
Comparaison de graphes conceptuels dans le cadre de l'acquisition des connaissances à partir de multiples experts

R. Dieng, S. Labidi, S. Lapalut, P. Martin

INRIA, projet ACACIA

2004, route des Lucioles, BP 93

06902 Sophia Antipolis Cedex

France

Tél. : (+33) 93 65 78 10

Fax : (+33) 93 65 77 83

email : {dieng, labidi, lapalut, phmartin}@sophia.inria.fr

Résumé

Résumé : Nous utilisons des graphes de connaissances dont le formalisme de base est celui défini par Sowa. Ces graphes servent à représenter le niveau domaine d'un modèle d'expertise. Dans le cas de la multi-expertise, nous avons besoin d'outils pour comparer des graphes de plusieurs agents sur un même domaine de connaissance et de déterminer ainsi les connaissances communes et spécifiques des agents. A ces fins, nous sommes amenés à spécifier un algorithme de comparaison de graphes conceptuels dans une démarche en plusieurs phases pour traiter ces comparaisons. Les relations et opérateurs introduits par Sowa ne permettent pas la description de l'algorithme complet. Après avoir placé dans son contexte et explicité l'algorithme dans son état actuel, nous proposons la définition de quatre opérateurs sur les ensembles partiellement ordonnés utilisés dans le formalisme de Sowa.

Mots clés : Acquisition des connaissances, multi-expertise, graphes conceptuels, graphes de connaissances, posets.

1 Introduction

Ce papier relate des travaux en cours. Nous exposons ici nos objectifs dans le cadre de notre thème de recherche principal : l'acquisition de connaissances. Comme nous allons le voir dans une première partie, nous avons choisi le formalisme des graphes conceptuels pour représenter des connaissances sur un domaine d'expertise. Nous nous intéressons actuellement à la comparaison de tel graphe.

La comparaison des graphes conceptuels a été diversement abordée par de nombreux travaux ([7], [10], «graph matching» in [11]). Un algorithme traitant ce problème général n'existe pas encore. Sous certaines hypothèses, des algorithmes particuliers ont été établis [11]. Nous présentons dans ce travail une démarche pour la comparaison et l'intégration de graphes conceptuels dans le cadre de l'acquisition des connaissances à partir de multiples experts. Ces graphes conceptuels relatent des connaissances exprimées par des experts différents. Nous cherchons à détecter les parties communes et à résoudre les conflits éventuels qui apparaissent entre leurs connaissances, c'est-à-dire leurs graphes respectifs concernant un même domaine. Après avoir fixé le contexte dans une première partie, nous présentons notre démarche en trois étapes d'unification dans la deuxième partie.

La dernière partie traite de l'étape de comparaison des graphes. Nous précisons les opérateurs utilisés et nous proposons un algorithme basé sur la décomposition des graphes en un ensemble de triplets de la forme [concept]→(relation)→[concept]. La comparaison portera sur ces triplets, traités élément par élément. La détection et la gestion des conflits et des incompatibilités éventuelles se basent sur des stratégies préétablies.

Nous illustrons notre exposé par des exemples tirés d'une application sur laquelle nous travaillons et qui concerne la modélisation d'expertise en accidentologie [1].

2 Contexte

Pour le développement d'un système à base de connaissances il est parfois intéressant, voire obligatoire, de recourir à de multiples experts pour la phase d'acquisition des connaissances comprenant l'élicitation et la modélisation de la connaissance. Il peut s'agir d'experts travaillant sur le même domaine et résolvant le même type de problème en utilisant, par exemple, des méthodes de résolution différentes, ou de spécialistes avec divers domaines de compétence résolvant chacun une partie d'un même problème.

La prise en compte de la multi-expertise dans les projets d'intelligence artificielle a montré l'insuffisance des méthodes et outils classiques d'acquisition de connaissances. Dans ce cadre, nous travaillons actuellement sur la conception d'un outil d'aide à l'acquisition des connaissances à partir de multiples experts [6]. Nous avons spécifié auparavant une méthode d'acquisition basée sur un modèle d'agent cognitif [4,5]. Ce modèle d'agent permet de décrire le comportement de l'ensemble des experts ainsi que leur processus de coopération et servira donc de guide méthodologique au cogniticien lors de la phase d'acquisition des connaissances.

