Synchronisation syntaxesémantique, des grammaires minimalistes catégorielles (GMC) aux Constraint Languages for Lambda Structures (CLLS)

Amblard Maxime (1)
(1) LaBRI -Université de Bordeaux 1
351 cours de la libération
33405 Talence cedex amblard@labri.fr
date de soutenance prévue : novembre 2006

Motselefs – Keywords

logique, grammaires minimalistes catégorielles, λealcul, portée des quantificateurs, Constraint Language for Lambda Structures
logic, minimalist grammars, λealculus, quantifiers scope, Constraint Language for Lambda Structures

Résumé -Abstract

Ces travaux se basent sur l'approche computationelle et logique de Ed Stabler (**?**), qui donne une formalisation sous forme de grammaire du programme minimaliste de Noam Chomsky (**?**). La question que je veux aborder est comment, à partir d'une analyse syntaxique retrouver la forme prédicative de l'énoncé. Pour cela, il faut mettre en place une interface entre syntaxe et sémantique. C'est ce que je propose en utilisant les Grammaires Minimalistes Catégorielles (GMC) extension des GM vers le calcul de Lambeck. Ce nouveau formalisme permet une synchronisation simple avec le λealcul.

Parmi les questions fréquemment rencontrées dans le traitement des langues naturelles, j'interroge la performance de cette interface pour la résolution des problèmes de portée des quantificateurs. Je montre pourquoi et comment il faut utiliser un λealcul plus élaboré pour obtenir les différentes lectures, en utilisant Constraint Languages for Lambda Structures CLLS.

This work is based on the computational and logical approach of Ed Stabler (**?**), which gives a formalization of the minimalist program of Noam Chomsky (**?**). The question I want to solve is, starting from a syntactic analysis, how to find the predicative forms of a sentence.

I propose an interface between syntax and semantic by using Categorical Minimalists Grammars (CMG) extension of the MG towards the Lambeck calculus and Constraint Language for Lambda Structures (CLLS). This interface is powerful for the resolution of quantifier scope ambiguities.

Introduction

Dans un premier temps, le formalisme des GMC, extension des grammaires minimalistes vers le calcul de Lambeck, (**?**) et (**?**), sera présenté. Puis une interface entre syntaxe et sémantique -calcul des formes prédicatives en logique d'ordre supérieur -sera exposée. Mais cette interface ne suffit pas pour conserver tous les calculs possibles avec les grammaires de Lambeck. Dans une seconde partie, j'étudie une solution qui consiste à utiliser des λtermes structurés avec contexte, via CLLS (**?**).

1 Grammaires Minimalistes Catégorielles -GMC

1.1 Syntaxe

Ce formalisme, présenté dans (**?**), est l'extension de la formalisation des grammaires minimalistes de Stabler (**?**). Tout comme ces dernières, les GMC sont lexicalisées et les expressions sont des arbres finis, ordonnées avec projection. La projection est la relation entre les éléments permettant de retrouver la tête du constituant.

Les fonctions génératrices dans les grammaires minimalistes sont de deux types. La première est la fusion qui permet d'agglomérer deux structures. La seconde est motivée par Chomsky comme la nécessité de vérifier certains traits, mettant en relation deux éléments de l'analyse, permettant de faire monter dans la dérivation une feuille en position basse : le déplacement.

Fusion : élimination de / ou \ -structurellement la concaténation droite ou gauche. Ce qui se traduit sous forme de séquents par les formules suivantes :

$$\frac{\Gamma \to x : A/B \quad \Delta \to y : B}{\Gamma, \Delta \to xy : A}\,[e/] \qquad \frac{\Delta \to y : B \quad \Gamma \to x : B\backslash A}{\Gamma, \Delta \to xy : A}\,[e\backslash]$$

Graphiquement, on ajoute une branche binaire à notre arbre entre une nouvelle feuille et la structure actuelle. Le nouveau sommet contiendra la réduction des types.

