Article publié dans R.I.A, vol.10, n°1, 1996, 7-56

Représenter des connaissances et raisonner avec des graphes

Marie-Laure Mugnier &* Michel Chein

LIRMM (CNRS & Université Montpellier II)
161, rue ADA, 34392 Montpellier Cedex 5,
{mugnier, chein}@lirmm.fr

*C'est un & commutatif.

Ce travail a été réalisé avec le soutien du PRC-GDR en IA.

RESUME. Nous exposons notre point de vue sur les graphes conceptuels : des graphes munis d'étiquettes ordonnées —qui servent à représenter des connaissances— sur lesquels sont définis des opérations de graphes —qui servent à raisonner. Puis nous présentons un modèle de graphes simples, généralisant le modèle habituellement considéré, tout en conservant les propiétés essentielles de celui-ci. Nous étudions ensuite les liens entre l'opération de base (la projection), et des notions importantes de combinatoire (morphisme de graphes, problèmes de satisfactions de contraintes). Nous terminons par la définition d'une notion de graphes conceptuels emboités qui constitue une extension significative du modèle simple.

ABSTRACT. We briefly present our point of view about conceptual graphs : graphs with ordered labels —for representing knowledge— with graph operations —for reasoning. We propose a formalism for simple conceptual graphs which generalizes the usually considered model, and preserves its fundamental properties. Links with important combinatorial notions (graph morphism, Constraint Satisfaction Problems) are established. Finally a notion of nested conceptual graph is formally defined, which significantly extends the simple model.

MOTS-CLES : graphes, graphes conceptuels, représentation de connaissances, sémantique logique, morphisme de graphes, problèmes de satisfaction de contraintes, graphes emboîtés.

KEY WORDS : graphs, conceptual graphs, knowledge representation, logical semantics, graph morphism, constraint satisfaction problems, nested graphs.

1. Introduction

Cet article concerne un modèle formel de représentation de connaissances, du type réseau sémantique, connu sous le nom de *graphes conceptuels (GC)*. Dans le livre de Sowa de 1984 [Sowa 84], le terme "graphe conceptuel" désignait à la fois : un modèle de base, défini d'une manière assez précise, et appelé parfois graphe conceptuel simple, et des extensions, ou des idées d'extensions, du modèle de base définies de manière plus ou moins formelle. Depuis 1984 de nombreux travaux, ayant leur origine dans le livre de Sowa, ont été menés sous le vocable graphe conceptuel. Les chercheurs ont développé le modèle de base proposé par Sowa, ou certaines de ses extensions, en fonction de leurs propres idées ou besoins. Aujourd'hui il existe une grande diversité de points de vue, concernant aussi bien la définition de notions fondamentales, comme, par exemple, celles de "joint étendu" ou de "graphe canonique", que les relations entre les graphes conceptuels et la logique. Sowa considère d'ailleurs que le terme graphe conceptuel représente une famille de ressemblance au sens de Wittgenstein! Nous avons publié en 1992 [Chein Mugnier 92] une première étude issue de notre vision du modèle des graphes conceptuels simples. Sur cette base nous avons créé une équipe de recherche au LIRMM qui a engagé des travaux sur divers aspects concernant les graphes conceptuels, ceci en collaboration avec d'autres équipes de recherche, dans le cadre d'un projet soutenu par le PRC-IA [GC 95]. Nous avons, en particulier, commencé la construction du système CORALI, qui est un environnement de développement d'applications à base de connaissances, utilisant comme modèle unique de représentation de connaissances les graphes conceptuels [Corali 94]. Notre objectif fondamental est de prouver que l'on peut représenter des connaissances par des graphes et que des raisonnements sur ces connaissances peuvent être simulés par des algorithmes de graphes.

Dans la partie suivante de cet article, nous expliquons pourquoi nous pensons que les graphes conceptuels sont un modèle de représentation de connaissances intéressant. Nous précisons en particulier notre point de vue sur les liens entre graphes conceptuels et logique.

La troisième partie est consacrée à la présentation du modèle des graphes conceptuels simples —lorsqu'il n'y a pas d'ambiguité nous utilisons la locution graphe simple à la place de graphe conceptuel simple— ce modèle généralisant légèrement le modèle proposé dans [Sowa 84] et étudié dans [Chein Mugnier 92]. Nous commençons par présenter la notion de *support* (3.1), qui constitue le vocabulaire de base du modèle, celui dans lequel tous les objets sont décrits. Le paragraphe 3.2 donne une définition formelle de la notion de graphe simple. Un graphe simple permet de représenter des connaissances de différente nature : faits, buts, définitions de types, règles, contraintes (aspect développé en 4.3). Le modèle des graphes conceptuels peut ainsi être qualifié d'*homogène*. Dans le paragraphe 3.3 nous présentons deux sortes d'*opérations* sur les graphes simples : des opérations élémentaires de graphes, dites de spécialisation/généralisation, et l'opération de

projection, qui correspond à la vision "globale" d'une séquence de spécialisation, et qui est l'opération fondamentale. Dans le paragraphe 3.4 nous montrons que le modèle est muni d'une *sémantique logique* consistante et complète (en logique du premier ordre). Une sémantique ensembliste, qui correspond à la sémantique ensembliste habituelle de la logique du premier ordre, est proposée en annexe. La quatrième partie de cet article présente quelques compléments et cas particuliers usuels du modèle.

La cinquième partie étudie les liens entre la projection et des notions importantes de combinatoire : morphisme de graphes et problèmes de satisfaction de contraintes (CSP). Nous montrons en particulier qu'il existe une correspondance forte entre un CSP et le problème de l'existence d'une projection entre deux graphes conceptuels simples.

La sixième partie concerne les *graphes conceptuels emboîtés*, une extension des graphes simples, et nous définissons sur ces graphes une projection et des opérations de spécialisation. La question de la logique qui peut être associée à ces graphes est en cours d'étude.

Pour des raisons de complétude de cet article, nous avons rappelé certaines notions et résultats de [Chein Mugnier 92], les liens précis entre ces deux articles étant explicités ci-dessous. Le modèle des graphes simples étudié ici (partie 3) correspond au modèle des S-graphes de [Chein Mugnier 92], avec les extensions suivantes :

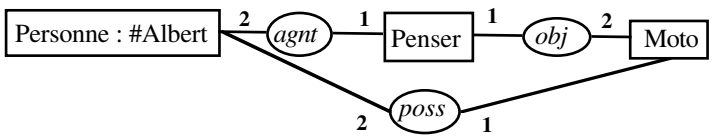
- Les graphes conceptuels ne sont pas nécessairement connexes.
- L'ensemble des types de concepts forme un ordre, muni d'un plus grand et d'un plus petit élément, mais n'est pas nécessairement un treillis.
- L'ensemble des types de relations est muni d'une structure, qui est un ordre.

Ces extensions nécessitent d'étendre les opérations élémentaires de spécialisation/généralisation et la notion de projection. La correspondance complète entre dérivation par spécialisation —ou projection— et déduction logique sur les formules associées est conservée. Certains aspects ont été précisés (comme la notion de dérivation ou la sémantique logique), des écritures ont été simplifiées (au niveau du support notamment).

2. Un point de vue sur les graphes conceptuels

Un modèle de représentation de connaissances du type graphe biparti étiqueté par des items lexicaux[1] est intéressant à plusieurs titres. Dans la *communication homme/machine,* il apporte des possibilités de représentations graphiques. En effet, une personne peut interpréter un dessin représentant un tel graphe en termes d'un ensemble de relations entre des concepts. Une telle interprétation est facile si le graphe a un petit nombre de sommets et d'arêtes, ou s'il a une structure particulière mise en évidence par le dessin. Cette facilité de lecture par une personne, l'aspect visualisable d'un graphe, est particulièrement intéressante en représentation de connaissances. En effet, dans une application utilisant des représentations de connaissances il y a, généralement, des phases d'interaction avec un utilisateur non informaticien, devant entrer certaines connaissances, ou faire des contrôles, ou comprendre certains résultats fournis par le système.

Mais il est clair que l'aspect descriptif n'est pas l'unique critère. Considérons la phrase "Albert pense à sa moto". La représenter en machine n'est pas difficile, il suffit de considérer la chaîne de caractères précédente. Il s'agit ensuite de pouvoir raisonner sur la représentation. Une représentation de cette phrase par un graphe conceptuel pourrait être (Figure 1):



agnt, *obj* et *poss* sont des abréviations de "agent", "objet", "possessif"

Figure 1. *Albert pense à sa moto.*

Ce graphe représente l'ensemble des mots non grammaticaux de la phrase et certaines relations typées entre ceux-ci : il y a, par exemple, entre un objet de type PERSONNE identifié par la chaîne "Albert" et un objet de type PENSER, une relation du type AGENT (le sens de lecture étant : le premier argument *a pour agent* le second argument). Sowa présente souvent le modèle des graphes conceptuels comme un intermédiaire entre langue naturelle et logique, et de nombreux chercheurs utilisent ce modèle en linguistique informatique. Pour notre part, nous n'aborderons

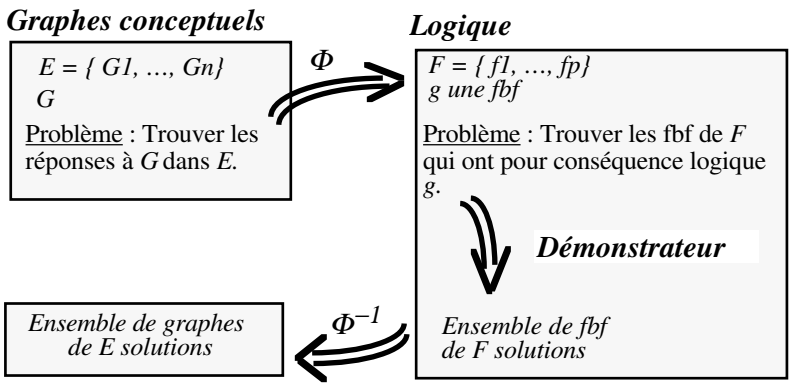
[1] Les étiquettes sur une des classes de sommets correspondent à des noms de concepts, et les étiquettes sur l'autre classe correspondent à des noms de relations conceptuelles.

pas dans cet article les difficiles problèmes concernant les relations entre les langues et les graphes conceptuels.

Un modèle du type graphe étiqueté est intéressant d'un point de vue *informatique* pour diverses raisons. De nombreux algorithmes efficaces ont été développés ces dernières années, pour résoudre des problèmes divers sur les graphes. Avoir des représentations de connaissances sous forme de graphes, et des raisonnements en termes d'algorithmes classiques de graphes, permet d'amarrer solidement la représentation de connaissances à l'une des bases solides de l'informatique : *l'algorithmique combinatoire*. Un graphe, en tant qu'objet mathématique, admet des représentations informatiques *naturelles*, par des structures composées de cellules reliées par des pointeurs. Ces structures permettent non seulement de construire des algorithmes efficaces, mais elles sont, de plus, analogues aux graphes représentés. Elles permettent ainsi à un utilisateur de suivre des calculs, en visualisant les graphes obtenus aux différentes étapes, ce qui est appréciable si l'on souhaite avoir des fonctionnalités du type explication. Ce point rejoint les aspects communication homme/machine évoqués précédemment. Enfin, il existe, et il est facile de construire, des éditeurs de graphes permettant de construire une des structures informatiques mentionnées ci-dessus, représentant un graphe, à partir d'un dessin sur un écran.

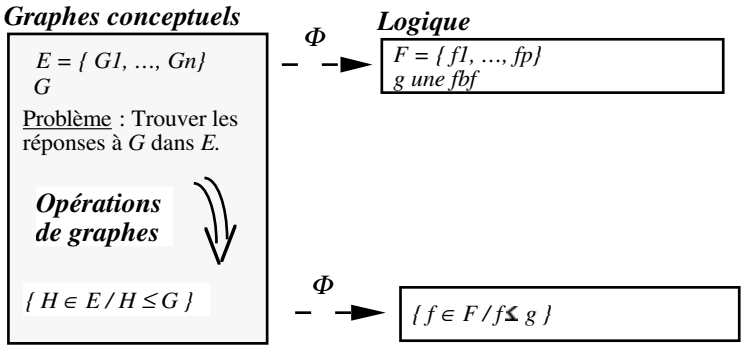
Ceci explique que l'on ait souvent associé à des modèles du type graphe étiqueté un aspect essentiellement procédural, la version la plus pauvre étant de faire simplement des parcours sur ces graphes. Nous verrons que, dans le modèle que nous proposons, les opérations sont plus complexes que des cheminements, en particulier l'opération de projection qui correspond à l'implication logique, et que, d'autre part, le modèle des graphes conceptuels peut être considéré comme un modèle *déclaratif*, au même titre que la logique, ce qui est important pour représenter des connaissances.

Deux types de relation entre les graphes conceptuels et la logique sont proposées dans le premier livre de Sowa. L'une consiste à considérer que les graphes conceptuels sont une *représentation graphique* (au sens dessin plutôt qu'au sens de la théorie des graphes) *de la logique* (Figure 2). Ce point de vue est issu de travaux du logicien Peirce, qui a introduit les "graphes existentiels", dans le but d'effectuer des déductions logiques de façon graphique (cf. par exemple [Roberts 93]). Un graphe existentiel est un ensemble de "boîtes", pouvant être emboîtées les unes dans les autres, contenant des graphes étiquetés, les sommets de ces graphes pouvant être reliés entre eux par des lignes de co-référence. Un graphe existentiel représente une formule de la logique du premier ordre, les lignes de co-référence représentant des variables, les boîtes des négations, et les graphes des formules conjonctives positives fermées existentiellement. Les calculs sur ces objets sont des manipulations graphiques (duplication, déplacement, ou effacement d'une boîte,...) qui ne correspondent pas à des algorithmes sur des graphes (toujours au sens théorie des graphes).



Exemple : étant donné un ensemble de faits *E*, il s'agit de trouver les réponses à une question *G*, les faits et la question étant représentés par des graphes conceptuels. Une fonction Φ associe aux graphes des formules logiques, et le calcul est fait par un démonstrateur logique.

Figure 2. *Les GC comme représentation graphique de la logique*



Exemple : étant donné un ensemble de faits *E*, il s'agit de trouver les réponses à une question *G*, les faits et la question étant représentés par des graphes conceptuels. Une fonction Φ associe aux graphes des formules logiques, *mais* le calcul est fait par un démonstrateur GC.

Figure 3. *Sémantique logique d'un démonstrateur* GC.

L'autre point de vue sur la relation entre graphes conceptuels et logique consiste également à associer à un graphe conceptuel *G* une formule $\Phi(G)$ de la logique du premier ordre, mais ceci de telle sorte que :

(1) $G \leq H$ *si et seulement si* $\Phi(G) \leq \Phi(H)$.

La relation ≤ entre graphes conceptuels, appelée *relation de spécialisation*, correspond à l'existence d'un morphisme de graphes entre *G* et *H,* satisfaisant une propriété sur les étiquettes. De plus, l'existence de cette relation de spécialisation, ou de ce morphisme, entre deux graphes peut également être définie constructivement en utilisant des opérations simples (et de type théorie des graphes : identification de deux sommets, suppression de sommets jumeaux,...) sur les graphes. La fonction Φ est alors une sémantique logique consistante et complète pour le modèle des graphes conceptuels. Sowa proposait, dans son livre de 1984, une fonction Φ pour les graphes conceptuels simples, et prouvait la partie nécessaire de (1) (i.e. la consistance), nous avons prouvé la partie suffisante (i.e. la complétude) dans [Chein Mugnier 92].

C'est ce deuxième point de vue sur les liens entre graphes conceptuels et logique que nous avons choisi de privilégier. Il permet de considérer le modèle des graphes conceptuels simples comme :

- un modèle *déclaratif* de représentation de connaissances basé sur des graphes étiquetés,
- les calculs d'inférences se faisant par des *algorithmes de graphes* (donc sans utiliser un démonstrateur de théorèmes en logique),
- ayant une *sémantique logique consistante et complète*.

Les graphes conceptuels simples sont pauvres d'un point de vue descriptif, puisqu'ils correspondent à des formules conjonctives, positives, ayant pour seuls termes des variables ou des constantes, et fermées existentiellement. L'un de nos axes prioritaires de recherche concernant les aspects théoriques est donc d'étendre les capacités descriptives du modèle tout en gardant les propriétés ci-dessus.

Mais quel est l'intérêt du modèle des graphes conceptuels (simples) s'il n'est équivalent qu'à une petite partie de la logique du premier ordre? Citons les points qui nous semblent essentiels.