Parmi les caractéristiques servant à décrire un agent on peut citer les ressources de l'agent. C'est une caractéristique très importante qui représente les moyens qu'utilise l'agent pour raisonner ou pour agir. On y trouve en particulier les connaissances d'expertise ou ressources cognitives. Nous adoptons le modèle en couches de KADS [13] pour la modélisation

des connaissances d'un agent (cf. figure 1)

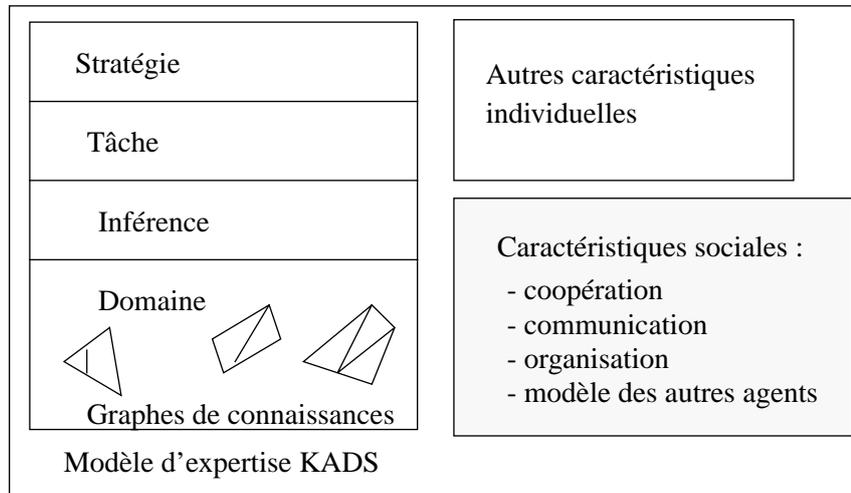


Fig.1: Notre modèle d'agent.

Ainsi, on distingue quatre niveaux dans le modèle d'expertise d'un agent : le niveau domaine, le niveau inférence, le niveau tâche et le niveau stratégie. Le niveau domaine représente les connaissances statiques de l'agent. Pour sa représentation nous utilisons le formalisme des graphes conceptuels de Sowa [14], adéquat pour la définition du niveau domaine [13]. A un agent est associé un treillis de concepts, une hiérarchie de relations et un ensemble de graphes conceptuels.

La comparaison de deux modèles d'expertise s'effectue niveau par niveau. Dans ce travail nous n'évoquons que la comparaison des niveaux domaines pour deux agents donnés, ce qui consiste à comparer leurs graphes de connaissances respectifs. Cette comparaison a pour but d'isoler les parties communes et spécifiques selon une stratégie donnée, de détecter et de résoudre si possible les conflits éventuels entre les différents experts. Le résultat de la comparaison servira au cognicien pour raffiner la modélisation de l'expertise. Nous avons antérieurement proposé des principes de comparaisons de graphes de connaissances (pas nécessairement des graphes conceptuels) [5].

Nous présentons dans ce qui suit notre démarche pour la comparaison de deux graphes du niveau domaine puis un algorithme de comparaison de graphes conceptuels.

3 Comparaison entre graphes conceptuels

3.1 Démarche

Nous cherchons à comparer des graphes de connaissances correspondant à un *même point de vue* mais provenant d'*experts différents*. Par «point de vue» nous désignons un graphe dont la plupart des relations appartiennent respectivement à une même catégorie. Par exemple, un graphe représentant le point de vue électrique d'un expert sur une machine, comprendra essentiellement des concepts reliés par des relations de type «électrique». Nous n'avons pas envisagé la comparaison de deux graphes de points de vue différents comme par exemple un point de vue «électrique» et un point de vue «mécanique» sur une même machine. Un type de relation peut être donné a priori ou défini via d'autres concepts et relations [14]. Nous supposons qu'il existe un *ordre partiel sur les relations* [8]. Ainsi nous disposons avant la comparaison des graphes des experts, des treillis de concepts et des hiérarchies de relations associés aux graphes de chaque expert. La comparaison de deux graphes CG_1 et CG_2 est illustrée sur la figure 2.