Déplacement : cette opération se base sur la notion de lien entre deux positions d'une dérivation. Elle se déroule en deux temps. Dans la dérivation principale, on introduit certaines hypothèses. Dans une seconde dérivation, on construit un élément de type $\times$. Si le type des éléments autours du $\times$ est le même que celui des hypothèses dans la dérivation principale, on peut substituer dans cette dernière les éléments de la seconde.

$$\frac{\Gamma \to w : A \times B \quad \Delta, x : A, y : B, \Delta' \to z : C}{\Delta, \Gamma, \Delta' \to let(x, y) = (\pi_1(w), \pi_2(w)) \; in \; z : C}\,[e\times]$$

Où π_1 est la projection de la première composante et π_2 de la deuxième.

Selon l'hypothèse de Chomsky, les formes phonologiques et logiques fonctionnent séparément. Le déplacement met en relation deux positions dans l'analyse et ces formes peuvent se substituer soit en position basse, soit en position haute ce qui motive cette opération.

Graphiquement, on ajoute une branche unaire à l'arbre pour marquer que la substitution a eu lieu. Pour une meilleure visibilité, on marquera les projections reliant les deux dérivations.

La section suivante présente un calcul sémantique parallèle à l'analyse syntaxique, dont un exemple est exposé.

1.2 Interface syntaxe -sémantique

La sémantique utilisée est la forme prédicative des constituants formant l'énoncé. Pour l'obtenir, le λ-calcul est la voie la plus naturelle. Cependant un λ-calcul classique ne permet pas d'aboutir au résultat escompté car les hypothèses utilisées pour le déplacement excluent une construction itérative.

L'analyse syntaxique est supposée normalisée pour rencontrer l'objet puis le sujet, ce qui donne l'ordre des variables dans le λ-terme du verbe.

Les règles de l'interface λ-termes contextués. A chaque règle syntaxique correspond une règle sémantique : la fusion est une application dans la contrepartie sémantique. On aura donc $[\backslash E]$ ou $[/E]$ devenant $[\rightarrow E]$.

L'équivalence du déplacement est un peu plus complexe. Il faut distinguer deux situations. On peut introduire soit une variable de type simple et dans ce cas, il faut faire une application standard, soit une variable de type supérieur, ayant subi un type-raising, donc inverser l'application. Ce qui se traduit par les deux règles suivantes :

$$\frac{\Delta \vdash z : T \rightarrow U \rightarrow V \quad \Gamma \cup [x : T] \vdash y : U}{\Delta \cup \Gamma \vdash z(\lambda x.y) : V}\ [RAISE/]$$

$$\frac{\Delta \vdash z : T \rightarrow U \rightarrow V \quad \Gamma \cup [x : T] \vdash y : U}{\Delta \cup \Gamma \vdash (\lambda x.y)(z) : V}\ [NORAISE/]$$

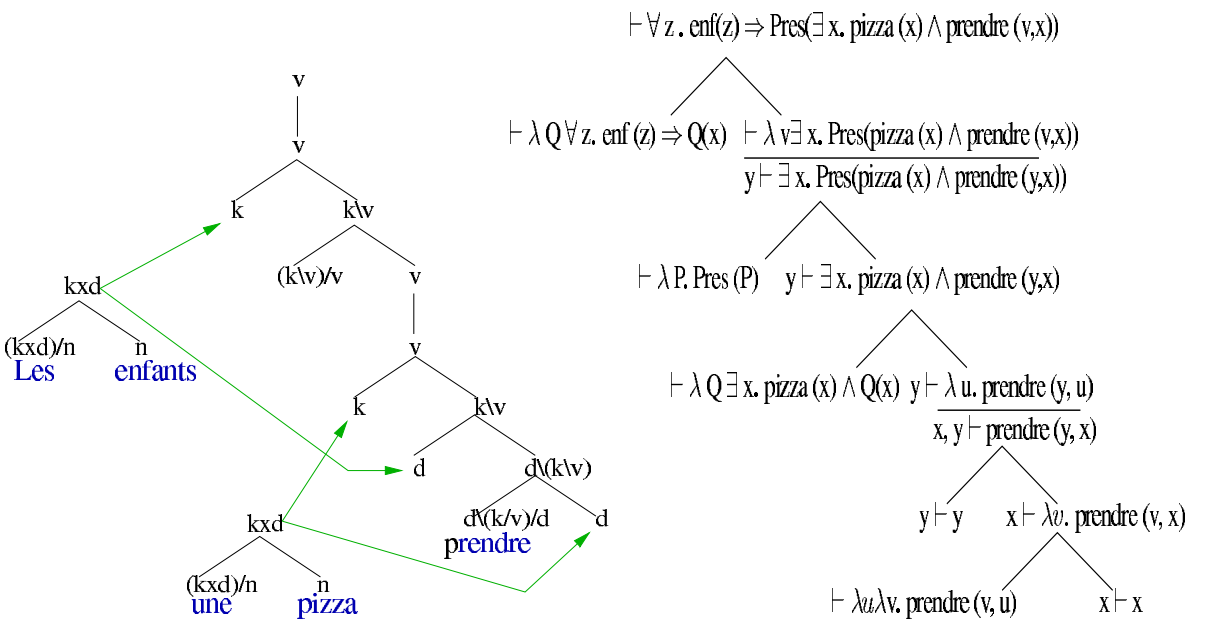
Pour les hypothèses, nous introduisons des variables neutres. Mais, avant qu'un déplacement n'intervienne, une λ-abstraction sur cette variable est nécessaire. Pour éviter de perdre la position relative à cette variable lors d'applications précédentes, on utilise un contexte. Les variables neutres ont une redondance dans le contexte et lors de leur extraction de ce dernier, on abstrait sur la bonne variable. Les contextes se conservent par application et sont marqués par le symbole $\vdash$.

