TALN 2004, Fès, 19–21 avril 2004

Un modèle d'acquisition de la syntaxe à l'aide d'informations sémantiques*

D. Dudau Sofronie, I. Tellier
Grappa - Université Lille 3 & INRIA Futurs, France
dudau@grappa.univ-lille3.fr, tellier@univ-lille3.fr

Résumé - Abstract

Nous présentons dans cet article un algorithme d'apprentissage syntaxico-sémantique du langage naturel. Les données de départ sont des phrases correctes d'une langue donnée, enrichies d'informations sémantiques. Le résultat est l'ensemble des grammaires formelles satisfaisant certaines conditions et compatibles avec ces données. La stratégie employée, validée d'un point de vue théorique, est testée sur un corpus de textes français constitué pour l'occasion.

This paper presents a syntactico-semantic learning algorithm for natural languages. Input data are syntactically correct sentences of a given natural language, enriched with semantic information. The output is the set of compatible formal grammars satisfying certain conditions. The strategy used, which has been proved theoretically valid, is tested on a corpus of French texts built for this purpose.

Mots-clefs – Keywords

Grammaires catégorielles, types sémantiques, apprentissage syntaxico-sémantique
Categorial grammars, semantic types, syntactico-semantic learning

1 Introduction

La modélisation du phénomène de l'acquisition du langage par les enfants suscite un intérêt croissant ces dernières années (voir les conférences CoNNL, (Brent 96)). Ce sujet se situe au croisement de la linguistique, du traitement automatique du langage naturel et des sciences cognitives. Les travaux évoqués ici sont issus d'une recherche ayant ses fondements dans la théorie des langages formels et l'apprentissage automatique et se concentrent sur deux niveaux de traitement de la langue : *la syntaxe* et *la sémantique*. L'objectif est de définir un *modèle d'acquisition du langage*, et plus précisément de la *grammaire*, qui soit à la fois crédible d'un point

*Recherche effectuée dans le cadre de l'ARC INRIA Gracq et de la MSH-Institut International Erasme

de vue psycholinguistique et calculable. Les modèles utilisés ici sont : *les grammaires catégorielles classiques* (Bar Hillel et al. 60) pour représenter la syntaxe et *la logique de Montague* (Montague 74) comme source d'inspiration pour la sémantique.

Les grammaires catégorielles, par leur nature lexicalisée, sont bien adaptées à un processus d'apprentissage : en effet, leurs règles sont exprimées par un nombre réduit et invariable de schémas. Apprendre une telle grammaire signifie donc simplement apprendre à associer des catégories syntaxiques aux items lexicaux. L'étude de l'apprentissage de ces grammaires a connu des développements importants ces dernières années (Adriaans 1992; Kanazawa 98; Bonato,Retore 01; Besombes,Marion 03). La pertinence des grammaires catégorielles dans le contexte du langage naturel a été justifiée par plusieurs travaux (Oehrle et. al 88). Or, un des intérêts majeurs de ce type de grammaires est leur connexion avec la sémantique, qui s'appuie sur le Principe de Compositionnalité (Janssen 97). Ainsi, il nous semble naturel d'adopter *une stratégie d'apprentissage de la syntaxe basée sur la sémantique*, c'est-à-dire de considérer que la capacité d'acquérir une grammaire à partir d'énoncés est conditionnée par celle de construire une représentation de la situation décrite par ces énoncés (Pinker 94). Nous présentons ici une nouvelle manière de concevoir le Principe de Compositionnalité et de le rendre partie prenante de la stratégie d'apprentissage. L'information sémantique utilisée n'est pas la représentation logique complète des mots ou des phrases mais seulement leur *type sémantique*. Ce type distingue les entités (individus), les valeurs de verité et les propriétés (prédicats). L'hypothèse faite est que cette information est extraite de l'environnement par l'apprenant.

La validation de notre modèle d'apprentissage passe par une double démarche, théorique et expérimentale. D'un point de vue théorique, nous avons défini une nouvelle classe de grammaires catégorielles classiques et nous avons déjà montré que cette classe est apprenable dans le modèle d'apprentissage à la limite de Gold (Gold 67) à l'aide d'informations sémantiques (Dudau-Sofronie et al. 01; Dudau-Sofronie et al. 03). Mais, dans cet article, nous développerons plutôt le versant expérimental de notre travail. Nous avons en effet implémenté notre algorithme et nous avons cherché à le tester sur un corpus de textes en français, spécialement conçu dans ce but.

Cet article débute par une courte description des formalismes utilisés, tant au niveau de la syntaxe que de la sémantique. Ensuite, notre algorithme d'apprentissage est expliqué sur un exemple. Enfin, nous présentons les principales étapes de la constitution du corpus et les tests expérimentaux réalisés pour valider notre stratégie.

