Génération semi-automatique de grammaires de commande

Thierry Poibeau, Bénédicte Goujon

Génération semi-automatique de grammaires de commande

Thierry Poibeau[*] *et Bénédicte Goujon*[**]

[*] LIPN (UMR CNRS 7030) – Université Paris 13
99, Avenue J.B. Clément – 93430 Villetaneuse
Tél.: +33 (0)1 49 40 28 26 - Fax: +33 (0) 1 48 26 07 12
Mél: thierry.poibeau@lipn.univ-paris13.fr

[**] Thales Recherche et Technologie
Domaine de Corbeville – 91404 Orsay
Tél.: +33 (0)1 69 33 08 07 - Fax: +33 (0) 1 69 33 08 65
Mél: benedicte.goujon@thalesgroup.com

ABSTRACT

This paper presents an original strategy for the production of command grammars for complex systems. We show that command grammars describe a sub-language that can be processed by local syntactic rules and compositional semantic strategies.

1 INTRODUCTION

Les interfaces vocales offrent des moyens simples d'agir sur des systèmes complexes. Il a ainsi été démontré que les commandes données à la voix permettent dans certains contextes d'interagir plus rapidement avec le système (Carbonell et Dauchy, 1999). Pour être efficace, la commande vocale doit toutefois respecter un certain nombre de critères ergonomiques comme la bonne compréhension des énoncés, la gestion de la variation linguistique ou le temps d'analyse du message (Scapin, 1986). Afin de répondre à ces critères, les systèmes possèdent généralement des lexiques et des grammaires du domaine visé, permettant de contrôler ce qui est dit et de l'analyser de manière relativement fine.

Les grammaires de commande sont des modèles assez lourds à développer. Il faut en effet introduire dans le système l'ensemble de la terminologie et des structures admises. Il faut de plus, pour chaque énoncé, indiquer des variantes, gérer l'implicite et résoudre les énoncés incomplets. Enfin, le système doit être capable de dire quand un énoncé n'est pas reconnu avec suffisamment de plausibilité pour en tirer une information correcte. C'est pourquoi les grammaires sont le plus souvent écrites à la main, à partir d'un modèle du domaine ou d'un ensemble d'énoncés donnés en exemple.

Plusieurs limites importantes empêchent la portabilité des grammaires de commande : il s'agit souvent d'un ensemble statique de règles mêlant syntaxe et sémantique. La base de connaissances est souvent dupliquée à partir de l'application en raison de la complexité des formalismes manipulés, ce qui pose des problèmes de mise à jour et de cohérence. Le développement des ressources est le fait de spécialistes et les tentatives pour transférer l'écriture de grammaires de commande à des utilisateurs a le plus souvent échoué du fait de la complexité de la tâche.

Nous nous penchons sur cette question en essayant de proposer des éléments permettant une automatisation partielle du processus. Nous proposons de dériver automatiquement une partie des connaissances nécessaires à l'écriture de la grammaire de commande à partir des modèles fournis et des énoncés exemples. Nous défendons ici l'idée que les grammaires de commande décrivent un sous-langage qui peut être décrit au moyen d'un ensemble limité de règles appliquées incrémentalement aux énoncés visés. La sémantique de l'ensemble est calculé à partir de règles fondées sur la notion de compositionnalité. Le processus est donc générique et fonde l'originalité de la méthode.

Nous présentons en premier lieu des travaux d'inspiration similaire. Nous examinons ensuite en quoi les grammaires de commande peuvent être assimilées à des sous-langages et nous proposons une modélisation dynamique, à partir de la base de connaissances de l'application. Nous présentons ensuite plusieurs exemples et la mise en oeuvre opérationnelle du système.

2 ÉTAT DE L'ART

La génération automatique de grammaires de commande à partir de modèles formels a déjà été étudiée par différents auteurs. Mathieu *et al.* (2001) ont proposé un modèle général où une grammaire abstraite est instanciée par un lexique sémantique écrit de manière indépendante. Cette approche est intéressante dans la mesure où le lexique n'est pas mêlé à la syntaxe mais des expérimentations ont montré que les constructions syntaxiques abstraites enregistrées par défaut sont souvent trop contraintes pour une application donnée. Il faut donc à chaque fois modifier le modèle de la grammaire pour qu'il reste cohérent et qu'il puisse être adapté aux énoncés visés.

