Un outil de représentation et de développement des Grammaires de Propriétés

Marie-Laure Guénot & Tristan Van Rullen

Laboratoire Parole et Langage – CNRS – Université de Provence
29 avenue Robert Schuman, 13621 Aix-en-Provence cedex 1

{mlg, tristan}@lpl.univ-aix.fr

Mots-clefs – *Keywords*

Linguistique descriptive – Linguistique formelle – Grammaires de Propriétés (GP) – Traitement Automatique des Langues Naturelles (TALN) – Syntaxe.

Descriptive linguistics – Formal linguistics – Property Grammars (PG) – Natural Language Processing (NLP) – Syntax.

Résumé – *Abstract*

Nous présentons dans cet article un outil graphique de développement de grammaire, basé sur le formalisme des *Grammaires de Propriétés*. Nous y exprimons les raisons pour lesquelles l'association d'une représentation complète et ergonomique, et d'un modèle formel flexible et homogène fournit un avantage considérable pour l'intégration des informations issues de la linguistique descriptive.

We present in this paper a graphical tool for grammar development, based upon the Property Grammars *formalism. We explain the reasons why the association of a complete and ergonomic representation and a neutral and homogeneous model, provides the considerable advantage of integrating information coming from descriptive linguistics.*

1 Introduction

La diversité croissante des besoins en TALN pousse aujourd'hui les recherches en linguistique informatique à développer des modèles grammaticaux aptes à traiter des productions de langue toutes hétérogènes qu'elles soient (*e.g.* sources orales, importants écarts à la norme). Or la linguistique descriptive a d'ores et déjà proposé quantité d'analyses empiriques approfondies de ce type d'attestations protéiformes ; cependant leurs propositions ne sont actuellement pas exploitées en informatique, faute d'un formalisme qui soit à même de leur proposer un com-

plétude, une homogénéité et une flexibilité suffisantes pour représenter l'intégralité de leurs réponses sous une forme tout à la fois ergonomique et directement implémentable.

Nous présentons dans cet article un projet qui vise à permettre de représenter et d'exploiter de telles théories, en utilisant le formalisme des *Grammaires de Propriétés*. Nous proposons à cette fin un format de représentation de ces grammaires qui soit à la fois *générique* par l'utilisation du langage XML pour l'encodage de la grammaire, et *simple d'utilisation* par la mise en place d'une interface graphique dédiée à la représentation, la création et l'utilisation des Grammaires de Propriétés.

2 Contraintes formelles et informatiques

2.1 Contraintes formelles

Le modèle des Grammaires de Propriétés[1] (ci-après GP) repose sur la constatation que toute information linguistique, quel qu'en soit le niveau, peut être intégralement exprimée en termes de contraintes (également appelées *propriétés*). L'utilisation d'un tel formalisme offre à notre sens un certain nombre d'avantages, et a d'importantes conséquences sur le développement d'une grammaire. Plus précisément, puisque l'information n'est représentée qu'en termes de contraintes, il est possible de concevoir une grammaire intégrant des informations aussi précises que celles fournies par les théories descriptivistes.

2.2 Contraintes informatiques

Du point de vue informatique, intégrer une grammaire à un système de TALN nécessite la vérification de plusieurs contraintes, provenant autant du type de problème à traiter que des futurs utilisateurs du système. Tout d'abord le linguiste – qui développe la grammaire – et l'informaticien – qui l'intègre à un programme de TALN – manipulent le même formalisme ; il est donc nécessaire de convenir d'un modèle de représentation des objets de la grammaire répondant simultanément aux besoins des deux protagonistes.

Il est essentiel, de plus, pour la validité des outils ainsi que par économie de moyens, que les modifications faites à une grammaire soient immédiatement répercutées dans le programme qui s'en sert sans modification[2]. A cette fin la représentation de la grammaire doit être parfaitement indépendante de son utilisation, ce qui implique d'exclure toute procédure *ad hoc*.

Enfin, l'outil informatique est lui-même soumis à des contraintes incontournables à plusieurs niveaux, allant du choix d'un format standardisé (tel que le XML), à celui de données et d'algorithmes évitant l'explosion combinatoire des traitements (graphes, bottom-up parsing, etc.). En conséquence de quoi les grammaires seront représentées en mémoire sous forme de structures de données suffisamment souples pour permettre à diverses applications de les manipuler et d'en obtenir rapidement les résultats attendus.