2 Formalismes employés

2.1 Grammaires Catégorielles Classiques

Pour définir une grammaire catégorielle, il faut un ensemble de catégories basiques, une assignation de catégories aux mots et des règles de réduction applicables sur les catégories. La notation que nous adoptons[1] ici pour les catégories est préfixée, sous forme des termes, qui correspondront à $/(A, B)$ pour B/A et à $\backslash(A, B)$ pour $A\backslash B$. Ainsi, la catégorie impliquée dans une réduction (en tant qu'argument) se trouve toujours sur la première position d'un terme.

Soit Σ un vocabulaire fini. Soit $\mathcal{B}$ un ensemble dénombrable de catégories basiques contenant

[1] Par rapport à la notation classique B/A, $A\backslash B$.

une catégorie spéciale $S \in \mathcal{B}$, dénommée l'axiome. L'ensemble de toutes les catégories obtenues à partir de $\mathcal{B}$ peut être vu comme l'algèbre $Cat(\mathcal{B})$ des termes construits sur $\mathcal{B}$ avec deux opérateurs binaires : $/$, $\backslash$. $Cat(\mathcal{B})$ est ainsi le plus petit ensemble tel que : (1) $\mathcal{B} \subset Cat(\mathcal{B})$ et (2) si $A \in Cat(\mathcal{B})$ et $B \in Cat(\mathcal{B})$ alors : $/(A, B) \in Cat(\mathcal{B})$ et $\backslash(A, B) \in Cat(\mathcal{B})$.

Une **grammaire catégorielle** sur Σ est une relation finie entre Σ et $Cat(\mathcal{B})$: $G \subseteq \Sigma \times Cat(\mathcal{B})$ et G finie. Une **grammaire catégorielle classique (GCC)** est une grammaire catégorielle qui admet seulement deux règles de réduction pour toutes catégories A et B de $Cat(\mathcal{B})$.
– forward application $FA : /(A, B)\ A \rightarrow B$;
– backward application $BA : A\ \backslash(A, B) \rightarrow B$.
Les règles de réduction justifient les notations fractionnelles des catégories construites avec les deux opérateurs $/$ et $\backslash$. Les termes $/(A, B)$ et $\backslash(A, B)$ sont des *foncteurs orientés* attendant, comme *argument* à droite (resp. à gauche), la catégorie A et donnant, comme résultat, la catégorie B. Nous notons par $\langle a, A\rangle \in G$, où $a \in \Sigma$ et $A \in Cat(\mathcal{B})$ l'assignation de la catégorie A au mot a, dans G. Le langage engendré par une telle grammaire est l'ensemble des suites d'éléments du vocabulaire Σ auxquelles on peut faire correspondre une suite de catégories de $Cat(\mathcal{B})$ qui se réduit par les règles de réduction FA et BA à la catégorie S.

Soit par exemple G une GCC élémentaire : $G = \{\langle homme, NC\rangle, \langle Jean, T\rangle, \langle petit, /(NC, NC)\rangle, \langle un, /(NC, T)\rangle, \langle court, \backslash(T, S)\rangle, \langle regarde, \backslash(T, /(T, S))\rangle\}$. Les catégories basiques T et NC signifient respectivement "terme" et "nom commun". Pour les déterminants, nous choisissons ici une catégorie simplifiée par rapport à celle utilisée dans la tradition de Montague, ne rendant pas compte de son caractère de quantificateur mais adaptée aux GCC. Cette grammaire reconnaît des phrases comme "un homme court", "Jean regarde un petit homme", etc.

2.2 Types sémantiques

Montague (Montague 74) a été le premier à proposer une logique typée pour représenter la sémantique des langues naturelles. Cette notion de type est depuis devenue classique. C'est elle que nous retenons ici comme information sémantique. Formellement, l'ensemble des types est construit récursivement à partir d'un ensemble de types basiques. L'ensemble basique le plus usuel est $\Theta = \{e, t\}$. Le type basique e est traditionnellement le type des entités élémentaires du modèle logique et le type t dénote les valeurs de vérité. Pour tout ensemble de types basiques Θ tel que $t \in \Theta$, l'ensemble de tous les types $Types(\Theta)$ est le plus petit ensemble tel que : $\Theta \subset Types(\Theta)$ et pour tout $u \in Types(\Theta)$ et $v \in Types(\Theta)$, $(u, v) \in Types(\Theta)$. Le type (u, v) est un foncteur attendant comme argument le type u pour donner comme résultat le type v. Classiquement, les identifiants de personne ("Jean", etc.) et les termes sont de type e, tandis que les noms communs et les verbes intransitifs, qui réfèrent tous les deux à un prédicat à une place, sont de type (e, t). Les verbes transitifs sont, eux, de type $(e, (e, t))$. Les adjectifs comme "petit" sont des modifieurs de noms communs, ils sont donc de type $((e, t), (e, t))$.

