Journées d'Intelligence Artificielle Fondamentale 2007 (IAF'07)

2-3 Juillet 2007 - Grenoble

Actes

Ces journées sont la principale animation du thème Intelligence Artificielle Fondamentale (thème 1) du GDR Information-Interaction-Intelligence. Ces journées ont été organisées dans le cadre de la plate-forme AFIA 2007.

Les journées étaient composées d'exposés de synthèse et de contributions sélectionnées. Ce document rassemble l'ensemble des contributions sélectionnées. Sur la page des actes électroniques des journées (http://www.cril.fr/~konieczny/IAF07/) sont également disponibles les transparents des exposés de synthèse et contributions sélectionnées. La première demi-journée était commune avec l'atelier Représentation et Raisonnement sur le Temps et l'Espace.

Contributions sélectionnées

Exposés de synthèse

- **Formalisations de la négociation**
  Leila Amgoud, Yann Chevaleyre, Sébastien Konieczny, Jérôme Lang, Nicolas Maudet
- **Résolution pratique de problèmes : SAT et Answer Set Programming**
  Pascal Nicolas, Laurent Simon
- **Des réseaux de contraintes valués aux problèmes de décision séquentielle**
  Cédric Pralet, Thomas Schiex

Comité de Programme

- Laurence Cholvy (ONERA, Toulouse)
- Philippe Dague (LRI, Paris)
- Didier Dubois (IRIT, Toulouse)
- Laurent Garcia (LERIA, Angers)
- Robert Jeansoulin (IGM, Marne la Vallée)
- Philippe Jégou (LSIS, Marseille)
- Narendra Jussien (LINA, Nantes)
- Sébastien Konieczny (CRIL, Lens)
- Pierre Marquis (CRIL, Lens)
- Pascal Nicolas (LERIA, Angers)
- Odile Papini (LSIS, Marseille)
- Henri Prade (IRIT, Toulouse)
- Lakhdar Sais (CRIL, Lens)
- Pierre Siegel (LSIS, Marseille)

Réseaux causaux possibilistes pour le traitement des interventions

Salem Benferhat[1], Salma Smaoui[1]

CRIL,Université d'Artois,
Rue Jean Souvraz SP 18, 62300 Lens, France
{benferhat,smaoui}@cril.univ-artois.fr

Résumé : Ce article présente deux principales contributions pour le développement des réseaux causaux possibilistes. La première concerne la représentation des interventions dans les réseaux possibilistes. Nous proposons une contrepartie de l'opérateur "DO", récemment introduit par Pearl, dans le cadre possibiliste. Nous montrons par la suite que les interventions peuvent être représentées de manières différentes mais équivalentes dans les réseaux causaux possibilistes. La deuxième contribution est un nouvel algorithme de propagation permettant de traiter aussi bien les observations que les interventions. Nous montrons que le coût supplémentaire pour le traitement des interventions est négligeable et que notre algorithme est plus approprié lorsqu'il s'agit de traiter des séquences d'observations et d'interventions.
Mots-clés : Réseaux possibilistes, causalité, propagation, inférence possibiliste, interventions.

1 Introduction

Les réseaux probabilistes bayésiens Pearl (1988); Jensen (1996); Lauritzen & Spiegelhalter (1988) représentent des outils de calcul puissants permettant d'identifier les interactions entre les variables à partir de données observées. Récemment, Pearl (2000) a proposé des approches basées sur la théorie de probabilité en utilisant des graphes causaux afin de formaliser la notion d'interventions. Du point de vue de la représentation, les interventions et les observations sont distinguées en utilisant le concept de l'opérateur "do" introduit par Goldszmidt & Pearl (1992); Pearl (2000). Du point de vue du raisonnement, le traitement des interventions consiste à altérer le graphe en excluant toutes les causes directes associées à la variable d'intérêt autres que les interventions en maintenant intact le reste du graphe. Les effets de telles interventions sur le reste des variables sont calculés en appliquant le conditionnement sur le graphe altéré.

L'opérateur "do" a été initialement proposé dans Goldszmidt & Pearl (1992) dans le cadre des fonctions ordinales conditionnelles de Spohn (1988b,a). Les révisions et les mises à jour sont unifiées à travers le conditionnement sur les actions afin de permettre le raisonnement causal. Les fonctions ordinales conditionnelles de Spohn représentent un modèle "qualitatif" pour la représentation de l'incertitude et possèdent des liens très étroits aussi bien avec les probabilités infinitésimales qu'avec la théorie des possibilités (voir Dubois & Prade (1991)).

Cet article se focalise sur le développement des réseaux causaux possibilistes dans le but de traiter aussi bien les interventions que les observations. Une intervention est le résultat d'une action externe (non prévue dans le système) qui force certaines variables à prendre des valeurs données. Une observation est une nouvelle information sur la valeur de certaines variables qui complète nos croyances sur le monde réel. Une observation concerne un monde statique alors qu'une intervention concerne un monde dynamique. Cette différence ressemble a la différence entre révision et mise a jour qui est très peu étudiée dans le cadre des représentations graphiques. Deux contributions principales y sont présentées :
- Les fondements théoriques des réseaux causaux possibilistes. Plus précisément, cet article permet de définir les réseaux causaux possibilistes et introduit l'opérateur "do" à la théorie des possibilités. Nous montrons que les différentes, mais équivalentes, représentations des interventions dans les réseaux probabilistes restent valables pour le cadre de la théorie des possibilités.
- Un nouvel algorithme de propagation permettant de traiter les interventions et les observations. Nous commençons par montrer que l'adaptation directe des algorithmes de propagation dans les réseaux probabilistes n'est pas appropriée pour le traitement incrémental de séquences de nouvelles interventions et observations. En effet, pour les réseaux à connexions multiples, lorsqu'il s'agit d'interventions, répondre aux requêtes n'est plus immédiat. Ceci requiert $O(|D|^{|u| \times |r|})$ calculs où $|D|$ dénote la taille du domaine de la variable concernée par des interventions, $|u|$ est le nombre des instances de parents et $|r|$ est le nombre des différentes interventions. Si on impose l'efficacité dans le traitement des requêtes alors une solution possible consiste à utiliser les graphes augmentés (les interventions sont dans ce cas considérées comme étant des observations au niveau du graphe augmenté). Cette solution n'est pas satisfaisante dans le cadre des réseaux probabilistes du moment que, pour les graphes à connexions multiples par exemple, le traitement incrémental de séquences d'observations et d'interventions est impossible (à moins de répéter à chaque intervention une étape d'initialisation du graphe).

Cet article tire profit des différents avantages qu'offrent les propriétés des réseaux possibilistes et propose un nouvel algorithme où d'une part, l'arbre de jonction associé au réseau causal possibiliste est construit une seule fois, d'autre part, la réponse aux requêtes est réalisée en un temps linéaire.

Le reste de cet article est organisé comme suit : la section suivante présente un bref rappel de la théorie des possibilités et des réseaux possibilistes. La contrepartie possibiliste de l'opérateur do est proposée dans la troisième section. Finalement, nous présentons notre nouvel algorithme permettant de traiter les séquences non-simultanées d'interventions. La dernière section conclut cet article.

2 La théorie des possibilités

Cette section présente un bref rappel de la théorie des possibilités ; pour plus de détails voir Dubois & Prade (1988).
Soit $V = \{A_1, A_2, ..., A_n\}$ un ensemble de variables. D_{A_i} dénote le domaine fini associé à la variable A_i. a_i dénote une instance quelconque de la variable A_i. X, Y, Z.. représentent des ensembles de variables. x est un élément du produit cartésien $\times_{A_i \in X} D_{A_i}$ qui est un sous-ensemble de $\Omega = \times_{A_i \in V} D_{A_i}$ l'univers de discours. ω, un élément de Ω, est appelée *interprétation* ou *évènement*. Il est noté par un tuple $(a_1, ..., a_n)$, où les a_i sont respectivement des instances des A_i. $\omega[A_i]$ correspond à la valeur associée à la variable A_i dans l'interprétation ω.

2.1 Mesures de possibilité et distributions de possibilité

Une distribution de possibilité π est une application de Ω vers l'intervalle $[0, 1]$. Le degré de possibilité $\pi(\omega)$ représente la compatibilité de ω avec l'information disponible. Par convention, $\pi(\omega) = 1$ signifie que ω est totalement possible, et $\pi(\omega) = 0$ signifie que ω est impossible. Lorsque $\pi(\omega) > \pi(\omega')$, ω est préférée à ω' pour être l'état réel du monde. La distribution de possibilité π est dite normalisée s'il existe au moins une interprétation cohérente avec les informations disponibles, c'est-à-dire : $\exists \omega \in \Omega,\ \pi(\omega) = 1$. Une mesure de possibilité Π est une fonction qui associe à chaque $\varphi \subseteq \Omega$ un poids dans l'intervalle $[0, 1]$. Π peut être simplement obtenue à partir de π comme suit : $\Pi(\varphi) = \max\{\pi(\omega) : \omega \in \varphi\}$.
Le conditionnement Dubois & Prade (1988) consiste à augmenter proportionnellement les éléments cohérents avec x :

$$\pi(\omega \mid x) = \begin{cases} \frac{\pi(\omega)}{\Pi(x)} & si\ \omega[X] = x \\ 0 & sinon. \end{cases} \tag{1}$$

2.2 Les réseaux possibilistes

Les réseaux possibilistes Fonck (1994); Borgelt *et al.* (1998); Ben Amor *et al.* (2003), notés par G, sont des graphes acycliques orientés (DAG). Les nœuds correspondent aux variables et les arcs représentent les relations entre les variables. Un nœud A_j est dit un parent de A_i s'il existe un lien du nœud A_j vers le nœud A_i. Les parents de A_i sont notés par U_{A_i} ou bien U_i.
Il existe deux types de réseaux possibilistes (selon le conditionnement utilisé). Cet article se focalise sur les réseaux possibilistes où le conditionnement est donné par l'équation 1.
L'incertitude au niveau des nœuds est représentée par des distributions de possibilité locale. Plus précisément, pour chaque variable A_i et pour chaque u_i un élément du produit cartésien des domaines des variables qui sont les parents de A_i, nous associons un degré de possibilité $\Pi(a_i \mid u_i)$ pour tout $a_i \in D_{A_i}$, tel que $\max_{a_i \in D_{A_i}} \pi(a_i | u_i) = 1$.
Les réseaux possibilistes constituent des représentations compactes des distributions de possibilité. Plus précisément, les distributions de possibilité jointes associées aux réseaux possibilistes sont calculées en utilisant la règle de chainage basée sur le produit

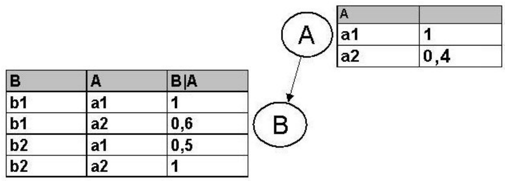

FIG. 1 – Exemple de réseau causal possibiliste G

qui est similaire à la règle de chainage probabiliste, à savoir :

$$\pi_{\Pi G}(a_1, ..., a_n) = \prod_{i=1,...,n} \Pi(a_i \mid u_i) \tag{2}$$

Exemple 1
La figure 1 donne un exemple des réseaux possibilistes. La distribution de possibilité jointe $\pi_G(AB) = \pi_G(A).\pi_G(B|A)$ associée à G est donnée par la table 1.

A	B	$\pi_G(AB)$
a_1	b_1	1
a_1	b_2	0.5
a_2	b_1	0.24
a_2	b_2	0.4

TAB. 1 – La distribution de possibilité jointe $\pi_G(AB)$

3 Les réseaux causaux possibilistes

La capacité de prédire l'effet des interventions dans les réseaux causaux requiert des conditions plus fortes lors de la construction de ces réseaux. Un réseau causal possibiliste est un réseau possibiliste où les arcs orientés dans le graphe représentent des relations causales entre les évènements. Les arcs suivent le sens du processus causal. Intuitivement, l'ensemble de parents U_i de A_i représente la totalité des causes directes de A_i. Les arcs indiquent seulement que la variable est causalement pertinente par rapport aux autres, mais n'indique rien concernant la manière avec laquelle elle les influencerait.

La relation entre l'information causale et l'information probabiliste a fait l'objet de plusieurs recherches (notamment Pearl (2000)). Notre travail est parti de notre conviction que les mêmes résultats tiennent aussi pour les autres théories d'incertitude du moment que la transformation d'un réseau probabiliste lui permettant le traitement de la causalité concerne essentiellement la structure graphique. Dans cet article, nous nous focalisons sur les interprétations possibilistes des relations causales.

Les interventions sont traitées comme des modalités sur les variables. Une simple intervention, dite "atomique", est une intervention qui force une variable unique A_i à prendre une valeur a'_i. Cette intervention sur A_i est notée $do(A_i = a'_i)$ ou $do(a'_i)$. $do(a'_i)$ est considérée comme étant une intervention externe qui n'altère qu'un seul mécanisme (parent-enfant) en laissant intact les autres mécanismes.

La section suivante propose un modèle possibiliste permettant de représenter les interventions en utilisant les réseaux causaux possibilistes.

3.1 L'intervention comme une négation de toutes les autres causes directes

Pearl et Verma (Verma & Pearl (1990)) interprètent les relations causales parent-enfant dans un DAG comme une fonction déterministe $a'_i = f_i(u_i, \gamma_i), i = 1, ..., n$ où u_i sont des parents de la variables A_i dans le graphe causal. $\gamma_i, 1 \leq i \leq n$ sont des perturbations indépendantes et sont des instances de l'ensemble de variables non mesurées Γ .
L'effet d'une intervention, notée $do(A_i = a'_i)$ ou $do(a'_i)$, sur Y (un sous-ensemble de V) est déduit du modèle en supprimant les fonctions définissant A_i et en remplaçant $A_i = a'_i$ dans le reste des fonctions.

Dans les modèles graphiques, les interventions sur une variable A_i expriment aussi que nos croyances (exprimées dans un cadre incertain, ici la théorie des possibilités) sur les parents U_i de A_i doivent être affectées. Ceci est représenté par la suppression (aussi appelé mutilation) des liens entre U_i et A_i. Le reste du graphe reste intact. Le graphe mutilé résultant est noté G_m tel que $\pi(\omega|do(a'_i)) = \pi_{G_m}(\omega|a'_i)$, où π_{G_m} correspond à la distribution de possibilité associée au graphe mutilé G_m.
Le calcul de l'effet de l'intervention $do(a'_i)$ consiste à transformer la distribution $\pi(\omega)$ en une autre distribution de possibilité $\pi_{a'_i}(\omega) = \pi(\omega|do(a'_i))$. Formellement,

$$\forall\omega, \pi_{a'_i}(\omega) = \pi(\omega|do(a'_i)) = \pi_{G_m}(\omega|a'_i) \tag{3}$$

En mutilant le graphe, toutes les autres causes directes (les parents) autres que l'action deviennent indépendantes de la variable d'intérêt.

De plus, l'évènement qui associe à la variable A_i la valeur a'_i après la réalisation de l'intervention $do(a'_i)$ devient certain. Formellement, $\pi_{G_m}(a'_i) = 1$ et $\forall a_i \in D_{A_i} : a_i \neq a'_i, \pi_{G_m}(a_i) = 0$. L'effet d'une telle intervention est donc donné comme suit, $\forall\omega$:

$$\pi(\omega|do(a'_i)) = \begin{cases} \prod_{j\neq i} \pi(a_j|U_j(\omega)) & si\ \ \omega[A_i] = a'_i \\ 0 & sinon \end{cases} \tag{4}$$

où $U_j(\omega) = u_j$ correspond à la valeur que ω associe aux parents de a_j.

Exemple 2
En considérant le réseau causal possibiliste G présenté dans l'exemple 1, le graphe mutilé G_m obtenu après l'intervention $do(B = b1) = do(b1)$ est donné par la figure 2. L'effet de cette intervention $do(B = b_1)$ sur la distribution jointe $\pi(AB)$ associée au graphe dans l'exemple 1 est présenté dans la table 2.

La forme de l'équation (4) est intéressante puisqu'elle permet de calculer l'effet d'une intervention $do(a'_i)$ sur un ensemble de variable Y disjoint de $\{A_i\} \cup U_i$:

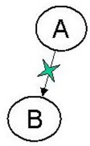

FIG. 2 – Le graphe mutilé G_m

A	B	$\pi(AB\|do(b_1))$
a_1	b_1	1
a_1	b_2	0
a_2	b_1	0.4
a_2	b_2	0

TAB. 2 – La distribution de possibilité jointe $\pi(AB|do(b_1))$

Proposition 1
Soit Y un ensemble de variables disjoint de $\{A_i \cup U_i\}$ où A_i est une variable dans V manipulée par une intervention $do(a'_i)$ et U_i est l'ensemble des causes directes A_i :

$$\pi(y|do(a'_i)) = \max_{u_i} \pi(y|a'_i, u_i).\pi(u_i) \tag{5}$$

En effet, il s'agit de la marginalisation sur les parents de A_i de l'expression équivalente à 4 : $\pi(a_1, ..., a_n|do(a'_i)) = \pi(a_1, ..., a_n|a'_i, u_i).\pi(u_i)$. La proposition 1 est la contre-partie possibiliste de la proposition de (Pearl (2000)). Ce résultat n'est pas surprenant étant donnée la similarité entre les réseaux causaux possibilistes basés sur le produit et le cas particulier des réseaux probabilistes où les probabilités conditionnelles sont soit proches de 1 soit proches de 0.

3.2 Ajout de nœuds supplémentaires

Une approche alternative mais équivalente Pearl (1993) consiste à considérer l'intervention comme étant une nouvelle variable dans le système. Cette sous section montre que cette approche alternative est aussi valable dans la théorie des possibilités. L'effet d'une intervention est calculé en appliquant le conditionnement sur le graphe altéré (augmenté). Cette altération consiste à ajouter un nœud parent DO_i à chaque variable A_i susceptible d'être concernée par une intervention. Le graphe résultant est dit augmenté. La variable DO_i prend ses valeurs dans $\{\{do_{a'_i} : \forall a'_i \in D_{A_i}\}, do_{i-noact}\}$. La valeur "$do_{i-noact}$" (ou "$do_{A_i-noact}$") signifie qu'aucune intervention n'est appliquée sur A_i. Les valeurs $do_{a'_i}$ signifient que la variable A_i est forcée à prendre la valeur a'_i. Le graphe augmenté est noté G_a. L'ensemble des parents U_i est augmenté de la variable DO_i ($U'_i = U_i \cup DO_i$).
La nouvelle distribution de possibilité locale au niveau de la variable A_i après augmentation du graphe est donnée par :

$$\pi(a_i|u'_i) = \begin{cases} \pi(a_i|u_i) & si \;\; DO_i = do_{i-noact} \\ 1 & si \;\; a_i = a'_i \\ 0 & si \;\; a_i \neq a'_i \end{cases} \tag{6}$$

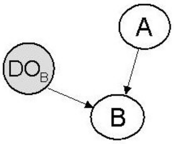

FIG. 3 – Le graphe augmenté G_a

Comme c'est le cas dans les réseaux probabilistes, les deux approches d'interprétation des interventions (mutilation et augmentation du graphe) sont aussi équivalentes dans le cadre de la théorie des possibilités. Plus précisément,

Proposition 2
Soient G un réseau causal possibiliste et G_m (resp. G_a) le graphe mutilé (resp. augmenté) obtenu à partir de G après l'application de l'intervention $do(a'_i)$ sur la variable A_i dans G. Nous avons,$\forall \omega,\ \forall a'_i \in D_{A_i}$,

$$\pi_{G_a}(\omega|DO_i = do_{a'_i}) = \pi_{G_m}(\omega|A_i = a'_i) = \pi(\omega|do(a'_i))$$

Cette approche a l'avantage de s'appliquer à tout type d'intervention. En ajoutant un lien $DO_i \to A_i$ à chaque nœud (sur lequel une intervention est possible) dans le graphe, il devient possible de construire une distribution de possibilité contenant des informations sur plusieurs types d'interventions.

Exemple 3
Le graphe augmenté G_a obtenu après l'intervention $do(b_1)$ à partir du graphe G introduit dans l'exemple 1 est donné par la figure 3.
La distribution de possibilité locale $\pi(B|A, DO_B)$ au niveau de B est donnée en utilisant l'équation (6). Par exemple, $\pi(b_1|a_2, do_{B-noact}) = \pi(b_1|a_2) = 0.6$ et $\pi(b_1|a_2, do_{b_2}) = 0$.

4 Propagation dans les réseaux causaux possibilistes

Dans la section précédente, nous avons montré que l'interprétation probabiliste des interventions peut facilement être adaptée au cadre de la théorie des possibilités. Cette section introduit un nouvel algorithme permettant le raisonnement à partir de réseaux causaux possibilistes. Notre algorithme est plus efficace qu'une simple adaptation de l'algorithme probabiliste. Par ailleurs, il permet le traitement de séquences non simultanées d'interventions et d'observations.

Le calcul de l'effet de séquences d'interventions et d'observations est réalisé :
- soit en généralisant la formule explicite (5) pour le traitement des ensembles d'observations $E \subseteq V$ (voir l'équation 1) et d'interventions $F \subseteq V$.
- soit en ajoutant un nœud parent à chaque variable dans F (augmenter le graphe) et appliquer par la suite le conditionnement possibiliste standard où les interventions sont traitées comme étant des observations au niveau des nœuds ajoutés.

La généralisation de (5) est donnée comme corollaire de la proposition 1.

Corollaire 1
Soit $E = e$ une observation. Soit F un ensemble de variables affectées par des interventions et f un élément de $\times_{A_i \in F} D_{A_i}$. Soit $U_F = \bigcup_{A_i \in F} U_i$, où U_i dénote l'ensemble de parents de A_i. Nous notons par u_F un élément de $\times_{A_i \in U_F} D_{A_i}$. L'effet de l'ensemble des interventions notées $do(f)$ et des observations e sur le reste des variables $A_j = a_j$ avec $A_j \notin E \cup F \cup U_F$ est donné par :

$$\Pi(a_j|e, do(f)) = \max_{u_F} \pi(a_j|e, f, u_F).\pi(u_F|e) \tag{7}$$

Ainsi, afin de calculer $\Pi(a_j|e, do(f))$ il suffit de calculer pour chaque u_F l'expression $\pi(a_j|e, f, u_F).\pi(u_F|e)$ et considérer par la suite le maximum des résultats obtenus. L'expression $\pi(a_j|e, f, u_F).\pi(u_F|e)$ est obtenue en utilisant une adaptation directe des algorithmes de propagation probabilistes puisqu'il s'agit simplement du conditionnement. Manifestement, cette solution n'est pas satisfaisante à moins que le nombre d'interventions est faible. En effet, utiliser telle équation semble inadéquat notamment pour le traitement de variables avec un nombre important de parents. Cette opération requiert $O(|D|^{|u| \times |r|})$ où $|D|$ correspond à la taille du domaine de la variable concernée par des interventions, $|u|$ est le nombre des instances de parents et $|r|$ représente le nombre d'interventions.

Un autre moyen permettant de calculer les effets des interventions consiste à mutiler le graphe initial. Cette approche est inadéquate notamment pour les graphes à connexions multiples qui impliquent une transformation graphique vers des arbres de jonction (à partir desquels une inférence efficace est effectuée). En effet, soit G un réseau possibiliste à connexions multiples. Supposons qu'une intervention $do(a'_i)$ est appliquée sur une variable A_i. Le graphe est alors mutilé et tous les liens vers A_i sont supprimés. L'arbre de jonction est formé après cette mutilation. Supposons maintenant qu'une autre intervention $do(a'_j)$ est appliquée par la suite sur A_j tel que $j \neq i$. En mutilant le graphe (déjà mutilé une fois), l'arbre de jonction résultant n'est pas le même et doit être reconstruit. Le processus de la construction de l'arbre de jonction est coûteux. Ce même processus doit être répété pour chaque modification au niveau du réseau.
Dans ce qui suit, nous présentons un nouvel algorithme de propagation utilisant les graphes augmentés permettant de traiter incrémentalement les séquences d'interventions et d'observations.

4.1 Un nouvel algorithme pour les réseaux causaux possibilistes

Notre objectif consiste à proposer un nouvel algorithme tel que :
– les réponses aux requêtes sont réalisées en un temps linéaire.
– une seule transformation du graphe augmenté initial à un arbre de jonction est requise.
– les observations et les interventions sont traitées incrémentalement (c.à.d. l'arbre de jonction est mis à jour incrémentalement sans avoir à le réinitialiser).

L'idée principale de notre algorithme est la capacité d'exprimer par défaut qu'il n'y a pas d'intervention au niveau du graphe augmenté et permettre par la suite la remise en question de cet état en mettant à jour les distributions de possibilité dans le cas où quelques interventions se produisent. C'est-à-dire, il s'agit d'associer des distributions

de possibilité aux nœuds ajoutés permettant d'exprimer qu'il n'existe pas d'intervention par défaut. Pour les réseaux probabilistes, il n'existe pas de distribution remplissant cette exigence à moins de proposer de réinitialiser l'arbre de jonction pour chaque observation ou intervention. En effet, soit BN un réseau bayésien et soit F un ensemble de variables dans G susceptible d'être affectées par des interventions. A chaque nœud A_i dans F nous associons un nœud parent DO_i. Le graphe résultant est noté BN_a.
Intuitivement, on s'attend à ce que la réalisation de l'inférence sans interventions et sans observations sur le graphe augmenté BN_a produise des distributions de probabilité (P_{BN_a}) équivalente à celle associée au graphe initial (P_{BN}) sur l'ensemble des variables initiales. Or, pour chaque distribution de probabilité associée aux nœuds ajoutés DO_i, y compris la distribution de probabilité uniforme, la distribution a posteriori (résultante) est différente de la distribution initiale à l'exception de la distribution qui associe le degré 1 aux valeurs "$do_{i-noact}$" pour chaque nœud DO_i et le degré 0 aux valeurs restantes. Plus formellement, $\forall \omega, \forall i : A_i \in F$,

$$P_{BN}(\omega) = P_{BN_a}(\omega) \text{ iff } P_{BN_a}(do_{i-noact}) = 1$$

De telles distributions (le degré 1 à $do_{i-noact}$) excluent toute intervention future. Autrement dit, associer le degré 0 aux autres valeurs que "$do_{i-noact}$" signifie qu'avoir des interventions sur A_i devient un évènement impossible.
En théorie des possibilités, la situation est différente ce qui nous a motivé à proposer notre nouvel algorithme.

Les distributions de possibilité locales pour les variables DO_i

La distribution de possibilité locale permettant d'exprimer "pas d'intervention" par défaut sans exclure la possibilité d'interventions futures est définie dans ce qui suit.

Définition 1
Soient G un réseau causal et G_a le graphe augmenté obtenu à partir de G en ajoutant un nœud parent DO_i pour chaque variable A_i dans F (l'ensemble de variables concernées par les interventions). Soit $A_i \in F$ un nœud susceptible d'être concernée par une intervention. La distribution de possibilité a priori (au départ) au niveau d'un nœud ajouté DO_i est définie par :
- $\pi_{G_a}(DO_i = do_{i-noact}) \leftarrow 1$,
- $\forall a_i \in D_{A_i}, \pi_{G_a}(DO_i = do_{a_i}) \leftarrow \epsilon$,
  où ϵ est un infinitesimal (en effet, ϵ doit être tel que $\epsilon \leq \min_{\omega \in \Omega} \pi_G(\omega)$),

où π_G (resp. π_{G_a}) est la distribution de possibilité associée au graphe G (resp. G_a).

En effet, en associant le degré de possibilité 1 à la valeur "$do_{i-noact}$" pour chaque nœud ajouté DO_i, les évènements $\{DO_i = do_{i-noact} : A_i \in F\}$ sont acceptés par défaut comme étant des situations normales. En associant le degré ϵ à $\pi(DO_i = do_{a_i})$ pour chaque instance $a_i \in D_{A_i} : A_i \in F$, les évènements $\{DO_i = do_{a_i} : A_i \in F\}$ sont considérés les moins normaux et les moins préférés dans le graphe évitant ainsi d'influencer nos connaissances initiales sur le reste des variables (les variables initiales).

La proposition suivante montre que tant qu'il n'y a ni observation ni intervention, les distributions jointes associées aussi bien pour le graphe initial que pour le graphe augmenté sont équivalentes. Plus précisément,

Proposition 3
Soit F l'ensemble des variables manipulées. G_a représente le graphe augmenté construit à partir de G en ajoutant les nœuds parents DO_i à chaque $A_i \in F$ dont les distributions locales sont données par la définition 1. La distribution de possibilité jointe π_{G_a} associée à G_a sur l'ensemble des variables initiales $V = \{A_1, ..., A_n\}$ est équivalente à la distribution de possibilité jointe associée au graphe initial G. Formellement,
i) $\forall \omega \forall i : 1, ..., n, \pi_{G_a}(\omega) = \pi_G(\omega) = \pi_{G_a}(\omega | do_{i-noact})$,
ii) $\forall \omega \forall i : 1, ..., n, \pi_{G_a}(\omega | DO_i = do_{a_i}) = \pi_G(\omega | do(a_i))$.

Preuve 1
Soient ω une interprétation sur l'ensemble de variables $V = \{A_1, ..., A_i, ..., A_n\}$ et do_i une instance quelconque de la variable DO_i. Nous avons,

i)
$$\begin{aligned}\pi_{G_a}(\omega) &= \max_{do_i} \pi_{G_a}(\omega, do_i)\\ &= \max_{do_i}(\pi_{G_a}(a_1|u_1)....\pi_{G_a}(a_i|u_i, do_i)....\pi_{G_a}(a_n|u_n).\pi_{G_a}(do_i))\\ &= [\prod_{a_j \neq a_i} \pi_{G_a}(a_j|u_j)].[\max(\pi_{G_a}(a_i|u_i, do_{i-noact}).\pi_{G_a}(do_{i-noact}),\\ &\quad \max_{do_{a'_i}}(\pi_{G_a}(a_i|u_i, do_{a'_i}).\pi_{G_a}(do_{a'_i})))]\\ &= [\prod_{a_j \neq a_i} \pi(a_j|u_j)].[\max(\pi_{G_a}(a_i|u_i, do_{i-noact}), \pi_{G_a}(do_{a_i}))]\\ &= [\prod_{a_j \neq a_i} \pi(a_j|u_j)].[\max(\pi_{G_a}(a_i|u_i), \epsilon)]\\ &= \prod_{a_i \in D_{A_i \in V}} \pi(a_i|u_i) = \pi_G(\omega)\end{aligned}$$

Nous avons aussi,

$$\begin{aligned}\pi_{G_a}(\omega|Do_i = do_{i-noact}) &= \pi_{G_a}(a_1|u_1)...\pi_{G_a}(a_i|u_i, do_{i-noact})...\pi_{G_a}(a_n|u_n)\\ &= \pi_{G_a}(a_1|u_1)...\pi_{G_a}(a_i|u_i)...\pi_{G_a}(a_n|u_n)\\ &= \pi_{G_a}(\omega) = \pi_G(\omega)\end{aligned}$$

ii) Lorsque $DO_i \neq do_{i-noact}$, nous obtenons $\forall a_i : \omega[A_i] = a_i$, :

$$\begin{aligned}\pi_{G_a}(\omega|Do_i = do_{a_i}) &= \pi_{G_a}(a_1|u_1, do_{a_i})...\pi_{G_a}(a_i|u_i, do_{a_i})...\pi_{G_a}(a_n|u_n, do_{a_i})\\ &= \pi_{G_a}(a_1|u_1)...\pi_{G_a}(a_i|u_i, do_{a_i})...\pi_{G_a}(a_n|u_n)\end{aligned}$$

En utilisant l'équation 6 (définition de $\pi_{G_a}(a_i|u_i, do_{a_i})$), nous obtenons le même résultat que $\pi_G(\omega|do(a_i))$ (voir 4).
Lorsque $\omega[A_i] \neq a_i$,
$\pi_{G_a}(\omega|Do_i = do_{a_i}) = \pi_G(\omega|do(a_i)) = 0.$
Ce résultat peut facilement être étendu pour le traitement de plusieurs interventions.

Exemple 4
Considérons le réseau causal possibiliste G dans la figure 1 et le graphe augmenté G_a dans la figure 3. La distribution de possibilité locale au niveau du nœud DO_B est donnée dans la table 3. La distribution de possibilité locale au niveau de B est calculée en utilisant (6). La distribution de possibilité π_{G_a} associée à G_a est donnée dans la table 4. La distribution de possibilité π_G (voir la table 1) et π_{G_a} sur les nœuds initiaux A et B sont équivalentes. Par exemple, $\pi_G(a_2, b_1) = \pi_{G_a}(a_2, b_1) = 0.24$.

DO_B	$\pi_{G_a}(DO_B)$
$do_{B-noact}$	1
do_{b_1}	0.001
do_{b_2}	0.001

TAB. 3 – La distribution de possibilité locale $\pi_{G_a}(DO_B)$

A	B	DO_B	π_{G_a}	A	B	DO_B	π_{G_a}
a_1	b_1	$noact$	1	a_2	b_1	$noact$	0.24
a_1	b_1	do_{b_1}	0.001	a_2	b_1	do_{b_1}	0.001
a_1	b_1	do_{b_2}	0	a_2	b_1	do_{b_2}	0
a_1	b_2	$noact$	0.5	a_2	b_2	$noact$	0.4
a_1	b_2	do_{b_2}	0.001	a_2	b_2	do_{b_2}	0.001

où $noact = do_{B-noact}$

TAB. 4 – La distribution de possibilité jointe $\pi_{G_a}(A, B, DO_B)$

4.2 Les arbres de jonction augmentés

Un deuxième point important est que les nœuds "DO_i" peuvent d'une manière équivalente être ajoutés soit avant soit après la construction de l'arbre de jonction. De point de vue calculatoire, il est préférable de commencer par la construction de l'arbre de jonction à partir du réseau possibiliste initial (avant de l'augmenter). Une fois l'arbre de jonction construit, nous procédons à l'addition de nouveaux nœuds (DO_i) ainsi que des distributions qui leur sont associées ($\pi(A_i|U_i, DO_i)$ et $\pi(DO_i)$) aux distributions locales au niveau des cliques lors de l'initialisation de l'arbre de jonction.
Notre nouvel algorithme se résume aux étapes suivantes :

- Construire un arbre de jonction (JT) à partir du graphe initial G.
- Pour chaque clique C_i, $\forall\omega, \pi_{C_i}(\omega) = 1$.
- Pour chaque nœud A_i dans G, choisir une clique C_i contenant $A_i \cup U_i$
  - si $A_i \in F$, alors
    - ajouter le nœud DO_i à C_i
    - $\pi_{C_i} \leftarrow (\pi_{C_i}.\pi(A_i|U_i DO_i).\pi(DO_i))$ (utilisant l'éq. 6 et la définition 1),
  - sinon $\pi_{C_i} \leftarrow (\pi_{C_i}.\pi(A_i|U_i))$.
- Intégrer l'évidence (observations et interventions).
- Propager l'évidence jusqu'à atteindre la stabilité de l'arbre de jonction.
- Répondre aux requêtes.

La proposition suivante montre que les distributions de possibilité jointes associées à l'arbre de jonction et au graphe augmenté G_a de G sont équivalentes.

Proposition 4
Soient G un réseau causal possibiliste et F un ensemble de variables manipulées dans G. do_F correspond à un élément de $\times_{A_i \in F} D_{DO_i}$ et ω est une interprétation sur $V = \{A_1, ..., A_n\}$. Soit G_a le graphe augmenté obtenu à partir de G en ajoutant à chaque nœud A_i dans F un nœud parent DO_i. JT dénote l'arbre de jonction formé à partir de G et initialisé comme nous l'avons indiqué dans ce qui a précédé. Nous avons donc,

$$\forall\omega\forall do_F, \pi_{JT}(\omega, do_F) = \pi_{G_a}(\omega, do_F)$$

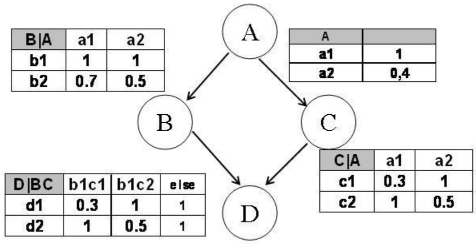

FIG. 4 – Le graphe à connexions multiples G

où π_{JT} (resp. π_{G_a}) dénote la distribution de possibilité associée à JT (resp. G_a).

Exemple 5
Considérons le réseau causal possibiliste dans la figure 4. Soient A et B des variables qui pourraient être manipulées. Le graphe augmenté résultant G_a est obtenu en ajoutant un nœud parent DO_A (resp. DO_B) à A (resp. B). Les distributions de possibilité locales associées aux nœuds DO_A et DO_B sont données comme suit : $\pi(do_{A-noact}) = \pi(do_{B-noact}) = 1$ et $\pi(do_{a_1}) = \pi(do_{a_2}) = \pi(do_{b_1}) = \pi(do_{b_2}) = 0.001$.
Le degré de possibilité associée à l'évènement $\omega^+ = (a_1, b_1, c_1, d_1, do_{a_1}, do_{B-noact})$ est calculé à partir de G_a comme suit :

$$\pi_{G_a}(\omega^+) = (\pi_{G_a}(a_1|do_{a_1}).\pi_{G_a}(b_1|a_1, do_{B-noact}).\pi_{G_a}(c_1|a_1).$$
$$\pi_{G_a}(d_1|b_1, c_1).\pi_{G_a}(do_{a_1}).\pi_{G_a}(do_{B-noact})) = 0.00009$$

où les distributions locales au niveau de A et B sont calculées en utilisant l'équation 6. En

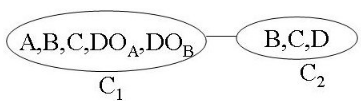

FIG. 5 – L'arbre de jonction JT après l'étape d'initialisation

initialisant l'arbre de jonction JT (formé à partir de G) consiste à initialiser les distributions de possibilité locales au niveau de chaque clique :
$\pi_{C_1}(A, B, C, DO_A, DO_B) = (\pi(A|DO_A).\pi(DO_A).$
$\pi(B|ADO_B).\pi(DO_B).\pi(C|A)).$
$\pi_{C_2}(B, C, D) = \pi(D|BC).$
L'arbre de jonction obtenu après l'initialisation est donné dans la figure 5.
Calculer le degré de possibilité de l'évènement ω^+ à partir de l'arbre de jonction initialisé JT, nous obtenons :

$$\pi_{JT}(\omega^+) = \pi_{C_1}(\omega^+).\pi_{C_2}(\omega^+)$$
$$= \pi_{C_1}(a_1, b_1, c_1, do_{a_1}, do_{B-noact}).\pi_{C_2}(b_1, c_1, d_1) = 0.00009$$

qui est égal au degré de possibilité calculé à partir du graphe augmenté G_a.

5 Conclusion

La première contribution de cet article concerne les fondements des réseaux causaux possibilistes. Nous avons montré comment les interventions peuvent être traitées en utilisant les graphes mutilés, les réseaux causaux possibilistes augmentés ainsi que les arbres de jonction augmentés. Un nouvel algorithme de propagation dans les réseaux causaux possibilistes traitant aussi bien les observations que les interventions représente la deuxième contribution principale de cet article. Notre travail futur consiste à appliquer les résultats de cet article aux réseaux causaux possibilistes basés sur le conditionnement ordinal. Les résultats actuels montrent que les définitions de la causalité s'adaptent bien au conditionnement ordinal cependant il est difficile d'avoir la contrepartie de l'algorithme proposé dans cet article pour le cadre ordinal.

6 Remerciements

Ce travail est soutenu par le projet national ANR Blanc MICRAC.

Références

BEN AMOR N., BENFERHAT S., DUBOIS D., MELLOULI K. & PRADE H. (2003). Anytime propagation algorithm for min-based possibilistic graphs. *Soft Computing, A fusion of foundations, methodologies and applications*, **8**, 150–161.

BORGELT C., GEBHARDT J. & KRUSE R. (1998). Possibilistic graphical models. In *Proceedings of International School for the Synthesis of Expert Knowledge (ISSEK'98)*, p. 51–68, Udine (Italy).

DUBOIS D. & PRADE H. (1988). *Possibility theory : An approach to computerized, Processing of uncertainty*. New York : Plenium Press.

DUBOIS D. & PRADE H. (1991). Possibilistic logic, preferential models, nonmonotonicity and related issues. In *IJCAI*, p. 419–425.

FONCK P. (1994). *Réseaux d'inférence pour le raisonnement possibiliste*. PhD thesis, Université de Liège, Faculté des Sciences.

GOLDSZMIDT M. & PEARL J. (1992). Rank-based systems : A simple approach to belief revision, belief update, and reasoning about evidence and actions. In *Proceeding of the Third International Conference (KR'92)*, p. 661–672. Kaufmann.

JENSEN F. V. (1996). *Introduction to Bayesien networks*. University college, London : UCL Press.

LAURITZEN S. L. & SPIEGELHALTER D. J. (1988). Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society*, **50**, 157–224.

PEARL J. (1988). *Probabilistic reasoning in intelligent systems : networks of plausible inference*. San Francisco (California) : Morgan Kaufmman.

PEARL J. (1993). Comment : Graphical models, causality and intervention. *Statistical Sciences*, **8**.

PEARL J. (2000). *Causality : models, reasoning, and inference*. New York, NY, USA : Cambridge University Press.

SPOHN W. (1988a). A general non-probabilistic theory of inductive reasoning. In *Proceedings of the Fourth Conference on Uncertainty in Artificial Intelligence (UAI'88)*, p. 149–158.

SPOHN W. (1988b). Ordinal conditional functions : a dynamic theory of epistemic states causation in decision. In *Belief Changes and Statistics*, p. 105–134. W. Harper and B. Skyrms.

VERMA T. & PEARL J. (1990). Equivalence and synthesis of causal models. In *UAI*, p. 255–270.

Vers une planification basée sur une Théorie Constructive des Types en logique Intuitionniste.

Patrick Barlatier, Richard Dapoigny

LISTIC (EA 3703), Eq. LS
Domaine Universitaire, B.P. 80439, 74944 Annecy-le-Vieux cedex
patrick.barlatier@univ-savoie.fr et
http://www.listic.univ-savoie.fr/
richard.dapoigny@univ-savoie.fr et
http://www.listic.univ-savoie.fr/

Résumé : Ce papier propose une nouvelle sémantique pour la logique de planification dans laquelle on considère un but comme une fonction de son contexte (le contexte étant une structure de connaissance sur le domaine). Étant donnés un but global et une situation initiale, le modèle de planification est généré directement à partir d'un ensemble structuré de buts primitifs par un raisonnement sur les types de buts et la connaissance du domaine. Cette dernière est extraite d'une ontologie locale de domaine par une sélection précise et ordonnée des entités disponibles et de leurs relations. Formellement, cette sélection est modélisée par des types d'enregistrements dépendants de la théorie intuitionniste des types (ITT). Ainsi, en utilisant un démonstrateur de théorème reposant sur cette théorie, il est possible de vérifier la validité (plus précisément le type) d'un ensemble d'objets instanciés représentant la connaissance et l'état actuel du système. Cette identification par un type correspond à une connaissance partielle du domaine associée à un but pour produire une action.

Mots-clés : Intuitionnisme, contraintes, types, sous-typage, planification, représentation des connaissances.

1 Introduction

En planification, il existe à une demande croissante pour une meilleure expressivité des connaissances du domaine afin d'améliorer les performances et de diminuer la complexité des planificateurs (Wilkins & DesJardins, 2001). Le besoin de dépasser les limites imposées par les générateurs de plans automatiques a débouché sur l'intégration de l'ingénierie des connaissances comme nouveau champ d'investigation (Doniat & Aylett, 2001; Barták & McCluskey, 2006). L'acquisition des connaissances sur le domaine et son contrôle peut également améliorer les performances des planificateurs mais nécessite une interaction entre l'utilisateur et le système via des langages contrôlés (Ferguson & Allen, 1998; Cortellesa & Cesta, 2006). Ce dernier aspect démontre l'existence d'un lien entre le Traitement Automatique des Langues Naturelles (TALN) et la planification.

Par ailleurs, dans les applications nécessitant un raisonnement sur les connaissances des systèmes, les approches traditionnelles telles que le *Situation Calculus* décrivent l'état du système en utilisant l'hypothèses du monde clos (Etzioni *et al.*, 1995; Reiter, 1978) qui est une notion non axiomatisable (Girard, 2006). De telles hypothèses supposent que l'information sur le monde est complète et qui plus est correcte. En outre, elles induisent de nombreux problèmes tels que le *frame problem*.

Dans le domaine de la planification, nous proposons dans cet article une approche différente pour répondre au problème de l'expressivité en terme de modélisation des connaissances. Ce modèle est basé sur la logique intuitionniste et les types dépendants. Son objectif est d'automatiser des raisonnements corrects sur ces connaissances dans le cadre de la planification de processus physiques. Le modèle est centré sur la représentation de trois concepts, le contexte des actions, les actions ainsi que les effets correspondants incluant le but. Pour cela, nous utilisons un formalisme de représentation intégrant ces concepts dans des structures de types dépendants. Ce formalisme permet d'expliciter la sémantique des diverses composantes des concepts et facilite la planification par la notion de sous-typage, méthode représentant une recherche de *pattern* à grain moins fin que les approches traditionnelles.

2 Bases Logiques

L'hypothèse sous-jacente à la présente approche utilise des connaissances partielles sur un système rassemblées sous le nom de contexte d'un processus physique. S'il n'est pas possible de connaître l'état total d'un système, il est envisageable de vérifier si certaines portions de cet état (contextes) sont valides et d'en déduire l'action possible ainsi que ses effets. Pour cela une logique adaptée associée à un démonstrateur de théorèmes est un élément essentiel. Les dernières décennies ont vu l'émergence de nombreuses logiques et parmi celles-ci, la logique intuitionniste ou logique constructive (Girard, 1989; Coquand & Coquand, 1999; Martin-Löf, 1982; Boldini & Bourdeau, 2004). C'est une logique de la connaissance aussi appelée logique active par opposition à la logique classique ou passive (Huet *et al.*, 1995). En effet, elle stipule que les objets dont les démonstrations intuitionnistes affirment l'existence sont effectivement calculables. Elle permet également, la définition et l'automatisation de raisonnements corrects sur des informations issues d'une base de données ou d'une ontologie. Seules les observations fournissent des informations, et l'absence d'observation d'une propriété ne peut être utilisée pour prouver que celle-ci est fausse. Les seuls éléments qui entrent en compte dans la représentation des croyances sont d'une part la connaissance de la structure du système (ontologie des descriptions possibles), et d'autre part une notion d'ordre partiel sur ces descriptions permettant de les comparer suivant la quantité d'informations qu'elles fournissent.

En observant les objets physiques comme des éléments essentiels de processus physiques et en se basant sur les résultats de travaux récents dans le domaine du traitement des langues naturelles (Boldini, 2000; Ginzburg, 2005; Cooper, 2005; Ranta, 2004), dans le domaine de la certification de programmes (Bove & Capretta, 2001; Coquand & Coquand, 1999; Paulson, 1989) et plus récemment dans le domaine du Web sémantique (Halpin & Thompson, 2006), nous avons proposé un modèle utilisant la théorie intuitionniste des types (Barlatier & Dapoigny, 2007; Dapoigny & Barlatier, 2006, 2007b) pour modéliser les contextes, les actions et les effets. La représentation des contextes de processus est décrite par des enregistrements à types dépendants. Nous considérons plus particulièrement le *contexte-de* l'action, c'est à dire le *contexte-de* l'action menant au but désiré. Par ailleurs, des travaux ont montré que la représentation de connaissances liées au contexte nécessite une ontologie (Strang & Linnhoff-Popien, 2004). La description des concepts manipulés est fournie par une ontologie de domaine dont les objets serviront de preuves formelles. L'association d'une ontologie au présent modèle formalise et généralise la notion de type.

