien sûr écrire que $(l \wedge \neg r) \vee r$ est un CK-but du groupe.

12.4.1 Définition d'une stratégie

Pour pouvoir sélectionner un ou plusieurs agents parmi $Ag(\varphi)$, nous allons utiliser un préordre sur $Ag(\varphi)$. Ce préordre est un préordre de préférence sur les agents de $Ag(\varphi)$ et peut refléter de nombreuses notions : compétence relative des agents, « coût » d'un agent etc.

Définition 12.4.2. *Une stratégie est une fonction* $\mathcal{S} : PROP \rightarrow \mathcal{A} \times \mathcal{A}$ *telle que pour toute proposition* φ, $\mathcal{S}(\varphi)$ *est un préordre* $\leq_{\mathcal{S}(\varphi)}$ *sur* $Ag(\varphi)$.

Plusieurs remarques sont à faire :
- le fait d'avoir une fonction nous permet d'assurer qu'à une proposition on associe au plus un préordre ;
- le préordre associé à chaque proposition nous permettra de déterminer quel est l'agent ou le groupe d'agent à choisir ;
- il se peut qu'on ne puisse pas choisir un agent unique (cas de la « coopération » par exemple).

La définition du minimum d'un ensemble ordonné est la suivante :

Définition 12.4.3. *Soit* E *un ensemble ordonné par* $\leq_E$.
$\min_{\leq_E}(E) = \{e_i \in E \ : \ \forall e_j \in E \ e_j \leq_E e_i \Rightarrow e_j = e_i\}$

Nous pouvons donc maintenant introduire une nouvelle définition des buts effectifs. La nouvelle définition d'un but effectif concerne un sous groupe d'agents, ce qui permet de traiter à la fois le cas d'un seul agent et le cas de la coopération.

12.4.2 Buts effectifs d'un sous groupe d'agents

Les buts effectifs d'un sous groupe d'agents sont maintenant définis de la façon suivante :

Définition 12.4.4. φ *est un but effectif pour* $\mathcal{A}' \subseteq \mathcal{A}$ *suivant la stratégie* $\mathcal{S}$, *noté* $EGoal^{\mathcal{S}}_{\mathcal{A}'}(\varphi)$ *ssi :*

$$\mathcal{A}' = \min_{\leq_{\mathcal{S}(\varphi)}} Ag(\varphi)$$

Soient $\mathcal{S}$ une stratégie, φ une proposition et $\mathcal{A}' \subseteq \mathcal{A}$ un sous-groupe d'agents de $\mathcal{A}$ tels que l'on ait $EGoal^{\mathcal{S}}_{\mathcal{A}'}(\varphi)$. Tout d'abord, on peut remarquer que $\forall a_i \in \mathcal{A}' \ a_i \in Ag(\varphi)$. Donc $\forall a_i \in \mathcal{A}' \ \Sigma \models I(\varphi | D(KB))$ et KB est un contexte pour φ par rapport à a_i. Tous les agents de $\mathcal{A}'$ étaient des candidats potentiels pour réaliser φ.

On peut vérifier que si une proposition φ est un but effectif pour un seul agent avec les définitions précédentes, elle restera un but effectif pour ce même agent avec cette nouvelle définition (en assimilant l'agent avec le sous-groupe réduit au singleton de cet agent).

En effet, si φ est un but effectif pour l'agent a_i et pour l'agent a_i seulement, alors on a d'une part $\Sigma \models I(\varphi | D(KB))$ et KB contexte pour φ par rapport à a_i. D'autre part, $Ag(\varphi) = \{a_i\}$, donc pour toute stratégie $\mathcal{S}$, $S(\varphi)$ est un ordre total sur $\{a_i\}$. Donc $\{a_i\} = \min_{\leq_{\mathcal{S}(\varphi)}} Ag(\varphi)$.

12.4.3 Cas particulier d'une proposition « non but »

Dans cette section, nous nous intéressons au cas particulier d'une proposition φ qui n'était pas un but effectif potentiel pour un agent. Dans ce cas, on a $Ag(\varphi) = \phi$, donc quelle que soit la stratégie $\mathcal{S}$ que l'on utilise, $\min_{\leq_{\mathcal{S}(\varphi)}} Ag(\varphi) = \phi$. On pourra donc toujours écrire que l'on a $EGoal^{\mathcal{S}}_{\phi}(\varphi)$, i.e. que φ n'est un but effectif pour personne.

12.4.4 Cas des stratégies non-sélectives

Le cas des stratégies non-sélectives est un cas très particulier. En effet, lorsque plusieurs agents $a_i, \ldots, a_j$ peuvent avoir pour but effectif de réaliser φ, une stratégie non-sélective consiste à attribuer la réalisation de φ non pas à un des deux agents, mais à un sous-groupe de $\{a_i, \ldots, a_j\}$ non réduit à un seul élément.

Définition 12.4.5. *Une stratégie $\mathcal{S}$ est une stratégie de distribution non-sélective pour φ ssi $card(\min_{\leq_{\mathcal{S}(\varphi)}} Ag(\varphi)) > 1$.*

La définition donnée dans la section 12.2 pour la satisfaction d'un CK-but du groupe est étendue comme il se doit avec les sous groupes d'agents.

Exemple. *Reprenons l'exemple de la section 12.3. On dispose d'un groupe de deux agents $\{a_1, a_2\}$. Les préférences du groupe sont modélisées par $\Sigma = \{I(l \wedge \neg r|p), I(r \wedge \neg l|\neg p)\}$.*

Supposons que l'on ait $KB = \{\neg p, \neg l, \neg r\}$, que $C_{a_1} = \{p, l\}$ et que $C_{a_2} = \{l, r, \neg r\}$. Si a_1 s'engage à réaliser p et que a_2 s'engage à réaliser l, alors on a $D(KB) = \{p, l\}$.

On a donc $EGoal_{a_1}(p \wedge l)$ et $EGoal_{a_2}(l \wedge \neg r)$. Ceci pose problème puisque à la fois a_1 et a_2 ont l pour but effectif.

Remarquons tout de suite que comme $Ag(p) = \{a_1\}$ et $Ag(\neg r) = \{a_2\}$, pour toute stratégie (S) on a encore $EGoal^{\mathcal{S}}_{\{a_1\}}(p)$ et $EGoal^{\mathcal{S}}_{\{a_2\}}(\neg r)$.

Supposons que $\mathcal{S}$ soit une stratégie non-sélective pour l, alors $card(\min_{\leq_{\mathcal{S}(\varphi)}} Ag(\varphi)) > 1$. Or $Ag(\varphi) = \{a_1, a_2\}$, donc $\min_{\leq_{\mathcal{S}(\varphi)}} Ag(\varphi) = \{a_1, a_2\}$. Dans ce cas, on ne pourra plus dériver $EGoal^{\mathcal{S}}_{\{a_1\}}(l)$ et $EGoal^{\mathcal{S}}_{\{a_2\}}(l)$. Par contre, on aura $EGoal^{\mathcal{S}}_{\{a_1,a_2\}}(l)$.

12.4.5 Cas des stratégies sélectives

Dans le cas des stratégies sélectives, il faut pouvoir sélectionner un seul agent parmi tous les agents « pressentis » pour réaliser φ. La définition d'une stratégie sélective est la suivante :

Définition 12.4.6. *Une stratégie $\mathcal{S}$ est une stratégie de distribution sélective pour φ ssi $card(\min_{\leq_{\mathcal{S}(\varphi)}} Ag(\varphi)) = 1$.*

Exemple. *Reprenons l'exemple précédent. On a alors $KB = \{\neg p, \neg l, \neg r\}$, $C_{a_1} = \{p, l\}$ et $C_{a_2} = \{l, r\}$. Si a_1 s'engage à réaliser p et que a_2 s'engage à réaliser l, alors on a $D(KB) = \{p, l\}$.*

Supposons que l'on ait une stratégie $\mathcal{S}$ sélective. Supposons de plus que $\min_{\leq_{\mathcal{S}(l)}} Ag(l) = \{a_1\}$, alors on a $EGoal^{\mathcal{S}}_{\{a_1\}}(p)$, $EGoal^{\mathcal{S}}_{\{a_2\}}(\neg r)$ et $EGoal^{\mathcal{S}}_{\{a_1\}}(l)$.

12.4.6 Fusion de deux ordres avec attitude prioritaire

Dans cette section, nous allons développer une méthode pour pouvoir fusionner deux ordres avec une attitude prioritaire. Cette méthode nous permettra, dans la section suivante, de combiner des stratégies entre elles.

L'objectif est le suivant : on dispose de deux ordres $\leq_1$ et $\leq_2$ sur un même ensemble E et on veut obtenir un ou plusieurs ordres $\leq_{1\circ 2}$, dits ordres fusionnés de $\leq_1$ prioritairement à $\leq_2$, qui vérifient en priorité l'ordre $\leq_1$ et qui sont complétés par une partie de l'ordre $\leq_2$.

Pour nous rapprocher des méthodes de fusion de bases de croyances prioritaires [31], nous allons dans représenter un ordre par un prédicat muni d'un ensemble de règles adéquates. En effet, on peut établir un parallèle entre un ensemble et les ordres possibles sur cet ensemble (qui sont des relations binaires particulières) et un langage du premier ordre muni de prédicats binaires et d'une theorie particulière.

Définition 12.4.7. *Soit $E = \{e_1, \ldots, e_n\}$ un ensemble d'éléments finis. Soit $\leq_E = \{\leq_i : i \in \mathbb{N}\}$ l'ensemble des ordres possibles sur E. On peut représenter E et $\leq_E$ par le langage du premier ordre $\mathcal{L}_E$ et la théorie $\mathcal{T}_E$ définis comme suit :*

1. *le langage $\mathcal{L}_E$ est constitué*
   - *des symboles logiques classiques (ensemble de variables dénombrable, connecteurs, quantificateurs, signes de ponctuation) ;*
   - *$\{e_1, \ldots, e_n\}$ comme ensemble de symboles de constantes ;*
   - *$\{\preceq_i : \leq_i \in \leq_E\} \cup \{=\}$ où tous les $\preceq_i$ et $=$ sont des symboles de prédicats binaires comme ensemble de symboles de prédicats ;*
   - *d'aucun symbole de fonction.*
2. $\mathcal{T}_E = \{\neg(e_i = e_j) \ : \ (i,j) \in \{1 \ldots n\}^2 \ i \neq j\} \cup \bigcup_{\leq_i \in \leq_E} RAT(\preceq_i)$

   où

$$\begin{aligned} RAT(\preceq_i) \ = \ & \{\forall x \ \preceq_i (x,x), \\ & \forall x \forall y \ \preceq_i (x,y) \wedge \preceq_i (y,x) \rightarrow x = y \\ & \forall x \forall y \forall z \ \preceq_i (x,y) \wedge \preceq_i (y,z) \rightarrow \preceq_i (x,z)\} \end{aligned}$$

La théorie $\mathcal{T}_E$ permet de représenter l'unicité des symboles de constantes et les propriétés mathématiques des ordres sur E en les traduisant en axiomes propres pour chacun des prédicats de $\mathcal{L}_E$.

Pour des facilités de notation, dans toute la suite, $x \preceq_i y$ sera équivalent à $\preceq_i (x,y)$.

Pour construire un ordre sur un ensemble, l'utilisateur donne un ensemble de relations dites *explicites* qui va permettre de générer l'ordre entier (en utilisant la transitivité, l'antisymétrie et la reflexivité).

Exemple. *Si on considère un ensemble $E_1 = \{a_1, a_2, a_3\}$, l'ensemble explicite $\{a_1 \leq_{E_1} a_2, a_2 \leq_{E_1} a_3\}$ nous permet de construire l'ordre $\leq_{E_1}$ sur $\{a_1, a_2, a_3\}$ tel que $a_1 \leq_{E_1} a_2$, $a_1 \leq_{E_1} a_3$, $a_2 \leq_{E_1} a_3$, $a_1 \leq_{E_1} a_1$, $a_2 \leq_{E_1} a_2$, $a_3 \leq_{E_1} a_3$, $a_2 \not\leq_{E_1} a_1$, $a_3 \not\leq_{E_1} a_2$ et $a_3 \not\leq_{E_1} a_1$.*

Un ensemble explicite associé à un ordre peut donc être facilement représenté par une théorie.

Définition 12.4.8. *Soient E un ensemble, $\mathcal{L}_E$ le langage associé à E et $\mathcal{T}_E$ la théorie associée à E. $\mathcal{E}_i$, ensemble de formules du type $e_i \preceq_i e_j$ avec $e_i \in E$ et $e_j \in E$ est un ensemble explicite ssi* [4] *$Cl(\mathcal{T}_E \cup \mathcal{E}_i)$ est cohérent.*

L'ordre $\leq_i$ sur E appelé ordre généré *par $\mathcal{E}_i$ est défini par : $\forall e_i \in E\ \forall e_j \in e\ \ e_i \leq_E e_j$ ssi $Cl(\mathcal{T}_E \cup \mathcal{E}_i) \vdash e_i \preceq_i e_j$*

On démontre facilement que l'on obtient bien un « vrai » ordre $\leq_i$ en utilisant les axiomes propres contenus dans la théorie $\mathcal{T}_E$.

Exemple. *Reprenons l'exemple précédent. On peut écrire que $\{a_1 \preceq_{E_1} a_2, a_2 \preceq_{E_1} a_3\}$ est un ensemble explicite générant $\leq_{E_1}$.*

Munis de ces définitions, nous pouvons définir comment combiner deux ordres, l'un étant prioritaire par rapport à l'autre. Pour cela, nous allons utiliser les ensembles explicites définissant chaque ordre en recherchant des ensembles maximaux cohérents de formules du premier ordre. Comme il peut en avoir plusieurs, on peut obtenir plusieurs ensembles explicites pour l'ordre construit.

Définition 12.4.9. *Soient E un ensemble et $\mathcal{L}_E$ le langage du premier ordre associé à E. Soient $\mathcal{E}_1$ et $\mathcal{E}_2$ deux ensembles explicites générant respectivement les ordres $\leq_1$ et $\leq_2$ sur E. On note :*

- *$\mathcal{E}_{1\to 1\circ 2} = \{e_i \preceq_{1\circ 2} e_j \ :\ e_i \preceq_1 e_j \in \mathcal{E}_1\}$;*
- *$\mathcal{E}_{2\to 1\circ 2} = \{e_i \preceq_{1\circ 2} e_j \ :\ e_i \preceq_2 e_j \in \mathcal{E}_2\}$.*

L'ensemble explicite $\mathcal{E}^i_{1\circ 2}$ est défini par :

$$\mathcal{E}^i_{1\circ 2} = \{e_j \preceq^i_{1\circ 2} e_k \ :\ (e_j \preceq_{1\circ 2} e_k) \in (\mathcal{E}_{1\to 1\circ 2} \cup \mathcal{E}^i_{2\to 1\circ 2})\}$$

où $\mathcal{E}^i_{2\to 1\circ 2}$ est un sous ensemble maximal (pour l'inclusion) de $\mathcal{E}_{2\to 1\circ 2}$ tel que $\mathcal{E}_{1\to 1\circ 2} \cup \mathcal{E}^i_{2\to 1\circ 2} \cup \mathcal{T}_E$ soit cohérent.

On appelle $\leq^i_{1\circ 2}$ l'ordre sur E généré par l'ensemble explicite $\mathcal{E}^i_{1\circ 2}$.

On appelle $n_{1\circ 2}$ le nombre d'ordres différents que l'on peut construire à partir de $\leq_1$ et $\leq_2$ en donnant priorité à $\leq_1$.

On peut remarquer qu'il se peut que l'on puisse réduire les différents ordres à un seul ordre. Dans ce cas, on peut parler de l'ordre $\leq_{1\circ 2}$.

Définition 12.4.10. *Si $\exists i \in \{1 \dots n_{1\circ 2}\}$ tel que $\forall j \in \{1 \dots n_{1\circ 2}\}$ $\mathcal{E}l^j_{1\circ 2} \subseteq \mathcal{E}^i_{1\circ 2}$, alors on note $\mathcal{E}_{1\circ 2} = \mathcal{E}^i_{1\circ 2}$.*

On peut maintenant définir un minimum pour un ordre composé.

Définition 12.4.11. *Soit E un ensemble et $\leq_1$ et $\leq_2$ deux ordres sur E. Deux cas se présentent :*

- *soit $\leq_{1\circ 2}$ existe et alors $\min_{\leq_{1\circ 2}} E = \{e_i \in E \ :\ \forall e_j \in E\ e_j \leq_{1\circ 2} e_i \Rightarrow e_j = e_i\}$;*
- *soit $\leq_{1\circ 2}$ n'existe pas et alors $\min_{\leq_{1\circ 2}} E = \bigcap_{i\in\{1\dots n_{1\circ 2}\}} \min_{\leq^i_{1\circ 2}} E$.*

Exemple. *Supposons que l'on ait un ensemble $E = \{e_1, e_2, e_3\}$. Examinons quelques exemples :*

[4] $Cl()$ désigne la fermeture d'un ensemble de formules.

1. *Supposons que l'ordre* $\leq_1$ *soit généré par* $\{e_3 \preceq_1 e_2\}$ *et que l'ordre* $\leq_2$ *soit généré par* $\{e_2 \preceq_2 e_3, e_1 \preceq_2 e_2\}$.
   On a $\mathcal{E}_{1\to 1\circ 2} = \{e_3 \preceq_{1\circ 2} e_2\}$ *et* $\mathcal{E}_{2\to 1\circ 2} = \{e_2 \preceq_{1\circ 2} e_3, e_1 \preceq_{1\circ 2} e_2\}$. *Le seul sous-ensemble de* $\mathcal{E}_{2\to 1\circ 2}$ *cohérent avec* $\mathcal{E}_{1\to 1\circ 2} \cup \mathcal{T}_E$ *est* $\{e_1 \preceq_{1\circ 2} e_2\}$ *(car* $\mathcal{E}_{1\to 1\circ 2} \cup \mathcal{T}_E \vdash \neg e_2 \preceq_{1\circ 2} e_3$*), donc on obtient bien un ordre* $\leq_{1\circ 2}$ *généré par* $\{e_3 \preceq_{1\circ 2} e_2, e_1 \preceq_{1\circ 2} e_2\}$.
   Dans ce cas, $\min_{\leq_{1\circ 2}} E = \{e_1, e_3\}$.
2. *Supposons que l'ordre* $\leq_1$ *soit généré par* $\{e_3 \preceq_1 e_2\}$ *et que l'ordre* $\leq_2$ *soit généré par* $\{e_3 \preceq_2 e_1, e_1 \preceq_2 e_2\}$.
   On a $\mathcal{E}_{1\to 1\circ 2} = \{e_3 \preceq_{1\circ 2} e_2\}$ *et* $\mathcal{E}_{2\to 1\circ 2} = \{e_3 \preceq_{1\circ 2} e_1, e_1 \preceq_{1\circ 2} e_2\}$. *Cette fois ci, on obtient deux sous-ensembles de* $\mathcal{E}_{2\to 1\circ 2}$ *maximaux cohérents avec* $\mathcal{E}_{1\to 1\circ 2} \cup \mathcal{T}_E$*, donc on obtient deux ordres :* $\leq^1_{1\circ 2}$*, généré par* $\{e_3 \leq^1_{1\circ 2} e_2, e_3 \leq^1_{1\circ 2} e_1\}$ *et* $\leq^2_{1\circ 2}$*, généré par* $\{e_3 \leq^2_{1\circ 2} e_2, e_1 \leq^2_{1\circ 2} e_2\}$.
   Dans ce cas, $\min_{\leq_{1\circ 2}} E = \{e_3\}$.

12.4.7 Familles de stratégies

Il s'agit maintenant de pouvoir définir pour chaque proposition φ $\mathcal{S}(\varphi)$, i.e. un ordre sur $Ag(\varphi)$. C'est ce qui va caractériser la stratégie.

Stratégies volontaires

L'objectif de cette stratégie est d'assigner la tâche à réaliser au sous-groupe d'agents qui s'est engagé a réaliser la tâche. L'ordre vérifie alors la propriété suivante :

Définition 12.4.12. *Soit* φ *une proposition et* $\mathcal{S}$ *une stratégie.* $\mathcal{S}$ *est une stratégie volontaire pour* φ *ssi* $\forall a_i \in Ag(\varphi)\ \forall a_j \in Ag(\varphi)\ a_i \leq_{\mathcal{S}(\varphi)} a_j$ *ssi* $Com_+(a_i) \models \varphi$ *et* $Com_+(a_j) \not\models \varphi$.

En utilisant un tel ordre, on est sûr que les agents appartenant à $\min_{\leq_{\mathcal{S}(\varphi)}} Ag(\varphi)$ se sont engagés à réaliser φ.

Stratégies non-volontaires

On peut également définir une stratégie qui empêche les agents qui se sont engagés à ne pas réaliser φ à être parmis les agents minimaux pour l'ordre $\leq_{\mathcal{S}(\varphi)}$. Cette stratégie est moins restrictive que la précédente : un agent qui ne s'est pas engagé à réaliser φ et à ne pas réaliser φ peut faire partie de ces agents minimaux.

Définition 12.4.13. *Soit* φ *une proposition et* $\mathcal{S}$ *une stratégie.* $\mathcal{S}$ *est une stratégie non-volontaire pour* φ *ssi* $\forall a_i \in Ag(\varphi)\ \forall a_j \in Ag(\varphi)\ a_i \leq_{\mathcal{S}(\varphi)} a_j$ *ssi* $Com_-(a_i) \not\models \varphi$ *et* $Com_-(a_j) \models \varphi$.

Stratégies combinant plusieurs ordres

Il se peut que l'on veuille utiliser plusieurs ordres pour pouvoir déterminer le sous-groupe d'agents. Par exemple, si on dispose d'un premier ordre $\leq^C_{\mathcal{S}(\varphi)}$ sur $Ag(\varphi)$ reflétant la compétence pour un agent à réaliser φ, et d'un deuxième ordre type volontaire $\leq^V_{\mathcal{S}(\varphi)}$, on peut les combiner de deux façons :

– soit on sélectionne les agents les plus compétents, puis on sélectionne parmi ceux-ci ceux qui se sont engagés ;
– soit on sélectionne les agents qui se sont engagés, puis on sélectionne les agents les plus compétents parmi ceux-ci.

Il faut donc combiner les deux ordres, l'un étant prioritaire par rapport à l'autre. Dans le premier cas, on va utiliser l'ordre $\leq_{C \circ V}$. Dans le deuxième cas, on va utiliser l'ordre $\leq_{V \circ C}$.

La plupart du temps, on va utiliser plusieurs ordres reflétant la compétence relative des agents, leurs besoins en ressources etc. que l'on fusionne ensuite pour obtenir un ordre correspondant à la stratégie voulue.

12.4.8 Exemple

Reprenons l'exemple de la section 12.3. On dispose d'un groupe de trois agents $\{a_1, a_2, a_3\}$. Les préférences du groupe sont modélisées par $\Sigma = \{I(l \wedge \neg r|p), I(r \wedge \neg l|\neg p)\}$.

Supposons que l'on ait $KB = \{\neg p, \neg l, \neg r\}$, que $C_{a_1} = \{p, l\}$, que $C_{a_2} = \{l, r, \neg r\}$ et que $C_{a_3} = \{l\}$. Si a_1 s'engage à réaliser p, que a_2 et a_3 s'engagent à réaliser l, alors on a $D(KB) = \{p, l\}$.

On a donc $EGoal_{a_1}(p \wedge l)$, $EGoal_{a_2}(l \wedge \neg r)$ et $EGoal_{a_3}(l)$. Les trois agents ont donc pour but de laquer la porte.

Supposons que l'on ne veuille pas que les agents coopérent, car on pense que la coopération ferait perdre du temps à l'équipe. Il va donc falloir trouver une stratégie sélective pour « départager » les trois agents.

Une stratégie volontaire $\mathcal{S}_V$ pour l nous donne le préordre suivant :

$$a_2 \leq_{\mathcal{S}_V(l)} a_1$$
$$a_3 \leq_{\mathcal{S}_V(l)} a_1$$

L'utilisation d'une telle stratégie ne suffit pas, car elle sert à éliminer a_1, mais ne permet pas de « départager » a_2 et a_3.