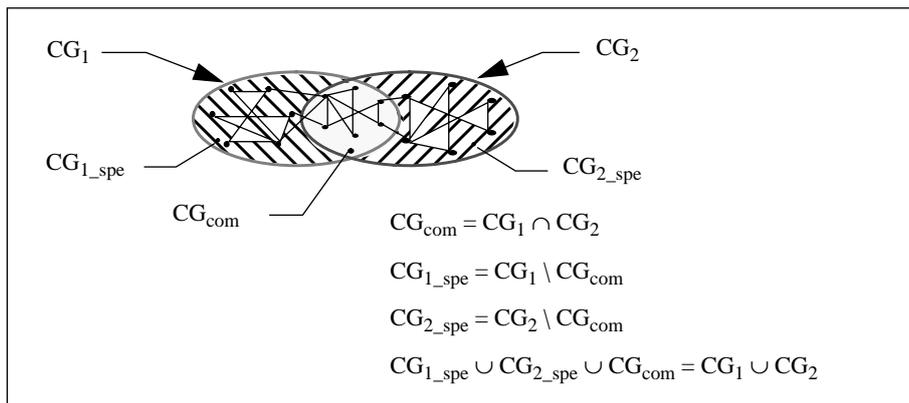


Fig.2: Résultat de la comparaison de deux graphes

Avant la phase de comparaison proprement dite des graphes de deux experts, nous procédons à l'unification de leurs treillis de concepts, puis à l'unification de leurs hiérarchies de relations. Pour cela, en accord avec les experts, le cogniticien doit résoudre les conflits d'étiquette (reconnaître les synonymes et les homonymes parmi les noms des concepts), mettre en correspondance la liste des sous-types d'un type donné reconnu chez les deux experts (résoudre les conflits d'étiquette, de type et de descendance) et de même sur les sous-types du type considéré. De nouveaux types peuvent être ajoutés et le nom de certains changés. Les liens entre le treillis unifié et les deux treillis initiaux sont mémorisés, ainsi que pour les hiérarchies. Le renommage de types dans le cas des homonymies doit être répercuté dans les graphes des experts.

Dans la suite de l'exposé, nous supposerons ces deux phases d'unification réalisées pour

nous intéresser à la phase de comparaison des graphes conceptuels. La figure 3 est un exemple de hiérarchies de relations pour deux experts et de leur unification.

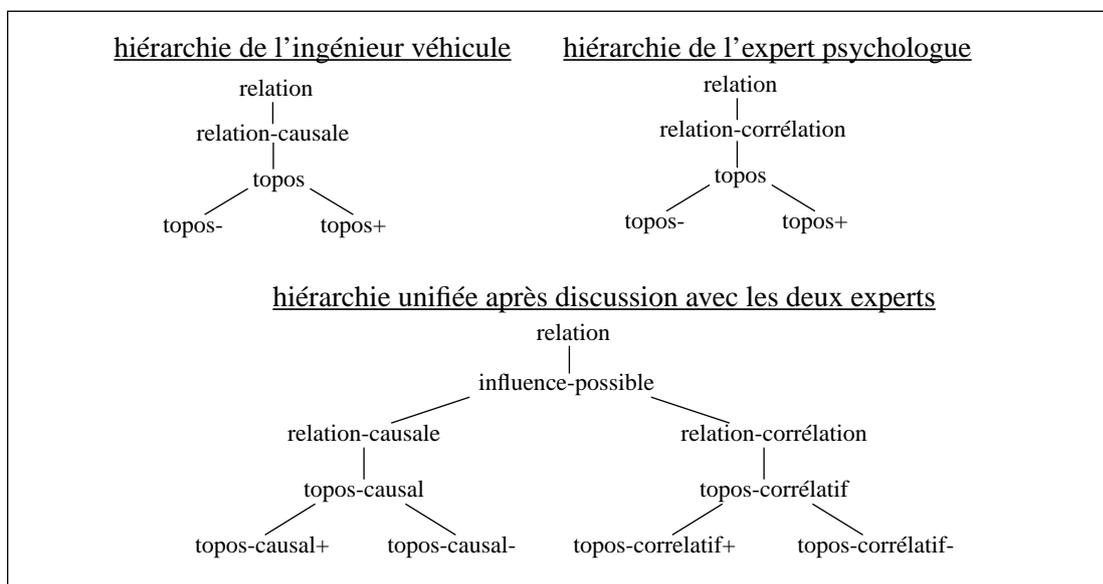


Fig.3: Unification de deux hiérarchies de relations

Les hiérarchies initiales contiennent la relation topos, spécialisée en topos-positif (i.e. corrélation entre valeurs, du type "plus x, plus y") et topos-négatif (corrélation du type "plus x, moins y"). Les topoï permettent par exemple d'effectuer du raisonnement qualitatif sur les graphes.

Pour une application d'analyse d'accident de la route, l'expert ingénieur véhicule utilise des topoï correspondant aux relations causales sous-jacentes aux équations physiques où interviennent la vitesse des véhicules, les masses, les quantités de mouvement, etc., tandis que l'expert psychologue utilise plutôt des topoï correspondant à des corrélations comme "Plus le conducteur d'une voiture GTI est jeune, plus il risque un accident dû à une vitesse excessive". L'unification de ces deux visions amène un raffinement dans la hiérarchie unifiée.

Deux relations peuvent être définies comme incompatibles, par exemple topos+ et topos-. Cette notion d'incompatibilité entre relations permet de détecter des conflits possibles entre des graphes correspondant à des experts différents.