D'une part, aspect déjà mentionné, la *facilité de représentation graphique* d'un graphe : "un bon dessin vaut mieux qu'un long discours". L'aspect visuel est d'ailleurs souvent, pour de nombreux chercheurs, la seule justification au modèle : un graphe conceptuel est "plus facile à lire" que la formule logique qui lui est associée. Ceci parce qu'il n'y a pas de variable, et que c'est une représentation "objet" faisant bien apparaître les relations entre les divers objets (cf. les nombreux articles de Sowa, et ce qu'écrit Peirce).

D'autre part, les *aspects algorithmiques*. Les inférences se calculent par des algorithmes de graphes, et les algorithmes polynomiaux que nous avons construits, ou les heuristiques que nous proposons, ne correspondent pas, à notre connaissance, à des propriétés mises en oeuvre dans des démonstrateurs de formules logiques. De plus, la projection, qui est l'opération fondamentale, et qui correspond à l'implication logique, est facilement visualisable, un utilisateur peut donc suivre visuellement un "raisonnement" (toujours avec les restrictions mentionnées : que les graphes concernés soient petits).

Enfin, la *facilité d'extension* des capacités descriptives du modèle des graphes conceptuels simples. Illustrons ce qu'est, à notre sens, une extension du modèle de

base. On peut considérer une modalité, appliquée à un graphe G, comme une relation unaire liée au graphe G, et proposer une représentation graphique simple, *(mod)—[représentation de G]*. Il faut alors étendre la relation de spécialisation en conservant une notion calculable à partir d'algorithmes de graphes. De plus, pour ne pas rester à un niveau procédural en se limitant à une sémantique opérationnelle, il faut ensuite construire une sémantique déclarative, consistante et complète, pour une telle extension. Par exemple proposer une extension de la fonction Φ, qui associerait à de tels graphes des formules, dans une logique pouvant être différente de la logique du premier ordre. Les preuves de complétude, en particulier, ne seront pas nécessairement simples, mais on peut espérer que le modèle permettra alors de représenter agréablement des connaissances non (encore) abordées dans le cadre de la logique, et de raisonner efficacement sur celles-ci

L'un des objectifs de nos travaux est ainsi de montrer que la théorie et l'algorithmique des graphes étiquetés constituent un domaine scientifique autonome utile pour la représentation de connaissances.

3. Le modèle des graphes simples

Nous présentons dans cette partie les éléments essentiels du modèle des graphes simples : le vocabulaire, les objets, les opérations et la sémantique logique.

Notation. On note $\leq$ un ordre partiel, et $<$ l'ordre strict associé ($i < j$ si $i \leq j$ et $i \neq j$).

3.1. *Support*

Un support définit le vocabulaire de base avec lequel on représente des connaissances sur un domaine, sa terminologie, ce qu'à un niveau plus philosophique on appellerait une ontologie du domaine. Les connaissances de toute nature sont représentées par des graphes conceptuels, qui n'ont de sens que relativement à un support donné.

Un support comprend des ensembles ordonnés de types de concepts et de relations, les relations étant munies de signatures. Les deux ordres sur les ensembles de types de concepts et de relations sont interprétés comme des liens *Sorte de*. Un support contient également un ensemble d'individus typés : à tout individu m on associe le plus petit type de concept t tel que m soit un t (ceci correspond donc à une relation *Est un*). Cf. la figure 4 pour un exemple de support, qui n'a d'autre prétention que d'illustrer les définitions (ce n'est pas un début d'ontologie!).

Définition. Un *support* est un quintuplet $S=(T_C, T_R, \sigma, I, \tau)$ vérifiant :
• T_C, ensemble des types de concepts, est un ensemble partiellement ordonné possédant un plus grand élément ($\top$, dit type universel) et un plus petit élément ($\bot$, dit type absurde).
• T_R, ensemble des types de relations, est un ensemble partitionné en ensembles de types de relations de même arité. $T_R = T_{Ri1} \cup ... \cup T_{Rip}$, où T_{Rij} est l'ensemble des types de relations d'arité i_j, $i_j \neq 0$. Tout T_{Rij} admet un plus grand et un plus petit élément.
• σ associe à tout type de relation le type maximal de chacun de ses arguments. C'est plus précisément une application qui à tout $t_r \in T_{Rij}$ associe un uplet $\sigma(t_r) \in (T_C)^{ij}$, et vérifie : pour tous t_{r1}, t_{r2} de TRi_j, si $t_{r1} \prec t_{r2}$ alors $\sigma(t_{r1}) \leq \sigma(t_{r2})$, où l'ordre considéré sur les signatures est l'ordre produit[2] sur $(T_C)^{ij}$ (autrement dit, le type associé au $k^{ième}$ argument de t_{r1} est $\leq$ au type associé au $k^{ième}$ argument de t_{r2}). On notera $\sigma_i(t_r)$ le $i^{ème}$ argument de $\sigma(t_r)$.
• I, ensemble des marqueurs individuels, est un ensemble non nécessairement fini mais dénombrable. Il existe en outre un marqueur dit générique, noté $*$. L'ensemble des marqueurs $I \cup \{*\}$ est muni de l'ordre suivant : $*$ est plus grand que tous les marqueurs individuels, qui sont deux à deux incomparables.
• τ est une application de I dans $T_C \setminus \{\bot\}$, qui à tout marqueur individuel m associe un type de concept t.

Quelques remarques sur le support.

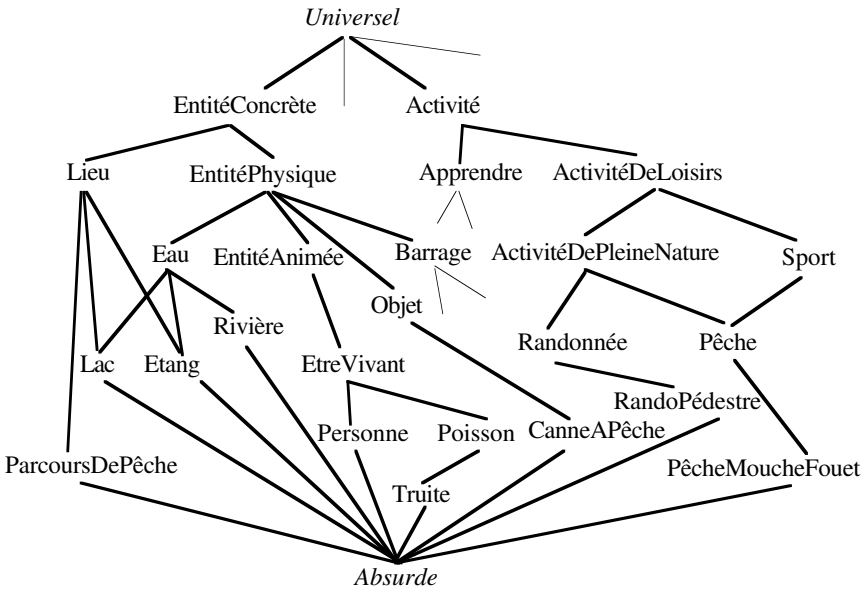
Dans les applications liées au traitement de la langue naturelle, les types de relation correspondent souvent aux cas des verbes [Sabah Vilnat 93], et sont alors binaires. Cependant, dans le cas général, il est utile de considérer des types de relations d'*arité quelconque*, par exemple lorsque les types de relations correspondent à des schémas relationnels d'une base de données relationnelle.

Nous ne considérons ici que des types (de concepts ou de relations) *atomiques*, c'est-à-dire non définis. Il existe des mécanismes de définition de types à partir de types existants, que nous n'aborderons pas ici. La relation *Sorte de* entre types n'est alors plus nécessairement donnée par le constructeur du support, mais elle peut être calculée (elle est basée sur la relation de spécialisation entre les graphes conceptuels). Se référer notamment à [Sowa 84] pour les mécanismes de définition de types et leur discussion, et à [Chein Leclère 94] [Leclère 95] pour l'étude de la relation de spécialisation entre types et l'intégration de types définis au support.

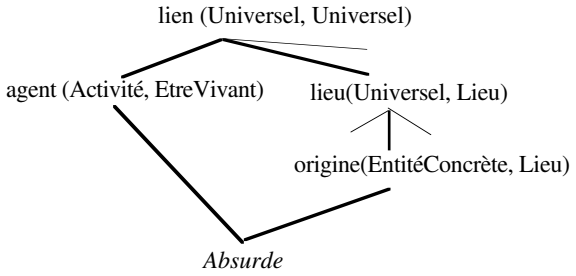
---

[2] Le produit de deux ordres $O_1=(X_1, \leq_1)$ et $O_2=(X_2, \leq_2)$ est l'ordre $O=(X_1 \times X_2, \leq)$ tel que : pour tous (x,y) et (z,t) éléments de $X_1 \times X_2$, $(x,y) \leq (z,t)$ ssi $x \leq_1 z$ et $y \leq_2 t$. Cette définition se généralise au produit d'un nombre quelconque d'ordres.

La construction d'un support est un problème difficile et crucial, car il détermine la façon d'exprimer toutes les connaissances (cf. par exemple [Bouaud et al. 94]). Il est donc nécessaire de disposer d'outils d'aide à la construction de supports [Chein Leclère 94] [Leclère 95].



Vue partielle d'un ensemble de types de concepts.



Quelques types de relations binaires et leur signature.

I = { Bayard, Altier, Barrandon, TourDuBarrandon}
τ = {(Bayard, Lac), (Altier, Rivière), (Barrandon, Etang), (TourDuBarrandon, RandoPédestre)}

Quelques marqueurs individuels et la relation τ.

Figure 4. *Définition partielle d'un support*

3.2. *Graphe conceptuel*

Définition. Un graphe conceptuel simple $G = (R, C, U, étiq)$ est un multigraphe[3] non orienté, biparti, non nécessairement connexe, avec :

• R et C sont les deux classes de sommets, appelés sommets concepts et sommets relations, $C \neq \emptyset$

• U est l'ensemble des arêtes; l'ensemble des arêtes adjacentes à tout sommet relation r est totalement ordonné, ce qu'on représente en numérotant les arêtes (de 1 au degré de r). On note $G_i(r)$ le $i^{ème}$ voisin de r dans G.

• *étiq* est une application, qui à tout sommet associe une étiquette :

Si $r \in R$, $étiq(r) \in T_R$, c'est le type du sommet relation.

Si $c \in C$, $étiq(c) \in (T_C \setminus \{\bot\}) \times (I \cup \{*\})$. L'étiquette d'un sommet concept est un couple $(type(c), marqueur(c))$. Si c possède un marqueur générique, c est dit sommet concept générique, sinon c est un sommet concept individuel.

• *étiq* obéit aux contraintes fixées par les applications σ et τ.

Pour tout $r \in$ R, $type(G_i(r)) \leq \sigma_i(type(r))$.

Pour tout $c \in C$, si $marqueur(c) \in I$, alors $type(c) = \tau(marqueur(c))$[4]

Le graphe de la figure 5 peut être interprété intuitivement comme représentant des informations sur le lac Bayard : celui-ci résulte d'un barrage sur une rivière ; il contient des truites que l'on peut pêcher avec une canne-à-pêche.

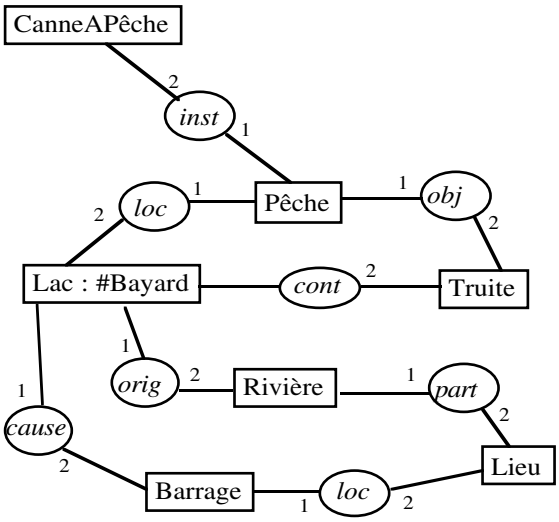
3.3. *Les opérations*

La notion fondamentale pour tout raisonnement concernant les graphes conceptuels est la relation de spécialisation/généralisation (ou encore subsomption) sur l'ensemble des graphes conceptuels. On la note ≤. Lorsqu'on considère la sémantique logique associée au modèle de base, cette relation correspond à l'implication logique. La relation ≤ peut être définie en termes d'une séquence de transformations élémentaires sur des graphes, les *opérations élémentaires de spécialisation/généralisation*, ou par un morphisme de graphes, la *projection*. Cette partie présente les deux types d'opérations, puis étudie la structure sur l'ensemble des graphes conceptuels induite par la relation ≤.

---

[3]Un multigraphe est un graphe tel qu'il peut exister plusieurs arêtes entre deux sommets, ici entre un sommet relation et un sommet concept.

[4]Une contrainte moins forte peut être : $type(c) \geq (marqueur(c))$. A toute étiquette $e = (t,m)$, où $m \in I$, on associe alors deux types, un type de "surface", t, et un type "profond", $\tau(m)$. C'est le type profond qui est utilisé dans les opérations.



Il est d'usage de représenter les sommets concepts par des rectangles, et les sommets relations par des ellipses. Par défaut, les marqueurs sont génériques.

Figure 5. *Un graphe conceptuel simple (connexe).*

3.3.1. *Opérations élémentaires de spécialisation.*

Définition. Les *opérations élémentaires de spécialisation* sont des opérations internes sur l'ensemble des graphes conceptuels définis sur un support donné. Il y a quatre opérations unaires (au sens où elles opérent à partir d'un seul graphe) et une opération binaire (i.e. qui se définit à partir de deux graphes). Le graphe obtenu est *disjoint* du (ou des) graphe(s) opérande(s).

Opérations unaires

Soit G un graphe conceptuel. On peut obtenir H à partir de G, par :

• *Simplification* : Si G possède deux sommets relations de même type et jumeaux (c'est-à-dire ayant exactement les mêmes voisins dans le même ordre), H s'obtient en supprimant l'un des jumeaux.

• *Restriction de relation* : Soit r un sommet relation de G. H s'obtient en diminuant l'étiquette de r : remplacer le type de r par un type plus petit $t_{r'}$. Cette opération doit conserver les contraintes fixées par σ : pour i de 1 à $degré(r)$, $type(G_i(r)) \leq \sigma_i(t_{r'})$.

• *Restriction de concept* : Soit c un sommet concept de G. H s'obtient en diminuant l'étiquette de c : remplacer $étiq(c)=(t,m)$ par $e'=(t',m')$ avec $t' \leq t$ et $m' \leq m$, tout en vérifiant : si m' est un marqueur individuel, alors $t' = \tau(m')$. Il est à noter que si $m' < m$, m est un marqueur générique, et m' un marqueur individuel.

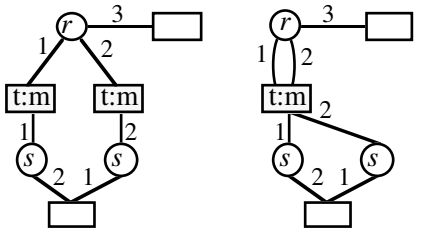
• *Joint interne* : Soient c_1 et c_2 deux sommets concepts de G ayant même étiquette. H s'obtient en fusionnant c_1 et c_2 (cf. Figure 6, où les sommets fusionnés

appartiennent à la même composante connexe. Ils pourraient aussi appartenir à deux composantes connexes distinctes.)

Opération binaire

Soient G_1 et G_2 deux graphes conceptuels disjoints. H s'obtient à partir de G_1 et G_2 par :

• *Somme (disjointe)* : H s'obtient par somme disjointe ("juxtaposition") de G_1 et G_2.