Supposons que l'on dispose également d'une stratégie $\mathcal{S}_E$ pour l qui reflète l'efficacité relative des agents pour réaliser l :

$$a_1 \leq_{\mathcal{S}_E(l)} a_2$$
$$a_2 \leq_{\mathcal{S}_E(l)} a_3$$

Dans ce cas, deux choix s'offrent à nous :
– soit on utilise $\leq_{\mathcal{S}_E(l) \circ \mathcal{S}_V(l)}$ (on privilégie l'efficacité des agents) et dans ce cas, l ne va être un but effectif que pour a_1 ;
– soit on utilise $\leq_{\mathcal{S}_V(l) \circ \mathcal{S}_C(l)}$ (on privilégie les agents qui sont volontaires) et dans ce cas, l'utilisation de la stratégie d'efficacité permet d'éliminer a_3 : l n'est un but effectif que pour a_2.

12.5 Conclusion

Nous avons présenté dans ce chapitre une extension du modèle d'agent exécutant à un groupe d'agents exécutants. Cette extension nous permet pour un groupe d'agents

exécutants de déterminer les buts du groupe d'agents et les buts effectifs de chacun des agents du groupe en fonction d'un ensemble de préférences. Dans le cadre de l'ingénierie des exigences, cet ensemble d'exigences représente la traduction de l'ensemble d'exigences cohérent obtenu après la phase de fusion des exigences.

On peut donc vérifier grâce à ce processus que les exigences vont bien être respectées, dans le sens où pour chaque exigence, il y aura au moins un agent exécutant qui va prendre en charge cette exigence. Dans le cas contraire, on peut alors demander aux agents exécutants de modifier leurs engagements par exemple pour que toutes les exigences soient satisfaites.

Nous avons également introduit la notion de stratégie, qui permet de raffiner le processus de distribution en sélectionnant, parmi tous les agents susceptibles d'effectuer une action, ceux qui correspondent le mieux à un critère particulier. Le fait d'utiliser des ordres nous permet d'avoir un formalisme assez souple : on peut par exemple fusionner deux ordres pour obtenir une stratégie hybride.

Chapitre 13

Conclusion et perspectives

Dans ce chapitre, nous concluons notre travail. Dans un premier temps, nous résumons les points principaux de notre contribution. Dans un second temps, nous présentons les points méritant un développement futur.

Contributions

Nous avons identifié trois phases distinctes dans le processus d'ingénierie des exigences. Pour ces trois phases, nous nous sommes efforcés de développer un cadre formel et logique permettant de faciliter l'expression, la gestion et la distribution d'exigences dans un projet industriel quelconque.

Dans la phase de modélisation des exigences, nous avons tout d'abord constaté que peu de formalismes proposaient d'intégrer la notion de réglementation et de contrainte du domaine dans le processus d'ingénierie des exigences. Or, ces deux notions nous paraissent primordiales, surtout dans le cadre de projets complexes nécessitant énormément de compétences distinctes. Dans un deuxième temps, nous avons choisi de représenter les exigences sous forme de position (notion introduite par Cholvy et Hunter), qui permet à l'agent émetteur d'exigences de classer celles-ci suivant un ordre de préférence. Nous avons donc choisi d'utiliser la logique de préférence CO pour représenter la position d'un agent. L'utilisation de CO nous permet de représenter les positions sous forme d'un ensemble de formules compact, tout en garantissant que la traduction sémantique de cet ensemble de formules est intuitivement correct. Nous avons également montré que CO pouvait être utilisée pour représenter des contraintes du domaine, mais également des phrases normatives complexes comme des normes avec exceptions et des Contrary-to-Duties. Ceci nous a amené à définir la notion de cohérence entre une position, un ensemble de normes et un ensemble de contraintes du domaine. Grâce à la notion de position, nous pouvons déterminer le meilleur ensemble d'exigences pour l'agent étant données une réglementation et des contraintes du domaine.

En ce qui concerne la résolution de conflits possibles entre exigences de différents agents, nous avons vu que la plupart des formalismes déjà proposés permettent de détecter les inconsistances, mais ne proposent pas un ensemble d'exigences « consensuel ». Seul le formalisme de Cholvy et Hunter utilise une méthode de fusion par priorité entre les émetteurs d'exigences et permet de déterminer automatiquement un seul ensemble d'exigences cohérent. En constatant qu'il n'existe pas forcément un ordre total sur les différents agents émetteurs d'exigences, nous avons proposé une méthode de fusion de type majoritaire. L'utilisation d'une méthode de fusion majoritaire nous permet de considérer les

agents comme étant au même niveau hiérarchique. Pour cela, nous avons développé MF, une logique modale permettant de raisonner sur le contenu de bases de croyances obtenues par fusion majoritaire de plusieurs bases de croyances primitives. Nous avons montré que la sémantique de MF respecte les postulats établis par Konieczny et Pino-Pérez pour les opérateurs de fusion majoritaire et nous avons développé un démonstrateur automatique pour MF implanté en PROLOG. Dans un deuxième temps, nous avons montré que notre approche pouvait être étendue pour des bases de croyances du premier ordre particulières et que cette extension nous permet de raisonner sur des ensembles d'exigences. Nous avons considéré les ensembles d'exigences comme étant soit non ordonnés (mais disposant de contraintes du domaine individuelles), soit comme étant représentés sous forme de positions. La méthode proposée nous permet d'obtenir un seul ensemble d'exigences cohérent à partir de plusieurs ensembles pouvant être inconsistants entre eux.

Enfin, nous avons proposé d'inclure dans le processus d'ingénierie des exigences une phase de distribution d'exigences sur des agents exécutants. Cette phase n'apparaît pas dans les autres travaux traitant d'ingénierie des exigences. Elle nous paraît pourtant intéressante à considérer, car elle permet de vérifier si étant donné un ensemble d'agents exécutants mis à notre disposition pour construire l'objet, on est capable de respecter les exigences. Nous avons proposé un modèle d'agent exécutant simple, qui est une extension du modèle proposé par Boutilier en théorie de la décision qualitative. Ce modèle nous permet de représenter les capacités de agents, leurs connaissances du monde et leurs préférences. On peut alors dériver quels sont leurs buts, mais également vérifier qu'un agent viole une réglementation ou dériver ses obligations idéales et effectives. Munis de ce modèle d'agent, nous nous sommes intéressés à la distribution d'exigences. Nous avons étendu le modèle d'agent à un groupe d'agents, puis nous avons défini un modèle de distribution basé sur une entité centrale connaissant les engagements des différents agents. À partir de ces engagements, nous sommes capables de déterminer quels vont être les buts effectifs de chacun des agents exécutants. Nous avons également étudié la notion de stratégie, qui nous permet de choisir entre plusieurs agents exécutants auxquels on a attribué le même but effectif. Grâce à une méthode de fusion, on peut combiner différentes stratégies pour obtenir des stratégies hybrides.

Perspectives

Ce travail peut être poursuivi dans plusieurs directions :

- la travail de modélisation pourraît être étendu en utilisant non pas une logique de base propositionnelle, mais une logique du premier ordre (ou tout du moins à un fragment décidable du premier ordre). Il faudrait alors redéfinir une logique modale du premier ordre permettant de raisonner avec de telles préférences ;
- la modélisation de réglementations peut être affinée. Nous avons vu que la logique CO nous permet de modéliser un grand nombre de phrases normatives, mais elle souffre de la plupart des défauts de SDL en ce qui concerne la dérivation de paradoxes. L'utilisation d'une relation de conséquence « restreinte » nous permettrait d'éviter de déduire un certain nombre de paradoxes à partir d'un ensemble de normes donné. Il faudrait également vérifier la validité des postulats de Carmo et Jones. Nous utilisons ces postulats pour construire un formalisme permettant de raisonner avec des CTDs, mais nous n'avons pas vérifié que ces postulats étaient complets par exemple ;
- il serait également intéressant de pouvoir doter CO d'un démonstrateur automa-

tique. L'utilisation d'un tel outil nous permettrait de vérifier plus facilement qu'une position est cohérente avec une réglementation et des contraintes du domaine par exemple. Il pourrait également servir à la dérivation d'obligations idéales et effectives grâce au modèle que nous proposons ;

- l'extension de la logique MF aux bases de croyances contenant des disjonctions est un travail important. Dans le cadre de l'ingénierie des exigences, la contrainte imposant aux bases de ne pas contenir de disjonctions n'est pas si restrictive qu'il n'y paraît. On peut en effet considérer que les agents émetteurs d'exigences n'exigent que des propriétés non disjonctives sur l'objet à concevoir. Par contre, dans le cadre de bases de données par exemple, il nous paraît intéressant de pouvoir exprimer de vraies informations disjonctives à l'intérieur de ces bases ;
- le processus de fusion d'exigences que nous proposons nous permet d'obtenir une représentation « sémantique » du résultat de la fusion. Il serait beaucoup plus intéressant de pouvoir obtenir une position, i.e. de pouvoir caractériser une position à partir d'un préordre sur les mondes possibles ;
- enfin, le processus de distribution peut être affiné sur plusieurs points. Tout d'abord, on peut donner un modèle plus fin des agents exécutants, en considérant par exemple qu'ils ont une connaissance incomplète du monde ou/et qu'ils n'ont pas la même représentation du monde. On peut également s'affranchir de l'entité centrale qui distribue le travail et considérer que les agents communiquent entre eux et fournissent partiellement leurs engagements. Il serait alors intéressant de pouvoir déterminer les propriétés du formalisme. On peut également étendre la notion de stratégie et définir de nouvelles familles. Enfin, on pourrait montrer sur un cas réel complexe que l'utilisation de ce processus de distribution permet de détecter des problèmes possibles et ainsi de vérifier que les exigences intiales vont être garanties. On peut également remarquer que nous avons commencé un travail préliminaire sur les responsabilités de groupe dans [41] qui permet de raisonner non plus simplement avec un agent, mais avec un groupe d'agents.

Annexe A

Preuves

A.1 Preuves de la partie I

Propriété 5.1.1. *$<_{\Gamma_a^{CO}}$ est une relation d'ordre sur les formules de PROP.*

Preuve. *La démonstration se fait en trois points :*

- *$<_{\Gamma_a^{CO}}$ est une relation irréflexive*
  La démonstration est aisée : supposons que $<_{\Gamma_a^{CO}}$ ne soit pas une relation irréflexive, alors pour il existe deux propositions α et β telles que $\alpha <_{\Gamma_a^{CO}} \beta$ et $\alpha \equiv \beta$ (au sens logique). On a donc $\Gamma_a^{CO} \models^{\overleftrightarrow{\Diamond}} (\alpha \wedge \overleftarrow{\Box} \neg\alpha \wedge \Box\neg\beta)$. Comme $\alpha \equiv \beta$, on a donc $\Gamma_a^{CO} \models^{\overleftrightarrow{\Diamond}} (\alpha \wedge \overleftarrow{\Box} \neg\alpha \wedge \Box\neg\alpha)$.
  Soit $M = \langle W, \leq, val\rangle$ un modèle de Γ_a^{CO}. Il existe un monde w_0 de W tel que $M, w_0 \models \alpha \wedge \Box\neg\alpha$, donc en particulier $M, w_0 \models \alpha \wedge \neg\alpha$ ce qui est impossible.
- *$<_{\Gamma_a^{CO}}$ est une relation antisymétrique*
  Supposons que $<_{\Gamma_a^{CO}}$ soit une relation symétrique, alors il existe deux propositions α et β telles que $\Gamma_a^{CO} \models \alpha <_E \beta$ et $\Gamma_a^{CO} \models \beta <_E \alpha$. Soit $M = \langle W, \leq, val\rangle$ un modèle de Γ_a^{CO}. On a :

  1. *$\exists w_\alpha$ tel que $M, w_\alpha \models \alpha \wedge \Box\neg\beta$*
  2. *$\exists w_\beta$ tel que $M, w_\beta \models \beta \wedge \Box\neg\alpha$*

  Supposons que $w_\alpha \leq w_\beta$, alors dans ce cas, $M, w_\alpha \models \alpha \wedge \neg\alpha$, ce qui est impossible. La démonstration est identique pour le cas $w_\beta \leq w_\alpha$.
- *$<_{\Gamma_a^{CO}}$ est une relation transitive*
  Soient α, β et γ trois propositions telles que $\Gamma_a^{CO} \models \alpha <_E \beta$ et $\Gamma_a^{CO} \models \beta <_E \gamma$. On a donc $\alpha <_{\Gamma_a^{CO}} \beta$ et $\beta <_{\Gamma_a^{CO}} \gamma$.
  Soit $M = \langle W, \leq, val\rangle$ un modèle de Γ_a^{CO}. On a :

  1. *$\exists w_\alpha$ tel que $M, w_\alpha \models \alpha \wedge \Box\neg\beta \wedge \overleftarrow{\Box} \neg\alpha$*
  2. *$\exists w_\beta$ tel que $M, w_\beta \models \beta \wedge \Box\neg\gamma \wedge \overleftarrow{\Box} \neg\beta$*

  Supposons que $w_\beta \leq w_\alpha$, alors on a $M, w_\beta \models \beta \wedge \neg\beta$ car $M, w_\alpha \models \Box\neg\beta$, donc $w_\alpha \leq w_\beta$.
  Comme $M, w_\beta \models \Box\neg\gamma$, $M, w_\alpha \models \Box\neg\gamma$. Donc $M, w_\alpha \models \alpha \wedge \Box\neg\gamma \wedge \overleftarrow{\Box} \neg\alpha$, donc $\Gamma_a^{CO} \models \alpha <_E \gamma$.

□ .

Propriété 5.1.2. *Soit $\Gamma_a = [\alpha_1, \ldots, \alpha_n]$ la position de l'agent a. Les CO-modèles $\langle W, \leq, val\rangle$ de Γ_a^{CO} sont tels que : $\|f'_{\Gamma_a}(0)\| \leq \ldots \leq \|f'_{\Gamma_a}(m_a)\|$.*

Preuve. *Soit $\Gamma_a = [\alpha_1, \ldots, \alpha_n]$ la position de l'agent a. Soit $M = \langle W, \leq, val\rangle$ un modèle de Γ_a^{CO}. Soit $j \in \{0, \ldots, m_a - 1\}$, on note w_j le monde de W tel que $M, w_j \models f'_{\Gamma_a}(j) \wedge \overleftarrow{\Box} \neg f'_{\Gamma_a}(j) \wedge \Box\neg f'_{\Gamma_a}(j+1)$.*

Démontrons dans un premier temps que $\forall j \in \{1, \ldots, m_a - 1\}$ $w_{j-1} \leq w_j$. Soit $j \in \{0, \ldots, m_a - 1\}$, alors $M, w_{j-1} \models \Box\neg f'_{\Gamma_a}(j)$. Or $M, w_j \models f'_{\Gamma_a}(j)$ donc $w_{j-1} \leq w_j$.

Soit $j \in \{0, \ldots, m_a - 1\}$, deux cas se présentent :

- *soit $j \neq 0$ et alors dans ce cas $M, w_j \models f'_{\Gamma_a}(j) \wedge \overleftarrow{\Box} \neg f'_{\Gamma_a}(j)$. Donc $\forall w \in \|f'_{\Gamma_a}(j)\|$ $w \leq w_j$.*
  De plus, $M, w_{j-1} \models \Box\neg f'_{\Gamma_a}(j)$, donc $\forall w \in \|f'_{\Gamma_a}(j)\|$ $w_{j-1} \leq w$.
  Comme $\forall j \in \{1, \ldots, m_a - 1\}$ $w_{j-1} \leq w_j$, on peut écrire que $M, w_{j-1} \models \bigwedge_{l \in \{0,\ldots,j-1\}} \overleftarrow{\Box} \neg f'_{\Gamma_a}(l)$ et que $M, w_j \models \bigwedge_{l \in \{j+1,\ldots,m_a\}} \Box\neg f'_{\Gamma_a}(l)$. Donc pour tout $w \in W$, si $w \leq w_j$ et $w_{j-1} \leq w$, $M, w \models \bigwedge_{\substack{l \in \{0,\ldots,m_a\} \\ l \neq j}} \neg f'_{\Gamma_a}(l)$.
  Soit $w \in W$ tel que $w \leq w_j$ et $w_{j-1} \leq w$. Si pour tout $l \in \{0, \ldots, m_a\}$, on écrit $f'_{ak}(l) = \alpha_1(l) \wedge \ldots \wedge \alpha_n(l)$, alors $\forall l \in \{1, \ldots, m_a\}$ $l \neq j \Rightarrow M, w \models \neg\alpha_1(l) \vee \ldots \vee \neg\alpha_n(l)$.
  Or par construction de f'_{Γ_a}, $\forall l \in \{1, \ldots, m_a\}$ $l \neq j \Rightarrow \exists l_j \in \{1, \ldots n\}$ tq $\alpha_{l_j}(l) \equiv \neg\alpha_{l_j}(j)$.
  Donc comme $\forall l \in \{1, \ldots, m_a\}$ $l \neq j \Rightarrow M, w \models \neg\alpha_1(l) \vee \ldots \vee \alpha_n(l)$, $M, w \models \alpha_1(j) \wedge \ldots \wedge \alpha_n(j)$. Donc $M, w \models f'_{\Gamma_a}(j)$.
  Les modèles de $f'_{\Gamma_a}(j)$ sont donc les mondes $w \in W$ tel que $w \leq w_j$ et $w_{j-1} \leq w$.
- *soit $j = 0$, $M, w_0 \models f'_{\Gamma_a}(0) \wedge \overleftarrow{\Box} \neg f'_{\Gamma_a}(0)$ donc $\forall w \in \|f'_{\Gamma_a}(0)\|$, $w \leq w_0$.*
  De plus, on peut écrire que $M, w_0 \models \bigwedge_{l \in \{1,\ldots,m_a\}} \Box\neg f'_{\Gamma_a}(l)$ donc d'après ce qui précéde, $\forall w \in W$, $w \leq w_0 \Rightarrow M, w \models f'_{\Gamma_a}(0)$.

Enfin, $M, w_{m_a-1} \models \Box\neg f'_{\Gamma_a}(m_a)$ donc $\forall w \in \|f'_{\Gamma_a}(m_a)\|$ $w_{m_a-1} \leq w$. De plus, $M, w_{m_a-1} \models \bigwedge_{l \in \{0,\ldots,m_a-1\}} \overleftarrow{\Box} \neg f'_{\Gamma_a}(l)$, donc $\forall w \in W$ $w_{m_a-1} \leq w \Rightarrow M, w \models f'_{\Gamma_a}(m_a)$.
$\Box$.

Propriété 5.2.1. *L'opérateur I vérifie les schémas d'axiomes et la régle suivants :*

OC $I(\alpha) \wedge I(\beta) \rightarrow I(\alpha \wedge \beta)$

ON $I(\top)$

OD $\neg I(\bot)$

ROM $\frac{\vdash \alpha \rightarrow \beta}{\vdash I(\alpha) \rightarrow I(\beta)}$

Preuve. *1. démonstration de **OC***

Supposons que $\vdash I(\alpha) \wedge I(\beta)$ alors $\models I(\alpha) \wedge I(\beta)$. Donc $\models \overleftrightarrow{\Diamond} \Box\alpha \wedge \overleftrightarrow{\Diamond} \Box\beta$, soit $\models \overleftrightarrow{\Diamond} (\Box\alpha \wedge \Box\beta)$. D'où $\models \overleftrightarrow{\Diamond} \Box\alpha \wedge \beta$.

*2. démonstration de **ON***

Supposons que l'on ait $\not\vdash I(\top)$, alors $\models \neg I(\top)$ soit $\models \overleftrightarrow{\Diamond} \Box\neg\top$, ce qui est impossible (il existerait alors un monde qui ne vérifie pas $\top$).

3. *démonstration de* ***OD***
   La démonstration est identique à la précédente.
4. *démonstration de* ***ROM***
   Cette propriété découle directement du fait que les opérateurs $\Box$ *et* $\overleftarrow{\Box}$ *vérifient les axiomes (K) et (K') (cf. section 4.2.3).*

□ .

Propriété 5.4.1. *Soit* $\mathcal{C}$ *un ensemble de contraintes du domaine.* $\mathcal{F}'_{\mathcal{C}}(a)$ *est non vide.*

Preuve. *Supposons que* $\mathcal{F}'(a)$ *soit vide. Alors dans ce cas,* $\forall i \in \{0,\ldots,m_a\}\ \ \exists\varphi_i$ *telle que* $\mathcal{C} \models \Box\varphi_i$ *et* $\varphi_i \wedge f'_a(i)$ *n'est pas satisfaisable.*
Dans un premier temps, remarquons que $\bigvee_{i\in\{1,\ldots,m_a\}} f'_a(i)$ *est équivalent à* $\top$*. C'est évident par définition de la fonction* f'_a*.*
On sait que $\forall i \in \{0,\ldots,m_a\}\ \ \varphi_i \wedge f'_a(i)$ *n'est pas satisfaisable, donc* $\forall i \in \{0,\ldots,m_a\}\ \ \models \varphi_i \rightarrow \neg f'_a(i)$*. Donc* $\models \bigwedge_{i\in\{1,\ldots,m_a\}} \varphi_i \rightarrow \neg \bigvee_{i\in\{1,\ldots,m_a\}} f'_a(i)$.
Or $\mathcal{C}$ *est tel que les contraintes du domaine sont cohérentes entre elles, donc ceci est impossible.*
□ .

Propriété 5.4.2. *Soient* $\mathcal{C}$ *un ensemble de contraintes du domaine et* $\mathcal{R}$ *une réglementation cohérente avec* $\mathcal{C}$*.* $\mathcal{F}'_{\mathcal{C},\mathcal{R}}(a)$ *est un ensemble non vide.*

Preuve. *Supposons que* $\mathcal{F}'_{\mathcal{C},\mathcal{R}}(a)$ *soit vide. Alors dans ce cas,* $\forall f'_a(i) \in \mathcal{F}'_{\mathcal{C}}(a)\ \ \mathcal{R} \models \neg T(f'_a(i))$*. Donc* $\forall f'_a(i) \in \mathcal{F}'_{\mathcal{C}}(a)\ \ \mathcal{R} \models I(\neg f'_a(i))$ *et on en déduit que* $\mathcal{R} \models I(\neg(\bigvee_{f'_a(i)\in\mathcal{F}'_{\mathcal{C}}(a)} f'_a(i)))$.
Deux cas se présentent :
– *soit* $\mathcal{F}'_{\mathcal{C}}(a) = \{f'_a(0),\ldots,f'_a(m_a)\}$*. Dans ce cas,* $\mathcal{R} \models I(\neg(\bigvee_{i\in\{1,\ldots,m_a\}} f'_a(i)))$*. Or* $\bigvee_{i\in\{1,\ldots,m_a\}} f'_a(i)$ *est équivalent à* $\top$*, donc* $\mathcal{R} \models I(\bot)$*, ce qui est impossible ;*
– *soit* $\exists\{i_0,\ldots,i_l\} \subset \{0,\ldots,m_a\}\ \exists\{\overleftarrow{\Box}\varphi_{i_0},\ldots,\overleftarrow{\Box}\varphi_{i_l}\} \subset \mathcal{C}$ *tels que* $\forall i \in \{i_0,\ldots,i_l\}\ f_a(i) \wedge \varphi_i$ *soit incohérente. Donc* $\forall i \in \{i_0,\ldots,i_l\}\ \models f'_a(i) \rightarrow \neg\varphi_i$*. Or* $\mathcal{R} \models I(\neg(\bigvee_{f'_a(i)\in\mathcal{F}'_{\mathcal{C}}(a)} f'_a(i)))$*. Comme* $\bigvee_{i\in\{0,\ldots,m_a\}} f'_a(i)$ *est équivalent à* $\top$*, on en déduit que* $\mathcal{R} \models I(\bigvee_{i\in\{i_0,\ldots,i_l\}} f'_a(i))$*. Donc* $\mathcal{R} \models I(\bigvee_{i\in\{i_0,\ldots,i_l\}} \neg\varphi_i)$ *et* $\mathcal{R} \models I(\neg\bigwedge_{i\in\{i_0,\ldots,i_l\}} \varphi_i)$*. Or* $\bigwedge_{i\in\{i_0,\ldots,i_l\}} \varphi_i$ *est une contrainte du domaine (d'après les propriétés de l'opérateur* $\overleftarrow{\Box}$*), donc* $\mathcal{R}$ *n'est pas cohérente avec* $\mathcal{C}$*, ce qui contredit l'hypothèse de départ.*
□ .