La théorie intuitionniste des types, que l'on dénotera par ITT par la suite, est issue de la synthèse de la logique intuitionniste et d'un système formel de types. Celle-ci exploite simultanément la logique intuitionniste et la théorie des types du λ-calcul typé en fournissant un système de typage élaboré dans lequel toute propriété en logique des prédicats peut être interprétée comme un type. C'est la base d'importants travaux en logique ainsi qu'en IA (démonstrateurs de théorème tels que Coq, LEGO, ...), languages fonctionnels et logiques (Logique Linéaire, Théorie des types de Martin Löf, Théorie Constructive des types de Coquand, ...). En logique intuitionniste, le sens de chaque constante logique doit être extrait de la spécification de ce qui peut constituer une preuve pour toute proposition dont cette constante est le principal connecteur. Un des apports majeurs de la théorie des types est le paradigme "proofs-as-programs" plus connu sous le nom d'isomorphisme de Curry-Howard associant une preuve constructive à un programme qui réalise la formule prouvée (Howard, 1980). Un autre atout réside dans la calculabité de n'importe quel jugement (Coquand & Coquand, 1999; Martin-Löf, 1982; Valentini, 1996) : la théorie intuitioniste des types est fonctionnellement décidable (Valentini, 1996). Dans le jugement de type $a : T$, on classifie un objet a comme étant de type T. Une innovation de ITT est l'introduction des types dépendants et des enregistrements à types dépendants (Betarte, 2000; Kopylov, 2003). Ceux-ci sont exprimés en termes de données et peuvent exprimer toute connaissance issue d'information en étant plus flexible que les systèmes de types conventionnels (i.e., ils offrent un continuum de précision s'étendant d'une simple assertion de base jusqu'à la spécification complète d'un problème). Le concept de contexte peut être exprimé par des enregistrements à types dépendants intégrant simultanément de simple types, des types de propositions et/ou des types de fonctions. Les types de contexte (ce qui est possible) se distinguent des objets de contexte appelés tokens (la réalité). Cette subdivision types/objets bénéficie en outre du support de la représentation ontologique pour la spécification des applications lors de la conception. La capacité de fournir une structure simple qui peut être ré-utilisée afin de spécifier différentes sortes d'objets sémantiques structurés est un élément clé du modèle (Dapoigny & Barlatier, 2007a).

Definition 1
Un enregistrement à types dépendants est une suite de champs dans laquelle les labels l_i *correspondent à certains types* T_i*, c'est à dire, que chaque champ successif peut*

dépendre des valeurs des champs précédents :

$$C = \begin{bmatrix} l_1 & : T_1 \\ l_2 & : T_2(l_1) \\ \dots & \\ l_n & : T_n(l_1 \dots l_{n-1}) \end{bmatrix} \qquad exemple : C_1 = \begin{bmatrix} x & : Person \\ y & : Ticket \\ p_1 & : own(x,y) \\ z & : Flight \\ p_2 & : has\_Flight(y,z) \\ t & : Town \\ p_3 & : has\_Destination(z,t) \end{bmatrix}$$

Une définition similaire introduit les objets (tokens) dans laquelle une suite de valeurs est telle qu'une valeur v_i peut dépendre des valeurs des champs précédents $l_1, ..., l_{i-1}$:

$$c = \begin{bmatrix} l_1 & = v_1 \\ l_2 & = v_2 \\ \dots & \\ l_n & = v_n \\ \dots & \end{bmatrix} \qquad exemple : c_1 = \begin{bmatrix} x & = John \\ y & = t0015JK \\ p_1 & = q_1 \\ z & = ZU515 \\ p_2 & = q_2 \\ t & = Zurich \\ p_3 & = q_3 \\ \dots & \end{bmatrix}$$

où q_1 est une preuve de $own(John, t0015JK)$, q_2 une preuve que $has\_Flight(John, ZU515)$ et q_3, une preuve de $has\_Destination(John, Zurich)$. Nous allons montrer dans les deux paragraphes suivants comment les enregistrements à types dépendants peuvent représenter les contextes ainsi que la structure intentionnelle associée aux actions et aux buts résultants.

3 Structures de types dépendants associées aux contextes

Dans le cadre des processus physiques, la notion de Context Record Type (CRT) est exprimée par un type d'enregistrements dépendants dans lequel les champs détaillent la connaissance (i.e., les concepts, les propriétés et les contraintes). Il n'y a pas de limite supérieure au nombre de champs. Le CRT vide $<>$ n'impose aucune propriété et aucune contrainte. En supposant que Γ est un contexte valide[1], les règles suivantes introduisent les CRTs comme des enregistrements dépendants ($record - type$) :

$$\frac{}{\Gamma \vdash <>: record - type} \tag{1}$$

$$\frac{\Gamma \vdash R : record - type \;\; \Gamma \vdash T : record - type \to type}{\Gamma \vdash < R, l : T >: record - type} \tag{2}$$

La première règle affirme l'existence d'un tout, le CRT vide tandis que la seconde définit la formation des enregistrements à types dépendants pourvu que le champ l

[1] Un contexte valide en théorie des types est une suite $x_1 : T_1, \dots x_n : T_n$ telle qu'il existe un jugement dont la partie gauche d'un séquent comporte cette séquence.

ne soit pas déjà déclaré dans R. Nous supposerons également que les constructions syntaxiques primitives (i.e., égalité, application fonctionnelle et lambda abstraction) sont valables (pour plus de détails voir (Martin-Löf, 1982)).

Un des aspects les plus importants des CRT est la notion de sous-typage. Par exemple, un objet d'enregistrement à types dépendants avec des champs additionnels non mentionnés dans le type de départ est encore du même type. Le mécanisme d'enrichissement d'informations correspond à l'extension d'un type de contexte C par un type de contexte C'. En théorie des types, la proposition et le jugement sont équivalents. Il est alors possible de conclure que le jugement C est étendu (*lifted*) au jugement C'. Comme la résolution du sous-typage nécessite la connaissance de toutes les coercions possibles pour un terme donné ainsi que leurs effets, elle est incalculable en pratique. Nous contournons ce problème en imposant des contraintes sémantiques sur les coercions (Betarte, 2000).

Definition 2
Soient deux CRT C et C', si C contient au moins tous les labels déclarés dans C' et si les types des labels communs sont dans la relation d'inclusion, alors C est un sous-type de C' :

$$C \sqsubseteq C' \tag{3}$$

L'inclusion de types et les règles correspondantes généralisent l'inclusion des CRT aux enregistrements à types dépendants et la propagent aux autres types du language.

4 Structures de types dépendants associées aux buts

Dans la suite, nous supposerons que les majuscules dénotent des types et les minuscules, les objets (ou tokens). L'observation des situations courantes conduit aux hypothèses suivantes :

- Une action peut s'exécuter dans plusieurs contextes.
- Un même contexte ne peut pas être commun à plusieurs actions simultanément.

L'idée de base consiste à considérer le contexte, l'action et les effets (incluant le but) comme des DRTs. On peut alors associer par une fonction l'action au contexte, puis à associer le résultat obtenu aux effets. La théorie des types d'enregistrement dépendants permet pour cela la définition de fonctions et de types de fonctions grâce à une version du λ-calcul typé.

Definition 3
Etant donné un CRT C, une action peut être intuitivement décrite par une famille de types d'enregistrements qui est une fonction d'enregistrements de type CRT vers des types d'enregistrements, comme la λ-abstraction :

$$\lambda c : C.[a : action\_verb(\ldots, c.l_i, \ldots) \tag{4}$$

où $action\_verb$ est un verbe d'action, l_i un champ de c et a une proposition preuve de l'action effectuée.

Les types sont extraits d'une ontologie locale pour composer le CRT servant d'entrée au concept d'action. En utilisant l'exemple précédent, une fonction du type :

$$\lambda c_1 : C_1 . [a : take\_flight(c_1.x, c_1.z)$$

applique un enregistrement de la forme c_1 vers un enregistrement de la forme $[a = r_1$ où r_1 est une preuve de $take\_flight(John, ZU515)$. Les types de base génèrent un contexte dans lequel les types et objets respectifs sont valides (récursivité possible). La notion de contrainte est représentée par une fonction entre DRT (Cooper, 2005). L'action est en fait considérée simultanément comme une conséquence de la présence d'un contexte et comme une pré-condition à l'existence des effets de cette action. Cet aspect correspond à la notion de point fixe, notion qui a notamment permis de décrire la quantification dynamique en TALN. Si $\mathcal{F}$ représente une famille de types d'enregistrement dépendants alors a est un point fixe pour $\mathcal{F}$ si $a : \mathcal{F}(a)$. Supposons que A soit la famille :

$$\lambda c_1 : \left[\begin{array}{ll} x & : Person \\ y & : Ticket \\ p_1 & : own(x,y) \\ z & : Flight \\ p_2 & : has\_Flight(y,z) \\ t & : Town \\ p_3 & : has\_Destination(z,t) \end{array}\right. . [a : take\_flight(c_1.x, c_1.z)$$

alors :

$$\left[\begin{array}{ll} x & : Person \\ y & : Ticket \\ p_1 & : own(x,y) \\ z & : Flight \\ p_2 & : has\_Flight(y,z) \\ t & : Town \\ p_3 & : has\_Destination(z,t) \\ a & : take\_flight(x,z) \end{array}\right.$$

représente le type de tous les points fixes de $\mathcal{A}$. La notion de point fixe est utilisée pour étendre la définition précédente à une structure intentionnelle en considérant le but et ses effets comme une fonction d'un contexte possible. Cette structure appelée OC est définie par une fonction admettant comme argument un enregistrement de type point fixe et qui retourne un type d'enregistrement décrivant le but associé ainsi que d'éventuelles propositions résultant de l'action sur l'environnement[2] :

$$\lambda r : \mathcal{F}(\mathcal{A}). \left[\begin{array}{l} g : [g_1 : ... \\ e : \left[\begin{array}{ll} e_1 : & \dots \\ e_2 : & \dots \end{array}\right. \end{array}\right. \tag{5}$$

Cette définition correspond à un lien dans lequel un agent observant une action de type $[a : action\_verb(...)$ prédit l'existence d'un but de type $\left[\ g_1 : \quad \dots\right.$ et des propositions associées. Ce type de fonction exprime le fait que le contexte de sortie (OC) est

[2]Le but est considéré comme un effet particulier.

une conséquence de l'existence d'un objet de type contexte (noté IC) associé à une action. A partir d'un même contexte, une seule action est possible (contexte de l'action). Les contextes résultants (OC) doivent tous différer pour préserver la nature fonctionnelle du modèle (cf fig.1). Cela revient à considérer que chaque contexte est relié à une seule action pour générer un but : l'existence de ce contexte 'cause' l'existence des propositions de l'OC (incluant le but). Notons qu'une action à effets conditionnels ayant des effets qui varient selon le contexte d'application sont bien décrites par ce modèle.

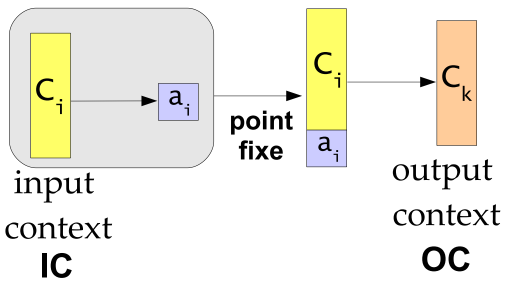

FIG. 1 – Contextes et structures intentionnelles.

5 Application au Planning

L'algorithme de planification est axé sur la causalité qui existe entre les CRTs de type IC et les types OC associés (voir fig. 2).

Proposition 1
Soit un objet valide $c_i : C_i$, s'il existe un objet $c_j : C_j$ tel que :

$$c_j \sqsubseteq \left[\begin{array}{l} g : [g_1 : ... \\ e : \left[\begin{array}{ll} e_1 : & \ldots \\ e_2 : & \ldots \end{array} \right. \end{array} \right.$$

alors c_i précède c_j.

La planification est réalisée par une progression dans laquelle la direction de recherche part de la situation initiale, mais au lieu de rechercher les états possibles, on recherche les objets de CRT valides et on essaie d'établir des chemins dans lesquels les noeuds sont des CRT via la relation de causalité. L'algorithme nécessite deux parties, une phase de vérification de type pour l'identification des contextes et une phase de backtraking. Le problème de planification peut se réduire au tuple $P = (S, \Sigma, \mathcal{C}, g)$, où S est la liste des contraintes initiales dynamiques (dont les valeurs peuvent changer durant le processus), Σ, les contraintes statiques (au moins pendant la durée du processus), $\mathcal{C}$ dénote l'ensemble des CRT et g le but final à réaliser. Alors $\Pi(S, \Sigma, \mathcal{C}, g)$ représente l'ensemble des plans possibles. La figure 3 dans laquelle a représente une

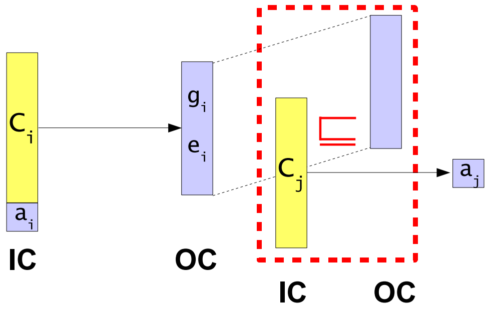

FIG. 2 – Opérateur de chaînage (SEQ).

		Sémantique STRIPS	Sémantique basée sur ITT
Comparaison structurelle	Approche	Epistémologique	Ontologique
	Logique	Premier ordre	ITT
	Hypothèse du monde clos	Oui	Non
	Problème traité	Etat initial + buts + ensemble d'actions	Situation initiale + buts + paires CRT-but
	Grain	Prédicats	DRTs
	Représentation des actions	Préconditions + effets	DRT + fonctions
	Effets conditionnels	Non	Oui
Comparaison fonctionnelle	Mécanisme	- si precond(a)=TRUE dans l'état s alors activer a	- déterminer la situation initiale et le(s) contexte(s) valide(s) par type-checking
		- substitution de variables	-sélection de l'action a
		- détermination de l'état final s' par mise à jour de s avec les effets	-recherche du nouveau contexte par sous-typage
		- arrêt quand le but appartient à s'	- arrêt quand le but est un sous-type du contexte obtenu
	Traitement du frame probleme	Tous les prédicats de s absents des effets restent inchangés	Seules les propositions et les types prouvés sont valides à un instant donné

FIG. 3 – Comparaison avec STRIPS.

action et s, s' des états, résume les différences essentielles avec la référence STRIPS. Au niveau structurel, l'approche basée sur ITT repose sur un modèle ontologique, présente un grain moins fin et gère les effets conditionnels. Sur le plan fonctionnel, son mécanisme n'utilise que le type-checking et le sous-typage. Enfin, notons que le frame problème est résolu par la logique elle-même qui ne considère que des types prouvés à un instant donné.

6 Conclusion

Le formalisme proposé ne prétend pas résoudre tous les problèmes de sémantique concernant la planification. Il s'agit plutôt, comme le suggère Girard, de proposer un formalisme capable d'unifier des branches aussi différentes de l'IA que le sont le TALN, la planification et la modélisation des web services. En effet, notre formalisme permet de représenter les connaissances en utilisant la théorie intuitionniste des types comme vecteur d'inter-opérabilité. Plus précisément, nous nous distinguons d'une approche classique basée sur une logique du premier ordre, par la possibilité de représenter des informations partielles et l'aspect dynamique de situations, tout en conservant l'avantage d'utiliser une logique décidable pour prouver nos programmes. Enfin, la compatibilité avec les systèmes de traitement des langues naturelles est un atout majeur pour la réalisation d'une interface utilisateur. Nos recherches s'orientent vers la génération d'une interface utilisateur graphique afin de résoudre des problèmes complexes de planification (Ferguson & Allen, 2005) et vers une extension du modèle aux systèmes distribués.

Références

BARLATIER P. & DAPOIGNY R. (2007). Using contexts to prove and share situations. In *Procs. of the 20th international FLAIRS conference*, p. 448–453 : AAAI Press.

BARTÁK R. & MCCLUSKEY L. (2006). The first competition on knowledge engineering for planning and scheduling. *AI magazine*, **27**(1), 97–98.

BETARTE G. (2000). Type checking dependent (record) types and subtyping. *Journal of Functional and Logic Programming*, **10**(2), 137–166.

BOLDINI P. (2000). Formalizing context in intuitionistic type theory. *Fundamenta Informaticae*, **42**, 1–23.

BOLDINI P. & BOURDEAU M. (2004). La theorie constructive des types. *Mathématiques et Sciences humaines*, **165**, 5–12.

BOVE A. & CAPRETTA V. (2001). Nested general recursion and partiality in type theory. In R. BOULTON & P. JACKSON, Eds., *TPHOL*, number 2152 in LNCS, p. 121–135 : Springer.

COOPER R. (2005). Records and record types in semantic theory. *J. Log. Comput.*, **15**(2), 99–112.

COQUAND C. & COQUAND T. (1999). Structured type theory. In *Workshop on Logical Frameworks and Meta-languages*.

CORTELLESA G. & CESTA A. (2006). Feature evaluation in mixed-initiative systems : An experimental approach. In S. S. D. B. DEREK LONG & L. MCCLUSKEY, Eds., *Procs. of ICAPS'06* : AAAI Press.

DAPOIGNY R. & BARLATIER P. (2006). Dependent record types for dynamic context representation. In F. C. M. BRAMER & A. TUSON, Eds., *Research and Development in Intelligent Systems : Procs. of AI-2006*, volume 23 : Springer.

DAPOIGNY R. & BARLATIER P. (2007a). Goal reasoning with context record types. In *Procs. of CONTEXT'07* : Springer. Accepted for publication.

DAPOIGNY R. & BARLATIER P. (2007b). Towards a context theory for context-aware systems. In *Procs. of the 2nd IJCAI Workshop on Artificial Intelligence Techniques for Ambient Intelligence*.

DONIAT C. & AYLETT R. (2001). Planform-ka tool : A cluster of constraint for knowledge acquisition and planning. In *Procs. of the 20th Workshop of the UK Planning and Scheduling Special Interest Group*.

ETZIONI O., GOLDEN K. & WELD D. S. (1995). *Sound and Efficient Closed-World Reasoning for Planning*. Rapport interne TR-95-02-02.

FERGUSON G. & ALLEN J. (1998). Trips : An intelligent integrated problem-solving assistant. In *Procs. of the Fifteenth National Conference on Artificial Intelligence (AAAI-98)*, p. 567–573.

FERGUSON G. & ALLEN J. (2005). Mixed-initiative dialogue systems for collaborative problem-solving. In *Procs. of the AAAI Fall Symposium on Mixed-Initiative Problem Solving Assistants*, p. 57–62.

GINZBURG J. (2005). Abstraction and ontology : Questions as propositional abstracts in type theory with records. *Journal of Log. Comput.*, **15**(2), 113–130.

GIRARD J.-Y. (1989). *Proofs and Types*. Cambridge University Press.

GIRARD J.-Y. (2006). *Le point aveugle*, volume 1 of *Vision des sciences*. Hermann.

HALPIN H. & THOMPSON H. (2006). One document to bind them : combining xml, web services and the semantic web. In *Procs. of the World Wide Web Conference*.

HOWARD W. A. (1980). *To H.B. Curry : Essays on Combinatory Logic, Lambda Calculus and Formalism*, chapter The formulae-as-types notion of construction, p. 479–490. Academic Press.

HUET G., KAHN G. & PAULIN-MOHRING C. (1995). *The Coq Proof Assistant - A tutorial*. Rapport interne 178, INRIA.

KOPYLOV A. (2003). Dependent intersection : A new way of defining records in type theory. In *Procs. of the 18th IEEE Symposium on Logic in Computer Science*, p. 86–95.

MARTIN-LÖF P. (1982). Constructive mathematics and computer programming. *Logic, Methodology and Philosophy of Sciences*, **6**, 153–175.

PAULSON L. (1989). The foundation of a generic theorem prover. *Journal of Automated Reasoning*, **5**, 363–397.

RANTA A. (2004). Grammatical framework : A type-theoretical grammar formalism. *Journal of Functional Programming*, **14**(2), 145–189.

REITER R. (1978). On closed world databases. In H. GALLAIRE & J. MINKER, Eds., *Proc. of ACM SIGMOD Int. Conf. on Management of Data*.

STRANG T. & LINNHOFF-POPIEN C. (2004). A context modeling survey. In *Sixth International Conference on Ubiquitous Computing (UbiComp2004)*, p. 34–41.

VALENTINI S. (1996). Decidability in intuitionistic type theory is functionally decidable. *Mathematical Logic*, **42**, 300–304.

WILKINS D. & DESJARDINS M. (2001). A call for knowledge-based planning. *AI Magazine*, **22**(1), 99–115.

Un algorithme pour la résolution optimale de problèmes de planification valués

Martin Cooper, Marie de Roquemaurel, Pierre Régnier

IRIT - Université Paul Sabatier
118, route de Narbonne
31062 Toulouse, cedex 9, France
{cooper,deroquemaurel,regnier}@irit.fr

Résumé : Nous montrons dans cet article comment utiliser le cadre des CSP pondérés (WCSP) pour trouver un plan-solution de coût optimal à des problèmes de planification avec actions valuées. La méthode proposée utilise un graphe de planification, structure disjonctive qui contient, pour un problème de planification donné, tous les plans-solutions potentiels d'une longueur k fixée. Ce graphe est codé en WCSP, puis nous utilisons l'algorithme de cohérence d'arc directionnelle FDAC à chaque noeud de la recherche pour trouver efficacement un plan-solution de coût optimal à un niveau k donné. Les bons résultats obtenus avec cette méthode nous permettent d'étendre ces travaux à la recherche d'un plan-solution de coût optimal. Notre algorithme poursuit la recherche dans les niveaux suivants du graphe après l'apparition d'une première solution et garantit l'obtention d'un plan-solution parallèle de nombre de niveaux minimum parmi ceux qui minimisent la métrique. Les résultats obtenus sont encourageants et montrent que l'utilisation des WCSP pour la résolution de problèmes de planification valués optimale est envisageable.

1 Introduction

Un des challenges actuels de la planification est la résolution de problèmes numériques pour lesquels on cherche à optimiser le coût d'un plan-solution. Parmi les planificateurs qui traitent ces problèmes numériques, certains font des choix heuristiques pour tenter d'optimiser le coût du plan produit (Do & Kambhampati, 2003; Gerevini *et al.*, 2004; Refanidis & Vlahavas, 2003; Haslum & Geffner, 2001; Chen *et al.*, 2004) alors que d'autres se contentent de calculer a posteriori le coût du plan-solution (Hoffmann, 2003).

En établissant des liens entre l'IA et la Recherche Opérationnelle, les CSP pondérés (WCSP) permettent de modéliser des contraintes strictes et des critères d'optimisation exprimés sous la forme d'une agrégation de fonctions de coût (Schiex *et al.*, 1995). Plusieurs codages de problèmes de planification en problèmes de satisfaction de contraintes ont déjà été proposés : CPLAN (van Beek & Chen, 1999), GP-CSP (Do & Kambhampati, 2001), puis CSPPLAN (Lopez & Bacchus, 2003). Aucune de ces approches n'a été étendue aux problèmes de planification avec actions valuées.

L'article est organisé de la manière suivante : la section 2 définit la planification avec actions valuées, GRAPHPLAN, et la notion de métrique d'un plan. La section 3 introduit un exemple qui nous servira à illustrer le fonctionnement de notre méthode. Dans la section 4 nous montrons ensuite comment coder un graphe de planification en WCSP. La section 5 présente l'algorithme de cohérence d'arc directionnelle FDAC qui nous permet d'extraire efficacement une solution optimale de ces WCSP. Nous montrons enfin, dans la section 6, comment l'algorithme GP-WCSP* utilise ces résultats pour obtenir un plan-solution parallèle de coût optimal.

2 Définitions préliminaires

Définition 2.1 (problème de planification avec actions valuées)
Un problème de planification avec actions valuées est un triplet $\Pi = \langle A, I, B\rangle$ tel que :

1. *l'état initial I est un ensemble fini de propositions (aussi appelés fluents) ;*
2. *A est un ensemble d'actions, i.e. de triplets $\langle prec(a), eff(a), cout(a)\rangle$, où $prec(a)$ est l'ensemble des préconditions propositionnelles de l'action a, $eff(a)$ est l'ensemble des effets propositionnels de a : ajouts et retraits de propositions $(add(a) \cup del(a))$, $cout(a)$ est un entier naturel non nul représentant le coût de l'application de l'action a ;*
3. *le but B est l'ensemble des propositions qui doivent être satisfaites. Une proposition p est satisfaite dans un état E ssi $p \in E$.*

Définition 2.2 (application d'une action)
L'application d'une action a à un état E (notée $E \uparrow a$) est possible ssi tous les fluents de prec(a) sont satisfaits dans E ; elle produit un état $E' = (E - del(a)) \cup add(a)$.

Définition 2.3 (graphe de planification)
Un graphe de planification (Blum & Furst, 1997) est un graphe orienté, construit en temps et en espace polynomial, composé de plusieurs niveaux qui comportent des noeuds d'actions et de fluents, et des arcs de préconditions, d'ajouts et de retraits. Le niveau 0 contient les fluents de l'état initial. Chaque niveau $i > 0$ est composé de toutes les actions applicables à partir des fluents du niveau $i-1$, et des fluents produits par les actions du niveau i. Les arcs représentent les relations entre les actions et les fluents : les actions du niveau i sont reliées aux fluents préconditions du niveau $i-1$ par des arcs préconditions, et à leurs ajouts et retraits du niveau i par des arcs d'ajouts et de retraits.

Le maintien des fluents du niveau $i-1$ au niveau i est assuré par des actions appelées $no-op$ qui ont un unique fluent comme précondition et ajout. Pour chaque niveau du graphe, on calcule des exclusions binaires (mutex) entre paires d'actions et paires de fluents d'un même niveau (cf. définition 2.4). Le graphe est étendu niveau par niveau jusqu'à ce que les fluents du but soient présents dans le niveau courant et qu'il n'existe pas de mutex entre eux.

Définition 2.4 (exclusion mutuelle)
Deux actions a_1 et a_2 sont mutex à un niveau i du graphe (noté $mutex(a_1, a_2, i)$) ssi :

1. *une action retire un fluent requis ou ajouté par l'autre action :*
$\Big((del(a_1) \cap (prec(a_2) \cup add(a_2))) \cup (del(a_2) \cap (prec(a_1) \cup add(a_1)))\Big) \neq \emptyset$ *ou*

2. *les actions ont en précondition des fluents mutex au niveau $i-1$ précédent (elles ne peuvent donc pas être déclenchées en même temps) :* $\exists (p,q) \in prec(a_1) \times prec(a_2)/mutex(p,q,i-1)$.

Deux fluents p et q d'un niveau i du graphe sont mutex ssi tous les couples d'actions qui les produisent à ce niveau sont mutex.

Définition 2.5 (action et ensemble indépendant)
Deux actions a et b sont des actions indépendantes (notée $a\#b$) ssi elles ne vérifient pas (1) dans la définition 2.4. Un ensemble d'actions $Q_i = \{a_1,\dots,a_n\}$ est un ensemble indépendant ssi tous les couples d'actions différentes qui le composent sont indépendantes deux à deux : $\forall\ a,b \in Q, a \neq b/a\#b$. *L'application d'un ensemble indépendant d'actions $Q = \{a_1,\dots,a_n\}$ sur un état E (notée $E \uparrow Q$) est possible ssi $\forall\ a \in Q$, $prec(a)$ est satisfait dans E. Il en résulte l'état E' tel que* $E' = \big(E - \bigcup\limits_{a\in Q} del(a)\big) \cup \bigcup\limits_{a\in Q} add(a)$. *Lorsqu'un ensemble d'actions Q est indépendant, l'application des actions de Q conduit à un état identique quel que soit leur ordre d'exécution.*

Définition 2.6 (plan parallèle)
Un plan parallèle de n niveaux est une séquence d'ensembles d'actions indépendantes Q_i, noté $P =< Q_1,\dots,Q_n >$. L'application d'un plan parallèle P de n niveaux à un état E_0 (noté $E_0 \uparrow P$) est possible ssi pour tout $i = 1,\dots,n$, Q_i est applicable dans E_{i-1} et produit l'état E_i. Dans ce cas, on dit que E_n est atteignable à partir de E et P est appelé plan parallèle correct. Un plan parallèle correct P est un plan-solution parallèle (ou plan parallèle valide) ssi E_0 est l'état initial I et que le but B est satisfait dans l'état final E_n.

Remarque : GRAPHPLAN construit des plans-solutions parallèles optimaux en nombre de niveaux. Dans la suite de cet article, nous nous intéresserons uniquement à ce type de plan.

Définition 2.7 (stabilisation du graphe)
La stabilisation du graphe est atteinte lorsque deux niveaux successifs contiennent les mêmes actions et fluents, et les mêmes mutex d'actions et de fluents.

Définition 2.8 (graphe réduit)
Un graphe réduit est un graphe de planification dans lequel en partant des buts présents au niveau k, on supprime toutes les actions du même niveau (et leurs arcs de précondition) qui ne permettent pas l'obtention de ces fluents ; on recommence ensuite au niveau $k-1$ en supprimant toutes les actions qui ne permettent pas l'obtention de préconditions des actions du niveau k qui ont été conservées et ainsi de suite jusqu'au niveau 1.

Définition 2.9 (métrique d'un plan)
La qualité d'un plan P est estimée par une fonction appelée métrique du plan. Nous considèrerons ici les métriques additives telles que $metrique(P) = \sum_{a\in P} cout(a)$. Nous noterons $\mathcal{P}_k$ l'ensemble des plans-solutions de longueur inférieure ou égale à k niveaux. $\mathcal{P}_k^*$ est l'ensemble des plans-solutions de $\mathcal{P}_k$ qui minimisent la métrique : $\forall\ P \in \mathcal{P}_k^* : metrique(P) = \min_{P'\in\mathcal{P}_k} metrique(P')$. *$P_k^*$ est un plan appartenant à $\mathcal{P}_k^*$. C_k^* sera le coût d'un plan-solution P_k^* de coût optimal parmi les plans de longueur inférieure ou égale à k niveaux.*

Définition 2.10 (plan optimal / coût optimal)
P^* est un plan-solution de nombre de niveaux minimum parmi ceux qui minimisent la métrique. C^* est le coût d'un tel plan.

Le problème que nous cherchons à résoudre consiste, pour un problème de planification avec actions valuées $\Pi = \langle A, I, B\rangle$, à trouver un plan-solution P^* (nous parlerons ici d'un problème de planification optimal).

3 Exemple

Les variables A, B, C, D et E représentent cinq villes, la variable v un véhicule, et c une caisse. Le fluent v_i représente le fait que le véhicule v est dans la ville i. v se déplace entre les villes en utilisant les actions $trajet_{ij}$ où i désigne la ville de départ et j celle d'arrivée. Le coût de cette action dépend de la distance entre les villes. Le fluent c_i représente le fait que la caisse c est dans la ville i, c_v le fait que la caisse est à bord du véhicule v. c peut être chargée dans le véhicule par l'action $prendre_i$ dans la ville i, avec un coût de 5. Cette caisse c peut être déchargée dans une des villes i par l'action $poser_i$ de coût 3. Le problème $\Pi = \langle A, I, B\rangle$ est défini par l'état initial $I = \{v_A, c_A\}$, le but $B = \{c_B\}$ et l'ensemble des actions A : $\forall\ i \in \{A, B, C, D, E\}$,
- $\forall\ j \in \{A, B, C, D, E\}, trajet_{ij} : < \{v_i\}, \{v_j, \neg v_i\}, cout_{ij} >$. La figure 1 donne les trajets possibles et leurs coûts.
- $prendre_i : < \{v_i, c_i\}, \{c_v, \neg c_i\}, 5 >$.
- $poser_i : < \{v_i, c_v\}, \{c_i, \neg c_v\}, 3 >$.

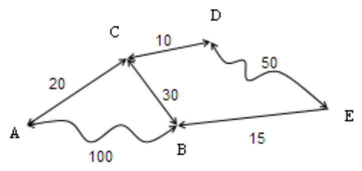

FIG. 1 – exemple

Dans un premier temps, la construction du graphe de planification se poursuit jusqu'au niveau 3, premier niveau dans lequel tous les buts apparaissent sans mutex. Le premier plan-solution trouvé $P_3^* = \langle \{prendre_A\}, \{trajet_{AB}, noop_{c_v}\}, \{poser_B\}\rangle$ correspond au seul plan séquentiel $\langle prendre_A, trajet_{AB}, poser_B\rangle$ qui a un coût $C_3^* = 108$. Ce plan-solution n'est cependant pas de coût optimal, puisqu'il existe un plan optimal en 4 niveaux $\langle prendre_A, trajet_{AC}, trajet_{CB}, poser_B\rangle$ de coût 58. Nous allons d'abord montrer comment obtenir un plan-solution de coût optimal de longueur donné puis, dans un deuxième temps, nous montrerons comment obtenir un plan optimal.

4 Codage du graphe en WCSP

Comme démontré dans (Do & Kambhampati, 2001), la phase d'extraction de GRAPHPLAN peut être vue comme un DCSP (Dynamic Constraint Satisfaction Problem) (Mittal & Falkenhainer, 1990). Initialement, seul un sous-ensemble de variables est actif, et l'objectif consiste à trouver une affectation pour toutes les variables actives telle qu'elle soit consistante avec l'ensemble des contraintes. Pendant l'extraction, chaque fluent f_i à un niveau i du graphe de planification peut être assimilé à une variable d'un CSP ; l'ensemble des actions qui ajoutent f_i constituent son domaine et les mutex produits durant l'étape de construction peuvent être assimilés à des contraintes du CSP. À partir des fluents du but, GRAPHPLAN essaye d'extraire un plan à partir du graphe de planification en affectant une valeur (une action) à chaque variable (fluent) satisfaisant l'ensemble de contraintes (mutex). L'affectation des valeurs aux variables est un processus dynamique car chaque affectation d'une variable à un niveau i active d'autres variables du niveau précédent (les préconditions de l'action choisie).

Le codage, par un WCSP, d'un plan issu d'un graphe de planification avec actions valuées nécessite plusieurs modifications de la méthode initialement proposée par (Do & Kambhampati, 2001) pour le codage d'un graphe de planification classique en CSP. Une fois le graphe de planification construit et réduit, son codage en WCSP nécessite sept étapes :

1. *Réécriture du graphe :* en partant du dernier niveau du graphe réduit, on renomme chaque fluent en f_i et on numérote chaque action en j, i et j étant des entiers strictement positifs. L'ordre dans lequel cette numérotation est réalisée est important car il détermine en partie l'ordre dans lequel les variables et les valeurs pourront être ordonnées pour la résolution du WCSP associé.
2. *Création des variables et des domaines :* pour chaque fluent n'appartenant pas à l'état initial, nous créons une variable dont le domaine est l'ensemble des actions qui produisent ce fluent. Pour les fluents n'appartenant pas au but, nous ajoutons la valeur -1 pour représenter sa non-activation éventuelle.
3. *Traduction des mutex entre fluents :* pour tous les fluents mutex f_i et f_j, l'exclusion mutuelle est traduite par une contrainte qui exprime le fait que f_i et f_j ne doivent pas être activés en même temps : $(f_i = -1) \vee (f_j = -1))$
4. *Traduction des mutex entre actions :* pour toutes les actions mutex a et b d'effets respectifs f_i et f_j $(f_i \neq f_j)$, l'exclusion mutuelle est traduite par une contrainte qui exprime le fait que f_i et f_j ne peuvent pas être activés en même temps par les actions a et b : $\neg((f_i = a) \wedge (f_j = b))$.
5. *Traduction des arcs d'activité :* l'activation d'un fluent f_i produit par une action a_i entraîne l'activation des préconditions de cette action. Ceci se traduit par la contrainte d'activité : $\forall f_j \in prec(a), (f_i = a) \Rightarrow (f_j \neq -1)$.
6. *Traduction du coût des actions :* Pour chaque valeur a, nous ajoutons une contrainte unaire pour chaque affectation $f = a$ correspondant au coût de l'action a. Les No-ops ont un coût nul.
7. *Prise en compte des ajouts multiples :* Lorsqu'une action a produit plusieurs fluents $f_i \in effet(a)$, le coût de cette action est compté plusieurs fois si plusieurs d'entre eux sont utiles à la résolution du WCSP. Pour résoudre ce problème, nous créons un fluent intermédiaire f^{int} de domaine $\{a, -1\}$. Ainsi l'action a est remplacée par une action fictive a^{int} (de coût nul) dans les domaines des fluents f_i. Nous ajoutons également une contrainte d'activité $(f_i = a^{int}) \Rightarrow (f^{int} = a)$ entre le fluent f^{int} et chaque fluent $f_i \in effet(a)$.

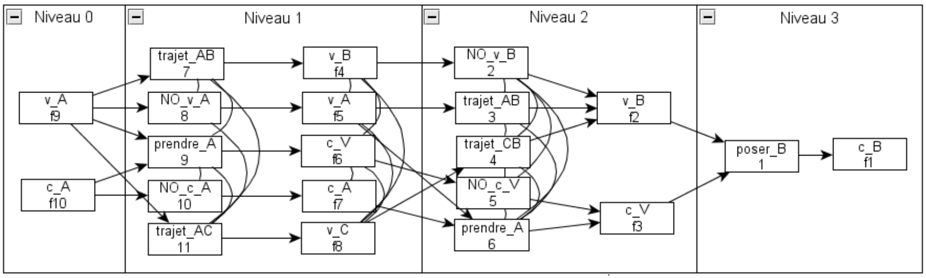

FIG. 2 – graphe de planification réduit et renommé de 3 niveaux.

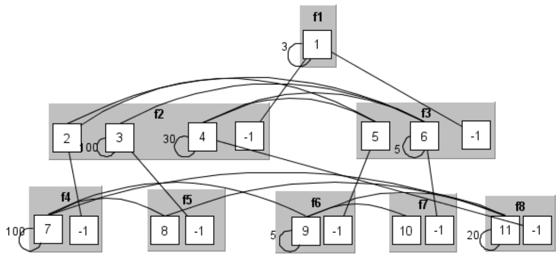

FIG. 3 – WCSP correspondant au graphe de la figure 2.

La figure 2 donne le graphe réduit de niveau 3 de l'exemple avec le renommage correspondant. Le codage du graphe en WCSP est présenté dans la figure 3 où toutes les contraintes binaires sont des mutex et ont donc des coûts infinis.

5 Recherche d'une solution optimale dans le WCSP

Dans cette section, nous utilisons c_{ij} pour représenter une fonction de coût binaire sur les variables X_i, X_j dans un WCSP, c_i représente une fonction unaire de coût sur X_i et $c_\emptyset$ représente une fonction de coût sans argument (indépendante des valeurs affectées aux variables). Une solution à un WCSP binaire est un n-tuple x qui minimise $\sum_i c_i(x_i) + \sum_{ij} c_{ij}(x_i, x_j) + c_\emptyset$.

La réduction de l'espace de recherche par propagation des inconsistances peut être généralisée des CSP aux WCSP, directement par propagation de coûts infinis. Les coûts finis peuvent aussi être propagés mais de telles propagations doivent être compensées : si l'on soustrait α de $c_{ij}(a, u)$ (pour chaque valeur u dans le domaine de la variable X_j), on doit simultanément ajouter α à $c_i(a)$ afin de conserver un WCSP équivalent. Quand $\alpha > 0$, l'opération est une projection ; si $\alpha < 0$ alors l'opération est une extension. Une troisième opération possible est la projection unaire, qui soustrait simultanément α de $c_i(u)$ (pour chaque valeur u dans le domaine de X_i) tout en ajoutant α à $c_\emptyset$. Dans ces trois cas, l'opération peut être appliquée seulement si tous les coûts résultants sont non négatifs. la propagation de coûts finis incrémente la borne inférieure $c_\emptyset$ et réduit l'espace de recherche exploré par le Branch-and-Bound.

En appliquant toutes les projections unaires possibles, nous établissons la cohérence de noeuds (Larrosa, 2002). En propageant tous les coûts infinis (entre les contraintes binaires et unaires) et en appliquant les opérations de projection jusqu'à la convergence,

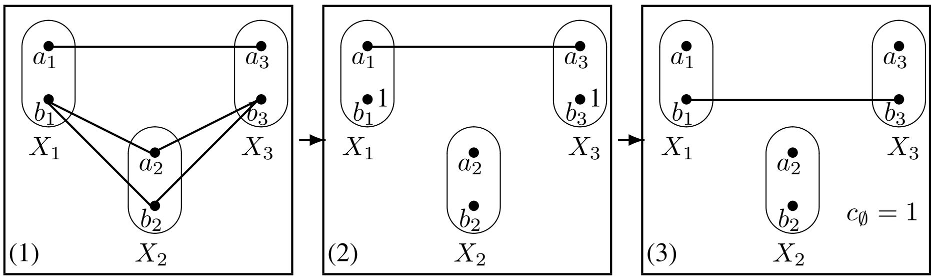

FIG. 4 – Exemple de simplification d'un WCSP par la méthode de propagation de contraintes FDAC.

nous établissons la cohérence d'arc (SAC) (Cooper & Schiex, 2004). Afin d'avoir des coûts non nuls disponibles le plus tôt possible pendant la recherche, la cohérence d'arc directionnelle (DAC) choisit toujours d'envoyer les coûts (par l'intermédiaire d'opérations de projection et d'extension) vers les variables qui apparaissent le plus tôt lors de l'instanciation. Ceci a également tendance à regrouper les coûts sur les mêmes variables, et permet ainsi d'arriver à une plus grande valeur de $c_\emptyset$ après l'établissement de la cohérence de noeud. L'algorithme de cohérence d'arc directionnelle totale FDAC (Cooper, 2003) est la combinaison de la cohérence d'arc directionnelle et de la cohérence d'arc.

Voici un exemple d'application de FDAC. Le WCSP initial décrit par la figure 4 (1) est composé de trois variables X_i de domaines $\{a_i, b_i\}$ $(i = 1, 2, 3)$. Un arc entre deux valeurs du WCSP représente un coût de 1. Par exemple, l'arc (a_1, a_3) signifie que $c_{13}(a_1, a_3) = 1$. FDAC commence par projeter un coût de 1 à partir de $c_{12}(b_1, a_2)$, $c_{12}(b_1, b_2)$ vers $c_1(b_1)$ (qui établit la cohérence d'arc directionnelle DAC) et puis à partir de $c_{23}(a_2, b_3)$, $c_{23}(b_2, b_3)$ vers $c_3(b_3)$ (une opération de cohérence d'arc (SAC)). Le WCSP résultant, montré dans la figure 4 (2), est presque arc consistant directionnel : étendre le coût de 1 à partir de $c_3(b_3)$ jusqu'à c_{13} nous permet de projeter un coût de 1 de c_{13} vers $c_1(a_1)$. La figure 4 (3) montre le WCSP obtenu après application de la projection unaire de c_1. Ce WCSP est totalement arc consistant directionnel par rapport à l'ordre des variables X_1, X_2, X_3, et le coût minorant de la solution $c_\emptyset$ est de 1.

Soit m le coût de la meilleure solution trouvé durant l'algorithme Branch-and-Bound. En posant que le but est de trouver une première solution optimale, nous pouvons couper les branches de l'arbre de recherche dès lors que $c_\emptyset \geq m$. De plus, nous pouvons attribuer ∞ à un coût unaire $c_i(a) \geq m - c_\emptyset$ et à un coût binaire $c_{ij}(a, b) \geq m - (c_i(a) + c_j(b) + c_\emptyset)$. Appliquer ces dernières opérations fournit une forme plus forte de cohérence d'arc, mais augmente la complexité en temps du calcul du pire cas de FDAC en passant de $O(ed^2)$ (Cooper, 2003) à $O(end^3)$ (Larrosa & Schiex, 2003), où e est le nombre de contraintes binaires, n le nombre de variables et d la taille maximum de domaine. FDAC a été récemment étendu à la cohérence d'arc directionnelle existentielle (EDAC) (de Givry *et al.*, 2005) qui exécute l'opération suivante : si pour *chaque* valeur a dans le domaine d'une variable X_i, il existe un voisin X_j tel qu'il est possible d'augmenter le coût de la contrainte unaire portant sur a en projetant le coût de c_j et de la contrainte c_{ij}, alors effectuer toutes ces opérations et appliquer une opération unaire de projection sur X_i, pour augmenter $c_\emptyset$. Comme tous les algorithmes de propagation de contraintes présentés précédemment, FDAC n'a pas de fermeture unique ; le résultat dépend de l'ordre dans lequel les différentes extensions et projections sont appliquées mais surtout de l'ordre des variables.

Les expérimentations que nous avons réalisées (Cooper *et al.*, 2006) nous ont permis de comparer l'efficacité des algorithmes de cohérence de noeud, EDAC et FDAC pour la résolution des WCSP issus des graphes de planification. Nos résultats montrent clairement que c'est l'utilisation de FDAC à chaque noeud qui permet de résoudre le plus efficacement ces WCSP en termes de temps de calcul. Ils montrent que la taille des WCSP qui peuvent être résolus en pratique est beaucoup plus importante que la taille du plus grand WCSP aléatoire résolu dans la littérature (de Givry *et al.*, 2005). Par exemple, l'espace de recherche d'une instance est de l'ordre de $3 * 10^{221}$ tandis que celui du plus gros problème aléatoire résolu dans (de Givry *et al.*, 2005) est de l'ordre de 10^{36}. Ceci peut être dû à la structure particulière du WCSP produit par le codage du graphe de planification : on retrouve la structure en niveau du graphe de planification avec une forte connectivité interne et entre niveaux adjacents).

Le wcsp de l'exemple est résolu avec FDAC appliqué à chaque noeud de l'arbre de recherche branch-and-bound. Le plan-solution optimal de niveau 3 correspondant est $P_3^* = \langle\{prendre_A\},\{trajet_{AB}, noop_{c_v}\},\{poser_B\}\rangle = \langle prendre_A, trajet_{AB}, poser_B\rangle$ de coût 108. Sachant efficacement trouver un plan-solution P_k^* de coût optimal à un niveau donné k, nous pouvons chercher un plan-solution optimal P^*.

6 Algorithme GP-WCSP*

L'algorithme anytime GP-WCSP* que nous avons développé est basé sur celui de GRAPHPLAN. Une fois la phase d'expansion du graphe de planification terminée, le graphe est codé en un WCSP équivalent (cf. partie 4) puis résolu (cf. partie 5). Le coût du plan-solution optimal trouvé à un niveau donné du graphe nous sert ensuite de majorant pour relancer la recherche d'une meilleure solution au niveau suivant.

Notre approche donne un planificateur complet dans le sens où l'algorithme s'arrête en retournant un plan-solution optimal P^* s'il existe. Par contre, comme pour tous les planificateurs de ce type (SATPLAN'06 (Kautz & Selman, 1999), (Kautz *et al.*, 2006), GP-CSP (Do & Kambhampati, 2001), MaxPlan (Chen *et al.*, 2007)...) le problème de l'arrêt lorsqu'il n'y a pas de solution n'est pas résolu.

