Logique et réseau sémantique : le raisonnement sur les graphes conceptuels.

Stéphane Lapalut
projet ACACIA, INRIA Sophia
BP 93, 06902 Sophia Antipolis Cédex
Stephane.Lapalut@sophia.inria.fr

29 mars 1996

Résumé

Dans le formalisme des graphes conceptuels coexistent deux ensembles de règles, provenant de sa consitution progressive : les règles de dérivation canonique, agissant sur la structure des graphes au sens algébrique, et les règles d'inférence, transpositions des règles du système β de Peirce. Malgré de récentes propositions, ces deux ensembles restent dissociés dans leurs utilisations. Il nous semble primordiale de clarifier le statut de ces deux ensembles l'un par rapport à l'autre, permettant leur plein usage dans la mise en œuvre de raisonnement sur les graphes conceptuels. Pour ce faire, nous proposons d'exprimer les règles des deux ensembles sur la même base, en utilisant des règles de grammaires de graphes. Nous présentons ici la démarche suivie et le modèle algébrique raffiné des graphes conceptuels permettant l'expression des règles de dérivation canonique. Nous discutons ensuite de l'expression des règles d'inférence, utilisant notamment la notion clef de contexte, telle qu'elle est actuellement utilisée dans le formalisme des graphes conceptuels. Nous concluons par les résultats espérés et leurs importances pour l'utilisation des graphes conceptuels pour la modélisation de raisonnements.

1 Introduction

A leur commencement dans les années soixante, les réseaux sémantiques se distinguaient de la logique, alors déjà bien établie formellement et utilisée dans l'automatisation des raisonnements. La distinction entre modélisation par la logique et modélisation avec des réseaux sémantiques tenait lieu de quasi-querelle, les deux paraissants antinomiques. Dès 1979, Sowa proposa un formalisme de réseau sémantique qui, dans sa version de 1984, les graphes conceptuels, intégre ces deux paradigmes de modélisation en intelligence artificielle.

Parallèlement, les écoles de Berlin et de Marynland proposaient les "web grammar", devenuent depuis les grammaires de graphes. L'exploration de ce champ de recherche commença avec un fort développement théorique, concrétisé depuis quelques années par des outils logiciels. Les prémissent des graphes conceptuels en 1979 s'apparentent plus aux grammaires de graphes [5] qu'aux formalismes logiques. En particuliers, les règles de dérivation canoniques sont des règles de grammaire de graphe génériques [6]. L'introduction des règles du système logique β de Peirce dans les graphes conceptuels lui conféra une capacité de modélisation des raisonnements tout en conservant l'aspect graphique. Néanmoins, ceci introduisit un deuxième ensemble de règles, ditent règles d'inférence qui transpose les règles du système β de Peirce aux graphes conceptuels. Cependant, la nature "logique" de la notation graphique de Peirce se différencie de la nature algèbrique des graphes conceptuels, expliquant cette séparation des deux ensembles de règles.

Nous pensons qu'une des voies permettant de pouvoir manipuler ces deux ensembles de règles sous une même forme est de les exprimer par des grammaires de graphes. Pour cela, nous avons détaillé le modèle algébrique décrivant les graphes conceptuels, notamment pour la prise en compte de l'opération de restriction sur les nœuds concepts par des règles de grammaires de graphes. Ceci permet d'écrire les règles canoniques sous forme de règles de grammaires de graphes. L'étape suivante consiste à exprimer les règles d'inférence à l'aide de grammaires de graphes. Le point clef de cette tâche est l'expression d'une notion de "contexte", telle que manipulée par Sowa dans sa proposition de 1984. Elle sert à donner une expression au *cut* de Peirce, mais son interprétation dans les graphes conceptuels reste à formaliser.

2 Modèles algébriques des graphes conceptuels

Nous ne détaillerons pas les modèles complets, mais seulement leurs principales caractéristiques et différences. Nous insisterons sur l'apport de notre modèle, par rapport à celui de Chein et Mugnier [4], permettant en particuliers l'utilisation de grammaires de graphes pour exprimer le premier ensemble, les règles de dérivation des graphes conceptuels.