Le graphe de droite s'obtient par joint interne à partir du graphe de gauche

Figure 6. *Joint interne*

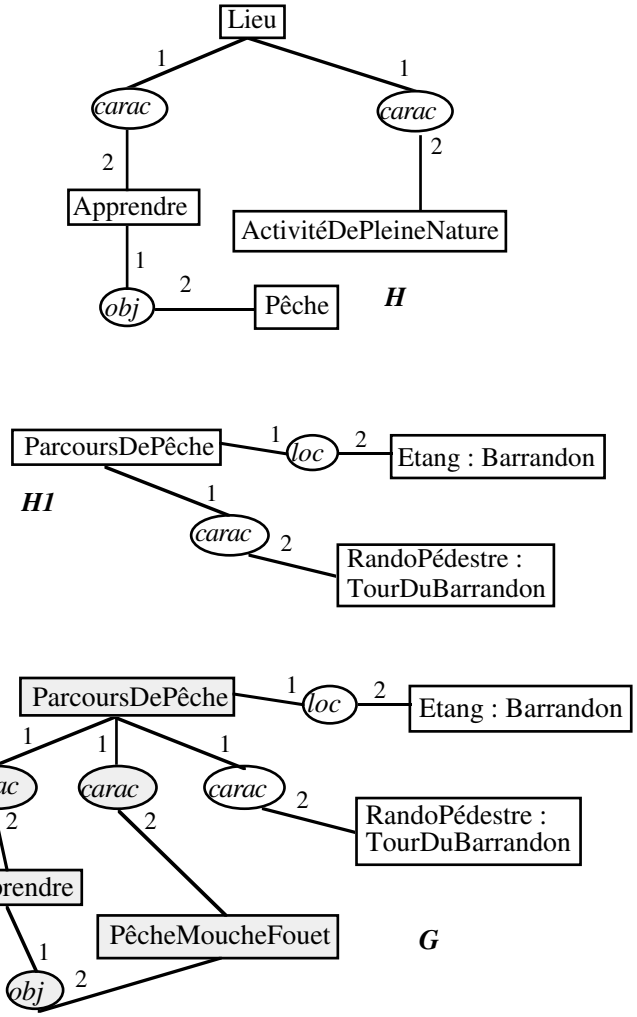
L'opération unaire "joint interne" est à mettre en correspondance avec une opération binaire dite de *joint externe*, et définie ainsi : soient G_1 et G_2 deux graphes conceptuels distincts. Soient c_1 et c_2 deux sommets concepts, de G_1 et G_2 respectivement, ayant même étiquette. H s'obtient en fusionnant c_1 et c_2. Le joint externe n'est pas une opération élémentaire, il peut s'écrire comme composition d'une somme disjointe et d'un joint interne. Si l'on impose la connexité des graphes conceptuels, il devient une opération élémentaire (cf. section 4.1.1).

G est dit spécialisation de H si G peut s'obtenir à partir de H, et éventuellement d'autres graphes, par un nombre fini d'opérations élémentaires. Plus formellement :

Définition. Une *dérivation* d'un graphe conceptuel simple G est un graphe (fini) orienté, sans circuit, dont les sommets sont étiquetés par des graphes conceptuels simples, et tel que:
(1) il a un unique puits étiqueté par G
(2) tout sommet (soit H son étiquette) vérifie :
- soit il est source,
- soit il a un seul prédécesseur (d'étiquette K), et H s'obtient à partir de K par une opération unaire,
- soit il a deux prédécesseurs (d'étiquettes K_1 et K_2 respectivement), et H s'obtient à partir de K_1 et K_2 par une opération binaire, ici la somme disjointe.

A partir de cette notion, on peut définir formellement la notion de séquence de dérivation, qui correspond à celle utilisée par [Sowa 84] : étant donnée une dérivation δ' de G, toute suite d'étiquettes associée à une extension linéaire de δ' (ou tri topologique de δ') est une séquence de dérivation de G.

Définition (*Relation de spécialisation* ≤). *G* est une spécialisation de *H* (noté *G*≤*H)* s'il existe une dérivation de *G* contenant un sommet étiqueté par *H*. Deux graphes isomorphes sont considérés comme étant identiques du point de vue de la relation ≤. *G* se dérive d'un ensemble γ de graphes conceptuels, s'il existe une dérivation de *G* dont toutes les sources correspondent à des éléments de γ.



G ≤ *H*. Par exemple, partant de *H,* en effectuant une restriction sur chacun des sommets concepts, puis un joint interne, on obtient la partie grisée (soit H_2). Il reste à effectuer une somme disjointe de H_2 et de H_1, puis un joint interne, pour obtenir *G*.

Figure 7. *Relation de spécialisation*

Définition. A toute signature de type de relation $\sigma(t_r) = (t_1, \ldots, t_p)$, on peut associer de façon naturelle un graphe conceptuel, qu'on appelle *graphe étoile* de t_r, comportant un unique sommet relation de type t_r qui possède p voisins, chaque $i^{ème}$ voisin étant étiqueté par le type t_i et le marqueur générique. L'ensemble des graphes étoiles associés à l'ensemble des types de relation T_R d'un support est appelé *base* de ce support.

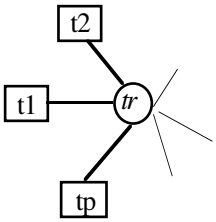


Figure 8. *Graphe étoile*

Propriété 1. *Soit B_0 la base d'un support S. L'ensemble des graphes conceptuels que l'on peut définir sur S est exactement l'ensemble des graphes dérivables de l'ensemble $B_0 \cup \{[\top:*]\}$.*

Dans la propriété 1, au lieu de B_0, on peut considérer $max(B_0)$ l'ensemble des graphes étoiles associés aux éléments maximaux de T_R (ensemble des plus grands éléments des TR_i), car tout graphe étoile s'obtient par restriction de concept et de relation à partir de $max(B_0)$.

3.3.2. *Opérations élémentaires de généralisation*

On peut définir des *règles élémentaires de généralisation*, qui sont les inverses des règles de spécialisation. On définit ainsi cinq opérations permettant d'obtenir un graphe H à partir d'un graphe G.

• *Duplication* : Soit r un sommet relation de G. On obtient H en ajoutant un sommet relation de même type que r et jumeau de r.

• *Augmentation de relation* : Soit r un sommet relation de G. H s'obtient en augmentant le type de r. Il n'y a pas de contrainte sur cette opération.

• *Augmentation de concept* : Soit c un sommet concept de G. On obtient H en augmentant l'étiquette de c : remplacer $étiq(c)=(t,m)$ par $e'=(t',m')$ avec $t' \geq t$ et $m' = *$. Cette opération doit conserver les contraintes fixées par σ : pour tout sommet relation r voisin de c, soit $c = G_i(r)$, on a $t' \leq \sigma_i(type(r))$.

• *Eclatement* : Soit c un sommet concept de G. On obtient H en éclatant c en deux sommets c_1 et c_2, de même étiquette que c, tel que les ensembles d'arêtes adjacentes à c_1 et c_2 forment une bipartition (au sens large, l'un des deux ensemble peut être vide) de l'ensemble des arêtes adjacentes à c.

• *Décomposition* : Supprimer certaines composantes connexes (pas toutes) de G.

Etant donnés deux graphes conceptuels G et H, on dit que H est une *généralisation* de G si H s'obtient à partir de G par une séquence finie d'opérations de généralisation. Une dérivation de G à H est un chemin, puisque les opérations de généralisation sont des opérations unaires. On a immédiatement la propriété : G est une spécialisation de H ($G \leq H$) *ssi* H est une généralisation de G. La relation $\leq$ sur les graphes conceptuels pourra donc être comprise indifféremment en termes de généralisation ou de spécialisation.

3.3.3. *Projection*

Une projection est un morphisme de graphes (cf. 5.1). Elle correspond à une vision globale d'une séquence de dérivation, c'est-à-dire : on peut dériver G à partir de H *ssi* il existe une projection de H dans G. C'est l'opération fondamentale du modèle, qui permet le calcul effectif de la relation de spécialisation sur les graphes conceptuels. Pratiquement toutes les autres opérations utilisées s'expriment en termes de projection.

Définition. Une *projection* d'un graphe conceptuel $H=(R_H, C_H, U_H, étiq_H)$ dans un graphe conceptuel $G = (R_G, C_G, U_G, étiq_G)$ est un couple d'applications $\prod=(f,g)$, $f : R_H \rightarrow R_G$, $g : C_H \rightarrow C_G$, qui :
1. conserve les arêtes et la numérotation des arêtes :
Pour tout arête rc de U_H, $f(r)g(c)$ est une arête de U_G. De plus, si $c = H_i(r)$, alors $g(c) = G_i(f(r))$.
2. peut restreindre les étiquettes des sommets
Pour tout sommet r de R_H, $étiq_G(f(r)) \leq étiq_H(r)$
Pour tout sommet c de C_H, $étiq_G(g(c)) \leq étiq_H(c)$

Dans la suite de cet article, la décomposition de $\prod$ en f et g sera implicite. On notera $\prod(H)$ le sous-graphe de G image de H par $\prod$. Il est à noter que, si H est connexe, $\prod(H)$ est forcément connexe. Il est également immédiat de vérifier que la composition de deux projections est une projection.

Théorème 1. *Etant donnés G et H deux graphes conceptuels, il existe une projection de H dans G si et seulement si* $G \leq H$.

Schéma de preuve (repose sur [Sowa 84] pour $\Leftarrow$ et [Chein Mugnier 92] pour $\Rightarrow$). La preuve de [Chein Mugnier 92] reste valide, même si le modèle de base présenté ici est une extension du modèle présenté dans [Chein Mugnier 92]. Nous ne rappelons que le canevas de la preuve.
$\Leftarrow$ Soit une dérivation de G où H apparait. A tout chemin de H à G on peut associer une projection de H dans G, car à tout arc de G_i à G_{i+1} on peut associer une projection "élémentaire" de G_i dans G_{i+1}, et la composition de deux projections, respectivement de G_i dans G_j, et de G_j dans G_k, est une projection de G_i dans G_k.
$\Rightarrow$ Soit $\prod$ une projection de H dans G. On construit une dérivation de H à $\prod(H)$ (c'est un chemin) par une séquence de restrictions (restreindre l'étiquette de tout

sommet à celle de son image par $\prod$; on obtient G_1), puis de joints internes (fusionner les sommets de G_1 ayant même image par $\prod$; on obtient G_2), puis de simplifications (les sommets relations de G ayant même image par $\prod$ correspondent à des sommets jumeaux de G_2; on obtient $G_3=\prod(H)$). Soit G' le graphe conceptuel obtenu en considérant $(G \setminus \prod(H))$, puis en complétant les arêtes pendantes avec leurs sommets extrémités dans $\prod(H)$. On effectue la somme disjointe de $\prod(H)$ et de G', ceci donne G_4, puis des joints internes entre sommets de G_4 correspondant aux sommets extrémités des arêtes pendantes de $(G \setminus \prod(H))$. On obtient G. ❑

En prenant pour H et G les graphes de la figure 7, il existe deux projections de H dans G. Les graphes images de H par chacune de ces projections sont donnés dans la figure 9. Si l'on suppose que H est une question portant sur un lieu où apprendre à pêcher et pratiquer une activité de pleine nature, le graphe G, qui décrit le parcours de pêche de l'étang du Barrandon, apporte des réponses de deux sortes : G_1 —la pêche à la mouche au fouet étant à la fois une sorte de pêche et une activité de pleine nature, on peut faire d'une pierre deux coups— et G_2.

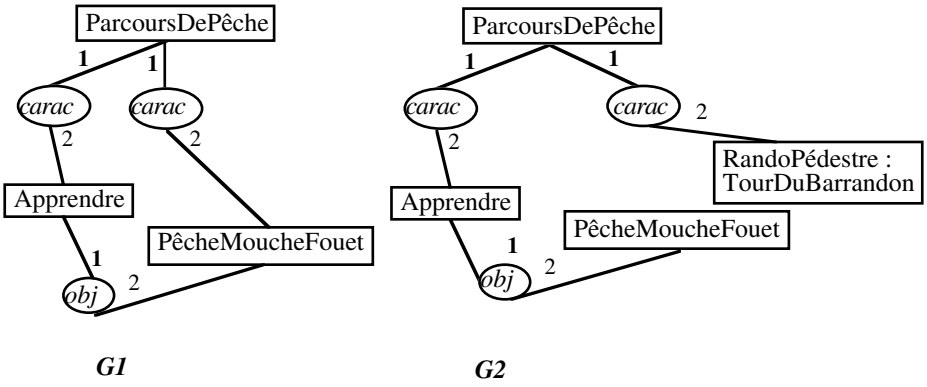
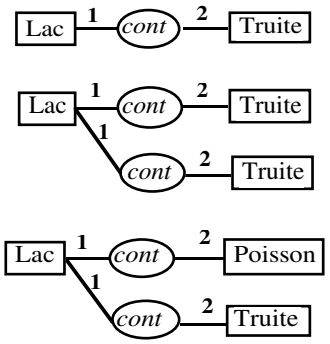


Figure 9. *Graphes images de H dans G (cf. Figure 7).*

3.3.4. *Structure de la relation ≤*

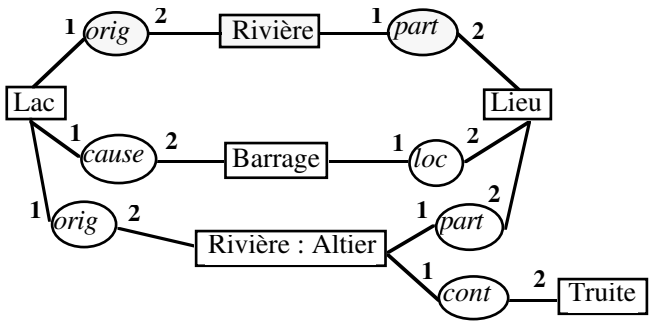
Rappelons que la relation ≤ n'est pas un ordre, mais un *préordre* : elle est réflexive, transitive, mais pas antisymétrique. On dira que deux graphes G et H sont équivalents si $G \leq H$ et $H \leq G$. Ainsi, les graphes de la Figure 10 sont équivalents.



"Un lac contenant (au moins) une truite"
"Un lac contenant (au moins) une truite et contenant (au moins) une truite"
"Un lac contenant (au moins) une truite et contenant (au moins) un poisson"

Figure 10. *Graphes équivalents.*

Afin de caractériser l'ordre quotient de la relation ≤, [Chein Mugnier 92] introduit la notion de graphe *irredondant* : un graphe *G* est irredondant s'il n'est pas équivalent à l'un de ses sous-graphes stricts (i.e. un sous-graphe différent de*G)*. Etant donné qu'il existe une projection triviale (l'identité) de tout sous-graphe de *G* dans *G, G* est irredondant *ssi G* ne se projette pas dans l'un de ses sous-graphes stricts (cf. [Chein Mugnier 92], [Mugnier 94] pour plus de détails sur cette notion). Des trois graphes de la figure 10, seul le premier est irredondant. La Figure 11 donne un exemple de graphe non irredondant plus complexe.



Le graphe se projette dans son sous-graphe formé par la partie non grisée.
La partie grisée est donc "redondante".

Figure 11. *Graphe non irredondant.*

Propriété 2 [Chein Mugnier 92]. *Toute classe d'équivalence pour ≤ admet un unique graphe irredondant (c'est d'ailleurs le plus petit graphe de la classe, au sens de la taille).*

La relation ≤ est un ordre sur l'ensemble des graphes irredondants, et c'est aussi un treillis[5].

Propriété 3. *La relation ≤ est un treillis sur l'ensemble des graphes irredondants.*

Preuve. Soient deux graphes conceptuels irredondants G_1 et G_2. Soit le graphe (G_1+G_2), obtenu par somme disjointe de G_1 et G_2. La propriété suivante est immédiate : pour tout graphe conceptuel K, $K \leq (G_1+G_2)$ *ssi* $K \leq G_1$ et $K \leq G_2$. Il existe un graphe irredondant unique équivalent à (G_1+G_2). L'ensemble des graphes irredondants ordonné par ≤ forme donc un inf-demi-treillis. Cet ensemble possède un plus grand élément, le graphe [⊤: ∗], sommet isolé générique de type universel. C'est donc un treillis. ❑

Cette structure de treillis est importante dès que l'on s'intéresse à trouver les généralisations communes minimales à deux ou plusieurs graphes (ou leurs spécialisations maximales). Toutefois, il faut noter qu'il n'existe pas d'algorithme polynomial pour mettre un graphe sous forme irredondante (voir 5.1). Ainsi, si, partant de deux graphes irredondants, on sait calculer facilement un graphe *équivalent* à leur borne inférieure, cela ne résoud pas le problème du calcul effectif de leur borne inférieure.On sait également calculer un graphe équivalent à la borne supérieure de deux graphes par une opération de produit de graphes[6], ce qui ne résoud pas non plus le problème du calcul effectif de la borne supérieure.

3.4. *Une sémantique logique*

A tout graphe conceptuel G, on associe une formule de la logique du premier ordre, notée $\Phi(G)$ [Sowa 84].

A tout sommet concept on associe un terme, variable ou constante, de la façon suivante. L'ensemble des variables est en bijection avec l'ensemble des sommets concepts génériques. L'ensemble des constantes est en bijection avec l'ensemble des marqueurs individuels (ainsi, à des sommets ayant même marqueur individuel est associée la même constante).