Ces contraintes interviennent simultanément dans les choix de représentation. La problématique qui les gouverne consiste à conserver un regard suffisamment en recul par rapport aux

---

[1] Pour une description détaillée du formalisme des GP, on peut se reporter par exemple à Blache 2001.

[2] On peut se reporter à ce propos à Kermes et Evert 2003.

relations qu'entretiennent modèles et théories linguistiques, et à choisir un modèle en démontrant qu'il est à la fois représentatif de la totalité des objets du formalisme, et générique – canonique – dans les représentations qu'il permet.

Partant des considérations et de la problématique exposées ci-dessus, le choix d'une représentation donnée pour les GP découle à la fois des travaux déjà réalisés sur ces mêmes grammaires et des perspectives de développement qu'elles offrent, tant dans le traitement de l'analyse syntaxique à granularité variable (*cf.* par exemple Balfourier *et al*. 2002) que dans les traitements particuliers de la synthèse vocale à l'aide de représentations sémantiques et prosodiques au sein même de la grammaire (*cf.* par exemple Blache et Hirst 2001, ou Blache 2003).

Le format de fichier XML, qui permet à la fois une lecture humaine et informatique des données de la grammaire, une correction manuelle comme un chargement aisé et précontraint grâce à une spécification de règles syntaxiques à vérifier au sein même de ce fichier (telle que le format DTD), est notre point de départ pour représenter les objets de la grammaire (voir à ce propos Simov *et al.* 2002 et Simov *et al.* 2001, par exemple).

Le modèle informatique qui s'apparente directement aux données des GP est celui des graphes, dans lesquels les noeuds sont associés à une structure « objet sémantique » permettant de les qualifier, et les arcs représentent les relations sémantiques entre ces objets.

2.3 Point de convergence de ces contraintes : l'outil Accolade

Les contraintes énumérées ci-dessus nous ont conduit à développer un outil, *Accolade*, réalisé en Java, qui se positionne au coeur de la problématique du formalisme et du traitement automatique. Cet outil réalise l'interface entre le développement assisté de grammaires (cf. Blache *et al.* 2003), les ressources (notamment un lexique de 450 000 formes développé au Laboratoire Parole et Langage) et les outils de TALN qui s'appuient dessus.

Cette application offre de fait la possibilité de compiler un graphe des GP en un analyseur superficiel (*shallow parser*) immédiatement fonctionnel (*cf.* à ce propos Blache et Van Rullen 2002), et ouvre une perspective pour la rédaction d'un analyseur syntaxique profond (*deep parser*). Elle met en évidence à la fois les caractéristiques des ressources linguistiques (lexique et grammaire) et celles des résultats (étiquetages, analyses syntaxiques, etc.) produits par les outils de TALN traditionnels.

3 Définition des objets du modèle

Le modèle formel des GP repose sur l'utilisation de deux couples de deux objets fondamentaux : *catégories* et *traits* d'une part, *propriétés* et *opérations* d'autre part. Nous allons dans cette partie les définir et donner à titre d'exemple un extrait de la grammaire du français à large couverture en cours de développement au Laboratoire Parole et Langage, et sa représentation dans l'interface de développement Accolade.

3.1 Catégories

Chaque *catégorie* est une unité syntaxique, constituant le vocabulaire de base de la grammaire. Il est important de souligner qu'il n'est fait aucune différence explicite entre catégories

dites « lexicales »[3] et « syntagmatiques »[4] ; cette distinction n'est opérée qu'en regard au lexique utilisé qui lui, de fait, ne contient que des unités minimales.

Dans le logiciel Accolade (*cf.* figure 1), la grammaire correspond à un noeud initial, qui est relié à autant de noeuds-fils qu'elle contient de catégories (*A* pour l'adjectif[5] et *R* pour l'adverbe dans notre exemple). Puis chaque catégorie a elle-même deux fils : un noeud pour rassembler ses traits (*A_traits* dans notre exemple, correspondant aux traits de l'adjectif), un autre pour ses propriétés (*A_props*). Priorité a été donnée à représenter les informations de la manière la plus proche possible de l'encodage XML, de façon d'une part à permettre de visualiser facilement l'intégralité des informations linguistiques, d'autre part à fournir un réel outil graphique de développement de grammaire (en permettant d'éditer, de modifier, de créer entièrement une grammaire via le graphe qui la représente).