L'algorithme garantit l'obtention d'un plan-solution P^* de coût optimal lorsqu'il atteint une borne supérieure au nombre de niveaux à développer (borne appelée $NivMax$). Une première estimation de cette borne peut-être facilement calculée à partir du premier plan-solution P_k^* obtenu au niveau k : dans le pire des cas, un plan optimal P^* aurait un coût $C^* = C_k^* - \epsilon$ (où $\epsilon > 0$) et serait un plan séquentiel composé de $(C_k^* - \epsilon)/C_{min}$ actions dont le coût ne pourrait être inférieur à $C_{min} = \min_{a \in G} cout(a)$, par conséquent $NivMax = \lfloor (C_k^* - \epsilon)/C_{min} \rfloor = \lceil C_k^*/C_{min} \rceil - 1$. Avec cette méthode de calcul, la valeur de $NivMax$ calculée au niveau 3 est 35, on construit donc le niveau suivant du graphe et on en extrait $P_4^* = \langle prendre_A, trajet_{AC}, trajet_{CB}, poser_B\rangle$ de coût $C_4^* = 58$. La valeur de $NivMax$ peut alors être diminuée ($NivMax \leftarrow 20$) et l'algorithme poursuit la recherche jusqu'au 20ème niveau afin de garantir que P_4^* est un plan optimal. En pratique les valeurs obtenues pour $NivMax$ ne permettent pas de résoudre des problèmes de planification optimaux de taille significative. Nos recherches actuelles (Cooper *et al.*, 2007) portent donc sur les moyens de diminuer efficacement la valeur de $NivMax$. Pour l'exemple, nous montrons ainsi que le développement du graphe au niveau 4 est suffisant pour garantir l'obtention d'un plan optimal.

Algorithme 1 Résolution optimale d'un problème de planification valué $\Pi\langle A, I, B\rangle$

Fonction :
– $constructionGraphe$: construit le graphe jusqu'à l'obtention de tous les buts sans mutex entre eux dans le niveau courant. Si le graphe ne contient pas tous les buts sans mutex au niveau de stabilisation alors il retourne un échec.
– $constructionGrapheNiveauSuivant(G_k)$: retourne le graphe de $k+1$ niveaux.
– $codageWCSP(G)$: retourne le wcsp correspondant au codage du graphe G réduit.
– $resolutionWCSP(wcsp, C)$: retourne une solution optimale au wcsp et son coût, strictement inférieur à C. Sinon retourne echec.
– $NivMax(C) = \lceil C/(min_{a\in A} cout(a)\rceil - 1$.

Algorithme GP-WCSP* :
// initialisation, recherche d'une première solution
$G_k \leftarrow constructionGraphe(A, I, B)$
si la construction du graphe G_k est un échec **alors**
  $echec$ // problème sans solution
finsi
$wcsp \leftarrow codageWCSP(G_k)$
$(P_k^*, C_k^*) \leftarrow resolutionWCSP(wcsp, \infty)$
tantque $P_k^* = echec$ **faire**
  $k \leftarrow k+1$
  $G_k \leftarrow constructionGrapheNiveauSuivant(G_{k-1})$
  $wcsp \leftarrow codageWCSP(G_k)$
  $(P_k^*, C_k^*) \leftarrow resolutionWCSP(wcsp, \infty)$
fin tantque
// boucle anytime
tantque $k < NivMax(C_k^*)$ **faire**
  $k \leftarrow k+1$
  $G_k \leftarrow constructionGrapheNiveauSuivant(G_{k-1})$
  $wcsp \leftarrow codageWCSP(G_k)$
  $(P_k^*, C_k^*) \leftarrow resolutionWCSP(wcsp, C_{k-1}^*)$
  si $P_k^* = echec$ **alors**
    $P_k^*, C_k^* \leftarrow P_{k-1}^*, C_{k-1}^*$
  finsi
fin tantque
return (P_k^*, C_k^*).

7 Conclusions et perspectives

Nous avons introduit une nouvelle méthode pour extraire un plan-solution de coût optimal à partir d'un graphe de planification avec actions valuées en utilisant des techniques de CSP pondérés. On obtient ainsi un plan-solution parallèle, optimal en nombre de niveaux, et de coût minimum parmi les plans d'une longueur donnée. Les très bons résultats obtenus nous permettent d'utiliser cette méthode dans l'algorithme GP-WCSP* pour obtenir une solution de coût optimal. Le nombre de niveaux à construire pour garantir l'obtention de cette solution optimale est fini mais inconnu au départ. Une borne supérieure à ce nombre peut être calculée à partir du premier plan-solution obtenu. Pour pouvoir résoudre pratiquement des problèmes de planification valués de taille significative, nous cherchons désormais à diminuer la valeur de cette borne en étudiant les propriétés des problèmes.

Références

BLUM A. & FURST M. (1997). Fast planning through planning graph analysis. *Artificial Intelligence (AI)*, **90**(1-2), 281–300.

CHEN Y., HSU C. & WAH B. (2004). Sgplan : Subgoal partitioning and resolution in planning. In *International Conference on Automated Planning and Scheduling (ICAPS)*, p. 30–33.

CHEN Y., ZHAO X. & ZHANG W. (2007). Long distance mutual exclusion for propositional planning. In *IJCAI*, p. 1840–1845.

COOPER M., CUSSAT-BLANC S., DE ROQUEMAUREL M. & RÉGNIER P. (2006). Soft arc consistency applied to optimal planning. In *International Conference on Principles and Practice of Constraint Programming, LNCS 4204*, p. 680–684.

COOPER M., DE ROQUEMAUREL M. & RÉGNIER P. (2007). Recherche d'une solution optimale dans des graphes de planification avec actions valuées. In *Journées Francophones Planification, Décision, Apprentissage pour la conduite de systèmes*.

COOPER M. C. (2003). Reduction operations in fuzzy or valued constraint satisfaction. *Fuzzy Sets and Systems*, **134**(3), 311–342.

COOPER M. C. & SCHIEX T. (2004). Arc consistency for soft constraints. *Artificial Intelligence*, **154**(1-2), 199–227.

DE GIVRY S., HERAS F., ZYTNICKI M. & LARROSA J. (2005). Existential arc consistency : Getting closer to full arc consistency in weighted csps. In *International Joint Conference on Artificial Intelligence (IJCAI)*, p. 84–89.

DO M. B. & KAMBHAMPATI S. (2001). Planning as constraint satisfaction : Solving the planning graph by compiling it into csp. *Artificial Intelligence*, **132**(2), 151–182.

DO M. B. & KAMBHAMPATI S. (2003). Sapa : A multi-objective metric temporal planner. *Journal of Artificial Intelligence Research (JAIR)*, **20**, 155–194.

GEREVINI A., SAETTI A. & SERINA I. (2004). Planning with numerical expressions in lpg. In *European Conference on Artificial Intelligence (ECAI)*, p. 667–671.

HASLUM P. & GEFFNER H. (2001). Heuristic planning with time and resources. In *Proceedings of the Sixth European Conference on Planning*, p. 121–132.

HOFFMANN J. (2003). The metric-ff planning system : Translating "ignoring delete lists" to numeric state variables. *JAIR*, **20**, 291–341.

KAUTZ H. & SELMAN B. (1999). Unifying SAT-based and graph-based planning. In *IJCAI*, p. 318–325.

KAUTZ H., SELMAN B. & HOFFMANN J. (2006). Satplan : Planning as satisfiability. In *Abstracts of the 5th International Planning Competition*.

LARROSA J. (2002). Node and arc consistency in weighted csp. In *AAAI*, p. 48–53.

LARROSA J. & SCHIEX T. (2003). In the quest of the best form of local consistency for weighted csp. In *IJCAI*, p. 239–244.

LOPEZ A. & BACCHUS F. (2003). Generalizing graphplan by formulating planning as a csp. In *IJCAI*, p. 954–960.

MITTAL S. & FALKENHAINER B. (1990). Dynamic constraint satisfaction problems. In *National Conference on Artificial Intelligence (AAAI)*, p. 25–32.

REFANIDIS I. & VLAHAVAS I. P. (2003). Multiobjective heuristic state-space planning. *Artificial Intelligence*, **145**(1-2), 1–32.

SCHIEX T., FARGIER H. & VERFAILLIE G. (1995). Valued constraint satisfaction problems : Hard and easy problems. In *IJCAI*, p. 631–639.

VAN BEEK P. & CHEN X. (1999). Cplan : A constraint programming approach to planning. In *AAAI*, p. 585–590.

Modélisation de la capacité d'un agent intentionnel en fonction de ses activités

Karl Devooght[1,2]

[1] France Télécom R&D
2, avenue Pierre Marzin, 22300 Lannion

[2] LLI-IRISA, 6, rue de Kerampont, 22300 Lannion
karl.devooght@orange-ftgroup.com

Résumé : Depuis plusieurs années, les systèmes d'agents intelligents, en particulier les agents dit BDI (*t Belief-Desire-Intention*), font l'objet d'un intérêt croissant. Dans cet article, nous proposons une logique modale de la capacité s'appuyant sur les modèles logiques BDI. En particulier, la capacité d'un agent est comprise comme la capacité à réaliser une activité pour atteindre un but. Une *activité* est un sous-modèle BDI résolvant une classe de problèmes particuliers. Dans cette optique, nous proposons un modèle d'agent dans lequel les activités de ce dernier sont organisées en contextes. La capacité permet alors d'établir le lien entre un agent et ses activités. Pour cela, nous proposons une structure sémantique ainsi qu'un système axiomatique adaptés à la logique de la capacité proposée. **Mots-clés** : capacité, activité, agent intelligent .

---

1 Introduction

Depuis plusieurs années, les systèmes d'agents intelligents font l'objet d'un intérêt grandissant. Les agents intelligents permettent de modéliser des processus complexes (par exemple, le contrôle du traffic aérien ou les services de renseignements en dialogue naturel). Les approches agent combinent habituellement logique symbolique et approche cognitive. En particulier, le paradigme d'agent dit BDI (Belief-Desire-Intention) a été largement étudié. Il présente des bases théoriques robustes comme, par exemple, les travaux de (Cohen & Levesque, 1990), de (Rao & Georgeff, 1995), de (Sadek, 1994)). Par ailleurs, plusieurs systèmes implémentés s'appuient sur ce paradigme.

Afin de considérer la dynamique d'un agent, les modèles logiques BDI prennent généralement en compte la notion d'action. Cela permet à l'agent aussi bien de réagir à court terme que de poursuivre des buts à plus long terme. Pour cela, les modèles d'agents doivent prendre en compte un large ensemble de connaissances et de descriptions d'activités. De plus, l'agent doit s'adapter à l'évolution des situations dans lesquelles il est impliqué. Par ailleurs, pour une situation particulière, seul un sous-ensemble des connaissances et des activités est généralement nécessaire à l'agent.

Comme la plupart des modèles d'agents BDI ne partitionnent pas explicitement les connaissances et les descriptions d'activités de l'agent, deux problèmes importants surviennent. Premièrement, plus la quantité de connaissances et d'activités considérée est grande, plus il est difficile de maintenir un modèle d'agent consistant. Deuxièmement, le manque de modularité au sein des modèles BDI oblige l'agent à considérer à chaque instant toutes ses connaissances. Ainsi, pour garantir la cohérence d'un agent et l'accès aisé aux connaissances pertinentes, on est contraint de limiter les capacités de l'agent.

En pratique, (Busetta *et al.*, 1999) ont proposé de modulariser chaque description d'activité de l'agent. L'idée est de les considérer comme des "modules mentaux" à part entière permettant de résoudre une classe de problème particulier. Pour établir le lien entre l'agent et ses activités modularisées, ils introduisent une notion de capacité. Autrement dit, la capacité comme attitude mentale particulière de l'agent prend en paramètre une activité, et réfère au "module mental" de celle-ci.

Dans ce papier, nous formalisons la notion de capacité suivant cette idée. Elle est comprise comme la capacité d'un agent à réaliser une activité pour atteindre un certain but. Nous proposons en particulier un cadre sémantique dédié aux modèles d'agents BDI, composé de contextes. Sous cet angle, notre opérateur de capacité signifie que l'agent accède au contexte de l'activité eét que la description de l'activité permet d'atteindre un but. Nous montrons par ailleurs l'intérêt de la capacité proposée notamment en termes de généricité.

En section 2, nous introduisons un bref état de l'art des modèles de capacité existants. Ensuite, nous présentons informellement notre modèle de la capacité. En section 3, un cadre sémantique basé sur la sémantique de Kripke est proposé consistant à partitionner les modèles d'agent en contexte. Dans cette optique, une logique de la capacité étendant une logique BDI est considérée. En particulier, les propriétés caractérisant la logique sont précisées. Nous discutons en section 4 des conséquences conceptuelles induites par le modèle d'agent proposé.

2 Capacité et raisonnement

2.1 État de l'art

Modéliser la notion de capacité implique d'expliciter les paramètres qui la composent. Il est évident que, en tant qu'acteur, l'agent lui-même ou, dans certains cas, le groupe d'agents représente un de ces paramètres. Il est cependant plus difficile de déterminer la nature de ce sur quoi porte la capacité. Par exemple, lorsque nous affirmons que *Ronaldo est capable de marquer un but*, est-il entendu que Ronaldo est capable de réaliser une action particulière (ou un plan particulier) consistant à tirer au but ? Ou est-il entendu que Ronaldo est capable de garantir qu'un but soit marqué indépendamment des moyens mis en jeu (en passant la balle, par exemple, à un coéquipier mieux placé que lui) ? Le choix d'une des deux possibilités dépend fortement du modèle d'action considéré. Autrement dit, définir un modèle de capacité exige au préalable d'opter pour un modèle d'action particulier.

Dans la littérature, il existe deux approches principales pour modéliser l'action. La première consiste à ne pas représenter explicitement les actions dans le langage logique. Dans ce cadre, les travaux sur la capacité expriment alors le fait qu'un agent parvient à un état du monde où un but est atteint (ou une propriété est vraie) (cfr. les travaux de (Brown, 1988), de (Horty & Belnap, Jr., 1995)). La seconde, au contraire, prend en compte explicitement les actions de l'agent. C'est, par exemple, le cas de la logique dynamique. L'accent est alors mis sur le fait qu'un agent est capable de réaliser une certaine action atomique ou complexe (van der Hoek *et al.*, 1994; van Linder *et al.*, 1998). Par ailleurs, en connaissant les effets d'une action, une forme dérivée de capacité peut être modélisée. Il s'agit de considérer l'effet d'une action comme une propriété vraie dans l'état du monde atteint après l'exécution de l'action.

En outre, le sens donné à la capacité doit être également désambiguisé. On distingue fréquemment la capacité *générique* de la capacité *occasionnelle*. La capacité générique réfère à l'ensemble des connaissances[1] permettant, par exemple, de marquer un but. Ce type de capacité est indépendant des états intellectuels et physiques ainsi que de l'état du monde. La capacité occasionnelle (appelée aussi *opportunité* [2]) réfère au contraire à la situation courante et aux conditions satisfaites dans celle-ci autorisant à appliquer une capacité générique particulière. Si Ronaldo possède le ballon et se trouve seul face au gardien, on comprend qu'il a l'opportunité de marquer un but.

Techniquement, la considération logique de ces deux types de capacité est complexe. Il est en effet difficile de les définir sans que chacun fasse référence à l'autre (cf. (Singh, 1991)). C'est pourquoi de nombreux travaux proposent une modélisation de la capacité combinant capacité générique et capacité occasionnelle. A notre connaissance, seuls les travaux de van Linder et al. (van Linder *et al.*, 1998) ainsi que ceux de Cholvy et al. (Cholvy *et al.*, 2005) introduisent une séparation claire.

2.2 Capacité et activité

Dans cet article, nous nous intéressons aux modèles de capacité générique dans lesquels l'action est explicitement représentée dans le langage. De tels modèles associent généralement la capacité d'un agent à sa connaissance de plans. Un *plan* se caractérise d'une part, par la connaissance d'un ensemble de préconditions et d'effets caractérisant *a priori* et *a posteriori* l'exécution du plan, et d'autre part, par la description de l'exécution du plan (i.e. appelée aussi le corps du plan). L'exécution d'un plan [3] se définit souvent comme une séquence particulière d'actions. Une capacité générique d'agent est alors comprise comme une capacité à réaliser un plan. La capacité à réaliser un plan s'interprète comme la connaissance par l'agent de la séquence d'actions caractérisant l'exécution du plan. Dans notre exemple, marquer un but pourrait être associé à un plan consistant à dribbler un adversaire puis à tirer au but.

La question qui se pose est la suivante : suffit-il d'un plan pour exprimer une capacité générique ? En pratique, Busetta et al. Busetta *et al.* (1999) ont montré que ce n'est pas le cas. Selon eux, déterminer une capacité générique implique au moins la prise en

[1] Ces connaissances sont parfois appelées des connaissances *procédurales*

[2] Dans cet article, les deux termes sont utilisés indifféremment.

[3] Les plans considérés sont complètement spécifiés et déterministes.

compte de connaissances [4], de plans (i.e. éventuellement plus d'un plan) et de règles de comportements (e.g., guidant le choix du plan). Dans cette optique, être capable de marquer un but nécessiterait certaines connaissances (e.g., où suis-je sur le terrain, où se trouvent mes coéquipiers), plusieurs plans (e.g., dribbler un adversaire puis tirer au but ou tirer directement au but), des règles de comportement (e.g., si je suis en position de tirer, je tire au but). L'ensemble de ces éléments caractérise ce que Busetta et al. nomme un composant fonctionnel. Un composant fonctionnel regroupe tous les élements permettant de résoudre une classe particulière de problème. Dans ce papier, nous nommons cela une *activité* de l'agent.

Dans le contexte des modèles d'agent BDI, à quoi correspond une activité d'agent ? Il existe en fait une analogie profonde entre la modélisation d'un agent et la modélisation d'une activité d'agent. En effet, modéliser un agent BDI particulier, c'est proposer dans une certaine mesure un modèle de son activité globale (e.g., l'activité de contrôler le traffic aérien). Par conséquent, on peut penser que modéliser une activité équivaut à définir un sous-modèle BDI permettant de résoudre une certaine classe de problème. Sous cette hypothèse, il s'agit néanmoins de souligner quelles sont les différences entre le modèle d'agent et le modèle de ses activités. Premièrement, l'agent possède une *vue* sur ses activités. Dans notre approche, cela se traduit par le biais de la capacité. Un agent capable de réaliser une activité est avant tout un agent ayant certaines informations sur le modèle de cette activité. Deuxièmement, l'agent est à un niveau global alors que l'activité est toujours à un niveau local (i.e., caractérisée par la classe de problème qu'elle est censée résoudre). En dernier lieu, nous distinguons la notion de *réalité* mentale de la notion de *projection* mentale. La réalité mentale de l'agent détermine les états mentaux "réels" de l'agent, c'est-à-dire ceux liés à son passé, à sa situation présente et à ses possibles situations futures. Autrement dit, la réalité mentale est la partie *située* du modèle d'agent, c'est-à-dire la partie composée d'éléments effectifs constituant la perception du monde que possède l'agent. La plupart des travaux envisage un modèle d'agent de cette manière. Une projection mentale réfère à des états mentaux imaginés par l'agent (qui ne réfère pas directement à la réalité mentale de l'agent). Par exemple, Ronaldo sait qu'il est en train de jouer un match de football (ce qui représente sa réalité mentale) et à la mi-temps, dans un moment d'égarement, il peut s'imaginer en train de cuisiner sa recette préférée (qui est de l'ordre de la projection mentale). Nous pensons que les modèles d'activité à travers la notion de capacité doivent être perçus par l'agent comme des projections mentales.

La notion de capacité que nous introduisons permet d'établir un lien entre la réalité mentale de l'agent et ses activités considérées comme des projections mentales. Cette vision nous amène à une interprétation plus intuitive de la capacité générique. De nombreux travaux interprètent la capacité générique comme une analyse des états futurs de l'agent envisageant ce qu'il peut faire. Cependant, cette position n'est pas adaptée dans de nombreux cas. Par exemple, imaginons un homme condamné à la prison à vie. Si nous lui demandons naïvement : "Es-tu capable de rouler à vélo ?". Il n'y aurait rien de choquant à ce qu'il nous reponde par l'affirmative. Il est fortement vraisemblable qu'il n'analyse pas cela sur la base de son avenir puisqu'à priori il ne roulera certainement jamais plus à vélo. Par contre, envisager la même situation sous l'angle de la projection

[4]Les connaissances sont ici des connaissances *factuelles*.

mentale offre une vision plus adéquate.

En résumé, notre considération de la capacité générique porte non plus sur l'action mais sur l'activité. Une activité se caractérise par un sous-modèle BDI résolvant une classe de problèmes particuliers. De plus, être capable de réaliser une activité ne s'interprète pas comme une analyse des états mentaux futurs de l'agent mais plutôt comme l'analyse du sous-modèle associé à l'activité et projeté mentalement par l'agent. Nous introduisons ainsi un modèle formel dans lequel les activités de l'agent sont organisées en modules.

3 Une logique de la capacité

Dans cette section, nous définissons une logique modale propositionnelle étendant une logique BDI "classique". En particulier, nous considérons les opérateurs modaux de croyance B, de but (ou désir) D ainsi que l'opérateur d'intention I. Cette approche de la logique BDI s'inspirent des travaux de Rao et Georgeff (Rao & Georgeff, 1995). Un opérateur $C(\alpha, \phi)$ est introduit. Il signifie que l'agent est capable de réaliser l'activité α pour atteindre le but ϕ. Par ailleurs, nous nous situons dans un cadre mono-agent[5].

Syntaxe.

Un langage $\mathcal{L}$ est introduit. Il contient un ensemble de propositions atomiques $\mathcal{P}$, un ensemble d'actions $\mathcal{ACT}$ ainsi qu'un ensemble d'activités primitives $\mathcal{A}$. Une formule bien formée ϕ de $\mathcal{L}$ respecte la grammaire suivante :

$$\phi ::= p \mid \neg\phi \mid \phi \rightarrow \phi \mid B\phi \mid D\phi \mid I\phi \mid C(\alpha, \phi)$$
$$a ::= e \mid a; a$$

Notons que p$\in\mathcal{P}$, e$\in\mathcal{ACT}$ et $\alpha\in\mathcal{A}$. Les connecteurs logiques $\wedge$ et $\vee$ sont pris en compte et sont définis de manière habituelle.

Sémantique.

Les modèles sémantiques traditionnels pour les logiques modales s'appuient sur des structures de modèles basées sur des cadres sémantiques de Kripke. Un cadre sémantique de Kripke se compose d'un ensemble d'états (ou de mondes possibles) W et d'un ensemble de relations binaires R sur W (i.e. $W\times W$). Nous proposons une forme dérivée de ces cadres. Nous considérons des cadres sémantiques composés de :

- un ensemble C de contextes. Un contexte c est un tuple (W_c,R_c) où :
  - W_c est un ensemble d'états (ou mondes possibles)
  - R_c est un ensemble de relations binaires sur W_c
- un ensemble R_{ic} de relations binaires "inter-contextes" de la forme $W_c\times W_{c'}$ (où c$\neq$c').

Remarquons que ces cadres se présentent comme une forme restreinte des cadres classiques (W,R) où :

[5] C'est pourquoi, les opérateurs modaux ne sont pas relatifs à un agent particulier

– $W = \bigcup_{c \in C} W_c$
– $R = R_{ic} \bigcup (\bigcup_{c \in C} R_c)$

La "réalité mentale" de l'agent est représentée par un contexte particulier c_{RM}. Chaque activité $\alpha \in \mathcal{A}$ est associée à un contexte noté c_α. Ces contextes représentent la réalisation des activités. En particulier, l'ensemble des états W_{c_α} de chaque contexte c_α se compose d'un ensemble d'états initiaux S_{c_α} ainsi qu'un ensemble d'états finaux F_{c_α}. Autrement dit, nous supposons que la réalisation d'une activité débute à partir d'un état $s \in S_{c_\alpha}$ et se termine dans un état $f \in F_{c_\alpha}$.

L'ensemble des relations d'accessibilité R pour chaque contexte $c \in$C se compose de :
– une relation d'accessibilité de croyance $\mathcal{B}_c$,
– une relation d'accessibilité de but $\mathcal{D}_c$,
– une relation d'accessibilité d'intention $\mathcal{I}_c$,
– une relation d'accessibilité événementielle $R_c(a)$ pour chaque action a$\in \mathcal{ACT}$.

Nous supposons que chaque relation $R_c(a)$ est contrainte sous les hypothèses de passé linéaire [6], de futur ramifié [7] et de déterminisme [8]. Par ailleurs, pour toute séquence d'action $a1; a2$, nous notons par abus de langage $R_c(a1; a2)$. Si nous avons $R_c(a1; a2)$(w,w"), cela est équivalent à dire qu'il existe w' tel que tel que $R_c(a1)$(w,w') et $R_c(a2)$(w',w").

L'ensemble R_{ic} des relations d'accessibilité inter-contextes correspond dans notre approche à un ensemble de relations d'accessibilité $R(\alpha)$ pour chaque activité $\alpha \in \mathcal{A}$. Ces relations se présentent sous la forme $W_{c_{RM}} \times S_{c_\alpha}$ reflétant la possibilité d'accéder à certains états initiaux des contextes c_α, à partir d'états $W_{c_{RM}}$ du contexte c_{RM}. Ces relations d'accessibilité caractérisent ce que nous avons nommé les "projections mentales" de l'agent.

Chaque contexte possède son propre vocabulaire. Un *vocabulaire* est un sous-ensemble de propositions atomiques pour lesquelles une interprétation sémantique est possible. Une proposition atomique n'appartenant pas au vocabulaire d'un contexte c est supposée faux dans chacun des états de c.

Ainsi, les modèles M dont nous nous servons pour interpréter le langage L sont des tuples (C,R_{ic},V,π) où :
– C est l'ensemble de contextes composé de c_{RM} et des contextes c_α pour chaque activité $\alpha \in \mathcal{A}$,
– R_{ic} est l'ensemble de relations d'accessibilité composé de $R_{ic}(\alpha)$ où $\alpha \in \mathcal{A}$,
– V est l'ensemble des vocabulaires $V_c \subseteq \mathcal{P}$ pour $c \in C$,
– π est l'ensemble des fonctions d'interprétation π_c : $V_c \times W \rightarrow$Bool pour $c \in C$.

Satisfiabilité.

Nous pouvons à présent définir la relation de satisfiabilité $\models$. Soit ϕ une formule bien formée (fbf) de $\mathcal{L}$, M un modèle (C,R,V,π), w_c appartenant au contexte c de C. On note M,$w_c \models \phi$ pour signifier que le monde w_c satisfait (ou rend vraie) la fbf ϕ. Elle est *contextuellement satisfaite* ssi M,$w_c \models \phi$ pour tout $w_c \in W_c$. Elle est valide ssi M,$w \models \phi$

[6] C'est-à-dire, pour tout état w, il n'existe qu'une et une seule action a et un état w'tel que $R_c(a)$(w',w).
[7] C'est-à-dire, pour chaque état w, il existe une action a et un état w' tel que $R_c(a)$(w,w').
[8] C'est-à-dire, si $R_c(a)$(w,w') et $R_c(a)$(w,w"), alors w'=w"

pour tout couple (M,w). La relation de satisfiabilité $\models$ est définie récursivement comme suit :

- M,$w_c\models$ p ssi p$\in V_c$ et π_c(p,w_c) = vrai ;
- M,$w_c\models \neg\phi$ ssi M,$w_c\not\models \phi$
- M,$w_c\models \phi \wedge \psi$ ssi M,$w_c\models \phi$ et M,$w_c\models \psi$
- M,$w_c\models B\phi$ ssi pour tout w'_c tq $B_c(w_c,w'_c)$, on a M,$w'_c\models\phi$
- M,$w_c\models D\phi$ ssi pour tout w'_c tq $D_c(w_c,w'_c)$, on a M,$w'_c\models\phi$
- M,$w_c\models I\phi$ ssi pour tout w'_c tq $I_c(w_c,w'_c)$, on a M,$w'_c\models\phi$

Avant de donner l'interprétation sémantique de l'opérateur de capacité C, nous introduisons la notion de stratégie. Une *stratégie* d'activité, noté $\sigma(a,s,f)$, pour une activité α associée à un contexte c_α est une action primitive ou une séquence d'actions a, telle que $R(a)_\alpha$(s,f) où $s\in S_{c_\alpha}$ et $f\in F_{c_\alpha}$. Autrement dit, une stratégie d'activité est une action particulière partant d'un état initial de l'activité et atteignant un de ses états finaux. L'opérateur de capacité C est interprété sémantiquement comme suit :

- M,$w_c\models$ C(α,ϕ) ssi
  - pour tout $s_{c_\alpha}\in S_{c_\alpha}$ tq $R_{ic}(\alpha)(w_c,s_{c_\alpha})$ et
  - pour toute stratégie σ(a,s_{c_α},f_{c_α}), on a M,$f_{c_\alpha}\models \phi$.

Remarquons que cette formule ne peut éventuellement être satisfaite que pour w_c où c=c_{RM}. Autrement dit, nous supposons que la capacité est seulement satisfiable dans les états mentaux "réels" de l'agent.

Attardons nous maintenant sur l'interprétation sémantique de cet opérateur. Informellement, cet opérateur signifie que l'agent se projette mentalement dans un état initial particulier de l'activité α. A partir de cet état, toutes les stratégies (projetées) de l'agent atteignent un état dans lequel ϕ est vraie. Notre opérateur modal de capacité s'apparente à un opérateur de nécessité. La nécessité est exprimée par l'accès aux états initiaux de l'activité. De plus, la nécessité porte sur les états finaux de l'activité et notamment sur le fait qu'une propriété doit être vraie dans chacun de ces états. Le caractère nécessaire de la capacité est critiqué dans certaines approches. L'argument le plus souvent avancé est qu'envisager une réussite totale est une hypothèse irréaliste. Dans notre approche, l'ensemble des états finaux F_{c_α} pour une activité α doit être compris comme les états de *succès* de l'activité. Dès lors, exiger qu'une propriété soit vraie dans tous les états de succès est une hypothèse acceptable.

Axiomes.

L'opérateur de capacité que nous proposons est un opérateur modal normal. Les propriétés de notre opérateur caractérisent une logique modale KD. En particulier, l'opérateur est clos sous l'implication logique, la conjonction et la disjonction. Il est par ailleurs consistant. La consistance de l'opérateur signifie, par exemple, qu'être capable de réaliser une activité qui me rend riche implique le fait de ne pas être capable de réaliser cette même activité avec, pour but, d'être pauvre. Ainsi, l'opérateur de capacité admet donc les axiomes suivants :

$$C(\alpha, \phi) \wedge C(\alpha, \phi \Rightarrow \psi) \Rightarrow C(\alpha, \psi) \tag{K}$$
$$C(\alpha, \phi) \Rightarrow \neg C(\alpha, \neg \phi) \tag{D}$$
$$C(\alpha, \phi \wedge \psi) \Leftrightarrow C(\alpha, \phi) \wedge C(\alpha, \psi)$$
$$C(\alpha, \phi \vee \psi) \Leftrightarrow C(\alpha, \phi) \vee C(\alpha, \psi)$$

En particulier, la validité de l'axiome D est due à la nature des relations d'accessibilité $R_{ic}(\alpha)$ et R_c(a). Nous imposons, en effet, que $R_{ic}(\alpha)$ soit sérielle, c'est-à-dire, pour tout état $w_{c_{RM}}$, il existe un état s_{c_α} tel que $R(\alpha)(w_{c_{RM}}, s_{c_\alpha})$. Par ailleurs, les états finaux atteints (à travers les relations d'accessibilité R_c(a)) à partir d'un état initial particulier s_{c_α} sont, par nature, des états consistants.

4 Conséquences conceptuelles

4.1 Analyse locale

Modéliser la capacité sur la base d'une sémantique contextuelle offre deux avantages importants. Premièrement, elle permet de décrire et d'analyser *localement* une activité par le biais de la capacité. L'intérêt logique est de ne pas ou plus chercher à établir un système axiomatique au modèle d'agent qui soit adapté à toutes les activités de l'agent. Par contre, il est possible d'établir les systèmes axiomatiques convenant individuellement à chacune des activités. Dans notre approche, cela correspond à identifier ce qui est contextuellement satisfait. Deuxièmement, la capacité comme attitude mentale abstrait la description d'une activité. Ainsi, un état "réel" d'agent donné ne prend pas en compte chaque description détaillée d'activité dans sa phase de raisonnement. Par ailleurs, l'agent est préservé d'un effort éventuel de raisonnement qu'il fournirait si chacune des descriptions d'activités étaient "physiquement" présentes dans chacun de ces états mentaux.

Pour illustrer le premier point, prenons un exemple. Rao et Georgeff dans (Rao & Georgeff, 1995) proposent un système axiomatique pour la logique BDI. Le système contient huit axiomes. Ils introduisent un neuvième axiome exprimant les conditions sous lesquelles un agent s'engage à accomplir une intention. Trois alternatives sont considérées : l'agent poursuit son intention (1) tant qu'il ne croit pas que le but est atteint (*"blind commitment"*), (2) tant qu'il croit que le but n'est pas atteint et que le but est possible à réaliser (*"single-mind commitment"*) ou (3) tant qu'il croit que le but n'est pas atteint et que ce dernier est encore un but de l'agent (*"open mind commitment"*). Pour eux, le choix d'une alternative dépend du profil d'agent à définir.

Dans notre approche, nous pensons que cela dépend non seulement de l'agent mais particulièrement de l'activité de l'agent à considérer. Par exemple, imaginons que Ronaldo ait son brevet de secouriste. Supposons qu'il soit capable de marquer des buts et de sauver une vie. Intuitivement, ces deux activités n'adhèrent pas au même type d'engagement. D'un côté, pour sauver une vie, Ronaldo poursuivra son intention de la sauver jusqu'au moment où il y arrive. L'axiome *"blind commitment"* est dans ce cas adapté. D'un autre côté, s'engager à marquer un but semble être contraint par le fait que cet objectif soit encore réalisable. L'axiome *"single-mind commitment"* semble mieux convenir. Ces axiomes sont dès lors parfaitement envisageables au sein d'un même modèle d'agent. Seulement, l'ensemble de ces axiomes ne sont pas consistants mais, par contre, sont contextuellement satisfaits dans le cadre des activités données en exemple.

4.2 Opportunité

Comme nous l'avons mentionné en section 2, la capacité générique accompagne automatiquement la capacité occasionnelle. Bien que nous n'ayons pas formalisé cette dernière, nous devons évoquer quelques perspectives intéressantes.

Traditionnellement, quand elle possède une action comme paramètre, la capacité occasionnelle est caractérisée par le fait qu'il existe une exécution de l'action. Supposons que w soit un état du monde et a une action. En termes logiques, un agent a la capacité occasionnelle de réaliser l'action a à partir de l'état w si et seulement s'il existe un état du monde w' résultant de l'éxécution de a. Cependant, nous pensons que la capacité occasionnelle ne doit pas être liée à quelconque état résultant puisque, par définition, celle-ci réfère "strictement" à l'état du monde *avant* l'exécution d'une action sans présupposer *a priori* l'existence d'une telle exécution. Par exemple, une personne peut avoir l'opportunité de jouer au tennis car elle possède une raquette et un adversaire, et se trouve sur un terrain de tennis. Supposons qu'elle ne sache pas comment jouer au tennis i.e. qu'elle n'ait pas la compétence. Dans ce cas, elle n'a donc *a priori* pas "conscience" de l'existence d'une exécution particulière d'action permettant de jouer au tennis.

Dans notre approche, la capacité générique est indépendante de l'état mental "réel" dans lequel se trouve l'agent. En particulier, elle réfère à un état initial de l'activité projeté mentalement par l'agent. Dans cette optique, la capacité occasionnelle peut être interprétée comme la compatibilité entre l'état mental "réel" et l'état mental initial de l'activité. Cela signifie, par exemple, que avoir l'opportunité de jouer au tennis résulte de la comparaison entre un état courant "réel" et un état initial de l'activité "jouer au tennis" imaginé par l'agent. Il s'agit alors de définir quand deux états sont compatibles. Une solution extrême est d'exiger que les états mentaux soient identiques. Une solution plus raisonnable est de définir un ensemble fini de conditions devant être satisfaites dans chacun des deux états. De cette manière, les notions de capacité générique et de capacité occasionnelle sont indépendantes. Avoir l'opportunité n'influence pas le fait d'avoir ou non la capacité générique (et réciproquement).

5 Conclusion

Nous avons proposé dans cet article un modèle logique de la capacité générique. La notion d'activité a été introduite et a été considérée comme la nature de ce sur quoi porte la capacité. Chaque activité est associée à un sous-modèle BDI explicitant "mentalement" sa réalisation. Le modèle général d'agent est ainsi décomplexifié. Par ailleurs, la capacité comme attitude mentale permet d'établir un lien entre l'agent et ses activités.

Formellement, la capacité générique est comprise comme la capacité générique à réaliser une activité pour atteindre un certain but. Un système axiomatique a été proposé caractérisant notre logique de la capacité comme une logique modale KD.

De plus, nous avons discuté de deux importantes conséquences conceptuelles de notre modèle. Premièrement, une logique d'agent BDI devrait être analysée en fonction des activités qu'un agent est amené à réaliser. Il s'agit alors de déterminer ce qui est vrai pour *une* activité et non pour l'ensemble des activités. Deuxièmement, la capacité générique est indépendante des états mentaux "réels" (intellectuels ou physiques) de l'agent. En particulier, nous avons proposé des pistes pour définir la capacité occasionnelle dans le cadre du modèle logique proposé.

En termes de perspectives, plusieurs pistes sont ouvertes. Premièrement, avec l'opérateur de capacité présenté, la réalisation du but est *nécessairement* vrai alors que, dans certaines activités, certains buts sont simplement *possibles* selon le cas de figure. Par exemple, Ronaldo peut marquer un but avec son pied gauche (même s'il est droitier). Si son objectif est de marquer un but avec son pied gauche, il est intuitif que celui-ci est possiblement (et non nécessairement) atteignable. Deuxièmement, comme pour les modèles d'actions, il serait intéressant que nous puissions paramétrer une activité. Ainsi, nous pourrions distinguer un type d'activité et une instance d'activité (i.e. un type d'activité dont les paramètres sont instanciés). Troisièmement, nous avons restreint la satisfiabilité d'une capacité aux états "réels" de l'agent. Il est envisagé de généraliser cela à n'importe quel état. Dans ce cas, cela signifie que, dans les contextes associés aux activités, il serait possible de faire référence à une autre capacité (et, par conséquent, à un autre contexte).

Références

BROWN M. A. (1988). On the logic of ability. *Journal of Philosophical Logic*, **17**, 1–26.

BUSETTA P., HOWDEN N., RÖNNQUIST R. & HODGSON A. (1999). Structuring BDI agents in functional clusters. In N. R. JENNINGS & Y. LESPÉRANCE, Eds., *ATAL*, volume 1757 of *Lecture Notes in Computer Science*, p. 277–289 : Springer.

CHOLVY L., GARION C. & SAUREL C. (2005). Ability in a multi-agent context : A model in the situation calculus. In F. TONI & P. TORRONI, Eds., *CLIMA VI*, volume 3900 of *Lecture Notes in Computer Science*, p. 23–36 : Springer.

COHEN P. R. & LEVESQUE H. J. (1990). *Intention is choice with commitment*. Toronto : Computer Science Dept., University of Toronto.

HORTY J. F. & BELNAP, JR. N. D. (1995). The deliberative STIT : A study of action, omission, and obligation. *Journal of Philosophical Logic*, **24**(6), 583–644.

RAO A. S. & GEORGEFF M. P. (1995). BDI agents : From theory to practice. In *ICMAS*, p. 312–319.

SADEK M. D. (1994). Communication theory = rationality principles + communicative act models.

SINGH M. P. (1991). A logic of situated know-how. In *IJCAI-91 Proceedings* : IJCAI.

VAN DER HOEK W., VAN LINDER B. & MEYER J.-J. C. (1994). A logic of capabilities. *Proceedings of Laboratory for Foundations of Computer Science*, **813**, 366–378.

VAN LINDER B., VAN DER HOEK W. & MEYER J.-J. C. (1998). Formalising abilities and opportunities of agents. *Fundamenta Informatica*, **34**.

Formules CNF et Graphes

Said Jabbour

CRIL-CNRS, Université d'Artois, 62307 Lens Cedex, France
jabbour@cril.univ-artois.fr

Résumé :
Nous proposons dans cet article une nouvelle représentation sous forme de graphe d'une formule CNF. Elle étend le graphe d'implication 2-SAT au cas général. Chaque clause est représentée comme un ensemble (conditionnel) d'implications et codée avec plusieurs arcs étiquetés contenant un ensemble de litéraux, appelé contexte. Cette nouvelle représentation permet d'étendre quelques caractéristiques intéressantes de l'algorithmique 2-SAT. Parmi elles, la résolution classique est reformulée en utilisant la fermeture transitive d'un graphe. Les chemins entre les noeuds permetent de donner une façon originale pour calculer les conditions minimales sous lesquels un litéral est impliqué.Deux utilisations concrètes de ce graphe sont présentés. La première concerne l'extraction des ensembles 2-SAT strong backdoor et la seconde consiste en une technique de pré-traitement des formules CNF. Des résultats expérimentaux prometteurs sont obtenus sur plusieurs classes d'instances issues des dernières compétitions.

1 Introduction

Le problème SAT, qui consiste à vérifier si un ensemble de clauses est satisfaisable ou pas, est central en informatique et dans le domaine de l'intelligence artificielle. Il est notamment utilisé pour la preuve de théorème, la planification, le raisonnement non monotone, la vérification de circuits. Durant ces deux dernières décennies, plusieurs approches ont été proposées pour résoudre des instances difficiles en utilisant des algorithmes complets ou incomplets. La recherche locale (e.g. [24]), les variantes élaborées à partir de la procédure DPLL de Davis-Putnam-Loveland-Logemann [7] (e.g. [22, 12]) permettent maintenant de résoudre des instances de plus en plus grandes issues du monde réel. La plupart des solveurs complets sont basés sur la technique du backtrack initiée par Davis Putnam Logemann Loveland (DPLL). Ces algorithmes de base intègrent des techniques de plus en plus efficaces comme l'apprentissage, la propagation de contraintes, le pré-traitement, l'exploitation des symétries, etc. L'impact de ces différentes améliorations dépend du type des instances à résoudre. Par exemple, l'apprentissage est plus efficace sur les instances issues du monde réel que sur les instances aléatoires. Un autre aspect important lié à l'efficacité des solveurs SAT concerne le codage des problèmes traditionnellement transformées sous forme normale conjonctive (CNF). Cependant, le codage sous forme CNF induit une perte des connaissances structurelles qui sont mieux exprimées dans d'autres formalismes et qui peuvent permettre une résolution plus efficace [16, 14, 26]. Pour exploiter ces connaissances structurelles, des travaux récents ont été proposés. Quelques uns utilisent la forme étendue des formules booléennes (nonCNF [26], contraintes pseudo booléennes [9]) pour le codage des formules. Tandis que d'autres essayent de récupérer et/ou déduire des propriétés structurelles à partir des formules CNF (symétries [1], dépendances fonctionnelles [14], équivalences [17]). Finalement, différents graphes ont été élaborés pour présenter ou modéliser ou bien même résoudre des instances SAT. Ce dernier type d'approche est motivé par la visualisation de structures [25], la décomposition [6], la propagation [18], l'apprentissage (à partir du graphe d'implication) [21, 22].

Dans cet article, un nouvelle représentation en graphe est proposée. Elle étend de manière originale le graphe d'implication pour les formules binaires (2-SAT) au cas général. Chaque clause est représentée par un ensemble d'implications (conditionnelles) et codée avec différents arcs étiquetés avec un ensemble de litéraux, appelé contexte (ou condition). Cette nouvelle représentation permet d'étendre quelques caractéristiques intéressantes de l'algorithmique 2-SAT. Parmi elles, la résolution classique est reformulée en utilisant la fermeture transitive du graphe. Les chemins entre les noeuds opposés permettent d'avoir des conditions pour la définition du backbone i.e. les conditions minimales sous lesquelles un litéral est impliqué. Cependant, dans le cas général, trouver l'ensemble minimal sous lequel un litéral est impliqué est intraitable. Clairement, cette nouvelle représentation admet plusieurs utilisations intéressantes.

Dans cet article, deux manières d'exploiter cette représentation sont décrites. La première concerne le calcul des ensembles strong backdoor et la seconde est une technique de pré-traitement des formules CNF. Le reste de l'article est organisé comme suit. Après quelques définitions préliminaires, le graphe associé à une formule CNF est décrit. Une description formelle de ces caractéristiques est détaillée. Plus particulièrement la résolution est reformulée en utilisant la fermeture transitive du graphe. Deux utilisations pratiques de cette représentation sont ensuite proposées. Nous terminons par une étude expérimentale menée sur des instances issues des dernières compétitions SAT.

2 Préliminaires

Soit $\mathcal{B}$ un langage booléen (i.e. propositionnel) de formules construites classiquement, en utilisant les connecteurs usuels ($\vee$, $\wedge$, $\neg$, $\rightarrow$, $\leftrightarrow$) et un ensemble de variables propositionnelles. Une *formule CNF* Σ est un ensemble (interprété comme une conjonction) de *clauses*, et une clause est un ensemble (interprété sous forme disjonctive) de *litéraux*. Un litéral est une variable ou sa négation. Pour un litéral x (resp. une clause c), $\mathcal{V}(x)$ (resp. $\mathcal{V}(c)$) correspond à la variable propositionnelle associé au litéral x (resp. l'ensemble de variables associées à c). On note $\mathcal{V}(\Sigma)$ (resp. $\mathcal{L}(\Sigma)$) l'ensemble des variables (resp. litéraux) apparaissant dans Σ. La taille d'une clause c, noté $|c|$, est égale au nombre de ces litéraux . Une *clause unitaire* est une clause de taille un. Un *litéral unitaire* est l'unique litéral d'une clause unitaire. Une clause est dite *clause binaire* si elle contient exactement deux litéraux. Une *clause* est *Horn* si elle contient au plus un litéral positif. On appelle clause *positive* (resp. *négative*) toute clause contenant uniquement des litéraux positifs (resp. négatifs). Une formule est *Horn-SAT* (rep. *2-SAT*) si elle est composée uniquement de clauses de Horn (clauses binaires). Ces deux fragments SAT sont connus pour être traitables en temps linéaire [2, 10].

En plus de ces notations usuelles, on définit la négation d'un ensemble de litéraux S comme l'ensemble des opposés de chaque litéral de S, e.g. $\neg S = \{\neg l | l \in S\}$.

Une *interprétation* I d'une formule booléenne est une affectation des valeurs de vérité de ces variables. I peut être représentée sous forme d'une conjonction (un ensemble) de litéraux. Soit $l \in \mathcal{L}(\phi)$, la formule obtenue en affectant l à vrai, c'est à dire $\phi(l) = \{c - \{\neg l\} | c \in \phi, \neg l \in c\} \cup \{c | c \in \phi, l \notin c, \neg l \notin c\}$. Pour une interprétation $I = \{l_1, l_2, \ldots, l_n\}$, $\phi(I) = \phi(l_1)(l_2)\ldots(l_n)$. Un *modèle* d'une formule est une interprétation I qui satisfait cette formule i.e. $\phi(I) = \emptyset$. En conséquence, résoudre le problème SAT consiste à trouver un modèle d'une formule CNF si ce modèle existe ou prouver l'inexistence d'aucun modèle pour cette formule.

Soit c_1 une clause contenant un litéral a et c_2 une clause contenant le litéral opposé $\neg a$, la *résolvante* entre c_1 et c_2 est la disjonction de tous les litéraux de c_1 et c_2 excepté a et $\neg a$, notée $res(a, c_1, c_2)$. Une résolvante est dite *tautologique* si elle contient un litéral et son opposé, sinon elle est dite fondamentale.

3 Graphe associé à une formule CNF

Comme mentionné dans l'introduction, différentes représentations sous forme de graphe d'une formule CNF ont été proposées. Elles ont induites différentes utilisations, comme le graphe d'apprentissage [22], le pré-traitement [4], la survey propagation pour résoudre les instances aléatoires 3-SAT [5], la visualisation [25]. Notre représentation peut être vue comme une extension du graphe d'implication 2-SAT [2] au cas général.

3.1 Vers un graphe étiqueté

Avant d'introduire notre approche, rappelons le graphe d'implication des formules 2-SAT.