A.2 Preuves de la partie II

Théorème 8.1.1. *Soit* ψ *la formule définie en définition 8.1.9. Soit* φ *une formule de PROP et db une base de croyances (primitive ou pas). On a alors :*

$$\models_{MF} \psi \rightarrow B_{db}\varphi \iff \vdash_{MF} \psi \rightarrow B_{db}\varphi$$

$$\models_{MF} \psi \rightarrow \neg B_{db}\varphi \iff \vdash_{MF} \psi \rightarrow \neg B_{db}\varphi$$

Preuve. *La démonstration se fait en deux étapes : on démontre tout d'abord la complétude (sens ⇒) en montrant que tous les schémas d'axiomes sont valides et que les règles d'inférence préservent la validité, puis on démontre la complétude (sens ⇔) par induction.*

A.2.1 Validité de MF

Dans toutes les démonstrations, on considérera que les modèles M de MF sont des modèles de ψ (pour respecter la restriction de la preuve à un fragment de la logique).

Schémas d'axiomes

On démontre que tous les schémas d'axiomes sont des formules valides.
Axiome (A1)
L'axiome est $\vdash_{MF} B_{db}\neg\varphi \rightarrow \neg B_{db}\varphi$.

$$\begin{aligned} M, w \models_{MF} B_{db}\neg\varphi &\Leftrightarrow g_{db}(w) \subseteq val(\neg\varphi) \\ &\Leftrightarrow g_{db}(w) \not\subseteq val(\varphi) \\ &\Leftrightarrow M, w \not\models_{MF} B_{db}\varphi \\ &\Leftrightarrow M, w \models_{MF} \neg B_{db}\varphi \end{aligned}$$

□ .
Axiome (A2)
L'axiome est $\vdash_{MF} B_{db}\varphi \wedge B_{db}(\varphi \rightarrow \varphi') \rightarrow B_{db}\varphi'$.

$$\begin{aligned} M, w \models_{MF} B_{db}\varphi &\Leftrightarrow g_{db}(w) \subseteq val(\varphi) \\ M, w \models_{MF} B_{db}(\varphi \rightarrow \varphi') &\Leftrightarrow g_{db}(w) \subseteq val(\varphi \rightarrow \varphi') \\ &\Leftrightarrow g_{db}(w) \subseteq val(\varphi \wedge \varphi') \\ &\Rightarrow g_{db}(w) \subseteq val(\varphi') \\ &\Rightarrow M, w \models_{MF} B_{db}\varphi' \end{aligned}$$

□ .
Axiome (A3)
L'axiome est $\vdash_{MF} B^i_{db}l \leftrightarrow B^i_{db}\neg\neg l$.

$$\begin{aligned} M, w \models_{MF} B^i_{db}l &\Leftrightarrow val(l) \in^i f_{db}(w) \\ &\Leftrightarrow val(\neg\neg l) \in^i f_{db}(w) \\ &\Leftrightarrow M, w \models_{MF} B^i_{db}\neg\neg l \end{aligned}$$

□ .
Axiome (A4)
L'axiome est $\vdash_{MF} B^i_{db}l \rightarrow \neg B^j_{db}l$ si $i \neq j$. On prend donc i et j tels que $i \neq j$.

$$M, w \models_{MF} B^i_{db} l \quad \Leftrightarrow \quad val(l) \in^i f_{db}(w)$$

Supposons que l'on ait $M, w \models_{MF} B^j_{db} l$. Dans ce cas, $val(l) \in^j f_{db}(w)$. Or $val(l) \in^i f_{db}(w)$ signifie que il y a exactement i occurences de $val(l)$ dans $f_{db}(w)$, donc $M, w \not\models_{MF} B^j_{db} l$, donc $M, w \models_{MF} \neg B^j_{db} l$.
□ .

Axiome (A5)
L'axiome est $\vdash_{MF} B^i_{db} l \wedge B^j_{db'} l \rightarrow B^k_{db*db'} l$ avec $k = i + j$. On prend donc i et j deux entiers et $k = i + j$.

$$\begin{array}{rcl} M, w \models_{MF} B^i_{db} l & \Leftrightarrow & val(l) \in^i f_{db}(w) \\ M, w \models_{MF} B^j_{db'} l & \Leftrightarrow & val(l) \in^j f_{db'}(w) \\ & \Rightarrow & val(l) \in^{(i+j)} f_{db} \sqcup f_{db'}(w) \\ & \Rightarrow & val(l) \in^k f_{db*db'}(w) \\ & \Rightarrow & M, w \models_{MF} B^k_{db*db'} l \end{array}$$

□ .

Axiome (A6)
L'axiome est $\vdash_{MF} B^i_{db} l \wedge B^j_{db} \neg l \rightarrow B_{db} l$ si $i > j$. On prend donc deux entiers i et j tels que $i > j$.

$$\begin{array}{rcl} M, w \models_{MF} B^i_{db} l & \Leftrightarrow & val(l) \in^i f_{db}(w) \\ M, w \models_{MF} B^j_{db} \neg l & \Leftrightarrow & val(\neg l) \in^j f_{db}(w) \end{array}$$

D'après la construction de $f_{db}(w)$, comme on ne considère que des modèles de ψ, $f_{db}(w)$ ne contient que des éléments de la forme $val(l)$ où l est un littéral, donc :

$$f_{db}(w) = [\underbrace{val(l) \dots val(l)}_{i \ fois} \ \underbrace{val(\neg l) \dots val(\neg l)}_{j \ fois} \ val(l_1) \dots val(l_n)]$$

où $\forall k \in \{1 \dots n\}$ l_k est un littéral différent de l et de $\neg l$ (il se peut qu'on retrouve le même littéral plusieurs fois).
Soit w_1 tel que $w_1 \models_{MF} \neg l$. On va montrer qu'il existe w_2 tel que $w_2 \leq_{f_{db}(w)} w_1$ et $w_2 \models_{MF} l$.
Considérons w_2 tel que $w_2 \in val(l)$ et tel que $\forall p$, p littéral différent de l et de $\neg l$, $w_1 \in val(p) \Rightarrow w_2 \in val(p)$.
Soit $k \in \{1 \dots n\}$. Soit $w_k \in W$ tel que $d(w_1, w_k) = \min_{w' \in val(l_k)} d(w_1, w')$. Deux cas se présentent alors :

- soit $w_k \models_{MF} l$. Dans ce cas, on a bien $d(w_2, w_k) < d(w_1, w_k)$ donc $\min_{w' \in val(l_k)} d(w_2, w') < \min_{w' \in val(l_k)} d(w_1, w')$;
- soit $w_k \models_{MF} \neg l$. Comme $l_k \neq l$, il existe $w'_k \in val(l_k)$ tel que w'_k s'identifie à w_k pour tous les littéraux différents de l mais tel que $w'_k \models_{MF} l$. Donc :

$$\begin{array}{rl} & d(w_2, w'_k) = d(w_1, w_k) \\ \Rightarrow & d(w_2, w'_k) = \min_{w' \in val(l_k)} d(w_1, w') \\ \Rightarrow & \min_{w' \in val(l_k)} d(w_2, w') \leq \min_{w' \in val(l_k)} d(w_1, w') \end{array}$$

De plus :

$$\begin{aligned} dsum(w_1, f_{db}(w)) &= i * \min_{w' \in val(l)} d(w_1, w') + j * \min_{w' \in val(\neg l)} d(w_1, w') \\ &\quad + \sum_{k=1}^{n} \min_{w' \in val(l_k)} d(w_1, w') \\ &= i + \sum_{k=1}^{n} \min_{w' \in val(l_k)} d(w_1, w') \end{aligned}$$

et :

$$\begin{aligned} dsum(w_2, f_{db}(w)) &= i * \min_{w' \in val(l)} d(w_2, w') + j * \min_{w' \in val(\neg l)} d(w_2, w') \\ &\quad + \sum_{k=1}^{n} \min_{w' \in val(l_k)} d(w_2, w') \\ &= j + \sum_{k=1}^{n} \min_{w' \in val(l_k)} d(w_2, w') \end{aligned}$$

Or, $\forall k \in \{1 \dots n\}$ $\min_{w' \in val(l_k)} d(w_2, w') < \min_{w' \in val(l_k)} d(w_1, w')$. Comme $j < i$, on a $dsum(w_2, f_{db}(w)) < dsum(w_1, f_{db}(w))$.
Donc $w_2 \leq_{f_{db}(w)} w_1$. Donc $g_{db} = \min_{\leq_{f_{db}(w)}} W \subset val(l)$. On a donc $M, w \models_{MF} B_{db} l$.
□ .

Axiome (A7)

L'axiome est $\vdash_{MF} B^i_{db} l \wedge B^i_{db} \neg l \rightarrow \neg B_{db} l$.
De la même façon que précédemment, nous allons montrer cette fois ci que $g_{db}(w) \not\subset val(l)$. Nous reprenons les notations précédentes pour la représentation de $f_{db}(w)$, sauf que $i = j$.
Soit $w_1 \in W$ tel que $w_1 \models_{MF} l$. On va montrer qu'il existe $W_2 \not\models_{MF} l$ tel que $dsum(w_2, f_{db}(w)) \leq d(w_1, f_{db}(w))$.
Soit $k \in \{1 \dots n\}$. Il existe $w_k \in val(l_k)$ tel que $d(w_1, w_k) = \min_{w' \in val(l_k)} d(w_1, w')$. Deux cas se présentent alors :

– soit $w_k \models_{MF} l$ et alors on choisit $w'_k \in val(l_k)$ tel que w'_k soit identique à w sauf en l où $w'_k \models_{MF} \neg l$. Dans ce cas :

$$\begin{aligned} & d(w_2, w'_k) = d(w_1, w_k) \\ \Rightarrow \quad & \min_{w' \in val(l_k)} d(w_2, w') \leq \min_{w' \in val(l_k)} d(w_1, w') \end{aligned}$$

– soit $w_k \models_{MF} \neg l$ et alors :

$$\begin{aligned} & d(w_2, w'_k) < d(w_1, w_k) \\ \Rightarrow \quad & \min_{w' \in val(l_k)} d(w_2, w') < \min_{w' \in val(l_k)} d(w_1, w') \end{aligned}$$

Par la même démonstration que pour l'axiome (A5), on peut écrire que $g_{db}(w) \not\subset val(l)$, donc que $M, w \not\models_{MF} B_{db} l$.
□ .

Axiome (A8)

L'axiome est $\vdash_{MF} B_{db}(l_1 \vee \ldots \vee l_n) \rightarrow B_{db}l_1 \vee \ldots \vee B_{db}l_n$ avec l_1, ..., l_n qui ne sont pas complémentaires.
On va montrer que dans ce cas $g_{bd}(w)$ est l'ensemble de tous les modèles d'une conjonction de littéraux.
Remarquons que comme on ne travaille qu'avec des modèles de ψ, les fonctions f_{db} sont telles que pour tout monde $f_{db}(w)$ n'est constitué que valuations de littéraux. Supposons dans un premier temps que pour tout littéral l et pour tout monde w,
$val(l) \in^i f_{db}(w) \Rightarrow val(\neg l) \in^i f_{db}(w)$. En particulier, $\forall k \in \{1 \ldots n\}$
$val(l_k) \in^i f_{db}(w) \Rightarrow val(\neg l_k) \in^i f_{db}(w)$. On peut remarquer que
$\forall k \in \{1 \ldots n\}\ W = val(l_k) \cup val(\neg l_k)$.
Soient w_1 et w_2 deux mondes de W. On va montrer que
$dsum(w_1, f_{db}(w)) = dsum(w_2, f_{db}(w))$. Nous allons distinguer 4 cas :

1. soit $w_1 \in val(l_1)$ et $w_2 \in val(l_1)$.

$$dsum(w_1, f_{db}(w)) = i * 1 + i * 0 + \sum_{k=2}^{n} \min_{w \in val(l_k)} d(w_1, w)$$

$$dsum(w_2, f_{db}(w)) = i * 1 + i * 0 + \sum_{k=2}^{n} \min_{w \in val(l_k)} d(w_2, w)$$

   Or $\forall k \in \{1 \ldots n\}\ l_k \neq l_1$ par hypothèse donc $\forall k \in \{1 \ldots n\}$
   $\sum_{k=2}^{n} \min_{w \in val(l_k)} d(w_1, w) = 0$ et $\sum_{k=2}^{n} \min_{w \in val(l_k)} d(w_2, w) = 0$. Donc
   $dsum(w_1, f_{db}(w)) = dsum(w_2, f_{db}(w))$.

2. soit $w_1 \in val(l_1)$ et $w_2 \in val(\neg l_1)$.

$$dsum(w_1, f_{db}(w)) = i * 1 + i * 0 + \sum_{k=2}^{n} \min_{w \in val(l_k)} d(w_1, w)$$

$$dsum(w_2, f_{db}(w)) = i * 0 + i * 1 + \sum_{k=2}^{n} \min_{w \in val(l_k)} d(w_2, w)$$

   Donc $dsum(w_1, f_{db}(w)) = dsum(w_2, f_{db}(w))$ par la même démonstration que précédemment.

3. soit $w_1 \in val(l_1)$ et $w_2 \in val(\neg l_1)$.

$$dsum(w_1, f_{db}(w)) = i * 1 + i * 0 + \sum_{k=2}^{n} \min_{w \in val(l_k)} d(w_1, w)$$

$$dsum(w_2, f_{db}(w)) = i * 0 + i * 1 + \sum_{k=2}^{n} \min_{w \in val(l_k)} d(w_2, w)$$

   Donc $dsum(w_1, f_{db}(w)) = dsum(w_2, f_{db}(w))$ par la même démonstration que précédemment.

4. soit $w_1 \in val(\neg l_1)$ et $w_2 \in val(\neg l_1)$.

$$dsum(w_1, f_{db}(w)) = i * 0 + i * 1 + \sum_{k=2}^{n} \min_{w \in val(l_k)} d(w_1, w)$$

$$dsum(w_2, f_{db}(w)) = i * 1 + i * 0 + \sum_{k=2}^{n} \min_{w \in val(l_k)} d(w_2, w)$$

Donc $dsum(w_1, f_{db}(w)) = dsum(w_2, f_{db}(w))$ par la même démonstration que précédemment.

Dans ce cas, pour tout monde w_1 et tout monde w_2 de W,
$dsum(w_1, f_{db}(w)) = dsum(w_2, f_{db}(w))$. Donc $g_{db}(w) = W$.

$$\begin{array}{rcl} M, w \models_{MF} B_{db}(l_1 \vee \ldots \vee l_n) & \Leftrightarrow & g_{db}(w) \subset val(l_1 \vee \ldots \vee l_n) \\ & \Rightarrow & W \subset val(l1 \vee \ldots \vee l_n) \\ & \Rightarrow & l1 \vee \ldots \vee l_n \text{ est une tautologie} \end{array}$$

Or les littéraux $l_1 \ldots l_n$ ne sont pas complémentaires, donc $l1 \vee \ldots \vee l_n$ ne peut pas être une tautologie.
L'hypothèse est fausse donc $\exists l$ l littéral tel que $val(l) \in^i f_{db}$ et $val(\neg l) \in^j f_{db}$ avec $j \geq 0$ et $i > j$ ou $j > i$. On peut considérer que $i > j$ sinon on « échange » l et $\neg l$.
Il existe donc k literaux L_1, ..., L_k tels que $\forall m \in \{1 \ldots k\}$ $\exists i_m$ $\exists j_m$ tels que $val(L_m) \in^{i_m} f_{db}(w)$, $val(\neg L_m) \in^{j_m} f_{db}(w)$ et $i_m > j_m$.
Comme l'axiome (A4) est valide, on peut écrire que $g_{db}(w) \subset val(L_1 \wedge \ldots \wedge L_k)$. Nous allons montrer que $g_{db}(w) = val(L_1 \wedge \ldots \wedge L_k)$.
Soit $w_1 \in W$ tel que $w_1 \models_{MF} L_1 \wedge \ldots \wedge L_k$. Soit $w_2 \in W$ tel que $w_2 \not\models_{MF} L_1 \wedge \ldots \wedge L_k$. Par exemple, $w_2 \not\models_{MF} L_m$ avec $m \in \{1 \ldots k\}$.

$$\begin{array}{rcl} dsum(w_1, f_{db}(w)) & = & i * \min_{w \in val(L_m)} d(w_1, w) + j * \min_{w \in val(\neg L_m)} d(w_1, w) \\ & & + \sum_{i \in \{1 \ldots k\}\ i \neq m} \min_{w \in val(L_i)} d(w_1, w) \\ & = & j \\ dsum(w_2, f_{db}(w)) & = & i * \min_{w \in val(L_m)} d(w_2, w) + j * \min_{w \in val(\neg L_m)} d(w_1, w) \\ & & + \sum_{i \in \{1 \ldots k\}\ i \neq m} \min_{w \in val(L_i)} d(w_2, w) \\ & = & i \end{array}$$

Or $i > j$ donc $dsum(w_1, f_{db}(w)) < dsum(w_2, f_{db}(w))$. On a donc
$g_{db}(w) = val(L_1 \wedge \ldots \wedge L_k)$.
Donc :

$$\begin{array}{rcl} M, w \models_{MF} B_{db}(l_1 \vee \ldots \vee l_n) & \Leftrightarrow & g_{db}(w) \subset val(l_1 \vee \ldots \vee l_n) \\ & \Leftrightarrow & val(L_1 \wedge \ldots \wedge L_k) \subset val(l_1 \vee \ldots \vee l_n) \\ & \Leftrightarrow & \models_{MF} L_1 \wedge \ldots \wedge L_k \rightarrow l_1 \vee \ldots \vee l_n \\ & \Rightarrow & \exists p \in \{1 \ldots k\}\ \exists r \in \{1 \ldots n\}\ L_p = l_r \\ & \Rightarrow & \exists r \in \{1 \ldots n\}\ \models_{MF} g_{db}(w) \rightarrow l_p \\ & \Rightarrow & \exists r \in \{1 \ldots n\}\ g_{db}(w) \subset val(l_r) \\ & \Rightarrow & \exists r \in \{1 \ldots n\}\ M, w \models_{MF} B_{db} l_r \\ & \Rightarrow & M, w \models_{MF} B_{db}(l_1) \vee \ldots \vee B_{db}(l_n) \end{array}$$

□ .

Règles d'inférence

(MP)
La règle est : si $\vdash_{MF} \varphi$ et $\vdash_{MF} \varphi \rightarrow \varphi'$ alors $\vdash_{MF} \varphi'$.
La démonstration est évidente.
□ .
(Nec)
La règle est : si $\vdash_{MF} \varphi$ alors $\vdash_{MF} B_{db}\varphi$
Supposons que $\models_{MF} \varphi$, i.e. que pour tout modèle M et tout monde w, on ait $M, w \models_{MF} \varphi$. On a alors en particulier $g_{db}(w) \subset val(\varphi)$ pour tout db donc $\models_{MF} B_{db}\varphi$.
□ .

A.2.2 Complétude de MF

Pour pouvoir démontrer la complétude de MF, nous allons avoir besoin de plusieurs lemmes.

Lemme A.2.1. *Soit $M = \langle W, val, R, B\rangle$ un modèle de MF.*

$$M \models_{MF} \psi \;\Rightarrow\; f_{db_i}(w) = \{val(l) \;:\; l \in DB_i\} \text{ pour toute base primitive } db_i$$

Preuve. *Soit M un modèle de MF tel que $M \models_{MF} \psi$.*

$$\begin{array}{rcl} M \models_{MF} \psi & \Rightarrow & M \models_{MF} \bigwedge_{i=1}^{n} (\bigwedge_{l \in DB_i} B^1_{db_i} l) \wedge \bigwedge_{l \notin DB_i} B^0_{db_i} l) \\ & \Leftrightarrow & \forall i \in \{1 \ldots n\}\ \forall l \in DB_i\ M \models_{MF} B^1_{db_i} l \\ & & \qquad\qquad \forall l \notin DB_i\ M \models_{MF} B^0_{db_i} l \\ & \Leftrightarrow & \forall i \in \{1 \ldots n\}\ \forall l \in DB_i\ val(l) \in^1 f_{db_i}(w) \\ & & \qquad\qquad \forall l \notin DB_i\ val(l) \notin f_{db_i}(w) \end{array}$$

Or d'après la contrainte C1, comme db_i est une base primitive, on a $f_{db_i}(w) = \{E \;:\; \exists l$ littéral de $PROP$ tel que $E = val(l)\}$.
Soit $l \in L$, l un littéral. Deux cas se présentent :
– $l \in DB_i$ et alors $val(l) \in^1 fdb_i(w)$;
– $l \notin DB_i$ et alors $val(l) \notin fdb_i(w)$.
Donc $fdb_i(w) = \{val(l) \;:\; l \in DB_i\}$.
□ .

Lemme A.2.2. *Soit DB_i une base primitive cohérente. Soit M un modèle de MF tel que $M \models_{MF} \psi$. Alors on a $val(l) \in f_{db_i}(w) \;\Rightarrow\; val(\neg l) \notin f_{db_i}(w)$.*

Preuve.

$$\begin{array}{rcl} M \models_{MF} \psi & \Rightarrow & f_{db_i}(w) = \{val(l) \;:\; l \in DB_i\} \\ & \Rightarrow & val(l) \in f_{db_i}(w) \;\Rightarrow\; l \in DB_i \end{array}$$

Or DB_i est cohérente donc :

$$\begin{array}{rcl} l \in DB_i & \Leftrightarrow & \neg l \notin db_i \\ & \Leftrightarrow & val(\neg l) \notin f_{db_i}(w) \end{array}$$

□ .

Lemme A.2.3. *Soit E un multi-ensemble de valuations de littéraux tel que $val(l) \in E \Rightarrow val(\neg l) \notin E$. Alors $\min_{\leq_E} W = val(\bigwedge_{val(l)\in E} l)$.*

Preuve. *Soit E un multi-ensemble de valuations de littéraux tel que $val(l) \in E \Rightarrow val(\neg l) \notin E$. $E = \{val(l_1) \dots val(l_n)\}$ où $\forall i \in \{1 \dots n\}\ \forall j \in \{1 \dots n\}\ l_i \neq \neg l_j$.*
Soit $w_1 \in W$ tel que $w_1 \models_{MF} l_1 \wedge \dots \wedge l_n$. Comme $l_1 \wedge \dots \wedge l_n$ est cohérent, on a :

$$\begin{aligned} w_1 \in \bigcap_{i=1}^{n} val(l_i) &\Rightarrow dsum(w_1, E) = 0 \\ &\Rightarrow \forall w_2 \in W\ dsum(w_1, E) \leq dsum(w_2, E) \end{aligned}$$

Les modèles de $l_1 \wedge \dots \wedge l_n$ sont donc parmi les plus proches de E et leur distance à E est nulle.
Soit $w \in \min_{\leq_E} W$ avec $dsum(w, E) = 0$ alors on a :

$$\begin{aligned} \forall i \in \{1 \dots n\}\ \min_{w' \in val(l_i)} d(w, w') = 0 &\Rightarrow \forall i \in \{1 \dots n\}\ w \models_{MF} l_i \\ &\Rightarrow w \models_{MF} l_1 \wedge \dots \wedge l_n \end{aligned}$$

Les modèles les plus proches de E sont des modèles de $l_1 \wedge \dots \wedge l_n$.
□ .