La phase de comparaison doit permettre de situer les deux graphes l'un par rapport à l'autre et de gérer les conflits possibles entre les connaissances exprimées ; un conflit caractérise a priori une même connaissance exprimée différemment par chacun des experts. Parmi ces conflits, certains recouvrent une *incompatibilité* entre deux expressions contradictoires d'une même connaissance. Ceci aide au raffinement des agents et de leurs connaissances associées. Les graphes initiaux sont conservés, les connaissances communes et celles spécifiques à chaque expert sont isolées, les incompatibilités relevées. Le cogniticien devra

résoudre avec l'aide des experts les incompatibilités détectées entre les deux graphes. Le résultat de la comparaison sera un graphe dit «unifié» intégrant les connaissances communes et complémentaires de spécialistes sur un même domaine. Pour deux graphes CG_1 et CG_2 donnés, nous recherchons les cas suivants :

1. un graphe peut être sous-graphe (resp. sur-graphe) de l'autre,
2. l'un peut être une expansion (resp. une contraction) de l'autre [14],
3. l'un peut être une généralisation (resp. une spécialisation) sur les types de l'autre :

au sens de Sowa [14], CG_1 est une restriction de CG_2 ou bien CG_2 est une restriction de CG_1 ,

4. l'un peut être une conceptualisation (resp. une instanciation) de l'autre :

au sens de Sowa, CG_1 est une restriction de CG_2 ou bien CG_2 est une restriction de CG_1 ,

5. les deux graphes peuvent posséder des parties comparables, c'est à dire :

$\exists \{u_i\}$ avec $i \in \{1, n\}$ un ensemble de graphes conceptuels tel que $u_i \leq CG_1$ et u_i sous-graphe de CG_2 ,
et
 $\exists \{v_j\}$ avec $j \in \{1, m\}$ un ensemble de graphes conceptuels tel que $v_j \leq CG_2$ et v_j sous-graphe de CG_1 .

Nous n'avons pas encore traité le cas 2 en détail, en particulier les problèmes algorithmiques qu'il pose. Nous l'illustrons sur l'exemple suivant :

Figure 4 : si CG_1 contient le chemin $[C_{in}] \rightarrow (rel_1) \rightarrow [C_{fin}]$ et CG_2 $[C_{in}] \rightarrow (rel_{2_1}) \rightarrow [C_{int_1}] \rightarrow \dots \rightarrow (rel_{2_n}) \rightarrow [C_{fin}]$, la relation rel_1 étant compatible avec les relations rel_{2_j} pour i entier de 1 à n , on pourra proposer le chemin de CG_2 comme une expansion de cette relation de CG_1 . Pour les stratégies définies au paragraphe 3.2 ci-dessous, le chemin détaillé de CG_2 sera conservé dans le graphe commun par la stratégie de plus grande spécialisation ; le chemin plus court de CG_1 sera gardé par la stratégie de plus grande généralisation.

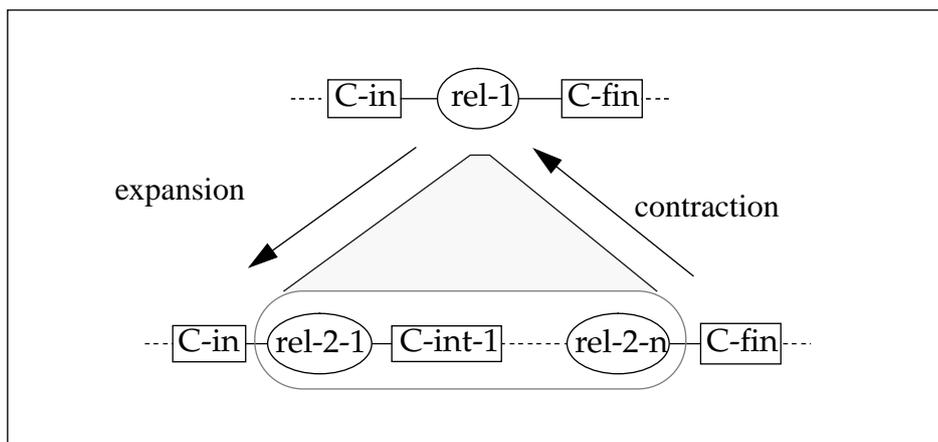


Fig.4: Mise en correspondance par expansion/contraction d'une relation et d'un chemin

Le résultat de la comparaison en accord avec une stratégie choisie au départ sera dans le cas 5 un graphe a priori non connexe (voir la partie 4). Un recollement partiel entre les composantes connexes du graphe résultant de la comparaison pourra s'effectuer en cherchant des contractions et expansions de morceaux d'un graphe dans l'autre. Cette voie semble prometteuse, tant pour établir un algorithme pour le cas 2, que pour traiter le problème du recollement du graphe résultant de la comparaison. Voyons plus en détail les différentes stratégies de comparaison que nous proposons.

3.2 Stratégies de résolution des conflits

Nous rappelons qu'un *conflit* caractérise une divergence sur la représentation d'une connaissance. Dans le cas où une incompatibilité est relevée, le conflit ne peut être résolu sans l'intervention du cognicien ou des experts.