Soit ϕ une formule 2-SAT. Le graphe associé à ϕ, est définit comme $G_\phi = (S, E)$, où $S = \{x, \neg x | x \in \mathcal{L}(\phi)\}$ et $E = \{(\neg x, y), (\neg y, x) | (x \vee y) \in \phi\}$. L'algorithme polynômial utilisé pour résoudre ϕ est basé sur l'application de la fermeture transitive de G_ϕ, $G_\phi^c = (S, E')$ et sur la vérification qu'il existe un litéral $x \in S$ tel que $(x, \neg x) \in E'$ et $(\neg x, x) \in E'$.

Évidemment, le calcul de la fermeture transitive du graphe G_ϕ est équivalent à la saturation de ϕ par résolution. En effet, pour (x, y) et (y, z) de E un nouveau arc (x, z) est généré et ajouté au graphe. Une telle application correspond à $res(y, (\neg x \vee y), (\neg y \vee z))) = (\neg x \vee z)$.

Pour une formule CNF quelconque, une représentation naturelle peut être obtenue en utilisant un hypergraphe où les noeuds sont représentés par les litéraux, et les hyper arrêtes correspondent aux clauses.

Dans ce qui suit, le graphe représentatif d'une formule CNF générale est décrit.

Définition 1
Soit c une clause telle que $|c| \geq 2$. Un contexte η_c associée à c est une conjonction de

littéraux tel que $\neg\eta_c \subset c$ *et* $|c - \{\neg\eta_c\}| = 2$ *i.e quand* η_c *est vrai, la clause* c *devient une clause binaire.*

Exemple 1
Soit $c = (a \vee \neg b \vee c \vee d)$ *une clause. Un contexte possible associé à* c *est* $\eta_c = (b \wedge \neg c)$*. La clause* c *peut être réécrite comme* $((\neg a \wedge \eta_c) \rightarrow d)$*.*

Pour une clause c de taille k, on a $|\eta_c| = k - 2$. Le contexte associé à une clause binaire est vide, Tandis que pour les clauses ternaires, le contexte est réduit à un seul litéral. Le nombre des contextes possibles de c est égal à $\frac{k(k-1)}{2}$. Une clause c peut être réécrite en $k(k-1)$ différentes façons. Quand $k = 1$, la clause est unitaire, aucun contexte n'est possible.

En utilisant la clause c de l'exemple 1, on obtient six contextes possibles de taille 2 et douze différentes façons pour réécrire c.
Définissons maintenant le graphe représentatif d'une formule CNF.

Définition 2 (Représentation SAT-graphe)
Soit ϕ *une formule CNF. On définit* $G_\phi = (S, E, v)$ *comme étant le graphe SAT associée à* ϕ *défini comme suit*

– $S = \{x, \neg x | x \in \mathcal{L}(\phi)\}$
– $E = \{a = (\neg x, y)$ *tel que* $\exists c \in \phi, c = (x \vee \neg\eta_c \vee y)$ *et* $v(a) = \eta_c\}$

Dans la suite, pour des raisons de clarté, on notera parfois l'arc étiqueté $a = (x, y)$ comme $(x, y, v(a))$. On note aussi $cla(a) = (\neg x \vee \neg v(a) \vee y)$ comme étant la clause associée à a.

Clairement la définition 2 généralise le graphe classique 2-SAT. En effet, si tous les contextes sont vide i.e. toutes les clauses de ϕ sont binaires, dans ce cas tous les arcs de G_ϕ sont étiquetés avec un ensemble vide de litéraux.

3.2 Résolution/fermeture transitive

Dans cette section, nous démontrons l'équivalence entre la résolution classique et la fermeture transitive du graphe. On commencera par introduite quelques définitions nécessaires et on considère dorénavant uniquement les formules ϕ sans clauses tautologiques et $G_\phi = (S, A, v)$ indique le graphe associé à ϕ.

Définition 3
Soit $G_\phi = (S, A, v)$ *un graphe,* $a_1 = (x, y, v(a_1)) \in A$ *et* $a_2 = (y, z, v(a_2)) \in A$*. On définit* $tr(a_1, a_2) = a_3$ *tel que* $a_3 = (x, z, v(a_1) \cup v(a_2) \backslash \{x, \neg z\})$*.*

Dans la définition ci-dessus, l'élimination de $\{x, \neg z\}$ du contexte associé à a_3 garantie que la clause $cla(a_3)$ ne contient pas plusieurs occurrences du même litéral. Ce qui correspond à l'application de la règle de fusion.

Exemple 2
Soit $a_1 = (x, y, \{\neg z, e\})$ *et* $a_2 = (y, z, \{x, f\})$*. Les deux clauses codées respectivement par* a_1 *et* a_2 *sont* $c_1 = (\neg x \vee z \vee \neg e \vee y)$ *et* $c_2 = (\neg y \vee \neg x \vee \neg f \vee z)$*. On obtient*

$tr(a_1, a_2) = (x, z, \{\neg e, \neg f\})$ et $cla(tr(a_1, a_2)) = (\neg x \vee \neg e \vee \neg f \vee z)$. Cette dernière clause ne contient pas de litéraux redondants. En utilisant la résolution, $res(c_1, c_2, y)$ on obtient donc $c_3 = (\neg x \vee z \vee \neg e \vee \neg x \vee \neg f \vee z)$.

En appliquant la règle de fusion sur c_3, on élimine une occurrence de $\neg x$ et z. On obtient $res(c_1, c_2, y) = cla(tr(a_1, a_2)) = (\neg x \vee \neg e \vee \neg f \vee z)$

Propriété 1
Soit $c_1 = (x \vee \eta_{c_1} \vee y) \in \phi$, $c_2 = (\neg y \vee \eta_{c_2} \vee z) \in \phi$, $a_1 = (x, y, \eta_{c_1}) \in A$ et $a_2 = (x, y, \eta_{c_2}) \in A$. On a $res(c_1, c_2, y) = cla(tr(a_1, a_2))$

La preuve de la propriété 1 est une conséquence directe des définitions 2 et 3.

Définition 4 (chemin)
Un chemin $p(x, y)$ entre x et y dans G_ϕ, est définit comme suit : $p(x, y) = [x_{i_1}, x_{i_2}, \dots, x_{i_k}]$ tel que $x_{i_1} = x$ et $x_{i_k} = y$ et $1 < j \leq k$ $(x_{i_{j-1}}, x_{i_j}) \in A$.

On définit $\eta_p = \bigcup_{1<j\leq k} v((x_{i_{j-1}}, x_{i_j}))$ le contexte associée à $p(x, y)$
et $tr(p(x, y)) = tr(\dots(tr((x_{i_1}, x_{i_2}), (x_{i_2}, x_{i_3}))\dots(x_{i_{k-1}}, x_{i_k})\dots)$,
comme étant la fermeture transitive associée à $p(x, y)$.

Définition 5 (chemin fondamental)
Soit $p(x, y) = [x = x_{i_1}, x_{i_2}, \dots, x_{i_k} = y]$ un chemin entre x et y. $p(x, y)$ est dit fondamental s'il satisfait les conditions suivantes :

- *η_p ne contient pas un litéral et son opposé.*
- *$\neg x \notin \eta_p$ et $y \notin \eta_p$*

La propriété suivante montre qu'à partir d'un chemin fondamental $p(x, y)$ on peut dériver une résolvante fondamentale en utilisant la fermeture transitive.

Propriété 2
Soit $p(x, y)$ un chemin. $p(x, y)$ est fondamental ssi $cla(tr(p(x, y)))$ est une clause fondamentale.

Preuve 1
Pour un chemin fondamental $p(x, y)$, η_p ne contient pas deux litéraux opposés. Comme $cla(tr(p(x, y))$ peut être réécrite sous forme $r = (\neg x \vee \neg\eta_p \vee y)$, r est clairement une clause fondamentale. En effet, d'après la définition 5, η_p ne contient pas un litéral et son opposé (première condition) et $x \in \neg\eta_p$ et $\neg y \notin \neg\eta_p$ (seconde condition). L'inverse est aussi vrai. On suppose que $cla(tr(p(x, y)))$ n'est pas fondamentale. Comme $tr(p(x, y)) = (x, y, \eta_p)$, la clause $cla(tr(p(x, y)) = (\neg x \vee \neg\eta_p \vee y)$ n'est pas fondamentale. Ce qui montre que la première ou la deuxième condition de la définition 5 n'est pas satisfaite.

Dans ce qui suit, on décrit comment la résolution classique peut être appliquée en utilisant la fermeture transitive.

Définition 6
Soit $G_\phi = (S, A, v)$ le graphe associé à ϕ. On définit $tr(G_\phi) = (S, A', v')$ tel que : $\forall x \in S$, $\forall y \in S$ tel que $\exists p(x, y)$ un chemin fondamental entre x et y, $A' = A \cup \{tr(p(x, y))\}$.

Propriété 3
Soit ϕ une formule CNF. ϕ est insatisfiable ssi $\exists k$ tel que $tr^k(G_\phi) = (S, A', v')$ et $\exists x \in S$ tel que $(x, \neg x, \emptyset) \in A'$ et $(\neg x, x, \emptyset) \in A'$

Preuve 2
Notre graphe est complet i.e. chaque clause est codée $k(k-1)$ fois. Comme montré précédemment, l'application de tr sur les arcs de G_ϕ est équivalent à l'application de la résolution entre les clauses de ϕ.

La preuve pour la réfutation complète peut être dérivée de la résolution classique.

Dans la propriété 3, nous avons montré comment on peut appliquer la résolution classique sur le graphe. Évidemment, d'autres techniques SAT (e.g. élimination de variable, propagation unitaire) peuvent être reformulées en utilisant ce même graphe. Notons que ce graphe peut avoir plusieurs caractéristiques intéressantes. L'un des principaux avantages est que la dépendance entre les clauses est mieux exprimée en utilisant une telle représentation. Une telle propriété structurelle est connue pour être déterminante dans l'efficacité des solveurs SAT. Plus intéressant encore, le graphe SAT peut être vu comme un graphe d'états, où chaque état représente un litéral et une transition entre les deux états s_1 et s_2 est représentée par une condition $v((s_1, s_2))$ (un ensemble de litéraux). Plusieurs problèmes intéressants peuvent être reformulés plus facilement. Par exemple, le calcul du plus court chemin entre un litéral et son opposé i.e. le calcul des conditions minimales pour qu'un litéral puisse être impliqué.

Dans la section suivante, deux exploitations possibles de notre graphe sont décrites.

4 Utilisation de representation sous forme de graphe

Dans la définition de G_ϕ, chaque clause est représenté $k(k-1)$ fois. Une telle représentation multiple d'une clause donnée est nécessaire pour que la fermeture transitive soit complète pour la réfutation. Pour des formules de grandes tailles, la représentation complète du graphe est impraticable.

Suivant à quelle fin ce graphe est utilisé, des restrictions utiles peuvent être définies. Par exemple, si on représente chaque clause avec seulement un seul arc, le graphe obtenu est équivalent à la formule CNF.

Nous devons seulement nous rappeler que dans ce cas certaines résolvantes pouvant être obtenues en employant la fermeture transitive du graphe total ne peuvent plus être dérivées à partir de ce graphe restreint.

4.1 Ensembles 2-SAT strong backdoor

Dans cette section, on va montrer comment la représentation sous forme de graphe peut être utilisée pour le calcul des ensembles strong backdoor.

La notion de (strong) backdoor introduite par Williams-etal in [27] est un domaine de recherche très actif actuellement. Un ensemble de variables forme un backdoor d'une formule, si il existe une affectation de ces variables rendant la formule simplifiée traitable en temps polynômial. Un tel ensemble de variables est strong backdoor si pour toute affectation de ces variables le reste de sous-formule est traitable polynomialement. Ce type de structure est relié à la notion de variables indépendantes [15, 13] et à l'ensemble coupe cycle introduit pour le problème de la satisfaction de contraintes [8].

Rappelons que le calcul de l'ensemble strong backdoor minimal est un problème NP-difficile. En pratique, approximer un strong backdoor de taille "raisonnable"est un challenge très intéressant.

Des précédents travaux ont été effectués dans ce but. Par exemple, l'approche proposée dans [14] essaye de couvrir un ensemble de portes (fonctions booléennes) à partir d'une formule donnée, et en exploitant les dépendances entre les variables un strong backdoor est calculé.

Le principal inconvénient de cette approche est que plusieurs problèmes ne contiennent pas de telles propriétés structurelles (fonctions booléennes). Recemment, dans [20] un concept de fraction reverse horn d'une formule CNF est proposé et une importante corrélation entre la densité des instances aléatoires 3-SAT et les performances des solveurs SAT est mise en évidence. Dans [19], la relation entre la complexité des instances aléatoires 3-SAT insatisfaisables et les noyaux et les ensembles strong backdoor est étudiée.

Comme notre représentation sous forme de graphe est une généralisation du graphe d'implication 2-SAT, les ensembles strong backdoor que nous allons calculer sont considérés par rapport au fragment polynômial 2-SAT i.e. l'ensemble de variables B tel que pour toute affectation I des variables de B, la formule simplifiée par I appartient à la classe 2-SAT.

Notons que pour calculer les ensembles 2-SAT strong backdoor, il est suffisant de considérer une restriction du graphe. Chaque clause de ϕ est représentée avec seulement un seul arc dans G_ϕ. On note G^u_ϕ le graphe restreint associé à ϕ, appelé graphe unique.

La propriété suivante montre comment on peut utiliser le graphe unique G^u_ϕ pour calculer les ensembles 2-SAT strong backdoor.

Propriété 4

Soit ϕ une formule, et $G^u_\phi = (S, A, v)$ son graphe associé.
$B = \cup_{a_i \in A}\{\mathcal{V}(l) | l \in v(a_i)\}$ est un ensemble 2-SAT strong backdoor.

Preuve 3

Pour prouver cette propriété, il est suffisant de montrer que pour toute affectation I de B, $\phi(I)$ est une formule 2-SAT. Chaque arc $a_i = (x, y, v(a_i)) \in A$ code une clause $c = (\neg x \vee \neg v(a_i) \vee y) \in \phi$. Pour chaque clause $c \in \phi$, toutes les variables de $v(a_i)$ apparaissent dans B. Conséquence I soit il satisfait au moins un litéral dans $\neg v(a_i)$, soit tous les litéraux dans $\neg v(a_i)$ sont faux. En Conséquence, dans tous les cas, la formule est soit 2-SAT, non satisfaite ou satisfaite. B est donc un 2-SAT strong backdoor.

Rappelons que notre objectif principal est l'approximation de l'ensemble strong backdoor 2-SAT minimal. Pour arriver à cette fin, on utilise G^u_ϕ le graphe unique de ϕ. De plus, comme une clause c de taille k peut être codée de plusieurs façons, nous avons besoin de choisir un arc parmi les $k(k-1)$ possibles. Pour construire G^u_ϕ, à chaque étape une nouvelle clause $c = (x \vee \eta_c \vee y) \in \phi$ de taille k est traitée et un nouvel arc $a = (x, y, v(a))$ où $v(a) = \neg\eta_c$, est ajouté au graphe. Premièrement, tous les litéraux de c appartenant aux précédents contextes sont inclus dans $v(a)$. Deuxièmement, les littéraux x et y sont choisi parmi les littéraux restants (ceux qui apparaissent le moins dans ϕ). Plus intéressant encore, la propriété suivante est utilisé pour réduire encore plus la taille de l'ensemble strong backdoor.

Propriété 5
Soit B un ensemble 2-SAT strong backdoor d'une formule CNF ϕ, et $b \in B$. si $\forall c \in \phi$ tel que $b \in c$ or $\neg b \in c$, $|\mathcal{V}(c) \cap B| > |c| - 2$, alors la variable b peut être retirée de B.

La preuve de la propriété 5 est évidente. Cette dernière exprime le fait qu'une variable de l'ensemble 2-SAT strong backdoor peut être éliminée si elle apparaît seulement dans les clauses avec au plus un litéral n'appartenant pas à B i.e. l'élimination de telles variables aboutit à des clauses contenant au plus deux litéraux qui ne sont pas dans B.

L'algorithme 1décrit notre approche pour le calcul du strong backdoor 2-SAT. Comme nous cherchons des ensembles 2-SAT strong backdoor, toutes les clauses binaires de ϕ sont supprimées (ligne 2), et en utilisant l'heuristique décrite précédemment, le graphe G^u_ϕ est construit sur l'ensemble des clauses restantes. Un premier ensemble B est calculé (ligne 4), et réduit en utilisant la propriété 5 (ligne 5 à 10).

Input: une formule CNF ϕ
Output: un ensemble 2-SAT strong backdoor B
1 **begin**
2   Supprimer les clauses binaires de ϕ
3   Créer le graphe $G^u_\phi = (S, E)$
4     $B = \bigcup_{a \in E} \mathcal{V}(v(a))$
5   **foreach** $c \in \phi$ **do** $nb(c) = |\{x \notin c \cap B\}|$
6   **foreach** $u \in B$ **do**
7   bool remove=true
8   **foreach** *($c \in \phi | u \in c$ or $\neg u \in c$)* **do**
9   **if** *($nb(c) = 2$)* **then** remove = false
10   **end**
11   **if** *remove* **then** $B = B - \{u\}$
12   **end**
13 **end**

Algorithm 1: Approximation d'un ensemble 2-SAT strong backdoor

4.2 Graphe et pré-traitement

La simplification des formules CNF est un moyen très important de pré-traitement inclus dans de nombreux solveurs [22]. Plusieurs techniques de pré-traitement ont été proposées. Citons les plus récentes comme "Sattelite" qui utilise la technique de l'élimination de variables [11] et "hypre" un pré-traitement basé sur l'hyper binaire résolution [3].

Dans cette section, un nouveau pré-traitement basé sur la représentation graphique d'une formule CNF est proposé, on y considère le graphe restreint suivant.

Définition 7 (Graphe SAT ordonné)
Soit ϕ une formule. On définit le graphe ordonné G^{or}_ϕ comme suit :

– $S = \{x, \neg x | x \in \mathcal{L}(\phi)\}$
– $E = \bigcup_{c \in \phi} arcs(c)$, *tel que* $arcs(c)$ *pour* $c = (x_1 \vee x_2 \vee .. \vee x_n)$ *est défini comme :*
  – *pour* $(1 \leq i < n)$, $a_i = (x_i, x_{i+1}, v(a_i))$ *et* $v(a_i) = \{x_j | 1 \leq j \leq n$ *et* $j \neq i$ *et* $j \neq i+1\}$, *et*
  – *pour* $(i = n)$, $a_n = (x_n, x_1, v(a_n))$ *et* $v(a_n) = \{x_2, ..., x_{n-1}\}$

Exemple 3
Soit $\phi = (x_1 \vee x_2 \vee x_3) \wedge (\neg x_1 \vee x) \wedge (\neg x_2 \vee x) \wedge (\neg x_3 \vee x)$ *une formule CNF. Le graphe ordonné* G^{or}_{ϕ} *est représenté dans la figure 1. Comme on peut le remarquer, pour les clauses binaires, le graphe obtenu (SAT graphe ordonné) est le même avec et sans resctriction. i.e. une clause binaire est representé par deux arcs dans les deux graphes.*

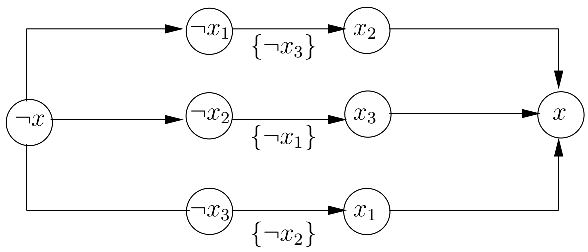

FIG. 1 – Graphe G^{or}_{ϕ} ordonné associé à la formule ϕ de (exemple 3)

Le graphe ordonné est un bon compromis entre une représentation totale et le graphe unique. Comme expliqué dans les sections précédentes, l'utilisation de la fermeture transitive sur un chemin du graphe, permet de générer facilement des résolvantes. L'application de la résolution classique revient à appliquer la fermeture transitive du graphe. En conséquence, un tel processus est clairement exponentiel dans le pire des cas.

Décrivons maintenant l'étape principale de notre pré-traitement utilisant le graphe représentatif d'une formule CNF ϕ. Le but de cette étape est de générer une résolvante fondamentale de taille limitée. A chaque étape une nouvelle clause $c = (x_1 \vee x_2 \vee \ldots \vee x_n)$ de ϕ est sélectionnée et traitée comme décrit dans l'algorithme 2. Pour chaque $x_i \in c$ (ligne 4), un arc $a = (x_i, x_j, v(a)) \in G^{or}_{\phi}$ est choisi tel que $res(c, (\neg x_i \vee \neg v(a) \vee x_j), x_i)$ est fondamentale (ligne 4 à 11). Plus précisément, la nouvelle résolvante est générée suivant les arcs de G^{or}_{ϕ}. Une telle résolvante est composée des ensembles de litéraux cumulées ($noeuds$) et contextes (η). Un arc a est choisi tel que $res(c, cla(a), x_i)$ est fondamentale (ligne 5). Si un tel arc existe, le contexte η (resp. $noeuds$) est augmenté de $a(v)$ (resp. $\{x_j\}$) (ligne 6) ; sinon x_i est simplement ajouté à l'ensemble des noeuds (ligne 8) si la résolvante obtenue est fondamentale. (ligne 7). Dans le dernier cas, si aucune clause fondamentale n'a pu être générée le pré-traitement s'arrête (ligne 10).

Pour une formule contenant une clause $c = (x_1 \vee x_2 \vee x_3)$ et un ensemble de clauses binaires $C = \{(\neg x_1 \vee y), (\neg x_2 \vee y), (\neg x_3 \vee y)\}$, le litéral y est généré par l'algorithme 2. A partir de la clause c et les arcs qui codent C. Dans ce cas, l'application de notre algorithme correspond exactement à l'application de l'hyper binaire

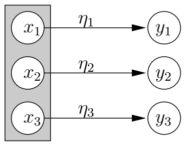

Fig. 2.a résolution par le graphe

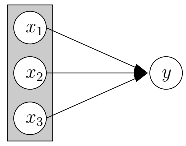

Fig. 2.b Hyper binaire résolution

résolution sur c et C [3] (see figure 2.b). Pour les clauses C de taille quelconque ($C = \{(\neg x_1 \vee \neg\eta_1 \vee y_1), (\neg x_2 \vee \neg\eta_2 \vee y_2), (\neg x_3 \vee \neg\eta_3 \vee y_3), \}$) (voir figure 2.a), notre algorithme génère la résolvante $(\neg\eta_1 \vee \neg\eta_2 \vee \neg\eta_3 \vee y_1 \vee y_2 \vee y_3)$. Clairement, une étape de notre pré-traitement comme décrit au-dessus peut être vue comme une extension de l'hyper binaire résolution proposée par [3].

Notre pré-traitement utilise deux paramètres fixes, le nombre de clauses considérées ($nbCla$) et la taille maximale de la résolvante générée ($maxRes$). Pour chaque clause c l'algorithme 2 est itéré. Ce processus est réitéré en utilisant la résolvante obtenue dans l'étape précédente, jusqu'à ce qu'aucune résolvante fondamentale ne puisse être obtenue.

Input: $G^{or}(V, E)$ la représentation en graphe de ϕ
$c \in \phi$
Output: c_{res} une clause résolvante

begin
  $\eta = \emptyset$
  $noeuds = \emptyset$
  foreach $x_i \in c$ **do**
    if $\exists a = (x_i, x_j, v(a)) \in E$ *tel que*
    $x_j \notin \eta$ *et* $v(a) \cap noeuds = \emptyset$ *et* $\neg x_j \notin c$ *et* $\neg v(a) \cap c = \emptyset$
    then $\eta = \eta \cup v(a)$, $noeuds = noeuds \cup \{x_j\}$
    else if *(*$x_i \notin \eta$ *et* $\neg x_i \notin noeuds$*)*
    then $noeuds = noeuds \cup \{x_i\}$
    else
    return *null*
    $c_{res} = noeuds \cup \neg\eta$
  end
  return c_{res}
end

Algorithm 2: Une étape du pré-traitement

5 Expérimentations

Les résultats expérimentaux reportés dans cette section ont été conduits sur de nombreuses instances SAT issues des dernières compétitions et réalisés sur un Xeon 3.2 GHz (2 GB RAM).

La table 1 présente une comparaison entre notre approche pour calculer l'ensemble 2-SAT strong backdoor (algorithme 1) et celle calculant l'ensemble Horn strong backdoor proposée récemment dans [23]. Pour chaque instance, le nombre de variables (#V), le

instance	#V	#C	Horn BD	2SAT BD
clauses-4-1967	267767	823658	–	60747 (22%)
clauses-2-1966	75528	226124	29357 (38%)	17945 (23%)
depots1_ks99i-4010	161	445	32 (19%)	31 (19%)
equilarge_l2-519	4487	18555	2582 (57%)	367 (8%)
equilarge_l3-520	4496	18591	2595 (57%)	373 (8%)
equilarge_l1-518	4574	18903	2635 (57%)	374 (8%)
ferry6_ks99i.renamed-sat05-3997	1425	14454	657 (46%)	539 (35%)
ferry8_ks99a.renamed-sat05-4004	1259	15206	573 (45%)	471 (37%)
linvrinv7-566	1274	4166	631 (49%)	467 (36%)
linvrinv8-567	1920	6337	961 (50%)	705 (36%)
linvrinv5-564	450	1426	221 (49%)	167 (37%)
iso-brn100-3025	3587	5279	372 (10 %)	349 (9%)
iso-brn009-2934	1770	4540	318 (17 %)	187 (10%)
iso-icl009-3243	2251	5177	442 (19 %)	512 (22%)
iso-ukn006-3387	1906	4188	407 (21%)	453 (23%)
iso-icl008-3242	2136	5938	461 (21%)	545 (25%)
mm-2x3-8-8-s.1-476	1148	8088	621 (54%)	132 (11%)
mm-2x3-8-8-sb.1-475	2326	80891	1661 (71%)	926 (39%)
mod2-rand3bip-unsat-120-1-2624	120	320	85 (69%)	50 (41%)
mod2c-rand3bip-unsat-120-3-2340	160	640	126 (78%)	86 (53%)
mod2c-rand3bip-unsat-150-1-2368	200	800	154 (77%)	111 (55%)
mod2c-rand3bip-unsat-105-2-2324	140	560	109 (77%)	78 (55%)
mod2c-rand3bip-unsat-120-1-2338	160	640	126 (78%)	90 (56%)
mod2-rand3bip-unsat-150-2-2655	150	400	106 (70%)	61 (40%)
par32-3-c	1325	5294	698 (52%)	199 (15%)
par32-1-c	1315	5254	696 (52%)	202 (15%)
par32-4-c	1333	5326	703 (52%)	202 (15%)
par32-5-c	1339	5350	708 (52%)	202 (15%)
par16-5-c	341	1360	189 (55%)	70 (20%)
par16-1-c	317	1264	176 (55%)	68 (21%)
pmg-12-UNSAT-3940	190	632	136 (71%)	77 (40%)
pmg-14-UNSAT-3942	577	1922	408 (70%)	232 (40%)
pmg-11-UNSAT-3939	169	562	122 (72%)	65 (38%)
pmg-13-UNSAT-3941	409	1362	291 (71%)	163 (39%)
series18	6410	32421	1665 (25%)	1122 (17%)
series16	4546	22546	1189 (26%)	870 (19%)
strips-gripper-08t14-1156	1966	23326	826 (42%)	773 (39%)
strips-gripper-16t31-1146	8904	222172	4176 (46%)	3788 (42%)
strips-gripper-18t35-1147	11318	316416	5331 (47%)	4835 (42%)
strips-gripper-10t19-1143	3390	53008	1475 (43%)	1400 (41%)
satellite3_v01a-3989	303	7840	226 (74%)	210 (69%)

TAB. 1 – 2-SAT vs Horn strong backdoor

instance	SAT	#V	#C	#CA	preproc	Minisat +preproc	Minisat
mod2-r3b.280-1-2263	Y	280	1120	183	0.76	574.49	–
mod2-r3b.240-1-2203	Y	240	960	162	0.73	250.35	–
mod2-r3b.270-1-2248	Y	270	1080	180	0.76	510.86	–
mod2-r3b.260-2-2234	Y	260	1040	190	0.74	657.93	–
mod2c-r3b.210-2-2474	Y	297	2092	341	1.14	488.36	–
mod2c-r3b.180-2.05-2429	Y	252	1784	303	1.08	204.37	411.16
mod2c-r3b.170-3.05-2415	Y	240	1724	276	1.04	286.37	368.76
mod2-r3b.150-1.05-2654	N	150	400	79	0.52	160.75	189.16
mod2-r3b.135-2.05-2640	N	135	360	71	0.50	336.46	828.83
mod2-r3b.135-1.05-2639	N	135	360	70	0.51	529.64	603.32
mod2-r3b.250-1.05-2218	Y	250	1000	169	0.72	699.16	753.9
fclqcolor-08-05-06-1273	N	116	1362	96	0.77	36.84	66.88
fphp-012-011.05-1228	N	132	1398	15	0.78	889.1	651.32
gensys-icl009-3135	N	926	17577	1375	6.63	759.11	–
gensys-icl008-3134	N	960	17640	1482	6.74	889.67	–
gensys-icl006.05-2718	N	1321	7617	2360	70.78	189.45	203.492
gensys-ukn001.05-2678	N	1886	7761	2260	70.57	655.86	728.51
gensys-ukn006.05-3349	N	874	16339	1203	5.56	410.73	365.82
gt-020-1305	N	400	7050	770	2.33	821.44	–
gt-018-1292	N	324	5066	577	1.81	18.79	127.74
grid-pbl-1342.05-1342	N	3660	7142	1483	3.98	350.32	192.17
grid-pbl-1342.05-1342	N	3660	7142	1489	7.06	64.31	153.51
homer11.shuffled	N	220	1122	28	0.71	48.94	59.99
homer09.shuffled	N	270	1920	1920	5.18	57.23	70.32
lisa21_0_a	Y	1453	7967	1401	2.92	32.67	77.04
php-012-012-1158	Y	144	804	23	0.63	66.3	210.19
strips-gripper-14t27-1145	Y	6778	93221	53	48.0	199.36	–
strips-gripper-12t23-1144	Y	4940	148817	38	26.49	167.44	–
vmpc_34-1958	Y	1156	194072	21109	53.63	60.86	–
vmpc_26-1946	Y	676	86424	9612	22.68	209.26	306.37
vmpc_27-1947	Y	729	96 849	10729	24.63	596.71	254.15
clqcolor-08-05-0605-1249	N	116	1114	94	4.8	48.39	85.80
series16	Y	4546	22546	4298	10.43	51.62	316.14
clus-45-30-003.03-1082	Y	1200	4800	1004	5.82	85.73	197.12
clus-20-20-003.sat03-1087	Y	1208	4800	982	6.17	77.39	174.89

TAB. 2 – Minisat [12] avec et sans pré-traitement nombre de clauses (#C) et la taille de l'ensemble 2-SAT (resp. Horn) strong backdoor est reporté. Ces résultats montrent clairement que notre algorithme est meilleur que celui de Paris et al.

Dans quelques cas , le gain est très important (e.g. `mm-*`, `equilarge-*`, `iso-*` `series-*`). Sur un large panel d'instances, l'algorithme 2-SAT strong backdoor est le meilleur.

Notre seconde proposition concerne le pré-traitement des formules CNF. (section 4.2). Comme mentionné, notre pré-traitement utilise deux paramètres fixes. Le premier, est le nombre de clauses à filtrer qui est fixé à 10% des clauses initiales et le second et la taille maximale des clauses résolvantes générées qui est lui fixé à 50.

La table 2 montre une comparaison expérimentale du solveur Minisat [12] avec et sans notre pré-traitement. Le temps de calcul est donné en secondes et limité à 900 secondes.

Pour chaque instances, on reporte le nombre de variables (#V), le nombre de clauses (#C), le nombre de clauses ajoutées à la formule originale (#CA), le temps dû au pré-traitement (preproc) et le temps total (pré-traitement + résolution) mis par Minisat. Ici aussi, les résultats sont prométeurs et montrent clairement l'intérêt de faire ce genre de pré-traitement. Par exemple, notre pré-traitement permet de résoudre 12 instances que Minisat classique. L'utilité de telles résolvantes est clairement mis en évidence (e.g. `strips-gripper-*`, `homer-*` and `series-*`). Bien evidemment, concernant les instances faciles, notre pré-traitement décroit les performances de Minisat.

6 Conclusion

Dans cet article nous avons présenté une nouvelle représentation de formules CNF sous forme de graphe. Elle étend le graphe d'implication 2-SAT. Cette nouvelle représentation offre des perspectives intéressantes. La structure de la formule (dépendances des variables) est clairement mieux exprimée. Nous avons montré que la résolution peut être appliquée à ce graphe en utilisant la notion de fermeture transitive du graphe. Deux façons d'exploiter ce graphe ont été présentées. Une technique d'extraction d'ensembles 2-SAT strong backdoor est proposée améliorant des résultats précédement obtenus.

Un pré-traitement des formules booléennes sous forme CNF est proposé étendant l'hyper binaire résolution. L'intégration de ce pré-traitement montre une amélioration des résultats du solveur Minisat sur une large collection d'instances.

Finalement, cette nouvelle représentation sous forme de graphe offre de perspectives intéressantes pour les recherches futures. Parmi elles, on envisage de calculer les conditions minimales sous lesquelles un litéral est impliqué. (le plus court chemin entre un litéral et son opposé).

Références

[1] F. Aloul, A. Ramani, I. Markov, and Karem A. Sakallah. Shatter : Efficient breaking for boolean satisfiability. In *proceedings of DAC*, pages 836–839. ACM Press, 2003.

[2] B. Aspvall, M. Plass, and R. Tarjan. A linear-time algorithm for testing the truth of certain quantified boolean formulas. *Information Processing Letters*, 8(3) :121–123, 1979.

[3] F. Bacchus and J. Winter. Effective preprocessing with hyper-resolution and equality reduction. In *proceedings of SAT*, pages 341–355, 2003.

[4] R. Brafman. A simplifier for propositional formulas with many binary clauses. In *proceddings of IJCAI*, pages 515–522, 2001.

[5] A. Braunstein, M. Mézard, and R. Zecchina. Survey propagation : An algorithm for satisfiability. *Random Struct. Algorithms*, 27(2) :201–226, 2005.

[6] A. Darwiche. New advances in compiling CNF to decomposable negational normal form. In *Proceedings of ECAI*, pages 328–332, 2004.

[7] M. Davis, G. Logemann, and D. Loveland. A machine program for theorem-proving. *Communications of the ACM*, 5(7) :394–397, July 1962.

[8] R. Dechter. "enhancement schemes for constraint processing : Backjumping, learning and cutset decomposition. *Artificial Intelligence*, 41(3) :273–312, 1990.

[9] H. Dixon and M. Ginsberg. Inference methods for a pseudo-boolean satisfiability solver. In *proceedings of AAAI*, pages 635–640, 2002.

[10] W. Dowling and J. Gallier. Linear-time algorithms for testing satisfiability of propositional horn formulae. *journal of logic programming*, 3 :267–284, 1984.

[11] N. Eén and A. Biere. Effective preprocessing in sat through variable and clause elimination. In *proceedings of SAT*, pages 61–75, 2005.

[12] N. Eén and N. Sörensson. An extensible sat-solver. In *proceedings of SAT*, pages 502–518, 2003.

[13] E. Giunchiglia, M. Maratea, and A. Tacchella. Dependent and independent variables in propositional satisfiability. In *proceedings of ECLAI*, pages 296–307, 2002.

[14] E. Gregoire, B. Mazure, R. Ostrowski, and L. Sais. Automatic extraction of functional dependencies. In *proceedings of SAT*, volume 3542 of *LNCS*, pages 122–132, 2005.

[15] H. Kautz, D. McAllester, and B. Selman. Exploiting variable dependency in local search. In *"Abstracts of the Poster Sessions of IJCAI-97"*, 1997.

[16] H. Kautz and D. McAllester B. Selman. Exploiting variable dependency in local search. In *Abstract appears in "Abstracts of the Poster Sessions of IJCAI-97"*, Nagoya (Japan), 1997.

[17] C. Li. Equivalent literal propagation in the dll procedure. *Discrete Applied Mathematics*, 130(2) :251–276, 2003.

[18] F. Lu, L. Wang, K. Cheng, and R. Huang. A circuit sat solver with signal correlation guided learning. In *proceedings of DATE*, pages 892–897, 2003.

[19] I. Lynce and J. Marques-Silva. Hidden structure in unsatisfiable random 3-sat : an empirical study. In *proceedings of ICTAI*, pages 246–251, 2004.

[20] H. Van Maaren and L. Van Norden. Correlations between horn fractions, satisfiability and solver performance for fixed density random 3-cnf instances. *Annals of Mathematics and Artificial Intelligence*, 44 :157– 177, 2005.

[21] J. Marques-Silva and K. Sakallah. GRASP : A new search algorithm for satisfiability. In *Proceedings of CAD*, pages 220–227, 1996.

[22] M. Moskewicz, C. Madigan, Y. Zhao, L. Zhang, and S. Malik. Chaff : Engineering an efficient sat solver. In *Proceedings of DAC*, 2001.

[23] L. Paris, R. Ostrowski, L. Saïs, and P. Siegel. Computing horn strong backdoor sets thanks to local search. In *proceedings of ICTAI*, pages 139–143, 2006.

[24] B. Selman, H. Levesque, and D. Mitchell. A New Method for Solving Hard Satisfiability Problems. In *Proceedings of AAAI*, pages 440–446, 1994.

[25] C. Sinz and E. Dieringer. Dpvis - a tool to visualize the structure of sat instances. In *proceedings of SAT*, pages 257–268, 2005.

[26] C. Thiffault, F. Bacchus, and T. Walsh. Solving non-clausal formulas with DPLL search. In *proceedings of CP*, pages 663–678, 2004.

[27] R. Williams, C. Gomes, and B. Selman. Backdoors to typical case complexity. In *proceedings of IJCAI*, pages 1173–1178, 2003.

De l'utilisation de la proportion analogique en apprentissage artificiel

Laurent Miclet, Sabri Bayoudh, Arnaud Delhay, Harold Mouchère

IRISA et ENSSAT
6, rue de Kérampont, BP 80518, 22305 Lannion Cedex
{miclet,bayoudh,delhay,mouchere}@irisa.fr

Résumé : Cet article s'intéresse à la *proportion analogique*, une forme simple du raisonnement par analogie, et décrit son utilisation en apprentissage artificiel. Nous nous attachons plus particulièrement à définir une nouvelle notion, la *dissimilarité analogique* et à l'appliquer à des séquences. Après avoir défini la proportion analogique, la dissimilarité analogique et la résolution approchée d'équations analogiques, nous décrivons deux algorithmes qui rendent opérationnels ces notions pour des objets numériques ou symboliques et pour des séquences de ces objets. Nous montrons ensuite leur efficacité au travers de deux cas pratiques : le premier est l'apprentissage d'une règle de classification pour des objets décrits par des attributs nominaux ; le second montre comment la génération de nouveaux exemples (par résolution approchée d'équations analogiques) peut aider un système de reconnaissance de caractères manuscrits à s'adapter très rapidement à un nouveau scripteur. Une discussion plus générale sur l'apport du raisonnement par analogie pour l'apprentissage artificiel termine cet article.

Mots-clés : Apprentissage, classification, analogie.

1 Introduction

Cet article se propose d'appliquer certaines notions du raisonnement par analogie dans le domaine de l'apprentissage supervisé non paramétrique. En apprentissage supervisé, on dispose d'un ensemble d'apprentissage $\mathcal{S}$ composé d'objets, chacun étant associé à sa supervision (une étiquette de classe, par exemple). La question est de savoir à quelle classe on doit affecter un nouvel objet, à partir de la seule connaissance de l'ensemble d'apprentissage (CORNUÉJOLS & MICLET (2002)).

Le principe de l'apprentissage non paramétrique est de ne faire aucune hypothèse sur la distribution statistique des classes. La technique la plus simple de ce domaine est celle de l'apprentissage par *plus proche voisin* : quand arrive un objet extérieur à $\mathcal{S}$, on lui attribue la classe de l'élément le plus ressemblant dans $\mathcal{S}$.

Dans la même famille, la technique de l'apprentissage par analogie fait appel à un argument plus sophistiqué. Donnons-en un exemple intuitif sur des objets qui sont représentés par des séquences de lettres. Soit la séquence `cherchera`, dont on veut déterminer la classe ; supposons que dans l'ensemble d'apprentissage se trouvent les trois séquences : `recommencer`, `commencera`, `rechercher`, avec respectivement pour classes *infinitif*, *futur*, *infinitif*. On attribuera à `cherchera` la classe *futur*, par un raisonnement qui s'énonce comme ceci. Puisque, dans l'univers des séquences

`recommencer` *est à* `commencera` *comme* `rechercher` *est à* `cherchera`

la supervision de `cherchera` est donc la solution de l'équation sur les classes :

infinitif est à *futur* comme *infinitif* est à x

d'où : $x =$ *futur*.

Ce petit exemple fait apparaître deux notions de base : d'abord celle de *proportion analogique entre quatre objets* (des séquences de lettres dans l'exemple ci-dessus) qui s'exprime sous la forme "A *est à* B *comme* C *est à* D". Il faut bien sûr donner une signification à « comme » et à « est à ». En effet, dans un autre exemple, la proportion

"`jument` *est à* `poulain` *comme* `vache` *est à* `veau`"

porte seulement sur la sémantique des mots, pas sur leur morphologie.

Nous verrons que cette notion peut s'étendre à une proportion approchée, que l'on pourrait exprimer par "A *est à* B *à peu près comme* C *est à* D". Par exemple :

"`jument` *est à* `poulain` *à peu près comme* `vache` *est à* `bufflon`"
"`recommencer` *est à* `commencera` *à peu près comme* `rechercher` *est à* `chercherai`"

Nous donnerons une quantification et une méthode de calcul du terme « à peu près ».

L'autre notion que cet exemple introduit est celle d'*équation analogique* : si l'on connaît trois termes d'une proportion analogique, peut-on calculer le quatrième ? Et peut-on calculer tous les termes qui sont « à peu près » en proportion analogique avec les trois premiers ? Nous donnerons des algorithmes pour résoudre ce problème et nous montrerons en quoi ils sont utiles pour l'apprentissage supervisé.

D'une manière générale, la proportion analogique est un cas particulier du *raisonnement par analogie* qui a été longuement décrit et étudié depuis les philosophes grecs ; ses applications récentes intéressent en particulier les sciences cognitives (Holyoak (2005)), la linguistique et l'intelligence artificielle. LEPAGE (2003) donne une histoire encyclopédique de ce concept et de ses applications à la science du raisonnement et à la linguistique. La restriction à la proportion analogique, en particulier pour les séquences, a été étudiée d'un point de vue méthodologique et algorithmique en particulier dans MITCHELL (1993), HOFSTADTER & the Fluid Analogies Research Group (1994), DASTANI *et al.* (2003)), SCHMID *et al.* (2003), YVON *et al.* (2004). Un autre domaine de l'Intelligence Artificielle est naturellement relié au raisonnement par analogie : celui du raisonnement à base de cas (CBR). Dans le livre de référence AAMODT & PLAZA (1994), ces deux notions sont confondues. Nous reviendrons dans la conclusion sur la distinction entre CBR et apprentissage par proportion analogique et sur la légitimité de ce dernier.

2 Proportion analogique et équations analogiques

2.1 Les axiomes de la proportion analogique

Il n'y a pas de définition générale d'une proportion analogique "A *est à* B *comme* C *est à* D" entre quatre objets pris dans un ensemble X, les relations "est à" et "comme" dépendant de la nature de X. Toutefois, d'après la signification usuelle du mot "analogie" en philosophie et en linguistique, trois axiomes de bases sont généralement requis (LEPAGE & ANDO (1996)) :

Definition 1 (Proportion analogique.)
Une proportion analogique sur X est une relation sur X^4. Quand $(A, B, C, D) \in \mathcal{A}$, les quatre éléments A, B, C et D sont dits en proportion analogique*, ce qui s'écrit $A : B :: C : D$ et se lit "A est à B comme C est à D". Pour chaque 4-uplet en proportion analogique, les trois axiomes suivants sont requis :*

S*ymétrie de la relation "comme" :*	$A : B :: C : D \;\Leftrightarrow\; C : D :: A : B$
E*change des médians :*	$A : B :: C : D \;\Leftrightarrow\; A : C :: B : D$
D*éterminisme :*	$A : A :: B : x \;\Rightarrow\; x = B$

2.2 Equations analogiques

Résoudre une équation analogique consiste à trouver le quatrième terme d'une proportion analogique, les trois premiers étant connus.

Definition 2 (Equation analogique.)
D est une solution à l'équation analogique $A : B :: C : x$ ssi $A : B :: C : D$.

Selon la nature des objets et la définition des relations, une équation analogique peut ne pas avoir de solution, avoir une solution unique ou avoir plusieurs solutions. Nous étudions au paragraphe 3.3 la résolution des équations analogiques dans différents ensembles d'objets et dans des structures séquencielles de ces objets, en proposant une manière leur trouver des solutions approchées si elles n'ont pas de solution.

2.3 Proportions analogiques sur $\mathbb{R}^n$ et sur $\{0, 1\}^n$

Quand on se place dans $\mathbb{R}^n$, il est simple de remarquer que quatre objets sont en proportion analogique quand ils forment un parallélogramme, ce qui peut s'écrire :

$$a : b :: c : d \;\Leftrightarrow\; \overrightarrow{Oa} + \overrightarrow{Od} = \overrightarrow{Ob} + \overrightarrow{Oc} \;\Leftrightarrow\; \overrightarrow{ab} = \overrightarrow{cd}$$

Il est simple aussi de remarquer que quatre objets binaires vérifient les axiomes de la proportion analogique soit quand ils ont tous les quatre la même valeur 0 ou 1, soit quand deux valent 1 et deux valent 0 (sauf dans les cas (0,1,1,0) et (1,0,0,1)). Il y a donc au total 6 quadruplets binaires en analogie parmi les 16 possibles. Quand les objets sont des éléments de $\{0, 1\}^n$, il suffit que chaque quadruplet de coordonnées vérifie l'une des 6 proportions analogiques précédentes pour que les objets vérifient les axiomes de la proportion analogique. Ces définitions, ainsi que celles dans d'autres types d'ensembles comme les groupes cycliques, sont détaillées dans MICLET & DELHAY (2005).

2.4 Proportions analogiques sur les séquences

Nous utilisons maintenant les notions usuelles de la théorie des langages : alphabet Σ, mot (ou séquence) sur $\Sigma^\star$, longueur d'un mot, mot vide ϵ, facteur, sous-séquence. Par exemple, sur l'alphabet $\Sigma = \{a, b\}$, la séquence $aabbaa$ est un élément de $\Sigma^\star$ de longueur $|aabbaa| = 6$, $bbaa$ en est un facteur et aba en est une sous-séquence.

Nous ajoutons une nouvelle lettre à Σ, que nous notons $\backsim$, pour obtenir un alphabet augmenté Σ'. Son interprétation est celle d'un symbole "vide" nécessaire pour les sections qui suivent.

Definition 3 (Equivalence sémantique.)
Soit x une séquence de $\Sigma^\star$ et y une séquence sur $\Sigma'^\star$. Les séquences x et y sont sémantiquement équivalentes *si la sous-séquence de y composée des lettres de Σ est x. Nous notons cette relation par $\equiv$. Par exemple : $ab \backsim a \backsim a \equiv abaa$.*

Un alignement est une correspondance lettre à lettre entre quatre séquences, dans lesquelles des lettres $\backsim$ peuvent être insérées de façon à ce qu'elles prennent la même longueur. La correspondance $(\backsim, \backsim, \backsim, \backsim)$ n'est pas permise.