Lemme A.2.4. *Soit l un littéral de L.*

$$\begin{aligned} \models_{MF} \psi \rightarrow B^i_{db} l &\Rightarrow \vdash_{MF} \psi \rightarrow B^i_{db} l \\ \models_{MF} \psi \rightarrow \neg B^i_{db} l &\Rightarrow \vdash_{MF} \psi \rightarrow \neg B^i_{db} l \end{aligned}$$

Preuve. *On démontre ce lemme par induction sur le nombre de bases primitives fusionnées.*
Posons pour n entier tel que $n > 0$:

Définition A.2.1. *$\mathcal{H}(n)$: pour tout l littéral de L*

$$\begin{aligned} \models_{MF} \psi \rightarrow B^i_{db_1+\dots+db_n} l &\Rightarrow \vdash_{MF} \psi \rightarrow B^i_{db_1+\dots+db_n} l \\ \models_{MF} \psi \rightarrow \neg B^i_{db_1+\dots+db_n} l &\Rightarrow \vdash_{MF} \psi \rightarrow \neg B^i_{db_1+\dots+db_n} l \end{aligned}$$

avec $\forall i \in \{1 \dots n\}$ db_i est une base primitive.

Démonstration de la base d'induction, i.e. *$\mathcal{H}(1)*
Soit db une base primitive.

1. *si $\models_{MF} \psi \rightarrow B^i_{db} l$, alors $val(l) \in^i f_{db}(w)$. Par définition de ψ, $i = 0$ ou $i = 1$.*
   (a) *si $i = 1$, alors $\psi = \dots \wedge B^1_{db} l$ donc $\vdash_{MF} \psi \rightarrow B^1_{db} l$;*
   (b) *si $i = 0$, alors $\psi = \dots \wedge B^0_{db} l$ donc $\vdash_{MF} \psi \rightarrow B^0_{db} l$*
2. *si $\models_{MF} \psi \rightarrow \neg B^i_{db} l$ alors :*

$$\begin{aligned} \models_{MF} \psi \rightarrow \neg B^i_{db} l &\Leftrightarrow val(l) \notin^i f_{db}(w) \\ &\Leftrightarrow \exists j \geq 0\ j \neq i\ val(l) \in^j f_{db}(w) \\ &\Leftrightarrow \exists j \geq 0\ j \neq i\ \models_{MF} \psi \rightarrow B^j_{db} l \\ &\Rightarrow \exists j \geq 0\ j \neq i\ \vdash_{MF} \psi \rightarrow B^j_{db} l \text{ d'après ce qui précéde} \\ \text{Axiome } \textbf{(A4)} &\Rightarrow \vdash_{MF} \psi \rightarrow \neg B^i_{db} l \end{aligned}$$

$\mathcal{H}(2)$ *est donc vraie.*

Démonstration du pas d'induction

Supposons qu'il existe $n \geq 1$ *tel que* $\mathcal{H}(n)$ *soit vraie. Démontrons que* $\mathcal{H}(n+1)$ *est vraie. Soit db une base résultant de la fusion de n bases primitives et* db_{n+1} *une base primitive.*

1. *si* $\models_{MF} \psi \rightarrow B^i_{db+db_{n+1}} l$ *alors :*

$$\begin{array}{rcl}
\models_{MF} \psi \rightarrow B^i_{db+db_{n+1}} l & \Leftrightarrow & val(l) \in^i f_{db}(w) \sqcup f_{db_{n+1}}(w) \\
& \Leftrightarrow & \exists i_1 \geq 0 \;\; \exists i_2 \geq 0 \;\; i_1 + i_2 = i \;\; val(l) \in^{i_1} f_{db}(w) \text{ et} \\
& & \qquad val(l) \in^{i_2} f_{db_{n+1}}(w) \\
& \Leftrightarrow & \exists i_1 \geq 0 \;\; \exists i_2 \geq 0 \;\; i_1 + i_2 = i \;\; \models_{MF} \psi \rightarrow B^{i_1}_{db} l \text{ et} \\
& & \qquad \models_{MF} \psi \rightarrow B^{i_1}_{db_{n+1}} l \\
\mathcal{H}(n) \text{ et } \mathcal{H}(1) & \Rightarrow & \exists i_1 \geq 0 \;\; \exists i_2 \geq 0 \;\; i_1 + i_2 = i \;\; \vdash_{MF} \psi \rightarrow B^{i_1}_{db} l \text{ et} \\
& & \qquad \vdash_{MF} \psi \rightarrow B^{i_1}_{db_{n+1}} l \\
\textit{Axiome } \textbf{(A5)} & \Rightarrow & \vdash_{MF} \psi \rightarrow B^i_{db+db_{n+1}} l
\end{array}$$

2. *si* $\models_{MF} \psi \rightarrow \neg B^i_{db+db_{n+1}} l$ *alors :*

$$\begin{array}{rcl}
\models_{MF} \psi \rightarrow \neg B^i_{db+db_{n+1}} l & \Leftrightarrow & val(l) \notin^i f_{db} \sqcup f_{db_{n+1}} \\
& \Leftrightarrow & \exists j \geq 0 \;\; j \neq i \;\; val(l) \in^j f_{db} \sqcup f_{db_{n+1}} \\
& \Leftrightarrow & \exists j_1 \geq 0 \;\; \exists j_2 \geq 0 \;\; j_1 + j_2 \neq i \;\; val(l) \in^{j_1} f_{db}(w) \text{ et} \\
& & \qquad val(l) \in^{j_2} f_{db_{n+1}}(w) \\
& \Leftrightarrow & \exists j_1 \geq 0 \;\; \exists j_2 \geq 0 \;\; j_1 + j_2 \neq i \;\; \models_{MF} \psi \rightarrow B^{j_1}_{db} l \text{ et} \\
& & \qquad \models_{MF} \psi \rightarrow B^{j_2}_{db} l \\
\mathcal{H}(n) \text{ et } \mathcal{H}(1) & \Rightarrow & \exists j_1 \geq 0 \;\; \exists j_2 \geq 0 \;\; j_1 + j_2 \neq i \;\; \vdash_{MF} \psi \rightarrow B^{j_1}_{db} l \text{ et} \\
& & \qquad \vdash_{MF} \psi \rightarrow B^{j_2}_{db} l \\
\textit{Axiome } \textbf{(A5)} & \Rightarrow & \exists j_1 \geq 0 \;\; \exists j_2 \geq 0 \;\; j_1 + j_2 \neq i \;\; \vdash_{MF} \psi \rightarrow B^{j_1+j_2}_{db+db_{n+1}} l \\
\textit{Axiome } \textbf{(A4)} & \Rightarrow & \vdash_{MF} \psi \rightarrow \neg B^i_{db+db_{n+1}} l
\end{array}$$

$\mathcal{H}(n+1)$ *est donc vraie.*

La propriété $\mathcal{H}$ *est vraie au rang 1. Si pour* $n \geq 1$ $\mathcal{H}(n)$ *est vraie alors* $\mathcal{H}(n+1)$ *est vraie. D'après le principe d'induction,* $\mathcal{H}(n)$ *est vraie pour tout* $n \geq 1$.

□ .

Munis de ces lemmes, nous allons pouvoir démontrer la propriété de complétude. Pour cela, nous allons poser pour $n \geq 1$ la propriété suivante :

Définition A.2.2. $\mathcal{H}'(n)$ *: pour toute formule* φ

$$\models_{MF} \psi \rightarrow B_{db_1 * \ldots * db_n} \varphi \Longrightarrow \vdash_{MF} \psi \rightarrow B_{db_1 * \ldots * db_n} \varphi$$

$$\models_{MF} \psi \rightarrow \neg B_{db_1 * \ldots * db_n} \varphi \Longrightarrow \vdash_{MF} \psi \rightarrow \neg B_{db_1 * \ldots * db_n} \varphi$$

où $\forall i \in \{1 \ldots n\}$ db_i *est une base de croyances primitive.*

La preuve par induction de $\mathcal{H}(n)$ pour tout $n \geq 1$ prouve la propriété de complétude.
Preuve de $\mathcal{H}'(1)$
Soit db une base de croyances primitive cohérente.

1. si $\models_{MF} \psi \rightarrow B_{db}\varphi$ alors $\models_{MF} \psi \rightarrow B_{db}\varphi \Rightarrow \forall M \forall w \;\; M, w \models_{MF} \psi \rightarrow B_{db}\varphi$, donc $\forall M \forall w \;\; M, w \models_{MF} \psi \;\Rightarrow\; M, w \models_{MF} B_{db}\varphi$.
   Soit M un MF-modèle qui satisfait ψ. D'après le lemme A.2.1,
   $\forall w \in W \;\; f_{db}(w) = \{val(l) \;:\; l \in DB\}$. Soit w un monde W. db est cohérente, donc d'après le lemme A.2.2, $val(l) \in f_{db}(w) \Rightarrow val(\neg l) \notin f_{db}(w)$. Donc, d'après le lemme A.2.3, $\min_{\leq_{f_{db}(w)}} W = val(\bigwedge_{val(l) \in f_{db}(w)} l)$. Mais $g_{db}(w) = \min_{\leq_{f_{db}(w)}} W$, donc $g_{db}(w) = val(\bigwedge_{val(l) \in f_{db}(w)} l)$.
   De plus, $M, w \models_{MF} B_{db}\varphi \;\Rightarrow\; g_{db}(w) \subset val(\varphi)$, donc $val(\bigwedge_{val(l) \in f_{db}(w)} l) \subset val(\varphi)$ et $\models_{MF} \bigwedge_{l \in DB} l \rightarrow \varphi$.
   (a) si φ est un littéral l', alors $\models_{MF} \bigwedge_{l \in DB} l \rightarrow l'$, donc $l' \in DB$.

$$\begin{aligned} l' \in DB &\Rightarrow \neg l' \notin DB \text{ car } db \text{ est cohérente} \\ &\Rightarrow \vdash_{MF} \psi \rightarrow B^1_{db} l' \text{ et} \\ &\quad \vdash_{MF} \psi \rightarrow B^0_{db} \neg l' \\ \text{schéma d'axiome } \mathbf{(A5)} &\Rightarrow \psi \rightarrow B_{db} l' \end{aligned}$$

   (b) si $\varphi \equiv l_1 \vee \ldots \vee l_m$ alors si $\varphi \equiv \top$, alors $\vdash_{MF} \varphi$, donc $\vdash_{MF} B_{db}\varphi$, et $\vdash_{MF} \psi \rightarrow B_{db}\varphi$.
   Si $\varphi \not\equiv \top$, alors $\models_{MF} \bigwedge_{l \in DB} l \rightarrow l_1 \vee \ldots \vee l_m$. Donc il existe $k \in \{1 \ldots m\}$ tel que $\models_{MF} \bigwedge_{l \in DB} l \rightarrow l_k$. Ce cas est le même que le cas 1.a.
   (c) si φ est une conjonction de disjonctions, i.e. $\varphi \equiv \bigwedge_{k \in \{1 \ldots n\}} (l_{k_1} \vee \ldots \vee l_{k_m})$, alors $\models_{MF} \bigwedge_{l \in DB} l \rightarrow \bigwedge_{k \in \{1 \ldots n\}} (l_{k_1} \vee \ldots \vee l_{k_m})$ donc $\forall k \in \{1 \ldots n\}$ $\models_{MF} \bigwedge_{l \in DB} l \rightarrow l_{k_1} \vee \ldots \vee l_{k_m}$. Ce cas est le même que le cas 1.b, à cause du comportement KD de l'opérateur B_{db}.

2. si $\models_{MF} \psi \rightarrow \neg B_{db}\varphi$, on prouve $g_{db}(w) = val(\bigwedge_{l \in DB} l)$ pour tout MF-modèle M de ψ et tout monde w de W par la même méthode.

$$\begin{aligned} M, w \models_{MF} \neg B_{db}\varphi \;\models_{MF}\Leftrightarrow\; & g_{db}(w) \not\subseteq val(\varphi) \\ \Leftrightarrow\quad & \not\models (\bigwedge_{l \in DB} l) \rightarrow \varphi \end{aligned}$$

   (a) si φ est un littéral l', alors $\not\models_{MF} (\bigwedge_{l \in DB} l) \rightarrow l'$. Dans ce cas, $l' \notin db$ (car db est cohérente).

$$l' \notin DB \;\Rightarrow\; \vdash_{MF} \psi \rightarrow B^0_{db} l'$$

   Il y a deux cas :

– soit $\vdash_{MF} \psi \rightarrow B^0_{db}\neg l'$ et alors d'après le schéma d'axiome **(A7)**, $\vdash_{MF} \psi \rightarrow \neg B_{db} l'$;
– soit $\vdash_{MF} \psi \rightarrow B^i_{db}\neg l'$ avec $i > 0$ et

$$\begin{array}{rcl} \text{schémas d'axiomes } \mathbf{(A3)} \text{ et } \mathbf{(A6)} & \Rightarrow & \vdash_{MF} \psi \rightarrow B_{db}\neg l' \\ \text{schéma d'axiome } \mathbf{(A1)} & \Rightarrow & \vdash_{MF} \psi \rightarrow \neg B_{db} l' \end{array}$$

(b) si $\varphi \equiv l_1 \vee \ldots \vee l_m$ alors :

$$\begin{array}{rcl} \not\models (\bigwedge_{l \in DB} l) \rightarrow \varphi & \Rightarrow & \forall i \in \{1 \ldots m\}\ l_i \notin DB \\ & \Rightarrow & \vdash_{MF} \psi \rightarrow B^0_{db} l_1 \wedge \ldots \wedge B^0_{db} l_m \\ \text{même preuve que le cas 2.a} & \Rightarrow & \vdash_{MF} \psi \rightarrow \neg B_{db} l_1 \wedge \ldots \wedge \neg B_{db} l_m \\ & \Rightarrow & \vdash_{MF} \psi \rightarrow \neg(B_{db} l_1 \vee \ldots \vee B_{db} l_m) \\ \text{schéma d'axiome } \mathbf{(A8)} & \Rightarrow & \vdash_{MF} \psi \rightarrow \neg B_{db}(l_1 \vee \ldots \vee l_m) \end{array}$$

(c) si φ est une conjonction de clauses, i.e. $\varphi \equiv \bigwedge_{k \in \{1 \ldots n\}} (l_{k_1} \vee \ldots \vee l_{k_m})$, alors :

$$\begin{array}{rcl} \not\models (\bigwedge_{l \in DB} l) \rightarrow \varphi & \Rightarrow & \exists i \in \{1 \ldots n\}\ \not\models_{MF} (\bigwedge_{l \in DB} l) \rightarrow (l_{k_1} \vee \ldots \vee l_{k_m}) \\ \text{par la preuve précédente} & \Rightarrow & \exists i \in \{1 \ldots n\}\ \vdash_{MF} \psi \rightarrow \neg B_{db}(l_{k_1} \vee \ldots \vee l_{k_m}) \\ & \Rightarrow & \vdash_{MF} \psi \rightarrow \bigvee_{k \in \{1 \ldots n\}} \neg B_{db}(l_{k_1} \vee \ldots \vee l_{k_m}) \\ & \Rightarrow & \vdash_{MF} \psi \rightarrow \neg(\bigwedge_{k \in \{1 \ldots n\}} B_{db}(l_{k_1} \vee \ldots \vee l_{k_m})) \\ \text{schémas d'axiome } \mathbf{(A1)} \text{ et } \mathbf{(A2)} & \Rightarrow & \vdash_{MF} \psi \rightarrow \neg B_{db}(\bigwedge_{k \in \{1 \ldots n\}} (l_{k_1} \vee \ldots \vee l_{k_m})) \end{array}$$

Nous avons prouvé que pour toute base de croyances primitive cohérente db et pour toute formule propositionnelle φ :

$$\begin{array}{rcl} \models_{MF} \psi \rightarrow B_{db}\varphi & \Rightarrow & \vdash_{MF} \psi \rightarrow B_{db}\varphi \\ \models_{MF} \psi \rightarrow \neg B_{db}\varphi & \Rightarrow & \vdash_{MF} \psi \rightarrow \neg B_{db}\varphi \end{array}$$

$\mathcal{H}'(1)$ est vraie.
Preuve de $\mathcal{H}'(n)$ pour $n > 1$
Soit db une base de croyances obtenue en fusionnant n bases de croyances ($n \geq 1$) et db_{n+1} une base de croyances primitive.

1. supposons que $\models_{MF} \psi \rightarrow B_{db * db_{n+1}}\varphi$.

   (a) si φ est un littéral l de $PROP$ alors pour tout MF-modèle M de ψ et pour tout monde w, $g_{db * db_{n+1}}(w) \subset val(l)$.
   Soient i et j les deux entiers tels que $val(l) \in^i f_{db * db_{n+1}}(w)$ et $val(\neg l) \in^j f_{db * db_{n+1}}(w)$. Supposons que $i \leq j$. Soient w_1 et w_2 deux mondes de W tels que $w_1 \models_{MF} \neg l$ et $w_2 \models_{MF} l$.
   Dans ce cas :

$$f_{db*db_{n+1}}(w) = [\underbrace{val(l)\dots val(l)}_{i\ fois}\ \underbrace{val(\neg l)\dots val(\neg l)}_{j\ fois}\ \underbrace{val(l_1)\dots val(l_m)}_{\forall k\in\{1\dots m\}\ l_k\neq l \text{ et } l_k\neq\neg l}\ ]$$

Donc :

$$\begin{aligned} dsum(w_1, f_{db*db_{n+1}}(w)) &= 0*j + 1*i + \sum_{k\in\{1\dots m\}} \min_{w'\in f_{db*db_{n+1}}(w)} d(w_1, w') \\ &= i \\ dsum(w_2, f_{db*db_{n+1}}(w)) &= 1*j + 0*i + \sum_{k\in\{1\dots m\}} \min_{w'\in f_{db*db_{n+1}}(w)} d(w_2, w') \\ &= j \end{aligned}$$

Supposons que $i \leq j$. Il y a deux cas différents :
– $i < j$, alors $g_{db*db_{n+1}}(w) \subset val(\neg l)$, donc $g_{db*db_{n+1}}(w) \not\subset val(l)$. Mais $g_{db*db_{n+1}}(w) \subset val(l)$ donc l'hypothèse $i < j$ est fausse ;
– $i = j$, et alors $g_{db*db_{n+1}}(w) \not\subset val(l)$ donc l'hypothèse est fausse.
Donc $\exists i\ \exists j\ i > j \geq 0$ tels que $val(l) \in^i f_{db*db_{n+1}}(w)$ et $val(\neg l) \in^j f_{db*db_{n+1}}(w)$.
D'après la construction de $f_{db*db_{n+1}}(w)$:
– $\exists i_1\ \exists i_2$ tels que $i_1 + i_2 = i$, $val(l) \in^{i_1} f_{db}$ et $val(l) \in^{i_2} f_{db_{n+1}}$;
– $\exists j_1\ \exists j_2$ tels que $j_1 + j_2 = i$, $val(l) \in^{j_1} f_{db}$ et $val(l) \in^{j_2} f_{db_{n+1}}$;
Donc :

$$\begin{aligned} & \left\{ \begin{array}{l} \models_{MF} \psi \rightarrow B^{i_1}_{db} l \quad \text{et} \quad \models_{MF} \psi \rightarrow B^{i_2}_{db_{n+1}} l \\ \models_{MF} \psi \rightarrow B^{j_1}_{db} \neg l \quad \text{et} \quad \models_{MF} \psi \rightarrow B^{j_2}_{db_{n+1}} \neg l \end{array} \right. \\ \text{lemma A.2.4} \quad \Rightarrow \quad & \left\{ \begin{array}{l} \vdash_{MF} \psi \rightarrow B^{i_1}_{db} l \quad \text{et} \quad \vdash_{MF} \psi \rightarrow B^{i_2}_{db_{n+1}} l \\ \vdash_{MF} \psi \rightarrow B^{j_1}_{db} \neg l \quad \text{et} \quad \vdash_{MF} \psi \rightarrow B^{j_2}_{db_{n+1}} \neg l \end{array} \right. \\ \text{schéma d'axiome } \mathbf{(A5)} \quad \Rightarrow \quad & \left\{ \begin{array}{l} \vdash_{MF} \psi \rightarrow B^{i}_{db*db_{n+1}} l \\ \vdash_{MF} \psi \rightarrow B^{j}_{db*db_{n+1}} \neg l \end{array} \right. \\ \text{schéma d'axiome } \mathbf{(A6)} \quad \Rightarrow \quad & \{ \vdash_{MF} \psi \rightarrow B_{db*db_{n+1}} l \text{ (car } i > j) \end{aligned}$$

(b) si φ est une disjunction de littéraux, i.e. $\varphi \equiv l_1 \vee \dots \vee l_m$ alors si $\varphi \equiv \top$, $\vdash_{MF} \varphi$, donc $\vdash_{MF} B_{db*db_{n+1}}\varphi$ (d'après (Nec)), et alors $\vdash_{MF} \psi \rightarrow B_{db*db_{n+1}}\varphi$.
Si $\varphi \not\equiv \top$, alors pour tout $w \in W$ $f_{db*db_{n+1}}(w)$ est tel que $val(l) \in f_{db*db_{n+1}}(w) \Longrightarrow val(\neg l) \notin f_{db*db_{n+1}}(w)$. Donc, d'après le lemme A.2.3, $g_{db*db_{n+1}}(w) =$ $\min_{\leq_{f_{db*db_{n+1}}(w)}}(W)$ est une conjonction de littéraux. Mais $g_{db*dbn+1}(w) \subseteq val(\varphi)$, donc $g_{db*dbn+1}(w) \subseteq val(l_1 \vee \dots \vee l_m)$. Dans ce cas, $\exists k \in \{1\dots m\}$ tel que $M \models_{MF} B_{db*db_{n+1}} l_k$. C'est le même cas que précédemment et alors $\vdash_{MF} \psi \rightarrow B_{db*db_{n+1}} l_k$. D'après le schéma d'axiome **(A2)**, $\vdash_{MF} \psi \rightarrow B_{db*db_{n+1}} l_1 \vee \dots \vee B_{db*db_{n+1}} l_m$.

(c) si φ est une conjonction de clauses, i.e. $\varphi \equiv \bigwedge_{k\in\{1\ldots n\}} (l_{k_1} \vee \ldots \vee l_{k_m})$, alors
$\models_{MF} \psi \rightarrow B_{db*db_{n+1}} \bigwedge_{k\in\{1\ldots n\}} (l_{k_1} \vee \ldots \vee l_{k_m})$ so $\forall k \in \{1 \ldots n\}$
$\models_{MF} \psi \rightarrow B_{db*db_{n+1}}(l_{k_1} \vee \ldots \vee l_{k_m})$ et c'est le même cas que précédemment parce que $B_{db*db_{n+1}}$ est un opérateur de type KD.

2. supposons que $\models_{MF} \psi \rightarrow \neg B_{db*db_{n+1}}\varphi$.

(a) si φ est un littéral l de $PROP$.
Pour tout MF-modèle M de ψ et pour tout monde w, $g_{db*db_{n+1}}(w) \not\subset val(l)$.
Donc $\exists i\ \exists j\ j \geq i \geq 0$ tels que $val(l) \in^i f_{db*db_{n+1}}(w)$ et $val(\neg l) \in^j f_{db*db_{n+1}}(w)$.
Par construction de $f_{db*db_{n+1}}(w)$, on peut remarquer que :
– $\exists i_1\ \exists i_2$ tels que $i_1 + i_2 = i$, $val(l) \in^{i_1} f_{db}$ et $val(l) \in^{i_2} f_{db_{n+1}}$;
– $\exists j_1\ \exists j_2$ tels que $j_1 + j_2 = i$, $val(l) \in^{j_1} f_{db}$ et $val(l) \in^{j_2} f_{db_{n+1}}$;
Donc :

$$\left\{\begin{array}{l} \models_{MF} \psi \rightarrow B^{i_1}_{db} l \quad \text{et} \quad \models_{MF} \psi \rightarrow B^{i_2}_{db_{n+1}} l \\ \models_{MF} \psi \rightarrow B^{j_1}_{db} \neg l \quad \text{et} \quad \models_{MF} \psi \rightarrow B^{j_2}_{db_{n+1}} \neg l \end{array}\right.$$

$$\text{d'après le lemme A.2.4} \Rightarrow \left\{\begin{array}{l} \vdash_{MF} \psi \rightarrow B^{i_1}_{db} l \quad \text{et} \quad \vdash_{MF} \psi \rightarrow B^{i_2}_{db_{n+1}} l \\ \vdash_{MF} \psi \rightarrow B^{j_1}_{db} \neg l \quad \text{et} \quad \vdash_{MF} \psi \rightarrow B^{j_2}_{db_{n+1}} \neg l \end{array}\right.$$

$$\mathbf{A5} \Rightarrow \left\{\begin{array}{l} \vdash_{MF} \psi \rightarrow B^{i}_{db*db_{n+1}} l \\ \vdash_{MF} \psi \rightarrow B^{j}_{db*db_{n+1}} \neg l \end{array}\right.$$

car $i = i_1 + i_2$ et $j = j_1 + j_2$
Il y a deux cas :
– soit $i = j$ et donc d'après l'axiome **(A7)**, $\vdash_{MF} \psi \rightarrow \neg B_{db*db_{n+1}} l$;
– soit $j > i$ et donc d'après les axiomes **(A3)** et **(A6)**, $\vdash_{MF} \psi \rightarrow B_{db*db_{n+1}} \neg l$.
D'après l'axiome **(A1)** $\vdash_{MF} \psi \rightarrow \neg B_{db*db_{n+1}} l$.
La propriété est vraie quand φ est un littéral.