A tout type de concept ou de relation, on associe un prédicat : prédicat unaire pour les types de concepts, pour les types de relation prédicat ayant même arité que le type. Il y a bijection entre l'ensemble (T_R+T_C) et l'ensemble des prédicats.

---

[5] Un ensemble X muni d'un ordre est un treillis si (1) toute paire d'éléments possède une borne inférieure, i.e. un unique minorant commun maximal et (2) toute paire d'éléments possède une borne supérieure, i.e. un unique majorant commun minimal. Si (1) est vérifié, X est un inf-demi-treillis. Si (2) est vérifié, X est un sup-demi-treillis. Par la suite, on utilisera les propriétés suivantes : un inf-demi-treillis possédant un plus grand élément est un treillis, et un sup-demi-treillis possédant un plus petit élément est un treillis.

[6] Cette opération est définie dans [Maurer et al. 81] pour les morphismes de graphes (cf. 5.1 pour le lien entre morphisme et projection). On peut l'étendre aux graphes étiquetés.

On construit la formule $\Phi(G)$ de la façon suivante : A tout sommet est associé un atome. A un sommet concept *c,* on associe l'atome $t_c(id_c)$, où t_c est le prédicat associé au type de c, et id_c est le terme associé à c. A un sommet relation *r,* on associe l'atome $t_r(id_{c1} \ldots id_{cp})$, où t_r est le prédicat associé au type de r, p est l'arité de ce prédicat, et idc_i désigne le terme associé au $i^{ème}$ voisin de r dans G. On obtient $\Phi(G)$ en faisant la conjonction de ces atomes, puis en fermant existentiellement la formule. On associera par exemple aux graphes de la figure 7 les formules suivantes :

$\Phi(H) = \exists x\ \exists y\ \exists z\ \exists t$ Lieu(x) Apprendre(y) $\wedge$ ActivitéDePleineNature(z) $\wedge$ Pêche(t) $\wedge$ carac(x,y) $\wedge$ carac(x, z) $\wedge$ obj(y,t)

$\Phi(G) = \exists x\ \exists y\ \exists z$ ParcoursDePêche(x) Apprendre(y) $\wedge$ PêcheMoucheFouet(z) $\wedge$ carac(x,y) $\wedge$ carac(x, z) $\wedge$ obj(y,z) $\wedge$ RandoPédestre(TourDuBarrandon) $\wedge$ carac(x,TourDuBarrandon) $\wedge$ Etang(Barrandon) $\wedge$ loc(x,Barradon).

Au support S on associe également un ensemble de formules $\Phi(S)$ résultant d'une interprétation logique des liens *Sorte De* entre types de concepts et entre types de relations. Ainsi :

(1) Pour tous t_1 et t_2 de T_C, tels que t_1 couvre[7] t_2 dans T_C, on considère la formule :

$\forall x\ t_2(x) \rightarrow t_1(x)$, qui correspond à l'interprétation des liens *Sorte de* entre types de concepts

(2) Pour tous t_{r1} et t_{r2} de T_{Rp} (p étant l'arité) tels que t_{r1} couvre t_{r2} dans T_R, on considère la formule :

$\forall x_1 \ldots \forall x_p\ t_{r2}(x_1 \ldots x_p) \rightarrow t_{r1}(x_1 \ldots x_p)$, qui correspond à l'interprétation des liens *Sorte de* entre types de relations

(3) $\forall x\ \top(x)$, interprétation du type universel

(4) $\forall x\ \bot(x)$, interprétation du type absurde

(5) Pour tout t_r de T_R, soit $\sigma(t_r) = (t_1 \ldots t_p)$,

$\forall x_1 \ldots \forall xp\ t_r(x_1 \ldots x_p) \rightarrow t_1(x_1) \wedge \ldots \wedge t_p(x_p)$, interprétation des signatures de relations

(6) Pour tout a de I on a : $\tau(a)(a)$

Les axiomes (3, 4, 5, 6) ne servent pas pour la preuve de complétude des opérations du modèle de base, mais ils sont nécessaires pour la sémantique ensembliste que nous proposons en annexe.

---

[7] c'est-à-dire $t_2 \prec t_1$ *et* il n'existe pas de type t_3, tel que $t_2 \prec t_3 \prec t_1$. En d'autres termes, on associe une formule à tout arc de la réduction transitive de T_C.

Les règles de généralisation forment un ensemble consistant de règles d'inférences, c'est-à-dire :

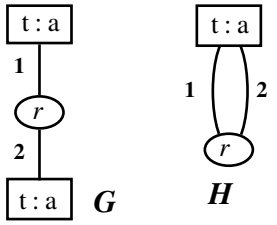
Propriété 4 (Consistance de Φ). *Soit $\Phi(S)$ l'ensemble des formules associées au support. Soient G et H deux graphes conceptuels. Si $G \leq H$ alors $\Phi(S), \Phi(G) \vdash \Phi(H)$.*

Preuve. Il est immédiat d'étendre la preuve de [Chein Mugnier 92] qui s'appliquait à des graphes connexes, et reprenait le schéma de preuve de [Sowa 84]. ❑

Réciproquement, considérons l'ensemble des formules logiques qui peuvent être associées à des graphes conceptuels, que l'on notera $\mathcal{F}$. Soient g et h deux formules de $\mathcal{F}$, associées aux graphes conceptuels G et H. La propriété "Si $\Phi(S), g \vdash h$, alors $G \leq H$" est vérifiée, à une restriction près sur la forme de G et H. G et H doivent être des graphes sous forme normale, au sens suivant :

Définition. Un graphe conceptuel G est sous *forme normale* si l'application *marqueur* : $T_C \rightarrow I \cup \{*\}$ est *injective* sur l'ensemble des sommets individuels de G. Mettre un graphe conceptuel sous forme normale consiste à fusionner ses sommets concepts ayant même marqueur individuel.

Lorsque les graphes ne sont pas sous forme normale, il existe des déductions sur les formules logiques associées, qui ne correspondent pas à des projections. Cf. les figures 12.a et 12.b. Cette condition préalable sur les graphes conceptuels considérés n'était pas explicitement présente dans [Chein Mugnier 92][8].

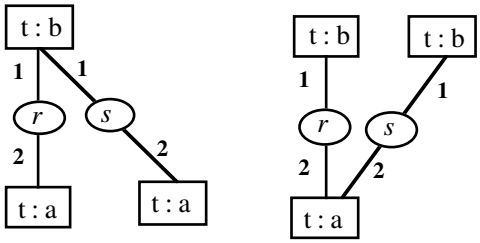


$\Phi(G) = t(a) \wedge t(a) \wedge r(a,a)$ $\quad$ $\Phi(H) = t(a) \wedge r(a,a)$

$\Phi(G)$ et $\Phi(H)$ sont des formules équivalentes, mais G et H ne sont pas équivalents : $G \succ H$

Figure 12.a. *Nécessité de forme normale.*

8 La notion de forme normale était évoquée au début de l'article (p. 376), et complètement absente dans la suite. L'exemple de la Figure 14.b est de Anne Preller, qui a noté que la preuve de complétude n'est pas valable si les graphes ne sont pas sous forme normale.



G ***H***

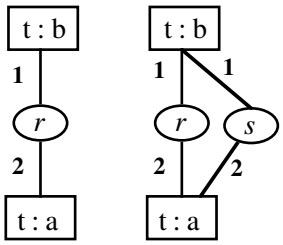
$\Phi(G) = t(b) \wedge t(a) \wedge t(a) \wedge r(a,b) \wedge s(a,b),$

$\Phi(H) = t(b) \wedge t(b) \wedge t(a) \wedge r(a,b) \wedge$ s(a,b)

$\Phi(G)$ et $\Phi(H)$ sont des formules équivalentes, mais G et H sont incomparables.

Figure 12.b. *Nécessité de forme normale.*

Notons que si l'on se limite à des graphes sous forme normale, les opérations élémentaires de spécialisation ne sont plus des opérations internes. De plus, étant donnés G et H deux graphes sous forme normale, tels que $H \leq G$, il n'existe pas forcément de dérivation de G à H n'impliquant que des graphes sous forme normale (Figure 13).



G ***H***

Toute dérivation de G à H contient un graphe qui n'est pas sous forme normale.

Figure 13. *Dérivation de graphes sous forme normale.*

Théorème 2 (Consistance et complétude de Φ). *Soit $\Phi(S)$ l'ensemble des formules associées au support. Soient G et H deux graphes conceptuels simples sous forme normale, alors : $G \leq H$ si et seulement si $\Phi(S)$, $\Phi(G) \vDash \Phi(H)$.*

Schéma de preuve (cf. [Chein Mugnier 92]). $\Rightarrow$ Voir la propriété 4. $\Leftarrow$ On définit une notion de "substitution" d'une formule h dans une formule g (application de l'ensemble des termes de h dans l'ensemble des termes de g) telle que : quels que soient les graphes G et H, il existe une substitution de $h = \Phi(H)$ dans $g = \Phi(G)$ *ssi* il existe une projection de H dans G (la partie $\Rightarrow$ est fausse si les graphes ne sont

pas sous forme normale). On montre ensuite que $\Phi(S), g \preceq h$ *ssi* il existe une "substitution" de h dans g. Le sens $\Leftarrow$ est immédiat. Le sens $\Rightarrow$ se démontre en utilisant la méthode de résolution;$\Phi(S), g \preceq h$ peut se reformuler par : la formule $(S_1 \ldots \wedge S_k \wedge g \wedge h)$ est insatisfiable, où $\Phi(S) = \{S_1 \ldots S_k\}$. La mise sous forme clausale de $(S_1 \ldots \wedge S_k \wedge g \wedge h)$ donne l'ensemble de clauses C, contenant les clauses suivantes :

• $(p(x_1 \ldots x_p) \vee q(x_1 \ldots x_p))$ pour toute formule S_i provenant de l'interprétation des liens *Sorte de* (p=1 si S_i provient de T_C),

—les formules associées aux axiomes 3, 4, 5, 6 de l'interprétation logique du support sont inutiles—

• $(p(a_1 \ldots a_q))$ pour tout atome de g, où tout a_i est une constante de g ou une constante de skolem

• $(p_1(x_{i1} \ldots x_{ip}) \vee \ldots \vee p_n(x_{j1} \ldots x_{jq}))$ pour h (n étant le nombre d'atomes de h).

La méthode de résolution étant complète, la clause vide se dérive de C. Pour ce faire, il faut "vider" la clause associée à h, ce qui se fait en n applications de la règle de résolution (la règle de résolution est aussi utilisée au besoin pour dériver de nouveaux atomes à partir des clauses singletons de f et des clauses provenant du support). Considérons l'ordre sur les atomes de h induit par l'ordre des applications. Soit h_i la formule h réduite à ses i premiers atomes, $1 \leq i \leq n$. On montre par récurrence sur i, qu'il existe une substitution de h_i dans g, qui s'obtient en étendant la substitution de h_{i-1} dans g grace à l'unificateur utilisé pour "supprimer" le $i^{ème}$ atome de la clause associée à h. Pour $i = n$, il existe une substitution de h dans g, donc une projection de H dans G. ❑

4. Compléments sur le modèle des graphes simples

Nous présentons dans cette partie des aspects qui sont relatifs à des utilisations particulières du modèle. D'abord trois restrictions du modèle, que l'on retrouve fréquemment dans les travaux sur les graphes conceptuels : les graphes sont *connexes*, l'ensemble des types de concepts forme un *treillis*, la projection est nécessairement *injective*. Ensuite, la notion de *base canonique*, généralisant la notion de "graphes étoiles", et la notion de *lambda-abstraction*, qui permet d'exprimer des connaissances de différente nature.

4.1. *Quelques cas particuliers usuels*

4.1.1. *Connexité des graphes conceptuels*

Les graphes conceptuels simples de [Sowa 84] sont connexes, car "two parts that were not connected would simply be called two conceptual graphs". Dans cet article, la suppression de la contrainte de connexité simplifie la présentation du modèle de base et des graphes emboités (tant au niveau des définitions qu'au niveau des

propriétés). Ce choix n'a qu'une importance technique, et non sémantique. Les deux définitions sont cohérentes avec la sémantique intuitive suivante : si on asserte deux graphes connexes G_1 et G_2, on asserte le graphe $G_1 + G_2$, et réciproquement. Si un graphe G a pour composantes connexes G_1, G_2, ..., G_k, G consiste en $G_1 + G_2 \ldots + G_k$. Lorsqu'on passe aux formules logiques, ce "+" correspond à la conjonction.

Quelles conséquences la contrainte de connexité induit-elle sur le modèle des graphes simples? Au niveau des opérations élémentaires de spécialisation, la somme disjointe disparait. Elle est remplacée par l'opération de joint externe (cf. 3.3.1). Au niveau des opérations de généralisation, la décomposition devient partie intégrante de l'éclatement : si après éclatement d'un sommet concept, le graphe n'est plus connexe, on ne garde qu'une composante connexe.

En ce qui concerne la sémantique logique, le théorème 2 (complétude de Φ) reste bien sûr applicable, mais, si on l'applique à des graphes connexes, sa puissance est restreinte. En effet, si les graphes sont connexes, la conjonction de deux formules de $\mathcal{F}$ n'est plus nécessairement une formule de $\mathcal{F}$ ($\mathcal{F}$ désignant maintenant l'ensemble des formules logiques associées à des graphes connexes). Ainsi, étant donnés trois formules g_1, g_2 et h, respectivement associées aux graphes G_1, G_2 et H, il se pourrait a priori que h se déduise de g_1 et g_2, sans qu'il n'existe de projection de G_1 dans H, ni de G_2 dans H. [Chein Mugnier 92] montre qu'il n'en est rien, si les formules satisfont à une condition dite de séparation : elles n'ont pas de constante en commun. On a alors :

Théorème de complétude [Chein Mugnier 92]. *Soit $F = \{g_1 \ldots g_k\}$ un sous-ensemble de $\mathcal{F}$, 2 à 2 séparées, et $G = \{G_1 \ldots G_k\}$ l'ensemble des graphes conceptuels sous forme normale associés. Soit h une formule de $\mathcal{F}$ associée à un graphe conceptuel H. Si $\Phi(S)$, $g_1 \ldots g_k \leq h$, alors il existe i, $1 \leq i \leq k$, tel que $\Phi(S)$, $g_i \leq h$ (il existe donc une projection de H dans G_i).*

Les formules $g_1 \ldots g_k$ ont des ensembles de constantes deux à deux disjoints. En d'autres termes, elles représentent un ensemble de graphes conceptuels qui ne partagent pas de marqueur individuel (et puisque ces graphes sont sous forme normale, tout marqueur individuel apparait au plus une fois dans l'ensemble). Cette contrainte de séparation n'est pas une réelle limitation : soit on l'impose au niveau des graphes conceptuels, il suffit alors de "fusionner" les marqueurs individuels communs; soit on l'impose au niveau des formules, et l'on remplace les formules ayant une constante en commun par une formule équivalente à la conjonction de ces formules, qui est bien une formule de $\mathcal{F}$.

4.1.2. *Treillis des types de concept*

L'ensemble des types de concepts est un treillis dans [Sowa 84]. La justification de cette structure de treillis, sur un plan conceptuel, pourrait être la suivante : s'il existe un type t_1 et un type t_2, alors il existe forcément un type correspondant à la conjonction de t_1 et t_2 et un type correspondant à la disjonction de t_1 et t_2 (dans une

interprétation ensembliste, ces types correspondent à l'intersection et à l'union des interprétations de t_1 et t_2).

On peut ainsi considérer que T_C est le treillis engendré par la fermeture d'un ensemble de types, par les opérations de conjonction (intersection) et disjonction (union). On peut aussi, c'est notre point de vue, considérer que T_C est un sous-ensemble de ce treillis "virtuel", tous les types concevables n'étant pas forcément pertinents pour une application donnée. T_C est alors un *ordre partiel* qui n'est pas forcément un treillis.

D'un point de vue algorithmique, la structure de treillis sur T_C peut être importante, ne serait-ce que pour limiter la combinatoire des choix possibles. C'est le cas si l'on s'intéresse à la spécialisation commune maximale de deux étiquettes de concepts (pour calculer un "joint maximal" [Sowa 84]) ou à la généralisation commune minimale de deux étiquettes de concepts (en apprentissage par exemple). En effet, si T_C est un treillis, on obtient un treillis sur les étiquettes en faisant le produit de $T_C \setminus \{\bot\}$ et de $I \cup \{*\}$, et en ajoutant un élément minimal, correspondant à l'étiquette absurde. Si l'on supprime de ce treillis les étiquettes (t,m), où m est un marqueur individuel et $\tau(m) \neq t$, on garde un treillis.