Definition 4 (Alignement entre quatre séquences.)
Un alignement *entre quatre sequences $u, v, w, x \in \Sigma^*$, est un mot z sur l'alphabet $(\Sigma \cup \{\backsim\})^4 \diagdown \{(\backsim, \backsim, \backsim, \backsim)\}$ dont la projection sur la première, la seconde, la troisième et la quatrième composante sont sémantiquement équivalentes à u, v, w and x.*

Nous supposons qu'il existe une relation de proportion analogique dans Σ', c'est-à-dire que pour chaque quadruplet a, b, c, d dans Σ', la relation $a : b :: c : d$ vaut soit $VRAI$ soit $FAUX$. Nous proposons maintenant de définir la proportion analogique entre quatre séquences d'objets en utilisant à la fois la proportion analogique entre les objets qui les composent et l'alignement entre les quatre séquences.

Definition 5 (Proportion analogique entre séquences.)
Soit u, v, w and x quatre séquences de Σ, sur lequel existe une proportion analogique. Nous disons que u, v, w et x sont en proportion analogique *s'il existe quatre séquences u', v', w' and x' de même longueur dans Σ', avec les propriétés suivantes :*

1. *$u' \equiv u$, $v' \equiv v$, $w' \equiv w$ et $x' \equiv x$.*
2. *$\forall i \in [1, n]$ sont des proportions analogiques $u'_i : v'_i :: w'_i : x'_i$ dans Σ'.*

Par exemple, soit $\Sigma' = \{a, b, \alpha, \beta, B, C, \backsim\}$ avec les proportions analogiques $a : b :: A : B$, $a : \alpha :: b : \beta$ and $A : \alpha :: B : \beta$. L'alignement suivant entre les quatres séquences aBA, αbBA, ba et βba est une proportion analogique sur $\Sigma^\star$:

$$\begin{array}{cccc} a & \backsim & B & A \\ \alpha & b & B & A \\ b & \backsim & a & \backsim \\ \beta & b & a & \backsim \end{array}$$

Nous traitons au paragraphe 3.3 de la résolution des équations analogiques dans les séquences, à partir de cette définition.

3 Dissimilarité analogique

Nous proposons dans cette section de généraliser la relation de proportion analogique entre quatre objets ou quatre séquences d'objets en introduisant la notion de *dissimilarité analogique* (DA). Pour des raisons méthodologiques et opérationnelles, il est souhaitable que la DA soit à la distance ce que la proportion analogique est à l'égalité, c'est à dire qu'elle vérifie les propriétés suivantes :

Cohérence analogique. $DA(u, v, w, x) = 0 \Leftrightarrow u : v :: w : x$

Symétrie de "comme". $DA(u, v, w, x) = DA(w, x, u, v)$

Echange des médians. $DA(u, v, w, x) = DA(u, w, v, x)$

Inégalité triangulaire. $DA(u, v, z, t) \leq DA(u, v, w, x) + DA(w, x, z, t)$

Asymétrie de "est à". En général, $DA(u, v, w, x) \neq DA(v, u, w, x)$

3.1 Dissimilarité analogique entre objets de $\mathbb{R}^n$ et de $\{0, 1\}^n$

Une DA dans $\mathbb{R}^n$ se définit naturellement par $DA(a, b, c, d) = \delta(d, e)$, où e est le quatrième sommet du parallélogramme construit sur a, b et c et où δ est une distance dans $\mathbb{R}^n$. Il est alors facile de vérifier que les propriétés précédentes sont vraies.

Pour quatre objets binaires, nous savons depuis le paragraphe 2.3 qu'il y a six cas de proportions analogiques exactes sur les 16 quadruplets binaires possibles. En affectant la valeur $DA(a, b, c, d) = 0$ à ces six quadruplets, la valeur 1 à tous ceux qui ont trois 0 et un 1 (ou le contraire) et la valeur 2 aux quadruplets $(0, 1, 1, 0)$ et $(1, 0, 0, 1)$, on démontre que les propriétés ci-dessus sont vérifiées.

Pour des objets de $\{0, 1\}^n$, additionner les valeurs précédentes sur les coordonnées (puis diviser par n si l'on veut normaliser) produit également une dissimilarité analogique ayant les quatre propriétés désirées (MICLET & DELHAY (2005)). C'est cette définition que nous utiliserons dans la suite.

3.2 Dissimilarité analogique entre séquences : définition et calcul.

Nous pouvons maintenant définir la notion de DA sur des séquences d'objets pour lesquels une DA a été définie (et étendue pour prendre en compte le symbole $\backsim$).

Definition 6 (Dissimilarité analogique entre séquences.)
Le coût *d'un alignement entre quatre séquences est la somme des dissimilarités analogiques entre les quadruplets de lettres définis par cet alignement. La* dissimilarité analogique entre quatre séquences *est le coût de l'alignement le moins coûteux entre les quatre séquences.*

La définition précédente permet de calculer la dissimilarité analogique $DA(u, v, w, x)$ avec un algorithme de programmation dynamique (appelé SEQUANA4), qui progresse de façon synchrone dans les quatres séquences pour construire un alignement optimal. L'entrée de cet algorithme est un alphabet augmenté Σ' sur lequel une dissimilarité analogique $DA(a, b, c, d)$ a été définie. La sortie est la dissimilarité analogique entre quatre séquences $\Sigma^\star$, c'est-à-dire $DA(u, v, w, x)$. Cet algorithme a une complexité en temps en $\mathcal{O}\big(|u|.|v|.|w|.|x|\big)$.

La validité de SEQUANA4 peut être montrée par récurrence, car il est construit sur le principe de la programmation dynamique. La DA calculée entre les séquences possède les propriétés suivantes : *cohérence avec l'analogie*, *symétrie du "comme"* et *l'échange des médians*. La propriété d'*inégalité triangulaire* n'est pas en général assurée.

3.3 Solutions approchées aux équations analogiques

Nous traitons dans ce paragraphe du calcul de la solution d'une équation analogique entre séquences, et ce d'une manière étendue : nous donnons en effet un algorithme capable de produire toutes les meilleures solutions à ce problème, c'est à dire de construire toutes les séquences en dissimilarité analogique minimale avec trois séquences données.

Definition 7 (Meilleure solution approchée à une équation analogique.)
Soit X un ensemble sur lequel est défini une analogie et une dissimilarité analogique DA. Soit $a : b :: c : x$ une équation analogique dans X. L'ensemble des meilleures solutions approchées à cette équation est donnée par :

$$\big\{y \ : \ \underset{y \in X}{ArgMin}\ DA(a, b, c, y)\big\}$$

Ce sont donc les objets $y \in X$ qui sont les plus proches d'être en proportion analogique avec a, b et c. Cette définition s'applique au cas de la solution exacte à une équation analogique quand on force la dissimilarité analogique à être nulle.

Nous pouvons facilement élargir ce concept et définir l'ensemble de k-meilleures solutions à une équation analogique $a : b :: c : x$. Informellement, c'est le sous-ensemble de k éléments de X qui ont une DA minimale quand ils sont associés en quatrième position de l'équation avec a, b et c.

Dans $\mathbb{R}^n$, il n'y a qu'une seule meilleure solution approchée à une équation analogique, qui peut-être explicitement calculée. Dans $\{0, 1\}^n$, l'approche naïve est d'examiner chaque élément de l'ensemble et de garder les y qui minimisent $DA(a, b, c, y)$. Comme dans ce cas DA possède la propriété d'inégalité triangulaire, nous avons montré dans Bayoudh *et al.* (2007) qu'un algorithme plus rapide pouvait être utilisé.

Le cas des séquences : l'algorithme SOLVANA

Nous décrivons maintenant un algorithme (appelé SOLVANA) qui trouve l'ensemble des meilleures solutions approchées à l'équation $a : b :: c : x$ quand les objets sont des séquences sur un alphabet étendu, sur lequel une DA a été définie.

Cet algorithme utilise la programmation dynamique pour construire un tableau à 3 dimensions. Quand cette construction est terminée, un retour en arrière est fait pour produire un graphe de toutes les meilleures solutions.

L'alignement de quatre séquences de longueurs différentes est réalisé en insérant optimalement des lettres $\backsim$ dans les trois séquences connues et en calculant en ligne la quatrième séquence, de telle façon que les quatre séquences aient finalement la même longueur. En parallèle, on cumule les dissimilarités analogiques dans l'alphabet augmenté sur chaque étape de l'alignement des séquences. La complexité algorithmique de cet algorithme est en $O(m * p^3)$, où $m = Card(\Sigma')$ et p est la longueur moyenne des séquences.

Si l'on cherche non pas l'ensemble des solutions optimales, mais les k meilleures solutions pour k quelconque, cet algorithme n'est plus adapté. Il faut alors employer une technique du type *branch and bound*, comme l'algorithme $A^\star$ (en version de base avec une heuristique identiquement nulle ou de manière plus élaborée si l'on peut définir une fonction heuristique admissible).

4 Applications en apprentissage artificiel

4.1 Classification d'objets binaires et nominaux

Règle de classification par analogie

Soit $\mathcal{S} = \big\{\big(c_i, h(c_i)\big) \mid 1 \leq i \leq m\big\}$ l'ensemble d'apprentissage, avec $h(c_i)$ la classe de l'objet c_i. Les objets sont définis par des attributs binaires (les attributs nominaux sont binarisés par codage *one per value*). Soit x un objet n'appartenant pas à $\mathcal{S}$. Pour trouver la classe de x, on définit une règle d'apprentissage par analogie dépendant d'un entier k par les étapes suivantes :

1. Calculer la dissimilarité analogique entre x et touts les n triplets appartenant à $\mathcal{S}$ dont la résolution dans les classes a la forme suivante : "$\omega_i \; : \; \omega_i \; :: \; \omega_j \; : ?$", ce qui donne "$? = \omega_j$".
2. Ordonner ces n triplets par ordre croissant par rapport à leurs valeurs de DA, quand ils sont associés à x.
3. Si le k^{eme} triplet a la valeur p, alors soit k' le plus grand nombre tel que le triplet k'^{eme} a la même valeur p.
4. Résoudre les k' équations analogiques sur les classes. Choisir la classe gagnante qui comptabilise le plus grand nombre de votes parmi les k' résultats.

Pondération des attributs pour la règle de classification par analogie

Nous ne détaillons pas dans cette section la méthode de pondération (voir Bayoudh *et al.* (2007)) mais nous en donnons l'idée de base : puisque tous les attributs n'ont pas la même importance pour la classification, il faut donner une plus grande importance aux plus discriminants. Cependant, dans la classification par analogie, il y a plusieurs façons de conclure l'appartenance d'un élément à une certaine classe. C'est pourquoi nous avons défini un poids par classe pour chaque attribut.

Resultats

Nous avons appliqué la classification par proportion analogique ($WAPC$: Weighted Analogical Proportion Classifier) sur huit bases de données prise du UCI Repository de NEWMAN *et al.* (1998). Afin de mesurer l'efficacité du classificateur $WAPC$, nous l'avons comparé à six classificateurs standard. Les résultats obtenus sont donnés dans le tableau 1. Ils montrent la qualité et la polyvalence de la méthode $WAPC$ (problèmes multiclasses, données manquantes, etc.).

Methods	*MO.1*	*MO.2*	*MO.3*	*SP.*	*B.S*	*Br.*	*H.R*	*Mu.*
nombre d'attributs nominaux	7	7	7	22	4	9	4	22
nombre d'attributs binaires	15	15	15	22	4	9	4	22
nombre d'instances d'apprentissage	124	169	122	80	187	35	66	81
nombre d'instances de test	432	432	432	172	438	664	66	8043
nombre de classes	2	2	2	2	3	2	4	2
WAPC ($k = 100$)	**98%**	**100%**	**96%**	**79%**	**86%**	**96%**	**82%**	**98%**
APC ($k = 100$)	98%	100%	96%	58%	86%	91%	74%	97%
Decision Table	100%	64%	97%	65%	67%	86%	42%	99%
Id3	78%	65%	94%	71%	54%	mv	71%	mv
PART	93%	78%	98%	81%	76%	88%	82%	94%
Multi layer Perceptron	100%	100%	94%	73%	89%	96%	77%	96%
LMT	94%	76%	97%	77%	89%	88%	83%	94%
IB1	79%	74%	83%	80%	62%	96%	56%	98%

TAB. 1 – Tableau Comparatif entre $WAPC$ et d'autres classificateurs.

$9 \backsim 9 \backsim 9\,9\,9\,9\,9 \backsim E\ 1\ 2 \backsim 4\ L\ 6\ 9\ 9\ 9$
$1\ L \backsim 8\ 9\,9\,9\,9\,9\,10\ E \backsim 2\ 2\ 4\ L \backsim 8\ 8\ 9$
$\backsim \backsim 9\ 8\ 9\,9\,9\,9\,9\,10\ E\ 2\ 2\ 3\ 3\ L\ 8\ 9\ 9 \backsim$
$1\ L \backsim 8\ 9\,9\,9\,9\,9\,10\ E\ 2\ 2\ 3\ 3\ L\ 8\ 8\ 8 \backsim$

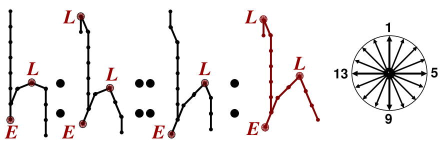

FIG. 1 – Resolution d'équations analogiques sur le code de Freeman augmenté pour la génération de caractères.

4.2 Reconnaissance de caractères manuscrits : génération de nouveaux exemples

Avec l'utilisation croissante du stylo dans l'interface homme machine, la reconnaissance des caractères manuscrits doit être de plus en plus précise et s'adapter rapidement aux nouveaux styles d'écritures (Mouchère *et al.* (2007)). La difficulté réside dans l'apprentissage avec très peu d'exemples. On peut augmenter efficacement l'ensemble d'apprentissage en générant automatiquement de nouveaux exemples. Pour ce problème, plusieurs stratégies de génération sont déjà employées. La première utilise la distortion d'image, la seconde est basée sur des transformations de la saisie on-line du scripteur (Mouchère & Anquetil (2006)). Nous proposons ici d'utiliser la proportion analogique pour la génération d'exemples.

Chaque caractère manuscrit est décrit par le code de Freeman à seize directions et enrichi par des points d'ancrages (représentés ici par des lettres majuscules). Les points d'ancrage sont par exemple des maxima ou des minima ou des variations angulaires. Ainsi, les caractères sont représentés par des séquences. En utilisant l'algorithme SOLVANA, on génère des nouvelles séquences synthétiques à partir de trois séquences originales (voir figure 1).

Afin de voir l'apport de la génération d'instance synthétique, nous avons appliqué les différentes méthodes de génération sur douze scripteurs qui ont saisi chacun 40 caractères de chaque lettre de l'alphabet sur un PDA. Ayant 1040 caractères de chaque scripteur, nous avons pris 2,3, 4 et 6 caractères de chaque lettre de l'alphabet pour en générer 300 afin d'apprendre un classificateur (il s'agit d'une méthode RBFN). Enfin on calcule le taux de reconnaissance sur l'ensemble de test composé des caractères restant non utilisée en apprentissage (voir table 2). Pour mieux voir l'apport de la génération, notons que le taux de reconnaissance en utilisant 10 et 30 caractères pour l'apprentissage sans génération est respectivement 82.3% et 94.5%. En résumé, le nouveau scripteur peut écrire trois fois moins d'exemples pour avoir une qualité de reconnaissance égale.

Nb. de caractères utilisés	2	3	4	6
(1) : Distortions d'image	76.1 $\pm$ 3.3	82.5 $\pm$2.4	85.8 $\pm$2.0	89.4 $\pm$1.7
(2) : On-line et (1)	84.4 $\pm$2.6	88.0 $\pm$2.1	90.3 $\pm$1.7	92.3 $\pm$1.6
(3) : Analogie et (2)	84.9 $\pm$2.6	89.3 $\pm$2.1	91.3 $\pm$1.4	93.5 $\pm$1.1

TAB. 2 – Taux de reconnaissance moyen $\pm$écart type pour trois méthodes de génération

5 Conclusion

En conclusion, cet article a montré qu'une définition fondée de la proportion analogique et de la dissimilarité analogique, ainsi que leur mise en forme algorithmique, peuvent apporter de bons résultats dans deux domaines de l'apprentissage artificiel. Pour prendre un peu de recul, nous allons d'abord distinguer ce travail de ceux réalisés dans le cadre du CBR et de l'analogie en général. Ensuite, nous tenterons de dégager d'autres domaines pour lesquels cette approche pourrait se révéler intéressante.

En Intelligence Artificielle, le raisonnement par analogie est souvent, comme la citation de l'introduction l'a déjà évoqué, pratiquement confondu avec le CBR. Il s'agit de transférer de l'information d'un domaine bien connu vers un autre, en utilisant le fait que les deux domaines ont une structure commune (GENTNER *et al.* (2001)). Dans le cas particulier de la proportion analogique, les deux domaines sont confondus. Cette restriction a l'inconvénient de limiter les ambitions (en particulier sur la véracité cognitive du modèle), mais en revanche permet comme on l'a vu de donner un cadre formel et algorithmique fondé.

Pour en rester aux problèmes d'apprentissage de règles de classification, la question centrale est de savoir s'il est plausible que des objets en proportion analogique aient obligatoirement des classes également en proportion analogique. C'est en effet sur cette hypothèse que notre étude est fondée. Si l'on s'en tient, comme nous l'avons fait, à des proportions analogiques triviales dans les classes, le raisonnement est expérimentalement valable, comme le montre l'expérience du paragraphe 4.1. Il serait intéressant de le poursuivre en extrayant de l'ensemble d'apprentissage des proportions analogiques approchées, du genre “la classe 1 est à la classe 3 à peu près comme la classe 2 est à la classe 5”, ou, dans le cas des caractères manuscrits, “a est à d à peu près comme o est à q”. On réaliserait alors un système de transfert analogique entre l'univers des objets et celui des classes. Ceci a déjà été réalisé, par exemple dans YVON (1997), où les objets sont la forme orthographique de noms propres et les “classes” les formes phonétiques.

D'autre part, nous envisageons d'étendre nos études à des données structurées en arbre, par exemple pour l'apprentissage de la prosodie en synthèse de la parole.

Références

AAMODT A. & PLAZA E. (1994). Case-based reasoning : Foundational issues, methodological variations, and system approaches. *Artificial Intelligence Communications*, **7**(1), 39–59.

BAYOUDH S., MICLET L. & DELHAY A. (2007). Learning by analogy : a classification rule for binary and nominal data. In *Proc. of the Int. Joint Conf. on Artificial Intelligence*, volume 20, p. 678–683.

CORNUÉJOLS A. & MICLET L. (2002). *Apprentissage artificiel : concepts et algorithmes*. Eyrolles, Paris.

DASTANI M., INDURKHYA B. & SCHA R. (2003). Analogical projection in pattern perception. *Journal of Experimental and Theoretical Artificial Intelligence*, **15**(4).

GENTNER D., HOLYOAK K. J. & KOKINOV B. (2001). *The analogical mind : Perspectives from cognitive science*. Cambridge, MA : MIT Press.

HOFSTADTER D. & THE FLUID ANALOGIES RESEARCH GROUP (1994). *Fluid Concepts and Creative Analogies*. New York : Basic Books.

HOLYOAK K. J. (2005). Analogy. In *The Cambridge Handbook of Thinking and Reasoning*, p. 117–142 : Cambridge University Press.

LEPAGE Y. (2003). *De l'analogie rendant compte de la commutation en linguistique*. Grenoble. Habilitation à diriger les recherches.

LEPAGE Y. & ANDO S. (1996). Saussurian analogy : a theoretical account and its application. In *Proceedings of COLING-96*, p. 717–722, København.

MICLET L. & DELHAY A. (2005). *Analogical Dissimilarity : definition, algorithms and first experiments in machine learning*. Rapport interne 5694, INRIA.

MITCHELL M. (1993). *Analogy-Making as Perception*. Cambridge, MA : MIT Press.

MOUCHÈRE H., ANQUETIL E. & RAGOT N. (2007). Writer style adaptation in on-line handwriting recognizers by a fuzzy mechanism approach : The adapt method. *Int. Journal of Pattern Recognition and Artificial Intelligence*, **21**(1), 99–116.

MOUCHÈRE H. & ANQUETIL E. (2006). Synthèse de caractères manuscrits en-ligne pour la reconnaissance de lécriture. In *Actes du Colloque International Francophone sur l'Ecrit et le Document (CIFED'06)*, p. 187–192.

NEWMAN D., HETTICH S., BLAKE C. & MERZ C. (1998). UCI repository of machine learning databases.

SCHMID U., GUST H., KÜHNBERGER K.-U. & BURGHARDT J. (2003). An algebraic framework for solving proportional and predictive analogies. In R. Y. F. SCHMALHOFER & G. KATZ, Eds., *Proceedings of the European Conference on Cognitive Science (EuroCogSci 2003)*, p. 295–300, Osnabrück, Germany : Lawrence Erlbaum.

YVON F. (1997). Paradigmatic cascades : a linguistically sound model of pronunciation by analogy. In *Proceedings of the 35th annual meeting of the Association for Computational Linguistics (ACL)*, Madrid, Spain.

YVON F., STROPPA N., DELHAY A. & MICLET L. (2004). *Solving analogical equations on words*. Rapport interne ENST2004D005, École Nationale Supérieure des Télécommunications.

Une approche programmation par contraintes pour la modélisation et la planification de mouvements en robotique humanoïde.

Mathias PAULIN

LIRMM - Univ. Montpellier 2, CNRS
161 rue Ada 34392 Montpellier
`Mathias.Paulin@lirmm.fr`

Résumé : Il existe de nombreuses démonstrations de comportements évolués pour les robots humanoïdes de dernière génération (montée de marches, préhension d'objet, *etc.*). Toutefois, il s'agit dans la plupart des cas de comportements préenregistrés qui nécessitent de nombreux calculs, et qui s'adaptent difficilement aux modifications de contexte. Dans cet article, nous présentons une approche innovante qui modélise les actions élémentaires d'un robot sous la forme de réseaux de contraintes, et combine ensuite ceux-ci à l'aide d'outils de planification. Lors de l'exécution d'une séquence d'actions, nous utilisons les bonnes propriétés des réseaux de contraintes pour contrôler et éventuellement corriger l'exécution d'une tâche. L'utilisation des contraintes fournit au robot un modèle de lui même et de son environnement, grâce auquel il peut raisonner sur ses comportements, ce qui lui confère une première forme d'autonomie. La fin de l'article présente des résultats expérimentaux préliminaires permettant de valider l'intérêt de notre approche, dont on peut désormais envisager le déploiement effectif sur un robot humanoïde tel que HOAP3 de FUJITSU.

Mots-clés : Programmation par contraintes, Planification de tâches.

1 Introduction

Les dernières générations de robots humanoïdes comme ASIMO (Honda) ou HRP-2 (Kawada) démontrent que des progrès colossaux ont été accomplis en termes d'habiletés physiques. Il existe désormais de nombreuses démonstrations de comportements évolués, telles que la préhension d'objets, la montée de marches, ou la danse. Toutefois, il s'agit dans la plupart des cas de comportements calculés à l'avance, qui sont ensuite déroulés de manière automatique, et qui s'adaptent difficilement aux modifications de contexte.

La difficulté à planifier des comportements est la conséquence de l'approche actuelle des roboticiens, qui s'attache à modéliser au plus près chaque action élémentaire en utilisant les lois physiques du monde (loi de conservation des énergies, moments cinétiques, *etc.*). Combiner plusieurs actions élémentaires se révèle être particulièrement difficile et nécessite de nombreux calculs, pouvant aller jusqu'à plusieurs heures. Pour pallier cet écueil, les mécaniciens tentent d'utiliser des modèles cinématiques simplifiés, ainsi que la recherche de comportements optimaux par analyse des comportements humains. L'étude de la dynamique posturale chez l'homme a ainsi permis de définir des dynamiques de marche très intéressantes (Collins & Luca, 1993). Cependant, les comportements ainsi générés sont très difficiles à exploiter dans un contexte réel. En effet, la modélisation employée ne tient pas compte des perturbations extérieures qui peuvent se produire lors d'une exécution réelle. Un certain nombre de techniques sont alors utilisées pour permettre une exploitation plus large. De nombreux travaux ont ainsi été réalisés dans l'apprentissage, via des réseaux de neurones, d'actions réflexes automatiques susceptibles de tolérer des perturbations extérieures (Henaff, 2007).

Les liens entre l'intelligence artificielle et la robotique humanoïde se font de plus en plus étroits (réseaux de neurones, reconnaissance vocale, vision stéréoscopique, . . .). Malgré ces liens, entre les robots actuels et le rêve de robots autonomes, capables de raisonner sur leurs comportements et de s'adapter à des modifications de contexte, l'écart reste encore important.

Dans cet article, nous proposons une amélioration de l'architecture logicielle proposée dans (Paulin, 2006), qui utilise le paradigme de la Programmation Par Contraintes (PPC) pour modéliser et planifier des comportements en robotique humanoïde. Notre approche consiste dans un premier temps à modéliser par apprentissage automatique les habiletés physiques d'un robot à l'aide de réseaux de contraintes (CSP), qui sont ensuite composés à l'aide de techniques de planification de tâches. Lors de l'exécution d'une séquence d'actions, nous utilisons les bonnes propriétés des CSP acquis, pour contrôler en temps réel la réalisation d'une tâche, en corrigeant les éventuels écarts existants entre les prévisions fournies par les CSP et les valeurs réelles.

Notre approche cherche à s'abstraire des lois physiques qui régissent la commande des robots humanoïdes. En effet, les réseaux de contraintes qu'elle manipule modélisent les conditions dans lesquelles chaque action élémentaire peut être exécutée, c'est-à-dire le *comment* et non le *pourquoi*. Cette modélisation est plus faible que ce qui est "mécaniquement" nécessaire, mais est en contrepartie beaucoup plus facile à manipuler car elle fournit, pour chaque action élémentaire, un *modèle* qui abstrait un espace de solutions possibles. La planification de séquences d'actions est en conséquence beaucoup plus aisée. Les réseaux de contraintes fournissent par ailleurs un modèle à même d'expliquer un état futur, la cause d'une replanification de comportement, ou bien encore les raisons d'un échec. Cette approche permet ainsi à un robot de raisonner sur ses aptitudes physiques, ce qui constitue un premier pas vers une autonomie comportementale non réflexe.

Le reste de cet article est organisé comme suit. Dans la section 2, nous introduisons les concepts utilisés dans l'article. La section 3 présente l'architecture générale de notre système de supervision, que nous définissons formellement dans la section 4. Avant de conclure en présentant nos perspectives de travail (section 6), la section 5 décrit quelques expérimentations préliminaires permettant de valider empiriquement l'intérêt de l'approche PPC pour la supervision de robots humanoïdes.

2 Préliminaires

2.1 Rappel sur la programmation par contraintes

Un réseau de contraintes $P = (\mathcal{X}, \mathcal{D}, \mathcal{C})$ est défini par un ensemble fini $\mathcal{X} = \{x_1, \ldots, x_n\}$ de n variables prenant chacune une valeur de leur domaine respectif $D_{x_1}, \ldots, D_{x_n}$, éléments de $\mathcal{D}$, et par $\mathcal{C} = \{c_1, \ldots, c_m\}$ une séquence de contraintes sur $\mathcal{X}$. Une contrainte c_i est définie par la séquence $var(c_i)$ des variables sur lesquelles elle porte, et par la relation $sol(c_i)$ qui spécifie les n-uplets autorisés sur $var(c_i)$. L'affectation de valeurs aux variables de $var(c_i)$ satisfait c_i si elle appartient à $sol(c_i)$. Une instance e sur $\mathcal{X}$ est un n-uplet $(v_1, \ldots, v_n) \in D_{x_1} \times \ldots \times D_{x_n}$. On dit qu'une instance est une solution du réseau si elle satisfait toutes les contraintes du réseau. Sinon, c'est une non solution. On note $Sol(\mathcal{X}, \mathcal{D}, \mathcal{C})$ l'ensemble des solutions de $(\mathcal{X}, \mathcal{D}, \mathcal{C})$.

La programmation par contraintes connaît un succès croissant et est désormais largement utilisée pour résoudre des problèmes réels difficiles hors de portée des autres approches (Wallace, 1996). La principale raison de ce succès réside dans son aspect déclaratif et dans la dissociation inhérente entre un modèle décrivant le problème (le "Quoi") et les techniques de résolution utilisées pour la recherche de solutions valides (le "Comment"). De nombreux travaux académiques et industriels ont permis la mise en place d'un large panel d'outils, tels que les contraintes globales (Bessiere & Hentenryck, 2003), les contraintes relaxées (Gaspin *et al.*, 2005) ou les QCSP (Benedetti *et al.*, 2006; Verger & Bessiere, 2006), qui permettent de modéliser une grande variété de problèmes. Par ailleurs, les dernières générations de résolveurs de contraintes comme Ilog Solveur ou Choco sont particulièrement performants et sont capables de déterminer efficacement si un problème est soluble et, le cas échéant, trouver une solution.

Forte de trente années de recherche et de progrès constants, la programmation par contraintes se révèle être un paradigme de choix, tant par son expressivité que par son efficacité à résoudre des problèmes d'envergure.

2.2 Le problème de planification de tâches

Dans sa définition la plus commune (Russell & Norvig, 2003), la planification classique consiste à établir, avant leur exécution, un ordre d'application d'actions afin d'atteindre un objectif défini dans un environnement particulier appellé *Monde*. Le formalisme STRIPS communément utilisé pour exprimer des problèmes de planification de tâches manipule les concepts d'état, de but et d'action.

Un *état* est une conjonction de littéraux positifs décrivant le *Monde* à un instant donné. Un *but* est un état particulier qui décrit de manière partielle le *Monde* dans un état final. On dit alors qu'un état s satisfait un but b si s contient tous les littéraux de b. Une *action* A est caractérisée par son nom, ses paramètres, sa pré-condition et son effet. La *pré-condition* de A est une conjonction propositionnelle qui représente l'ensemble des conditions qui doivent être vérifiées pour que A puisse être exécutée. L'*effet* de A est une conjonction propositionnelle qui décrit les changements qui surviennent lorsque A est exécutée.

Étant donné E_i un état initial correspondant à une description complète du *Monde*, E_f un état final correspondant à un but, et $\mathcal{A}$ un ensemble d'actions, le problème de planification de tâches consiste à trouver une séquence d'actions issues de $\mathcal{A}$ telle que si ces actions sont exécutées à partir de E_i, elles permettent d'aboutir à un état qui satisfait E_f.

3 Architecture générale du système de supervision

La modélisation des lois physiques actuellement utilisée par les roboticiens se révèle très difficile à manipuler lorsque l'on cherche à planifier une séquence d'actions. L'approche que nous proposons consiste à modéliser les actions élémentaires d'un robot à l'aide de réseaux de contraintes simples, ayant de bonnes propriétés calculatoires, puis à combiner et composer ces réseaux afin de planifier des comportements.

Nous utilisons la plate-forme CONACQ (Bessiere *et al.*, 2005) pour modéliser les actions élémentaires d'un robot par apprentissage automatique. Pour modéliser chaque action élémentaire a_i, il suffit de fournir un ensemble d'instances valides de a_i, ainsi qu'un ensemble d'instances ne correspondant pas à a_i. CONACQ modélise alors automatiquement a_i sous la forme d'un réseau de contraintes qui décrit les conditions dans lesquelles a_i peut être exécutée (*i.e.* sa précondition), ses effets, et enfin la manière dont les différents actionneurs du robot doivent réagir pour effectuer a_i. Les réseaux de contraintes ainsi acquis deviennent alors des unités élémentaires de contrôle que l'on peut composer à l'aide des outils de planification (figure 1).

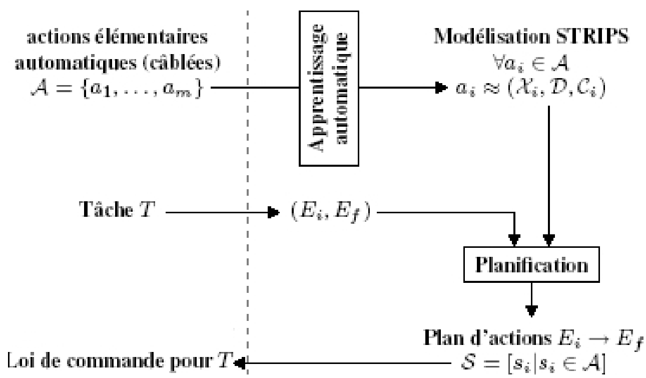

FIG. 1 – Acquisition des unités élémentaires de contrôle et planification.

La figure 2 illustre le principe de fonctionnement de notre système de supervision. Étant donnée une tâche T ne pouvant être réalisée au moyen d'une unique action élémentaire, un planificateur de tâches se charge de trouver une séquence S d'actions élémentaires dont l'exécution doit permettre de réaliser T. Pour réaliser ses calculs, le planificateur manipule et compose les unités élémentaires de contrôle modélisées par les réseaux de contraintes à l'étape précédente. Une fois calculée, la séquence S est transmise au module de contrôle qui se charge de faire exécuter cette séquence d'actions au robot ou à son simulateur. Cette exécution se fait alors pas à pas, c'est-à-dire une action après l'autre. Après l'exécution de chaque action élémentaire, le module de contrôle confronte les mesures-capteurs réelles aux prévisions établies via la modélisation en contraintes. En cas d'écart mineur, un simple appel au solveur de contraintes permettra d'ajuster les prochaines commandes moteur à exécuter. En revanche, si l'écart entre les prévisions et la réalité est trop important, le module de supervision fera à nouveau appel au planificateur pour établir une nouvelle séquence d'actions permettant d'atteindre le but T depuis le nouvel état courant.

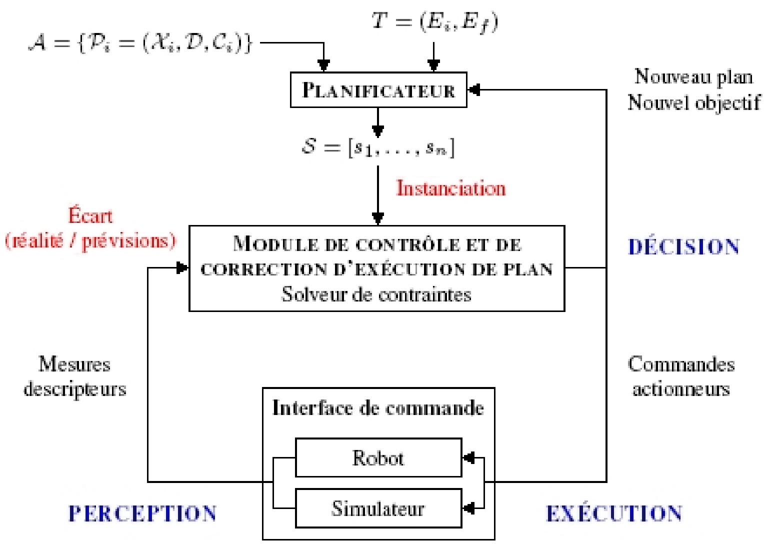

FIG. 2 – Contrôle et correction de l'exécution d'une séquence d'actions.

4 Formalisation du processus de supervision

Dans cette section, nous définissons de manière formelle le processus de supervision de notre plate-forme. Nous considérons pour cela un robot R constitué de p descripteurs (ses capteurs), qui décrivent son état courant, et de q actionneurs (ses moteurs). Soient $\delta = \{\delta_1, ..., \delta_p\}$ l'ensemble des descripteurs de R, $\alpha = \{\alpha_1, ..., \alpha_q\}$ l'ensemble de ses actionneurs et $\mathcal{A} = \{a_1, \ldots, a_m\}$ l'ensemble des actions élémentaires qu'il peut exécuter.

4.1 Apprentissage des unités élémentaires de contrôle

Comme nous l'avons décrit dans la section précédente, notre approche consiste à modéliser chaque action élémentaire $a_i \in \mathcal{A}$ à l'aide d'un réseau de contraintes. Afin d'acquérir ces unités élémentaires de contrôle, nous définissons les fonctions $varI$, $varA$ et $varF$ définies sur $\mathcal{A}$ de la manière suivante : $varI(a_i)$ renvoie l'ensemble

$\{\delta_1^I, \ldots, \delta_p^I\}$ des variables qui modélisent les descripteurs de R **avant** l'exécution de a_i, $varA(a_i)$ renvoie l'ensemble $\{\alpha_1, \ldots, \alpha_q\}$ des variables qui modélisent les actionneurs de R **pendant** l'exécution de a_i et $varF(a_i)$ renvoie l'ensemble $\{\delta_1^F, \ldots, \delta_p^F\}$ des variables qui modélisent les descripteurs de R **après** l'exécution de a_i.

Chaque action élémentaire $a_i \in \mathcal{A}$ est alors modélisée en suivant la définition 1.

Définition 1 (Modélisation par contraintes)
Une action élémentaire $a_i \in \mathcal{A}$ est modélisée par le réseau de contraintes $\mathcal{P}_i = (X_i, D, C_i)$ tel que :

$$\begin{cases} X_i = varI(a_i) \cup varA(a_i) \cup varF(a_i), \\ D = \{D(x_j) | x_j \in X_i \text{ et } D(x_j) \text{ est le domaine de valeurs possibles pour } x_j\}, \\ C_i = Pre(a_i) \cup Action(a_i) \cup Post(a_i). \end{cases}$$

où $Pre(a_i)$ est l'ensemble des contraintes qui modélisent les conditions dans lesquelles a_i peut être exécutée, $Action(a_i)$ est l'ensemble des contraintes qui expriment comment a_i est exécutée, et $Post(a_i)$ est l'ensemble des contraintes qui modélisent les effets de a_i sur le robot et son environnement.

4.2 Planification d'une séquence d'actions

Après acquisition des unités élémentaires de contrôle, chaque action élémentaire $a_i \in \mathcal{A}$ est modélisée par le réseau $\mathcal{P}_i$. Il reste alors à combiner ces différentes unités de contrôle pour établir une séquence $\mathcal{S}$ d'actions élémentaires que le robot devra exécuter pour accomplir une tâche T.

Nous exprimons chaque action élémentaire a_i dans le formalisme STRIPS. La précondition de a_i est donnée par restriction du réseau $\mathcal{P}_i$ à $Pre(a_i)$, et son effet par $Post(a_i)$. La tâche T est quant à elle exprimée au moyen de l'état initial E_i et de l'état final E_f (*i.e.* le but de T) tels que définis à la section 2.2. Exprimé de la sorte, le problème de départ consistant à trouver une séquence d'actions élémentaires permettant de réaliser T revient désormais à résoudre le problème de planification de tâches $(E_i, E_f, \mathcal{A})$. Pour le résoudre, nous faisons appel à un planificateur de tâches utilisant le formalisme STRIPS.

S'il existe un plan permettant d'atteindre E_f à partir de E_i, le planificateur renvoie une séquence $\mathcal{S} = [s_i | s_i \in \mathcal{A}]$ qui renferme les différentes actions élémentaires a_i à exécuter. $\mathcal{S}$ correspond dans ce cas à une séquence respectant les conditions d'enchaînements des actions élémentaires permettant d'accomplir T. En l'état, elle ne permet cependant pas de déterminer quelles tensions doivent être successivement appliquées aux moteurs du robot pour que ce dernier réalise effectivement T.

4.3 Instanciation d'une séquence d'actions élémentaires

Soit T une tâche non élémentaire et $\mathcal{S} = [s_1, \ldots, s_n]$ une séquence d'actions élémentaires planifiée selon le processus décrit précédemment. Pour déterminer quelles sont les tensions à appliquer pour réaliser T, il faut dans un premier temps créer le réseau de contraintes, appelé dans la suite CSP global, correspondant à $\mathcal{S}$. Ce CSP global est défini par instanciation (au sens *Objet* du terme) des réseaux de contraintes modélisés lors du processus d'apprentissage, et subit une phase de propagation pour éliminer les valeurs inconsistantes. Dans un second temps, il convient de résoudre ce CSP global afin d'obtenir une séquence d'actions exécutable par le robot.

L'instanciation d'une action élémentaire $a_i \in \mathcal{A}$ à un instant t se définit à l'aide des définitions 2 et 3.

Définition 2 (Instanciation des variables)
Soit $a_i \in \mathcal{A}$ une action élémentaire modélisée par le réseau de contraintes $\mathcal{P}_i = (X_i, D, C_i)$ où $X_i = \{\delta_1^I, \ldots, \delta_p^I, \alpha_1, \ldots, \alpha_q, \delta_1^F, \ldots, \delta_p^F\}$. Une variable $x \in X_i$ est instanciée à un instant t au moyen de la fonction $inst$ définie sur $X_i \times \mathbb{N}$ comme suit :

$$inst(x,t) = \begin{cases} \delta_j^{I,t} & \exists j \in [1..p] \text{ tel que } x = \delta_j^I \\ \alpha_j^t & \exists j \in [1..q] \text{ tel que } x = \alpha_j \\ \delta_j^{F,t} & \exists j \in [1..p] \text{ tel que } x = \delta_j^F \end{cases}$$

On note alors $inst(X_i, t) = \{inst(x,t) | x \in X_i\}$ l'instanciation de l'ensemble X_i à l'instant t.

Définition 3 (Instanciation d'une action)
L'instanciation de $a_i \in \mathcal{A}$ à un instant t, notée $a_{i,t}$ est définie par le triplet $(X_{i,t}, D, C_{i,t})$ tel que :

$$\begin{cases} X_{i,t} = inst(X_i, t), \\ C_{i,t} = \{c_{i,t} = (scope, sol(c_i)) | c_i \in C_i \text{ et } scope = inst(var(c_i), t)\}. \end{cases}$$

Afin d'associer à chaque élément s_t de $\mathcal{S}$ l'action élémentaire $a_i \in \mathcal{A}$ que l'on doit effectuer à l'étape t, on utilise la fonction d'association φ, définie sur $\{1..n\}$, et à valeurs dans $\{1..m\}$, qui à t associe $\varphi(t) = i$ si et seulement si $a_i \in \mathcal{A}$ est l'action à exécuter à l'instant t. La séquence $\mathcal{S}$ s'écrit alors $\mathcal{S} = [a_{\varphi(1),1}, \ldots, a_{\varphi(n),n}]$. Afin de déterminer quelles tensions doivent être successivement appliquées aux actionneurs du robot, nous devons créer le réseau de contraintes correspondant à l'exécution de la séquence $\mathcal{S} = [a_{\varphi(1),1}, \ldots, a_{\varphi(n),n}]$, qui est défini comme suit :

Définition 4 (CSP global)
Le réseau de contraintes permettant d'instancier $\mathcal{S}$ est le réseau de contraintes $\mathcal{P} = (X, D, C)$ où :

$$\begin{cases} X = \displaystyle\bigcup_{t=1}^{n} X_{\varphi(t),t}, \\ C = \displaystyle\bigcup_{t=1}^{n} C_{\varphi(t),t} \quad \cup \quad \bigcup_{t=1}^{n-1} \{(\delta_j^{F,t} = \delta_j^{I,t+1}) | j \in [1..p]\}. \end{cases}$$

Soient T une tâche que l'on souhaite faire réaliser par le robot R et (E_i, E_f) modélisant T. Soient alors $\mathcal{S}$ la séquence d'actions élémentaires permettant de réaliser T et $\mathcal{P} = (X, D, C)$ le CSP global permettant d'instancier $\mathcal{S}$ au sens de la définition 4.

Afin que $\mathcal{P}$ modélise correctement la tâche T, il convient d'ajouter à C un ensemble de contraintes d'égalité : $\forall \delta_j^{I,1} \in X$ (resp. $\delta_j^{F,n}$) modélisant un descripteur de R apparaissant dans E_i (resp. E_f), il faut ajouter à C la contrainte $(\delta_j^{I,1} = v)$ (resp. $(\delta_j^{F,n} = v)$) où v est la valeur de ce descripteur dans E_i (resp. E_f).

4.4 Exécution d'une séquence d'actions

Soit s une solution du CSP global, la propriété 1 nous permet d'établir que la restriction de s aux variables α_j^t $\forall (j,t) \in [1..q] \times [1..n]$ nous indique quelles tensions doivent être successivement appliquées aux actionneurs du robot pour réaliser T.

Propriété 1 (Solutions du CSP global)
Soient (E_i, E_f) le couple d'états initial et final caractérisant une tâche T que l'on souhaite exécuter et $\mathcal{S} = [s_1, \ldots, s_n]$ une séquence d'actions élémentaires permettant de réaliser T. Soit $\mathcal{P} = (X, D, C)$ le CSP global modélisant $\mathcal{S}$ et le couple (E_i, E_f). Si $\mathcal{P}$ admet des solutions, alors toute instanciation valide (solution) de $\mathcal{P}$ est un plan valide pour T.

5 Expérimentations préliminaires

Pour valider expérimentalement l'intérêt de notre approche, nous avons réalisé des expérimentations préliminaires sur les robots TWIG et PIRI respectivement représentés par les figures 3 et 4.

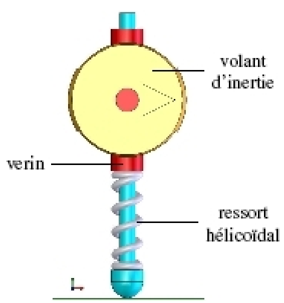

FIG. 3 – Le robot TWIG.

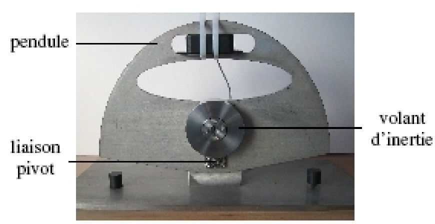

FIG. 4 – Le robot PIRI.

Dans un premier temps, nous avons poursuivi l'étude du robot TWIG initiée pour (Paulin *et al.*, 2006). Cette étude amont avait permis de valider l'utilisation de CONACQ pour modéliser automatiquement en contraintes les actions élémentaires de ce robot sauteur unijambiste modélisé sur simulateur. Le travail réalisé dans cette deuxième étude a consisté à combiner ces actions élémentaires (saut vertical, atterrissage vertical, saut horizontal, rester stable) à l'aide d'un planificateur de tâches capable de manipuler des réseaux de contraintes. Nous avons pour cela implémenté un planificateur STRIPS inspiré de GRAPHPLAN, à l'aide du résolveur de contraintes CHOCO,[1] auquel nous

[1] *http ://choco.sourceforge.net/* The CHOCO constraint programming system.

avons fait appel pour planifier des séquences d'actions pour TWIG. Cette série d'expérimentations a ainsi permis de valider l'intérêt de notre approche *contraintes* pour une planification de tâches élégante et performante.

Au travers de l'étude d'un Pendule Inversé stabilisé par une Roue d'Inertie (PIRI), nous avons cherché à montrer la capacité des contraintes à établir une loi de commande (robuste aux perturbations) de recherche d'équilibre pour un robot cadencé à 100Hz. Nous avons pour cela implémenté en langage CHOCO la loi de commande bas niveau[2] utilisée par les roboticiens et avons ensuite commandé PIRI à l'aide de ce réseau de contraintes. Cette étude expérimentale a mis en lumière la capacité de notre approche PPC à répondre aux impératifs du réel (fréquence élevée, perturbations extérieures) au moyen du cycle *perception / décision / action* sur lequel est basé notre système de supervision.

6 Conclusion et perspectives

Dans cet article, nous avons proposé un système de supervision qui utilise les réseaux de contraintes pour modéliser les actions élémentaires d'un robot humanoïde. Notre approche permet de modéliser de manière élégante et performante les habiletés physiques du robot sous la forme d'unités élémentaires de contrôle, qui sont ensuite combinées à l'aide d'outils de planification. L'utilisation des contraintes permet de fournir un *modèle* du robot et de son environnement, à partir duquel nous sommes capables d'établir des prévisions sur son état futur, puis de confronter ces prévisions à la réalité de l'exécution d'une action.