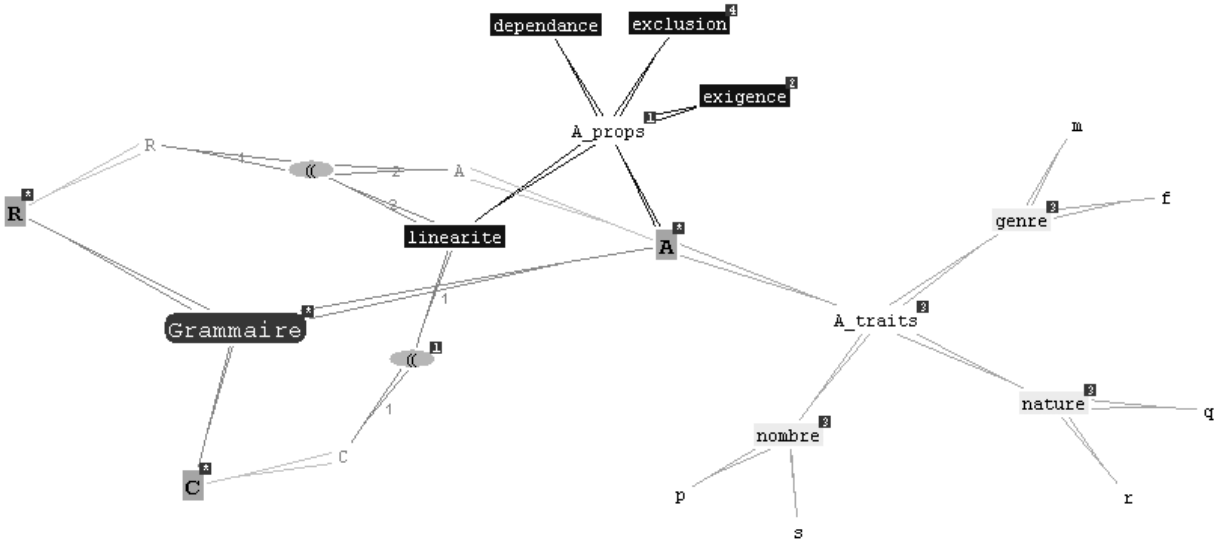

Figure 1 : Extrait de représentation graphique dans Accolade.

3.2 Traits

A chaque catégorie est attribué un *ensemble de traits possibles*, ordonné, dans lequel un trait est, tout à fait usuellement, un couple attribut-valeur apportant des spécifications caractéristiques. Cependant à la différence des théories classiques basées sur les contraintes, ces ensembles de traits sont statiques dans le sens où ils ne contiennent aucune information opérationnelle (qui sont intégralement reléguées dans les propriétés).

Dans Accolade (*cf.* figure 1), le noeud correspondant à l'ensemble des traits possibles (*A_traits*) présente autant de fils que d'attributs définis pour la catégorie (soit *genre, nombre* et *nature*). Chacun de ces fils a, lui-même, autant de fils que de valeurs possibles (*m* et *s* pour l'attribut *nombre*, par exemple).

---

3 Ou « terminales », ou encore « minimales ».

4 Ou « non terminales », ou encore « construites ».

5 Comme on l'a fait remarquer précédemment, il est à noter que le terme *adjectif* s'applique ici à la fois aux lexèmes et aux syntagmes adjectivaux. Il en est de même pour *adverbe*, qu'on emploie pour signifier à la fois lexème et syntagme adverbial.

3.3 Propriétés

Les *propriétés* constituent des relations entre catégories ou ensembles de catégories. Elles sont parfaitement indépendantes les unes des autres, et définies au même niveau. Dans la grammaire que nous mettons en place actuellement, nous utilisons le jeu de propriétés suivant[6] : *exigence* (relations de cooccurrence), *exclusion* (restriction de cooccurrence), *linéarité* (contraintes de précédence linéaire) et *dépendance* (relations de dépendance lexico-sémantique).

Dans Accolade (*cf.* figure 1), à chaque type de propriété (*e.g. exigence*, *linéarité*) correspond un noeud, qui a comme fils autant de noeuds que la propriété contient d'opérations (on peut ainsi voir deux clauses de linéarité développées dans la figure 1, dont le symbole est <<), et qui eux-mêmes ont deux fils, un par opérande, représentant des références à des catégories (comme *R* et *A* fils de << dans notre exemple). On remarque que chacune de ces références se manifeste par la présence d'un noeud qui est relié à la fois à la propriété à laquelle on s'intéresse (on voit que le noeud de référence *R* est le fils de la clause de linéarité <<), et à la catégorie correspondante (le noeud de référence *R* est relié au noeud de catégorie *R*). Ceci fournit un confort considérable dans la visualisation des relations entre catégories, peu évidente dans le codage XML.