Notre problématique consiste à élaborer des grammaires guidées par le domaine et par l'application. Cette voie a été explorée par Brown et Burton vers 1976 à travers la notion de « grammaire sémantique » puis reprise à Stanford pour pourvoir le système expert MYCIN d'interfaces en anglais (Bonnet, 1980). Les limites de

cette approche résident dans le fait que l'élaboration de la grammaire et la description des connaissances lexicales constituaient des tâches trop coûteuses et peu réutilisables.

Thot *et al.* (2002) proposent un processus pour générer automatiquement des grammaires pour l'interrogation de bases de données. Une grammaire abstraite comporte des « trous » qui sont automatiquement instanciés par les éléments nommés de la base de données (noms des tables, noms de champs...). Cette application est intéressante car elle établit clairement un lien avec le modèle de l'application. Cette expérience est en revanche limitée aux bases de données.

3 NOTION DE SOUS-LANGAGE

La commande vocale présente des particularités importantes par rapport à la langue générale. Les énoncés sont essentiellement des ordres exprimés par des verbes à l'impératif ou à l'infinitif.

```
Zoomer
Afficher la liste des vols
```

Les compléments du verbe sont des groupes nominaux, composés ou non. Partant, un des problèmes majeurs de ce type d'énoncés consiste à choisir le bon découpage et à résoudre les ambiguïtés entre compléments du nom et compléments du verbe.

```
Afficher la liste des vols au départ de Paris
Afficher la liste des vols à l'écran
Afficher la liste des vols à l'arrivée
```

On voit sur ces exemples le problème du rattachement du complément introduit par la préposition *à*. Les séquences *à l'arrivée* et *au départ de Paris* constituent des compléments du groupe nominal *vols* tandis que *à l'écran* est complément du verbe (*afficher ~ à l'écran*).

Des informations sur la sous-catégorisation des verbes, sur la sémantique des propositions et sur le modèle de l'application permettent de traiter la plupart de ces ambiguïtés. Ainsi, seul les vols ont un lieu de départ et une période d'approche appelée arrivée. Un vol n'a en revanche pas d'écran (un avion peut comporter un jeu d'écrans, mais dans ce cadre visé on ne désigne pas l'objet à travers la notion de *vol* mais un moyen de transport d'un point à un autre).

On voit ainsi qu'une grammaire de commande décrit un sous-langage, c'est-à-dire à une partie limitée de la langue générale, offrant des régularités linguistiques et lexicales limitées. Cette description est cohérente avec l'approche proposée par Mathieu *et al.* (2001). Elle permet en outre, à la différence de cette étude, de prévoir une phase d'acquisition semi-automatique à partir d'un ensemble de phrases exemples fournies au système. Cette approche convient bien au domaine, dans la mesure où les utilisateurs doivent fournir un ensemble de structures types permettant à l'analyste d'élaborer la grammaire. La base de données de l'application sert en outre de dictionnaire sémantique pour les objets du domaine et les relations qu'ils entretiennent entre eux. Ainsi, dans une application de simulation tactique, l'ensemble des objets manipulés est géré au sein d'une base de données où sont enregistrées les informations concernant la nature de l'objet, son nom, son identifiant, sa position, etc.

4 MODELES FORMELS DE DONNEES

Les données manipulables au sein d'une application sont généralement stockées dans une base de données. Cette base fournit une image formalisée du domaine considéré.

Une base est constituée d'objets. Les objets sont pourvus d'attributs ayant des valeurs spécifiques pouvant le plus souvent être spécifiées *a priori* (l'interface permet de spécifier un ensemble de valeurs numériques, un intervalle de valeurs numériques, une valeur symbolique, etc.). La plupart des actions effectuées à travers l'interface d'une application visent à changer les valeurs d'un objet. Une action peut donc être définie comme un processus amenant un objet à changer de valeur.