Propriété 5. *Si T_C est un treillis, alors l'ensemble des étiquettes de concept valides, muni d'un plus petit élément ("l'étiquette absurde"), forme un treillis.*

Preuve. Soient $e_1 = (t_1, m_1)$ et $e_2 = (t_2, m_2)$ deux étiquettes valides différentes. Ou bien l'un des deux marqueurs est générique, ou bien m_1 et m_2 sont deux marqueurs individuels différents, donc incomparables. L'étiquette $(t_1 \vee t_2, *)$ est l'unique plus petit majorant de e_1 et e_2. L'ensemble des étiquettes valides forme donc un sup-demi treillis. Si on lui adjoint un plus petit élément (i.e. l'étiquette absurde), on obtient un treillis.

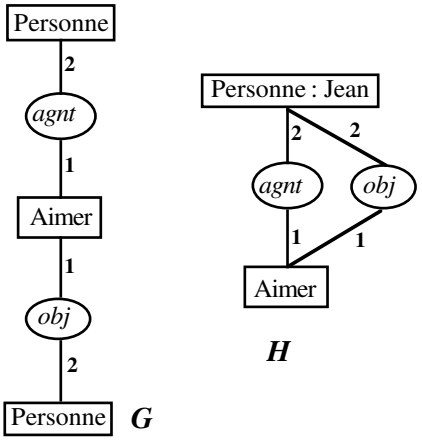
4.1.3. *Projection injective*

Certains systèmes de traitement des graphes conceptuels sont basés sur une notion de projection *injective* (par exemple [Fargues et al. 86] ou [Levinson Ellis 91]). Dans [Sowa 84], il y a un flou sur cette notion. Prendre la projection injective comme opération de base pour le raisonnement sur les graphes conceptuels induit des changements importants dans le modèle, et ceci à plusieurs points de vue.

Définissons la relation de spécialisation suivante : G est une spécialisation de H *ssi* il existe une projection *injective* de H dans G. Autrement dit, H est isomorphe à un sous-graphe de G, cet isomorphisme associant à un sommet de H un sommet d'étiquette inférieure ou égale. On notera $G \leq_i H$.

D'un point de vue sémantique, $\leq_i$ réduit considérablement les "comparabilités" entre graphes conceptuels. En effet, deux sommets concepts distincts représentent nécessairement des entités distinctes (Figure 14). Ceci entraine en particulier que les graphes soient sous forme normale (*i.e.* tout graphe comporte au plus un marqueur individuel donné; cf. 3.4). Du point de vue de la sémantique logique, on traduit la

condition d'injectivité en ajoutant *(x ≠ u),* pour toute variable *x* et toute variable ou constante *u*, ce qui est donc restrictif.



H ≤ *G,* mais pas *H* ≤*i G.*

Figure 14. *Projection injective.*

Du point de vue des opérations de graphes, la correspondance entre l'opération globale de morphisme, ici la projection injective, et des règles élémentaires de spécialisation disparait. La notion d'injectivité ne semble pas pouvoir se décomposer en opérations *locales*.

Les deux relations $\leq$ et $\leq_i$ n'induisent pas la même structure : $\leq_i$ est un ordre, autrement dit deux graphes *G* et *H* tels que $G \leq_i H$ et $H \leq_i G$ sont nécessairement isomorphes. Cet ordre ne possède pas la propriété d'être un treillis, quelle que soient les contraintes mises sur les ensembles de types de concepts et de relations. Ces relations, $\leq$ et $\leq_i$, diffèrent aussi d'un point de vue algorithmique, ce que nous n'aborderons pas dans cet article (cf. [Mugnier 94]).

En définitive, les différences entre les deux relations de spécialisation sont telles, qu'elles conduisent à deux modèles différents. Nous pensons donc qu'il faut garder comme notion de base la projection (quelconque). Toutefois, en pratique, il est souvent nécessaire de pouvoir dire que deux sommets concepts représentent des entités nécessairement distinctes. Une solution ad hoc consiste à ajouter un type de relation binaire, par exemple DIF, de signature $(\top,\top)$. Pour tous sommets concepts individuels c_1 et c_2, on considère que l'on a (explicitement ou implicitement) les relations $DIF(c_1, c_2)$ et $DIF(c_2, c_1)$. Les possibilités d'intégration de cette contrainte de différence par extension du modèle de base lui-même n'ont pas été étudiées à notre connaissance.

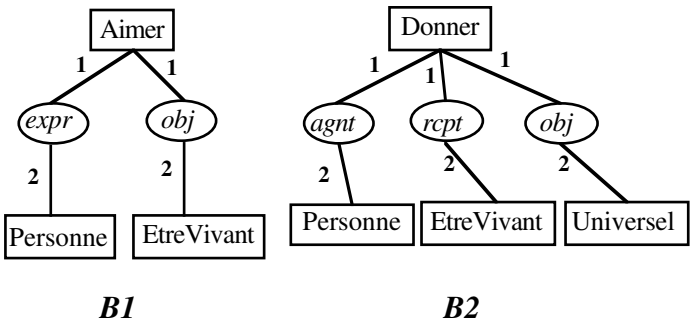
4.2. *Graphes canoniques*

Une base canonique est un ensemble de graphes conceptuels *connexes*, utilisé comme générateur de graphes conceptuels. Elle permet d'imposer des contraintes de construction supplémentaires, qui sont conservées par les règles de spécialisation.

Définition. Une *base canonique B* associée à un support *S* est un ensemble fini de graphes conceptuels connexes sur S. L'ensemble des *graphes canoniques* relativement à *B* est la fermeture de *B* par les règles élémentaires de spécialisation. On peut aussi définir cet ensemble de façon équivalente par : *G* est un graphe canonique relativement à *B* s'il existe une dérivation de *G* dont toutes les sources sont des graphes de *B*.

Cette notion de base canonique est une généralisation de $B_0 \cup \{[\top : *]\}$. Autrement dit, $B_0 \cup \{[\top : *]\}$ définit les contraintes minimales de formation des graphes conceptuels. A un support peuvent être associées plusieurs bases canoniques.

Plaçons nous, par exemple, dans le cadre d'une application où les objets à manipuler sont représentés par des graphes conceptuels, et correspondent à des instanciations de prototypes qui peuvent être combinées entre elles par les opérations de spécialisation. Les prototypes correspondront aux éléments de la base canonique, et les objets devront être des graphes canoniques relativement à cette base.



B1 ***B2***

Ces deux graphes sont supposés être définis sur un même support *S*.

Figure 15. *Une base canonique à deux éléments.*

Les opérations élémentaires de spécialisation conservent la canonicité : si un graphe *G* se dérive d'un ensemble de graphes canoniques, alors *G* est canonique. En revanche, les règles de généralisation ne conservent pas la canonicité, à cause de l'opération d'éclatement. L'équivalence entre les notions de dérivation par spécialisation et de projection est conservée.

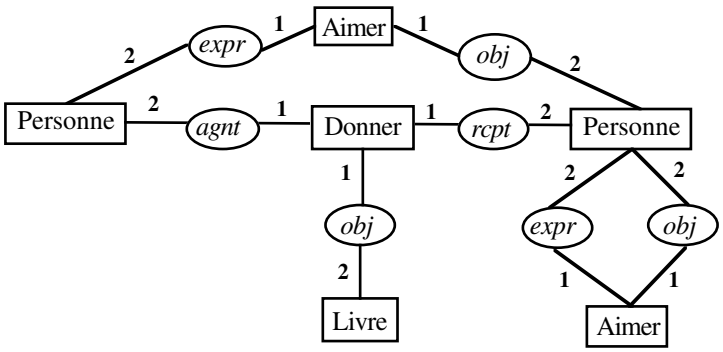
Théorème 3. *Etant donnés G et H deux graphes canoniques relativement à une base B, il existe une projection de H dans G ssi G ≤ H (toute dérivation de G à H ne devant comporter que des graphes canoniques).*

Indication de preuve. Par rapport au théorème 1, la partie ⇒ de la preuve est à reconsidérer car *G'* n'est pas forcément un graphe conceptuel canonique. Voir [Mugnier Chein 93] ou [Mugnier 94]. ❑

Cette équivalence nous donne une définition des graphes canoniques en termes de projection. Une *couverture* d'un graphe *G* est un ensemble de sous-graphes connexes de *G* tel que toute arête et tout sommet de *G* appartiennent à l'un des sous-graphes.

Propriété 6. [Mugnier Chein 93] *Un graphe conceptuel G est canonique relativement à une base B ssi il existe une couverture de G par des projections d'éléments de B (ie des sous-graphes $\prod(B_i)$, tel qu'il existe une projection $\prod: B_i \rightarrow G$, $B_i \in B$).*

Si l'on excepte le cas des graphes ayant une composante connexe réduite à un sommet, la condition ci-dessus peut être reformulée ainsi : pour tout sommet relation *r* de *G*, il existe une projection $\prod : B_i \rightarrow G$, où $B_i \in B$, telle que $r \in \prod(B_i)$.



Il existe une couverture de ce graphe par deux projections de B_1 et une projection de B_2.

Figure 16. *Graphe canonique relativement à la base de la figure* 15

4.3. *Expression de connaissances de différente nature*

Un support permet de représenter "l'ontologie" d'un domaine, puisqu'il décrit les types de concept et de relation. Un graphe simple sur un support *S* permet de représenter des faits, des buts et des hyptohèses concernant le domaine dont les connaissances générales ont été représentées par *S*.

Sowa a proposé d'appeler *λ-abstraction n-aire* un graphe conceptuel simple dans lequel on a privilégié $n \geq 1$ sommets concepts génériques. Une telle expression est notée $G\ (c_1,c_2,...c_n)$. Dans le cas où $n = 1$, on appelle *tête* de G le sommet concept privilégié. En utilisant des λ-abstractions unaires, on peut définir, par genre et différence, des nouveaux types de concept [Sowa 84] [Leclère 95]. Des λ-abstractions n-aires permettent de définir des types de relations n-aires. L'ensemble constitué par un support et les définitions de types est analogue à la "T-Box" des logiques terminologiques [Woods Schmolze 92], un ensemble de graphes simples pouvant alors représenter une "A-Box". Les λ-abstractions permettent également de représenter d'autres types de connaissances, dont nous donnons quelques exemples ci-après.

A un type de concept t on peut vouloir associer des *contraintes locales*. C'est-à-dire exprimer le fait que dans tout graphe satisfaisant une telle contrainte, l'environnement de tout sommet concept de type inférieur ou égal à t est conforme à cette contrainte. Ceci peut permettre de représenter notamment des contraintes fonctionnelles. Supposons par exemple que nous voulions exprimer le fait que toute rivière qui a une origine a une origine unique, donc que tout sommet concept de type inférieur ou égal à *Rivière* est lié à au plus un sommet relation de type *orig*. Ceci peut s'exprimer formellement de la manière suivante.

Définition. Soit $C(c)$ une λ-abstraction unaire avec $type(c) = t$. Un graphe conceptuel G est *bien formé* relativement à C si pour tout sommet concept x de type inférieur ou égal à t il existe au plus une projection de C dans G telle que la tête de C ait x pour image.

Avec l'exemple ci-dessus le graphe C serait [**Rivière**]—1—(*orig*)—2—[Lieu], la tête étant le sommet en gras.

On peut associer d'autres sortes de connaissances à des types de concept. Notamment des *connaissances nécessaires implicites* exprimant le fait que si dans un graphe on a un sommet concept de type inférieur ou égal à t alors ce sommet a nécessairement un graphe particulier dans son environnement. On peut par exemple vouloir dire qu'un match de volley-ball implique l'existence d'un ballon. Une manière de faire est d'associer au type *MatchVolley* l'abstraction unaire suivante : [**MatchVolley**]—1—(*Instr*)—2—[Ballon]. Ayant un graphe conceptuel G représentant un fait on peut alors l'enrichir en faisant un joint sur tous les sommets de type inférieur ou égal à *MatchVolley* avec le graphe ci-dessus. Une connaissance (nécessaire) implicite liée à un type peut donc se représenter par une lambda-abstraction unaire dont la tête est de type t.

On peut également définir des *règles*, au sens règles de production, sur des graphes (cf. [Fargues et al. 86] [Ghosh Wuwongsee 94]). Une *règle* est définie par un couple de λ-abstractions *k-aires* $G(c_1,c_2,...,c_k)$, $D(d_1,d_2,...,d_k)$, avec, pour tout i, $type(c_i) = type(d_i)$. Une telle règle sera interprétée par : si G alors D. Plus précisément : un graphe conceptuel fait F peut être enrichi par une telle règle de la manière suivante : si $\prod$ est une projection de G dans F, on identifie, pour tout i, le sommet d_i de D avec l'image par $\prod$ dans F du sommet c_i de G.

5. Liens avec la combinatoire

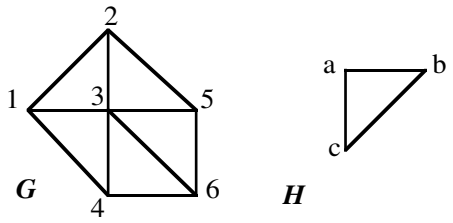
La projection est l'opération de base des graphes conceptuels, puisqu'elle permet le calcul de la relation de spécialisation. Le problème de décision associé (on l'appellera *Projection*) est NP-complet. Dans [Chein Mugnier 92], il est prouvé que des problèmes a priori plus simples sont également NP-complets : équivalence de composantes connexes (étant donnés G et H deux graphes conceptuels connexes, a-t-on $G \leq H$ et $H \leq G$?), redondance (étant donné G un graphe conceptuel, G est-il équivalent à l'un de ses sous-graphes stricts? —c'est-à-dire G se projette-t-il dans l'un de ses sous-graphes stricts?). Cette partie étudie plus en détail l'opération de projection d'un point de vue algorithmique, sous l'angle de ses liens avec deux domaines : la théorie des graphes et les problèmes de satisfaction de contraintes (CSP).

5.1. *Graphes conceptuels et graphes, projection et morphisme.*

L'opération de théorie des graphes sous-jacente à la notion de projection est celle de morphisme. On désigne par X l'ensemble des sommets d'un graphe et par U l'ensemble de ses arêtes.

Définition. Un *morphisme* d'un graphe non orienté $G = (X_G, U_G)$ dans un graphe non orienté $H = (X_H, U_H)$ est une application $f : X_G \rightarrow X_H$ qui conserve les arêtes (pour toute arête xy de U_G, $f(x)f(y)$ est une arête de U_H).

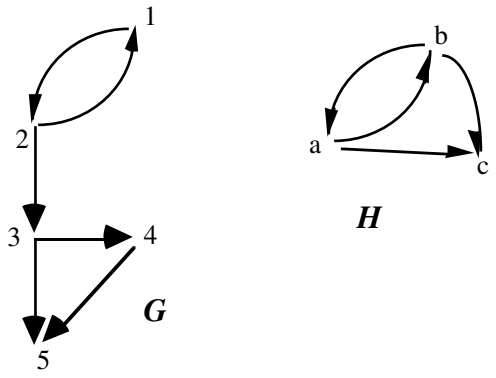
La Figure 17 illustre cette notion. L'exemple pourrait aussi être interprété comme une instance d'un problème de coloration de graphes (étant donné un graphe G non orienté, et un entier positif k, existe-t-il une coloration des sommets de G en k couleurs, i.e. une application de X_G dans $\{1, \ldots, k\}$ tel que deux sommets adjacents n'aient pas la même couleur?). Ici, il s'agirait du problème 3-coloration. Plus généralement, il existe une coloration d'un graphe G en k couleurs *ssi* il existe un morphisme de G dans la clique (graphe non orienté complet) à k sommets.



Il n'existe pas de morphisme de G dans H. Mais si l'on considère G privé de n'importe laquelle de ses arêtes, par exemple l'arête 12, il en existe plusieurs.
Par exemple {(1, a), (3, b), (4, c), (6, a), (5, c), (2,a)}.

Figure 17. *Morphisme de graphes non orientés.*

Si les graphes sont orientés, le sens des arcs doit être conservé (Figure 18). Si les graphes sont étiquetés sur les sommets et/ou sur les arcs, on ajoute des contraintes de correspondance entre sommets et entre arcs, liées aux étiquettes. Si les ensembles d'étiquettes n'ont pas de structure particulière, la contrainte de correspondance est l'égalité. La définition s'étend sans modification aux multigraphes.

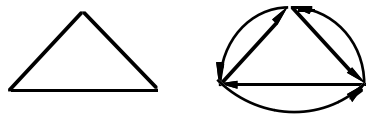


f_1={(1, a), (2, b), (3, a), (4, b), (5, c)} f_2={(1, b), (2, a), (3, b), (4, a), (5, c)}
Il existe 2 morphismes de *G* dans *H* : f_1 et f_2.

