Grew : un outil de réécriture de graphes pour le TAL

Bruno Guillaume[1, 2] Guillame Bonfante[1, 3] Paul Masson
Mathieu Morey[4,5] Guy Perrier[1, 3]
(1) LORIA - Campus Scientifique - BP 239 - 54506 Vandœuvre-lès-Nancy cedex
(2) INRIA Grand Est - 615, rue du Jardin Botanique - 54600 Villers-lès-Nancy
(3) Université de Lorraine - 34, cours Léopold - CS 25233 - 54502 Nancy cedex
(4) Laboratoire Parole et Langage, Aix-Marseille Université
(5) Linguistics and Multilingual Studies, Nanyang Technological University

RÉSUMÉ
Nous présentons un outil de réécriture de graphes qui a été conçu spécifiquement pour des applications au TAL. Il permet de décrire des graphes dont les nœuds contiennent des structures de traits et dont les arcs décrivent des relations entre ces nœuds. Nous présentons ici la réécriture de graphes que l'on considère, l'implantation existante et quelques expérimentations.

ABSTRACT
Grew: a Graph Rewriting Tool for NLP

We present a Graph Rewriting Tool dedicated to NLP applications. Graph nodes contain feature structures and edges describe relations between nodes. We explain the Graph Rewriting framework we use, the implemented system and some experiments.

MOTS-CLÉS : réécriture de graphes, interface syntaxe-sémantique.

KEYWORDS: graph rewriting, syntax-semantics interface.

1 Réécriture de graphes

Même si la structure d'arbre est mise en avant pour les modélisations linguistiques, elle est souvent insuffisante, notamment pour les représentations sémantiques. Même en syntaxe, autour d'une structure d'arbre, des mécanismes comme la coindexation ou les structures de traits réentrantes font apparaître des structures de graphes. Partant de ces observations, pour avoir un cadre unique d'étude, nous avons pris le parti de considérer toutes ces structures comme des graphes. Évidemment, un cadre naturel pour décrire les calculs et les transformations sur les graphes est la réécriture de graphes. En effet, la réécriture est un modèle de calcul qui permet de décrire n'importe quelle transformation ; c'est également un domaine très actif en informatique théorique et de nombreux résultats de confluence ou de terminaison existent.

Par rapport à la réécriture de mots ou de termes, la réécriture de graphes est plus délicate et il n'existe pas de définition canonique de cette réécriture. Les travaux théoriques, utilisant la théorie des catégories, décrivent deux types de réécriture (SPO et DPO) ; cependant ces définitions, mathématiquement élégantes, sont peu pratiques pour écrire effectivement des règles. Nous avons donc utilisé une autre présentation qui décrit explicitement les transformations à appliquer au graphe à l'aide d'une suite d'actions élémentaires.

Actes de la conférence conjointe JEP-TALN-RECITAL 2012, volume 5: Démonstrations, pages 1–2,
Grenoble, 4 au 8 juin 2012.

2 le logiciel Grew

Nous donnons quelques détails et particularités de notre système ci-dessous. Plus d'informations dont disponibles sur le site `grew.loria.fr`.

Graphes. Les graphes que nous considérons sont composés d'un ensemble de nœuds qui contiennent des structures de traits non récursives et d'un ensemble d'arcs étiquetés. Ces arcs codent en fait des relations, ils vérifient donc toujours la contrainte qu'il n'y peut pas y avoir deux arcs avec la même étiquette, la même source et le même but. Dans les expérimentations, les relations utilisées sont des dépendances mais le système permet d'exprimer n'importe quel type de relations (comme la dominance dans un arbre syntagmatique par exemple).

Règles. Pour manipuler les graphes, nous utilisons des règles de réécriture. Une règle de réécriture est définie en trois parties : un *patron positif* qui est un graphe qu'on va chercher à apparier avec une partie du graphe à réécrire ; un ensemble éventuellement vide de *patrons négatifs* qui peuvent bloquer l'application de la règle ; une liste de *commandes* qui décrivent les transformations à apporter au graphe. Les commandes permettent d'ajouter, de modifier ou de supprimer des traits, des nœuds ou des arcs.

Modules. Dans les applications, le nombre de règles à considérer peut être grand (plusieurs centaines). Pour contrôler le comportement global du calcul et pour faciliter le développement et la maintenance d'un système, les règles sont réparties dans différents *modules*. À l'intérieur d'un module, les règles ne sont pas ordonnées et tous les résultats sont calculés. En revanche, les modules sont ordonnés et la réécriture de l'ensemble du système se fait en appliquant chaque module sur l'ensemble des résultats du module précédent. Un système de règles organisées en modules s'appelle un GRS (pour *Graph Rewriting System*).

Interfaces utilisateurs. Le logiciel GREW permet de définir un GRS et de l'appliquer à des graphes. D'une part, une interface graphique permet de visualiser le détail d'une réécriture : il est possible de visualiser l'ensemble des étapes de la réécriture, règle par règle. D'autre part, il est possible d'appliquer un GRS sur un ensemble de graphes (éventuellement sur un cluster de calcul) et d'obtenir des statistiques sur l'ambiguïté du calcul et sur les fréquences d'utilisation des règles ou des modules.

Notre système a été expérimenté (Guillaume et Perrier, 2012) sur des données de tailles réelles : réécriture de 12 000 graphes à l'aide d'un système de 34 modules contenants plus de 400 règles. Parmi des fonctionnalités récentes, on peut noter la possibilité de paramétrer les règles avec des informations lexicales et l'ajout un nouveau type de règles qui filtrent les résultats d'un module pour ne garder que ceux qui respectent certains motifs. Ces deux nouveautés permettent de mieux factoriser les règles et donc facilitent le développement de plus grands systèmes de règles.

Références

GUILLAUME, B. et PERRIER, G. (2012). Annotation sémantique du French Treebank à l'aide de la réécriture modulaire de graphes. *In Actes de TALN 2012 (Traitement automatique des langues naturelles)*, Grenoble. ATALA.