Il existe de très nombreux modèles formels de représentation de données. La communauté *représentation des connaissances* a produit dans les années 1980 et 1990 des modèles qui n'étaient pas limités au monde informatique mais les méthodes MAD, KADS ou KOD ont aussi servi à modéliser des connaissances complexes concernant par exemple des filières nucléaires particulières, des processus de l'industrie chimique ou automobile (Pomian 1998).

Un des problèmes connus de ce type d'approche est de faire le lien entre le domaine formel (la base de connaissances formalisées) et le langage naturel (la façon dont un objet est effectivement désigné en langue). C'est pourquoi un pont est fait entre niveau conceptuel et niveau linguistique dans la base de connaissances même. Un module permet en effet l'interrogation de bases de connaissances (lexiques, ontologies du domaine, etc.) afin d'enrichir de manière semi-automatique le modèle. Une source privilégiée pour l'anglais est le réseau Wordnet ; pour le français, diverses sources existent afin de fournir des listes de synonymes ou de termes apparentés.

5 ANALYSE INCREMENTALE

L'analyse linguistique vise alors à construire de manière incrémentale des représentations formelles à partir des énoncés (ou fragments d'énoncés) analysés.

Règles de la grammaire

La grammaire est fondée sur des ensembles de règles de regroupement des mots en syntagmes. Lors des étapes 3 et 4, le système vise à élaborer une représentation sémantique partielle. Si cette étape échoue, le système essaie d'autres regroupements en fonction des règles de regroupement encore disponibles.

1. Les groupes syntagmatiques simples sont isolés, ainsi que les diverses propositions (relatives, complétives)

2. La sous-catégorisation de la tête (généralement le verbe) est prioritaire. En conséquence, les énoncés sont tout d'abord segmentés en fonction de la sous-catégorisation du verbe (plusieurs découpages peuvent être produits en cas d'ambiguïté).
3. Les compléments introduits par les prépositions *à* ou *de* sont ensuite arbitrairement rattachés au groupe situé immédiatement à gauche.
4. Les syntagmes introduits par d'autres prépositions sont analysés en fonction de la sémantique calculée de la séquence visée et de la préposition utilisée. La sémantique des prépositions concernées est enregistrée dans une table, à l'image des propositions de Jensen et Nillson (2003).

Faute de place, nous ne décrivons pas ce dernier point dans cet article. Nous décrivons essentiellement des exemples concernant des syntagmes introduits par des compléments en *à* ou en *de* qui posent le plus de problèmes d'ambiguïté. Le calcul de la valeur sémantique des séquences ainsi analysées est inféré en cherchant à mettre en correspondance les éléments nommés et la base de connaissances. Une séquence peut ainsi correspondre à un attribut et à sa valeur, à une description d'objet, à l'instanciation d'un élément de la structure argumentale d'un prédicat complexe, etc.

Le mécanisme d'analyse compositionnelle n'est pas monotone. Certains éléments font l'objet d'une analyse et d'une modélisation spécifique. Ainsi, les syntagmes pouvant être analysés comme des déterminants complexes font l'objet d'une modélisation logique. Par exemple la séquence *la liste des vols* revient à rechercher l'ensemble de tous les objets qui sont des vols. Il en va de même de l'ensemble des quantifieurs et des opérations ensemblistes. Les problèmes de portée sont limités dans la mesure où la quantification concerne quasi systématiquement l'élément syntagmatique suivant immédiatement le quantifieur.