Il y a bien sûr un lien étroit entre la notion de catégorie utilisée dans les grammaires catégorielles et les types sémantiques. Les deux s'expriment par des foncteurs. Nous pouvons en fait caractériser plus précisément ce lien grâce à la notion de **fonction de typage**. Une fonction de typage h est un morphisme de $Cat(\mathcal{B})$ vers $Types(\Theta)$ satisfaisant les conditions suivantes : (1) $h(S) = t$; (2) $\forall X \in \mathcal{B}$, on a : $h(X) \in Types(\Theta)$ (3) $\forall X, Y \in Cat(\mathcal{B})$: $h(/(Y, X)) = h(\backslash(Y, X)) = (h(Y), h(X))$. Le Principe de Compositionnalité affirme que le sens d'une phrase ne dépend que du sens des mots qui la constituent et de sa structure syntaxique. Il se traduit habituellement par une similarité de structure entre les arbres syntaxique et sémantique. Dans la mesure où les

catégories et les types sont des *structures lexicalisées*, la fonction de typage peut être considérée comme l'expression de la *lexicalisation du Principe de Compositionnalité*.

Si nous posons : $h(T) = e, h(S) = t, h(NC) = (e,t)$, nous avons défini une fonction de typage parfaitement compatible avec la grammaire G donnée en exemple précédemment, et avec la sémantique de son vocabulaire. Remarquons qu'à une catégorie basique (par exemple NC) peut être associé un type non basique et que deux catégories différentes (non basiques) peuvent donner un type identique, puisque $h(\backslash(T,S)) = (h(T), h(S)) = (e,t) = h(NC)$.

3 Inférence de grammaires à partir de phrases typées

Nous cherchons à modéliser et à simuler l'aide qu'apporte la connaissance d'informations sémantiques dans le processus d'acquisition de la syntaxe. L'algorithme que nous exposons brièvement ici (et décrit en détail dans (Dudau-Sofronie et al. 01)) prend comme données d'entrées des *phrases typées*, c'est-à-dire des énoncés syntaxiquement corrects d'une langue donnée, où chaque mot est associé à son type sémantique. Il donne comme résultat un ensemble de GCCs, chacune associée à sa fonction de typage.

Remarquons tout d'abord que les types non basiques se présentent sous la forme de foncteurs. Mais, contrairement aux catégories syntaxiques des GCCs, ces foncteurs ne sont pas *orientés*. On peut néanmoins préciser comment ils se combinent les uns avec les autres, à la façon des règles FA et BA utilisées dans les GCC. $\forall u, v \in \mathit{Types}(\Theta)$, on a les règles suivantes :

- Type Forward TF : $(u,v)\ u \rightarrow v$;
- Type Backward TB : $u\ (u,v) \rightarrow v$.

Les types donnent donc des indications sur la nature de foncteur ou d'argument des éléments du vocabulaire auxquels ils sont associés, mais où la direction de l'opérateur, $\backslash$ ou $/$ est perdue. Ils sont en quelque sorte des *catégories syntaxiques dégradées*. Mais la dégradation subie est régulière, puisque c'est la fonction de typage, qui est un morphisme, qui en est responsable. Ce sera tout l'enjeu de notre algorithme d'apprentissage de reconstituer des catégories syntaxiques à partir de ces types. Ce processus sera traité en deux étapes : *une étape d'analyse* et *une étape de déduction des catégories*.

3.1 L'analyse des phrases typées

Comme toute phrase typée, fournie comme donnée d'entrée, est issue d'une phrase syntaxiquement correcte, la séquence des types associés aux mots de la phrase peut, elle, être réduite au type t, en utilisant les règles TF et TB comme le montre la Figure 1 (à gauche).

L'*étape d'analyse* de l'algorithme global est en fait plus complexe et consiste en deux étapes :
– une étape de variabilisation des types de toutes les phrases typées, qui consiste à introduire des variables distinctes dans les expressions de type, en position d'opérateur, sachant qu'une assignation formée par le même mot et le même type reçoit la même variabilisation. Ainsi, la phrase typée $\langle un, ((e,t),t)\rangle\ \langle homme, (e,t)\rangle\ \langle court, (e,t)\rangle$ devient la phrase typée variabilisée $\langle un, x_1(x_2(e,t),t)\rangle\ \langle homme, x_3(e,t)\rangle\ \langle court, x_4(e,t)\rangle$.
– une étape de recherche des réductions à t de la séquence des types dans chaque phrase typée variabilisée (Figure 1 à droite). Cette analyse sert à identifier la direction dans laquelle chaque