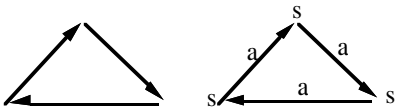
Figure 18. *Morphisme de graphes orientés.*

Ainsi, on appellera *Morphisme* le problème de décision générique suivant :
Données : deux graphes *G* et *H* (tous deux orientés ou non orientés, étiquetés ou non, admettant des multi-arêtes, …)
Question : existe-t-il un morphisme de *G* dans *H* ?

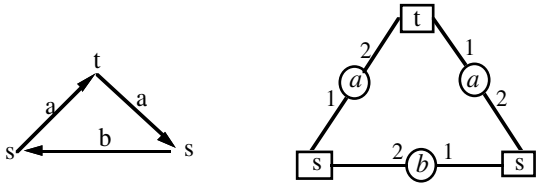
Les graphes conceptuels peuvent être vus comme des graphes non orientés particuliers : étiquetés sur les arêtes et les sommets (et bipartis de surcroît). Les graphes non orientés peuvent également être vus comme des graphes conceptuels particuliers. En effet, un graphe non orienté peut être considéré, du point de vue de l'opération de morphisme, comme un graphe orienté, sans boucle, toute arête étant remplacée par deux arcs symétriques (Figure 19). Autrement dit, étant donnés deux graphes non orientés *G* et *H,* il existe un morphisme de graphes non orientés de *G* dans *H ssi* il existe un morphisme de graphes orientés du graphe orienté associé à *G* dans le graphe orienté associé à *H*. Un graphe orienté est lui même un cas particulier de graphe orienté étiqueté sur les sommets et sur les arcs, tous les sommets (*resp*. arcs) ayant la même étiquette. Enfin, un graphe orienté étiqueté sur les sommets et sur les arcs est un cas particulier de graphe conceptuel (à un arc *(x,y)* est associé un sommet relation binaire, ayant pour premier voisin *x* et pour deuxième voisin *y*, et pour étiquette celle de l'arc).



Un graphe *non orienté* vu comme un graphe *orienté*



Un graphe *non étiqueté* vu comme un graphe *étiqueté*



Un graphe *étiqueté* vu comme un *graphe conceptuel*

Figure 19. *Transformations.*

Tout cas polynomial pour Morphisme, basé sur la *structure* de *G* ou celle de *H,* fournit un cas polynomial pour Projection, si l'on transfère la propriété de structure aux graphes conceptuels. L'étude des cas polynomiaux pour l'opération de morphisme a été particulièrement étudiée dans le cadre du problème suivant *H-coloration* (où *H* est un graphe *fixé*, orienté ou non orienté) :

Donnée : un graphe *G*
Question : existe-t-il un morphisme de *G* dans *H* ?

Ce problème est une généralisation du problème de coloration de graphes, *H* n'étant plus nécessairement une clique. L'étude de la complexité du problème *H* -coloration s'est surtout attachée à la structure de *H.* Ainsi, en ce qui concerne les graphes non orientés, [Hell Nesetril 90] a prouvé que : *H* -coloration est polynomial si *H* est un graphe biparti, ou réduit à un sommet, ou possédant au moins une boucle, et NP-complet dans tous les autres cas. En ce qui concerne les graphes orientés, [Gutjahr et al. 92] et [Hell et al. 93] ont montré qu'il existait des graphes sans cycles *H* (c'est-à-dire des arbres avec une orientation quelconque des arêtes) tel que *H* -coloration est NP-complet. Par conséquent, le problème Morphisme est NP-complet même lorsque *H* est un arbre. [Gutjahr et al. 92] et [Hell et al. 93] recensent les cas polynomiaux de *H* -coloration connus, ceux-ci fournissent tous des algorithmes polynomiaux pour le problème Morphisme.

En ce qui concerne la structure de *G,* le problème Morphisme est polynomial si *G* est un arbre[9] (ce qui a été montré dans plusieurs travaux indépendamment). En ce qui concerne les graphes conceptuels, nous donnons en [Mugnier Chein 92] un algorithme polynomial pour déterminer l'existence d'une projection d'un graphe conceptuel sans cycles (autres que les cycles dus aux multi-arêtes) *G* dans un graphe conceptuel quelconque *H* (voir aussi [Mugnier 94]). Sa complexité est en O(taille(*G*) X taille(*H)*), si l'on considère que la comparaison de deux étiquettes de sommets se fait en temps constant. Cet algorithme fournit en résultat, pour tout sommet de *G*, l'ensemble de ses images possibles par une projection de *G* dans *H*. Il permet également de compter le *nombre de projections* de *G* dans *H,* sans augmentation de complexité.

Il arrive fréquemment dans les problèmes de calcul d'une projection de *G* dans *H* que le graphe *G* soit beaucoup plus petit et plus simple (de par sa structure) que le graphe *H* : c'est le cas notamment lorsque *G* représente une question et *H* un fait, dans le cadre d'un système question/réponse. C'est aussi le cas lorsqu'on cherche à déterminer si un graphe donné est canonique [Mugnier Chein 93]. Ainsi, si nous ne pouvons nous restreindre au cas où *G* est un arbre, nous avons développé un algorithme basé sur l'hypothèse que *G* est un graphe peu complexe (en nombre de cycles notamment), qui utilise l'algorithme de projection d'un arbre dans un graphe comme phase de prétraitement (cf. [Mugnier 94], et [Salvat 93] pour une implémentation).

Les études sur les morphismes citées plus haut ne s'intéressent pas au cas des graphes étiquetés. Lorsque les ensembles d'étiquettes ne sont pas munis de structures particulières, autrement dit deux sommets ou arcs mis en correspondance doivent avoir la même étiquette, le fait que des graphes soient étiquetés ne peut que simplifier le problème. Par exemple, si l'étiquetage est localement injectif (c'est-à-dire injectif sur l'ensemble des voisins de tout sommet, ou, si les arêtes sont étiquetées, sur l'ensemble des arêtes adjacentes à tout sommet), le problème Morphisme est polynomial (cf. [Liquière Brissac 94] pour le cas des graphes conceptuels, lorsque les types de relations ne sont pas ordonnés).

Par contre, que les ensembles d'étiquettes soient munis de structures, d'ordre notamment, peut introduire un niveau de complexité suplémentaire. Exemple : alors que le problème de l'existence d'un isomorphisme entre deux graphes est un problème non classé (et conjecturé non NP-complet), le problème de l'existence d'un isomorphisme entre deux graphes conceptuels, tel que tout sommet a une étiquette *supérieure ou égale* à l'étiquette de son image, est NP-complet (cf. par exemple le problème isoprojection dans [Chein Mugnier 92]).

---

9 [Hell et al. 93] (version révisée, 1994) montre que le problème reste polynomial si *G* est un k-arbre partiel, *k* étant fixé.

5.2. *Projection, morphisme et CSP.*

Il existe une correspondance forte entre un problème de satisfaction de contraintes (CSP, défini juste après) et le problème Projection ou Morphisme[10]. Les problèmes sont non seulement équivalents, mais les ensembles de solution et certains cas polynomiaux se transfèrent de l'un à l'autre. Nous donnons dans cette partie des transformations qui permettent ces tranferts.

Nous nous baserons sur une définition très simple d'un CSP, où les définitions des contraintes sont données en extension, et les contraintes ne sont pas typées. Le terme CSP désignera également le problème de décision associé à l'existence d'une solution à un CSP.

Définition. Un CSP est un quadruplet $P = (X, C, D, R)$, où (cf. Figure 20) :

X est l'ensemble des variables, $X = \{x_1, ..., x_n\}$

$C \subseteq \mathcal{P}(X)$ est l'ensemble des contraintes, $C = \{C_1, ..., C_p\}$

On appelle *graphe des contraintes* le couple (X, C), qui est un hypergraphe dans le cas général, et un graphe non orienté dans le cas de contraintes binaires.

D est l'ensemble des domaines des variables, en bijection avec X, $D = \{D_1, ... , D_n\}$

R est l'ensemble des définitions des contraintes, en bijection avec C, $R = \{R_1, ..., R_p\}$, vérifiant pour toute contrainte C_i : soit $(x_{i1}, ..., x_{iq})$ un ordonnancement quelconque des variables de C_i, alors $R_i \subseteq D_{i1} \times ... \times D_{iq}$.

Une *solution* d'un CSP est une instanciation des n variables satisfaisant les contraintes, c'est-à-dire plus formellement une application

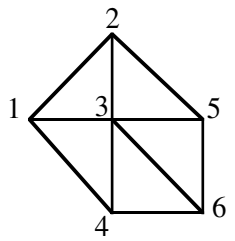
$$\sigma : \quad X \to \bigcup D_i \in D$$
$$x_i \to a \in D_i$$

vérifiant pour toute contrainte $Cj = (x_{j1}, ..., x_{jq})$: $(\sigma(x_{j1}), ..., \sigma(x_{jq})) \in R_j$. (autrement dit, le uplet obtenu en remplaçant chaque variable par son instanciation est un uplet de la définition de la contrainte).

On appellera CSP le problème de décision associé : étant donné un CSP P, P admet-il une solution?

---

[10] La recherche de transformations polynomiales entre Morphisme et CSP est partie de la lecture de [Feder Vardi 93], avec le théorème suivant : "Every constraint-satisfaction problem is polynomially equivalent to a digraph-homomorphism problem", où un problème de satisfaction de contraintes est défini de façon légèrement différente de celle donnée précédemment. La preuve du théorème n'était pas donnée dans le papier —mais suggérée par des exemples.



(X, C)

$\forall$ x $\in$ X, Dx = {a, b, c}

$\forall$ {x, y} $\in$ C, Rxy =

x	*y*
a	*b*
a	*c*
b	*a*
b	*c*
c	*a*
c	*b*

Ce CSP représente une instance du problème "coloration en 3 couleurs". Le graphe des contraintes et le graphe à colorier ont la même structure. Ce CSP n'admet pas de solution (cf. Figure 17 : un morphisme de *G* dans *H* correspond à une solution de ce CSP)

Figure 20. *Un CSP.*

Théorème 4. *Les problèmes* CSP *et* Morphisme de graphes orientés étiquetés *sont polynomialement équivalents, et*
(1) toute solution à l'un est une solution à l'autre
(2) le graphe des contraintes et G ont même graphe non orienté sous-jacent.

La formulation suivante "résume" les transformations d'un problème à l'autre : un morphisme est une application d'un ensemble de sommets vers un autre, qui respecte les arcs; une solution d'un CSP est une application de l'ensemble des variables vers l'ensemble des éléments des domaines, qui respecte les contraintes. Dans la suite de cette partie, nous donnons les transformations (polynomiales) d'un problème à l'autre, il est immédiat de les étendre au problème Projection.

• *Transformation de Morphisme à CSP*

Soit une instance de Morphisme : $G = (X_G, U_G)$ et $H = (X_H, U_H)$, graphes que l'on considérera dans un premier temps comme des graphes orientés, anti-symétriques, sans boucles et sans étiquettes, la transformation étant ensuite étendue.

On construit à partir de *G* et *H* un CSP binaire $P = (X, C, D, R)$ de la façon suivante :

$X = X_G$.

$C = U_G$, en supprimant l'orientation des arcs.

(X, C) correspond donc au graphe non orienté sous-jacent à *G*.

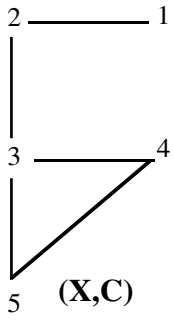
A toute variable x_i, on associe le domaine $D_i = X_H$.

A toute contrainte C_i on associe comme définition $R_i = U_H$.

Lorsqu'on passe à des graphes plus complexes, on ne fait que restreindre les domaines associés aux variables et/ou les définitions des contraintes. Si les sommets sont étiquetés, à toute variable x_i, provenant d'un sommet x_i de X_G, on associe le domaine $D_i = \{y \in X_H \mid étiquette(x_i) = étiquette(y)\}$. Si les arcs sont étiquetés, à toute contrainte $C_i = (x_j , x_k)$, on associe la définition $R_i = \{(y\ z) \in U_H \mid étiquette((x_j, x_k)) = étiquette((y, z))\}$. On restreint le domaine de tout sommet

x_i ayant une boucle à $D_i = \{y \in X_H \mid (y,y) \in U_H\}$. Autrement dit, une boucle peut être vue comme une sorte d'étiquette. Si G possède des arcs symétriques, et plus généralement des multi-arcs étiquetés, alors C n'est plus égal à U_G, mais à $\{(xj, xk) \in X_G \times X_G \mid j < k$ et $(\exists (xj, xk) \in U_H$ ou $\exists (xk, xj) \in U_H)\}$; autrement dit, on ne considère qu'une contrainte par paire d'arcs symétriques, ou multi-arcs; puis à toute contrainte $C_i = (x_j, x_k)$, on associe comme définition $R_i = \{(y\ z) \mid \forall (x_j, x_k) \in U_G, \exists (y, z) \in U_H$ *avec étiquette*$((x_j, x_k)) =$ *étiquette*$((y, z))$, *et* $\forall (x_k, x_j) \in U_G, \exists (z, y) \in U_H$ *avec étiquette*$((x_k, x_j)) =$ *étiquette*$((z, y))\}$

La Figure 21 présente l'instance de CSP construite par la transformation précédente à partir de l'instance de *Morphisme de graphes orientés* de la figure 18.



D1 = D2 = D3 = D4 = D5 = {a, b, c}

1 | 2	2 | 3	3 | 4	3 | 5	4 | 5
a | b	a | b	a | b	a | b	a | b
b | a	b | a	b | a	b | a	b | a
	a | c	a | c	a | c	a | c
	b | c	b | c	b | c	b | c
	R12	**R23**	**R34**	**R35**	**R45**

Figure 21. *Transformation de Morphisme (Figure* 18*) à CSP.*

Cette transformation est polynomiale en la taille de l'instance du problème Morphisme. Il est immédiat de vérifier qu'il existe un morphisme de G dans H *ssi* il existe une solution au CSP P. Les points (1) et (2) du théorème 4 sont trivialement vérifiés.

• *Transformation de CSP à Morphisme*

On considère dans un premier temps un CSP binaire, soit $P = (X, C, D, R)$, que l'on transforme en deux graphes $G = (X_G, U_G, étiq_G)$ et $H = (X_H, U_H, étiq_H)$, instance de Morphisme de graphes étiquetés (sur les arcs, ici).

$X_G = X$.

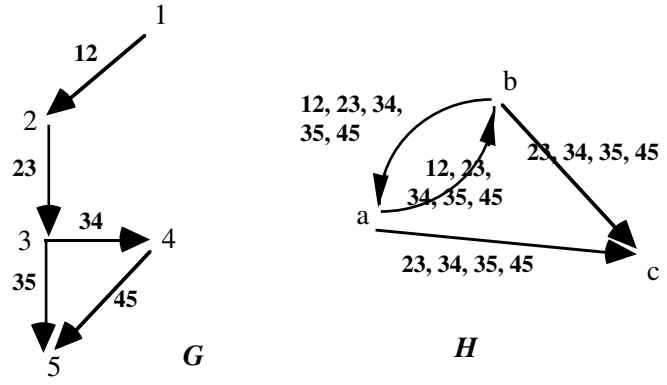
U_G est obtenu à partir de C en donnant une orientation quelconque à chaque arête (on prendra ici l'ordre donné par les définitions des contraintes). L'ensemble des étiquettes est l'ensemble des "noms" de contraintes. Tout arc de U_G est étiqueté par le nom de la contrainte dont il provient.

$X_H = \cup(D_i \in D)$, union des domaines des variables.

U_H est obtenu par union disjointe des $Ri \in R$: pour tout doublet (a , b) d'une contrainte C_i, on crée un arc (a, b) étiqueté par le nom de C_i.

Cette transformation est polynomiale en la taille du CSP P. Il est immédiat de vérifier qu'il existe une solution au CSP ssi il existe un morphisme de G dans H. Les points (1) et (2) du théorème 4 sont trivialement vérifiés.

La Figure 22 présente l'instance de *Morphisme de graphes orientés étiquetés sur les arcs* construite par la transformation précédente à partir de l'instance de CSP de la figure 21.



Les listes d'étiquettes représentent des multi-arcs. Ainsi, il existe 5 arcs de a vers b.