Notre algorithme procède à une comparaison triplet par triplet de chaque graphe, un triplet étant de la forme (concept₁,relation,concept₂), «relation» allant de «concept₁» à «concept₂». En effet, le triplet nous semble le plus petit atome représentatif d'une connaissance de l'expert. Nous appliquons une stratégie permettant de construire le triplet résultat de la comparaison s'il existe. La comparaison porte sur les types des concepts et relations de chacun des deux triplets. Dans [1,5] nous avons défini les stratégies suivantes, que le cognicien pourra utiliser telles quelles, combiner ou étendre suivant ses besoins :

- *Stratégie de généralisation directe* : s'il existe une relation de spécialisation entre les deux entités comparées (types de concepts ou de relations), choisir l'entité la plus générale. Par exemple, pour [Femme] et [Etre-humain], c'est [Etre-humain] qui sera retenu.
- *Stratégie de généralisation* : s'il existe une relation de spécialisation entre les deux entités comparées, appliquer la stratégie de généralisation directe, sinon choisir un plus petit généralisant commun s'il existe. En cas de choix multiples, plusieurs alternatives seront proposées au cognicien.
- *Stratégie de spécialisation directe* : s'il existe une relation de spécialisation entre les deux entités comparées, choisir l'entité la plus spécialisée.
- *Stratégie de spécialisation* : s'il existe une relation de spécialisation entre les deux entités comparées, appliquer la stratégie de spécialisation directe, sinon choisir un plus grand spécialisant commun s'il existe. En cas de choix multiples, plusieurs alternatives seront proposées au cognicien.
- *Stratégie de la plus grande compétence* : choisir l'entité appartenant à l'expert le plus compétent dans le domaine concerné.
- *Stratégie du consensus entre experts* : aucun choix n'est fait entre des entités comparables, à moins que les experts ne se mettent d'accord sur laquelle choisir.

La stratégie de résolution de conflits peut être globale et s'appliquer tout au long du processus de comparaison d'un couple de graphes. Mais elle peut aussi être locale et changer selon le contexte. Pour une comparaison donnée, les stratégies précédentes peuvent se combiner.

Par exemple, considérons les graphes de connaissances de deux experts psychologues, analysant un accident de la circulation du point de vue des conducteurs. Si un des experts est connu comme un spécialiste des conducteurs de voiture de type GTI, l'autre comme un spécialiste des erreurs de conduite, nous pouvons adopter (a) la stratégie de la plus grande compétence lors de la comparaison des parties des graphes concernant les conducteurs de GTI ou celles concernant les erreurs de conduite (à chaque fois, nous préférons le point de vue du spécialiste concerné), (b) la stratégie de la plus grande généralisation directe quand elle s'applique, et (c) dans les cas restant, la stratégie du consensus.

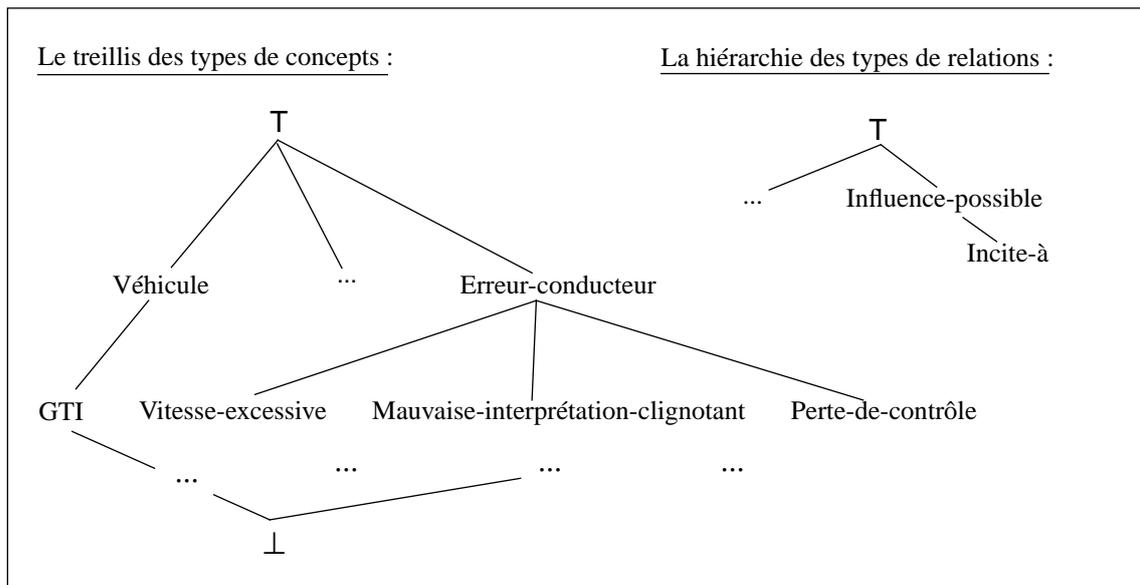


Fig.5: Le treillis des types de concepts et la hiérarchie des types de relations utilisés dans les exemples suivants.