Par ailleurs, la réalité opérationnelle d'un robot autonome nécessite une réponse efficace et symbolique, que la programmation par contraintes est à même de fournir. Le système de supervision que nous proposons permet ainsi d'entrevoir une première génération de robots humanoïdes capables de raisonner minimalement sur leur comportement.

Les expérimentations préliminaires présentées dans la section 5 semblent démontrer la capacité de notre système de supervision à répondre aux impératifs d'expressivité et d'efficacité liés à la robotique humanoïde. Notre objectif est désormais de déployer et d'adapter ce système aux spécificités du robot humanoïde HOAP3 de FUJITSU (figure 5) récemment arrivé au Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier (LIRMM).

Fig. 5 - Le robot HOAP3.

Avec HOAP3, nous allons nous confronter à un cahier des charges d'envergure : 28 articulations motorisées, des descripteurs nombreux et de nature diverses (accéléromètres, télémètres, caméras stéréoscopiques, *etc.*) ou bien encore la nécessité de garan-

[2]La loi de commande utilisée est en effet la somme pondérée de la vitesse du volant d'inertie, l'angle du pendule par rapport à la verticale, et sa vitesse.

tir un fonctionnement sécurisé pour éviter chutes et casses mécaniques. En partenariat avec les roboticiens du laboratoire, nous souhaitons réaliser une intégration de notre approche cohérente et respectueuse des techniques de commande existantes, afin d'exploiter au mieux les habiletés physiques de HOAP3.

Remerciements

L'auteur tient à remercier Jean SALLANTIN qui a soutenu ce travail avec le plus vif intérêt. Certaines idées ont émergé suites à d'agréables et fructueuses discussions avec Eric BOURREAU, dont les remarques ont par la suite contribué à améliorer la lisibilité de cet article. Enfin l'auteur tient à remercier vivement Sébastien ANDARY pour les heures de travail passées à la programmation du robot PIRI.

Références

BENEDETTI M., LALLOUET A. & VAUTARD J. (2006). Reusing csp propagators for qcsps. In *Proceedings of the Annual ERCIM Workshop on Constraint Solving and Constraint Logic Programming (CSCLP-06)*, Lisbonne, Portugal.

BESSIERE C., COLETTA R., KORICHE F. & O'SULLIVAN B. (2005). A sat-based version space algorithm for acquiring constraint satisfaction problems. In *Proceedings of ECML'05*, p. 747–751, Porto, Portugal.

BESSIERE C. & HENTENRYCK P. V. (2003). To be or not to be...a global constraint. In L. 2833, Ed., *Proceedings of CP-2003*, p. 789–794, Springer Kinsale, Cork, Ireland.

COLLINS J. & LUCA C. D. (1993). Open-loop and closed-loop control of posture : A random-walk analysis of center-of-pressure trajectories. In *Experimental Brain Research*, p. 308–318.

GASPIN C., SCHIEX T. & ZYTNICKI M. (2005). Bound arc consistency for weighted csps. In *7th International CP-2005 Workshop on Preferences and Soft Constraints.*

HENAFF P. (2007). Genèse des mouvements rythmiques et réflexes inspirée de la biologie : Application à la locomotion et au contrôle d'équilibre des robots humanoïdes. In *Journées Nationales de la Robotique Humanoïde*, Montpellier, France.

PAULIN M. (2006). Supervision of robot tasks planning through constraint networks acquisition. In *CP-2006 Doctoral Programme*, p. 180–185, Nantes, France.

PAULIN M., BOURREAU E., DARTNELL C. & KRUT S. (2006). Modélisation et planification d'actions élémentaires robotiques par apprentissage de réseaux de contraintes. In *Proceedings of JFPC'2006*, p. 405–414, Nîmes, France.

RUSSELL S. & NORVIG P. (2003). *Artificial Intelligence - A Modern approach.* Second Edition. Prentice Hall.

VERGER G. & BESSIERE C. (2006). Blocksolve : a bottom-up approach for solving quantified csps. In *Proceedings of CP-2006*, p. 635–649, Nantes, France.

WALLACE M. (1996). Practical applications of constraint programming. *Constraints*, **1**(1/2), 139–168.

States and Time in Modal Action Logic

Camilla Schwind

LIF- CNRS, Luminy, Université de la Méditerranée
163 Avenue de Luminy, Case 901 13288 Marseille Cedex 9, France,
E-mail: schwind@lif.univ-mrs.fr

Abstract. We present a multi-modal action logic with first-order modalities, which contain terms which can be unified with the terms inside the subsequent formulas and which can be quantified. This makes it possible to handle simultaneously time and states. We discuss applications of this language to action theory where it is possible to express many temporal aspects of actions, as for example, beginning, end, time points, delayed preconditions and results, duration and many others.

1 Introduction

Most action theories consider actions being specified by their preconditions and their results. The temporal structure of an action system is then defined by the sequence of actions that occur. A world is conceived as a graph of situations where every link from one node to the next node is considered as an action transition. This yields also a temporal structure of the action space, namely sequences of actions can be considered defining sequences of world states. The action occurs instantantly at one moment and its results are true at the "next" moment.

However, the temporal structure of actions can be much more complex and complicated.

– Actions may have a duration.
– The results may be true before the action is completed or after it is finished.
– Actions may have preconditions which have to have been true during some interval preceding the action occurrence.

In order to represent complex temporal structures, underlying actions' occurrences, we have developed an action logic which allows to handle both states and time simultanuously.

We want to be able to express, for instance, that action a occurs at moment t if conditions $p_1, \ldots p_n$ have been true during the intervals $i_1, \ldots i_k$ all preceding t.

Here we present an approach where it is possible to describe actions in a complex temporal environment. In reality, actions have sometimes a beginning time, a duration, preconditions which may also have temporal aspects. Action results may become true only instants after the end of the action performance. For an example, consider the action of calling an elevator, taking place at instant t_1. Depending on the actual situation this action may cause the elevator to move only many instants later, to stop still later, an so on. The action of pressing the button of a traffic light, in order to get green light to cross the street may result in a switch immediately or after some seconds and in another switch after some more minutes.

In order to represent such issues, we define a modal action logic, $\mathcal{D}al$, where the modalities are terms containing variables which can be quantified. The same variables can occur inside the modalities as well as in the formulas after the modalities, allowing for unification between action term components and logical terms. This language makes it possible to express reasoning on states and the action terms allow to express temporal aspects of the actions.

A similar logic, called term modal logic, has been defined in [1]. In this work, any term of the first-order language can be a modal operator. This makes it possible to quantify over variables naming the accessability relation. In our appraoch, we quanantify over terms which index the accessability relation. It is possible to simulate term modal logic in our logic $\mathcal{D}al$ by having one single action a of arity 1 and by using as action operators any operator $a(t)$ for any term t of the language. The simulation of $\mathcal{D}al$ by term modal logic is not so straightforward, because in our logic action names cannot occur within the underlying first-order language (they are different from all predicate and function symbols of underlying the first-order logic). Another difference of our logic with term modal logic is that our logic includes equality.

Another related formalism is hybrid logic where it is possible to quantify over state variables naming worlds. In hybrid logic however these state variables belong to the object language.

2 The first-order modal action logic $\mathcal{D}al$

The language of first-order action logic is an extension of the language of classical predicate logic, $\mathcal{L}_0$. $\mathcal{L}_0$ consists of a set of variables $x, y, x_1, y_1, \ldots$, a set $\boldsymbol{F}$ of function symbols F, where $|F| \in \omega$ is the arity of F, a set $\boldsymbol{P}$ of predicate symbols P, including $\top$ and $\bot$, where $|P| \in \omega$ is the arity of P[1], an equality symbol $=$, the logical symbols $\neg$, $\wedge$,$\forall$. Terms and formulas are defined as usual and so are $\exists$ and $\vee$. We denote by V_t the set of all terms of $\mathcal{L}_0$. Free occurrence of a variable x in a formula ϕ is defined as usual.

$\mathcal{D}al$ is a multi-modal logic where each modality is a term of the form $a(t_1, \ldots t_n)$ for some *action* name a. We call these terms actions terms. They only occur within modalities. In addition, $\mathcal{D}al$ contains the **S4**-modal operator $\Box$ which will be used to express that a sentence is true in every world of a Kripke model.

Action terms The language for action operators consists of

– a set $\boldsymbol{A}$ of *action symbols* $a_1, a_2, \ldots$ where $|a| \in \omega$ is the arity of a and such that $\boldsymbol{A} \cap \boldsymbol{P} = \emptyset$

Action terms are built from action symbols and terms of $\mathcal{L}_0$.

[1] There are two predicate symbols of arity 0 which can be identified with $\top$ and $\bot$. Sometimes $\top$ and $\bot$ are also considered as empty conjunction and disjunction, hence as logical symbols.

– if a is an action symbol of arity n and $t_1, \dots t_n$ are terms of $\mathcal{L}_0$, then $a(t_1, \dots t_n)$ is an action term.

An action term is called *grounded* if no variable occurs free in it. The set of grounded action terms is denoted by $\boldsymbol{A}_t$.

Action operators If a and $a_1, a_2, \dots a_n$ are action terms, then

– $[a]$ is an action operator
– $[a_1; a_2; \dots a_n]$ is an action operator
– For $n = 0$, the corresponding action operator is noted $[\varepsilon]$

$[\varepsilon]$ is the "empty" action operator. We use it in order to define the initial state of a system.

Modal operator $\Box$ is the standard modal operator (**S4**)

We use $\Box$ for describing general laws (constraints) which hold in all states of a system.

An action operator is called *grounded* if all the action terms occurring in it are grounded.

Example: $[a]$, $[a(c_1, c_2, c_3)]$ are grounded, $[a(x, c_2, y)]$ is not grounded.

$\mathcal{D}al$ formulas

– Every first-order formula (of $\mathcal{L}_0$) is a $\mathcal{D}al$ formula.
– If ϕ is a $\mathcal{D}al$ formula and $[A]$ is an action operator, then $[A]\phi$ is a $\mathcal{D}al$ formula.
– If ϕ is a $\mathcal{D}al$ formula and $\Box$ is the modal operator, then $\Box\phi$ is a $\mathcal{D}al$ formula.
– If ϕ is a $\mathcal{D}al$ formula and x is a variable, then $\forall x\phi$ is a $\mathcal{D}al$ formula.

Instantiation If ϕ is a formula and t is a term, then ϕ^x_t is the formula obtained from ϕ by replacing every free occurrence of x by t. If t is the name of an element of a set $\mathcal{O}$ then ϕ^x_t is called $\mathcal{O}$-instance of ϕ.

Example:

$$[a(x, c)](\neg\phi(c, x) \vee \psi(x))^x_{c_1} = [a(c_1, c)](\neg\phi(c, c_1) \vee \psi(c_1))$$
$$[a_1; a_2; a_3(c, y)]P(c, y)^y_{c_3} = [a_1; a_2; a_3(c, c_3)]P(c, c_3)$$

A formula is called *grounded* if there is no variable occurring free in it. The notion of free occurrence of a variable within a formula is extended to $\mathcal{D}al$ -formulas as follows: If a variable occurs free within an action term A then it occurs free in $[A]\phi$.

2.1 Semantical Characterization of $\mathcal{D}al$

$\mathcal{D}al$ formulas are semantically characterized by Kripe structures, i.e. sets of worlds where each world is a classical structure. Since action operators are indexed by terms of the language, we have a different action operator $[a(t_1, \dots t_n)]$ for any grounded tuple of terms $t_1, \dots t_n$ of the first-order language $\mathcal{L}_0$. And consequently, we have a different transition relation for each of these action operators. The semantics of $\mathcal{D}al$ is defined as follows:

A $\mathcal{D}al$ structure is a Kripke-type structure, such that the transition relation between worlds depends on grounded action terms.

A $\mathcal{D}al$ structure is a tuple $\mathcal{M} = (\mathcal{W}, \{\mathcal{S}_w : w \in \mathcal{W}\}, \mathcal{A}, \mathbf{R}, \tau)$, where

- $\mathcal{W}$ is a set of *worlds*
- for every $w \in \mathcal{W}$, $\mathcal{S}_w = (\mathcal{O}, \mathcal{F}_w, \mathcal{P}_w)$ is a classical structure, where $\mathcal{O}$ is the set of individual objects (the same set in all worlds), $\mathcal{F}_w$ is a set of functions over $\mathcal{O}$ and $\mathcal{P}_w$ is a set of predicates over $\mathcal{O}$.
- $\mathcal{A}$ is a set of *action functions*, for $f \in \mathcal{A}$, $f : \mathcal{W} \times \underbrace{\mathcal{O} \times \ldots \times \mathcal{O}}_{n} \longrightarrow 2^{\mathcal{W}}, n \in \omega$.

  Action functions will characterize the action operators (every action symbol of arity n in $\boldsymbol{A}$ will be associeted with an action function of arity $n+1$).
- $\mathbf{R} \subseteq \mathcal{W} \times \mathcal{W}$ is a binary accessability relation on $\mathcal{W}$, which will characterize the modal operater $\Box$. We will write $R(w) = \{w' : (w, w') \in R\}$.
- τ is a valuation, $\tau = (\tau_0, \tau_1, \tau_2, \tau_3)$, where τ_0 is a function assigning objects from $\mathcal{O}$ to terms. In order to speak about objects from $\mathcal{O}$, we introduce into the language, for every $o \in \mathcal{O}$, an o-place function symbol (denoted equally o, for simplicity).
  τ_1 is a function assigning, for every world $w \in \mathcal{W}$, functions (from $\mathcal{F}_w$) to function symbols (from $\boldsymbol{F}$), of the same arity,
  $\tau_1 : \mathcal{W} \times \boldsymbol{\mathcal{F}} \longrightarrow \mathcal{F}$ such that $|\tau_1(w, F)| = |F|$.
  τ_2 is a function assigning, for every world $w \in \mathcal{W}$, predicates to predicate symbols of the same arities,
  $\tau_2 : \mathcal{W} \times \boldsymbol{\mathcal{P}} \longrightarrow \mathcal{P}$, such that $|\tau_2(w, P)| = |P|$.
  τ_3 is a function assigning action functions to action symbols,
  $\tau_3 : \boldsymbol{A} \longrightarrow \mathcal{A}$, such that $|\tau_3(a)| = |a| + 1$
- $\tau_3(a)(w, \tau_0(t_1), \ldots, \tau_0(t_n)) \subseteq R(w)$. If a world can be reached from w by the execution of action $a(t_1, t_2, \ldots, t_n)$ then it is accessible (via the relation R).

τ_0, τ_1, τ_2 and τ_3 define the valuation τ as follows:

- If $F(t_1, t_2, \ldots, t_m)$ is a term then
  $\tau_0(w, F(t_1, t_2, \ldots, t_m)) = \tau_1(w, F)(\tau_0(w, t_1), \tau_0(w, t_2), \ldots, \tau_0(w, t_m))$.
- if P is an n-ary predicate symbol and $t_1, t_2, \ldots, t_n$ are free object variables then
  $\tau(w, Pt_1, t_2, \ldots, t_n) = \top$ iff $(\tau_0(t_1), \ldots \tau_0(t_n)) \in \tau_2(w, P)$
- $\tau(w, t_1 = t_2) = \top$ iff $\tau_0(w, t_1) = \tau_0(w, t_2)$
- $\tau(w, \neg\phi) = \top$ iff $\tau(w, \phi) = \bot$
- $\tau(w, \phi \wedge \psi) = \top$ iff $\tau(w, \phi) = \tau(w, \psi) = \top$
- $\tau(w, \forall x \phi) = \top$ iff for every $o \in \mathcal{O}$[2] $\tau(w, \phi^x_o) = \top$
- $\tau(w, [a(t_1, t_2, \ldots, t_n)]\phi) = \top$ iff for every $w' \in \tau_3(a)(w, \tau_0(t_1), \ldots, \tau_0(t_n))$, $\tau(w', \phi) = \top$
- $\tau(w, \Box\phi) = \top$ iff for every $w' \in R(w)$, $\tau(w', \phi) = \top$

[2] Note that we added every $o \in \mathcal{O}$ as a new term (the name of o) to the language.

Let ϕ be a formula and $x_1, x_2, \ldots, x_n$ be the free object variables occurring in ϕ. Then $\tau(w, \phi) = t$ iff for every tuple $t_1, t_2, \ldots, t_n$ of ground terms,

$$\tau(w, \phi^{x_1, x_2, \ldots, x_n}_{t_1, t_2, \ldots, t_n}) = t$$

A formula ϕ is called valid in state $w \in \mathcal{W}$ of a $\mathcal{D}al$ -structure $\mathcal{M}$ iff $\tau(w, \phi) = \top$. This is denoted by $\mathcal{M}, w \models \phi$. We also say then that ϕ is satisfiable. A formula ϕ is called valid in a $\mathcal{D}al$ - structure $\mathcal{M}$ with the set of states $\mathcal{W}$, iff ϕ is valid in every $w \in \mathcal{W}$. We denote that by $\mathcal{M} \models \phi$. A formula ϕ is called $\mathcal{D}al$ - valid iff ϕ is valid in every $\mathcal{D}al$ - structure. This is denoted by $\models_{\mathcal{D}al} \phi$. We suppress the index $\mathcal{D}al$, whenever it is clear from the context, in which system we are.

Remark 1 $[a]\bot$ *is satisfiable and we have* $\tau(w, [a]\bot) = \top$ *iff* $\tau_3(a)(w, \tau_0(t_1), \ldots, \tau_0(t_n)) = \emptyset$

2.2 Axioms and inference rules of $\mathcal{D}al$

In addition to the axioms and inference rules of classical first - order logic and those of the system K, which rule all action operators including $[\varepsilon]$, and those of the system and $\mathcal{S}_4$, which rule the operator $\Box$, we have the following axioms and inference rules, (where $[A]$, $[A_1]$ and $[A_2]$ are arbitrary action operators):

$[A1]$ $[A_1; A_2]\alpha \leftrightarrow [A_1][A_2]\alpha$
$[A2]$ $\Box\alpha \rightarrow [A]\alpha$
$[A3]$ $[\varepsilon]\alpha \rightarrow \alpha$
$[A4]$ $\forall x\alpha \rightarrow \alpha^x_c$ for any term c of $\mathcal{L}$,
$[A5]$ $\forall x[X]\alpha \leftrightarrow [X]\forall x\alpha$ for any modal operator X, with no occurrence of x

$[R1]$ From α infer $\Box\alpha$
$[R2]$ From $\alpha \rightarrow \beta$ infer $\alpha \rightarrow \forall x\beta$ provided x has no free occurrence in α

$\vdash_{\mathcal{D}al}$ is defined as usual, such that $\vdash_{\mathcal{D}al} \phi$ for any instance ϕ of one of the axioms; and $\vdash_{\mathcal{D}al} \psi$, whenever ψ can be inferred from ϕ, for any ϕ, such that $\vdash_{\mathcal{D}al} \phi$ by use of one of the inference rules. Again, we suppress the index $\mathcal{D}al$, whenever it is clear from the context, in which system we are.

$A2$ says that a formula which is true "in all worlds of a model" will still be true after the occurrence of actions $[A]$. $A3$ asserts the "emptyness" of the action operator $[\varepsilon]$: if formula α is true after the occurrence of ε it is already true. $A5$ is the well-known Barcan formula (and its contrapositive). We need $A5$ because we choose the same set of objects for every world. Objects cannot disappear neither new objects can appear.

2.3 Soundness, Completeness, Decidability

The $\mathcal{D}al$ -logic is sound and complete:

Theorem 1 $\vdash_{\mathcal{D}al} \phi$ *if and only if* ϕ *is* $\mathcal{D}al$ *- valid (*$\models_{\mathcal{D}al} \phi$*)*

The soundness proof is easy and the completeness proof goes along the lines of completeness proofs for modal logics by construction of a canonical model. The proof, which can be found in the full paper [13], bears several modifications according to the specific language which allows to quantify over terms occurring within modal operators.

$\mathcal{D}al$ is a first order language and therefore undecidable in the general case. But for action logics, we will make use of a decidable subset of $\mathcal{D}al$.

$\mathcal{D}al$ is very close to term modal logic introduced by [1]. Term modal logic allows terms in general as modalities, whereas our action logic only admits action terms. Moreover $\mathcal{D}al$ contains the **S4** modal operator $\Box$ which is not part of term modal logic.

Decidable subsets of first-order logic can "yield" decidable subsets of $\mathcal{D}al$: the subset of $\mathcal{D}al$ without function symbols and existential quantifiers can be shown to be decidable by the finite model property. We omit the proof for space limit.

3 Temporal Action Theories

Here we present one application of the formalism introduced above. This application is related to the formal description of actions by modal logic. Frequenly in action theories "time points" are identified with states (or worlds of a Kripke model). This yields a discrete conception of systems where actions can occur in a state of the system producing as a result a "next" state. But we argue that in real world systems both notions co-exist: time and state. Frequenly, we conceive an action as occurring instantly and producing its results at some "next" instant. But at the same time, we have an underlying idea of *time*, measured eventually by a clock even when we do not systematically need to refer to this time axis. For some scenarios it is necessary to take into account different (and various) temporal a spects of actions, simply because there may be termporally delayed preconditions or results of actions.

We can modelize these temporal aspects of actions using $\mathcal{D}al$. The modal logic allows to define action operators as modalities much like in [3, 12]. The first order logic is used to formulate actions at a more general level. Here, we show an example where in addition to the relative representation of time by the modal operators, it is possible to add assertions about time points or intervals.

Subsequently, we use a restricted subset $\mathcal{D}al$, whith no positive occurrence of existential quantifiers and no negative ocurrence of universal quantifiers.

We presuppose a time axis, $\mathcal{T}$, linearly ordered (dense or continuous or discrete). Given a $\mathcal{D}al$ -structure, we will define a transitive relation on the set of states, $\mathcal{W}$, which will be related to the order on $\mathcal{T}$.

Definition 1 *Let* $\mathcal{M} = (\mathcal{W}, \{\mathcal{S}_w : w \in \mathcal{W}\}, \mathcal{A}, \mathbf{R}, (\tau_0, \tau_1, \tau_2, \tau_3))$ *, be a* $\mathcal{D}al$ *-model. Then* $w \prec_0 w'$ *iff* $\exists a \in \mathcal{A}$ *of arity* n *and there are terms* $t_1, \dots, t_n$*, such that* $w' \in \tau_3(a)(w, \tau_0(t_1), \dots, \tau_0(t_n))$*. Let be* $\preceq$ *the reflexive and transitive closure of* $\prec_0$*.*

Intuitively, this means that $w \prec w'$ if we can possibly "reach" w' from w by performing actions $a_1, a_2, \dots, a_n$. Obviously, $\preceq$ is transitive and reflexive. Since we want to "link" worlds of $\mathcal{W}$ to time points in $\mathcal{T}$, which is ordered, $\prec$ must also be antisymmetric. The temporal entrenchment of the states is defined by a homomorphism $time : \mathcal{W} \longrightarrow \mathcal{T}$ from $\mathcal{W}$ into $\mathcal{T}$, where $w \preceq w'$ implies $time(w) \leq time(w')$. Using this construction, action operators can be defined admitting complex temporal structures, including beginning and ending instants and a duration, which can be 0, when the result is immediate. The preconditions and results of actions can be defined to occur at freely determinable time instants before or after the instant when the action occurs. When an action a occurs in the state w, $time(w)$ gives us the time point at which a occurs. If the duration of the action is Δ, the time point of the resulting state w' is $time(w') = time(w) + \Delta$.

In this particular framework, we define

- *Action terms* as binary action predicates $a(t, d, \overrightarrow{x})$, where t denotes the instant on which a occurs and d denotes the duration of a, i.e. the interval on $\mathcal{T}$ after which the results of a will hold. , $\overrightarrow{x}$ is the sequence of other variables denoting the other entities or objects involved in the action occurrence.
  To give an example, let $\mathcal{T} = \{1, \ldots, 24\}$ be discrete and finite, denoting the hours during one day. Then action $move(t, 3, TGV, Marseille, Paris)$ is the action "train TGV goes from Marseille to Paris, the duration being 3 hours".
- *Action axioms*. An action axiom is characterized by a precondition $\pi(t, \overrightarrow{x})$ and a result $\rho(t + d, \overrightarrow{x})$, where π is a $\mathcal{D}al$ formula describing all preconditions of action a and ρ is a conjunction of litterals describing the results of a.
  To continue the previous example, the action execution axiom of the move-action is $at(t, x, y) \rightarrow [move(t, d, x, y, z)]at(t + d, x, z)$ (and can be instantiated to $at(6, TGV, Marseille) \rightarrow [move(6, 3, TGV, Marseille, Paris)]at(9, TGV, Paris)$), which means: if x is at y at instant t, then, after moving from y to z, x is at z at instant $t + d$.

The general form of an action law is

$$(*)\ \Box(\pi(t_1, \overrightarrow{x_1}) \rightarrow [a(t, d, \overrightarrow{x_2})]\rho(t_2, \overrightarrow{x_3})), where\ \overrightarrow{x_1} \cup \overrightarrow{x_2} \subseteq \overrightarrow{x_3}$$

Note that it is then possible to derive one action law

$$\Box(\pi(t_1, \overrightarrow{x_1}) \rightarrow [a(t, d, \overrightarrow{x_2})]l(t_2, \overrightarrow{x_3}))$$

from axiom (*) for each of the litterals occurring within ρ.

We define action systems much as in [3, 4] or [2].

An *action system* is a tuple $(\Pi, Frame, \mathcal{C})$ where Π is a set of action laws, and of *causal laws*, $\mathcal{C}$ is a set of constraints (or general laws) and $Frame$ provides a classification of the atoms of the language as frame fluents and non-frame fluents. As in [3, 4] we use a completion construction in order to solve the frame problem.

Causal laws have the general form $\Box(\alpha \wedge [a]\gamma \rightarrow [a]f)$ or $\Box(\beta \wedge [a]\delta_j \rightarrow [a]\neg f)$, where f is an atom (a fluent) and α, β, γ and δ are first-order formulas (not containing modalities).

$Frame$ is a set of pairs (f, a), where $a \in \boldsymbol{A}$ is an action symbol and f is an atom. Our action system relies on solutions to the frame problem similar to those described in [3, 4]. In this paper a completion construction is defined which, given a domain description, introduces frame axioms for all frame fluents in the style of the successor state axioms in situation calculus [10]. This completion construction applies only to action laws and causal laws and not to the constraints.

Let be Π contain action and causal laws which both have the general form

$$\Box(\alpha_i \wedge [a]\gamma_i \rightarrow [a]f) \qquad \Box(\beta_j \wedge [a]\delta_j \rightarrow [a]\neg f),$$

where $\alpha_i, \beta_j, \gamma_i, \delta_j$ are arbitrary (non-temporal) formulas and some of the conjuncts in the antecedents may be missing (in the case of action laws). This is an enumeration of the action laws for action a.

The completion of Π is the set of formulas $Comp(\Pi)$ containing, for all actions a and fluents f such that $(f, a) \in Frame$, the following axioms:

$$\Box(\langle a\rangle\top \rightarrow ([a]f \leftrightarrow \bigvee_i(\alpha_i \wedge [a]\gamma_i) \vee (f \wedge \neg[a]\neg f))) \tag{1}$$

$$\Box(\langle a\rangle\top \rightarrow ([a]\neg f \leftrightarrow \bigvee_j(\beta_j \wedge [a]\delta_j) \vee (\neg f \wedge \neg[a]f))) \tag{2}$$

Notice that, for each action a and fluent f which is nonframe with respect to a, i.e. $(f, a) \notin Frame$, axioms (1) and (2) above are not added in $Comp(\Pi)$. As in [10], these laws express that a fluent f (or its negation $\neg f$) holds either as a consequence of some action a or some causal law, or by persistency, since f (or $\neg f$) held in the state before the ocurrence of a and $\neg f$ (or f) is not a result of a. The occurrences of $\langle a\rangle\top$ assure that there is a succeeding state after action a occurred (formula $\langle a\rangle\top$ is true if there is a resulting state after occurrence of action a).

From the two axioms above we can derive the following axioms, which are similar, in their structure to Reiter's successor state axioms [10]:

$$\Box(\langle a\rangle\top \rightarrow ([a]f \leftrightarrow (\bigvee_i(\alpha_i \wedge [a]\gamma_i)) \vee (f \wedge \bigwedge_j(\neg\beta_j \vee \neg[a]\delta_j))))$$
$$\Box(\langle a\rangle\top \rightarrow ([a]\neg f \leftrightarrow (\bigvee_j(\beta_j \wedge [a]\delta_j)) \vee (\neg f \wedge \bigwedge_i(\neg\alpha_i \vee \neg[a]\gamma_i))))$$

The construction above is similar to the one that we have introduced in [3], though there are some differences for the fact that in [3], we have adopted a different formalization of causal laws by using the next operator. Also, in the present paper, causal laws are more general than in [3], as they refer (in their antecedent) to the values of fluents in the current state as well as in the next state.

4 Example

The following example is due to Lewis [6] and has been discussed by Halpern and Pearl in [5] in the framework of a theory of causation. Interestingly, this example defines actions with a complex temporal structure.

Billy and Suzanne throw rocks at a bottle. Suzanne throws first and her rock arrives first. The bottle shatters. When Billy's rock gets to where the bottle used to be, there is *nothing there but flying shards of glass. Without Suzanne's throw, the impact of Billy's rock on the intact bottle would have been one of the final steps in the causal chain from Billy's throw to the shattering of the bottle. But, thanks to Suzanne's preempting throw, that impact never happens.*

In our formulation, we focalize on the temporal structure of the throw action. We consider that the action occurs along a continuous (or dense) time axis, $[0, \infty[$. We define one action term for "throw", T, and two predicates H for "hits" and BB for "the bottle is broken". The action term $T(t, d, p)$ means that "person p throws a stone to a bottle at instant t and the result of the action (the stone hits its target) occurs at instant $t + d$". The formula $H(t, p)$ means that "the stone thrown by person p hits the bottle at instant t and formula $BB(t)$ means that the bottle is broken at instant t. The intended result of the action is to hit the bottle, but this result can only be achieved if the bottle is still at the intended place and nothing else has been happened to it, namely if it is not broken in the meantime. In this example it is not enough to have the precondition that the bottle is there and not broken at the instant of throwing, but it must be non-broken at the moment when the action is to be completed, just before it is to be hit. Therefore the action law for "throw" has a precondition which must hold after the instant when the action occurs.

Example 1 *The following set of laws represents the framework of this story:*

$$
\begin{array}{l}
(1)\ \Box(\neg BB(t+d) \rightarrow [T(t,d,p)]H(t+d,p)) \\
(2)\ \Box(H(t,p) \rightarrow BB(t+d_1)) \\
(3)\ \Box(BB(t) \rightarrow \forall t'(t < t' \rightarrow BB(t'))) \\
(4)\ [\varepsilon]\neg BB(0)
\end{array}
$$

(1) is the action law for successful execution of the throw action, (2) describes the impact of hitting the bottle (d_1 is infinitesimally small) and the general law (3) says that a broken bottle remains broken "forever" [3].

Several scenarios can happen within this framework. Here we discuss the scenario where Suzanne throws at instant 0 and Billy throws some instant later[4].

$$
\begin{array}{l}
(5) < T(0, d_s, suzy) > \top \\
(6) < T(t_1, d_b, billy) > \top
\end{array}
$$

Three cases can be distinguished:

1. The moment when the bottle can be hit (and broken) after Suzanne's throw ($d_s + d_1$) occurs **before** Billy's stone could possibly hit the bottle $t_1 + d_b$.

[3] In this example we focus on the temporal relations between the different instants of throwing (by Suzanne and by Billy), so we neglected other preconditions, as for example having a stone, heavy enough, but not too heavy, having members enabling the person to throw, seeing the object to aim, etc. The throw action defined here is highly abstracted for the purpose of our temporal action theory.

[4] In order to express that action $a$occurs, we write $[a]\top$, which simply means that action a occurs (even when nothing can be said about its results). It is always possible to throw a stone at a bottle, even if the intended result of hitting cannot be achieved.

(7) $d_s + d_1 < t_1 + d_b$
(8) $\Box(BB(d_s + d_1) \to BB(t_1 + d_b))$ from (3) and (7)
(9) $\neg BB(d_s)$ by persistency from (4)[5]
(10) $[T(0, d_s, suzy)]H(d_s, suzy)$ from (1) and (9)
(11) $[T(0, d_s, suzy)]BB(d_s + d_1)$ from (2), (10), K for the action modality and (A2)
(12) $[T(0, d_s, suzy)]BB(t_1 + d_b)$ from (11), (8), K and (A2)

In this scenario, the law $\neg BB(t_1+d_b) \to [T(t_1, d_b, billy)]H(t_1+d_b, billy)$ cannot be used to derive $T(t_1, d_b, billy)]H(t_1 + d_b, billy)$ because $BB(t_1 + d_b)$ holds after Suzanne's throw (12). Billy's stone cannot hit the bottle, because it is already broken when his stone could hit it and we have just $[T(t_1, d_b, billy)]\top$ ((6), Billy has thrown).

2. Billy's stone hits the bottle, which breaks, **before** Suzanne's stone could possibly hit the bottle.

(13) $t_1 + d_b + d_1 < d_s$
(14) $\Box(BB(t_1 + d_b + d_1) \to BB(d_s))$ from (3) and (13)
(15) $\neg BB(t_1 + d_b)$ by persistency from (4), see (9)
(16) $[T(t_1, d_b, billy)]H(t_1 + d_b, billy)$ from (1) and (15)
(17) $[T(t_1, d_b, billy)]BB(t_1 + d_b + d_1)$ from (2), (16), K and (A2)
(18) $[T(t_1, d_b, billy)]BB(d_s)$ from (14), (17), K and (A2)

Here, Suzanne's stone, which could hit the bottle at instant d_s, will not hit it since we have $BB(d_s)$ and therefore the precondition $\neg BB(d_s)$ is not more true. The law $\Box(\neg BB(d_s) \to [T(0, d_s, suzy)]H(d_s, suzy))$ cannot be used to derive $[T(0, d_s, suzy)]H(d_s, suzy)$ because $BB(t_1 + d_b + d_1)$ holds after Billy's throw (17). All we have is $[T(0, d_s, suzy)]\top$ (Suzanne throws).

3. Suzanne's and Billy's stone hit the bottle precisely at the same moment.

(19) $t_1 + d_b = d_s$
(20) $\neg BB(t_1 + d_b) \wedge \neg BB(d_s)$ by persistency from (4), see (9)
(21) $[T(0, d_s, suzy)]H(d_s, suzy)$ like (10)
(22) $[T(t_1, d_b, billy)]H(t_1 + d_b, billy)$ as (16)
(23) $[T(0, d_s, suzy)]BB(d_s + d_1)$ from (21)
(24) $[T(t_1, d_b, billy)]BB(t_1 + d_s b + d_1)$ from (22))

In this case, both stones hit the bottle which breaks as a result of Suzanne's throw and Billy's throw.

5 Conclusion and Related Work

Modal logic approaches to action theories define a space of states but cannot handle time, neither explicitly not implicitly [2, 7]. In situation calculus [11, 8] reasoning about time was not foreseen, properties change discretely and actions do not have durations. Remember that in situation calculus, there is a starting state, s_0 and for any action a

and state s, $do(a, s)$ is a resulting state of s. One can consider that the set of states is given by $\{s : \exists a_1 \dots a_n s = do(a_n, do(a_{n-1}, \dots do(a_1, s_0)))\}$. Hence the temporal structure of situation calculus is discrete and branching and does not allow for actions of different duration neither for preconditions or results which become true during the action execution or later after the action is ended.

Javier Pinto has extended situation calculus in order to integrate time [9]. He conserves the framework of situation calculus and introduces a notion of time. Intuitively, every situation s has a starting time and an ending time, where $end(s, a) = start(do(a, s))$ meaning that situation s ends when the succeeding situation $do(a, s)$ is reached. The end of the situation s is the same time point as the beginning of the next situation resulting from the occurrence of action a in s. The obvious asymmetry of the $start$ and end functions is due to the fact that the situation space has the form of a tree whose root is the beginning state s_0. Thus, every state has a unique preceding state but eventually more that one succeeding state.

Paolo Tereziani proposes in [14] a system that can handle temporal constraints between events and temporal constraints between instants of events.

In this present article, we have introduced a new modal logic formalism which can handle simultaneously states and time. We did not address here the problem of the persistency of facts over time (or over the execution of actions), because we wanted to focus on the modal temporal formalism. We have adopted a solution similar to the one presented in [12], i.e. "weak" frame laws are nonmonotonically added to the theory. But this solution is a bit more complicated in the case of our first-order action logic presented in this paper, because we need to restrict ourselves to a decidable subset of $\mathcal{D}al$.

Concerning the implementation, we use a labelled analytic tableaux approach including an abductive mechanism for the weak persistency laws, which will be described in more detail in a following paper.

Further work is carried out in two directions:

1. Concerning the logical formalism we want to make a more complete study of the undecidability and complexity issues. We want to study decidability for subclasses corresponding to well known first-order decidability classes.
2. Concerning applications, we will apply this formalism to planning problems where a hybrid approach (states and time) can be very powerful. The idea is to infer temporal constraints from a $\mathcal{D}al$ specification in order to create a plan for a problem. This will yield a hybrid approach to planning and scheduling

Acknowledgement: Many thanks to Jelle Gerbrandy for his critical remarks to a former version of this paper.

References

1. Melvin Fitting, Lars Thalmann, and Andrei Voronkov. Term-modal logics. *Studia Logica*, 69(1):133–169, 2001.
2. Laura Giordano, Alberto Martelli, and Camilla Schwind. Ramification and causality in a modal action logic. *Journal of Logic and Computation*, 10(5):625–662, 2000.

3. Laura Giordano, Alberto Martelli, and Camilla Schwind. Reasoning about actions in dynamic linear time temporal logic. In *Proceedings of the 3rd International Conference on Formal and Applied Practical Reasoning*, London, United Kingdom, 2000. Imperial College.
4. Laura Giordano, Alberto Martelli, and Camilla Schwind. Specifying and verifying interaction protocols in a temporal action logic. *Journal of Applied Logic*, 5(2):214–234, 2007.
5. Joseph Halpern and Judea Pearl. Causes and explanations: A structural model approach, part i. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence*, pages 27–34, Seattle, USA, 2001. Morgan Kaufmann.
6. David Lewis. Causation as influence. *The Journal of Philosophy*, 97:181–197, 2000.
7. Andreas Herzig Marcos A. Castilho, Olivier Gasquet. Formalizing action and change in modal logic i: the frame problem. *Journal of Logic and Computation*, 9(5):701–735, 1999.
8. John McCarthy. Actions and other events in situation calculus. In Deborah McGuinness Dieter Fensel, Fausto Giunchiglia and Mary-Anne Williams, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the 8th International Conference*, pages 615–626, 2002.
9. Pinto. Occurrences and narratives as constraints in the branching structure of the situation calculus. *JLC: Journal of Logic and Computation*, 8, 1998.
10. Ray Reiter. The frame problem in the situation calculus: A simple solution (sometimes) and a completeness result for goal regression. In *Artificial Intelligence and Mathematical Theory of Computation, Papers in Honor of John McCarthy*, pages 359–380. Academic press, 1991.
11. Raymond Reiter. *Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems*. MIT Press, 2001.
12. Camilla Schwind. A logic based framework for action theories. In J. Ginzburg, Z. Khasidashvili, C. Vogel, J.-J. Lévy, and E. Vallduví, editors, *Language, Logic and Computation*, pages 275–291, Stanford, USA, 1998. CSLI publication.
13. Camilla Schwind. A first-order temporal logic for actions. *The Computing Research Repository (CoRR)*, (http://arxiv.org/abs/0705.1999v1), 2007.
14. Paolo Terenziani. Temporal reasoning with classes and instances of events. In *9th International Symposium on Temporal Representation and Reasoning, TIME'2002*, pages 100–107. IEEE Computer Society, 2002.

Raisonner avec une politique d'échange d'informations incomplète

Laurence Cholvy[1] and Stéphanie Roussel[1,2]

[1] ONERA Centre de Toulouse
2 avenue Edouard Belin
31055 Toulouse, France
[2] SUPAERO
10 avenue Edouard Belin,
31055 Toulouse, France

Résumé : Dans ce papier, on étudie les politiques d'échange d'informations qui peuvent être présentes au sein de systèmes multi-agents pour réguler les échanges d'informations entre agents. Plus précisément, on se préoccupe de deux propriétés des politiques d'échange d'informations, à savoir la cohérence et la complétude. Après avoir défini les notions de cohérence et de complétude pour de telles politiques, on propose une méthode pour raisonner avec des politiques incomplètes.

Mots-clés : complétude, politique d'échange d'informations, système multi-agents

1 Introduction

Les systèmes multi-agents représentent un cadre intéressant pour la modélisation de systèmes dans lesquels des entités (atomiques ou complexes) coopèrent de façon à remplir une tâche commune ou atteindre un but commun. Pour que la coopération soit efficace, il faut que les entités, que l'on appellera agents, échangent des informations, en particulier pour avoir une vue plus générale de leur environnement et une meilleure compréhension de la situation actuelle.

Dans certains systèmes, les échanges d'informations sont complètement libres et les agents peuvent transmettre n'importe quelle information à n'importe qui. Au contraire, dans beaucoup d'autres systèmes, les échanges d'informations sont réglementés par une politique, en particulier lorsqu'il s'agit de satisfaire des contraintes de sécurité, telle que la confidentialité, ou des contraintes d'efficacité (diffusion ou communication peer-to-peer d'informations pertinentes). Les "Systèmes de Systèmes" (appelés ainsi dans le domaine de la défense ou de la sécurité civile IEE (2006)) sont des instances de tels systèmes multi-agents, tout comme toute organisation de personnes ou de moyens (entreprises, par exemple). Ces systèmes ont en commun le fait qu'ils soient composés d'autres systèmes (humains ou non, atomiques ou non) qui sont distribués géographiquement, gérés de façon indépendante et qui doivent partager des informations dans un environnement où les échanges d'informations entre ces systèmes doivent être réglementés par une politique.

Le travail présenté ici traite de ce type de systèmes. L'exemple utilisé tout au long de ce papier est l'exemple d'une entreprise avec un patron et des employés qui échangent des informations relatives au matériel utilisé dans l'entreprise. Ces échanges doivent être en accord avec une politique qui va, par exemple, imposer la diffusion des informations pertinentes et utiles dès que possible, tout en respectant certaines règles de confidentialité.

Une *politique d'échange d'informations* peut alors être perçue comme la réglementation que les agents doivent respecter et qui spécifie quels échanges d'informations sont obligatoires, interdits ou permis et sous quelles conditions. Mais, pour qu'une telle politique soit vraiment utile, il faut qu'elle vérifie un certain nombre de propriétés, et en particulier la *cohérence* et la *complétude*.

Selon Bieber & Cuppens (1991) où les politiques de confidentialité sont étudiées, la cohérence permet d'éviter les cas où l'agent a à la fois la permission et l'interdiction de savoir quelque chose. Plus généralement, selon Cholvy (1997) et Cholvy (1999), où l'on étudie la cohérence de réglementations en général, la cohérence d'une réglementation n'est pas équivalente à la simple cohérence d'un ensemble de formules. Selon ce travail, une réglementation est cohérente s'il n'existe pas de situation dans laquelle un agent pourrait se trouver face à des *contradictions normatives* ou *dilemmes* que l'on appelle également *conflits contradictoires* (une attitude donnée est à la fois prescrite et non prescrite, ou à la fois permise et interdite) et *conflits contraires* (une attitude donnée est à la fois prescrite et interdite) dans Vranes (2006). Suite à cette définition, la cohérence des politiques de sécurité a été étudiée dans Cholvy & Cuppens (1997).

Si la cohérence des politiques est une notion plutôt bien étudiée, la complétude a, quant à elle, reçu beaucoup moins d'attention. Bieber & Cuppens (1991) propose une définition de la complétude entre deux politiques de confidentialité (pour chaque information, l'agent doit avoir soit la permission, soit l'interdiction de la connaître), définition qui a été reprise dans Cuppens & Demolombe (1997) pour des politiques de sécurité à plusieurs niveaux.

Plus récemment, Cholvy *et al.* (2006) donne une définition de la cohérence et une définition de la complétude pour des politiques d'échange d'informations. Ces définitions constituent un point de départ pour le travail présent et ont été raffinées.

Ce papier est organisé de la façon suivante.

La section 2 présente le formalisme logique utilisé pour représenter des politiques d'échanges d'informations, la définition de la cohérence pour de telles politiques ainsi que la définition que l'on donne à la complétude. La section 3 traite du problème du raisonnement avec une politique incomplète. En reprenant l'approche qui a mené à la CWA (Closed World Assumption) dans le domaine des bases de données (Reiter (1998)), on présente des règles de complétion qui peuvent être utilisées pour compléter des politiques incomplètes et on discute de la façon dont elles pourraient être mises en oeuvre. Enfin, dans la section 4, on discute le travail réalisé et on propose des extensions à celui-ci.

2 Politiques d'échanges d'informations

2.1 Préliminaires

On reprend ici le langage défini dans Cholvy *et al.* (2007) pour modéliser une politique d'échange d'informations dans un système multi-agents. Rappelons le formalisme utilisé. Le langage logique, noté *L*, est un langage du premier ordre avec l'égalité. L'alphabet de *L* est basé sur quatre groupes de symboles : symboles représentant des constantes, symboles représentant des variables, symboles représentant des prédicats et des symboles représentant des fonctions.

Définition 1 (Constantes)
Les constantes sont divisées en :
- *ag-constantes : constantes représentant des agents*
- *i-constantes : constantes représentant des informations*
- *o-constantes : autres constantes*

Définition 2 (Variables)
Les variables sont divisées en :
- *ag-variables : variables représentant des agents*
- *i-variables : variables représentant des informations*
- *o-variables : autres variables*

Définition 3 (Prédicats)
Les prédicats sont divisés en :
- *D-prédicats : prédicats unaires O, P, F et T (signifiant respectivement Obligatoire, Permis, Interdit et Toléré).*
- *P-prédicats : prédicats pour n'importe quelle propriété sur les agents, les informations, ...*

Définition 4 (Fonctions)
Les fonctions sont définies de la façon suivante :
- *not(.) : fonction unaire pour représenter la négation.*
- *dire(.,.,.) : dire(x,y,i) représente l'évènement "l'agent x dit l'information i à l'agent y".*
- *i-fonctions : fonctions pour représenter les différentes caractéristiques des informations.*

Définition 5 (Termes)
Les termes sont définis de la façon suivante :
- *ag-terme : ag-constante ou ag-variable*
- *i-terme : les i-termes sont définis récursivement. Les i-constantes et les i-variables sont des i-termes et si $i_1, ...i_n$ sont des i-termes et f une i-fonction à n places alors $f(i_1, ...i_n)$ est également un i-terme.*
- *d-terme : les d-termes sont définis récursivement. Si x et y sont des ag-termes et i un i-terme, alors $dire(x, y, i)$ est un d-terme. De plus, si d est un d-terme alors $not(d)$ est également un d-terme.*

– *o-terme : o-constante ou o-variable*

Définition 6 (Formules de L)
Les formules de L sont définies récursivement de la façon suivante :
– *Soit d un d-terme. Alors $O(d)$, $P(d)$, $F(d)$ et $T(d)$ sont des D-littéraux et des formules de L.*
– *Si $t_1, ...t_n$ sont des termes autres que des d-termes, si R est un P-prédicat, alors $R(t_1, ...t_n)$ est un P-littéral et une formule de L.*
– *Soient F_1 et F_2 des formules de L et x une variable. Alors $\neg F_1$, $F_1 \wedge F_2, F_1 \vee F_2$, $\forall x\ F_1$ et $\exists x\ F_1$ sont des formules de L.*

Exemple 1
Dans toute la suite, on reprendra l'exemple d'une entreprise où le patron et les employés doivent échanger des informations sur le matériel utlisé. On considère le langage suivant : a, b, c, Patron, Employe sont des ag-constantes, i_1 et i_2 sont des i-constantes. RisqExp et VerifMat sont des i-constantes signifiant respectivement risque d'explosion et vérification du matériel. On utilisera les symboles x et y pour désigner des ag-variables et i pour désigner des i-variables. $Role(.,.)$ est un P-prédicat. $Role(x, Patron)$ signifie que l'agent x joue le rôle $Patron$. $Theme(.,.)$ est un P-prédicat. $Theme(i, RisqExp)$ signifie que l'information i a pour thème $RisqExp$. $Agent(.)$ est un P-prédicat. $Agent(b)$ signifie que b est un agent. $Recoit(.,.)$ est un P-prédicat. $Recoit(x, i)$ signifie que l'agent x reçoit l'information i. $O(dire(x, y, i))$ est un D-littéral signifiant que l'agent x est obligé de dire l'information i à l'agent y.