foncteur va trouver ses arguments. Si la règle TF (resp. TB) est employée dans une analyse, cela signifie qu'au niveau syntaxique, c'est la règle FA (resp. BA) qui devait être employée, et donc un opérateur $/$ (resp. $\backslash$) a été identifié là où une variable avait été introduite. Dans l'arbre de la Figure 1, on a fait figurer ces identifications : $x_1 = /$ est induit lors de l'application de la règle TF et $x_4 = \backslash$ lors l'application de la règle TB. De plus, pour que les règles TF et TB soient applicables sur les types variabilisés, il faut aussi que certaines conditions d'égalité entre sous-types soient vérifiées : dans l'arbre de la Figure 1, les deux sous-types soulignés doivent être égaux pour que TF puisse s'appliquer. Ces conditions entraînent à leur tour une (ou des) égalité(s) entre variables : $x_2 = x_3$ ici. Pour implémenter cette analyse, nous nous sommes inspirés d'algorithmes classiques dans les grammaires hors-contexte.

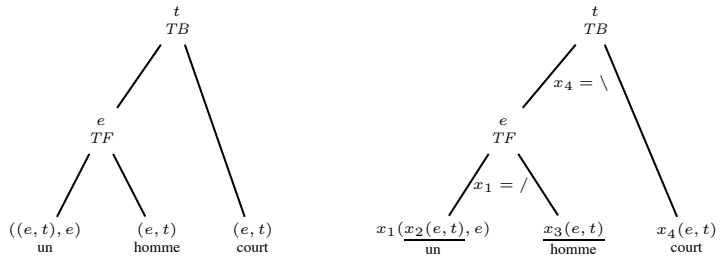

FIG. 1 – arbres d'analyses pour une phrase typée et pour une phrase typée variabilisée

À l'issue de cette étape, on obtient en fait des ensembles de contraintes sur les variables introduites dans les types variabilisés, qui peuvent aussi se traduire sous forme de substitution(s) sur ces variables (on ne détaille pas cet aspect ici).

3.2 La déduction des catégories et des fonctions de typage

L'algorithme ne sait rien des catégories syntaxiques possibles : il doit les définir. Comme un type non basique peut correspondre à une catégorie basique (par exemple (e, t) qui correspond à NC), il est nécessaire d'appliquer une étape supplémentaire de *déduction de catégories*, après l'analyse. L'idée est dans ce cas qu'un tel type ne mettra jamais en oeuvre sa nature de foncteur dans une analyse puisque, au niveau syntaxique, il n'en est pas réellement un.

La déduction des catégories s'effectue par les règles suivantes : (1) chaque plus petit sous-type variabilisé distinct qui est utilisé en tant qu'argument dans une analyse est associé à une catégorie basique nouvelle (2) le type t en tant que résultat est, par convention, toujours associé à la catégorie axiomatique S. La fonction de typage est directement déduite de ces règles.

La GCC G induite par notre algorithme avec la phrase typée donnée en exemple dans la Figure 1 est donc la suivante : $G = \{\langle un, /(A, B)\rangle, \langle homme, A\rangle, \langle court, \backslash(B, S)\rangle\}$ avec A et B catégories basiques nouvelles vérifiant : $h(A) = (e, t)$, $h(B) = e$. Cette solution coïncide, à un renommage des catégories basique près (A pour NC, B pour T) avec la GCC "naturelle" citée en exemple jusqu'à présent. Comme "Jean" est de type e (donc de catégorie B dans G), cette grammaire reconnaît aussi la phrase « Jean court », elle a donc généralisé par rapport à l'exemple qui lui a été présenté. Sur cet exemple, une seule grammaire est apprise mais dans le cas général, le résultat de l'algorithme est un ensemble de grammaires, chacune associée à sa fonction de typage. Cette stratégie peut être appliquée à un échantillon de phrases en entrée. Elle induit des contraintes pour chaque phrase et, de manière incrémentale, propage les

contraintes compatibles (aucune variable ne peut être à la fois égale à / et à \) d'une phrase à une autre. Nous avons étudié les propriétés formelles de cet algorithme et défini les conditions sur les GCC et les fonctions de typage qui assurent que le résultat fourni est correct et complet.