Par exemple, avec le treillis unifié et la hiérarchie unifiée de la figure 5, si un expert exprime :

[Véhicule] → (Influence-possible) → [Erreur-conducteur]

et le second :

[GTI] → (Influence-possible) → [Erreur-conducteur], ou
 [Véhicule] → (Influence-possible) → [Vitesse-excessive], ou
 [Véhicule] → (Incite-à) → [Erreur-conducteur], ou
 [GTI] → (Incite-à) → [Erreur-conducteur], ou
 [GTI] → (Incite-à) → [Vitesse-excessive].

dans le cas de la stratégie de spécialisation directe, les expressions du second expert seront choisies. Mais si les deux experts expriment respectivement :

[Véhicule] → (Influence-possible) → [Vitesse-excessive] pour l'un et
[GTI] → (Incite-à) → [Erreur-conducteur] pour l'autre,

la stratégie de spécialisation échoue, tandis que la stratégie de généralisation va construire :

[Véhicule] → (Influence-possible) → [Erreur-conducteur].

Nous venons de préciser ce que nous attendons de l'algorithme de comparaison et nous travaillons actuellement dessus. Nous détaillons dans la suite le point actuel de nos travaux.

4 Algorithme de comparaison de graphes

Nous présentons ici les spécifications partielles de l'algorithme et nous introduisons quatre opérateurs mettant en oeuvre les stratégies de choix précédemment décrites. Du fait de leur particularité, les stratégies de consensus et de plus grande confiance ne sont pas abordées.

4.1 Hypothèses de départ et données initiales

Nous supposons les unifications des treillis de concepts et des hiérarchies de relations des deux graphes à comparer réalisées. Dans cette première étude, les deux graphes comparés sont supposés ne contenir que des concepts génériques et que des relations binaires, auxquelles on peut se ramener en utilisant des relations simples (attribut, acteur, cause, modalité, etc.).

Les données sont : deux graphes CG_1 et CG_2 de deux experts selon un point de vue donné, le treillis unifié T de leurs types de concepts, la hiérarchie unifiée H de leurs types de relations et la stratégie de comparaison choisie, notée $Strat$. Pour garder une trace de la structure des graphes, on indexera (numérotera) chaque nœud (concept ou relation) de chaque graphe. De même, nous indexerons chaque type dans les treillis et hiérarchie afin de coder l'ordre partiel de ces ensembles d'éléments. L'indexation par vecteurs binaires proposée dans [2] permet une recherche efficace de descendants communs dans un treillis, mais elle n'est pas toujours efficace pour la recherche d'ancêtres commun [15]. Nous proposons l'utilisation d'une indexation par un parcours topologique des treillis et hiérarchies [15].

4.2 Algorithme général

Appelons G le graphe commun résultat de la comparaison de CG_1 et CG_2 selon la stratégie $Strat$, et *comparer* la fonction de comparaison :

$$G = \text{comparer}(CG_1, CG_2, H, T, Strat)$$

Un algorithme possible actuellement étudié pour cette fonction est :

1. Soit n_T le nombre de types dans T , numéroter les types de n_T à 1 par un parcours topologiques en partant du type générique,
2. Soit n_H le nombre de types dans H , numéroter les types de n_H à 1 par un parcours topologiques en partant du type générique,
3. Soient n_1 et n_2 les nombres de concepts de CG_1 et CG_2 , numéroter les concepts de CG_1 de 1 à n_1 et ceux de CG_2 de n_1+1 à n_1+n_2 ,
4. Soient L_1 la liste des triplets orientés de CG_1 , L_2 la liste des triplets orientés de CG_2 et L une liste vide,
5. Choisir l'opérateur Op réalisant $Strat$,
6. Pour chaque triplet (t_1, r_1, t_1') de L_1 faire
 - pour chaque triplet (t_2, r_2, t_2') de L_2 et pour trois ensembles E, F, G de types faire,
 - $E = Op(t_1, t_2)$, si $Card(E) = 1$ et si $Element(E) = \perp$ passer au triplet suivant,
 - $F = Op(r_1, r_2)$, si $Card(F) = 1$ et si $Element(F) = \perp$ passer au triplet suivant,
 - $G = Op(t_1', t_2')$, si $Card(G) = 1$ et si $Element(G) = \perp$ passer au triplet suivant,
 - Si $Card(E) + Card(F) + Card(G) = 3$, rajouter $(Element(E), Element(F), Element(G))$ dans L , sinon, demander l'intervention du cogniticien pour choisir le(s) triplet(s) à rajouter dans L puis passer au triplet suivant.
 - passer au triplet suivant..
7. $G = \text{recollement}(L)$.