Interface syntaxe-sémantique SYN est le calcul syntaxique ($\times$, / et $\backslash$) et SEM le calcul sémantique ($\rightarrow$, $[RAISE]$ et $[NORAISE]$).

Chaque étape dans un calcul trouve sa contrepartie dans l'autre. Deux preuves, l'une dans SYN et l'autre dans SEM sont dites synchronisées si : chaque feuille dans SEM est en bijection avec une feuille de SYN et chaque étape et sa contrepartie sont réalisées en même temps. Pour synchroniser ces deux calculs, on construit pour la sémantique un arbre binaire en utilisant le λ-terme associé dans la partie syntaxique.

Exemple d'analyse sémantique synchronisée à l'analyse syntaxique : *Les enfants prendront une pizza.*

items lexicaux	λ-terme sémantique	types syntaxique
$prendre$	$\vdash \lambda u \lambda v.\ prendre(v,u)$	$(d\backslash k\backslash v)/d$
les	$\vdash \lambda P \lambda Q \forall x.\ P(x) \rightarrow Q(x)$	$k \times d/n$
une	$\vdash \lambda P \lambda Q \exists x.P(x) \wedge Q(x)$	$k \times d/n$
$pizza$	$\vdash \lambda x.pizza(x)$	n
$enfants$	$\vdash \lambda x.enfant(x)$	n
$infl$	$\vdash \lambda P.Pres(P)$	$(k\backslash v)/v$



SYN SEM

La première partie de l'analyse syntaxique sature les positions du verbe avec des hypothèses dans les deux calculs. On obtient dans SEM : $x, y \vdash prendre(y, x)$ où deux variables neutres ont saturé les positions dans le verbe et sont également présente dans le contexte.

Puis le déplacement relatif à l'objet est déclenché dans SYN, marqué par une branche unaire. Dans SEM, une λabstraction s'opère par extraction d'un élément du contexte. Le déplacement permet l'arrivée effective de la forme logique, ici : $\vdash \lambda Q \exists x.pizza(x) \wedge Q(x)$. Une application est donc possible et suit immédiatement cette arrivée dans la dérivation.

La dérivation se poursuit avec l'inflexion qui apporte dans la dérivation la nécessité d'une hypothèse de type cas. Celle-ci correspond à la vérification que le constituant introduit via sa position aura le bon cas, en l'occurrence le nominatif pour le sujet. Elle déclenche le second déplacement, avec projection des formes logiques et phonologiques. La contrepartie dans SEM est une λabstraction puis une application.

L'analyse syntaxique est terminée et acceptante. Dans le même temps la formule sémantique voulue a été construite : $\vdash \forall z.enf(z) \Rightarrow Pres(\exists x.pizza(x) \wedge prendre(v, x))$.