4 Expérimentations et résultats

Notre approche était, à l'origine, essentiellement théorique. Nous nous sommes initialement placés dans le cadre du modèle d'apprentissage de Gold, dans lequel le critère d'apprenabilité ne requiert aucune validation expérimentale, ni même de "praticabilité" quant à la complexité algorithmique. Mais l'algorithme que nous proposons mérite d'être testé sur des données réelles. Son principal inconvénient est qu'il nécessite des données d'entrée très spécifiques (des phrases typées), qu'aucun corpus actuel ne fournit. Néanmoins, ces données sont moins coûteuses à produire que les corpus arborés généralement nécessaires pour tester les techniques d'inférence grammaticale appliquées au langage naturel (Sakakibara 92; Kanazawa 98; Bonato,Retore 01; Besombes,Marion 03). Les types, en effet, sont des données lexicalisées et nous avons donc eu l'idée de produire un ensemble de phrases typées à partir de textes étiquetés par un tagger lexical. D'un point de vue psycholinguistique, la donnée de phrases typée est aussi plus crédible que la donnée de structures arborescentes. Nous décrivons ici la démarche suivie pour constituer ce corpus, et la nature des expériences que nous avons menées pour valider notre approche et tester certaines hypothèses.

4.1 Constitution d'un corpus

Le point de départ de notre démarche est la recherche d'un ensemble de textes[2]. Nous avons aussi identifié les besoins suivants : (1) Nous souhaitions un corpus de textes en français, afin d'être à même de valider la qualité des grammaires qui seront apprises, et si possible libres de droit. (2) Les textes doivent être écrits dans une langue homogène. Notre algorithme n'est pas spécialisé dans l'acquisition d'une langue plutôt qu'une autre : il apprend celle à laquelle appartiennent les phrases qui lui sont soumises. Mais il n'apprendra quelque chose que s'il existe des grammaires compatibles avec l'ensemble des phrases typées qui lui sont proposées. (3) Comme notre algorithme prétend simuler ce que fait peut-être un enfant en phase d'acquisition de sa langue maternelle, il est naturel de lui fournir des phrases simples, avec un vocabulaire si possible limité et répétitif.

Les textes de livres pour enfant, auxquels nous avons d'abord pensé, ne convenaient pas car leur langue regorge de dialogues. Or, les énoncés intervenant dans les dialogues ne sont pas des propositions au sens logique : ils ne sont pas de type t, il faudrait donc envisager un typage spécifique pour les prendre en compte. Nous nous sommes finalement fixés sur une collection de textes *produits* par des enfants, et disponible sur le site www.momes.net. Ces textes sont des récits écrits par des enfants de 6 à 10 ans, et corrigées par des adultes. Ils sont constitués de phrases assez courtes, qui ne dépassent pas 10 mots, et correctes du point de vue syntaxique.

Ces textes ont été nettoyés à la main (pour en retirer, par exemple, les parties dialoguées) et ont subi des traitements automatiques successifs, présentés sur la Figure 2. Ils ont d'abord été

[2]T. Desvenain, étudiant en Maîtrise de science du langage à l'Université Lille 3 a contribué à ce travail

soumis à un étiqueteur lexical, en l'occurrence Tree Tagger[3] (étape I sur le schéma). De l'étiquetage complet fourni par le tagger, on n'a retenu que les informations concernant le rôle de chaque mot dans la phrase (nom, verbe, adjectif, adverbe, déterminant, etc.) et son lemme. La partie cruciale et spécifique de la constitution de notre corpus est la définition d'une *table de correspondance* entre les étiquettes lexicales et les types sémantiques. Pour les principales étiquettes, la correspondance est immédiate : nom propre, pronom personnel - e ; nom commun - (e, t) ; adjectif - $((e, t), (e, t))$; déterminant - $((e, t), e)$. Néanmoins cette correspondance n'est pas toujours unique : le Tree tagger ne fait par exemple pas de distinction entre les verbes intransitifs et les verbes transitifs, qui doivent pourtant recevoir des types sémantiques différents. La table associe donc à chaque étiquette un *ensemble de types possibles* (étape II). Une fois les étiquettes transformées en types, il faut éliminer les phrases typées incorrectement, c'est-à-dire ne se réduisant pas à t. Cette phase est la plus coûteuse, elle est réalisée en appliquant un algorithme d'analyse simplifié, avec les règles TF et TB, sur tous les typages possibles (étape III). Enfin, il reste encore une étape de sélection manuelle pour arriver à un seul typage par phrase.

Le corpus final, au format XML, contient du texte annoté avec les étiquettes lexicales et avec les types sémantiques. Il est constitué de 1008 phrases contenant 5851 mots parmi lesquels 2632 sont différents. Un point essentiel à comprendre est que même si deux mots distincts (par exemple deux noms communs) ont été étiquetés de la même façon par le tagger, ils ne seront pas considérés par notre algorithme comme appartenant a priori à la même catégorie syntaxique. En effet, lors de la phase de variabilisation des types, des variables différentes seront introduites dans les types de ces deux mots. Ce n'est que si une contrainte d'égalité entre ces variables est inférée que ces mots seront reconnus comme appartenant à la même catégorie syntaxique. Et cette contrainte d'égalité ne sera elle-même inférée que si les deux mots apparaissent dans un même contexte (deux phrases ne différant que par ces mots, par exemple). Ce mécanisme permet de distinguer les catégories NC et $\backslash(T, S)$, correspondant pourtant au même type (e, t). La contrepartie est que l'apprentissage ne sera efficace qu'en présence de redondances et de répétitions. C'est ce que les expériences qui suivent vont nous permettre de mesurer.