L'indexation proposée aux étapes 1 et 2 permet un test simple de la relation d'ordre sur les types de concepts et de relations.

Dans le cas où un des ensembles de types renvoyés par l'opérateur en fonction de la stratégie n'est pas un singleton, les données ne sont pas suffisantes pour que l'algorithme décide d'un triplet à retenir. Plusieurs triplets issus des deux triplets à comparer sont possibles. L'intervention du cogniticien et des experts est alors nécessaire pour déterminer si un triplet convient plus que les autres selon la stratégie de comparaison, ou si plusieurs alternatives conviennent. Le système proposera alors les triplets de $ExFxG$ et éventuellement les comparaisons deux à deux selon la même stratégie $Strat$ de ces triplets.

Le graphe résultat G est de connexité quelconque. Le problème du recollement de chaque graphe connexe de G est facilité par l'indexation des sommets de chaque graphe, ce qui conserve la structure des graphes comparés. Lors du recollement, ces informations serviront mais des problèmes de concordances de types pourront se poser pour un même nœud concept selon les triplets résultant de la comparaison.

Cet algorithme ne traite pas le cas de l'expansion précédemment évoquée. L'algorithme de recherche d'un chemin comparable à un autre (au sens d'une des stratégies précédemment évoquées), soit un sous-graphe dans un graphe, est a priori de complexité exponentielle. En connaissant les projections permettant de passer des graphes connexes composant G à leurs images (cas de stratégies de généralisation) ou images inverses (cas des stratégies de spécialisation) dans chacun des deux graphes initiaux, on peut obtenir des informations sur des contractions/expansions envisageables permettant de lier entre elles les composantes con-

nexes du graphe résultat G.

Précisons maintenant les opérateurs utilisés en fonction de la stratégie dans l'algorithme général.

4.3 Propositions de définition d'opérateurs

Pour le traitement de l'héritage multiple dans les langages à objet, les opérateurs GLB et LUB ont été définis [2]. Notre besoin est différent. Dans le cas des stratégies directes, un opérateur nous renvoyant pour deux types, (a) dans le cas où l'un est sur-type de l'autre, le sur-type (généralisation) ou le sous-type (spécialisation), et (b) le type absurde dans le cas contraire est suffisant. Dans le cas des stratégies de plus grande généralisation ou de plus grande spécialisation, nous aurions besoin d'opérateurs plus précis que GLB et LUB, nous renvoyant le ou les types les plus proches dans le treillis des types initiaux.

Nous proposons la définition des opérateurs *E-min*, *E-max*, *plus-petits-ancêtres-communs* (*ppac*) et *plus-grands-descendants-communs* (*pgdc*) suivants :

Soient deux types t_1 et t_2 appartenant au treillis unifié des types de concepts ou à la hiérarchie unifiée des types de relations,

- $E-min(t_1, t_2) = \{t_1\}$ si $t_1 \leq t_2$, $\{t_2\}$ si $t_2 \leq t_1$, $\{\perp\}$ sinon.
- $E-max(t_1, t_2) = \{t_2\}$ si $t_1 \leq t_2$, $\{t_1\}$ si $t_2 \leq t_1$, $\{\perp\}$ sinon.
- $ppac(t_1, t_2)$ renvoie l'ensemble T_A des types t vérifiant :
 $t_1 \leq t, t_2 \leq t$ et il n'existe aucun t' tel que $t_1 \leq t', t_2 \leq t'$ et $t' \leq t$,
si $T_A = \{T\}$, on renvoie $\{\perp\}$,
- $pgdc(t_1, t_2)$ renvoie l'ensemble T_D des types t vérifiant :
 $t \leq t_1, t_2 \leq t$ et il n'existe aucun t' tel que $t' \leq t_1, t' \leq t_2$ et $t \leq t'$.
Si $T_D = \{\}$, on renvoie $\{\perp\}$.

4.4 Quelques méthodes algorithmiques

La complexité de l'algorithme général proposé dépend de manière cruciale de celles des algorithmes implantant les opérateurs utilisés pour la comparaison. Un algorithme simple pour les opérateurs *E-min* et *E-max* consiste à attribuer à un type donné un numéro d'index résumant le sous arbre dont ce type est la racine. Aït-Kaci propose un tel algorithme dans [2]. Le numéro d'index est un vecteur binaire dont la première composante est un bit d'information sur l'élément racine du poset et les suivantes les bits d'information sur les éléments du poset parcouru par niveaux. Le test de la relation d'ordre entre deux éléments revient à tester dans le numéro d'index le plus élevé la valeur du bit concernant l'élément de numéro d'index le plus faible. Cette opération peut se réaliser par un ou-exclusif entre chaque valeur du bit d'information de l'autre vecteur de chacun des deux vecteurs.