2.1 Le modèle de Chein et Mugnier

Dans leur formalisation des principales idées proposées par Sowa, notamment dans [5] puis dans [6], Chein et Mugnier proposent le modèle algébrique suivant des graphes conceptuels, mettant en évidence la notion de support permettant la construction d'un ensemble de graphes. Un support se définit comme un quintuplet $S = (T_c, T_r, M, conf, B)$. T_c (resp. T_r) est un treillis de type de concepts (resp. type de relations), M un ensemble de marqueurs individuels, $conf$ une relation de conformité des marqueurs selon le type du concept, et B un ensemble de graphes étoiles canoniques, (c.a.d. bien formés). Un graphe conceptuel est un graphe bipartie fini, connexe ou non, dont les deux types de sommets sont les sommets concepts et les sommets relations. Un sommet concept s'écrira $\boxed{\text{CHAT : mistigri}}$, pour désigner un chat appelé Mistigri par le référent de type CHAT associé au marqueur individuel Mistigri. Les relations sont selon Sowa, des relations simples comme AGENT, SUJET, SUR. Leur organisation découle de choix ontologiques qui sortent du cadre de cet article. Un graphe conceptuel s'exprimera par un quadruplet $G = (R, C, U, lab)$. Avec C (resp. R), l'ensemble des nœuds concept (resp. relation), U l'ensemble des arêtes entres les nœuds concepts et les nœuds relations, et lab est la relation d'attribution d'une étiquette à un nœud, qui associe un type de concept à un nœud concept. Pour un nœud concept, lab va lui attribuer une étiquette complexe composée du type et du référent du concept représenté : $\forall c \in C$, $lab(c) = (type(c), ref(c))$, avec $type(x)$ (resp. $ref(x)$) la fonction renvoyant le type (resp. le référent) d'un nœud concept x. Le lien entre un graphe et son support réside donc dans cette fonction traitant de la *sémantique* des graphes conceptuels. Nous trouvons que c'est là une faiblesse de ce modèle pour l'usage de règle de grammaire de graphes, agissant sur la *structure* des graphes.

Nous proposons d'exprimer par un modèle algèbrique le typage des nœuds concepts, ramenant leur étiquette à un simple référent. Ce point de vue semble plus naturel à l'écriture des graphes, le référent individuel d'un nœud concept d'un graphe représentant un individu particulier, caractérisé en premier lieu par *ce* référent individuel.

2.2 Un modèle algébrique intégré

Plutôt que de séparer le support des graphes, nous redéfinissons un ensemble de graphe utilisant le même support, comme une base de graphe. Une base de graphe B_{CG} sera un nuplet $B_{CG} = (S_c, S_r, B_c, B_r, M, conf, G)$, avec S_c (resp. S_r) un graphe supportant la structuration des types de concepts (resp. des types de relations), B_r (resp. B_c) est l'ensemble des graphes étoiles définissant les signatures de relation (pour les concepts, cette possiblité n'est pas encore exploité), M et $conf$ ont le même sens que dans le modèle précédant, et enfin G est l'ensemble des graphes conceptuels se rapportant au "support" ainsi défini.

Les graphes dans G se représentent par un nuplet $G = (R, C, s, t, u_r, u_c, m)$, avec R l'ensemble des nœuds relations, C l'ensemble des nœuds concepts, s (resp. t) l'ensemble des arêtes orientées des éléments de R vers les éléments de C (resp. ceux de C vers ceux de R). Le point important de notre modèle intervient dans les relations de typages des nœuds concepts (resp. relations) exprimés par l'application u_c (resp. u_r). Ces deux relations définissent des arêtes étiquetées *type* entre un nœud concept (resp. relation) et le nœud type de concept (resp. type de relation) dans S_c (resp. S_r), définis dans la base B_{CG}. Enfin, l'application m définit l'application entre les nœuds concepts et l'ensemble M des marqueurs individuels, dans la base B_{CG}.