(b) si φ est une clause, i.e. $\varphi \equiv l_1 \vee \ldots \vee l_m$, alors pour tout MF-modèle M de ψ, $M \models_{MF} \neg B_{db*db_{n+1}}(l_1 \vee \ldots \vee l_m)$.
Comme les schémas d'axiomes (A0), (A1) et (A2) sont valides,
$\models_{MF} B_{db*db_{n+1}}(l_1) \vee \ldots \vee B_{db*db_{n+1}}(l_m) \rightarrow B_{db*db_{n+1}}(l_1 \vee \ldots \vee l_m)$. Donc
$\models_{MF} \neg B_{db*db_{n+1}}(l_1 \vee \ldots \vee l_m) \rightarrow B_{db*db_{n+1}}(l_1) \wedge \ldots \wedge B_{db*db_{n+1}}(l_m)$.
Donc $\forall k \in \{1 \ldots m\}$ $M \models_{MF} \neg B_{db*db_{n+1}} l_k$. Par la même preuve que précédemment, comme M est un modèle de ψ, $\forall k \in \{1 \ldots m\}$
$\vdash_{MF} \psi \rightarrow \neg B_{db*db_{n+1}} l_k$. D'après l'axiome **(A8)**,
$\vdash_{MF} \psi \rightarrow \neg B_{db*db_{n+1}}(l_1 \vee \ldots \vee l_m)$.

(c) si φ est une conjonction de clauses, i.e. $\varphi \equiv \bigwedge_{k\in\{1\ldots n\}} (l_{k_1} \vee \ldots \vee l_{k_m})$,
$\models_{MF} \psi \rightarrow \neg B_{db*db_{n+1}} \bigwedge_{k\in\{1\ldots n\}} (l_{k_1} \vee \ldots \vee l_{k_m})$.
Soit M un MF-modèle de ψ, alors $M \models_{MF} \neg B_{db*db_{n+1}} \bigwedge_{k\in\{1\ldots n\}} (l_{k_1} \vee \ldots \vee l_{k_m})$,
donc $\exists k \in \{1 \ldots m\}$ $M \models_{MF} \neg B_{db*db_{n+1}}(l_{k_1} \vee \ldots \vee l_{k_m})$.

Donc :

$$\begin{array}{rcl} & \exists k \in \{1 \dots m\} \ \models_{MF} \psi \rightarrow \neg B_{db*db_{n+1}}(l_{k_1} \vee \dots \vee l_{k_m}) \\ \Rightarrow & \vdash_{MF} \psi \rightarrow \neg B_{db*db_{n+1}}(l_{k_1} \vee \dots \vee l_{k_m}) \\ \Rightarrow & \vdash_{MF} \psi \rightarrow \bigvee\limits_{k \in \{1 \dots m\}} \neg B_{db*db_{n+1}}(l_{k_1} \vee \dots \vee l_{k_m}) \\ \Rightarrow & \vdash_{MF} \psi \rightarrow \neg(\bigwedge\limits_{k \in \{1 \dots m\}} B_{db*db_{n+1}}(l_{k_1} \vee \dots \vee l_{k_m})) \\ \textbf{(A1) et (A2)} & \Rightarrow \vdash_{MF} \psi \rightarrow \neg B_{db*db_{n+1}}(\bigwedge\limits_{k \in \{1 \dots m\}} (l_{k_1} \vee \dots \vee l_{k_m})) \end{array}$$

La propriété $\mathcal{H}'(n)$ est vraie pour tout $n \geq 1$.
□.

Théorème 8.1.2. *Soit ψ la formule définie en définition 8.1.9. Soit φ une formule de PROP et db une base de croyances (primitive ou pas). On a alors :*

$$\nvdash_{MF} \psi \rightarrow B_{db}\varphi \iff \vdash_{MF} \psi \rightarrow \neg B_{db}\varphi$$

Preuve. *Pour prouver $\Longrightarrow$), nous prouvons tout d'abord les lemmes suivants :*

Lemme A.2.5. *Soient $M_1 = \langle W_1, val_1, R_1, B_1 \rangle$ et $M_2 = \langle W_2, val_2, R_2, B_2 \rangle$ deux MF-modèles. $\forall w_1 \in W_1 \ \exists w_2 \in W_2$ tel que $\{l \ : \ w_1 \in val(l)\} = \{l \ : \ w_2 \in val(l)\}$.*

Ce lemme montre que tout monde de M_1 a un « équivalent » dans M_2.

Preuve. *Soit $w_1 \in W_1$. Considérons la formule $\varphi = \bigwedge\limits_{l \ : \ w_1 \in val_1(l)} l$.*
φ est satisfaisable puisque $w_1 \in val_1(\varphi)$, donc $\exists w_2 \in W_2$ tel que $w_2 \in val_2(\varphi)$ par définition d'une valuation.
Il existe donc $w_2 \in W_2$ tel que $w_2 \in val(l)$ pour tout l tel que $w_1 \in val_1(l)$, donc $\exists w_2 \in W_2 \ \{l \ : \ w_1 \in val(l)\} = \{l \ : \ w_2 \in val(l)\}$.
□ .

Lemme A.2.6. *Soient $M_1 = \langle W_1, val_1, R_1, B_1 \rangle$ et $M_2 = \langle W_2, val_2, R_2, B_2 \rangle$ deux MF-modèles. $\forall w_1 \in W_1 \ \exists w_2 \in W_2$ tel que $\forall \varphi \ \ w_1 \in val_1(\varphi) \Longleftrightarrow w_2 \in val_2(\varphi)$.*

Preuve. *Soit $w_1 \in W_1$ tel que $w_1 \in val(\varphi)$. Pour démontrer le lemme A.2.6 nous allons distinguer les cas où φ est un littéral, la négation d'une formule et la conjonction de deux formules.*
- *si φ est un littéral l_0, alors $w_1 \in val(l_0)$. D'après le lemme A.2.5, $\exists w_2 \in W_2$ tel que $\{l \ : \ w_1 \in val_1(l)\} = \{l \ : \ w_2 \in val_2(l)\}$, donc $\exists w_2 \in W_2$ tel que $w_2 \in val_2(l_0)$;*
- *si $\varphi \equiv \neg\varphi'$, alors :*

$$\begin{array}{rcl} w_1 \in val_1(\neg\varphi') & \Leftrightarrow & w_1 \notin val_1(\varphi') \\ & \Leftrightarrow & \exists w_2 \in W_2 \ \ w_2 \notin val_2(\varphi') \text{ d'après le lemme A.2.5} \\ & \Leftrightarrow & \exists w_2 \in W_2 \ \ w_2 \in val_2(\neg\varphi') \end{array}$$

– *si* $\varphi \equiv \gamma \wedge \gamma'$, *alors :*

$$\begin{aligned} w_1 \in val_1(\gamma \wedge \gamma') &\Leftrightarrow w_1 \in val_1(\gamma) \text{ et } w_1 \in val_1(\gamma') \\ &\Leftrightarrow \exists w_2 \in W_2 \;\; w_2 \in val_2(\gamma) \text{ et } w_2 \in val_2(\gamma') \text{ d'après le lemme A.2.5} \\ &\Leftrightarrow \exists w_2 \in W_2 \;\; w_2 \in val_2(\gamma \wedge \gamma') \end{aligned}$$

□ .

Lemme A.2.7. *Soient* M_1 *et* M_2 *deux* MF*-modèles qui satisfont* ψ*, alors* $M_1 \models B_{db}\varphi \Longleftrightarrow M_2 \models B_{db}\varphi$.

Preuve. *Soient* M_1 *et* M_2 *deux* MF*-modèles qui satisfont* ψ*. Nous distinguons trois cas :*
– *soit* φ *est un littéral et alors on doit montrer :*

$$\forall w \in \min_{\leq_{f_{db}}} W_1 \;\; w \in val_1(l) \Leftrightarrow \forall w \in \min_{\leq_{f_{db}}} W_1 \;\; w \in val_2(l)$$

soit :

$$\forall w \in \min_{\leq_{[\ldots val_1(l_i)\ldots]}} W_1 \;\; w \in val_1(l) \Leftrightarrow \forall w \in \min_{\leq_{[\ldots val_1(l_i)\ldots]}} W_1 \;\; w \in val_2(l)$$

Soient w *et* w' *deux mondes de* W_1*. On a :*

$$\begin{aligned} w \leq_{[\ldots val_1(l_i)\ldots]} w' &\Leftrightarrow dsum(w, [\ldots val_1(l_i)\ldots]) \leq dsum(w', [\ldots val_1(l_i)\ldots]) \\ &\Leftrightarrow \text{nombre de } l_i \text{ tq } w \notin val_1(l_i) \leq \text{ nombre de } l_i \text{ tq } w' \notin val_1(l_i) \end{aligned}$$

Soient w_t *et* w'_t *les mondes de* W_2 *associés respectivement à* w *et* w' *d'après le lemme A.2.5. On a alors :*

$$\begin{aligned} w \leq_{[\ldots val_1(l_i)\ldots]} w' &\Leftrightarrow \text{nombre de } l_i \text{ tq } w_t \notin val_2(l_i) \leq \text{ nombre de } l_i \text{ tq } w'_t \notin val_2(l_i) \\ &\Leftrightarrow dsum(w_t, [\ldots val_2(l_i)\ldots]) \leq dsum(w'_t, [\ldots val_2(l_i)\ldots]) \end{aligned}$$

Donc $w \in \min_{\leq_{[\ldots val_1(l_i)\ldots]}} \Leftrightarrow w_t \in \min_{\leq_{[\ldots val_2(l_i)\ldots]}}$*. Donc si* $\forall w \in \min_{\leq_{[\ldots val_1(l_i)\ldots]}} W_1 \;\; w \in val_1(l)$*, comme* w_t *est « équivalent » à* w *(cf. lemme A.2.6), on a* $\forall w_t \in \min_{\leq_{[\ldots val_1(l_i)\ldots]}} W_1 \; w_t \in val_2(l)$.
Donc $M_1 \models B_{db}l \Leftrightarrow M_2 \models B_{db}l$.
– *soit* φ *est une clause* $l_1 \vee \ldots \vee l_n$ *qui n'est pas une tautologie (sinon la démonstration est évidente) et alors :*

$$\begin{aligned} M_1 \models B_{db}(l_1 \vee \ldots \vee l_n) &\Leftrightarrow M_1 \models B_{db}l_1 \vee \ldots \vee B_{db}l_n \\ &\Leftrightarrow M_1 \models B_{db}l_1 \text{ ou } \ldots \text{ ou } M_1 \models B_{db}l_n \\ &\Leftrightarrow M_2 \models B_{db}l_1 \text{ ou } \ldots \text{ ou } M_2 \models B_{db}l_n \text{ d'après ce qui précéde} \\ &\Leftrightarrow M_2 \models B_{db}l_1 \vee \ldots \vee B_{db}l_n \end{aligned}$$

– *soit* φ *est une conjonction de clauses* $c_1 \wedge \ldots \wedge c_n$ *et alors :*

$$\begin{aligned} M_1 \models B_{db}(c_1 \wedge \ldots \wedge c_n) &\Leftrightarrow M_1 \models B_{db}c_1 \wedge \ldots \wedge B_{db}c_n \\ &\Leftrightarrow M_1 \models B_{db}c_1 \text{ et } \ldots \text{ et } M_1 \models B_{db}c_n \\ &\Leftrightarrow M_2 \models B_{db}c_1 \text{ et } \ldots \text{ et } M_2 \models B_{db}c_n \text{ d'après ce qui précéde} \\ &\Leftrightarrow M_2 \models B_{db}c_1 \wedge \ldots \wedge B_{db}c_n \end{aligned}$$

□ .

Lemme A.2.8. *Soit M un MF-modèle qui satisfait ψ. Alors* $\forall w \in W \forall w' \in W \ g_{db}(w) = g_{db}(w')$.

Preuve. *Soit M un MF-modèle satisfaisant ψ. On a :*

$$\begin{array}{rcl} M \models \psi & \Rightarrow & \forall db \text{ base primitive } \forall w \forall w' \ f_{db}(w) = f_{db}(w') = [\ldots val(l) \ldots]_{l \in DB} \\ & \Rightarrow & db \text{ base quelconque } \forall w \forall w' \ f_{db}(w) = f_{db}(w') \\ & \Rightarrow & db \text{ base quelconque } \forall w \forall w' \ g_{db}(w) = g_{db}(w') \end{array}$$

□ .

On peut prouver maintenant le sens $\Longrightarrow$). Supposons que $\not\vdash_{MF} \psi \rightarrow B_{db}\varphi$. D'après le théorème 8.1.1, $\not\models_{MF} \psi \rightarrow B_{db}l$. Donc $\exists M_1$ MF-modèle de ψ tel que $M_1 \not\models B_{db}l$. D'après le lemme A.2.7, $\forall M$ MF-modèle de ψ, $M \not\models B_{db}l$. Nous prouvons le sens $\Longrightarrow$) en distinguant trois cas :
– *soit φ est un littéral l et :*

$$\begin{array}{rcl} \forall M \ M \models \psi & \Longrightarrow & \exists w \in W \ \ M, w \not\models B_{db}l \\ \forall M \ M \models \psi & \Longrightarrow & \exists w \in W \ \ \exists w' \in g_{db}(w) \ \ w' \notin val(l) \\ \forall M \ M \models \psi & \Longrightarrow & \forall w \in W \ \ \exists w' \in g_{db}(w) \ \ w' \notin val(l) \text{ d'après le lemme A.2.8} \\ \forall M \ M \models \psi & \Longrightarrow & \forall w \in W \ \ M, w \not\models B_{db}l \\ \forall M \ M \models \psi & \Longrightarrow & M \models \neg B_{db}l \\ & \Longrightarrow & \models \psi \rightarrow \neg B_{db}l \end{array}$$

– *soit φ est une clause $l_1 \vee \ldots \vee l_n$ et :*

$$\begin{array}{rcl} \forall M \ M \models \psi & \Longrightarrow & \exists w \in W \ \ M, w \not\models B_{db}(l_1 \vee \ldots \vee l_n) \\ \forall M \ M \models \psi & \Longrightarrow & \exists w \in W \ \ \exists w' \in g_{db}(w) \ \ w' \not\models l_1 \vee \ldots \vee l_n \\ \forall M \ M \models \psi & \Longrightarrow & \exists w \in W \ \ \exists w' \in g_{db}(w) \ \ \forall i \in \{1, \ldots, n\} \ \ w' \not\models l_i \\ \forall M \ M \models \psi & \Longrightarrow & \exists w \in W \ \ \exists w' \in g_{db}(w) \ \ \forall i \in \{1, \ldots, n\} \ \ w' \notin val(l_i) \\ \forall M \ M \models \psi & \Longrightarrow & \forall w \in W \ \ \forall i \in \{1, \ldots, n\} \ \ w \not\models B_{db}(l_i) \text{ d'après le lemme A.2.8} \\ \forall M \ M \models \psi & \Longrightarrow & \forall w \in W \ \ M, w \not\models B_{db}(l_1) \vee \ldots \vee B_{db}(l_n) \\ \forall M \ M \models \psi & \Longrightarrow & \forall w \in W \ \ M, w \models \neg(B_{db}(l_1) \vee \ldots \vee B_{db}(l_n)) \\ \forall M \ M \models \psi & \Longrightarrow & \forall w \in W \ \ M, w \models \neg B_{db}(l_1 \vee \ldots \vee l_n) \text{ car le schéma d'axiome} \\ & & \text{(A8) est valide} \\ & \Longrightarrow & \models \psi \rightarrow \neg B_{db}(l_1 \vee \ldots \vee l_n) \end{array}$$

– *soit φ est une conjonction de clauses $C_1 \wedge \ldots \wedge C_m$ et :*

$$\begin{array}{rcl} \forall M \ M \models \psi & \Longrightarrow & \exists w \in W \ \ M, w \not\models B_{db}(C_1 \wedge \ldots \wedge C_m) \\ \forall M \ M \models \psi & \Longrightarrow & \exists w \in W \ \ \exists w' \in g_{db}(w) \ \ w' \not\models C_1 \wedge \ldots \wedge C_m \end{array}$$

$$\begin{array}{rcl}
\forall M\ M \models \psi & \Longrightarrow & \exists w \in W\ \ \exists w' \in g_{db}(w)\ \ \exists i \in \{1,\ldots,m\}\ \ w' \not\models C_i \\
\forall M\ M \models \psi & \Longrightarrow & \forall w \in W\ \ \exists i \in \{1,\ldots,n\}\ \ w \not\models B_{db}(C_i)\ \textit{d'après le lemme A.2.8} \\
\forall M\ M \models \psi & \Longrightarrow & \forall w \in W\ \ w \not\models B_{db}(C_1) \wedge \ldots \wedge B_{db}(C_m) \\
\forall M\ M \models \psi & \Longrightarrow & M \models \neg(B_{db}(C_1) \wedge \ldots \wedge B_{db}(C_m)) \\
\forall M\ M \models \psi & \Longrightarrow & M \models \neg B_{db}(C_1 \wedge \ldots \wedge C_m)\ \textit{car les schémas d'axiomes} \\
 & & \textit{(A0), (A1) et (A2) sont valides} \\
 & \Longrightarrow & \models \psi \rightarrow \neg B_{db}(C_1 \wedge \ldots \wedge C_m)
\end{array}$$

Pour prouver $\Longleftarrow$), supposons que $\vdash_{MF} \psi \rightarrow B_{db}\varphi$ et $\vdash_{MF} \psi \rightarrow \neg B_{db}\varphi$. Dans ce cas, d'après le théorème 8.1.1, $\models_{MF} \psi \rightarrow B_{db}\varphi$ et $\models_{MF} \psi \rightarrow \neg B_{db}\varphi$. Donc pour tout MF-modèle M de ψ, $M \models_{MF} B_{db}\varphi$ et $M \models_{MF} \neg B_{db}\varphi$. C'est impossible (car ψ a au moins un modèle), donc $\vdash_{MF} \psi \rightarrow \neg B_{db}\varphi \implies \not\vdash_{MF} \psi \rightarrow B_{db}\varphi$.
□.

A.2.3 Relation avec les travaux de Konieczny et Pino-Pérez

Théorème 8.2.1. *Soient $db_1, \ldots, db_n$ n ensembles de littéraux finis et cohérents à fusionner et φ une formule de $PROP$. Avec les notations introduites précédemment, on a :*

$$\vdash \psi \rightarrow B_{(\ldots(db_1 * db_2) * \ldots db_n)}\varphi \Longleftrightarrow \Delta_\Sigma([db_1, \ldots, db_n]) \models \varphi$$

Preuve. *Remarquons tout d'abord que si $M = \langle W, val, R, B\rangle$ est un MF-modèle, alors pour tout monde w de W, il existe une interprétation w' de $PROP$ telle que $\{l : w' \models l\} = \{l : w \in val(l)\}$ (cf. lemmes A.2.5 et A.2.6). Et si w' est une interprétation de $PROP$, alors il existe un monde w de W tel que $\{l : w' \models l\} = \{l : w \in val(l)\}$. En d'autres termes, chaque monde de W correspond à une interprétation de $PROP$ et toute interprétation de $PROP$ correspond à au moins un monde de W.*
Supposons que $\forall i \in \{1,\ldots,n\}\ \ db_i = \{l_i^1, \ldots, l_i^{m_i}\}$.
Soit $M = \langle W, val, R, B\rangle$ est un MF-modèle qui satisfait ψ. Soit w un monde de W et w' l'interprétation de $PROP$ caractérisée précédemment. On prouve que :

$$d_{sum}(w, [...val(l_1^i)...val(l_n^j)...]) = dist_\Sigma(w', [db_1, \ldots, db_n])$$

Cela montre que les mondes de W qui sont minimaux par rapport au préordre induit par la distance d_{sum} correspondent aux interprétations de $PROP$ qui sont minimales par rapport au préordre induit par la distance d_Σ.
□.

A.2.4 Démonstrateur automatique

Propriété 8.3.1. *Soit l un littéral, soit db une base de croyances primitive ou pas. Dans ce cas, en utilisant la négation par échec sur le méta-programme META,*
(1) PROLOG réussit à prouver $B(db, l)$ ssi $\models_{MF} (\psi \rightarrow B_{db} l)$
(2) PROLOG échoue à prouver $B(db, l)$ ssi $\models_{MF} (\psi \rightarrow \neg B_{db}\ l)$
Pour prouver (1), nous prouvons tout d'abord le lemme suivant :

Lemme A.2.9. *PROLOG réussit à prouver* $R(db_1 * (db_2 * \ldots * nil)), l, i)$ *ssi* $\models_{MF} \psi \rightarrow B^i_{db} l$.

Preuve. *Preuve du sens* $\Longrightarrow$*)*
Nous allons prouver le sens $\Longrightarrow$*) par induction. On pose pour tout* $n > 0$ *:*

Définition A.2.3. $\mathcal{H}(n)$ *: PROLOG réussit à prouver* $R(db_1 * (db_2 * \ldots * nil)), l, i)$ *en n pas* $\Rightarrow \models_{MF} \psi \rightarrow B^i_{db_1 * \ldots * db_n} l$

Preuve de $\mathcal{H}(1)$
Supposons que PROLOG prouve $R(db_1 * (db_2 * \ldots * nil)), l, i)$ *en 1 pas. Dans ce cas, on a appliqué la règle (3) ou la règle (4).*
- *si on a appliqué la règle (3), on a alors* $META \vdash_{PROLOG} R(db_1 * nil, l, 1)$, $META \vdash_{PROLOG} NIL(nil)$ *et* $META \vdash_{PROLOG} B_{exp}(db_1, l)$*. Or* $META \vdash_{PROLOG} B_{exp}(db_1, l)$ *implique que* $l \in DB_1$ *d'après la règle (1), donc par définition de* ψ*, on a* $\vdash_{MF} \psi \rightarrow B^1_{db_1} l$ *;*
- *si on a appliqué la règle (4), on a alors* $META \vdash_{PROLOG} R(db_1 * nil, l, 0)$, $META \vdash_{PROLOG} NIL(nil)$ *et* $META \nvdash_{PROLOG} B_{exp}(db_1, l)$*. PROLOG échoue donc à prouver le but* $B_{exp}(db_1, l)$*, donc* $l \notin DB_1$ *et donc par définition de* $\psi \vdash_{MF} \psi \rightarrow B^0_{db} l$.