Figure 22. *Transformation de CSP (Figure* 21*) à Morphisme.*

H est en réalité un multigraphe. Si l'on veut supprimer les multi-arcs, on peut passer à un graphe biparti, en subdivisant tout arc. L'étiquetage sur les arcs disparait, et tout sommet est étiqueté : les sommets ajoutés sont étiquetés par le nom de la contrainte correspondante, les autres par une même étiquette supplémentaire.

Lorsqu'on s'intéresse à la transformation d'un CSP n-aire, on peut le transformer préalablement en un CSP binaire (transformation polynomiale, cf. par exemple [Berge 70]), ou le transformer directement en une instance de "Morphisme étiqueté sur les sommets et sur les arêtes" (on obtient une "sorte" de graphe conceptuel : un graphe biparti, une classe de sommets correspondant aux contraintes, étiquetés par le nom de la contrainte, l'autre classe de sommets correspondant aux variables, étiquetés par une même valeur, les arêtes adjacentes à un sommet contrainte étant totalement ordonnées, ce qu'on représente par un étiquetage de ces arêtes de 1 à l'arité de la contrainte).

L'intérêt de ces transformations n'est pas de montrer que les problèmes CSP et Morphisme (ou Projection) sont équivalents (il suffirait de dire qu'ils sont tous deux NP-complets), mais bien de montrer que :

• le graphe des contraintes et *G* ont même graphe non orienté sous-jacent. Ainsi, tout cas polynomial connu pour CSP et basé sur la structure du graphe des contraintes s'applique au graphe *G* de Morphisme, et réciproquement. Par exemple, il est bien connu dans le domaine des CSP que le problème CSP est polynomial lorsque le graphe des contraintes est sans cycle [Freuder 82].

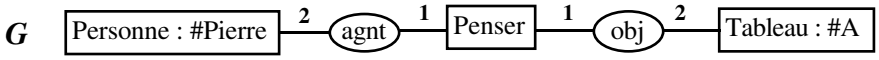
• Toute solution au CSP *est* un morphisme de *G* dans *H,* et réciproquement. Les techniques algorithmiques pour résoudre le problème de l'existence d'une solution, mais aussi pour énumérer les solutions, se transfèrent immédiatement d'un domaine à l'autre.

Les Figures 17 et 20 font également apparaître la similitude entre les deux problèmes. La transformation de CSP à Morphisme donnée plus haut peut s'appliquer pour passer de la figure 20 à la figure 17, mais elle exploiterait alors le fait que le CSP de la figure 20 est particulier : les définitions de contraintes sont symétriques (il n'est donc pas nécessaire de considérer des graphes orientés) et toutes les contraintes ont même définition (il n'est donc pas nécessaire de considérer des graphes étiquetés sur les arêtes). Quant au passage de la figure 17 à la figure 20, il résulte de l'application directe de la transformation de Morphisme à CSP.

6. Graphes emboîtés

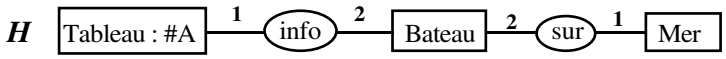
Notre objectif, dans cette partie, est d'étendre —au sens donné dans l'introduction— le modèle des graphes conceptuels simples de telle sorte que l'on puisse représenter des connaissances plus ou moins précises, en raisonnant à un niveau de précision donné. Il s'agit de pouvoir faire des "zooms" sur certains sommets, sans limiter, a priori, les types de concepts sur lesquels on peut zoomer. On peut aussi dire que l'on souhaite pouvoir ajouter des informations à un sommet concept, informations décrivant une partie de l'intérieur de ce sommet. Ces informations internes sont relatives au sommet concept, donc, dit d'une autre façon, elles constituent une description partielle du sommet, et elles ne sont pertinentes que dans le contexte constitué par ce sommet.

Illustrons sur un exemple le type de connaissances que nous voulons représenter. Considérons le graphe simple *G* ci-dessous :

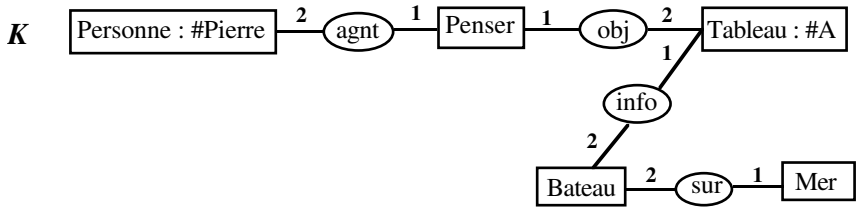


qui peut représenter "Pierre pense au tableau *A*".

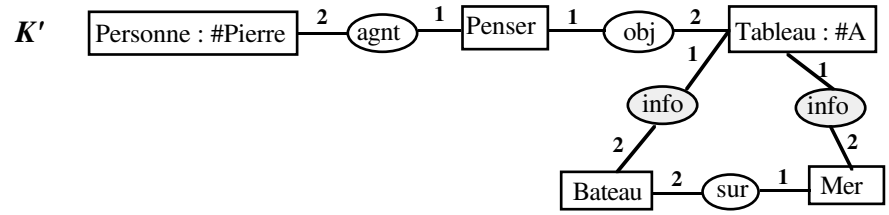
Supposons que nous voulions ajouter des précisions concernant le contenu du tableau *A* auquel pense Pierre. Par exemple, que dans le tableau *A*, il y a une représentation d'un bateau sur la mer. On peut envisager d'ajouter ces informations sur *A* de plusieurs façons. On peut représenter ces informations par un graphe *H* :



puis faire un joint sur *A* entre *G* et *H,* on obtient alors le graphe *K* suivant :

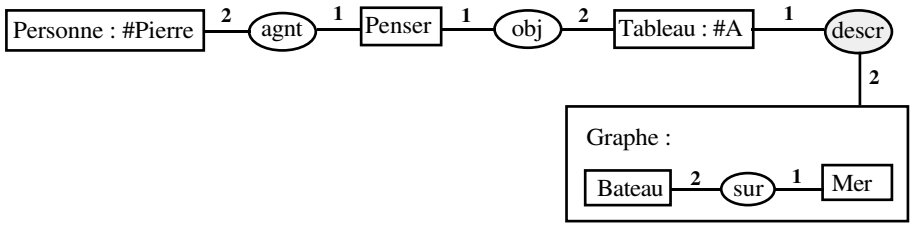


Une variante de ceci peut consister à lier le sommet *x*=[Tableau:#A] par la relation *(info)* —ou par une autre relation— non plus à un sommet particulier mais à tous les sommets *concepts* apportant des informations sur *x*. Ceci donnerait par exemple le graphe *K'* suivant :

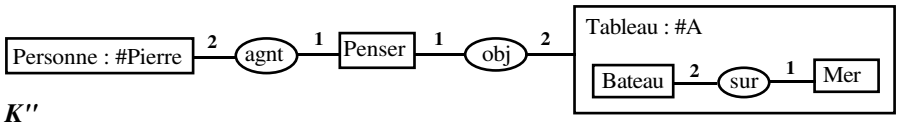


Ce genre de représentations est utilisé par plusieurs chercheurs. Mais elle nous semble présenter un certain nombre d'inconvénients.
1- Toutes les connaissances sont mises au même niveau. Ainsi si on pose, sur *K* ou sur *K'* la question "existe-t-il un bateau?" la réponse sera "oui".
2- On est obligé d'utiliser une relation particulière, comme *(info)*, qui est en quelque sorte une méta-relation. De plus, cette méta-relation ne peut pas porter sur les relations. Ainsi, dans le graphe *K'*, l'occurrence de la relation *(sur)* est implicitement relative au tableau *A*. Mais, imaginons que le tableau représente Albert devant une moto; le fait que cette moto soit la moto d'Albert peut être représenté par une relation, qui ne fait pas partie du tableau.
3- Dans une telle représentation, l'aspect "lecture facile" est perdu, en particulier s'il n'y a pas de restriction à l'usage de cette relation particulière.

Un autre type de solution est de lier directement le sommet, auquel on veut ajouter une description partielle, à un graphe contenant ces informations, ce lien se faisant par une relation particulière, dont la signature comporte un argument de type "graphe". Cette relation est parfois appelée *"description"* [Sowa 94].



C'est en partant de cette idée, mais en cherchant à supprimer la relation "*description*" qui devrait avoir un statut particulier (nous discutons ce point un peu plus loin), que nous proposons la solution suivante. On considère maintenant un graphe, non simple, *K"*, obtenu à partir de *G* en insérant à l'intérieur du sommet [tableau : #A], la description partielle de *A* et contenue dans le graphe *H* :



K″

Si on pose sur *K″* la question "existe-t-il un bateau?" la réponse sera : "oui, dans contexte du tableau *A*". Les informations complémentaires représentées par *H* concernent le tableau *A*, elles n'ont de sens que dans le "contexte" déterminé par *A*. C'est la sémantique intuitive de la description partielle, qui est à la base du modèle des graphes emboîtés. Mais, comme toujours quand on définit un modèle formel, la sémantique initiale n'est pas exclusive, et les graphes emboîtés devraient pouvoir servir à représenter d'autres types de connaissances.

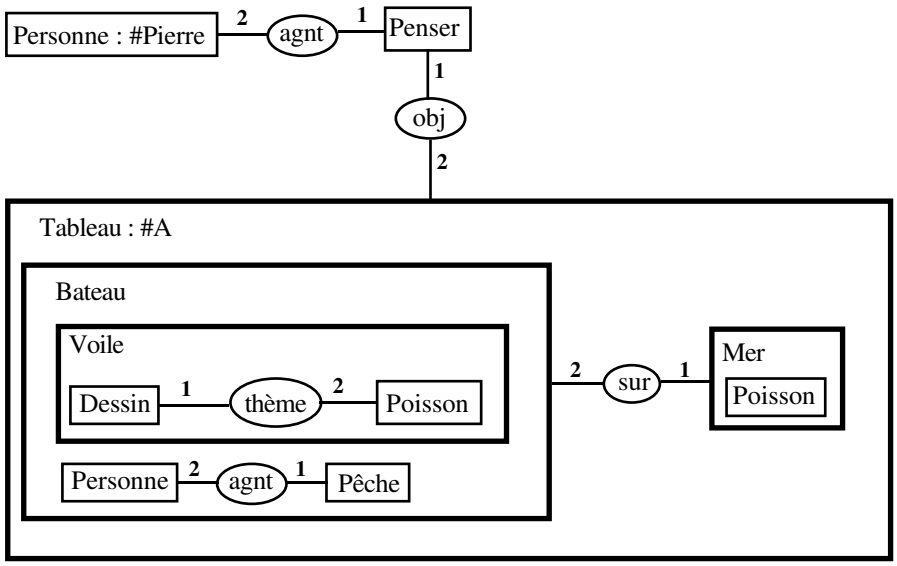
Les graphes emboîtés, qui nous semblent constituer une extension significative des graphes conceptuels simples, sont définis sur un support *S*, la notion de support étant inchangée.

L'étiquette d'un sommet concept a maintenant trois champs, on la note *(t, m, d)*. Les champs *t* et *m* représentent respectivement le type et le marqueur comme dans le modèle de base. Le troisième champ *d*, appelé *description partielle interne*, correspond aux informations plus précises associées à ce sommet. Ce champ peut être égal soit à $**$, la description générique, soit à un graphe. La sémantique d'un sommet ayant pour étiquette *(t, m, G)*, où *G* est un graphe, pouvant être : *m* est un objet de type *t* et *G* est une description partielle de *m*. Cf. la figure 23.

Définition *(Graphes emboîté).*
(1) Un graphe conceptuel simple, dont on a complété toutes les étiquettes de sommets concepts par un troisième champ de valeur $**$, est un graphe emboîté de niveau 0.
(2) Si *G* est un graphe emboîté, en remplaçant un marqueur $**$ d'un sommet concept de *G,* par un graphe conceptuel simple *H* (dont on aura complété les étiquettes par un troisième champ de valeur $**$), on obtient un graphe emboîté Si le niveau de *G* est *k*, alors le niveau de *H* est $k+1$.
(3) Tout graphe emboîté s'obtient à partir de (1) par un nombre fini d'opérations (2).

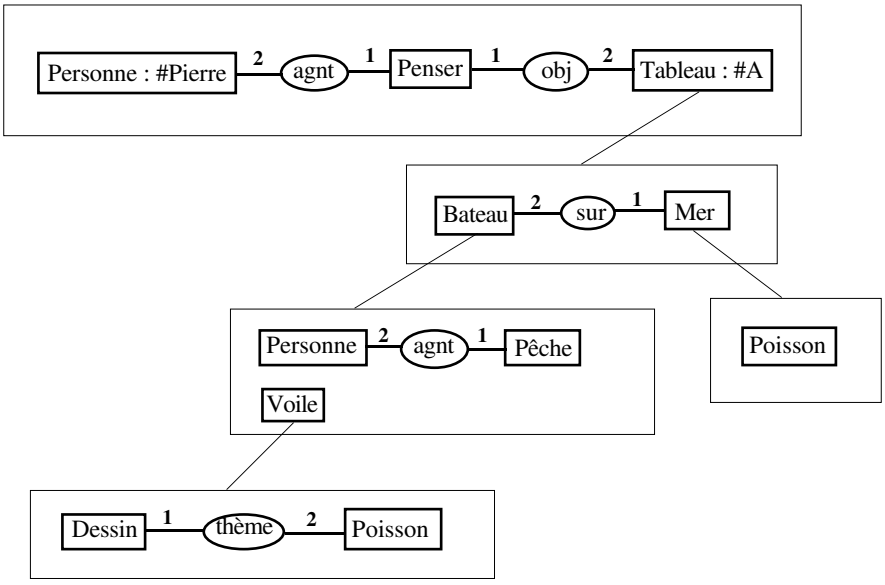
Si *G* est un graphe emboîté, sa *profondeur* est définie comme étant le maximum des niveaux des graphes qui le composent. Le graphe de la figure 23 est ainsi de profondeur 3.

On peut associer à un graphe emboîté *G* une *arborescence* ***A(G)*** dans laquelle chaque sommet est un graphe simple, la racine étant le graphe simple G_0 à partir duquel on a construit *G,* les successeurs d'un sommet *H* étant les graphes simples introduits dans les sommets de *H* par une opération de type (2). La profondeur de *G* est la *profondeur* de *A(G)*. Cf. la figure 24.



La description générique est implicite dans les figures, tout comme l'est le marqueur générique.

Figure 23. *Un graphe emboîté.*



Cette arborescence fait référence au graphe de la figure 23.

Figure 24. *Arborescence associée à un graphe emboîté.*

Si on remplace les arcs de cette arborescence par des liens *(description)* entre le sommet auquel on veut ajouter des informations, et les *sommets concepts du graphe* représentant cette information, on obtient une des solutions évoquées précédemment. On peut donc considérer notre proposition comme une utilisation implicite, et contrainte par la structure d'arborescence, de la relation *(description)*.

La définition récursive des graphes emboîtés, et l'arborescence des graphes simples associée, permet d'étendre facilement, aux graphes emboîtés, les notions et opérations fondamentales du modèle des graphes simples, et ceci en raisonnant niveau par niveau.

Définition *(Projection entre graphes emboîtés)*. Une *projection* d'un graphe emboîté $H=(R_H, C_H, U_H, étiq_H)$ dans un graphe emboîté $G = (R_G, C_G, U_G, étiq_G)$ est un couple d'applications $\prod=(f,g), f : R_H \rightarrow R_G, g : C_H \rightarrow C_G,$ qui :
1. conserve les arêtes et la numérotation des arêtes :
Pour toute arête rc de U_H, $f(r)g(c)$ est une arête de U_G. De plus, si $c = G_i(r)$, alors $g(c) = G_i(f(r))$.
2. peut "restreindre" les étiquettes des sommets
2.1. Pour tout sommet r de R_H, $étiq_G(f(r)) \leq étiq_H(r)$
2.2. Pour tout sommet c de C_H, soit $étiq_H(c) = (t, m, d)$. $étiq_G(g(c)) = (t', m', d')$ vérifie :
- $(t', m') \leq (t, m)$, et
- si d est un graphe, alors d' est un graphe tel qu'il existe une projection de d dans d' (d et d' sont des graphes emboîtés de profondeurs strictement plus petites que celles de G et G'). Si $d = **$, il n'y a pas de contrainte sur d', qui peut être $**$ ou un graphe quelconque.

Soit $\prod$ une projection de G dans G'. On peut lui associer une projection de $A(G)$ dans $A(G')$, ces arborescences étant considérées comme des graphes simples avec la relation *(description)*, cette projection conservant les graphes et les niveaux. C'est-à-dire que la restriction de $\prod$ à un graphe simple H de $A(G)$ est une projection de H dans un graphe simple H' de $A(G')$, H et H' étant de même niveau. Réciproquement à toute projection de $A(G)$ dans $A(G')$ conservant les graphes simples et les niveaux on peut associer une projection de G dans G'.