Cependant, à chaque analyse syntaxique ne correspond qu'une forme sémantique. Or, lorsqu'un énoncé contient des quantificateurs, plusieurs lectures sont possibles. La section suivante présente une extension de l'interface pour calculer les différentes formules.

2 Utilisation de λstructures

CLLS (Constraint Language for Lambda Structures) est un formalisme élaboré par Markus Egg et al en Allemagne. L'idée est de représenter les λtermes par des arbres binaires et de relier ces arbres entre eux par deux types de relations pour permettre la sousspécification.

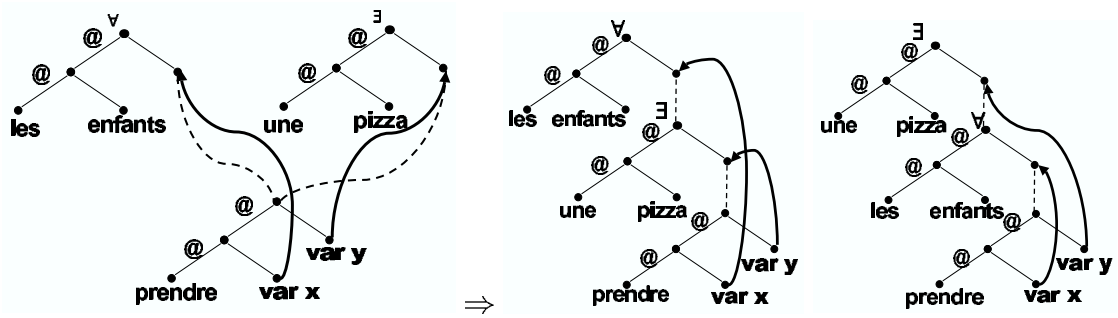
Il existe donc trois types de liens possibles entre les divers éléments de la structure.

1. une arrête d'un arbre qui est une application d'un terme à un autre, notée @.

2. une relation de substitution qui lie une variable à une autre suite à une application entre deux arbres. Graphiquement ce lien sera une flèche en gras.
3. une relation de dominance entre deux arbres, reliant une feuille d'un arbre au sommet d'un autre, graphiquement celuiei sera en pointillés. Elle intervient lorsque deux arbres entrent en relation par application. La racine d'un arbre peut être sujet de plusieurs relations de dominance.

Grâce à ces définitions, la construction de représentations sousspécifiées entre les différents constituants est permise. En effet, les relations de type 2 permettent de conserver l'ordre dans lequel les variables sont substituées et la relation de dominance (type 3) permet de modifier la portée des quantificateurs.

Exemple de CLLS et de réductions : *les enfants prendront une pizza*



lecture
nombre enfants = nombre pizzas

lecture
une unique pizza

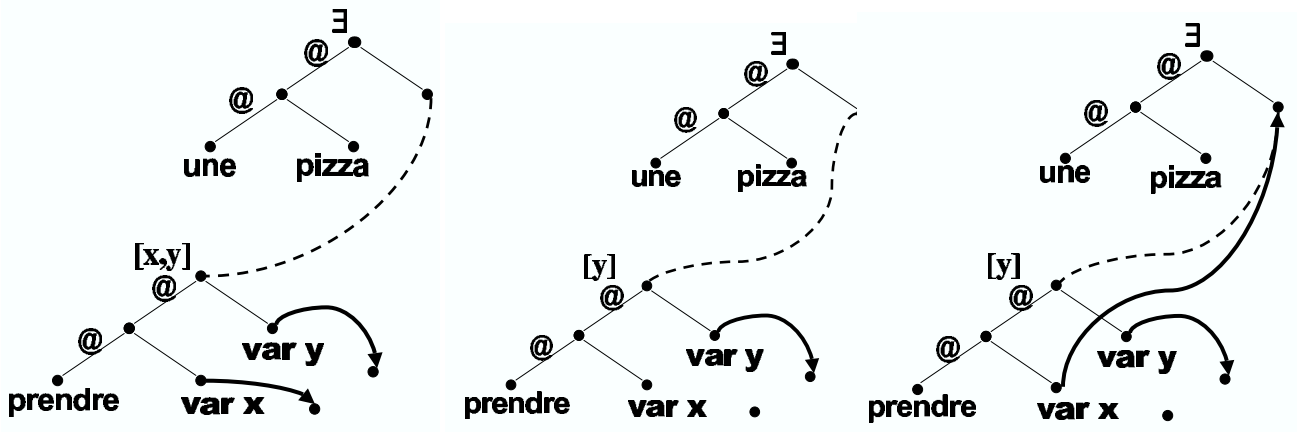
λstructures contextués

Pour synchroniser les GMC et CLLS, il faut conserver la notion de contexte dont on a vu la nécessité dans la partie précédente.