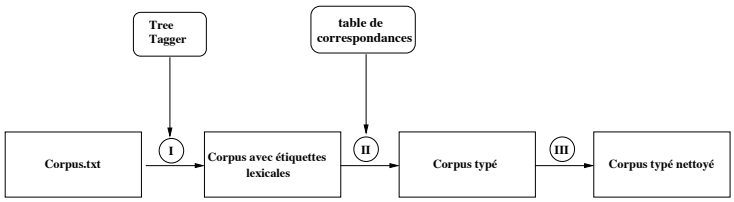

FIG. 2 – Traitements automatiques successifs pour obtenir un corpus typé

4.2 Efficacité et validité du programme

Les propriétés de notre algorithme que des expériences sur corpus permettront de tester sont : son *efficacité* et sa *validité*. L'efficacité mesure la praticabilité du programme, en termes de ressources utilisées. La validité est un critère plus qualitatif, qui dépend de la correction des grammaires obtenues par rapport aux grammaires usuelles du langage naturel. On propose plusieurs expériences pour tester ces paramètres.

Pour ce qui est de l'efficacité, il convient de mesurer tout d'abord la complexité théorique du problème. Si on se contente de tirer aléatoirement des phrases typées de notre corpus et de lancer

[3]http ://www.ims.uni-stuttgart.de/projekte/tc

notre algorithme sur ces données, il y a fort à parier que rien d'intéressant ne sortira. La plupart du temps, en effet, les phrases n'ont aucun mot en commun, sinon quelques mots grammaticaux. Supposons qu'on tire 15 phrases typées et que pour chacune d'elles, l'algorithme infère au moins 2 grammaires différentes compatibles possibles (moyenne calculée sur le corpus). Si les phrases n'ont aucun mot en commun, alors le nombre total de grammaires compatibles avec l'ensemble des données sera, par une simple combinatoire, supérieur à $2^{15} = 32768$ grammaires. C'est à cette aune qu'il faut se comparer. Lorsque nous avons fait cette expérience, nous avons en fait obtenu, en moyenne sur 8 tirages différents, 2560 grammaires. La taille du lexique, pour les 15 phrases tirées était, en moyenne, de 85 mots parmi lesquels, en moyenne, 81 étaient différents. Chacun des 4 mots qui se répètent a donc, en moyenne, divisé par 2 de nombre de grammaires solution. Ce genre de calculs prend beaucoup de temps. Dans ce qui suit, nous ne nous attacherons pas à la complexité en temps, en revanche, le nombre de grammaires obtenues à chaque étape en sortie du programme est une évaluation des ressources de mémoire mises en oeuvre au cours de l'apprentissage. Quels sont les facteurs qui peuvent contribuer à réduire ce nombre et donc à améliorer l'efficacité de l'apprentissage ? Nous faisons l'hypothèse que deux facteurs indépendants doivent intervenir : d'une part l'ordre de présentation des phrases, d'autre part la redondance du vocabulaire.

Il est naturel de supposer que face à un novice, l'entourage va avoir tendance à d'abord produire des énoncés simples, comprenant un nombre limité de mots, avant de produire des phrases à la structure plus complexe. Nous voulons mesurer l'importance de l'effet facilitateur obtenu en contrôlant l'ordre de présentation des phrases typées à notre algorithme. Le protocole employé est le suivant : (a) tirer aléatoirement une présentation, contenant n phrases, du corpus initial ; (b) calculer le nombre de grammaires obtenues après chaque phrase ; (c) ordonner en ordre croissant, respectivement décroissant par taille, les phrases dans la présentation tirée, et réinitialiser le processus d'apprentissage. Les résultats obtenus en moyenne en suivant ce scénario sont présentés sur le premier graphique de la Figure 3, pour les 8 présentations de taille $n = 15$ tirées du corpus, évoquées précédemment. L'effet est visible, mais limité.