2.2 Politiques d'échange d'informations

Dans cette partie, on définit les règles pour une politique d'échange d'informations avec le langage défini ci-dessus.

Définition 7 (Règle)
Une règle est une formule de L qui, mise sous forme clausale, est une conjonction de clauses $l_1 \vee l_2 \vee ... \vee l_n$ telle que :
– *$\forall i \in \{1, ..., n\}$, l_i est un un P-littéral ou D-littéral.*
– *$\exists i \in \{1, ..., n\}$ tel que l_i est un D-littéral positif.*
– *si $\exists i \in \{1, ..., n\}$ tel que x est une variable dans l_i, alors $\exists j \in \{1, ..., n\}$ tel que l_j est un littéral négatif et contient la variable x.*

Définition 8 (Politique d'échange d'informations)
Une politique d'échange d'informations est un ensemble de règles.

Exemple 2
Considérons la règle (R_0) : lorsque le patron apprend une information sur la vérification du matériel, il a l'interdiction de la transmettre à ses employés. Cette règle peut être exprimée par la formule suivante :

$$(R_0) \quad \forall(x, y, i) \quad Role(x, Patron) \wedge Role(y, Employe)$$

$$\wedge Recoit(x, i) \wedge Theme(i, VerifMat) \rightarrow F(dire(x, y, i))$$

2.3 Cohérence et complétude d'une politique

On considère l'ensemble $\mathcal{A}$ d'axiomes suivants :
- $(Ax1) \quad \forall x \quad P(x) \leftrightarrow \neg O(not(x))$.
- $(Ax2) \quad \forall x \quad F(x) \leftrightarrow O(not(x))$.
- $(Ax3) \quad \forall x \quad T(x) \leftrightarrow P(x) \wedge P(not(x))$
- $(D) \quad \forall x \quad O(not(x)) \rightarrow \neg O(x)$.
- $(NO) \quad \forall x \quad O(not^{2n}(x)) \leftrightarrow O(x)$.
- $(NP) \quad \forall x \quad P(not^{2n}(x)) \leftrightarrow P(x)$.
- $(NF) \quad \forall x \quad F(not^{2n}(x)) \leftrightarrow F(x)$.

NOTATION : Soient $A_1, A_2, et A_3$ des formules de L. On écrira :
$A_1 \otimes A_2$ à la place de $(A_1 \vee A_2) \wedge \neg(A_1 \wedge A_2)$
et on écrira $A_1 \otimes A_2 \otimes A_3$ à la place de $(A_1 \vee A_2 \vee A_3) \wedge \neg(A_1 \wedge A_2) \wedge \neg(A_2 \wedge A_3)$ $\wedge \neg(A_1 \wedge A_3)$.
Cette notation représente le fait qu'une seule des A_i est vraie.

Théorème 1

Pour tout d-terme d on a : $\mathcal{A} \models O(d) \otimes T(d) \otimes F(d)$

Définition 9 (Monde)
On appelle monde, noté W, un ensemble de P-littéraux. Lorsque cet ensemble est complet (c'est à dire lorsque pour tout P-littéral l, on a $l \in W$ ou $\neg l \in W$), on parle de monde complet.

On appelle Dom l'ensemble des contraintes qui sont supposées vraies dans toute instance du monde régi par la politique. Par exemple, Dom pourrait contenir la formule suivante $(D_1) \neg(Calme \wedge Crise)$ indiquant qu'il n'est pas possible d'être à la fois dans un contexte de crise et dans un contexte calme.

Définition 10 (Cohérence dans un monde)
Soit $\mathcal{P}$ une politique et W un monde complet dans lequel elle est appliquée. $\mathcal{P}$ est cohérente dans W (vis à vis de Dom) si et seulement si $W \wedge Dom \wedge \mathcal{P} \wedge \mathcal{A}$ est cohérent.

Exemple 3
Supposons $Dom = \emptyset$. Considérons le monde W_0 suivant [1] :

$$W_0 = \{Agent(a), Agent(b), Role(a, Patron), Role(b, Employe)$$

$$Theme(i_1, VerifMat), Theme(i_2, RisqExp), Recoit(a, i_2)\}.$$

Soit $\mathcal{P}_0$ la politique contenant la règle (R_0). $(W_0, Dom, \mathcal{P}_0, \mathcal{A})$ est cohérent. $\mathcal{P}_0$ est donc cohérente dans le monde W_0 vis à vis de Dom.

[1] Pour des soucis de lisibilité, on n'écrit dans le monde W que les littéraux positifs. Tous les littéraux qui ne sont pas écrits sont donc considérés comme négatifs

Définition 11 (Cohérence)
Une politique $\mathcal{P}$ *est dite cohérente vis à vis de* Dom *si et seulement si il n'existe pas d'ensemble de formules f du langage* L *sans* D*-littéral tel que* $f \wedge Dom$ *cohérente et* $\mathcal{P} \wedge \mathcal{A} \wedge f \wedge Dom$ *incohérent.*

Proposition 1
$\mathcal{P}$ *est cohérente vis à vis de* Dom *si et seulement si pour tout monde* W *complet,* $\mathcal{P}$ *est cohérente dans* W*.*

Intuitivement, pour un monde donné, une politique est complète si elle prescrit l'attitude que tout agent devrait avoir vis à vis d'une information qu'il reçoit et vis à vis d'un autre agent. Le fait d'envoyer l'information à l'autre agent peut être obligatoire, interdit ou toléré (permis de faire et permis de ne pas faire). La complétude d'une politique dans un monde donné peut être formalisée de la façon suivante :

Définition 12 (Complétude dans un monde)
Soit $\mathcal{P}$ *une politique et* W *un monde complet dans lequel elle est appliquée.* $\mathcal{P}$ *est complète pour l'inférence classique (*$\models$*) dans* W *si et seulement si pour tout* x, y, i,

$$W \models Recoit(x,i) \wedge Agent(y) \wedge \neg(x = y) \Rightarrow$$

$$(\mathcal{P}, \mathcal{A}, W \models O(dire(x,y,i)) \text{ ou } \mathcal{P}, \mathcal{A}, W \models F(dire(x,y,i)) \text{ ou } \mathcal{P}, \mathcal{A}, W \models T(dire(x,y,i)))$$

On peut généraliser cette définition et définir la complétude globale.

Définition 13 (Complétude globale)
Soit $\mathcal{P}$ *une politique.* $\mathcal{P}$ *est complète globalement pour l'inférence classique (*$\models$*) si et seulement si pour tout monde complet* W*,* $\mathcal{P}$ *est complète dans* W*.*

Exemple 4
On a $W_0 \models Recoit(a,i_2) \wedge Agent(b) \wedge \neg(a = b)$ *mais* $\mathcal{P}_0, W_0, \mathcal{A} \not\models O(dire(a,b,i_2))$ *et* $\mathcal{P}_0, W_0, \mathcal{A} \not\models T(dire(a,b,i_2))$ *et* $\mathcal{P}_0, W_0, \mathcal{A} \not\models F(dire(a,b,i_2))$*. Ainsi,* $\mathcal{P}_0$ *est incomplète pour* $\models$*.*

Le problème de la complétude pour une politique est important. En effet, si une politique est incomplète, cela signifie qu'il y a des cas où elle ne précise pas quelle attitude suivre. Sans attitude à suivre, "n'importe quelle" attitude pourrait être observée et pourrait avoir des conséquences assez importantes. Nous sommes face à deux approches :
- soit on analyse la politique a priori pour détecter tous les “trous” (c'est à dire tous les cas où un agent va recevoir une information sans que la politique ne lui dise s'il est obligatoire, toléré ou interdit de la transmettre à un autre agent) puis on les fait corriger un à un par les concepteurs de la politique.
- soit sans analyser la politique, on prévoit des règles par défaut qui seront appliquées, par les agents du système, lorsqu'ils se trouveront face à un “trou”.

La première solution paraît fastidieuse à appliquer. En effet, l'ensemble des cas pour lesquels il manque une règle peut être assez élevé et les corriger un à un peut être long. C'est donc la deuxième solution que nous mettons en oeuvre ici.

3 Règles de complétion

3.1 Définitions

Dans ce paragraphe, on présente une solution qui s'inspire de la CWA définie par Reiter (1998) pour compléter les bases de données du premier ordre.

Selon la CWA, si la base de données est incomplète pour un littéral l (c'est à dire que l ne peut pas être déduit de la base de données), alors on considère que sa négation ($\neg l$) peut être déduite. Cette règle est motivée par le fait qu'une base de données est utilisée pour modéliser le monde réél. Comme dans le monde réél, un fait est soit vrai soit faux ($l \otimes \neg l$ est une tautologie en logique du premier ordre), alors une base de données doit permettre de déduire un fait ou sa négation.

Ici, étant donné un littéral l (l étant de la forme $dire(x, y, i)$), ce n'est pas la valeur de vérité de l qui nous intéresse mais le fait que, étant donnée une politique, on peut déduire que l est obligatoire, interdit ou toléré. Ces trois cas sont les seuls car l'ensemble d'axiomes $\mathcal{A}$ implique $O(l) \otimes F(l) \otimes T(l)$. Ainsi, si la politique est incomplète pour un littéral l (c'est à dire qu'elle ne permet pas de déduire ni $O(l)$, ni $F(l)$, ni $T(l)$) alors, elle ne peut être complétée qu'en considérant que l'on peut déduire $O(l)$, $T(l)$ ou $F(l)$. Ceci nous amène aux trois règles de complétion qui sont définies ci-dessous.

Pour rester le plus général possible, on définit des règles de complétion paramétrisées de façon à ce que la complétion par $O(l)$, $T(l)$ ou $F(l)$ dépende de certaines conditions. Ces conditions, que l'on note E_i, représentent des propriétés sur les agents (les agents ayant un rôle spécifique par exemple), sur les informations (les informations traitant d'un thème particulier par exemple), etc.

Soit $\mathcal{P}$ une politique et W un monde complet réglementé par $\mathcal{P}$.

NOTATION : Pour plus de lisibilité, on écrira "$\mathcal{P}, W\ incomplet\ pour\ (x, y, i)$" à la place de : $W \models Recoit(x, i) \wedge Agent(y) \wedge \neg(x = y)$ et $\mathcal{P}, W, \mathcal{A} \not\models O(dire(x, y, i))$ et $\mathcal{P}, W, \mathcal{A} \not\models T(dire(x, y, i))$ et $\mathcal{P}, W, \mathcal{A} \not\models F(dire(x, y, i))$

Soient E_1, E_2 et E_3 des formules dépendant de *x* et/ou de *y* et/ou de *i*. On prend $X = (x, y, i)$. Les trois règles d'inférence sont les suivantes :

$$(R_{E_1}) \qquad \frac{P, W\ incomplet\ pour\ X, \quad W \models E_1(X)}{F(dire(X))}$$

$$(R_{E_2}) \qquad \frac{P, W\ incomplet\ pour\ X, \quad W \models E_2(X)}{T(dire(X))}$$

$$(R_{E_3}) \qquad \frac{P, W\ incomplet\ pour\ X, \quad W \models E_3(X)}{O(dire(X))}$$

La politique peut être complétée de façon à ce qu'il soit interdit (R_{E_1}), toléré (R_{E_2}) ou obligatoire (R_{E_3}) pour un agent de dire une information à un autre agent. On définit ici une **nouvelle inférence** $\models_*$. Les règles d'inférence de $\models_*$ sont les règles d'inférence classique ($\models$) auxquelles on ajoute les trois règles d'inférence R_{E_1}, R_{E_2} et R_{E_3}.

La prochaine étape est de vérifier que la politique est cohérente et complète pour cette nouvelle inférence. Pour cela, il faut étendre la définition de la complétude d'une politique pour cette inférence $\models_*$.

3.2 Propriétés

Définition 14 (Complétude dans un monde)
Soit $\mathcal{P}$ une politique et W un monde complet. $\mathcal{P}$ est complète pour l'inférence $\models_*$ dans W si et seulement si pour tout $x, y, i,$, on a

$$\mathcal{P}, W \text{ incomplet pour } (x, y, i) \Rightarrow$$

$$(\mathcal{P}, W, \mathcal{A} \models_* O(dire(x,y,i)) \text{ ou } \mathcal{P}, W, \mathcal{A} \models_* T(dire(x,y,i)) \text{ ou } \mathcal{P}, W, \mathcal{A} \models_* F(dire(x,y,i)))$$

On peut généraliser cette définition et définir la complétude globale pour $\models_*$ de la même façon que dans la définition 10.

Proposition 2
Soit $\mathcal{P}$ une politique et W un monde complet dans lequel $\mathcal{P}$ est appliquée. Si

$$\forall X = (x, y, i), \mathcal{P}, W \text{ incomplet pour } X \Rightarrow W \models E_1(X) \vee E_2(X) \vee E_3(X)$$

alors $\mathcal{P}$ est complète pour l'inférence $\models_*$ dans le monde W.

Exemple 5
$E_1(x,y,i) = Theme(i, VerifMat), E_2(x,y,i) = False, E_3(x,y,i) = Theme(i, RisqExp)$.
On a $\mathcal{P}_0, W_0$ incomplet uniquement pour le triplet (a, b, i_2). On a bien $W_0 \models E_3(a, b, i_2)$ donc $W_0 \models (E_1(a,b,i_2) \vee E_2(a,b,i_2) \vee E_3(a,b,i_2))$. La politique est donc complète.

Il faut de même étendre la définition de la cohérence pour la nouvelle inférence.

Définition 15 (Cohérence pour l'inférence $\models_*$ dans un monde)
Soit W un monde complet et $\mathcal{P}$ une politique cohérente pour l'inférence $\models$ dans ce monde [2]. $\mathcal{P}$ est cohérente vis à vis de Dom dans W pour l'inférence $\models_*$ vis à vis de Dom si et seulement si $W, Dom, P, \mathcal{A}$ est cohérent pour l'inférence $\models_*$ (c'est à dire si $W, Dom, P, \mathcal{A} \not\models_* \perp$).

Proposition 3
Une politique $\mathcal{P}$ complète pour l'inférence $\models_*$ dans un monde complet W est cohérente pour l'inférence $\models_*$ dans W vis à vis de Dom si et seulement si

$$\forall X = (x, y, i) \quad \mathcal{P}, W \text{ incomplet pour } X \Rightarrow$$

$$W \models \neg(E_1(X) \wedge E_2(X)) \wedge \neg(E_1(X) \wedge E_3(X)) \wedge \neg(E_2(X) \wedge E_3(X))$$

Exemple 6
$\mathcal{P}_0, W_0$ est incomplet pour (a, i_2, b). Or, on a $W_0 \models \neg(E_1(a,b,i_2) \wedge E_3(a,b,i_2))$. Donc $W_0 \models \neg(E_1(a,b,i_2) \wedge E_3(a,b,i_2)) \wedge \neg(E_1(a,b,i_2) \wedge E_2(a,b,i_2)) \wedge \neg(E_2(a,b,i_2) \wedge E_3(a,b,i_2))$. La politique est donc cohérente pour $\models_*$ dans W_0.

Corollaire 1
Soit $\mathcal{P}$ une politique et W un monde réglementé par $\mathcal{P}$. $\mathcal{P}$ est cohérente pour $\models_*$ dans W vis à vis de Dom et complète pour $\models_*$ dans W si et seulement si

$$\forall X = (x, y, i) \mathcal{P}, W \text{ incomplet pour } X \Rightarrow W \models E_1(X) \otimes E_2(X) \otimes E_3(X)$$

[2] Il n'y a pas d'intérêt à prendre une politique qui est incohérente pour $\models$

3.3 Commentaires

Le corollaire qui vient d'être énoncé caractérise les conditions nécessaires et suffisantes pour que les trois règles de complétion $(R_{E_1}), (R_{E_2})$ et (R_{E_3}) complètent une politique de façon cohérente. Plus précisément, il exprime que si pour chaque cas où la politique ne prescrit rien (cas que l'on avait appelés "trous" plus haut) une et une seule des E_i est vraie, alors les trois règles complètent la politique de façon cohérente (puisqu'une et une seule des règles sera appliquée).

Cependant, cette condition nécessaire et suffisante, même si elle présente un intérêt d'un point de vue théorique, n'est pas très utile d'un point de vue pratique. En effet, pour vérifier qu'elle est satisfaite, il faut détecter tous les "trous" de la politique or c'est une opération que l'on souhaite éviter. Il faut donc essayer de trouver des conditions plus générales, qui seront suffisantes mais plus nécessaires pour garantir que les règles de complétion complètent la politique de façon cohérente.

La proposition suivante nous indique une telle condition :

Proposition 4
Soit $\mathcal{P}$ une politique et W un monde réglementé par $\mathcal{P}$. Si

$$\forall X = (x, y, i) W \models Recoit(x, i) \wedge Agent(y) \wedge \neg(x = y) \rightarrow E_1(X) \otimes E_2(X) \otimes E_3(X)$$

alors $\mathcal{P}$ est cohérente et complète pour $\models_*$ dans W

Une autre alternative serait de trouver une condition suffisante qui ne dépende pas d'un monde particulier mais qui soit valable pour l'ensemble des mondes. Le problème qui se pose alors est de déterminer quelles sont les éléments communs à l'ensemble des mondes. En effet, les E_i prennent en paramètre deux agents et une information, avoir des E_i valables pour tous les mondes signifierait que l'ensemble des agents et l'ensemble des informations soient communs à tous les mondes.

Une dernière alternative est alors de prendre des E_i qui ne dépendent ni des agents ni des informations.

Par exemple, nous pourrions poser que l'un des E_i soit Vrai et les deux autres soient Faux (ce qui représente une partition pour tous les mondes).

– Prenons par exemple $E_1 = True$, $E_2 = False$ et $E_3 = False$. Dans ce cas, selon les règles de complétion, tout ce qui n'est pas spécifié obligatoire ou toléré par la politique est interdit. On se trouve donc face à une attitude que l'on peut qualifier de *sévère*. Cette attitude pourrait être observée pour des réglementations qui régissent un système hautement sécurisé où chaque action doit être explicitement autorisée avant d'être effectuée.
– Un autre cas serait de prendre $E_1 = False$, $E_2 = True$ et $E_3 = False$. Ici, on se trouve dans la situation inverse à savoir que tout ce qui n'est pas interdit ou obligatoire par la politique est toléré. Cette attitude *tolérante* se retrouve par exemple dans des réglementations pour des lieux publics où tout ce qui n'est pas interdit est a priori autorisé.

Toujours dans cet ordre d'idée, on pourrait considérer par exemple le contexte général dans lequel la politique est appliquée. En prenant par exemple $E_1 = crise$, $E_2 = \neg crise$ et $E_3 = False$, on aura une attitude sévère dans un contexte de crise et une attitude tolérante sinon.

4 Discussion

Après avoir donné le cadre logique et montré comment formaliser une politique d'échange d'informations dans ce cadre, on a rappelé la définition de la cohérence et l'on a défini ce qu'était la complétude. Le problème était alors de raisonner avec des politiques incomplètes. Pour cela, une méthode a été proposée pour compléter une politique : utiliser une nouvelle inférence avec trois règles d'inférence qui peuvent être appliquées pour les éléments pour lesquels la politique est incomplète. Après avoir complété une politique, on peut vérifier que le résultat obtenu est toujours cohérent. Etant donnée une situation, dès qu'un agent reçoit une information, la question est de savoir si la politique permet d'inférer qu'il est obligatoire, interdit ou toléré de dire cette information à un autre agent. Si la réponse à cette question est négative, alors la nouvelle question est de savoir quelle condition E_i est vraie. Si E_1 (resp. E_2, E_3) est vrai, alors l'agent peut déduire qu'il est interdit (resp. toléré, obligatoire) de dire l'information. Les conditions sur les E_i permettent d'assurer que l'agent peut se trouver dans un et un seul de ces trois cas. Le problème est de généraliser ces conditions le plus possible de façon à ce qu'elles soient facilement vérifiables (et donc peu coûteuses) et que les règles de complétion puissent donc être mises en oeuvre. On peut noter que les règles de complétion ressemblent beaucoup aux défauts de Reiter (1980). Dans Roussel (2007), une reformulation des règles de complétion dans la théorie des défauts a été faite. Elle conduit à la définition de trois défauts normaux. L'équivalence des approches est également prouvée.

Le travail effectué ici peut être étendu dans différentes directions.

Tout d'abord, on pourrait ajouter la notion de temps à ce qui a été fait. Comme cela a été montré dans Demolombe *et al.* (2005), le problème du temps est très important lorsque l'on aborde la notion d'obligation. Dans notre cas, l'impact du temps serait considérable et plutôt difficile à gérer. En effet, il faudrait considérer différents temps (ou dates) : date à laquelle l'information est créée, date à laquelle cette dernière est reçue par un agent, date à laquelle l'obligation est créée, date à laquelle l'agent dit l'information à un autre agent, date jusqu'à laquelle l'obligation est valide, etc.

Ensuite, le prédicat "Recoit" mériterait qu'on lui accorde plus d'attention et qu'on étudie sa sémantique en relation avec la révision de la base de croyances de l'agent. En effet les obligations, interdictions et tolérances exprimées dans la politique ne devraient pas être déclenchées par l'arrivée d'une nouvelle information dans la base de croyances de l'agent, mais par le calcul des "nouvelles" croyances (c'est à dire celles qui appartiennent à la différence entre la base avant et après la révision des croyances).

Références

(2006). IEEE international conference on systems of systems engineering.

BIEBER P. & CUPPENS F. (1991). Expression of confidentiality policies with deontic logic. In *Proceedings of the First Workshop on Deontic Logic and Computer Science (DEON'91)*.

CHOLVY L. (1997). An application of SOL-deduction : checking regulation consistency. In *IJCAI'97 Poster Collection*.

CHOLVY L. (1999). Checking regulation consistency by using SOL-resolution. In *International Conference on Artificial Intelligence and Law*, p. 73–79.

CHOLVY L. & CUPPENS F. (1997). Analysing consistency of security policies. In *IEEE Symposium on Security and Privacy*.

CHOLVY L., GARION C. & SAUREL C. (2006). Information sharing policies for coalition systems. In *IST-062 Symposium on "Dynamic Communications Management" (RTO)*.

CHOLVY L., GARION C. & SAUREL C. (2007). Modélisation de réglementations pour le partage d'informations dans un SMA. In *Modèles Formels de l'Interaction*.

CUPPENS F. & DEMOLOMBE R. (1997). A modal logical framework for security policies. In *(Proceedings of ISMIS'97) Lectures Notes in Artificial Intelligence, 1325* : Springer.

DEMOLOMBE R., BRETIER P. & LOUIS V. (2005). Formalisation de l'obligation de faire avec délais. In *Proc. MFI'2005*.

REITER R. (1980). A logic for default reasoning. *Artificial Intelligence*, **13**(1,2).

REITER R. (1998). On closed world data bases. In J. N. H. GALLAIRE, J. MINKER, Ed., *Logic and Databases* : Plenum Publications, New-York.

ROUSSEL S. (2007). Complétude d'une politique de partage d'informations. *Rapport de stage Master Recherche*.

VRANES E. (2006). The definition of "norm conflict" in international law and legal theory. *The European Journal of International Law*, **17**(2), 395–418.

Représentation de contraintes qualitatives pour le temps et l'espace en SAT

Jean-François Condotta et Dominique D'Almeida

CRIL-CNRS, Université d'Artois, 62307 Lens Cedex, France
{condotta, dalmeida}@cril.univ-artois.fr

Résumé : Dans cet article nous considérons le problème de la cohérence des réseaux de contraintes qualitatives pour le temps et l'espace. Une nouvelle traduction permettant de représenter et de résoudre ce problème dans le cadre de la logique propositionnelle est proposée. La définition de cette représentation présuppose l'existence d'un ordre particulier sur les relations de base du formalisme qualitatif utilisé tel que celui du treillis conceptuel de l'algèbre des intervalles.

1 Introduction

La représentation et le raisonnement sur les informations temporelles et spatiales est une tâche importante de nombreuses applications de l'Intelligence Artificielle : les systèmes d'informations géographiques (GIS), la compréhension du langage naturelle, la navigation de robot, la planification temporelle et spatiale. Ces dernières années de nombreux formalismes ont été développés pour représenter et raisonner sur les configurations temporelles ou spatiales à l'aide de relations dites qualitatives. Un formalisme qualitatif [1, 15, 8, 14] utilise des éléments particuliers (sous-ensembles d'un espace topologique, points d'une droite des rationnels, intervalles d'une droite, rectangles d'un plan, tuples de points, ...) pour représenter les entités spatiales ou temporelles du système, et considère un nombre limité de relations entre ces éléments. Chacune de ces relations correspond à une situation temporelle ou spatiale particulière. Par exemple, considérons le bien connu formalisme temporel d'Allen appelé l'algèbre des intervalles [1]. Il utilise des intervalles de la droite des rationnels pour représenter les entités temporelles. Treize relations de base entre ces intervalles sont utilisées pour modéliser les différentes situations qualitatives possibles entre les entités temporelles (Figure 1).

Des réseaux de contraintes appelés réseaux de contraintes qualitatives (RCQ en abrégé) peuvent être utilisés pour représenter les contraintes posées sur les positions relatives d'un ensemble d'entités temporelles ou spatiales. Chaque contrainte d'un RCQ est définie par l'ensemble des relations de base permises entre les entités temporelles ou spatiales concernées. Étant donné un RCQ, le principal problème qui se pose est le problème de sa cohérence. Ce problème consiste à déterminer si ce réseau possède des solutions satisfaisant ses contraintes ou non. Pour résoudre ce problème de manière efficace, des méthodes de recherche utilisant la méthode de fermeture par faible composition comme propagation locale de contraintes d'une part, et une décomposition des contraintes en relations d'une classe dite traitable d'autre part ont été définies [11]. Concrètement, lors de la recherche d'une solution chaque contrainte est découpée en relations d'une classe pour laquelle la méthode de fermeture par faible composition est complète et est successivement instanciée par une de ces relations. Après chaque instanciation, la méthode de fermeture par faible composition est appliquée. Le facteur de branchement lors de la recherche est ainsi grandement diminué et rend la recherche plus rapide. Pour un formalisme qualitatif donné, de nombreux travaux ont eut pour objectif la caractérisation de classes dites traitables. L'ensemble des relations convexes définies à l'aide d'un treillis dit conceptuel sur les relations de base correspond à une classe traitable de nombreux formalismes [13, 12, 8, 3, 2, 14].

Des travaux ont également consisté à représenter les RCQ en logique propositionnelle traduisant ainsi le problème de la cohérence d'un RCQ en un problème de satisfiabilité d'un ensemble de clauses propositionnelles. Nous pouvons citer la traduction proposée par Nebel et Burckert [12] permettant de modéliser des contraintes de l'algèbre des intervalles en un ensemble de clauses SAT. Grossièrement, cet ensemble de clauses SAT modélise l'orde linéaire correspondant à la configuration des bornes des intervalles du système. Des traductions plus génériques et pouvant être utilisées dans la cadre de tout formalisme qualitatif ont été définies et étudiées récemment [7, 6]. L'une de ces traductions consiste en la modélisation en SAT de toutes les combinaisons possibles de relations de base entre chaque triplet de variables. Ces combinaisons possibles sont fournies par la table de composition faible. Une autre approche consiste en la modélisation des combinaisons interdites. L'intérêt principal de ces traductions vers SAT est l'utilisation de solveurs SAT très performants tels que zChaff ou MiniSAT pour résoudre le problème de la cohérence des RCQ. Les tailles très importantes des ensembles de clauses obtenus par ces traductions limitent leur utilisation.

Force est de constater qu'aucune traduction SAT proposée dans la littérature utilise de manière générique la définition d'une classe traitable. Nous remédions à cela en proposant une traduction exploitant la définition des relations convexes. Plus exactement, la représentation SAT proposée utilise un treillis défini sur les relations de base possédant des caractéristiques similaires à celui du treillis conceptuel du formalisme d'Allen. À l'aide de cette nouvelle traduction les RCQ dont les contraintes sont définies par des intervalles d'un tel treillis seront représentés à l'aide d'un ensemble de clauses de Horn.

Cet article est organisé de la manière suivante. Dans la section 2 se trouvent des rappels concernant les formalismes qualitatifs. La section 3 est consacrée à la définition de notre traduction des RCQ en logique propositionnelle. Dans la section 4 nous montrons que la traduction proposée est complète pour le problème de la cohérence. La section 5 est dévolue à une comparaison de notre traduction avec une traduction SAT classique. Nous exposons les perspectives futures de nos travaux et concluons dans la section 6.

2 Rappels sur les formalismes qualitatifs

Dans la suite, nous supposons donné un formalisme qualitatif pour le temps ou l'espace défini sur un ensemble fini B de relations de base binaires sur un domaine D. Nous effectuons également l'hypothèse qu'elles sont complètes et deux à deux disjointes, c'est-à-dire que tout couple d'éléments de D appartient à exactement une relation de B. De plus, nous supposons que B contient la relation identité sur D, dénotée par Id dans la suite. Pour illustration, considérons le bien connu formalisme d'Allen : l'algèbre des intervalles (AI). AI est basée sur 13 relations binaires définis sur un ensemble d'intervalles, celui de la droite des nombres rationnels par exemple. Chacune de ces relations correspond à un ordre partculier des 4 bornes de deux intervalles.

Relation	Symbole	Inverse	Signification
precedes	b	bi	
meets	m	mi	
overlaps	o	oi	
starts	s	si	
during	d	di	
finishes	f	fi	
equals	eq	eq	

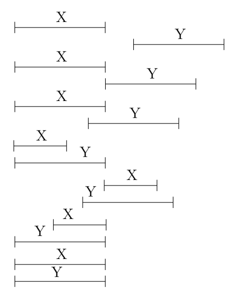

FIG. 1 – Les relations de base de AI

Les informations temporelles ou spatiales sur la configuration d'un ensemble d'entités peuvent être représentées à l'aide d'un réseau de contraintes dit réseau de contraintes qualitatives (RCQ en abrégé). Formellement, un réseau de contraintes qualitatives $\mathcal{N}$ est un couple (V, C) où V est un ensemble fini de n variables $v_0, \ldots, v_{n-1}$ (avec n est un entier positif) et C est une application qui, pour chaque couple (v_i, v_j) de variables de V associe un sous-ensemble $C(v_i, v_j)$ de l'ensemble des relations de base : $C(v_i, v_j) \subseteq \mathsf{B}$. Nous dénoterons par A l'ensemble 2^{B} de tous les ensembles de B. Pour $r \in \mathsf{A}$, deux éléments $x, y \in \mathsf{D}$ satisfont r, ce que nous dénoterons par $x\ r\ y$, ssi il existe une relation de base $a \in r$ telle que $(x, y) \in a$. Ainsi chaque élément r de A peut être considéré comme l'union de toutes les relations de base la composant. Nous utilisons le terme de "relation" pour désigner une telle union de relations de base. L'ensemble A est muni de l'opération d'intersection ($\cap$) et l'opération d'union ($\cup$). Il est également muni avec l'opération unaire inverse ($^{-1}$) et l'opération binaire de faible composition ($\circ$) : l'inverse d'une relation r de A est la relation de A correspondant à la transposée de r ; elle correspond à l'union des inverses des relations de base appartenant à r. La faible composition $a \circ b$ de $a, b \in \mathsf{B}$ est la relation r définie par $r = \{c : \exists x, y, z \in \mathsf{D}, x\ a\ y, y\ b\ z \text{ et } x\ c\ z\}$. La faible composition $r \circ s$ de $r, s \in \mathsf{A}$ est la relation $t = \bigcup_{a \in r, b \in s} \{a \circ b\}$. Intuitivement, $r \circ s$ est l'ensemble de toutes les relations de base possibles entre $x \in \mathsf{D}$ et $y \in \mathsf{D}$ lorsqu'il existe $z \in \mathsf{D}$ avec $x\ r\ z$ et $z\ s\ y$. Concernant les RCQ nous utiliserons les définitions suivantes dans la suite :

Définition 1
Soit $\mathcal{N} = (V, C)$ un RCQ*, avec $V = \{v_0, \ldots, v_{n-1}\}$. Une* solution partielle *de $\mathcal{N}$ sur $V' \subseteq V$ est une application σ de V' vers* D *telle que $\sigma(v_i)\ \ C(v_i, v_j)\ \ \sigma(v_j)$, pour tout $v_i, v_j \in V'$. Une* solution *de $\mathcal{N}$ est une solution partielle de V. $\mathcal{N}$ est* cohérent *si et seulement si il admet une solution. $\mathcal{N}$ est* $\circ$-fermé *si et seulement si pour tout*

$v_k, v_i, v_j \in V$, $C(v_i, v_j) \subseteq C(v_i, v_k) \circ C(v_k, v_j)$ et $C(v_i, v_j) \neq \emptyset$ (remarquons que nous pouvons nous restreindre aux triplets $v_k, v_i, v_j \in V$ avec $i < j$). Un sous-RCQ *$\mathcal{N}'$ de $\mathcal{N}$ est un* RCQ *(V, C') où $C'(v_i, v_j) \subseteq C(v_i, v_j)$ pour tout $v_i, v_j \in V$. La notation $\mathcal{N}' \subseteq \mathcal{N}$ dénotera le fait que $\mathcal{N}'$ est un sous-*RCQ *de $\mathcal{N}$. Un* scénario *de $\mathcal{N}$ est un* sous-RCQ *(V, C') de $\mathcal{N}$ tel que $C'(v_i, v_j) = \{a\}$ avec $a \in$* B. *Un* scénario cohérent *de $\mathcal{N}$ est un scénario de $\mathcal{N}$ admettant une solution. $\mathcal{N}' = (V, C')$ est* équivalent *à $\mathcal{N}$ ssi les deux* RCQ *ont mêmes solutions.*

Étant donné un RCQ $\mathcal{N} = (V, C)$, des algorithmes de propagation locale de contraintes [10, 9, 4, 5] peuvent être utilisés pour dériver en temps polynomial un sous-RCQ équivalent et $\circ$-fermé (ou contenant la relation vide comme contrainte). Le principe de ces algorithmes est d'itérer l'opération $C(v_i, v_j) \leftarrow C(v_i, v_j) \cap (C(v_i, v_k) \circ C(v_k, v_j))$ pour tout triplet de variables $v_i, v_j, v_k \in V$ jusqu'à ce qu'un point fixe soit obtenu. Dans la suite, méthode de la fermeture par faible composition désignera un tel algorithme. Étant donné un sous-ensemble E de A, nous dirons que la methode de la fermeture par faible composition est complète pour E lorsque cette méthode est suffisante pour résoudre le problème de la cohérence des RCQ définis sur E. C'est-à-dire lorsque la non obtention de la relation vide comme contrainte permet d'affirmer que le RCQ initial défini sur E admet une solution.

3 Une traduction des RCQ en logique propositionnelle

Dans cette section nous proposons et étudions une traduction permettant de représenter un RCQ en un ensemble de clauses propositionnelles dont la satisfiabilité permet de déterminer la cohérence ou la non cohérence du RCQ. Un prérequis à la définition de cette traduction est l'existence d'un ordre partiel $\preceq$ sur B devant satisfaire certaines propriétés. Tout d'abord $(B, \preceq)$ doit être un treillis, ainsi pour tout $a, b, c \in$ B nous avons : (1) $a \preceq a$, (2) si $a \preceq c$ et $c \preceq b$ alors $a \preceq b$, (3) si $a \preceq b$ et $b \preceq a$ alors $a = b$, (4) il existe deux éléments de B correspondant à $\mathsf{Inf}\{a, b\}$ et $\mathsf{Sup}\{a, b\}$. Étant donné deux éléments $a, b \in$ B, l'intervalle $[a, b]$ représentera la relation de A définie par $\{c \in \mathsf{B} : a \preceq c$ et $c \preceq b\}$. $\mathcal{C}_\preceq$ dénotera le sous-ensemble de A correspondant aux intervalles de $(\mathsf{B}, \preceq)$. Formellement, $\mathcal{C}_\preceq = \{[a, b] : a, b \in \mathsf{B}\}$. Dans la suite, nous appelons simple RCQ, SRCQ en abrégé, tout RCQ $= (V, C)$ dont chaque contrainte C_{ij} est définie par une relation $[C^-_{ij}, C^+_{ij}]$ de $\mathcal{C}_\preceq$.

Concernant $(\mathsf{B}, \preceq)$ nous posons deux nouvelles propriétés concernant les opérations d'inverse et de faible composition :

- pour tout $a, b \in$ B, si $a \preceq b$ alors $b^{-1} \preceq a^{-1}$;
- pour tout $a, b, c, d \in$ B, $[a, b] \circ [c, d] = [\mathsf{Inf}(a \circ c), \mathsf{Sup}(b \circ d)]$.

Pour de nombreux formalismes qualitatifs [13, 12, 8, 3, 2, 14] un tel treillis existe, il est parfois appelé treillis qualitatif ou conceptuel tandis que les relations de l'ensemble $\mathcal{C}_\preceq$ sont nommées relations convexes. Le treillis conceptuel de l'algèbre des intervalles est représenté dans la figure 2. Un SRCQ défini sur ce treillis est également représenté dans cette même figure. La contrainte C_{ij} n'est pas représentée lorsque C_{ji} est déjà présente ou bien lorsqu'elle corespond à la relation totale (*i.e.* B) ou bien encore lorsque $i = j$.

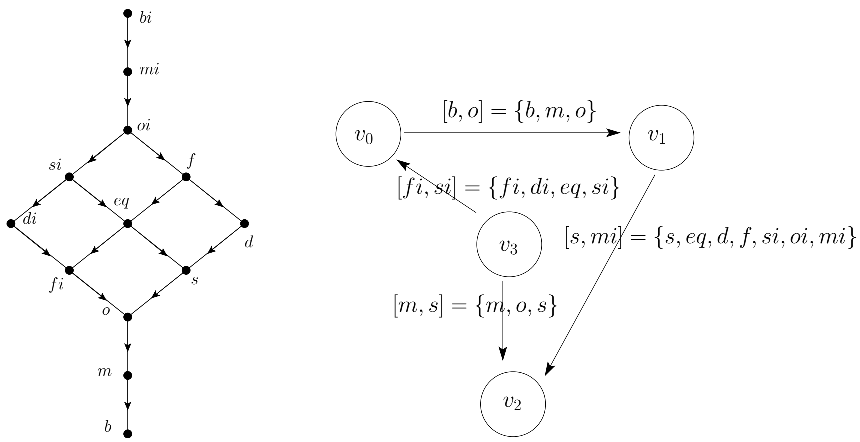

FIG. 2 – Le treillis conceptuel $(\mathsf{B}, \preceq)$ de AI (à gauche) et un SRCQ cohérent (à droite)

Il est temps maintenant de définir notre traduction permettant la représentation d'un RCQ en un ensemble de clauses. Cet ensemble de clauses permet de modéliser des propriétés sur les liens entre le treillis $(\mathsf{B}, \preceq)$ et les relations de base devant être satisfaites entre les différentes variables.

Définition 2
Soit $\mathcal{N} = (V, C)$ un SRCQ et $n = |V|$. Sur l'ensemble de propositions $\{C_{ij} \preceq a$ avec $a \in B$ et $i, j \in \{0, \ldots, n-1\}\} \cup \{a \preceq C_{ij}$ avec $a \in B$ et $i, j \in \{0, \ldots, n-1\}\}$ nous définissons $\mathsf{Sat}(\mathcal{N})$ par l'ensemble des clauses suivantes :

– *pour chaque contrainte $C_{ij} = [a_{ij}, b_{ij}]$ avec $i, j \in \{0, \ldots, n-1\}$, deux clauses unitaires bornant les relations de base possibles sont introduites :*

$$a_{ij} \preceq C_{ij} \text{ et } C_{ij} \preceq b_{ij} \quad (I)$$

– *Nous introduisons également des clauses modélisant une propriété sur les infimums et les supremums de $\preceq$ pour tout $a, b \in \mathsf{B}$:*

$$\neg(a \preceq C_{ij}) \vee \neg(b \preceq C_{ij}) \vee \mathsf{Sup}\{a, b\} \preceq C_{ij} \quad (II\ a)$$

$$\neg(C_{ij} \preceq a) \vee \neg(C_{ij} \preceq b) \vee C_{ij} \preceq \mathsf{Inf}\{a, b\} \quad (II\ b)$$

– *Des clauses correspondant à la propriété de transitivité de $\preceq$ sont introduites pour tout $a, b \in \mathsf{B}$ tels que $a \not\preceq b$:*

$$\neg(a \preceq C_{ij}) \vee \neg(C_{ij} \preceq b) \quad (III)$$

– *Deux clauses concernant l'opération inverse est introduite pour tout $a \in \mathsf{B}$:*

$$\neg(a \preceq C_{ij}) \vee C_{ji} \preceq a^{-1} \text{ et } \neg(C_{ij} \preceq a) \vee a^{-1} \preceq C_{ji} \quad (IV)$$

– *Enfin nous introduisons pour chaque triplet de contraintes C_{ik}, C_{kj}, C_{ij}, avec $i, j, k \in \{0, \ldots, n-1\}$ et $i < j$, deux clauses découlant de propriétés de l'opération de faible composition :*

$$\neg(a \preceq C_{ik}) \vee \neg(b \preceq C_{kj}) \vee \mathsf{Inf}(a \circ b) \preceq C_{ij} \quad (V a)$$

$$\neg(C_{ik} \preceq a) \vee \neg(C_{kj} \preceq b) \vee C_{ij} \preceq \mathsf{Sup}(a \circ b) \quad (Vb)$$

Notons que $\mathsf{Sat}(\mathcal{N})$ ne contient que des clauses de Horn. En conséquence de quoi sa satisfiabilité peut être déterminée en temps polynomial. De plus, nous pouvons remarquer que les traductions de deux SRCQ définis sur un même formalisme qualitatif et sur un même ensemble de variables ne vont différer que pour l'ensemble des clauses (I). Des clauses peuvent être également trivialement vraies du fait qu'elles contiennent un littéral et son opposé. Elles peuvent donc être omises. Par exemple, pour deux relations de base $a, b \in \mathsf{B}$ tels que $\mathsf{Sup}\{a, b\} = a$ ou $\mathsf{Sup}\{a, b\} = b$, les clauses (II a) sont trivialement satisfaites. Considérons le SRCQ représenté dans la figure 2. Sa traduction en SAT contient notamment les clauses $eq \preceq C_{00}$, $C_{00} \preceq eq$, $b \preceq C_{01}$, $C_{01} \preceq o$, $oi \preceq C_{10}$, $C_{10} \preceq bi$, $b \preceq C_{02}$, $C_{02} \preceq bi$, $b \preceq C_{20}$, $C_{20} \preceq bi$ parmi les clauses (I). L'ensemble des clauses (II) contiendra par exemple $\neg(fi \preceq C_{01}) \vee \neg(d \preceq C_{01}) \vee f \preceq C_{01}$ puisque $f = \mathsf{Sup}\{fi, d\}$. Dans les ensembles (III) et (IV) on aura entre autres clauses $\neg(di \preceq C_{01}) \vee \neg(C_{01} \preceq eq)$ et $\neg(f \preceq C_{10}) \vee C_{01} \preceq fi$ respectivement. Comme $m \circ f = \{o, s, d\} = [o, d]$, les clauses $\neg(m \preceq C_{01}) \vee \neg(f \preceq C_{12}) \vee o \preceq C_{02}$ et $\neg(C_{01} \preceq m) \vee \neg(C_{12} \preceq f) \vee C_{02} \preceq d$ apprtiennent à l'ensemble de clauses (V).

Précédemment nous avons proposé une traduction des SRCQ en logique propositionnelle. Cette traduction peut-être facilement généralisée au cas de RCQ quelconques. En effet, étant donnée une contrainte définie par une relation $C_{ij} \in \mathsf{A}$, nous pouvons toujours découper C_{ij} en un ensemble de k relations $\{C_{ij}^1, \ldots, C_{ij}^{k_{ij}}\}$, avec k_{ij} un nombre entier infèrieur à $|\mathsf{B}|$, tel que $C_{ij} = \bigcup_{l \in 1,\ldots,k_{ij}} C_{ij}^l$ et C_{ij}^l est une relation appartenant à $\mathcal{C}_{\preceq}$ pour tout $l \in \{1, \ldots, k_{ij}\}$. Ainsi, $C_{ij}^l = [a_{ij}^1, b_{ij}^1] \cup \ldots \cup [a_{ij}^{k_{ij}}, b_{ij}^{k_{ij}}]$ avec $a_{ij}^1, \ldots, a_{ij}^{k_{ij}}, b_{ij}^1, \ldots, b_{ij}^{k_{ij}} \in \mathsf{B}$. La modélisation d'une telle contrainte sera effectuée en remplaçant les deux clauses unitaires introduites dans (I) par un ensemble de clauses correspondant à la formule logique $(a_{ij}^1 \preceq C_{ij} \wedge C_{ij} \preceq b_{ij}^1) \vee \ldots \vee (a_{ij}^{k_{ij}} \preceq C_{ij} \wedge C_{ij} \preceq b_{ij}^{k_{ij}})$. Par exemple, supposons que pour le RCQ représenté dans la figure 2 la contrainte C_{02} ne soit plus définie par la relation totale mais par la relation $\{m, o, s, eq, f, d\}$. Nous avons $C_{02} = \{m, o, s, eq, f, d\} = \{m, o\} \cup \{s, eq, f, d\} = [m, o] \cup [s, d]$. C_{02} pourra donc être modélisée par des clauses correspondant à la formule logique $(m \preceq C_{02} \wedge C_{02} \preceq o) \vee (s \preceq C_{02} \wedge C_{02} \preceq d)$: $m \preceq C_{02} \vee s \preceq C_{02}$, $m \preceq C_{02} \vee C_{02} \preceq d$, $C_{02} \preceq o \vee s \preceq C_{02}$, $C_{02} \preceq o \vee C_{02} \preceq d$. Remarquons que ces clauses ne sont pas des clauses de Horn. Notons également que dans le cas des relations d'Allen, le nombre de relations de $\mathcal{C}_{\preceq}$ nécessaires pour un découpage minimal est de 3.55 en moyenne [11]. En règle générale, le "découpage" d'une relation quelconque de A en relations de $\mathcal{C}_{\preceq}$ n'est pas unique. Pour la suite, nous supprimons ce non determinisme en supposant fixé un découpage unique pour chaque relation $r \in \mathsf{A}$.

4 Complétude de la traduction Sat

Dans cette section nous prouvons que la satisfiabilité de l'ensemble de clauses issu de la traduction Sat appliquée à un RCQ permet de déterminer la cohérence ou non de ce dernier. Avant cela, prouvons une propriété qui nous sera utile dans la suite et qui concerne le treillis $(\mathsf{B}, \preceq)$:

Proposition 1
Soient $a, b, c, d \in \mathsf{B}$. *Si* $a \preceq c$ *et* $b \preceq d$ *alors nous avons* $\mathsf{Inf}(a \circ b) \preceq \mathsf{Inf}(c \circ d)$ *et* $\mathsf{Sup}(a \circ b) \preceq \mathsf{Sup}(c \circ d)$.