3.4 Opérations

Les propriétés sont constituées d'un ensemble de clauses, c'est-à-dire d'expressions mettant en relation un certain nombre de références à des catégories données à l'aide d'une *opération* définie dans la grammaire[7]. On utilise à cette fin des opérateurs binaires[8], s'appliquant sur deux membres, chacun constitués d'une référence à une catégorie ou d'un ensemble de références à des catégories (rassemblées à l'aide d'opérateurs proches des ET et OU logiques).

L'ordre des opérandes est pertinent pour les propriétés d'exigence, d'exclusion et de linéarité ; il ne l'est pas pour la dépendance. Dans le premier cas, la différenciation entre les deux membres d'une clause est essentielle puisque les deux opérandes ne seront pas sujets au même traitement[9] ; cette distinction doit donc figurer clairement dans Accolade (*cf.* figure 1).

4 Conclusion

La représentation formelle des résultats d'analyses descriptives des langues semble être une perspective enrichissante dans l'objectif d'un TALN qui soit capable de traiter des entrées « tout-venant ». Cependant il est nécessaire pour représenter ce type d'informations linguisti-

---

[6] Dans sa forme actuelle, notre grammaire ne présente que des propriétés de niveau syntaxique, mais l'intégration de propriétés d'autres niveaux, notamment sémantique, est en cours.

[7] Pour être tout à fait précis, la sémantique des propriétés (*i.e.* le mode de fonctionnement de chacun des opérateurs de propriétés employés) est l'objet d'un fichier séparé, comparable à un DTD, indépendant de la grammaire et des logiciels l'exploitant, de façon à pouvoir en modifier le contenu sans avoir à modifier le reste, et *vice versa*.

[8] Le fait que nous n'utilisions actuellement que des opérateurs binaires constitue une coïncidence et n'exclut en rien le fait qu'on puisse en définir d'autres, non binaires, pour des propriétés différentes.

[9] Pour plus de détails à propos des opérations des propriétés, on peut se référer à Van Rullen 2003.

ques extrêmement fines de disposer d'un modèle suffisamment souple, complet et accessible – modifiable –. Nous proposons à cette fin d'associer la *flexibilité* du formalisme des GP (entièrement basé sur les contraintes), à la *généricité* du XML (pour faciliter l'exploitation des données), et au *confort* de la plate-forme de développement Accolade (permettant une représentation graphique intégrale des informations linguistiques, et donc une visualisation des relations nettement facilitées).

Références

Balfourier JM., Blache P., Van Rullen T. (2002), From shallow to deep parsing using constraint satisfaction, in *proceedings of COLING-2002*.

Blache P. (2001), *Les Grammaires de Propriétés : des contraintes pour le traitement automatique des langues naturelles*, Paris, Hermès Sciences Publications.

Blache P., Hirst D. (2001), Aligning prosody and syntax in Property Grammars, in *proceedings of EuroSpeech 2001*.

Blache P., Van Rullen T. (2002), An evaluation of different symbolic shallow parsing techniques, in *proceedings of LREC-02*.

Blache P., Guénot ML., Van Rullen T. (2003), Corpus-based grammar development, in *proceedings of Corpus Linguistics 2003*, 124-131.

Blache P. (2003), Vers une théorie cognitive de la langue basée sur les contraintes, in *actes de TALN-2003* (à paraître).

Butt M., King T., Nino ME., Segond F. (1999), *A Grammar Writer's Cookbook*, CSLI Publications.

Kaplan R., King T., Maxwell J. (2002), Adapting existing grammars: the XLE experience, in *proceedings of Workshop on Grammar Engineering and Evaluation (COLING-02).*

Kermes H., Evert S. (2003), Text analysis meet corpus linguistics, in *proceedings of Corpus Linguistics 2003*, 402-411.

Kinyon A., Prolo C. (2002), A classification of grammar development strategies, in *proceedings of Workshop on Grammar Engineering and Evaluation (COLING-02).*

Simov K., Peev Z., Kouylekov M., Simov A., Dimitrov M., Kiryakov A. (2001), CLaRK – an XML-based system for corpora development, in *proceedings of Corpus Linguistics 2001*, 558-560.

Simov K., Kouylekov M., Simov A. (2002), Cascaded regular grammars over XML documents, in *proceedings of the 2nd Workshop on NLP and XML (NLPXML-2002)*.

Van Rullen T., Guénot ML., Bellengier E. (2003), Formal representation of Property Grammars, in *proceedings of ESSLLI 2003 Student Session* (à paraître).