Preuve de $\mathcal{H}(m)$ ***avec*** $m > 1$
Soit $m > 1$*. On suppose que* $\mathcal{H}(k)$ *est vraie pour tout* $k \leq m$ *et on va démontrer que* $\mathcal{H}(m+1)$ *est vraie.*
Supposons que $META \vdash_{PROLOG} R(db_1 * \ldots * (db_n * nil), l, i)$*. Dans ce cas, on a appliqué la règle (2). Donc PROLOG a prouvé* $R(db_1 * db, l, i)$ *(avec* $db \equiv db_2 * \ldots * (db_n * nil)$*),* $\neg NIL(db)$*,* $R(db_1, l, i_1)$*,* $R(db, l, i_2)$ *et* $(i = i_1 + i_2)$*. Donc* $\exists i_1 \exists i_2$ *tq* $i_1 + i_2 = i$ *et* $META \vdash_{PROLOG} R(db_1, l, i_1)$ *et* $META \vdash_{PROLOG} R(db, l, i_2)$*. Ces preuves ont été faites en m pas.*
D'après le pas d'induction, $\exists i_1 \exists i_2$ *tq* $i_1 + i_2 = i$*,* $\vdash_{MF} \psi \rightarrow B^{i_1}_{db_1} l$ *et* $\vdash_{MF} \psi \rightarrow B^{i_2}_{db} l$*. Donc d'après l'axiome (A5), on a* $\vdash_{MF} \psi \rightarrow B^i_{db_1 * db} l$.
On a donc démontré le sens $\Longrightarrow$*) du lemme. Nous montrons le sens* $\Longleftarrow$ *du lemme toujours par induction en posant :*

Définition A.2.4. $\mathcal{H}'(n)$ *: on prouve dans* $MF \vdash_{MF} \psi \rightarrow B^i_{db_1 * \ldots * db_n} l$ *en n pas* $\Rightarrow$ $META \vdash_{PROLOG} R(db_1 * \ldots * (db_n * nil), l, i)$.

Preuve de $\mathcal{H}'(0)$
On suppose que la longueur de la preuve dans MF *est nulle. Dans ce cas,* $B^i_{db_1 * \ldots * db_n} l$ *est de la forme* $B^1_{db} l$ *ou* $B^0_{db} l$ *avec db une base primitive.*
- *si on a* $B^1_{db} l$*, alors* $l \in DB$*. Donc d'après la règle (1)* $META \vdash_{PROLOG} B_{exp}(db, l)$ *et donc d'après la règle (3)* $META \vdash_{PROLOG} R(db * nil, l, 1)$ *;*
- *si on a* $B^0_{db} l$*, alors* $l \notin DB$*. D'après la règle (1),* $META \nvdash_{PROLOG} B_{exp}(db, l)$ *et par négation par echec* $META \vdash_{PROLOG} \neg B_{exp}(db, l)$*. Par application de (4), on a alors* $META \vdash_{PROLOG} R(db * nil, l, 0)$.

Preuve de $\mathcal{H}'(m)$ ***avec*** $m > 0$
Supposons que $\mathcal{H}'$ *soit vraie pour tous les rangs inférieurs à un certain rang* m*,* $m > 0$ *et montrons que* $\mathcal{H}'$ *est vraie au rang* $m + 1$.
Supposons que l'on prouve $\vdash_{MF} \psi \rightarrow B^i_{db} l$ *en m pas. Cette formule a forcément été déduite en utilisant (A5). Donc on a prouvé* $\vdash_{MF} \psi \rightarrow B^{i_1}_{db_1} l$ *et* $\vdash_{MF} \psi \rightarrow B^{i_2}_{db_2} l$ *avec* $i_1 + i_2 = i$ *et* $db = db_1 * db_2$*. Ces preuves ont été faites avec une longueur inférieure ou*

égale à m. On peut donc appliquer l'hypothèse d'induction et donc
$META \vdash_{PROLOG} R(db_1, l, i_1)$, $META \vdash_{PROLOG} R(db_2, l, i_2)$ *et*
$META \vdash_{PROLOG} (i_1 + i_2 = i)$.
D'après la règle (2), on a $META \vdash_{PROLOG} R(db_1 * db_2, l, i)$.
Le sens $\Longleftarrow$ *est donc prouvé.*
□ .

On peut maintenant démontrer la propriété.

Preuve. ***Démonstration de (1)***
Démonstration du sens $\Longrightarrow$*)*
On a $META \vdash_{PROLOG} B(db_1 * \ldots * (db_n * nil), l)$, *donc on a appliqué la règle (5). Il existe donc i et j tels que* $META \vdash_{PROLOG} R(db_1 * \ldots * (db_n * nil), l, i)$, $META \vdash_{PROLOG} R(db_1 * \ldots * (db_n * nil), l', j)$, $META \vdash_{PROLOG} neg(l, l')$ *et* $META \vdash_{PROLOG} (i > j)$.
Donc d'après le lemme A.2.9, $\exists i \exists j\ i > j$ *tels que* $\vdash_{MF} \psi \rightarrow B^i_{db_1 * \ldots * db_n} l$ *et* $\vdash_{MF} \psi \rightarrow B^j_{db_1 * \ldots * db_n} \neg l$. *Donc d'après l'axiome (A7),* $\vdash_{MF} \psi \rightarrow B_{db_1 * \ldots * db_n} l$.
Démonstration du sens $\Longleftarrow$*)*
$\vdash_{MF} \psi \rightarrow B_{db_1 * \ldots * db_n} l$. *On a donc appliqué (A7). Donc* $\exists i \exists j$ *tels que* $i > j$, $\vdash_{MF} \psi \rightarrow B^i_{db_1 * \ldots * db_n} l$ *et* $\vdash_{MF} \psi \rightarrow B^j_{db_1 * \ldots * db_n} \neg l$. *D'après le lemme A.2.9, on a donc* $META \vdash_{PROLOG} R(db_1 * \ldots * (db_n * nil), l, i)$, $META \vdash_{PROLOG} R(db_1 * \ldots * (db_n * nil), l', j)$, $META \vdash_{PROLOG} neg(l, l')$ *et* $META \vdash_{PROLOG} (i > j)$. *En appliquant la règle (5), on obtient* $META \vdash_{PROLOG} B(db_1 * \ldots * (db_n * nil), l)$.

Démonstration de (2)
Démonstration du sens $\Longrightarrow$*)*
PROLOG échoue à prouver $B_{db}l$, *donc (PROLOG réussit à prouver* $B_{db}l$*) est fausse, donc par la proposition précédente* $\nvdash_{MF} \psi \rightarrow B_{db}l$. *D'après le théorème 8.1.2, on a* $\vdash_{MF} psi \rightarrow \neg B_{db}l$.
Démonstration du sens $\Longleftarrow$*)*
On a $\vdash_{MF} psi \rightarrow \neg B_{db}l$, *donc* $\nvdash_{MF} psi \rightarrow B_{db}l$ *(d'après le théorème 8.1.2. Donc (*$\vdash_{MF} psi \rightarrow B_{db}l$*) est fausse, d'après le lemme A.2.9 (PROLOG réussit à prouver* $B(db, l)$*) est fausse donc PROLOG échoue à prouver* $B(db, l)$.
□ .

A.2.5 Évaluateur de requêtes

Propriété 8.4.1. *Soit* $DB = \langle EDB, IDB \rangle$ *une base de données qui est équivalente à un ensemble de littéraux de base. Soient* $l_1, \ldots, l_n$ *des littéraux de base de LO tels que* $l_1 \vee \ldots \vee l_n$ *n'est pas une tautologie. Alors,*
$EDB \cup IDB \models l_1 \ \vee \ \ldots \ \vee \ l_n$ ssi $\exists i_0 \in \{1 \ldots n\}\ \ EDB \cup IDB \models l_{i_0}$

Preuve. *La preuve de* $\Longleftarrow$*) est évidente.*
Preuve de $\Longrightarrow$*)*
Soient $l_1, \ldots, l_n$ *des littéraux de base de LO tels que* $l_1 \vee \ldots \vee l_n$ *n'est pas une tautologie et* $EDB \cup IDB \models l_1 \vee \ldots \vee l_n$.
Supposons que $\forall i \in \{1, \ldots, n\}\ EDB \cup IDB \not\models l_i$ *(hyp).*
D'un côté, dénotons par $\{i_1, \ldots, i_m\}$ *le sous-ensemble minimal de* $\{1, \ldots, n\}$ *tel que* $EDB \cup IDB \models l_{i_1} \vee \ldots \vee l_{i_m}$ *(i.e.* $\forall j \in (\{1, \ldots, n\} - \{i_1, \ldots, i_m\})\ \ EDB \cup IDB \models \neg l_j$*).*

Comme $EDB \cup IDB \models l_{i_1} \vee \ldots \vee l_{i_m}$ $\forall i \in \{i_1, \ldots, i_m\}$ $\exists HM_i$ *modèle de Herbrand de* $EDB \cup IDB$ *tel que* $HM_i \models l_i$.
De plus, remarquons que $m > 1$, *sinon* $\forall HM$ *modèle de Herbrand de* $EDB \cup IDB$ $HM \models l_{i_1}$ *ce qui contredit (hyp).*
D'un autre côté, $\forall i \in \{i_1, \ldots, i_m\}$ $\exists HM_i$ *modèle de Herbrand de* $EDB \cup IDB$ *tel que* $HM_i \models \neg l_i$, *cf. (hyp).*
Donc, $\forall i \in \{i_1, \ldots, i_m\}$ $l_i \in L$ *et* $\neg l_i \in L$ *(cf. définition 8.4.3).*
$l_{i_1} \vee \ldots \vee l_{i_m}$ *n'est pas une tautologie, donc* $\forall i \in \{i_1, \ldots, i_m\}$ $\forall j \in (\{i_1, \ldots, i_m\} - \{i\})$ $l_i \not\equiv \neg l_j$. *Donc,* $\neg l_{i_1} \wedge \ldots \wedge \neg l_{i_m}$ *est satisfaisable.*
Dans ce cas, d'après la définition 8.4.3, comme $\langle EDB, IDB \rangle$ *est équivalent à un ensemble de littéraux de base,* $\exists HM_{i_0}$ *modèle de Herbrand de* $EDB \cup IDB$ *tel que* $HM_{i_0} \models \neg l_{i_1} \wedge \ldots \neg \wedge l_{i_m}$. *Donc,* $HM_{i_0} \models \neg l_1 \wedge \ldots \neg \wedge l_n$.
Ceci contredit le fait que $EDB \cup IDB \models l_1 \vee \ldots \vee l_n$, *donc (hyp) est fausse.*
□.

Propriété 8.4.2. *Soit* $db = \langle EDB, IDB \rangle$ *une base de données telle que* IDB *n'est pas récursive. Soit* l *un littéral de base. Alors PROLOG prouve* $B_{exp}(db, l)$ *ssi* $EDB \cup IDB \models l$.

Preuve. *Preuve de* $\Longrightarrow$*)*
Nous prouvons dans le même temps les deux propositions suivantes :
(1) PROLOG prouve $B_{exp}(db, l) \Longrightarrow EDB \cup IDB \models l$ *et*
(2) PROLOG prouve $B_{conj}(db, l_1 \wedge \ldots \wedge l_n) \Longrightarrow EDB \cup IDB \models l_1 \wedge \ldots \wedge l_m$
Nous noterons :

$\mathcal{H}(n)$: *PROLOG prouve* $B_{exp}(db, l)$ *en* n *pas* $\Longrightarrow EDB \cup IDB \models l$
PROLOG prouve $B_{conj}(db, l_1 \wedge \ldots \wedge l_m)$ *en* $n+1$ *pas* $\Longrightarrow$ $EDB \cup IDB \models l_1 \wedge \ldots \wedge l_m$

Nous prouverons le sens $\Longrightarrow$*) de (1) et de (2) en prouvant par induction que* $\mathcal{H}(n)$ *est vraie pour tout* $n \geq 2$.
Premièrement, prouvons que PROLOG prouve toujours $B_{exp}(db, l)$ *ou* $B_{conj}(db, l_1 \wedge \ldots \wedge l_m)$ *en un nombre fini de pas.*
Supposons que PROLOG ne puisse pas prouver $B_{exp}(db, l)$ *ou* $B_{conj}(db, l_1 \wedge \ldots \wedge l_m)$ *en un nombre fini de pas. Dans ce cas, cela signifie que PROLOG essaye de prouver une formule du type* $B_{exp}(db, l')$ *ou* $B_{conj}(db, l'_1 \wedge \ldots \wedge l'_m)$ *en utilisant la même formule (cf. axiomes (1.4) et (1.6)). Ces cas ne peuvent se produire que lorsque PROLOG peut prouver* $IDB(db, r \rightarrow l')$ *où* l' *apparait dans la conjonction* r. *Mais* IDB *n'est pas récursive, donc cela ne peut pas arriver.*
Preuve de $\mathcal{H}(2)$ *: PROLOG prouve* $B_{exp}(db, l)$ *en deux cas. Dans ce cas, le premier axiome utilisé par PROLOG est (1.3). PROLOG doit donc prouver* $EDB(db, l)$ *en utilisant (1.1). Prouver* $EDB(db, l)$ *signifie que* $l \in EDB$. *Donc* $EDB \cup IDB \models l$.
PROLOG prouve $B_{conj}(db, l_1 \wedge \ldots \wedge l_m)$ *en trois pas. Le seul axiome que PROLOG puisse utiliser pour le premier pas est (1.6). Prolog doit donc prouver :*

- $B_{conj}(db, l_2 \wedge \ldots \wedge l_m)$ *en deux pas. Le seul axiome que PROLOG puisse utiliser est (1.5), donc en fait* $l_2 \wedge \ldots \wedge l_m \equiv nil$;
- $B_{exp}(db, l)$ *en deux pas. D'après la preuve précédente,* $EDB \cup IDB \models l_1$.

Donc, $EDB \cup IDB \models l_1 \wedge \ldots \wedge l_m$ *et* $\mathcal{H}(2)$ *est vraie.*
Prouvons également que $\mathcal{H}(3)$ *est vraie parce que c'est un cas particulier.*
Preuve de $\mathcal{H}(3)$ *: supposons que PROLOG prouve* $B_{exp}(db, l)$ *en trois pas. D'après l'axiome (1.4), PROLOG doit donc prouver* $IDB(db, r \rightarrow l)$ *et* $B_{conj}(db, r)$ *en deux pas.*

Mais PROLOG ne peut prouver $B_{conj}(db, r)$ en deux pas (il a besoin d'au moins trois pas pour prouver une formule du type B_{conj}), donc la proposition « PROLOG prouve $B_{exp}(db, l)$ en trois pas $\Longrightarrow EDB \cup IDB \models l$ » est vraie.
PROLOG prouve $B_{conj}(db, l_1 \wedge \ldots \wedge l_m)$ en quatre pas. Le seul axiome que PROLOG peut utiliser pour le premier pas est (1.6). Donc PROLOG doit prouver :
- *$B_{conj}(db, l_2 \wedge \ldots \wedge l_m)$ en trois pas. Comme $\mathcal{H}(2)$ est vraie, $EDB \cup IDB \models l_2 \wedge \ldots \wedge l_m$;*
- *$B_{exp}(db, l)$ en au moins trois pas. Donc PROLOG prouve $B_{exp}(db, l)$ en deux pas. Comme $\mathcal{H}(2)$ est vraie, $EDB \cup IDB \models l_1$.*

Donc $EDB \cup IDB \models l_1 \wedge \ldots \wedge l_m$ et $\mathcal{H}(3)$ est vraie.
***Pas d'induction pour** $\mathcal{H}$: soit n un entier tel que $n \geq 3$. Supposons que $\mathcal{H}(k)$ soit vraie pour tout $k \in \{3, \ldots, n\}$.*
Premièrement, supposons que PROLOG prouve $B_{exp}(db, l)$ en $(n+1)$ pas. Dans ce cas, d'après l'axiome (1.4) et (1.2) :
- *PROLOG prouve $B_{conj}(db, r)$ en n pas. Comme $\mathcal{H}(n-1)$ est vraie, $EDB \cup IDB \models r$;*
- *PROLOG prouve $IDB(db, r \rightarrow l)$ en deux pas. Donc $EDB \cup IDB \models r \rightarrow l$.*

Donc $EDB \cup IDB \models l$.
Supposons que PROLOG prouve $B_{conj}(db, l_1 \wedge \ldots \wedge l_m)$ en $n+2$ pas. Dans ce cas, d'après l'axiome (1.6) :
- *PROLOG prouve $B_{exp}(db, l_1)$ en $n+1$ pas. D'après la preuve précédente, $EDB \cup IDB \models l_1$;*
- *PROLOG prouve $B_{conj}(db, l_2 \wedge \ldots \wedge l_m)$ en $n+1$ pas. Comme $\mathcal{H}(n)$ est vraie, $EDB \cup IDB \models l_2 \wedge \ldots \wedge l_m$.*

Donc $EDB \cup IDB \models l_1 \wedge \ldots \wedge l_m$, donc $\mathcal{H}(n+1)$ est vraie.
Donc $(H)(n)$ est vraie pour tout $n \geq 2$. Donc PROLOG prouve $B_{exp}(db, l) \Longrightarrow EDB \cup IDB \models l$.
Preuve de $\Longleftarrow$)
Supposons que LO soit un langage du premier ordre tel que les symboles de constantes soient notés $\{a_1, \ldots, a_n\}$, les symboles de prédicats $\{P_1, \ldots, P_m\}$ et les symboles de variables $\{x_1, \ldots, x_k\}$. Pour toute clause $\mathcal{C}$ de IDB contenant les variables $x_{i_1}, \ldots, x_{i_{k_\mathcal{C}}}$, nous appelons une instance de base de $\mathcal{C}$ toute formule $\mathcal{F}$ de LO telle que $\mathcal{F} \equiv \mathcal{C}[x_{i_1}/a_{j_1}, \ldots, x_{i_{k_\mathcal{C}}}/a_{j_{k_\mathcal{C}}}]$ (i.e. une clause dans laquelle tout symbole de variables est remplacé par un symbole de constante).
Soit succ la fonction telle que pour tout ensemble E de littéraux de base de LO :

$$\begin{aligned} succ(E) \;=\; & E \cup \{P_i(a_{i_1}, \ldots, a_{i_{m_i}}) \;:\; \neg P_{j_1}(a_{j_{1,1}}, \ldots, a_{j_{1,l_1}}) \wedge \ldots \wedge \neg P_{j_h}(a_{j_{h,1}}, \ldots, a_{j_{h,l_h}}) \\ & \rightarrow P_i(a_{i_1}, \ldots, a_{i_{m_i}}) \text{ est une instance de base d'une clause de } h(c) \text{ où } c \in IDB \text{ et} \\ & \forall k \in \{j_1, \ldots, j_h\} \;\; \neg P_{j_k}(a_{j_{1,1}}, \ldots, a_{j_{1,l_k}}) \in E\} \end{aligned}$$

Definissons la fonction $\mathcal{T}$ par induction :

$$\begin{aligned} \mathcal{T}_1(EDB) &= EDB \\ \forall n \geq 1 \;\; \mathcal{T}_{n+1}(EDB) &= succ(\mathcal{T}_n(EDB)) \end{aligned}$$

Comme LO est un langage fini, il est facile de prouver que $\exists i_0 \geq 1$ tel que $\lim_{n \to \infty} \mathcal{T}_n(EDB) = \mathcal{T}_{i_0}(EDB)$ (succ est une suite croissante dans un ensemble fini). Il est également évident que pour tout littéral de base l, $EDB \cup IDB \models l \Longleftrightarrow l \in \mathcal{T}_{i_0}(EDB)$.
Dénotons $\mathcal{T}_{i_0}(EDB)$ par $\mathcal{T}^{i_O}(EDB)$.
Soit l un littéral de base. Prouvons par induction que pour tout $i \geq 1$:

$\mathcal{K}(i)$: $l \in \mathcal{T}^i(EDB) \Longrightarrow$ *PROLOG prouve* $B_{exp}(db, l)$.
Preuve de $\mathcal{K}(1)$ *:* $l \in \mathcal{T}^1(EDB)$*, donc par définition* $l \in EDB$*. D'après les axiomes (1.1) et (1.3), PROLOG prouve* $B_{exp}(db, l)$.
Pas d'induction de $\mathcal{K}$ *: supposons que pour un certain* $i \geq 1$ $\mathcal{K}(i)$ *est vraie. Supposons que* $l \in \mathcal{T}^{i+1}(EDB)$*. Dans ce cas, il existe une formule* $\neg P_{j_1}(a_{j_{1,1}}, \ldots, a_{j_{1,l_1}}) \wedge \ldots \wedge \neg P_{j_h}(a_{j_{h,1}}, \ldots, a_{j_{h,l_h}}) \rightarrow l$ *qui est une instance d'une clause de* IDB *telle que* $\forall k \in \{j_1, \ldots, j_h\}$ $\neg P_{j_k}(a_{j_{1,1}}, \ldots, a_{j_{1,l_k}}) \in \mathcal{T}_i(EDB)$.
D'après l'axiome (1.2), PROLOG prouve $IDB(db, \neg P_{j_1}(a_{j_{1,1}}, \ldots, a_{j_{1,l_1}}) \wedge \ldots \wedge \neg P_{j_h}(a_{j_{h,1}}, \ldots, a_{j_{h,l_h}}) \rightarrow l)$.
Comme $\forall k \in \{j_1, \ldots, j_h\}$ $\neg P_{j_k}(a_{j_{1,1}}, \ldots, a_{j_{1,l_k}}) \in \mathcal{T}_i(EDB)$ *et* $\mathcal{K}(i)$ *est vraie, PROLOG prouve* $B_{exp}(db, \neg P_{j_k}(a_{j_{1,1}}, \ldots, a_{j_{1,l_k}}))$ *pour tout* $k \in \{j_1, \ldots, j_h\}$*. Donc, d'après les axiomes (1.5) et (1.6), PROLOG prouve* $B_{conj}(db, \neg P_{j_1}(a_{j_{1,1}}, \ldots, a_{j_{1,l_1}}) \wedge \ldots \wedge \neg P_{j_h}(a_{j_{h,1}}, \ldots, a_{j_{h,l_h}}))$.
Donc, d'après l'axiome (1.4), PROLOG prouve $B_{exp}(db, l)$*. Donc* $\mathcal{K}(i+1)$ *est vraie. Par induction,* $\mathcal{K}(i)$ *est vraie pour tout* $i \geq 1$*. En particulier,* $\mathcal{K}(i_0)$ *est vraie, donc PROLOG prouve* $EDB \cup IDB \models l \Longrightarrow B_{exp}(db, l)$.
□.

A.3 Preuves de la partie III

Propriété 11.1.1.
– φ *est contrôlable par* a_i *ssi* $\forall \mathcal{U}_{\neg\varphi} = \bigcup_{p \in PI(\neg\varphi)} p$ *tel que* $\forall l \in lit(PROP)$ $l \in \mathcal{U}_{\neg\varphi} \Rightarrow \neg l \notin \mathcal{U}_{\neg\varphi}$ *alors* $\exists \mathcal{U}_\varphi \subseteq PI(\varphi)$ *tel que*
  1. $\mathcal{U}_\varphi \neq \phi$;
  2. $\forall \mathcal{C} \subseteq 2^{lit(PROP)}$ *tel que* $l \in \mathcal{C} \Rightarrow \neg l \notin \mathcal{C}$ *et* $\mathcal{U}_{\neg\varphi} \subseteq \mathcal{C}$*, alors* $\exists p' \in \mathcal{U}_\varphi$ *tel que* $\overline{C}_{a_i}(p') \subseteq \mathcal{C}$.

– φ *est influençable par* a_i *ssi :*
  1. $\exists \mathcal{U}_{\neg\varphi} = \bigcup_{p \in PI(\neg\varphi)} p$ *tel que* $\forall l \in lit(PROP)$ $l \in \mathcal{U}_{\neg\varphi} \Rightarrow \neg l \notin \mathcal{U}_{\neg\varphi}$;
  2. $\exists \mathcal{C} \subseteq 2^{lit(PROP)}$ *tel que* $l \in \mathcal{C} \Rightarrow \neg l \notin \mathcal{C}$ *et* $\mathcal{U}_{\neg\varphi} \subseteq \mathcal{C}$;
  3. $\exists p' \in PI(\neg\varphi)$ *tel que* $\overline{C}_{a_i}(p') \subseteq \mathcal{C}$.