Nous avons ainsi généralisé et intégré le modèle de Chein et Mugnier en une base de graphes conceptuels, définie par un modèle purement algèbrique. Il précise en particuliers leur fonction $lab(c)$ par un lien explicite. Nous sommes donc en possession d'un modèle utilisant des graphes au sens mathématique et isomorphe au modèle actuel des graphes conceptuels. Nous pouvons dès lors appliquer pleinement les résultats obtenus sur les grammaires de graphes. Nous allons brièvement voir ce qu'est une grammaire de graphe et surtout une règle de grammaires de graphes, pour ensuite donner rapidement une expression des règles de dérivation canonique.

3 Grammaires de graphes et graphes conceptuels

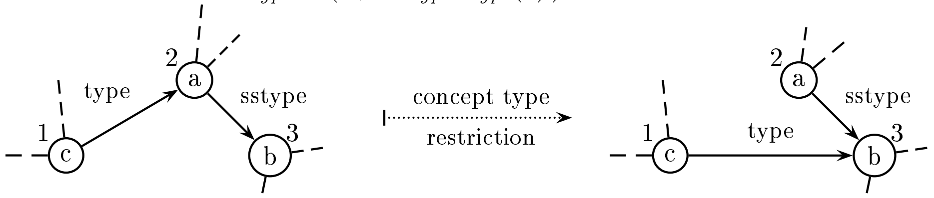
Les grammaires de graphes sont la généralisation des grammaires classiques sur des chaînes de symboles. Elles comportent des règles de réécriture, les règles de production de graphes, qui permettent de réécrire une partie d'un graphe. La partie à réécrire est identifiée par des conditions de recollement, c'est à dire un ensemble de nœuds délimitant un sous-graphe. Ce sous-graphe va être identifié par la partie gauche de la règle, effacée du graphe et remplacée par la partie droite de la règle. Ce dernier sous-graphe de remplacement va être reconnecté au reste du graphe d'origine à l'aide de conditions d'enchâssement. Tout ceci se formalise dans le cadre de la théorie des catégories algébriques pour un diagramme "pushout".
Formellement, une production de graphe p sera une paire de morphisme $p = (B_1 \leftarrow K \rightarrow B_2)$, où B_1 (resp. B_1) sera la partie gauche (resp. droite) de la règle, K le graphe d'interface servant à déterminer les conditions de recollement. La transformation d'un graphe G en un graphe H par la production p se décrit par le diagramme "pushtout" suivant, avec D l'identification du graphe d'interface K par le morphisme $(K \rightarrow D)$:

$$\begin{tikzcd}
B_1 \arrow[d] \arrow[dr, phantom, "PO_1"] & K \arrow[l] \arrow[r] \arrow[d] \arrow[dr, phantom, "PO_2"] & B_2 \arrow[d] \\
G & D \arrow[l] \arrow[r] & H
\end{tikzcd}$$

Pour une bonne introduction sur les grammaires de graphes, voir [1]. D'après [3], six opérations élémentaires suffisent pour exprimer les règles de grammaires de graphes : l'addition, la suppression et le renomage de nœuds et d'arêtes. Nous pouvons d'ores et déjà avec ces éléments discuter de l'expression des règles de dérivation canonique par des règles de grammaires de graphes.

3.1 Expression des règles de dérivation canonique

L'ensemble donné par Sowa dans [6] comporte quatre règles de dérivation canonique : la jointure, la copie conforme, la simplification et la restriction. Les trois premières trouvent une expression naturelle par des productions de graphe. Par contre, la restriction de concept n'a pas d'expression triviale. Si seul le référent est restreint, elle consiste juste en un changement d'étiquette d'un nœud concept. Si la restriction s'applique au type, nous pouvons l'exprimer grâce à notre modèle et à une règle d'enchâssement, que nous donnons ci-après selon la syntaxe proposée dans [3]. Nous donnons l'exemple de la restriction d'un concept $\boxed{\text{a:c}}$ en un concept $\boxed{\text{d:c}}$. Après avoir identifié le nœud concept à restreindre, ici le nœud 1, son sous-typage consiste à effacer l'arc entre ce nœud et son nœud type de concept dans S_r, puis à déterminer le nouveau type par l'expression d'enchâssement $l_{type} = (\,1;\, L_{sstype}L_{type}(1)\,)$ et à établir un nouvel arc entre les deux :