Mais, comme nous l'avons déjà remarqué, ce dont notre algorithme a crucialement besoin, c'est de mots qui se répètent d'une phrase à une autre. Pour générer artificiellement des énoncés où de telles répétitions se produisent, nous avons automatiquement opéré des substitutions de vocabulaire sur les phrases extraites du corpus, avant de les soumettre au programme. Nous avons ainsi réduit les mots étiquetés respectivement par le tagger comme pronoms personnels, noms, déterminants, déterminants possessifs et verbes transitifs à uniquement deux instances différentes possibles (tirées au sort) pour chaque étiquette. Le résultat obtenu en moyenne, pour 5 présentations de 30 phrases construites sur un lexique initial de 191 mots en moyenne, ramenées par les substitutions à 50 mots en moyenne, est donné dans le deuxième graphique de la Figure 3. Sur la même figure, sont représentées aussi les courbes où l'ordre sur la taille à été inversé et tiré aléatoirement. L'effet facilitateur de l'ordre de présentation est nettement plus visible et le nombre final de grammaires obtenues bien plus raisonnable que précédemment. On constate sur ces courbes des variations importantes : la répétition d'un seul mot typé déjà rencontré auparavant peut provoquer une élimination de beaucoup de grammaires candidates par la simple détection d'une incompatibilité ($x_i = /$ et $x_i = \backslash$ pour une certaine variable x_i).

Les expériences précédentes ne font que compter le nombre de grammaires résultats : elles ne permettent pas de valider leur pertinence. Pour cela, l'idéal serait de tester les grammaires apprises sur de nouvelles phrases. Mais, en raison du nombre de grammaires résultats et de la difficulté à trouver un ensemble de phrases tests construites sur le même vocabulaire que celles ayant permis l'apprentissage, nous n'avons pu mener à bien de telles expériences. Nous

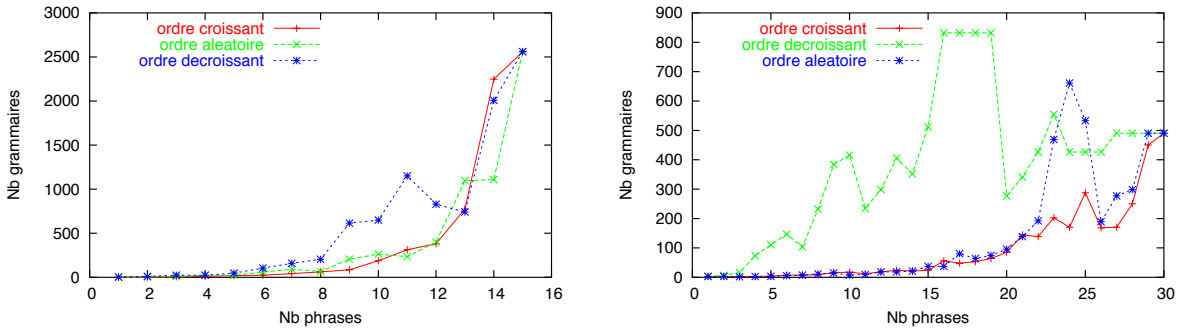

FIG. 3 – Le nombre total moyen de grammaires obtenues à chaque étape pour des présentations de 15 phrases sur le lexique complet (à gauche) et de 30 phrases sur le lexique réduit (à droite)

devons donc chercher d'autres moyens de mesurer la validité de notre programme. Il serait en particulier intéressant de vérifier si les mots étiquetés de la même façon par le tagger sont finalement reconnus comme appartenant à la même catégorie syntaxique dans les grammaires résultats. Nous détaillons dans ce qui suit une étude de cas qui nous semble intéressante, pour une des présentations de 30 phrases avec lexique réduit évoquée précédemment. Nous nous concentrons ici sur les noms communs auquel le type (e, t), variabilisé en $x(e, t)$, a été associé. Le lexique des noms communs est réduit à deux instances distinctes ("soleil" et "fleur"). Les différentes catégories syntaxiques qui peuvent être inférées par le programme à partir de ce type correspondent soit à la catégorie $\backslash(T, S)$, soit à $/(T, S)$ (pour $h(T) = e$), soit à une ou plusieurs catégories basiques. Nous comptons, à chaque étape de la présentation, le nombre de grammaires qui contiennent une de ces instanciations de catégorie pour les noms communs (la valeur $x = /$, n'a jamais été inférée, c'est pourquoi une colonne manque dans le tableau) :

Étape	Nom commun	Cat. basique	Nb occ	Cat avec $x = \backslash$	Nb occ
1	fleur	-	-	-	-
	soleil	A_0	2	-	-
3	fleur	A_1	2	-	-
	soleil	A_0	2	-	-
4	fleur	A_1	3	-	-
	soleil	A_0	3	-	-
7	fleur	A_1	3	$\backslash(T, S)$	1
	soleil	A_0	4	-	-
10	fleur	A_1	9	$\backslash(T, S)$	1
	fleur	A_0	8		
	soleil	A_0	10	-	-
	soleil	A_1	8		
18	fleur	A_2	6	$\backslash(T, S)$	12
	fleur	A_0	6		
	soleil	A_0	18	$\backslash(T, S)$	6
22	fleur	A_1	48	$\backslash(T, S)$	48
	soleil	A_1	18	$\backslash(T, S)$	48
30	fleur	A_2	192	$\backslash(T, S)$	192
	soleil	A_2	192	$\backslash(T, S)$	192