Pour cela, on ajoute sur la racine de la CLLS une liste représentant le contexte, notée entre crochet : [ ]. On admettra la propriété selon laquelle les contextes sont sédentaires. Ce qui signifie que lorsque la structure est liée par une application à une autre structure, on maintient le contexte sur la racine de la sousstructure.

Détails du calcul dans la partie sémantique pour le déplacement :

(1) (2) (3)



(1) verbe chargé de ses deux hypothèses avec redondance dans le contexte

(2) première phase du déplacement, lambda-abstraction, une variable est libérée en fonction du contexte. Ici, celle correspondant à l'objet est à nouveau libre.

(3) application avec l'arbre correspondant à l'objet. Les deux arbres sont maintenant liés et on peut réaliser une réduction de la relation de dominance. La variable esseulée disparaît.

Cette représentation est nécessaire car, par définition, l'apport logique d'un constituant dans les grammaires minimalistes se fait toujours en position haute.

Le défaut de ce modèle est qu'il est finalement trop souple. En effet, pour une phrase avec deux quantificateurs, toutes les lectures ($2! = 2$) sont possibles. Par contre pour une phrase à trois quantificateurs, il n'y a pas 6 ($3! = 6$) lectures mais seulement 5. Une des solutions à l'étude est d'ajouter un ordre sur les relations de dominance. Pour cela il faut différencier ces relations selon leur genre. Une classification sur ces dernières est en cours d'étude.

Conclusion

A partir d'une idée calculatoire simple qui sous-tend la théorie de Stabler et qui encode le programme minimaliste, on obtient un système formel pour l'analyse syntaxique, les GMCs. A partir de cette analyse, j'ai proposé une interface pour le calcul de la sémantique où chaque utilisation de règle dans la partie syntaxique trouve sa contrepartie dans celle sémantique, par l'utilisation de λ-termes structurés avec contexte. L'utilisation du λ-calcul est naturelle pour le calcul de la forme prédicative d'un énoncé, cependant, il ne suffit pas dans sa forme standard. CLLS avec contexte est une solution viable qui permet de reprojeter la structure obtenue vers plusieurs formules et d'obtenir les différentes lectures.

Le concept de synchronisation entre analyse syntaxique et sémantique montre qu'il est difficile de mettre en place un système formel répondant à toutes les caractéristiques de la langue, car il se heurte à de nombreuses questions. Celle de la portée des quantificateurs reste ouverte. Une implémentation à partir de celui des GM devient envisageable.

Références

Amblard M., Lecomte A. et Retoré C. (2004), Synchronization Syntax Semantic for a minimalism theory *Journée Sémantique et Modélisation 2004*,

Chomsky Noam (1995), *The Minimalist Program*, Cambridge, MIT Press.

Egg M., Koller A. et Niehren J. (2001) The Constraint Language for Lambda Structures, *Journal of Logic, Language, and Information*, To appear.

Heim I. et Kratzer A. (1998), *Semantics en Generative Grammar*, Oxford, Blackwell.

Koller A., Burchardt A. et Walter S. (2004), Computational semantics, Actes de *ESSLLI 2004*.

Lecomte A. et Retoré C. (2001), Extending Lambeck Grammars, Actes de *Algebraic Mahods in Lanuage Processing 2003*, 354-361

Stabler Ed. (1997), Derivational Minimalism, Actes de *Logical Aspect of Computational Linguistics 1996*, vol 1328, Springer-Verlag.

Stabler Ed. (1999), Remnant movement and structural complexity, Actes de *Constraints and Resources in Natural Language Syntax and Semantics 1999*, 299-326.