Cette expression d'enchâssement se lit "remplacer par une arête de label *type* sortant du nœud 1 et le reliant au nœud trouvé en sortant du nœud 1 par une arête de label *type* puis en sortant par une arête de label *sous-type*. La production de graphe attachée est l'identité $c ::= c$.

3.2 Problématique de l'expression des règles d'inférence

Les règles d'inférence sont directement basées sur le système β de Peirce. Ce dernier a défini une logique existentielle conjonctive et positive, les assertions étant des *tokens* tracés sur une *feuille d'assertion*, éventuellement reliés par des *lignes d'identité*, qui captent la notion de variable logique. La négation correspond au *cut*, une "fine ligne tracée autour d'un ensemble de tokens de manière à l'isoler du reste de la feuille d'assertion." Un double *cut*, le *cut* d'un *cut*, est la feuille d'assertion elle-même (hypothèse du monde clos). Sowa, pour réutiliser les quatre règles d'inférence, agissant surtout sur des configurations (imbrications et juxtapositions) de *cut* et de lignes d'identité (traduites par des liens de coréférence dans les graphes conceptuels), se sert de "contextes négatifs", qui n'ont actuellement aucune expression formelle et recèlent une ambiguïté dans la proposition d'origine, soulignée par Nazarenko dans [4]. Nous voudrions clarifier cette notion et la transposition du *cut* aux graphes conceptuels par les outils des grammaires de graphes. La notion informelle de contexte a été proposée dans [2], mais reste pour le moment insuffisante pour traiter le *cut*.

4 Conclusion

L'expression des deux ensembles de règles manipulant des graphes conceptuels sur une même base permettrait de cerner les raisonnements possibles sur ces graphes. Nous avons proposé l'utilisation des grammaires de graphe et un raffinement du modèle algébrique des graphes conceptuels pour y parvenir. Nous avons brièvement vu l'expression des règles de dérivation canonique, en particuliers la restriction. Pour produire un résultat similaire sur les règles d'inférence, nous travaillons sur une représentation du *cut* dans notre modèle.

Références

[1] V. Claus, H. Ehrig, and G. Rozenberg, editors. *Graph-Grammars and their application to computer science and biology*, number 73 in Lecture Notes in Computer Science. Springer-Verlag, 1979.

[2] G. Engels and A. Schürr. Encapsulated hierarchical graphs, graph types, and meta types. Technical Report 95-21, Departement of Computer Science, Leiden University, P.O. Box 9512, 2300 RA Leiden, The Netherlands, July 1995.

[3] M. Nagl. Set theoric approaches to graph-grammars. In H. Ehrig, M. Nagl, G. Rozenberg, and A. Rosenfeld, editors, *Graph-Grammars and Their Application to Computer Science, 3rd International Workshop*, number 291 in Lecture Notes in Computer Science, pages 41–54. Springe-Verlag, 1986.

[4] G. Sabah. Formalisation et extension des graphes conceptuels. In *Actes des 4ème Journées nationales du PRC-GRD Intelligence Artificielle*, pages 57–119, Marseille, October 1992. TEKNEA.

[5] J. Sowa. Definitional mechanisms for conceptual graphs. In V. Claus, H. Erhig, and G. Rozenberg, editors, *Graph-Grammars and Their Application to Computer Science and Biology*, number 73 in Lecture Notes in Computer Science, pages 426–439, Bad Honnef, October 1979. Springer-Verlag.

[6] J.F. Sowa. *Conceptual Structures*, volume Information processing in mind and machine of *The systems programming series*. Addison-Wesley, 1984.