Dès la phrase 3, les deux noms communs ont chacun été présentés au moins une fois. La phrase 7 induit une catégorie incorrecte pour "fleur". Cette phrase et l'analyse incorrecte (présentée synthétiquement sous forme de terme) à laquelle elle a donné lieu est la suivante :
$TB(TB(TB(\langle Pierre, e\rangle, \langle regarde, (e, (e, t))\rangle), \langle ma, ((e, t), e)\rangle), \langle fleur, (e, t)\rangle)$
L'application de la règle TB entre le type de "Pierre regarde", (e, t), et celui de "ma", $((e, t), e)$ n'est pas linguistiquement légitime. Elle entraîne l'identification de $\backslash$ comme valeur possible pour la variable du type associé au nom "fleur". Comme le déterminant "ma" a une seule occurrence dans toute la présentation, cette valeur est propagée à travers d'autres analyses, conduisant à la situation finale décrite dans le tableau, où elle fait jeu égal avec la catégorie basique légitime. Notons par ailleurs qu'entre les phrases 10 et 18, deux catégories syntaxiques basiques

ont été introduites. C'est la propagation de contraintes d'égalité qui finit, à partir de la phrase 22, par unifier ces deux catégories. La même évolution a été observée pour le nombre total d'introductions de catégories basiques lors de présentations quelconques. Mais, pour discréditer la catégorie $\backslash(T, S)$ associée aux noms communs, et les associer à une seule catégorie syntaxique basique, il suffirait d'une seule phrase ou "ma" introduit un nom commun en position sujet.

5 Conclusion

Nous proposons dans cet article un programme d'inférence grammaticale dirigée par la sémantique. Les grammaires formelles qui représentent la syntaxe ont de bonnes propriétés et sont adaptées à un apprentissage syntaxico-sémantique. Les résultats théoriques ont été testés sur des données réelles du langage naturel. Les expériences réalisées sur corpus, bien qu'encore limitées, montrent que l'acquisition du premier langage est un processus complexe, très sensible à un environnement propice. Ainsi, la taille du lexique et l'ordre dans lequel les phrases sont présentées ont une grande influence sur les performances du système.

Références

ADRIAANS P. W. (1992), *Language Learning from a Categorial Perspective.* PhD thesis, University of Amsterdam, Amsterdam, The Netherlands.

BAR-HILLEL Y., GAIFMAN C., SHAMIR E. (1960), On categorial and phrase structure grammars. *Bulletin of the Research Council of Israel*, 9F.

BESOMBES J., MARION J.Y. (2003), Apprentissage de langages réguliers d'arbres et applications. In *TAL*, vol 44, n°1/2003, pages 121–153.

BONATO R., RETORÉ C. (2001) Learning rigid lambek grammars and minimalist grammars from structured sentences. In *Proceedings of the Third Learning Language in Logic Workshop*, pages 23–34.

BRENT M. R. (1996), *Computational Approaches to Language Acquisition.* MIT Press

DUDAU-SOFRONIE D., TELLIER I., TOMMASI M. (2001), From logic to grammars via types. In *Proceedings of Learning Language in Logic (LLL) 2001*, pages 35–46.

DUDAU-SOFRONIE D., TELLIER I., TOMMASI M. (2003), A learnable class of Classical Categorial Grammars from typed examples. In *Proceedings of the 8th conference on Formal Grammar*, pages 77–88

GOLD E. M. (1967), Language identification in the limit. *Information and Control*, 10 : pages 447–474

JANSSEN T. M. V. (1997) Compositionality. In J. V. Benthem and A. ter Meulen, editors, *Handbook of Logic and Language*, pages 417–473. MIT Press.

KANAZAWA M. (1998), *Learnable Classes of Categorial Grammars*, The European Association for Logic, Language and Information. CLSI Publications.

MONTAGUE R. (1974), *Formal Philosophy ; Selected papers of Richard Montague*, Yale University Press.

OEHRLE R.T., BACH E., WHEELER D., editors. *Categorial Grammars and Natural Language Structures.* D. Reidel Publishing Company, Dordrecht, 1988.

PINKER S. (1994), *The Language Instinct* Penguin Press, London

SAKAKIBARA Y. (1992), Efficient learning of context-free grammars from positive structural examples, *Information and Computation*, 97(1) : pages 23–60.