Preuve Considérons les relations $[a, \mathsf{Sup}(\mathsf{B})]$, $[b, \mathsf{Sup}(\mathsf{B})]$, $[c, \mathsf{Sup}(\mathsf{B})]$ et $[d, \mathsf{Sup}(\mathsf{B})]$. Comme $a \preceq c$ et $b \preceq d$ nous avons $[c, \mathsf{Sup}(\mathsf{B})] \subseteq [a, \mathsf{Sup}(\mathsf{B})]$ et $[d, \mathsf{Sup}(\mathsf{B})] \subseteq [b, \mathsf{Sup}(\mathsf{B})]$. En conséquence de quoi $[c, \mathsf{Sup}(\mathsf{B})] \circ [d, \mathsf{Sup}(\mathsf{B})] \subseteq [a, \mathsf{Sup}(\mathsf{B})] \circ [b, \mathsf{Sup}(\mathsf{B})]$. Du fait que $[c, \mathsf{Sup}(\mathsf{B})] \circ [d, \mathsf{Sup}(\mathsf{B})] = [\mathsf{Inf}(c \circ d), \mathsf{Sup}(\mathsf{Sup}(\mathsf{B}) \circ \mathsf{Sup}(\mathsf{B}))]$ et $[a, \mathsf{Sup}(\mathsf{B})] \circ [b, \mathsf{Sup}(\mathsf{B})] = [\mathsf{Inf}(a \circ b), \mathsf{Sup}(\mathsf{Sup}(\mathsf{B}) \circ \mathsf{Sup}(\mathsf{B}))]$ il s'ensuit que $\mathsf{Inf}(a \circ b) \preceq \mathsf{Inf}(c \circ d)$. En considérant les relations $[\mathsf{Inf}(\mathsf{B}), a]$, $[\mathsf{Inf}(\mathsf{B}), b]$, $[\mathsf{Inf}(\mathsf{B}), c]$, $[\mathsf{Inf}(\mathsf{B}), d]$ et en utilisant un raisonnement similaire nous pouvons démontrer que $\mathsf{Sup}(a \circ b) \preceq \mathsf{Sup}(c \circ d)$. $\dashv$
Maintenant prouvons que la traduction SAT d'un SRCQ fermé par faible composition est satisfiable :

Proposition 2
Soit $\mathcal{N} = (V, C)$ *un* SRCQ *défini sur* $(\mathsf{B}, \preceq)$. *Si* $\mathcal{N}$ *admet un scénario* $\circ$*-fermé alors* $\mathsf{Sat}(\mathcal{N})$ *est satisfiable.*

Preuve Soit $\mathcal{S}$ un scenario $\circ$-fermé de $\mathcal{N}$. Notons s_{ij} la relation de base correspondant à la contrainte entre les variables v_i et v_j de $\mathcal{S}$ pour tout $i, j \in \{0, \ldots, n-1\}$ avec $n = |V|$. Nous définissons une interprétation I de $\mathsf{Sat}(\mathcal{N})$ de la manière suivante : pour chaque contrainte C_{ij} et chaque $a \in \mathsf{B}$: si $a \preceq s_{ij}$ alors $I(a \preceq C_{ij}) = vrai$, $I(a \preceq C_{ij}) = faux$ sinon. Pour chaque contrainte C_{ij} et chaque $a \in \mathsf{B}$: si $s_{ij} \preceq a$ alors $I(C_{ij} \preceq a) = vrai$, $I(C_{ij} \preceq a) = faux$ sinon. Comme $\mathcal{S}$ est un scénario de $\mathcal{N}$, nous avons $a_{ij} \preceq s_{ij} \preceq b_{ij}$, avec $C_{ij} = [a_{ij}, b_{ij}]$. Il s'ensuit que I satisfait les clauses (I). $s_{ij} \in \mathsf{B}$ et $(\mathsf{B}, \preceq)$ est un treillis, il s'ensuit que les clauses (II) et (III) sont satisfaites. $s_{ij} = s_{ji}^{-1}$ et nous savons que si $a \preceq b$ alors $b^{-1} \preceq a^{-1}$. Il s'ensuit que les clauses (IV) sont satisfaites par I. Comme $\mathcal{S}$ est $\circ$-fermé nous avons $s_{ij} \in [s_{ik}, s_{ik}] \circ [s_{kj}, s_{kj}]$ pour tout $i, j, k \in \{0, \ldots, n-1\}$. Ainsi, $\mathsf{Inf}(s_{ik} \circ s_{kj}) \preceq s_{ij} \preceq \mathsf{Sup}(s_{ik} \circ s_{kj})$. Pour tout $a, b \in \mathsf{B}$, si $a \preceq s_{ik}$ et $b \preceq s_{kj}$ alors $\mathsf{Inf}(a \circ b) \preceq \mathsf{Inf}(s_{ik} \circ s_{kj})$ (Prop. 1). Il s'ensuit que $\mathsf{Inf}(a \circ b) \preceq s_{ij}$. Pour tout $a, b \in \mathsf{B}$, si $s_{ik} \preceq a$ et $s_{kj} \preceq b$ alors $\mathsf{Sup}(s_{ik} \circ s_{kj}) \preceq \mathsf{Sup}(a \circ b)$ (Prop. 1). Il s'ensuit que $s_{ij} \preceq \mathsf{Sup}(a \circ b)$. Les clauses (V) sont donc satisfaites. Nous pouvons donc conclure que I est un modèle de $\mathsf{Sat}(\mathcal{N})$. $\dashv$

À partir d'une interprétation satisfaisant $\mathsf{Sat}(\mathcal{N})$ nous pouvons définir un sous-RCQ de $\mathcal{N}$ $\circ$-fermé et défini sur $\mathcal{C}_{\preceq}$. Ce SRCQ est défini formellement de la manière suivante :

Définition 3
Soient $\mathcal{N} = (V, C)$ *un* SRCQ *et* I *un modèle satisfaisant* $\mathsf{Sat}(\mathcal{N})$. $\mathsf{srcq}(\mathsf{Sat}(\mathcal{N}))$ *est le* SRCQ (V, C') *défini par* $C'_{ij} = [l_{ij}, u_{ij}]$, *avec* $l_{ij} = \mathsf{Sup}\{a \in \mathsf{B} : I(a \preceq C_{ij}) = vrai\}$ *et* $u_{ij} = \mathsf{Inf}\{a \in \mathsf{B} : I(C_{ij} \preceq a) = vrai\}$, *pour tout* $i, j \in \{0, \ldots, |V| - 1\}$.

Notons que $\mathsf{srcq}(\mathsf{Sat}(\mathcal{N}))$ est bien défini car du fait de la satisfaction des clauses (IV) nous pouvons montrer que $l_{ij} = l_{ji}^{-1}$ et $u_{ij} = u_{ji}^{-1}$. De plus, nous avons $l_{ii} = u_{ii} = \mathsf{Id}$ car $\mathsf{Id} \preceq C_{ii}$ et $C_{ii} \preceq \mathsf{Id}$ appartiennent aux clauses (I). Ce SRCQ possède également les caractéristiques suivantes :

Proposition 3
Soient $\mathcal{N} = (V, C)$ *un* SRCQ *et* I *un modèle satisfaisant* $\mathsf{Sat}(\mathcal{N})$. *Nous avons :* (a) $\mathsf{srcq}(\mathsf{Sat}(\mathcal{N})) \subseteq N$ *et,* (b) $\mathsf{srcq}(\mathsf{Sat}(\mathcal{N}))$ *est un* RCQ $\circ$*-fermé.*

Preuve (a) Montrons que pour tout $i, j \in \{0, \ldots, n-1\}$, avec $n = |V|$, $a_{ij} \preceq l_{ij} \preceq u_{ij} \preceq b_{ij}$, avec $C_{ij} = [a_{ij}, b_{ij}]$ et $l_{ij} = \mathsf{Sup}\{a \in \mathsf{B} : I(a \preceq C_{ij}) = vrai\}$ et $u_{ij} = \mathsf{Inf}\{a \in \mathsf{B} : I(C_{ij} \preceq a) = vrai\}$. Du fait des clauses (I) nous savons que I satisfait $a_{ij} \preceq C_{ij}$ et $C_{ij} \preceq b_{ij}$. Par définition de l_{ij} et u_{ij} nous avons donc $a_{ij} \preceq l_{ij}$ et $u_{ij} \preceq b_{ij}$. Montrons que $l_{ij} \preceq u_{ij}$. I satisfait les clauses (II), nous pouvons en déduire que I satisfait $l_{ij} \preceq C_{ij}$ et $C_{ij} \preceq u_{ij}$. Dans le cas où $l_{ij} \not\preceq u_{ij}$ I ne satisferait pas les clauses (III). Il s'ensuit que $l_{ij} \preceq u_{ij}$. (b) Montrons que pour tout $i, j, k \in \{0, \ldots, n-1\}$, avec $i < j$, nous avons $[l_{ij}, u_{ij}] \subseteq [l_{ik}, u_{ik}] \circ [l_{kj}, u_{kj}]$. Nous savons que I satisfait les littéraux $l_{ik} \preceq C_{ik}$, $l_{kj} \preceq C_{kj}$, $C_{ik} \preceq u_{ik}$ et $C_{kj} \preceq u_{kj}$. Du fait que I satisfait les clauses (V) nous pouvons affirmer que $\mathsf{Inf}(l_{ik} \circ l_{kj}) \preceq C_{ij}$ et $C_{ij} \preceq \mathsf{Sup}(u_{ik} \circ u_{kj})$ sont deux littéraux satisfaits. Par définition de l_{ij} et u_{ij} nous en déduisons que $\mathsf{Inf}(l_{ik} \circ l_{kj}) \preceq l_{ij}$ et $u_{ij} \preceq \mathsf{Sup}(u_{ik} \circ u_{kj})$. Comme $[l_{ik}, u_{ik}] \circ [l_{kj}, u_{kj}] = [\mathsf{Inf}(l_{ik} \circ l_{kj}), \mathsf{Sup}(u_{ik} \circ u_{kj})]$, nous avons $[l_{ij}, u_{ij}] \subseteq [l_{ik}, u_{ik}] \circ [l_{kj}, u_{kj}]$. $\mathsf{srcq}(\mathsf{Sat}(\mathcal{N}))$ est donc un RCQ $\circ$-fermé. $\dashv$

Proposition 4
Soit $\mathcal{N} = (V, C)$ *un* SRCQ*. Si la méthode de la fermeture par faible composition est complète pour les* SRCQ *alors* $\mathcal{N}$ *est cohérent ssi* $\mathsf{Sat}(\mathcal{N})$ *est satisfiable.*

Preuve $-$ Supposons que $\mathcal{N}$ soit cohérent. Il existe un scénario de $\mathcal{N}$ cohérent. Ce scénario est fermé par faible composition. D'après la proposition 2, nous pouvons affirmer que $\mathsf{Sat}(\mathcal{N})$ admet un modèle. $-$ Supposons que $\mathsf{Sat}(\mathcal{N})$ admet un modèle I. D'après la proposition 3 nous pouvons affirmer que $\mathsf{srcq}(\mathsf{Sat}(\mathcal{N}))$ est un sous-RCQ de $\mathcal{N}$ $\circ$-fermé. $\mathsf{srcq}(\mathsf{Sat}(\mathcal{N}))$ est donc cohérent. Il s'ensuit que $\mathcal{N}$ est cohérent. $\dashv$
Nous pouvons étendre ce théorème au cas général des RCQ :

Théorème 1
Soit $\mathcal{N} = (V, C)$ *un* RCQ*. Si la méthode de la fermeture par faible composition est complète pour les* SRCQ *alors* $\mathcal{N}$ *est cohérent ssi* $\mathsf{Sat}(\mathcal{N})$ *est satisfiable.*

Preuve Rappelons nous que la traduction SAT d'un RCQ $\mathcal{N}$ est basée sur un découpage en relations de $\mathcal{C}_{\preceq}$ de chacune de ses contraintes : $C_{ij} = [a^1_{ij}, b^1_{ij}] \cup \ldots \cup [a^{k_{ij}}_{ij}, b^{k_{ij}}_{ij}]$ avec $a^1_{ij}, \ldots, a^{k_{ij}}_{ij}, b^1_{ij}, \ldots, b^{k_{ij}}_{ij} \in \mathsf{B}$ et k_{ij} un entier. $-$ Si $\mathcal{N}$ est cohérent alors il admet un sous-RCQ $\mathcal{N}' = (V, C')$ cohérent dont chaque contrainte C'_{ij} est définie par une relation $[a^l_{ij}, b^l_{ij}]$ avec $l \in \{1, \ldots, k_{ij}\}$ issue du découpage de C_{ij}. De la proposition précédente nous savons que $\mathsf{Sat}(\mathcal{N}')$ admet un modèle. En examinant la définition de $\mathsf{Sat}(\mathcal{N})$ on peut affirmer que ce modèle satisfait également $\mathsf{Sat}(\mathcal{N})$. $-$ Soit un modèle I satisfaisant $\mathsf{Sat}(\mathcal{N})$. Nous savons que pour chaque contrainte C_{ij}, il existe un entier $k \in \{1, \ldots, k_{ij}\}$ tel que I satisfait la conjonction $a^k_{ij} \preceq C_{ij} \wedge b^k_{ij} \preceq C_{ij}$. En définissant $\mathcal{N}' = (V, C')$ par $C'_{ij} = [a^k_{ij}, b^k_{ij}]$, nous obtenons un SRCQ sous-RCQ de $\mathcal{N}$ dont la traduction SAT $\mathsf{Sat}(\mathcal{N}')$ est satisfaite par I. Par conséquent, d'après la proposition précédente, $\mathcal{N}'$ est cohérent. Il s'ensuit que $\mathcal{N}$ est également cohérent. $\dashv$

5 Discussion concernant les traductions SAT existantes

La traduction existante [7, 6] la plus "naturelle" exploite le fait que dans la plupart des formalismes, un scénario $\circ$-fermé est cohérent. Elle est définie de la manière suivante :

Définition 4 (Support encoding)
Soit $\mathcal{N} = (V, C)$ *un* RCQ *tel que* $V = \{v_0, ..., v_{n-1}\}$. $\mathsf{Sat}_S(\mathcal{N})$ *est l'ensemble de clauses défini sur l'ensemble de propositions* $\{r_{ij}$ *avec* $a \in \mathsf{B}, i, j \in \{0, \ldots, n-1\}$ *et* $i < j\}$ *par :*

1. *pour tout* $1 \leq i < j \leq n$*, la clause* $\bigvee_{a \in C_{ij}} a_{ij}$ *est introduite (ALO),*
2. *pour tout* $1 \leq i < j \leq n$ *et pour tout* $a, b \in C_{ij}$*, si* $a \neq b$*, la clause* $\neg a_{ij} \vee \neg b_{ij}$ *est introduite (AMO)*
3. *pour tout* $1 \leq i < k < j \leq n$*, pour tout* $a \in C_{ik}$*,* $b \in C_{kj}$*, la clause* $\neg a_{ik} \vee \neg b_{kj} \vee \bigvee_{c \in (a \circ b) \cap C_{ij}} c$*, est introduite (Support).*

Intuitivement, les clauses (ALO) et (AMO) font que tout modèle "satisfera" une et une seule relation de base a_{ij} par contrainte C_{ij}.Les clauses (Support) permettent de spécifier tous les triplets de relations de base possibles à l'aide de la table de composition. On peut prouver que si la méthode de la fermeture par faible composition est complète pour les scénarios alors $\mathsf{Sat}_S(\mathcal{N})$ est satisfiable ssi $\mathcal{N}$ est cohérent. En comparant Sat et Sat_S, le premier constat qu'il est possible de faire concerne les clauses de Horn. En effet, Sat garantit à toutes les clauses d'être de Horn dans le cas où le RCQ traduit est un SRCQ. Ce n'est pas le cas de la traduction Sat_S. À des fins d'illustration considérons de nouveau le SRCQ de la figure 2. La clause (ALO) associée à C_{01} sera définie par $b_{01} \vee m_{01} \vee o_{01}$. L'une des 3 relations b, m, o dans C_{01}. Les clauses (AMO) associées à C_{01} sont au nombre de 3 : $\neg b_{01} \vee \neg m_{01}$, $\neg b_{01} \vee \neg o_{01}$ et $\neg m_{01} \vee \neg o_{01}$. Enfin, pour représenter le fait que $m \circ o = \{o, s, d\}$ pour les contraintes C_{01}, C_{12}, C_{23} par exemple, on introduit la clause (Support) $\neg m_{01} \vee \neg o_{12} \vee o_{02} \vee s_{02} \vee d_{02}$. On constate que la plupart de ces clauses ne sont pas de Horn, ce qui peut être aussi le cas des clauses de Sat, dans le cas d'une traduction des RCQ. Le nombre de littéraux des clauses (I) dépendra du découpage en relations de $\mathcal{C}_{\preceq}$. Dans le cas d'Allen, en moyenne, ce nombre est plus faible que la taille des clauses (ALO). La représentation de la relation totale est la relation pour laquelle la différence est la plus extrême. De loin, l'ensemble de clauses le plus important résultant des deux traductions correspond aux clauses représentant la composition des relations de base : l'ensemble de clauses (V) pour Sat et l'ensemble de clauses (Support) pour Sat_S. En prenant l'exemple d'un RCQ de 50 variables, le nombre de clauses (V) obtenu par Sat est de l'ordre de la dizaine de millions, tandis que dans le cas des clauses (Support) obtenues par Sat_S, le nombre de clauses est 3 fois moindre. Ceci est dû notamment à la définition des clauses (Support) qui se contente d'un ordre strict sur les indices des contraintes considérées. Une étude théorique encore approfondie doit aussi permettre de pouvoir réduire les clauses (V).

6 Conclusion et travaux futurs

Dans ce papier nous avons défini un codage des réseaux de contraintes qualitatives en logique propositionnelle utilisant une structure correspondant à un treillis particulier sur les relations de base. Dans le cas particulier des SRCQ nous obtenons un ensemble de clauses de Horn. Nous avons prouvé que le problème de satisfiabilité de l'ensemble des clauses résultant de cette traduction permet de résoudre le problème de la cohérence des RCQ traduits. Des expériences sont en cours pour comparer les différentes traductions SAT existantes et cette nouvelle traduction. Notre objectif est de comparer d'une part les tailles des ensembles de clauses obtenues par ces traductions, et d'autre part nous souhaitons comparer le temps nécessaire à différents solveurs SAT pour résoudre ces ensembles de clauses. La traduction présentée dans ce papier s'inspire de propriétés des relations dites convexes. Une perspective de recherche est de définir une nouvelle traduction s'inspirant de propriétés des relations dites préconvexes qui correspondent à un sur-ensemble traitable de l'ensemble des relations convexes.

Références

[1] J. F. Allen. An interval-based representation of temporal knowledge. In *Proc. of the Seventh Int. Joint Conf. on Artificial Intelligence (IJCAI'81)*, pages 221–226, 1981.

[2] P. Balbiani, J.-F. Condotta, and L. Fariñas del Cerro. Spatial reasoning about points in a multidimensional setting. In *Proc. of the work. on temp. and spatial reasoning (IJCAI'99)*, pages 105–113, 1999.

[3] Philippe Balbiani, Jean-François Condotta, and Luis Fariñas del Cerro. A new tractable subclass of the rectangle algebra. In T. Dean, editor, *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI'99)*, pages 442–447, 1999.

[4] Peter Van Beek and Dennis W. Manchak. The design and experimental analysis of algorithms for temporal reasoning. *Journal of Artificial Intelligence Research*, 4 :1–18, 1996.

[5] J.-F. Condotta, G. Ligozat, and M. Saade. An empirical study of algorithms for qualitative temporal or spatial networks. In *Proceedings of the workshop on spatial reasoning (ECAI'06)*, 2006.

[6] Dominique D'Almeida. Résolution de CSP qualitatifs par l'utilisation de CSP discrets et de SAT. Technical report, Rapport de Master I, Université d'Artois, CRIL-CNRS, 2007.

[7] Abdul Sattar Duc Nghia Pham and John Thornton. Towards an efficient sat encoding for temporal reasoning. In F. Benhamou, editor, *Proceedings of the 12th International Conference on Principles and Practice of Constraint Programming (CP'06)*, volume Lecture Notes in Computer Science 4204, pages 421–436, Nantes, France, 2006.

[8] Gérard Ligozat. Reasoning about cardinal directions. *Journal of Visual Languages and Computing*, 1(9) :23–44, 1998.

[9] A. K. Mackworth and E. C. Freuder. The Complexity of Some Polynomial Network Consistency Algorithms for Constraint Satisfaction Problem. *Artificial Intelligence*, 25(1) :65–74, 1985.

[10] U. Montanari. Networks of constraints : Fundamental properties and application to picture processing. *Information Sciences*, 7(2) :95–132, 1974.

[11] Bernhard Nebel. Solving hard qualitative temporal reasoning problems : Evaluating the efficienty of using the ord-horn class. In *Proc. of the Twelfth Conference on Artificial Intelligence (ECAI'96)*, 1996.

[12] Bernhard Nebel and Hans-Jürgen Bürckert. Reasoning About Temporal Relations : A Maximal Tractable Subclass of Allen's Interval Algebra. *Journal of the ACM*, 42(1) :43–66, 1995.

[13] K. Nökel. Temporally distributed symptoms in technical diagnosis. *LNCS*, 517 :1–184, 1991.

[14] Arun K. Pujari, G. Vijaya Kumari, and Abdul Sattar. Indu : An interval and duration network. In *Australian Joint Conference on Artificial Intelligence*, pages 291–303, 1999.

[15] D. A. Randell, Z. Cui, and A. G. Cohn. A spatial logic based on regions and connection. In *Proc. of the 3rd Conf. on Principles of Knowledge Representation and Reasoning (KR'92)*, pages 165–176, 1992.

Arguing about potential causal relations

Leila Amgoud [1] et Henri Prade[1],

[1] Institut de Recherche d'Informatique de Toulouse, CNRS
{amgoud, prade}@irit.fr
Université Paul Sabatier, 118 route de Narbonne,
31062 Toulouse cedex 09
http://www.irit.fr

Abstract: This short note aims at providing a general discussion about patterns of reasoning, and their argumentative counterparts, involved in the way human agents ascribe causality or reason from causal relations. This study is mainly motivated by the need for providing information systems users with appropriate explanations, according to the way these agents are likely to perceive and understand a reported situation.

Key words: Causality, explanation, argumentation.

1 Introduction

Causality is pervasive in the way humans try to understand and make sense of the world. This paper does not aim at discussing the philosophical underpinnings of the idea of causality, but more simply to lay bare different inference patterns that may be used for ascribing causal relations between reported events, or for reasoning about the expected consequences of known facts and given causal relations, be them hypothetical. Indeed, when a sequence of events takes place, a natural question is to understand what has caused what. Such a study is motivated by the need to be able in future information handling systems to predict how agents perceive and understand reported information and what kind of explanation it may be appropriate to give them.

Explanations have usually an argumentative structure, which may be often challenged by counter-arguments questioning the basis of the explanation. The arguments in favor or against a potential causal explanation clearly depend on the knowledge that the agent is supposed to have about the normal course of things. Moreover, argumentation in causal settings may refer to different issues, such as the apparent incompleteness or imprecision of the reported sequence of events, or the inconsistency of the agent's beliefs about the normal course of things with what is reported. Argumentation may also involve hypothetical reasoning, and even include the use of unreal conditionals, such as "interventions" (see Woodward, 2001; Pearl, 2000) aiming at distinguishing causality from correlation. However, interventions are not considered in the following and left for further research. Lastly, argumentation may also take an analogical form when situations are encountered that are similar to other situations that already took place where cause(s) was/were identified.

This discussion paper is organized as follows. The next section, Section 2, provides a background about different patterns of reasoning involving causality and their argumentative counterparts. In particular, the assessment of causality relations on the basis of beliefs on the normal course of things is considered. Section 3 examines different scenarios, where according to what is reported and what are the beliefs of the agent, the agent may have different perceptions of the information. Section 4 discusses the handling of causal arguments and their strength. The concluding remarks briefly consider more sophisticated patterns with hypothetical conditionals that pertain to the ascription of responsibility.

2 Background: Schema for reasoning about or assessing causality

When reasoning about causality, one should distinguish between different types of problems:

- i) generic causal relations are given, and we apply them to a particular situation in order to predict what is going to take place;
- ii) generic causal relations are given and we use them for diagnosing plausible causes that may have led to the observed acts;
- iii) given a reported ordered sequences of facts (including potential causes and effects), and given generic knowledge about the (normal) course of the world, the causal relation(s) that took place between the reported facts have to be identified;
- iv) from past experience, viewed as a set of reported ordered sequences of facts with identified causal relations, try to guess the causality relations that took place in a new reported sequence of facts, on the basis of similarity relations;
- v) from a sufficiently large set of reported sequences, try to learn generic causal relations.

Case (i) is just deductive causal reasoning, and is exemplified by the pattern

(generally) A causes B
A is true

B should be true (and might be expected to be reported as such)

As any deductive pattern, the set of the two premises constitute an argument in order to conclude that B is true. However, if the first premise has exceptions, one may face non-monotonic reasoning behaviors, namely "special situations where A is true may not cause B".

Case (ii) correspond to the pattern

(generally) A causes B
B is true

A might be true

This is an abductive pattern. Note that in case one knows several A_i's such that 'A_i causes B' (each A_i stands for a conjunction of elementary facts describing a situation), one would conclude that A_1 or… A_i or … might be true. Case (ii) can be turned into a deductive manner by changing the first premise into "B evokes A as a potential cause". Then, turning patterns (i) (or (ii) once modified) in an argumentative manner is straightforward, since the two premises of the pattern constitutes a minimal consistent subset that supports the conclusion.

Case (iv) and case (v) are respectively analogical and inductive reasoning. Case (v) is out of the scope of the paper. Case (iv) resembles case (iii) replacing generic knowledge by case-based knowledge.

Case (iii) should not be confused with case (ii). In case (iii), the causal relation(s), if any, is/are to be identified between reported facts. It is not so much a matter of finding out unreported facts (such as events that may have taken place and caused the observed – and reported – facts), as in (ii).

In case (iii), it is assumed that one is in a (reported) context C (C represents the partial available knowledge about the context). Moreover, it is supposed that a sequence such as Bt, At, ¬Bt' is reported, where t' denotes a time instant strictly after t (Bt means that B is reported true at time t). Besides, the agent that receives this sequential information has some knowledge on what is the normal course of the world in context C, and maybe also in context $C \wedge A$, regarding B. Namely, the agent may either believe that $C \mid\approx B$ (B is expected to be true), or that $C \mid\approx \neg B$ (B is expected to be false), or that $C \not\mid\approx B$ and $C \not\mid\approx \neg B$ (the truth or the falsity of B is contingent), where $\mid\approx$ is a non-monotonic consequence relation describing what is normal, and $\not\mid\approx$ stands for its negation. Similarly, in context $C \wedge A$, the agent may have the same form of belief. It is assumed, that $C \wedge A$ is consistent (otherwise, one would have to take into account that the fact that A becomes true should modify C into a known way C'). Note also that 'A true' may as well recover an action that has been executed and whose execution is finished (e.g. the driver drank before taking his car), or something that has become true and that remains true (e.g., the driver is inebriated).

In such a situation, the two following definitions have been recently introduced in (Bonnefon et al., 2006).

Definitions 1 (Facilitation and Causation) Let us assume that an agent learns of the sequence Bt, At, ¬Bt'. Let us call C (the context) the conjunction of all other facts known by, or reported to, the agent at time t' > t. Given a nonmonotonic consequence relation $\mid\approx$, if the agent believes that $C \mid\approx B$, and that $C \wedge A \not\mid\approx B$ (resp. $C \wedge A \mid\approx \neg B$), the agent will perceive A as having facilitated the occurrence of (resp. as being the cause of) ¬B in context C, noted $C : A \Rightarrow_{fa} \neg B$ (resp. $C : A \Rightarrow_{ca} \neg B$).

In (Bonnefon et al., 2006) experiments are reported, which tend to indicate the cognitive validity of the two above notions. Moreover, the properties of these definitions are studied in detail, and in particular, it is shown that

- If C: A $\Rightarrow$ca B or if C: A $\Rightarrow$fa B then C $|\approx \neg$A.
- a restricted transitivity property holds: If C: A $\Rightarrow$ca B, if C: B $\Rightarrow$ca D and if B $\wedge$ C $|\approx$ A then C: A $\Rightarrow$ca D. The two properties hold for $\Rightarrow$ca provided that $|\approx$ is a preferential entailment in the sense of (Lehmann and Magidor, 1992). The first property holds for facilitation ($\Rightarrow$fa) if $|\approx$ is a rational closure entailment.

Note that here transitivity requires B $\wedge$ C $|\approx$ A, i. e. A is not too specific with respect to B (it means that the normal way to have B (in context C), is to have A). For instance, for A = drinking, B = inebriated, D: staggering, we have 'drinking' $\Rightarrow$ca 'inebriated' and 'inebriated' $\Rightarrow$ca staggering' entail 'drinking' $\Rightarrow$ca 'staggering', since 'inebriated' $|\approx$ 'drinking'.

Note also that the causation definition can be presented in an argumentative manner (it would be the same for facilitation):
- at time t we were in context C and B was true,
- it is normal in context C that B be true and B should persist,
- at time t, C has been modified by the fact that A took place,
- normally in context C $\wedge$ A, B is false,
- indeed at time t' > t, B is reported to be false,

Then, an agent sharing the beliefs C $|\approx$ B and C $\wedge$ A $|\approx \neg$B, can argue that A has caused the fact that B has become false (in the absence of any other change reported in context C), or that B has become false, because A took place, otherwise if A had not taken place B would have persisted to be true (counterfactual).

Lastly, (Bonnefon et al., 2006) have also introduced the idea of justification, which corresponds for the agent to an epistemic state different from those in the above definitions. Namely, the idea of justification corresponds to the following patterns.

Definition 2 ("Justification" or Explanation) Let us assume that an agent learns of the sequence B_t, A_t, $\neg B_{t'}$. Let us call C (the context) the conjunction of all other facts known by or reported to the agent at time t' > t. If the agent believes C $|\not\approx$ B, C $|\not\approx \neg$B and C $\wedge$ A $|\approx \neg$B, given a non-monotonic consequence relation $|\approx$, he will perceive A as justifying the fact that B is now false in context C.

3 Different possible scenarios

As already said, human agents when they are faced to reports try to make sense of them on the basis of their own beliefs. Depending on the cases, the agent may have the feeling to understand what took place, or to miss some important piece of information, or even to be puzzled. In the following, we examine different possible types of scenarios, extending the setting underlying Definitions 1 and 2. First, these scenarios include one of the four possible (sub)-sequences of reported facts:

- C, Bt, ¬Bt', corresponding to a change without reported event
- C, Bt, Bt', corresponding to persistence without reported event
- C, Bt, At, ¬Bt', corresponding to a change with reported event
- C, Bt, At, Bt', corresponding to persistence with reported event

and are also characterized by the two following possible pieces of beliefs supposed to belong to the (supposedly consistent) epistemic state of the agent, in context :

- either $C \mid\approx B$, or $C \mid\approx \neg B$, or $C \not\mid\approx B$ and $C \not\mid\approx \neg B$;
- either $C \wedge A \mid\approx B$, or $C \wedge A \mid\approx \neg B$, or $C \wedge A \not\mid\approx B$ and $C \wedge A \not\mid\approx \neg B$.

Note that the case of believing both $C \mid\approx B$ and $C \mid\approx \neg B$ would be inconsistent with the hypothesis that $\mid\approx$ is a preferential entailment. Besides, if the agent has the default rule $C \wedge A \mid\approx B$ (or one of the other ones pertaining to A) in his epistemic state, and if At is *not* reported, it should be understood as the agent knows about some A such that $A \wedge C \mid\approx B$; obviously he may also know about some A' such that $A' \wedge C \mid\approx B$, or such that $A' \wedge C \mid\approx \neg B$, but this will be covered by another elementary scenario.

All together, we thus have 4×3×3 = 36 elementary scenarios, reviewed in Table 1. It is also assumed that B persists in context C.

Table 1. Epistemic states and perception of scenarios

1	C, Bt, At, ¬Bt'	$C \mid\approx B$	$C \wedge A \mid\approx B$	unexplained change, B should have persisted
2	C, Bt, At, ¬Bt'	$C \mid\approx B$	$C \wedge A \not\mid\approx B$ and $C \wedge A \not\mid\approx \neg B$	change *facilitated* by A
3	C, Bt, At, ¬Bt'	$C \mid\approx B$	$C \wedge A \mid\approx \neg B$	change *caused* by A
4	C, Bt, At, ¬Bt'	$C \not\mid\approx B$ and $C \not\mid\approx \neg B$	$C \wedge A \mid\approx B$	unjustified change after A
5	C, Bt, At, ¬Bt'	$C \not\mid\approx B$ and $C \not\mid\approx \neg B$	$C \wedge A \not\mid\approx B$ and $C \wedge A \not\mid\approx \neg B$	contingent change
6	C, Bt, At, ¬Bt'	$C \not\mid\approx B$ and $C \not\mid\approx \neg B$	$C \wedge A \mid\approx \neg B$	change *justified* by A
7	C, Bt, At, ¬Bt'	$C \mid\approx \neg B$	$C \wedge A \mid\approx B$	unexplained change, double defeated expectations!
8	C, Bt, At, ¬Bt'	$C \mid\approx \neg B$	$C \wedge A \not\mid\approx B$ and $C \wedge A \not\mid\approx \neg B$	from exceptionality to contingency
9	C, Bt, At, ¬Bt'	$C \mid\approx \neg B$	$C \wedge A \mid\approx \neg B$	back to normality thanks to A
10	C, Bt, ¬Bt'	$C \mid\approx B$	$C \wedge A \mid\approx B$	change for unknown reason
11	C, Bt, ¬Bt'	$C \mid\approx B$	$C \wedge A \not\mid\approx B$ and $C \wedge A \not\mid\approx \neg B$	change for unknown reason, A is a potential facilitating factor
12	C, Bt, ¬Bt'	$C \mid\approx B$	$C \wedge A \mid\approx \neg B$	A is a potential *cause* for the change

13	C, Bt, ¬Bt'	$C \not\mid\!\approx B$ and $C \not\mid\!\approx \neg B$	$C \wedge A \mid\!\approx B$	unexplainable change
14	C, Bt, ¬Bt'	$C \not\mid\!\approx B$ and $C \not\mid\!\approx \neg B$	$C \wedge A \not\mid\!\approx B$ and $C \wedge A \not\mid\!\approx \neg B$	fully contingent change
15	C, Bt, ¬Bt'	$C \not\mid\!\approx B$ and $C \not\mid\!\approx \neg B$	$C \wedge A \mid\!\approx \neg B$	A would *justify* the change
16	C, Bt, ¬Bt'	$C \mid\!\approx \neg B$	$C \wedge A \mid\!\approx B$	back to normality (not due to A)
17	C, Bt, ¬Bt'	$C \mid\!\approx \neg B$	$C \wedge A \not\mid\!\approx B$ and $C \wedge A \not\mid\!\approx \neg B$	back to normality, (could have been facilitated by A)
18	C, Bt, ¬Bt'	$C \mid\!\approx \neg B$	$C \wedge A \mid\!\approx \neg B$	back to normality (maybe due to A)
19	C, Bt, At, Bt'	$C \mid\!\approx B$	$C \wedge A \mid\!\approx B$	A agrees with persistence of B
20	C, Bt, At, Bt'	$C \mid\!\approx B$	$C \wedge A \not\mid\!\approx B$ and $C \wedge A \not\mid\!\approx \neg B$	B has persisted in spite of A
21	C, Bt, At, Bt'	$C \mid\!\approx B$	$C \wedge A \mid\!\approx \neg B$	unexplained persistence of B
22	C, Bt, At, Bt'	$C \not\mid\!\approx B$ and $C \not\mid\!\approx \neg B$	$C \wedge A \mid\!\approx B$	A explains persistence of B
23	C, Bt, At, Bt'	$C \not\mid\!\approx B$ and $C \not\mid\!\approx \neg B$	$C \wedge A \not\mid\!\approx B$ and $C \wedge A \not\mid\!\approx \neg B$	contingent persistence of B
24	C, Bt, At, Bt'	$C \not\mid\!\approx B$ and $C \not\mid\!\approx \neg B$	$C \wedge A \mid\!\approx \neg B$	A disagrees with persistence of B
25	C, Bt, At, Bt'	$C \mid\!\approx \neg B$	$C \wedge A \mid\!\approx B$	back to normality
26	C, Bt, At, Bt'	$C \mid\!\approx \neg B$	$C \wedge A \not\mid\!\approx B$ and $C \wedge A \not\mid\!\approx \neg B$	from exception to contingency
27	C, Bt, At, Bt'	$C \mid\!\approx \neg B$	$C \wedge A \mid\!\approx \neg B$	double defeated expectations, exceptional situation
28	C, Bt, Bt'	$C \mid\!\approx B$	$C \wedge A \mid\!\approx B$	expected persistence
29	C, Bt, Bt'	$C \mid\!\approx B$	$C \wedge A \not\mid\!\approx B$ and $C \wedge A \not\mid\!\approx \neg B$	expected persistence
30	C, Bt, Bt'	$C \mid\!\approx B$	$C \wedge A \mid\!\approx \neg B$	expected persistence
31	C, Bt, Bt'	$C \not\mid\!\approx B$ and $C \not\mid\!\approx \neg B$	$C \wedge A \mid\!\approx B$	contingent persistence
32	C, Bt, Bt'	$C \not\mid\!\approx B$ and $C \not\mid\!\approx \neg B$	$C \wedge A \not\mid\!\approx B$ and $C \wedge A \not\mid\!\approx \neg B$	contingent persistence
33	C, Bt, Bt'	$C \not\mid\!\approx B$ and $C \not\mid\!\approx \neg B$	$C \wedge A \mid\!\approx \neg B$	contingent persistence
34	C, Bt, Bt'	$C \mid\!\approx \neg B$	$C \wedge A \mid\!\approx B$	from exception to normality in case A took place
35	C, Bt, Bt'	$C \mid\!\approx \neg B$	$C \wedge A \not\mid\!\approx B$ and $C \wedge A \not\mid\!\approx \neg B$	persistence of exceptionality, might be facilitated to A
36	C, Bt, Bt'	$C \mid\!\approx \neg B$	$C \wedge A \mid\!\approx \neg B$	persistence of exceptionality

Table 1 briefly outlines the different perceptions that an agent may have about a reported scenario (possibly incompletely stated) depending on what he believes to be the normal course of things in different contexts, and thus what should be expected according to him. In case no particular event A is mentioned in the report, the agent may try to diagnose some A that would facilitate, cause, or justify that B is true or is false in t'.

However, agents may start to argue in favor of causality on a basis slightly weaker than assumed in Definitions 1 and 2, namely only on the basis of a reported modification of context (from C to C∧A), B becoming false, while the agent believes that C $|\approx$ B, (but not necessarily that C∧A $|\not\approx$ B, or that C∧A $|\approx \neg$B). Indeed, argumentation is a dynamical process where arguments and counter-arguments may interact with each other in order to assess a given claim (here, a possible cause). That's why in an argumentation approach a weak view of causality is used. As we shall see, the different cases appearing in Table 1 may give birth to different forms of counter-arguments.

4 Causal Argument

Causal arguments are a particular kind of arguments. They underlie two of the most common, challenging, and difficult questions we confront in our lives: "Why?" and "What if?" When historians debate the causes of a war, when environmentalists speculate on the effects of pollution, and when psychologists study the effects of racism, they are working in the realm of causal argument. That is, they are examining the complex process by which people, forces, events, and other phenomena interact to bring about other phenomena. Although some people may speak of *proving* a causal connection between two things, causal argument is by its very nature highly speculative and prone to mistakes. Part of the difficulty, as any scientist can attest, lies in isolating variables. In other words, when examining the many factors that may have caused an event to occur or the many effects that may be traced back to a cause, we must be careful to determine exactly which ones really are valid. Take, for example, the apparently simple case of the American Civil War. Anyone who has studied this conflict knows that slavery was an important issue that divided the northern and southern states. In the three decades preceding the Civil War, however, America also was experiencing a number of other important phenomena: social upheaval, migration and immigration, technological changes, and even an economic crisis. How can we prove that it was slavery and not one of these other factors that caused the war? The answer is that we can't. Indeed, as in other kinds of arguments, we rarely can prove our causal claims definitively.

Like inductive or analogical reasoning, these causal arguments are based on observed instances. They follow the form of an analogical argument up to one important point: whereas this type of argument carries as part of its premises the assumption that there is no significant difference between previously observed cases and what is known of the situation at hand, causal arguments rather heavily relies on the existence of a particular difference altering the current context and perceived as significant. More precisely, a causal argument is defined in the following way:

Definition 3 (Causal argument scheme)
A "argumentedly" *caused* ¬ B because:

a. Normally, in context C one has B that is true (i.e. C $|\approx$ B)
b. The actual context is C' = C ∧ A (assuming consistency of C and A)
c. In the new context C', ¬B is reported as true.

A is said to be the *relevant or (significant) difference* between the two contexts C and C'. A *caused* $\neg$B is the *conclusion* of the argument, and points a, b and c constitute the *support* of the argument.

Definition 3 is exemplified by the following example.
Example 1. A bicyclist moves into the traffic lane in order to pass a truck illegally parked in the bike lane. The driver of a car approaching from the rear slams on her brakes in order to avoid hitting the bicycle. A following car fails to stop in time, and smashes into the back of the first.
The bicyclist's insurance company will probably claim that the illegally parked truck caused her client to swerve into the lane of traffic, using the following argument:
Let s = to swerve into the lane of traffic, i = illegally parked truck.

Argument A_1: i caused s because:

a. $C \mid\approx \neg s$
b. $C' = C \wedge i$
c. s is true.

As it is well-known in argumentation theory, an argument provides a reason for its conclusion. However, this does not necessarily mean that the conclusion is true. For that purpose, the argument needs to be defended against all its counter-arguments (called also defeaters). In our particular case of causal arguments, defeaters may be built by answering the following critical questions:

I. Does it hold that $C \mid\approx B$? Are there cases where $C \wedge \neg B$ holds?
II. Is it really the case that $\neg B$ is true?
III. Is there another A' such that both $C \wedge A'$ and $\neg B$ hold?
IV. Is the difference A pointed out between contexts C and C' relevant (w. r. t. a possible change from B to $\neg B$)?
V. Does the possible cause A invariably, or at least generally, produce the effect $\neg B$?

It is worth noticing that answering the above questions amounts to exhibit one or several of the prototypical situations listed in Table 1. For instance, arguments against the causal argument can be built by answering the question 'Does it hold that $C \mid\approx B$?' (see point I above). This is the case if we are for instance in situations 7, 8, 9, 16, 17, 18, 25, 26, 27, 34, 35, 36 of Table 1. The point III above is illustrated by an agent who believes $C \mid\approx B$ and $C \wedge A \mid\approx B$, and thus expresses that he does not understand why the reported sequence C, Bt, At, $\neg$Bt' took place (case 1 in Table 1). Then pointing out that in fact we are in context $C \wedge A \wedge D$ and that $C \wedge A \wedge D \mid\approx \neg B$ holds, would for instance solve the case (case 3 of Table 1).

Let us further illustrate the above ideas through a toy example.

Example 2. Suppose that several persons all get sick after eating a pizza during a party organized by their friend Mary. Moreover, each of them had a fancy hat also.
Let p: eating the pizza; h: wearing a hat, s: being sick, pa: attending a party.

Mary tries to understand what causes the sickness of her friends. She builds the two following causal arguments:

Argument A_2: p caused s because:
pa $|\approx \neg$s (expressing that generally when we go to a party, we are not sick)
C' = pa $\wedge$ p
s is true.

Argument A_3: h caused s because:
pa $|\approx \neg$s
C' = pa $\wedge$ h
s is true.

Paul, who took part to the party, does not agree with the causal argument A_3 provided by Mary, and presents the following counter-argument that says that having a hat does not cause being sick. Indeed, this argument shows clearly that a part of the intended cause is not "relevant". This refers to the critical point IV.

Argument A_4: h $|\approx \neg$s

Note that the above argument (A_4) is not a causal argument. Indeed, causal arguments may be defeated by other causal arguments, or simply by classical explanatory arguments.
According to Dung (1995), the argument A_3 is defeated by A_4, which is undefeated. Thus, the causal argument A_3 of Mary will be rejected.

Suppose now that the friends learn that their fancy hats were treated by means of some toxic product (to). Then, one can build the following causal argument against A_4.

Argument A_5: to caused s because:
h $|\approx \neg$s
C' = h $\wedge$ to
s is true

It is clear that A_5 defeats the argument A_4, since A_5 shows a situation where one can wear a hat and get sick.

5 Concluding remarks

This paper is only intending to offer a preliminary discussion about two related issues that are both connected with the way agents may understand or not, maybe in a causal way or not, a reported sequence of events. The first issue was to figure out what may be the different types of reaction an agent may have in face of such a sequence, depending on his beliefs on the normal course of things, while the second issue was to discuss the form of causal arguments, where do they come from, and how they may be refuted. Moreover, other related issues such as the use of argumentation by an agent involved in reported events for presenting these events in a way advantageous for him (as, e. g., in the ongoing work by Boutouhami (2007)) have not been considered. Another important issue is the fact that Dung's acceptability semantics are not sufficient in case of causal arguments, although they are required for a proper development of an argumentative approach to causality assessment.

When arguing about changes that took place in scenarios of the form Bt, At, ¬Bt' (t' > t), and trying to look for responsibility (either for rewarding, or for suggesting guilt), one often uses patterns of hypothetical reasoning of the form: "If A' had taken place, ¬B would not have happened". This covers both situation where A' is an uncontrolled event, as in "if no storm had taken place, there would be no flood", or where A' is an action that had be performed by some agent, as in "if the driver had abstained drinking, he would not have got a fee". In the above two examples ¬B is something undesirable, and A' may be regarded as a cause for it. But similar patterns exist where ¬B is desirable, as in "if Peter had not received a solid education, he would have not succeeded", or in "if embankments had not be built, the flood would have not been avoided". Note that the condition part of the conditional statement may appear either in a positive or in a negative form, both when ¬B is desirable or not: one may say "if the driver had abstained drinking, …" or "if the driver had not drunk, …", as well as "if Peter had not received a solid education, …", or "if Peter had received a poor education, …", depending on what the arguer decides to emphasize. Apart from assessing if such conditionals hold or not, a clearly important issue for further research, when looking for responsibility, is to take into the feasibility, the cost, the permission status of doing, or not doing, A'.

6 Acknowledgements:

This work has been supported by a grant from the "Agence Nationale pour la Recherche (ANR)", project number NT05-3-44479 (project 'Micrac').

References

BONNEFON J.-F., DA SILVA NEVES R. M., DUBOIS D., & PRADE H. (2006). Background default knowledge and causality ascriptions, in: G. Brewka, S. Coradeschi, A. Perini, P. Traverso (Eds.), *Proc. of the 17th European Conference on Artificial Intelligence* (ECAI'06), Riva del Garda, Italy, Aug. 29 – Sept.1, IOS Press, Zurich, 2006, pp. 11–15.

BOUTOUHAMI S. (2007) D'une description objective à une description argumentée. Présentation au séminaire MICRAC, Lens, March 15-16, 2007.

DUNG P. M. (1995). On the acceptability of arguments and its fundamental role in nonmonotonic reasoning, logic programming and n-person games, *Artificial Intelligence*, 77, 321–358.

LEHMANN D. & MAGIDOR M. (1992). What does a conditional knowledge base entail?, *Artificial Intelligence*, 55, 1–60.

PEARL J. (2000). *Causality.* Cambridge University Press, New York

WOODWARD J., (2001). Causation and manipulability. In: *Stanford Encyclopedia of Philosophy*, (E. N. Zalta, Principal Ed.), 2001. http://plato.stanford.edu/entries/causation-mani/.