Comme dans le cas des graphes simples, une projection n'est pas nécessairement injective. Ainsi, deux sommets d'un même graphe simple peuvent avoir la même image, aussi bien que deux sommets de deux graphes simples différents.

Les *opérations élémentaires de spécialisation* sur les graphes emboîtés peuvent se définir de deux façons équivalentes : soit récursivement à partir de la définition récursive des graphes emboîtés, soit à partir des opérations élémentaires de graphes simples sur un des graphes de $A(G)$. C'est cette deuxième solution que nous avons choisie.

Opérations unaires

Soit G un graphe emboîté. On peut obtenir H à partir de G, en effectuant l'une des opérations unaires suivantes sur un graphe quelconque G_i de $A(G)$.

• *Simplification* : Si G_i possède deux sommets relations de même type et jumeaux (c'est-à-dire ayant exactement les mêmes voisins dans le même ordre), H s'obtient en supprimant l'un des jumeaux.

• *Restriction de concept* : Soit c un sommet concept de G_i d'étiquette (t, m, d). H s'obtient en remplaçant l'étiquette de c par (t', m', d') avec : $t' \leq t$, $m' \leq m$ (si m' est un marqueur individuel, alors $\tau(m') = t'$) et $d'=d$ si $d \neq **$, sinon d' est une description quelconque ($**$ ou un graphe quelconque).

• *Restriction de relation* : Soit r un sommet relation de G_i. H s'obtient en diminuant le type de r : remplacer le type de r par un type plus petit $t_{r'}$. Cette opération doit conserver les contraintes fixées par σ : pour i de 1 à $degré(r)$, $type(G_i(r)) \leq \sigma_i(t_{r'})$.

• *Joint interne* : Soient c_1 et c_2 deux sommets concepts de G_i ayant pour étiquettes (t, m, d_1) et (t, m, d_2). H s'obtient en fusionnant c_1 et c_2, et en donnant au nouveau sommet l'étiquette $(t, m, d_1 + d_2)$, où + est définie de la façon suivante : $** + ** = **$, $** + G_1 = G_1 + ** = G_1$. $G_1 + G_2$ est le graphe formé par la juxtaposition de G_1 et G_2.

Opération binaire

• *Somme (disjointe)* : Soient G_i de $A(G)$ et soit K un graphe emboîté n'appartenant pas à $A(G)$ —K peut cependant être isomorphe ou identique du point de vue de la relation de spécialisation à un graphe de $A(G)$. On obtient H à partir de G en remplaçant G_i par la juxtaposition de G_i et de K.

Si $G_i = G$, l'opération de somme disjointe est la même que pour les graphes simples, sinon elle consiste à faire la somme disjointe d'un graphe de $A(G)$ et d'un graphe extérieur à $A(G)$.

Les notions de dérivation et de spécialisation (toujours notée ≤) se définissent de la même façon que pour les graphes simples.

Théorème 5. *Etant donnés G et H deux graphes emboîtés, il existe une projection de H dans G ⇔ G ≤ H.*

Preuve. La partie ⇐ est la même que pour les graphes simples. A chaque opération élémentaire on associe une projection, et on compose ces projections ⇒ par récurrence sur la profondeur du graphe. ❑

La question de la sémantique logique à associer au modèle des graphes emboîtés est en cours d'étude.

7. Conclusion

Notre objectif fondamental est de prouver qu'on peut représenter des connaissances par des graphes et que des raisonnements sur ces connaissances peuvent être simulés par des algorithmes de graphes. Nous pensons que le modèle des graphes simples proposé dans cet article est une première étape vers cet objectif. En effet :

1. le modèle proposé est un modèle formel précis et simple,
2. des opérations élémentaires de graphes permettent de parcourir l'espace des graphes conceptuels par spécialisation/généralisation. De plus, toute séquence de telles opérations est équivalente à une opération fondamentale d'algorithmique sur les graphes (morphisme), elle-même équivalente à la résolution d'un CSP, le modèle est donc solidement amarré à l'algorithmique combinatoire,
3. le modèle a une sémantique en logique du premier ordre consistante et complète,
4. le modèle permet de représenter des connaissances de diverses natures.

Ceci nous a semblé suffisant pour commencer la réalisation d'un système —le système CORALI [Corali 94]— permettant le développement d'applications, à base de connaissances, et utilisant exclusivement les graphes conceptuels. Mais cette première étape ne sera vraiment terminée que lorsque nous aurons validé nos idées et outils sur de vraies applications, ce que nous espérons pouvoir faire dans le cadre du projet GRAFIA [GC 95].

Pour que les divers articles indéfinis du premier paragraphe de cette conclusion —*des* connaissances, *des* raisonnements— ne soient pas trop restrictifs, il faut pouvoir étendre sérieusement le modèle de base, c'est-à-dire passer à une deuxième étape. Il est bien connu que le point crucial en IA est précisément le passage à une deuxième étape, c'est-à-dire à une extension en un modèle plus riche tout en gardant les caractéristiques essentielles du noyau.

Les graphes emboîtés définis à la fin de cet article, permettent d'enrichir notablement le modèle, tout en gardant les points 1 et 2. Nous sommes en train d'étudier le point 3 —la construction d'une sémantique logique— avec Anne Preller.

Deux autres extensions sont en cours d'étude : introduction de définitions de types de concepts et de relations, et introduction de formes simples de négation (c'est-à-dire portant sur des types de concept ou de relation, et non pas sur des graphes).

Remerciements. Nous remercions Anne Preller pour avoir mis en évidence le problème posé par les marqueurs individuels dans la complétude logique, et Roland Ducournau, Olivier Guinaldo, Michel Leclère, Eric Salvat, pour leur lecture attentive et critique de versions préalables de cet article.

Annexe. Sémantique ensembliste.

La sémantique ensembliste que nous proposons correspond, dans le cas des graphes simples, à la sémantique ensembliste habituelle en logique du premier ordre. Nous la donnons ici car c'est une formalisation de la sémantique intuitive que l'on associe habituellement au modèle des graphes simples, et que, de plus, elle devrait pouvoir servir de base pour une sémantique ensembliste de certaines extensions des graphes simples n'ayant pas de sémantique logique simple.

Une *interprétation d'un support S* est constituée par un ensemble non vide D et par une application δ qui associe : à un type de concept une partie de D, à un type de relation d'arité p une partie de D^p, et à un individu (*i.e.* un marqueur individuel) un élément de D. Une interprétation de S est un *modèle* de S si elle satisfait :

(1) pour tout $t \leq t'$ de T_C $\delta(t) \subseteq \delta(t')$
(2) pour tout $r \leq r'$ de T_R $\delta(r) \subseteq \delta(r')$
(3) $\delta(\top) = D$
(4) $\delta(\bot) = \emptyset$
(5) pour tout r de T_r de signature $(t_1,t_2,...,t_p)$ $\delta(r) \subseteq \delta(t_1) \times \delta(t_2) \times ... \times \delta(t_p)$
(6) pour tout a de I $\delta(a) \in \delta(\tau(a))$

D représente l'ensemble de tous les objets que l'on veut envisager de considérer, il correspond donc au type universel (3). L'ensemble vide représente le type absurde (4). A un individu correspond un élément de l'ensemble représentant son type (6), à un type de relation correspond une partie du produit cartésien de D par lui-même, cohérent avec la signature du type de relation (5). Les ordres entre les types se traduisent par l'inclusion entre les interprétations des types (1) et (2).

Propriété 7. *Il existe une bijection entre les modèles, au sens ensembliste, de S et les modèles, au sens logique, de $\Phi(S)$.*

Preuve. Cette bijection associe à un modèle ensembliste de (D,δ) de S l'interprétation (D,δ') de $\Phi(S)$ définie par : d étant un élément de D, $\delta'(t)(d)$ est vrai *ssi* d est un élément de $\delta(t)$, de même pour les types de relations. ❑

Soit (D,δ) un modèle ensembliste de S. Considérons un graphe conceptuel simple G, sur S, ayant k sommets concepts génériques $\{c_1,c_2,...,c_k\}$. On dira qu'un k-uplet d'éléments $(a_1,a_2,...,a_k)$ de D *satisfait* G si, pour tout i, a_i est un élément de $\delta(type(c_i))$, que nous noterons $\delta(c_i)$, avec : pour tout sommet relation de type r ayant $(x_1,x_2,...,x_p)$ pour voisins, $(\delta(x_1),\delta(x_2),...,\delta(x_p))$ appartient à $\delta(r)$. Notons par $\delta(G)$ l'ensemble des k-uplets de D satisfaisant G. On dit que le graphe G *est satisfiable par* δ si $\delta(G)$ est non vide.

On a immédiatement la propriété suivante :

Propriété 8. *Un modèle ensembliste (D,δ) de S satisfait G si et seulement si (D,δ') satisfait $\Phi(G)$.*

Un support peut être considéré comme le *schéma conceptuel* d'une base de données dans le modèle relationnel. Un type de concept correspond à un attribut, et un type de relation correspond à un schéma relationnel. Les ordres sur les ensembles de types correspondent à des contraintes d'inclusion sur les schémas relationnels et sur les attributs. La partie d'un modèle d'un support, associée aux types de relations correspond à une instance du schéma conceptuel. Un graphe correspond à une vue. Certaines extensions des graphes simples correspondent à des extensions classiques dans le modèle relationnel (par exemple introduire des règles sur les graphes correspond aux bases de données déductives, on peut également considérer des contraintes du type dépendances fonctionnelles). Mais d'autres extensions, comme les graphes emboîtés par exemple, ne semblent pas correspondre à des notions étudiées dans les bases de données. Signalons un autre type de lien entre les graphes conceptuels et le modèle relationnel de données, c'est celui étudié par [Boksenbaum et al. 93], où toutes les notions du modèle relationnel de données (schéma conceptuel, schéma relationnel, vues, requêtes) sont représentées par des graphes conceptuels.

Soit H et G deux graphes sur un même support S, on dira que *G est une conséquence de H,* notation : $H \models G$, si tout modèle d de S qui satisfait H satisfait aussi G. On a immédiatement :

Propriété 9. *$H \models G$ si et seulement si $\Phi(H), \Phi(S) \vdash \Phi(G)$.*

Si G et H sont deux graphes sous forme normale, on obtient alors comme corollaire du théorème de consistance et de complétude de Φ (théorème 2):

Corollaire. *Il existe une projection de G dans H si et seulement si $H \models G$.*

Il est facile de faire une démonstration directe de la partie nécessaire. En effet, soit δ un modèle de H, démontrons que, s'il existe une projection de G dans H, δ est aussi un modèle de G. On considère les sommets concepts de G : on a $\delta(\Pi(c)) = \delta(c)$ si c est un individuel, et si c est générique on peut prendre $\delta(\Pi(c))$ pour $\delta(c)$. Considérons un sommet relation de G de type r ayant $(x_1,x_2,...,x_p)$ pour voisins, comme $(\delta(\Pi(x_1)),\delta(\Pi(x_2)),\ldots,\delta(\Pi(x_p)))$ appartient à $\delta(r)$, on a bien $(\delta(x_1),\delta(x_2),...,\delta(x_p))$ qui appartient à $\delta(r)$. Il semble plus difficile de faire une démonstration directe de la réciproque.

Références

[Berge 70] C. Berge, *Graphes et hypergraphes*, Dunod, 1970.

[Boksenbaum et al. 93] C. Boksenbaum, B. Carbonneill, O. Haemmerlé, T. Libourel, Conceptual Graphs for Relational Databases. Proc. of International Conference on Conceptual Structures'93, *Lecture Notes in Artificial Intelligence,* #699, Springer Verlag, 1993, 142-161.

[Bouaud et al.94] J. Bouaud, B. Bachimont, J. Charlet, P. Zweigenbaum, Acquisition and Structuring of an Ontology with Conceptual Graphs. Proc. of *International Conference on Conceptual Structures'94 - Workshop on Knowledge Acquisition using Conceptual Graph Theory*, Springer Verlag, 1994, 1-25.

[Chein Leclère 94] M. Chein, M. Leclère, A Cooperative Program for the Construction of a Concept Type Lattice, Proc. of *International Conference on Conceptual Structures'94*, Proceedings Supplement, 1994, 16-30.

[Chein Mugnier 92] M. Chein, M.L. Mugnier, Conceptual Graphs : Fundamental notions. *Revue d'intelligence artificielle*, 6, 4, 1992, 365-406.

[Corali 94] M. Chein, Présentation du projet CORALI, *GRECO Informatique*, Paris, Juin 1994.

[Fargues et al. 86] J. Fargues, M.C. Landau, A. Dugourd, L. Catach, Conceptual Graphs for Semantics and Knowledge Processing. *IBM Journal of Research and Development,* 30, 1, 1986, 70-79.

[Feder Vardi 93] T. Feder, M. Vardi, Monotone Monadic SNP and Constraint Satisfaction, Proc. 25th ACM *STOC'93*, 1993, 612-622.

[Freuder 82] E.C. Freuder, A sufficient condition for backtrack-free search, *JACM*, 29-1, 1982, 24-32.

[GC 95] M. Chein, J. Bouaud, J.P. Chevallet, R. Dieng, B. Levrat, G. Sabah, Graphes conceptuels, *Actes des cinquièmes journées nationales du PRC-GDR IA,* Teknéa, Nancy, 1995, 179-212.

[Ghosh Wuwongse 94] B. C. Gosh, V. Wuwongse, Inference systems for Conceptual Graph Programs, Proc. of International Conference on Conceptual Structures'94, *Lecture Notes in Artificial Intelligence*, #835, Springer Verlag, 1994, 214-225.

[Gutjahr et al. 92] W. Gutjahr, E. Welzl, *G.* Woeginger, Polynomial graph colourings, *Discrete Applied Mathematics,* 1992.

[Hell Nesetril 90] P. Hell, J. Nesetril, On the complexity of H-coloring, *Journal of Combinatorial Theory,* (B), 48, 1990, 92-110.

[Hell et al. 93] P. Hell, J. Nesetril, X. Zhu, *Complexity of tree homomorphisms*, manuscript, 1993.

[Leclère 95] M. Leclère, dans ce numéro de RIA.

[Levinson Ellis 91] R. Levinson, *G.* Ellis, Multi-level Hierarchical Retrieval. Proc. of *6th Annual Workshop on Conceptual Structures*, 1991, 11-13. Cf. aussi *Knowledge Based Systems*, vol. 5, 3, 1992, 233-244.

[Liquière Brissac 94] M. Liquière, O. Brissac, A class of conceptual graphs with polynomial Iso-projection, version révisée de l'article publié dans Proc. of *International Conference on Conceptual Structures'94,* Proceedings Supplement, 1994.

[Maurer et al. 81] *H.*A. Maurer, J.*H.* Sudborough, E. Welzl, On the Complexity of the General Coloring Problem, *Information and Control*, 51, 1981, 128-145.

[Mugnier 94] M.L. Mugnier, On Specialization/Generalization for Conceptual Graphs, à paraître dans *Journal of Experimental and Theoretical Artificial Intelligence.*

[Mugnier Chein 92] M.L. Mugnier, M. Chein, Polynomial algorithms for projection and matching, Proc. of *7th Annual Workshop on Conceptual Structures*, 1992, 49-58.

[Mugnier Chein 93] M.L. Mugnier, M. Chein, Characterization and Algorithmic Recognition of Canonical Conceptual Graphs, Proc. International Conference on Conceptual Structures'93, *Lecture Notes in Artificial Intelligence,* #699, Springer Verlag, 1993, 294-311.

[Roberts 93] D.D. Roberts, The existential graphs, in *Semantics Networks in Artificial Intelligence*, F. Lehman Ed., Pergamon Press, 1992, 639-664.

[Sabah Vilnat 93] A Hierarchy of Relational types in Conceptual Graphs to handle Natural Language Processing, Proc. *International Conference on Conceptual Structures'93*, 1993, 198-215.

[Salvat 93] E. Salvat, Algorithmes de projection pour les graphes conceptuels, *DEA Informatique*, LIRMM/Université Montpellier II, Juin 1993.

[Sowa 84] J.F. Sowa, *Conceptual Structures - Information Processing in Mind and Machine*, Addison Wesley, 1984.

[Sowa 94] J.F. Sowa, Logical Foundations for Representing Object-Oriented Systems, *Journal of Experimental and Theoretical Artificial Intelligence,* vol. 5, 1993.