Preuve. *Nous allons démontrer l'équivalence pour chacune des notions. Soit* φ *une proposition.*

1. *nous allons tout d'abord montrer l'équivalence pour la contrôlabilité.*
   – *sens* $\Rightarrow$.
   Supposons que φ *soit contrôlable par* a_i*. Alors dans ce cas,* $\forall w \in W$ $w \models \neg\varphi$ $\Rightarrow \exists w' \in W$ $w' \models \varphi$ *et* $(w' - w) \subseteq C_{a_i}$.
   Soit $\mathcal{U}_{\neg\varphi} = \bigcup_{p \in PI(\neg\varphi)} p$ *tel que* $\forall l \in lit(PROP)$ $l \in \mathcal{U}_{\neg\varphi} \Rightarrow \neg l \notin \mathcal{U}_{\neg\varphi}$*. Soit* $w \in W$ *tel que* $w \models \bigwedge_{p \in \mathcal{U}_{\neg\varphi}} \bigwedge p$ *(c'est possible d'après la définition de* $\mathcal{U}_{\neg\varphi}$*), alors* $w \models \neg\varphi$.
   Comme $w \models \neg\varphi$*,* $\exists w' \in W$ $w' \models \varphi$*, donc* $\exists \mathcal{U}_\varphi(w) \subseteq PI(\neg\varphi)$ *tel que* $\forall p' \in \mathcal{U}_\varphi(w)$ $w' \models p'$.
   Soit $p' \in \mathcal{U}_\varphi(w)$*.* $w' \models p'$ *donc* $w' \models \overline{C}_{a_i}(p')$*. Or* $(w' - w) \subseteq C_{a_i}$*, donc* $w \models \overline{C}_{a_i}(p')$.

Soit $\mathcal{C} \in 2^{lit(PROP)}$ *tel que* $w \models \bigwedge_{l' \in \mathcal{C}} l'$ *et* $\mathcal{U}_{\neg\varphi} \subseteq \mathcal{C}$. *Alors* $\forall l' \in \mathcal{C}\ \neg l' \notin \mathcal{C}$ *(car* $\bigwedge_{l' \in \mathcal{C}} l'$ *est cohérente).*
De plus, $\forall p' \in \mathcal{U}_\varphi(w)\ w \models \overline{C}_{a_i}(p')$, *donc* $\forall p' \in \mathcal{U}_\varphi(w)\ \overline{C}_{a_i}(p') \subseteq \mathcal{C}$.
Si l'on considère tous les mondes $w \in W$ *tels que* $w \models \bigwedge_{p \in \mathcal{U}_{\neg\varphi}} p$, *alors on peut construire un sous-ensemble* $\mathcal{U}_\varphi$ *de* $PI(\varphi)$ *non vide et tel que* $\forall \mathcal{C} \in 2^{lit(PROP)}$ *tel que* $l \in \mathcal{C} \Rightarrow \neg l \notin \mathcal{C}$ *et* $\mathcal{U}_{\neg\varphi} \subseteq \mathcal{C}$, *alors* $\exists p' \in \mathcal{U}_\varphi$ *tel que* $\overline{C}_{a_i}(p') \subseteq \mathcal{C}$.

– *sens* $\Leftarrow$.
Soit φ *une proposition telle que* $\forall \mathcal{U}_{\neg\varphi} = \bigcup_{p \in PI(\neg\varphi)} p$ *tel que* $\forall l \in lit(PROP)$ $l \in \mathcal{U}_{\neg\varphi} \Rightarrow \neg l \notin \mathcal{U}_{\neg\varphi}$ *alors* $\exists \mathcal{U}_\varphi \subseteq PI(\varphi)$ *tel que*

(a) $\mathcal{U}_\varphi \neq \phi$;

(b) $\forall \mathcal{C} \in 2^{lit(PROP)}$ *tel que* $l \in \mathcal{C} \Rightarrow \neg l \notin \mathcal{C}$ *et* $\mathcal{U}_{\neg\varphi} \subseteq \mathcal{C}$, *alors* $\exists p' \in \mathcal{U}_\varphi$ *tel que* $\overline{C}_{a_i}(p') \subseteq \mathcal{C}$.

Soit $w \in W$ *tel que* $w \models \neg\varphi$. *Alors* $\exists \mathcal{P} = \{p_1, \ldots, p_n\} \subseteq PI(\neg\varphi)$ *tel que* $\forall i \in \{1, \ldots, n\}\ w \models p_i$ *et* $\forall p \in PI(\neg\varphi) - \mathcal{P}\ w \not\models p$.
Soit $\mathcal{U}_{\neg\varphi} = \bigcup_{i \in \{1,\ldots,n\}} p_i$.
Comme $\forall i \in \{1, \ldots, n\}\ w \models p_i$, *alors* $\forall l \in lit(PROP)\ l \in \mathcal{U}_{\neg\varphi} \Rightarrow \neg l \notin \mathcal{U}_{\neg\varphi}$.
Donc $\exists \mathcal{U}_\varphi \subseteq PI(\varphi)$ *respectant les propriétés (a) et (b).*
Soit $\mathcal{C}(w) \subseteq 2^{lit(PROP)}$ *tel que* $w \models \bigwedge \mathcal{C}(w)$. *Dans ce cas,* $\exists p' \in \mathcal{U}_\varphi$ *tel que* $\overline{C}_{a_i}(p') \subseteq \mathcal{C}$.
Soit $w' \in W$ *tel que* $(w' - w) = C_{a_i}(p')$. *Soit* $l \in \overline{C}_{a_i}$ *alors* $l \in \mathcal{C}(w)$, *donc* $w \models l$, *donc* $w' \models l$ *(car* $\overline{C}_{a_i}(p')$ *et* $C_{a_i}(p')$ *sont disjoints). On peut donc écrire que* $w' \models \bigwedge \overline{C}_{a_i}(p')$.
Comme $(w' - w) = C_{a_i}(p')$, $w' \models \bigwedge C_{a_i}(p') \wedge \bigwedge \overline{C}_{a_i}(p')$, *donc* $w' \models p'$. *Dans ce cas,* $w' \models \varphi$.
En conclusion, $\forall w \in W$ *tel que* $w \models \neg\varphi\ \exists w' \in W$ *tel que* $(w' - w) \subseteq C_{a_i}$ *et* $w' \models \varphi$ *donc* φ *est contrôlable par* a_i.

2. *nous allons maintenant démontrer l'équivalence pour l'influençabilité.*
– *sens* $\Rightarrow$.
Supposons que φ *soit influençable par* a_i. *Alors dans ce cas,* $\exists w \in W\ w \models \neg\varphi$ *et* $\exists w' \in W\ w' \models \varphi$ *et* $(w' - w) \subseteq C_{a_i}$.
$w \models \neg\varphi$ *donc* $\exists \mathcal{P} \subseteq PI(\neg\varphi)$ *tel que* $\forall p \in \mathcal{P} w \models p$. *Soit* $\mathcal{U}_{\neg\varphi} = \bigcup_{p \in \mathcal{P}'} p$, *alors* $\forall l \in lit(PROP)\ l \in \mathcal{U}_{\neg\varphi} \Rightarrow \neg l \notin \mathcal{U}_{\neg\varphi}$ *car* $w \models \bigwedge_{p \in \mathcal{U}_{\neg\varphi}} \bigwedge p$.
$w' \models \varphi$, *donc* $\exists p' \in PI(\neg\varphi)$ *tel que* $w' \models p'$. $w' \models p'$ *donc* $w' \models \overline{C}_{a_i}(p')$. *Or* $(w' - w) \subseteq C_{a_i}$, *donc* $w \models \overline{C}_{a_i}(p')$.
Soit $\mathcal{C} \in 2^{lit(PROP)}$ *tel que* $w \models \bigwedge_{l' \in \mathcal{C}} l'$ *et* $\mathcal{U}_{\neg\varphi} \subseteq \mathcal{C}$. *Alors* $\forall l' \in \mathcal{C}\ \neg l' \notin \mathcal{C}$ *(car* $\bigwedge_{l' \in \mathcal{C}} l'$ *est cohérente).*
De plus, $w \models \overline{C}_{a_i}(p')$, *donc* $\overline{C}_{a_i}(p') \subseteq \mathcal{C}$.
Donc :

(a) $\exists \mathcal{U}_{\neg\varphi} = \bigcup_{p \in PI(\neg\varphi)} p$ *tel que* $\forall l \in lit(PROP)\ l \in \mathcal{U}_{\neg\varphi} \Rightarrow \neg l \notin \mathcal{U}_{\neg\varphi}$;

(b) $\exists \mathcal{C} \subseteq 2^{lit(PROP)}$ *tel que* $l \in \mathcal{C} \Rightarrow \neg l \notin \mathcal{C}$ *et* $\mathcal{U}_{\neg\varphi} \subseteq \mathcal{C}$;

(c) $\exists p' \in PI(\neg\varphi)$ *tel que* $\overline{C}_{a_i}(p') \subseteq \mathcal{C}$.

– *sens* $\Leftarrow$.
Supposons que :

(a) $\exists \mathcal{U}_{\neg\varphi} = \bigcup_{p \in PI(\neg\varphi)} p$ *tel que* $\forall l \in lit(PROP)\ l \in \mathcal{U}_{\neg\varphi} \Rightarrow \neg l \notin \mathcal{U}_{\neg\varphi}$;

(b) $\exists \mathcal{C} \subseteq 2^{lit(PROP)}$ *tel que* $l \in \mathcal{C} \Rightarrow \neg l \notin \mathcal{C}$ *et* $\mathcal{U}_{\neg\varphi} \subseteq \mathcal{C}$;

(c) $\exists p' \in PI(\neg\varphi)$ *tel que* $\overline{C}_{a_i}(p') \subseteq \mathcal{C}$.

Soit $w \in W$ *le monde tel que* $w \models \bigwedge \mathcal{C}$. *Alors d'après (a) et (b),* $w \models \neg\varphi$. *De plus, d'après (c),* $w \models \bigwedge \overline{C}_{a_i}(p')$.
Soit $w' \in W$ *le monde tel que* $(w' - w) = C_{a_i}(p')$. *Dans ce cas* $w' \models \bigwedge C_{a_i}(p')$. *Or* $(w' - w) = C_{a_i}(p')$ *et* $w \models \bigwedge \overline{C}_{a_i}(p')$ *donc* $w' \models \bigwedge \overline{C}_{a_i}(p')$ *(car* $C_{a_i}(p')$ *et* $\overline{C}_{a_i}(p')$ *sont disjoints).*
Donc $w' \models \varphi$, *donc* φ *est influençable par* a_i.

□ .

Propriété 12.2.1. $D(KB)$ *est un ensemble cohérent.*

Preuve. $D(KB) = NC(KB) \cup Com_{+,A} \cup Com_{-,A}$.
On sait d'après l'hypothèse 12.1.4 que $Com_{+,A} \cup Com_{-,A}$ *est cohérent.*
$NC(KB)$ *est l'ensemble des formules* φ *de* $Cl(KB)$ *pour lesquelles* KB *n'est pas un contexte pour* $\neg\varphi$. $NC(KB)$ *est cohérent car* KB *l'est par définition.*
Supposons qu'il existe un littéral l *tel que* $l \in NC(KB)$ *et* $\neg l \in Com_{+,A} \cup Com_{-,A}$ *(comme* $Com_{+,A} \cup Com_{-,A}$ *est un ensemble de littéraux, on peut généraliser aux formules de* $NC(KB)$ *facilement). Dans ce cas,* KB *n'est pas un contexte pour* $\neg l$. *Or,* $\neg l \in Com_{+,A} \cup Com_{-,A}$, *donc* $\neg l$ *est contrôlable, donc par définition,* KB *est un contexte pour* $\neg l$ *(car* l *est un littéral). Donc* $NC(KB) \cup Com_{+,A} \cup Com_{-,A}$ *est bien cohérent.*
□ .

Propriété 12.2.2. *Soit* l *un littéral de* $PROP$. *Si* l *est un CK-but de* $\mathcal{A}$, *alors* $\exists a_i \in \mathcal{A}$ *tel que l'on a* $EGoal_{a_i}(l)$.

Preuve. *Soit* l *un CK-but du groupe* $\mathcal{A}$. *Par définition, on a* $\Sigma \models I(\varphi|NC(KB))$ *et* KB *est un contexte pour* φ *par rapport à* $\mathcal{A}$ *(0). En particulier,* l *est contrôlable par* $\mathcal{A}$.
Supposons que l *ne soit le but effectif d'aucun agent de* $\mathcal{A}$. *On a donc* $\forall a_i \in \mathcal{A}$ $\Sigma \not\models I(l|D(KB))$ *ou* l *non contrôlable par* a_i *(car comme* l *est un littéral, tous les* KB *sont un contextes pour lui s'il est contrôlable).*
Comme l *est un littéral, il existe au moins un agent* a_i *de* $\mathcal{A}$ *qui contrôle* l. *On a donc* $\Sigma \not\models I(l|D(KB))$. *Donc :*

$$\begin{aligned}
& \exists M_0, M_0 \models \Sigma \text{ et } M \not\models I(l|D(KB)) \\
\Longrightarrow\ & M_0 \not\models \overleftrightarrow{\Diamond}\,(D(KB) \wedge \Box(D(KB) \rightarrow l)) \\
\Longrightarrow\ & \forall w \in W \quad M_0, w \not\models D(KB) \text{ ou } \exists w' \leq w \quad M_0, w' \not\models D(KB) \rightarrow l \\
\Longrightarrow\ & \forall w \in W \quad M_0, w \models D(KB) \quad \Rightarrow \quad \exists w' \leq w \quad M_0, w' \models D(KB) \wedge \neg l \qquad \text{(A.1)}
\end{aligned}$$

(0) implique $\exists w_0 \in W \quad M_0, w_0 \models UI(KB)$ *et* $\forall w \leq w_0 \quad M_0, w \models UI(KB) \rightarrow l$.

Donc, puisque $UI(KB) \subseteq D(KB)$ *:*

$$\exists w_0 \in W \quad M_0, w_0 \models UI(KB) \text{ et } \forall w \leq w_0 \quad M_0, w \models D(KB) \rightarrow l$$

Soit $w_1 \leq w_0$*. D'après (1) si* $w_1 \models D(KB)$ *alors il existe un* $w' \in W$ *tel que* $w' \leq w_1$ *et* $w' \models D(KB) \wedge \neg l$*. Or* $w' \leq w_0$ *(par transitivité de* $\leq$*) donc* $w' \models D(KB) \rightarrow l$ *ce qui est impossible.*
Donc :

$$\begin{array}{rl} & \forall w \leq w_0 \quad w \models \neg D(KB) \\ \Longrightarrow & \forall w \leq w_0 \quad w \models \neg(UI(KB) \wedge (Com_{+,\mathcal{A}} \wedge Com_{-,\mathcal{A}})) \\ \Longrightarrow & \forall w \leq w_0 \quad w \models UI(KB) \rightarrow \neg(Com_{+,\mathcal{A}} \wedge Com_{-,\mathcal{A}}) \end{array}$$

Or $M_0 \models I(l|UI(KB)$*, donc* $\exists w_2 \in W \quad M_0, w_2 \models UI(KB)$ *et*
$\forall w \leq w_2 \quad M_0, w \models UI(KB) \rightarrow l$*.*
Donc $M_0, min_{\leq}(w_0, w_2) \models l \wedge \neg(Com_{+,A} \wedge Com_{-,A}))$*. Ceci falsifie l'hypothèse 12.1.4.*
□ .

Bibliographie

[1] C. Alchourrón, P. Gardenfors, and D. Makinson. On the logic of theory change : Partial meet functions for contraction and revision. *Journal of Symbolic Logic*, 50 :513–530, 1985.

[2] L. Åqvist. Deontic logic. In D. Gabbay and F. Guenthner, editors, *Handbook of philosophical logic vol. II*, pages 605–714. Dordrecht :Reidel, 1984.

[3] L. Åqvist and J. Hoepelman. Some theorems about a "tree" system of deontic tense logic. In R. Hilpinen, editor, *New studies in deontic logic*, pages 187–221. Reidel, Dordrecht, 1981.

[4] K.J. Arrow. *Social choice and individual values*. Yale University Press, second edition, 1963.

[5] G. Attardi and M. Simi. Proofs in context. In J. Doyle and P. Torasso, editors, *Proceedings of the Fourth International Conference on Principles of Knowledge Representation and Reasoning (KR'94)*, pages 15–26, San Mateo, California, 1994. Morgan Kaufmann.

[6] G. Attardi and M. Simi. A formalization of viewpoints. *Fundamenta Informaticae*, 23(2,3,4) :149–174, 1995.

[7] G. Attardi and M. Simi. Communication across viewpoints. *Journal of Logic, Language and Information*, 7 :53–75, 1998.

[8] C. Barral, S. Kraus, J. Minker, and V.S. Subrahmanian. Combining knowledge bases consisting of first-order theories. *Computational Intelligence*, 8(1) :45–71, 1992.

[9] N. Belnap and M. Perloff. Seeing to it that : a canonical form for agentives. *Theoria*, 54 :175–199, 1988.

[10] S. Benferhat, D. Dubois, S. Kaci, and H. Prade. Bipolar representation and fusion of preferences in the possibilistic logic framework. In D. Fensel, F. Guinchiglia, M.-A. Williams, and D. McGuiness, editors, *Proceedings of the Eight International Conference on Principles of Knowledge Representation and Reasoning (KR'02)*, pages 421–432. Morgan Kaufmann, 2002.

[11] P. Besnard and A. Hunter. Quasi-classical logic : Non-trivializable classical reasoning from inconsistent information. *Lecture Notes in Computer Science*, 946 :44–51, 1995.

[12] C. Boutilier. Conditional logics of normality : a modal approach. *Artificial Intelligence*, 68 :87–154, 1994.

[13] C. Boutilier. Toward a logic for qualitative decision theory. In J. Doyle, E. Sandewall, and P. Torasso, editors, *Principles of Knowledge Representation and Reasoning (KR'94)*, pages 75–86. Morgan Kaufmann, 1994.

[14] C. Boutilier, R.I. Brafman, H.H. Hoos, and D. Poole. Reasoning with conditional ceteris paribus preference statements. In *Proceedings of the fifteenth Annual Conference on Uncertainty in Artificial Intelligence (UAI'99)*, pages 71–80, 1999.

[15] B. Bérard, M. Bidoit, F. Laroussinie, A. Petit, and P. Schnoebelen. *Vérification de logiciels. Techniques et outils du model-checking.* Vuibert, 1999.

[16] M.E. Bratman. *Intentions, plans and practical reason.* Harvard University Press, 1987.

[17] M.A. Brown. Conditional obligation and positive permission for agents in time. *Nordic Journal of Philosophical Logic*, 5 :83–112, 2001.

[18] M.A. Brown. Rich deontic logic : a preliminary study. In J.F. Horty and A.J. Jones, editors, *Proceedings of the Sixth International Workshop on Deontic Logic in Computer Science (ΔEON'02)*, pages 39–54, London, May 2002.

[19] S. Buvač, V. Buvač, and I.A. Mason. The semantics of propositional contexts. In Z.W. Ras and M. Zemankova, editors, *Proceedings of the Eighth International Symposium on Methodologies for Intelligent Systems, ISMIS'94*, volume 869 of *Lecture Notes in Artificial Intelligence*, pages 468–477, Charlotte, North Carolina, USA, October 1994.

[20] S. Buvač, V. Buvač, and I.A. Mason. Metamathematics of contexts. *Fundamenta Informaticae*, 23(3) :263–301, 1995.

[21] S. Buvač and I.A. Mason. Propositional logic of contexts. In R. Fikes and W. Lehnert, editors, *Proceedings of the Eleventh National Conference on Artificial Intelligence*, pages 412–419. AAAI Press, 1993.

[22] J. Carmo and A. Jones. Deontic logic and contrary-to-duties. In *Handbook of Philosophical Logic*, volume 8 : Extensions to Classical Systems 2. Kluwer Publishing Company, 2001.

[23] J. Carmo and O. Pacheco. Deontic and action logics for organized collective agency, modeled through institutionalized agents and roles. *Fundamenta Informaticae*, 48(2,3) :129–163, 2001.

[24] H.-N. Castañeda. The paradoxes of deontic logic : the solution to all of them in one fell swoop. In R. Hilpinen, editor, *New studies in deontic logic*, pages 37–85. Reidel, Dordrecht, 1981.

[25] B.F. Chellas. *Modal logic. An introduction.* Cambridge University Press, 1980.

[26] R. Chisholm. Contrary-to-duty imperatives and deontic logic. *Analysis*, 24 :33–36, 1963.

[27] L. Cholvy. Proving theorems in a multi-sources environment. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence*, pages 66–71, Chambéry, France, 1993.

[28] L. Cholvy. A logical approach to multi-sources reasoning. In M. Masuch and L. Polos, editors, *Knowledge Representation and Reasoning under Uncertainty : Logic at Work*, volume 808 of *Lecture Notes in Artificial Intelligence*, pages 183–196. Springer-Verlag, Berlin, Heidelberg, 1994.

[29] L. Cholvy. Automated reasoning with merged contradictory information whose reliability depends on topics. In C. Froideveaux and Jürg Kohlas, editors, *Symbolic and Quantitative Approaches to Reasoning with Uncertainty - Proceedings of*

European Conference ECSQARU'95, volume 946 of *Lecture Notes in Artificial Intelligence*, pages 125–132, Fribourg, July 1995. Springer.

[30] L. Cholvy. Reasoning about data provided by federated deductive databases. *Journal of Intelligent Information Systems*, 10 :49–80, 1998.

[31] L. Cholvy. Reasoning about merged information. In D.M. Gabay and Ph. Smets, editors, *Handbook of Defeasible Reasoning and Uncertainty Management Systems*, volume 3, pages 233–263. Kluwer Academic Publishers, 1998.

[32] L. Cholvy and F. Cuppens. Reasoning about norms provided by conflicting regulations. In P. McNamara and H. Prakken, editors, *Norms, logics and information systems : new studies in deontic logic and computer science*, volume 49 of *Frontiers in artificial intelligence and applications*, pages 247–262. IOS Press, Amsterdam, 1999.

[33] L. Cholvy and R. Demolombe. Reasoning with information sources ordered by topics. In *Proceedings of Artificial Intelligence : Methods, Systems and Applications (AIMSA)*, pages 151–162, Sofia, September 1994.

[34] L. Cholvy and C. Garion. An attempt to adapt a logic for conditional preferences for reasoning with contrary-to-duties. In R. Demolombe and R. Hilpinen, editors, *Proceedings of the* 5^{th} *International Workshop on Deontic Logic In Computer Science (*Δ*EON'00)*, pages 125–145. ONERA-DGA, January 2000.

[35] L. Cholvy and C. Garion. Allocation des buts affectés à un groupe d'agents. In B. Chaib-Draa and P. Enjalbert, editors, *Premières Journées Francophones Modèles Formels de l'Interaction*, pages 193–203, Toulouse, May 2001.

[36] L. Cholvy and C. Garion. An attempt to adapt a logic of conditional preferences for reasoning with contrary-to-duties. *Fundamenta Informaticae*, 48(2,3) :183–204, November 2001.

[37] L. Cholvy and C. Garion. A logic to reason on contradictory beliefs with a majority approach. In *Proceedings of IJCAI'01 Workshop on Inconsistency in Data and Knowledge*, pages 22–27, 2001.

[38] L. Cholvy and C. Garion. Utilisation d'une logique de préférences conditionnelles pour raisonner avec des normes contrary-to-duties. In *Journées nationales sur les modèles de raisonnement*, pages 109–123, Arras, France, May 2001.

[39] L. Cholvy and C. Garion. Allocation de buts affectés à un groupe d'agents. *Information - Interaction - Intelligence*, 2002. À paraître.

[40] L. Cholvy and C. Garion. Answering queries addressed to merged databases : a query evaluator which implements a majority approach. In M-S Hacid, Z.W. Raś, D.A. Zighed, and Y. Kodratoff, editors, *Foundations of Intelligent Systems - Proceedings of the* 13^{th} *International Symposium on Methodologies for Intelligent Systems, ISMIS 2002*, volume 2366 of *Lecture Notes in Artificial Intelligence*, pages 131–139. Springer, June 2002.