Cette indexation par vecteurs binaires permet de trouver tous les descendants communs de

manière indiscriminée. Les problèmes se posent pour obtenir en un temps de calcul raisonnable les descendants les plus proches des deux éléments comparés de même que pour obtenir les ancêtres les plus proches. De nombreux travaux sur les classes de graphes de type treillis ont été menés [2,3,15]. Nous travaillons actuellement sur des algorithmes efficaces pour les opérateurs *ppac* et *pgdc*.

4.5 Conclusion

Comme nous venons de le voir, la définition des quatre opérateurs ci-dessus permettent la précision de l'algorithme général. De nombreux problèmes restent à résoudre, comme celui de la reconstitution du graphe commun aux deux graphes initiaux au sens de la stratégie de comparaison choisie. D'autre part, la validité sémantique des triplets obtenu par généralisation n'est pas prouvée. Selon les connaissances exprimées par chacun des deux experts, un triplet obtenu par généralisation peut n'avoir aucun sens. Ce problème demande une plus grande finesse du formalisme et de la comparaison pour prendre en compte plus d'information.

5 Conclusion

De nombreux problèmes restent à résoudre, tant au niveau du formalisme d'expression des connaissances que dans l'algorithme de comparaison. Les stratégies de comparaisons seront en particulier raffinées et la construction du graphe unifié plus fouillée. Nous n'avons pas encore fait tous les liens entre nos objectifs et des travaux précédents sur des problèmes proches [2,3,9].

Bibliographie

- [1] C.Amergé, O. Cordy, R. Dieng, D. Fleury, A. Giboin, S. Labidi, S. Lapalut. *Acquisition et modélisation des connaissances dans le cadre d'une coopération entre plusieurs experts : Application à un système d'aide à l'analyse de l'accident de la route*. Rapport du poste 1 du contrat MRE n° 92 C 075, décembre 1993.
- [2] H. Aït-Kaci, R. Boyer, P. Lincoln, R. Nasr. Efficient Implementation of Lattice Operation. *ACM Transaction on Programming Languages and Systems*, 11(1):115-146, January 1989.
- [3] J.P. Bordat. *Sur l'algorithmique combinatoire d'ordres finis*. Thèse, université de Montpellier, 1992.
- [4] R. Dieng, O. Corby, S. Labidi and P.-A. Tourtier. *A Model of Cognitive Agent for the Design of cooperative, explanatory systems*. In CKBS-SIG Workshop on Cooperating Knowledge-based Systems. Keelle University, UK, 7-9 September, 1993.
- [5] R. Dieng, O. Corby and S. Labidi. *Expertise conflicts in Knowledge Acquisition*. In Proceedings of the Banff Workshop on Knowledge Acquisition for Knowledge Based Systems (KAW'94), Banff, Canada. January 30-February 4, 1994.
- [6] R. Dieng, A. Giboin, P.-A. Tourtier, O. Corby. *Knowledge Acquisition for Explainable Multi-Expert Knowledge-Based Design Systems*. In T. Wetter, K.-D. Althoff, J. Boose, B. Gaines, M. Linster, F. Schmalhofer eds, Currents Developments in Knowledge Acquisition : EKAW-92, Springer-Verlag, Berlin/Heidelberg, pages 298-317, May 1992.
- [7] J. Fargues. *Conceptual Graph Infomation Retrieval using Linear Resolution, Generalization and Graph Splitting*. (in [12]).
- [8] Groupe «Graphe Conceptuel».. *Formalisation et extension des graphes conceptuels*. Actes des 4^{èmes} journées nationales du PRC-GDR Intelligence Artificielle. Marseille, pp. 57-119, 1992.
- [9] G. Mineau. *Structuration des bases de connaissances par généralisation*. Ph.D., université de Montréal, 1990.
- [10] G. Mineau. *Induction on Conncptual Graphs: Finding Common Generalisation and Compatible Projections*. (in [12]).
- [11] E. Nagle, J. Esch, G. Mineau. A Notation for Conceptual Structure Graph Matcher. (in [12]).
- [12] H. Pfeiffer, T. Nagle (eds). *Conceptual structures : theory and implementation* ; Proceedings of the 7th Annual Workshop, Las Cruces, NM, USA, July 8-10, 1992. (LNCS 754).
- [13] G. Schreiber, B. J. Wielinga and J. Breuker. *KADS A Principled Approach to Knowledge-Based System Development*. Academic Press, 1993.
- [14] J. F. Sowa. *Conceptual Structures, information processing in mind and machine*. Addison-Wesley System Programming Series. 1984.
- [15] M. Wermelinger. *Searching Efficiently in Posets*. Departamento de Informatica, Universidade Nova de Lisboa, 2825 Monte da Caparica, Portugal, mars 1993.