[41] L. Cholvy and C. Garion. Collective obligations, commitments and individual obligations : a preliminary study. In J.F. Horty and A.J.I. Jones, editors, *Proceedings of the* 6^{th} *International Workshop on Deontic Logic In Computer Science (*Δ*EON'02)*, pages 55–71, Londres, May 2002.

[42] L. Cholvy and C. Garion. Merging conflictual requirements with a majority approach. In *Proceedings of the International Workshop for High Assurance Systems (RHAS'02)*, Berlin, September 2002.

[43] L. Cholvy and C. Garion. Méthodes formelles pour l'ingénierie des exigences : fusion de points de vue par une approche majoritaire. In *15èmes Journées Internationales ICSSEA 2002 Génie Logiciel et Ingénierie des Systèmes et leurs Applications*, Paris, december 2002.

[44] L. Cholvy and A. Hunter. Merging requirements from different agents. *Knowledge Based Systems*. A paraître.

[45] L. Cholvy and A. Hunter. Information fusion in logic : A brief overview. In D.M. Gabbay, R. Kruse, A. Nonnegart, and H.J. Ohlbach, editors, *Proceedings of the European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty ECSQARU)*, pages 86–95, Bad Honnef, Germany, june 1997. Springer.

[46] P.R. Cohen and H.J. Levesque. Intention is choice with commitment. *Artificial Intelligence*, 42 :213–261, 1990.

[47] M. Dalal. Investigations into a theory of knowledge base revision : preliminary report. In *Proceedings of the National Conference on Artificial Intelligence (AAAI'88)*, pages 475–478, 1988.

[48] M. Dastani, J. Hulstijn, and L. van der Torre. BDI and QDT : a comparison based on classical decision theory. In *Proceedings of GTDT2001*, pages 16–26, AAAI Spring Symposium on Game-Theoretic and Decision-Theoretic Approaches to Agency (GTDT'2001), 2001. AAAI Press.

[49] L. Fariñas del Cerro and A. Herzig. A modal analysis of possibility theory. In P. Jorrand and J. Kelemen, editors, *Fundamentals of Artificial Intelligence Research, International Workshop FAIR'91*, volume 535 of *Lecture Notes in Computer Science*, pages 11–18. Springer, 1991.

[50] D. Dubois, J. Lang, and H. Prade. Possibilistic logic. Inconsistency in possibility knowledge bases - to live or not to live with it. In L. Zadeh and J. Kacpryzk, editors, *Fuzzy logic for the management of uncertainty*, pages 335–351. Wiley, 1992.

[51] D. Dubois, J. Lang, and H. Prade. Possibilistic logic. In D.M. Gabbay, C.J. Hogger, and J.A. Robinson, editors, *Handbook of logic in artificial intelligence and logic programming*, volume 3, pages 439–513. Clarendson Press, Oxford, 1994.

[52] S. Easterbrook. *Elicitation of Requirements from Multiple Perspectives*. PhD thesis, Department of Computing, Imperial College of Science, Technology and Medicine, University of London, London SW7 2BZ, June 1991.

[53] S. Easterbrook, A. Finkelstein, J. Kramer, and B. Nuseibeh. Coordinating distributed viewpoints : The anatomy of a consistency check. *International Journal on Concurrent Engineering : Research and Applications*, 2(3) :209–222, 1994.

[54] S. Easterbrook and B. Nuseibeh. Managing inconsistency in an evolving specification. In *Second IEEE International Symposium on Requirements Engineering (RE'95)*, pages 191–199. IEEE Press, April 1995.

[55] A. Finkelstein, D. Gabbay, A. Hunter, J. Kramer, and B. Nuseibeh. Inconsistency handling in multi-perspective specifications. In *IEEE Transactions on Software Engineering*, volume 20, pages 569–578, August 1994.

[56] A. Finkelstein, J. Kramer, B. Nuseibeh, L. Finkelstein, and M. Goedicke. Viewpoints : A framework for integrating multiple perspective in system development. *International Journal of Software Engineering and Knowledge Engineering, Special Issue on Trends and Future Research Directions in SE*, 2(1) :31–58, March 1992.

[57] D. Føllesdan and R. Hilpinen. Deontic logic : an introduction. In R. Hilpinen, editor, *Deontic logic : introductory and systematic readings*, pages 1–35. Dordrecht : Reidel, 1971.

[58] P. Gardenfors. *Knowledge in Flux : Modelling the Dynamics of Epistemic States*. MIT Press, 1988.

[59] C. Garion. Distributions des exigences : Un problème de calcul de buts individuels en fonction de buts collectifs. In M. Ayel and J.-M. Fouet, editors, *Actes des Cinquièmes Rencontres Nationales des Jeunes Chercheurs en Intelligence Artificielle*, pages 159–173, Lyon, September 2000.

[60] C. Garion. Une logique pour la fusion majoritaire de croyances. In M. Ayel and J.-M. Fouet, editors, *Actes des Cinquièmes Rencontres Nationales des Jeunes Chercheurs en Intelligence Artificielle*, pages 149–158, Lyon, September 2000.

[61] C. Garion. Representation and distribution of requirements with a logic of preferences. In *Journées FAC'2001 : Formalisation des activités concurrentes*, pages 13–20. ONERA-Toulouse, April 2001.

[62] M. Georgeff, B. Pell, M.E. Pollack, M. Tambe, and M. Wooldridge. The Belief-Desire-Intention model of agency. In J. Muller, M. Singh, and A. Rao, editors, *Intelligent Agents V*, volume 1365 of *Lecture Notes in Artificial Intelligence*. Springer Publishers, 1999.

[63] F. Giunchiglia and L. Serafini. Multilanguage hierarchical logics (or : how can we do without modal logic). *Artificial Intelligence*, 65 :1–42, 1994.

[64] B.J. Grosz, L. Hunsberger, and S. Kraus. Planning and acting together. *AI Magazine*, 20(4) :23–34, 1999.

[65] R.V. Guha. *Contexts : A Formalization and Some Applications*. PhD thesis, Standford University, 1991.

[66] B. Hansson. An analysis of some deontic logics. In R. Hilpinen, editor, *Deontic logic : introductory and systematic readings*, pages 121–147. Dordrecht : Reidel, 1971. (Originally published 1969 in *Noûs* 3 : 373-398).

[67] J. Hintikka. Impossible possible worlds vindicated. *Journal of Philosophical Logic*, 4 :475–484, 1975.

[68] J.F. Horty. Moral dilemmas and non-monotonic logic. *Journal of Philosphical Logic*, 23 :33–65, 1994.

[69] J.F. Horty and N. Belnap. The deliberative stit : a study of action, omission, ability and obligation. *Journal of Philosophical Logic*, 24 :583–644, 1995. Reprinted in *The Philosopher's Annual, Volume 18-1995*, Ridgeview Publishing Company, 1997.

[70] M.H. Huhns and M.P. Singh. A multiagent treatment of agenthood. *Applied Artificial Intelligence*, 13(1-2) :3–10, 1999.

[71] I.L. Humberstone. Inaccessible worlds. *Notre Dame Journal of Formal Logic*, 24(3) :346–352, 1983.

[72] A. Hunter and B. Nuseibeh. Analysing inconsistent specifications. In *3rd International Symposium on Requirements Engineering (RE97)*, pages 78–86, Annapolis, MD, USA, January 1997.

[73] A. Hunter and B. Nuseibeh. Managing inconsistent specifications : Reasoning, analysis and action. *Transactions on Software Engineering and Methodology*, 7(4) :335–367, October 1998.

[74] A.J.I. Jones and I. Pörn. Ideality, sub-ideality and deontic conditionals. *Synthese*, 65 :275–290, 1985.

[75] S. Kaci. *Connaissances et Préférences : Représentation et Fusion en Logique Possibiliste*. PhD thesis, Université Paul Sabatier, 2002.

[76] S. Konieczny. *Sur la logique du changement : Révision et Fusion de bases de connaissance*. PhD thesis, Université des Sciences et Technologies de Lille, November 1999.

[77] S. Konieczny. On the difference between merging knowledge bases and combining them. In *Proceedings of the Seventh International Conference on Principles of Knowledge Representation and Reasoning (KR'00)*, pages 135–144, Breckenridge, Colorado, USA, April 2000. Morgan Kaufmann.

[78] S. Konieczny and R. Pino-Pérez. On the logic of merging. In *Proceedings of the Sixth International Conferences on Principles of Knowledge Representation and Reasoning (KR'98)*, pages 488–498, Trento, Italy, June 1998. Morgan Kaufmann.

[79] S. Konieczny and R. Pino-Pérez. Merging with integrity constraints. In *Proceedings of the Fifth European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU'99)*, volume 1638 of *Lecture Notes in Artificial Intelligence*, pages 233–244. Springer-Verlag, July 1999.

[80] S. Konieczny and R. Pino-Pérez. On the frontier between arbitration and majority. In D. Fensel, F. Giunchiglia, D. McGuinness, and M.-A. Williams, editors, *Proceedings of the eight International Conference on Principles of Knowledge Representation and Reasoning (KR'02)*, pages 109–118, Toulouse, France, April 2002. Morgan-Kaufmann.

[81] S. Kraus. Beliefs, time and incomplete information in multiple encounter negociations among autonomous agents. *Annals of Mathematics and Artificial Intelligence*, 7(1) :109–156, 1996.

[82] S. Kraus. Negociation and cooperation in multi-agent environments. *Artificial Intelligence Journal - Special Issue on Economic Principles of Multi-Agent Systems*, 94(1-2) :79–98, 1997.

[83] C. Lafage. *Représentation de préférences en logiques, application à la décision de groupe*. PhD thesis, Université Paul Sabatier, Toulouse, June 2001.

[84] J. Lang. Possibilistic logic as a logical framework for min-max discrete optimisation problems and prioritized constraints. In P. Jorrand and J. Kelemen, editors, *Fundamentals of Artificial Intelligence Research, International Workshop FAIR'91*, volume 535 of *Lecture Notes in Computer Science*, pages 112–126. Springer, 1991.

[85] J. Lang. Conditional desires and utilities - an alternative logical approach to qualitative decision theory. In W. Wahlster, editor, *Proceedings of the Twelth European Conference on Artificial Intelligence ECAI 96*, pages 318–322. Jonh Wiley and Sons, Ltd., 1996.

[86] J. Lang, L. van der Torre, and E. Weydert. Utilitarian desires. *Autonomous Agents and Multi-Agent Systems*, 5(3) :329–363, 2002.

[87] D. Lewis. *Counterfactuals*. Blackwell, Oxford, 1973.

[88] D. Lewis. Semantic analyses for dyadic deontic logic. In S. Stendlund, editor, *Logical theory and semantic analysis*, pages 1–14. Reidel, Dordrecht, 1974.

[89] P. Liberatore and M. Schaerf. Arbitration : A commutative operator for belief revision. In *Proceedings of the Second World Conference on the Fundamentals of Artificial Intelligence*, pages 217–228, 1995.

[90] P. Liberatore and M. Schaerf. Arbitration (or how to merge knowmedge bases). *IEEE Transactions on Knowledge and Data Engineering*, 10(1) :76–90, 1998.

[91] J. Lin and A.O. Mendelzon. Knowledge base merging by majority. Technical Report M5S 1A4, Dept. of Computer Science, Univ. of Toronto, Toronto, 1994.

[92] J. Lin and A.O. Mendelzon. Merging databases under constraints. *International Joint Cooperative Information Systems*, 7(1) :55–76, 1998.

[93] L. Lindahl and J. Odelstad. Normative positions within an algebraic approach to normative systems. In J.F. Horty and A.J. Jones, editors, *Proceedings of the Sixth International Workshop on Deontic Logic in Computer Science (ΔEON'02)*, pages 149–180, London, May 2002.

[94] D. Makinson. On a fundamental problem of deontic logic. In P. McNamara and H. Prakken, editors, *Norms, logics and information systems : new studies in deontic logic and computer science*, volume 49 of *Frontiers in artificial intelligence and applications*, pages 29–53. IOS Press, Amsterdam, 1999.

[95] D. Makinson and L.W.N. van der Torre. Input/output logics. *Journal of Philosophical Logic*, 29 :383–408, 2000.

[96] D. Makinson and L.W.N. van der Torre. Constraints for input/output logics. *Journal of Philosophical Logic*, 30(2) :155–185, 2001.

[97] D. Makinson and L.W.N. van der Torre. Permission from an input/output perspective. In J.F. Horty and A.J. Jones, editors, *Proceedings of the Sixth International Workshop on Deontic Logic in Computer Science (ΔEON'02)*, pages 233–264, London, May 2002.

[98] J. McCarthy and S. Buvač. Formalizing context : Expanded notes. Technical Report STAN-CS-TN-94-13, Computer Science Departement - Standord University, 1994.

[99] L.T. McCarthy. Defeasible deontic reasoning. *Fundamenta Informaticae*, 21 :125–148, 1994.

[100] P. McNamara. Toward an integrated agential and aretaic framework. In R. Demolombe and R. Hilpinen, editors, *Proceedings of the Fifth International Workshop on Deontic Logic in Computer Science (ΔEON'00)*, pages 281–300, Toulouse, January 2000. ONERA-DGA.

[101] P. McNamara. A preliminary exploration of agential obligation as non-agential personal obligation plus agency. In J.F. Horty and A.J. Jones, editors, *Proceedings of the Sixth International Workshop on Deontic Logic in Computer Science (ΔEON'02)*, pages 203–232, London, May 2002.

[102] J.-J.Ch. Meyer. A different approach to deontic logic : deontic logic viewed as a variant of dynamic logic. *Notre Dame Journal of Formal Logic*, 21(1) :109–136, 1988.

[103] R. Montague. Syntactical treatment of modalities, with corollaries on reflexion principles and finite axiomatizability. *Acta Philosophica Fennica*, 16 :153–157, 1963.

[104] B. Nuseibeh. To be and not to be : On managing inconsistency in software development. In *8th IEEE International Workshop on Software Specification and Design (IWSSD-*)*, pages 164–169, March 1996.

[105] B. Nuseibeh and A. Finkelstein. Viewpoints : a vehicle for method and tool integration. In *Proceedings of the Fifth International Workshop on Computer-Aided Software Engineering (CASE'92)*, pages 50–60, Montreal, Canada, July 1992. IEEE CS Press.

[106] B. Nuseibeh, J. Kramer, and A. Finkelstein. A framework for expressing the relationships between multiple views in requirements specification. *IEEE Transactions on Software Engineering*, 20(10) :760–773, 1994.

[107] J. Pearl. System Z : A natural ordering of defaults with tractable applications to default reasoning. In M. Vardi, editor, *Proceedings of Theoretical Aspects of Reasoning about Knowledge*, pages 121–135, San Mateo, 1990. Morgan Kaufmann.

[108] J. Pearl. From conditional ought to qualitative decision theory. In *Proceedings of the Ninth Conference on Uncertainty in Artificial Intelligence (UAI'93)*, pages 12–20. Jonh Wiley and Sons, 1993.

[109] L. Perrussel. Expressing inter-perspective relationships : a logical approach. In *Proceedings of the second Asian-Pacific software engineering conference (APSEC'95)*, Brisbane, Australia, december 1995.

[110] L. Perrussel. Solving conflicts produces new contexts. In *Proceedings of the multiple perspectives in software development workshop - ACM SIGSOFT'96*, San Fransisco, CA, 1996. ACM Press.

[111] L. Perrussel. *Un Outillage Logique Pour l'Ingénierie des Exigences Multi-Points de Vue*. PhD thesis, Université de Toulouse I, 1998.

[112] L. Perrussel, P.J. Charrel, and B. Rothenburger. A formalism for inter-perspective relationships. In *Information modelling and knowledge bases VIII : principles and formal technics*, Hornbaek, Denmark, 1996.

[113] H. Prakken. Two approaches to the formalization of defeasible deontic reasoning. *Studia Logica*, 57 :73–90, 1996.

[114] H. Prakken and M. Sergot. Contrary-to-duty obligations. *Studia Logica*, 57(1) :91–115, 1996.

[115] H. Prakken and M. Sergot. Dyadic deontic logic and contrary-to-duty obligations. In D.N. Nute, editor, *Defeasible Deontic Logic*, pages 223–262. Synthese Library, 1997.

[116] A. Rao and M. Georgeff. Modeling rational agents within a bdi architecture. In *Proceedings of the Second International Conference on Konwledge Representation and Reasoning (KR'91)*, pages 473–484. Morgan Kaufmann, 1991.

[117] A. Rao and M. Georgeff. An abstract architecture for rational agents. In *Proceedings of the Third International Conference on Knowledge Representation and Reasoning (KR'92)*, pages 439–449. Morgan Kaufmann, 1992.

[118] R. Reiter. On closed world data bases. In H. Gallaire and J. Minker, editors, *Logic and Data Bases*, pages 55–76. Plenum Press, 1978, 1978.

[119] R. Reiter. A logic for default reasoning. *Artificial Intelligence*, 13(1-2) :81–132, 1980.

[120] P. Z. Revesz. On the semantics of arbitration. *International Journal of Algebra and Computation*, 7(2) :133–160, 1997.

[121] P.Z. Revesz. On the semantics of theory change : arbitration between old and new information. In *Proceedings of the* 12^{th} *ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Databases*, pages 71–82, 1993.

[122] F. Santos and J. Carmo. Indirect action, influence and responsability. In M. Brown and J. Carmo, editors, *Deontic logic, agency and normative systems (Proceedings of the Third International Workshop on Deontic Logic)*, Workshops in Computing Series, pages 194–215. Springer, 1996.

[123] M.P. Singh. *Multiagent Systems. A Theoretical Framework for Intentions, Know-How and Communications*, volume 799 of *Lecture Notes in Artificial Intelligence*. Spinger-Verlag, 1994.

[124] M.P. Singh. Know-how. In A.S. Rao and M.J. Wooldridge, editors, *Foundations of Rational Agency*, Applied Logic Series, pages 105–132. Kluwer, 1999.

[125] M.P. Singh, A.S Rao, and M.P. Georgeff. Formal methods in DAI : Logic-based representation and reasoning. In G. Weiss, editor, *Multiagent Systems : A Modern Approach to Distributed Artificial Intelligence*, chapter 8, pages 331–376. MIT Press, 1999.

[126] W. Spohn. An analysis of Hansson's dyadic deontic logic. *Journal of Philosophical Logic*, 4 :237–252, 1975.

[127] V.S. Subrahmanian. Amalgamating knowledge bases. *ACM Transactions on Database Systems*, 19(2) :291–331, 1994.

[128] V.S Subrahmanian, S. Adali, A. Brink, R. Emery, J.J. Lu, A. Rajput, T.J. Rogers, R. Ross, and C. Ward. Hermes : Heterogeneous reasoning and mediator system.

[129] S.-W. Tan and J. Pearl. Specification and evaluation of preferences under uncertainty. In J. Doyle and al, editors, *Proceedings of the 4th International Conference on Principles of Knowledge Representation and Reasoning (KR'94)*, pages 530–539, Bonn, Germany, 1994. Morgan Kaufmann.

[130] Y.-H. Tan and L.W.N. van der Torre. Multi preference semantics for a defeasible deontic logic. In H. Prakken, A.J. Muntjewerff, and A. Soeteman, editors, *Legal knowledge based systems. The relation with legal theory*, pages 115–126. Koninklijke Vermande BV, Lelystad, 1994.

[131] Y.-H. Tan and L.W.N. van der Torre. Contextual deontic logic. In *Formal Models of Agents*, volume 1760 of *Lecture Notes in Artificial Intelligence*, pages 240–251. Springer, 1997.

[132] L. van der Torre and Y. Tan. The many faces of defeasibility in defeasible deontic logic. In D. Nute, editor, *Defeasible Deontic Logic*, volume 263 of *Synthese Library*, pages 79–121. Kluwer, 1997.

[133] L. van der Torre and Y. Tan. An update semantics for prima facie obligations. In H. Prade, editor, *Proceedings of the Thirteenth european conference on artificial intelligence (ECAI'98)*, pages 38–42, 1998.

[134] L. van der Torre and Y. Tan. Contrary-to-duty reasoning with preference-based dyadic obligations. *Annals of Mathematics and Artificial Intelligence*, 27(1-4) :49–78, 1999.

[135] L. van der Torre and Y. Tan. An update semantics for defeasible obligations. In K. Laskey and H. Prade, editors, *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence (UAI'99)*, pages 631–628, 1999.

[136] L.W.N van der Torre and E. Weydert. Parameters for utilitarian desires in a qualitative decision theory. *Applied Intelligence*, 14 :285–301, 2001.

[137] J. von Neumann and O. Morgenstern. *Theory of Games and Economic Behaviour*. Princeton University Press, 1947.

[138] G.H. von Wright. Deontic logic. *Mind*, 60 :1–15, 1951.

[139] M.J. Wooldridge and N.R. Jennings. Agent theories, architectures, and languages : a survey. In M.J. Wooldridge and N.R. Jennings, editors, *Intelligent Agents : proceedings of the ECAI-94 Workshop on Agent Theories, Architectures, and Languages*, volume 890 of *Lecture Notes in Artificial Intelligence*, pages 1–25. Springer-Verlag, 1994.

APPORTS DE LA LOGIQUE MATHÉMATIQUE EN INGÉNIERIE DES EXIGENCES

Résumé : Cette thèse s'intéresse à l'ingénierie des exigences (IE), qui caractérise le processus qui conduit à la construction d'un ensemble cohérent de spécifications portant sur un produit à construire. Nous avons identifié trois phases distinctes dans le processus d'IE : modélisation des exigences, gestion des incohérences et distribution des exigences.
Dans la phase de modélisation, nous nous sommes appuyés sur une logique de préférences, CO, qui nous a permis d'exprimer les exigences de chaque agent de façon ordonnée, mais également des contraintes du domaine et des phrases normatives complexes. Nous avons alors pu définir une notion de cohérence entre ces trois notions.
Pour la résolution des éventuels conflits entre exigences émises par différents agents, nous avons développé MF, une logique modale permettant de raisonner sur le contenu de bases de croyances obtenues par fusion majoritaire de plusieurs bases primitives ainsi qu'un démonstrateur automatique en PROLOG pour MF. Dans un second temps, nous avons montré que notre approche permettait de raisonner sur des ensembles d'exigences ordonnées ou non.
Enfin, nous avons proposé d'inclure dans le processus d'IE une phase de distribution des exigences à un ensemble d'agents exécutants. Après avoir proposé un modèle d'agent simple, nous avons montré que ce modèle permettait de déterminer les buts d'un agent. Nous avons étendu cette approche à un groupe d'agents, puis nous avons défini un modèle de distribution comportant une entité centrale connaissant les engagements des agents.

Mots clés : Intelligence Artificielle - logique - logique déontique - fusion de bases de croyances - représentation de préférences - théorie de la décision qualitative - systèmes multi-agents

CONTRIBUTIONS OF MATHEMATICAL LOGIC TO REQUIREMENTS ENGINEERING

Abstract : This thesis deals with requirements engineering (RE). RE characterizes the process leading to a consistent set of specifications about some product. We have identified three distinct phases in RE process : requirements modelling, inconsistency management and requirements distribution.
In the modelling phase, we have used CO, a logic of preferences, which has allowed us to express each agent's requirements in an ordonned way, but also domain constraints and complex normative sentences. We have then defined the notion of consistency between those three notions.
Concerning the possible conflicts between requirements emitted by different agents, we have developped MF, a modal logic allowing to reason on belief bases obtained by majority merging. We have also developped a PROLOG automatic prover for MF. We have then shown that our approach allows to reason on ordonned or unordonned requirements sets.
Finally, we have proposed to include in the RE process a distribution phase. The requirements are distributed among a set of executive agents. We have defined a simple model of agency from which we can derive the agent's goals. We have then extended this approach to multiagents systems and defined a distribution model based on a central entity controlling the distribution process.

Keywords : Artificial Intelligence - logics - deontic logics - belief bases merging - preferences representation - qualitative decision theory - multiagent systems

10, avenue Édouard-Belin - BP 4032 - 31055 Toulouse cedex 4 - France
Tél : 33 (0)5 62 17 80 80 - Fax : 33 0(5) 62 17 83 30 - http://www.supaero.fr
Établissement public à caractère administratif sous tutelle du Ministère de la Défense