Exemple d'analyse simple

Le verbe est tout d'abord isolé. Les compléments sont ensuite repérés puis, parmi ceux-ci, les compléments indirects sont typés d'après la préposition introductrice (analyse en syntagmes minimaux).

```
(V Afficher) (NP la liste) (PP des vols) (PP
au départ) (PP de Paris)
```

Si le verbe sous-catégorise un complément introduit par une préposition donnée et qu'un syntagme est introduit par cette préposition, le découpage est effectué conformément à la sous-catégorisation du verbe. Sinon, les éléments introduits par une préposition vide *à* ou *de* sont par défaut rattachés au groupe qui précède. Cela permet de regrouper aisément certains groupes et de diminuer d'autant l'ambiguïté.

```
(V Afficher) (NP (NP la liste) (PP (PP des
vols) (PP (PP au départ) (PP (PP de
Paris)))))
```

Le système essaie parallèlement d'apparier les têtes de chaque syntagme avec des éléments du lexique. Ainsi *Paris* est identifié comme un nom de ville, le mot *départ* correspond lui à un nom d'attribut de l'objet `vol` et la séquence *vol* permet d'identifier l'objet `vol` (d'après la base de connaissances) L'analyse est donc la suivante :

```
(PP (PP au départ) (PP de Paris))
départ = Paris

(PP (PP des vols) (PP (PP au départ) (PP (PP
de Paris)))
Vol : départ = Paris

(NP (NP la liste )(PP des vols) (PP (PP au
départ) (PP (PP de Paris))))
∀x tel que x = (Vol : départ = Paris)

(V Afficher (NP (NP la liste )(PP des vols)
(PP (PP au départ) (PP (PP de Paris)))))
Afficher ∀x tel que x = (Vol : départ =
Paris)
```

Les mots désignant des regroupements (*la liste* de, *l'ensemble* de) font l'objet d'une représentation pseudo-logique permettant d'interroger directement la base de connaissances.

Le lexique doit être connu du système. Pour des raisons de fiabilité vis-à-vis de l'application, il n'est pas possible de proposer une analyse si la tête d'un syntagme n'est pas connue (mais il peut y avoir une certaine tolérance si certains éléments du syntagme en dehors de la tête ne sont pas reconnus, grâce au paramétrage possible de l'application).

Sous-catégorisation complexe du verbe

Quand le verbe sous-catégorise plusieurs syntagmes, le découpage vise à repérer ceux-ci au plus vite. Soit l'exemple *Déplacer le char du point alpha au point beta*. Du point de vue de la grammaire, *déplacer* est un verbe demandant trois arguments : l'objet déplacé (*le char*), un point de départ (*point alpha*) et un point d'arrivée (*point beta*).

```
(V Déplacer (NP le char) (PP du point alpha)
(PP au point beta))
```

Plusieurs analyses sont gérées en parallèle en cas d'ambiguïté. Les erreurs de découpage sont résolues lors de l'analyse : si un découpage ne peut pas produire de structure logique valide, il est rejeté. Cette stratégie permet de retarder la prise de décision jusqu'au moment où le système dispose de l'ensemble de l'information nécessaire.

Cas des expressions discontinues

Une séquence comme *Afficher la liste des vols à l'écran* pose problème si le complément *à l'écran* n'a pas été inclus dans la base de connaissance. L'analyse suivante n'est pas possible :

```
(PP (PP de vols) (PP à l'écran))
vol = écran
```

car *vol* ne désigne pas un attribut de la base et *écran* n'est pas une valeur d'attribut possible. Pour que la grammaire de commande accepte la structure *Afficher la liste des vols à l'écran*, il faut au préalable avoir enregistré *Afficher ~ à l'écran* dans la base de connaissances. Lors du découpage en syntagmes minimaux, un prétraitement est effectué pour regrouper les mots composés et autres syntagmes complexes. On obtient alors le découpage suivant :

```
(V Afficher_à_l_écran) (NP la liste) (PP des
vols)
```

Finalement, comme *Afficher à l'écran* est une variante libre de *afficher* (simple synonyme ne modifiant pas le résultat de l'analyse), le complément est supprimé et on obtient la même analyse que pour *Afficher la liste des vols*.

Déduction du nom de l'attribut

On parle d'ellipses quand un élément de la base de connaissance est implicite dans le discours. Les grammaires développées acceptent un nombre limité d'ellipses afin de maintenir la fiabilité de l'analyse. Les cas d'ellipses pris en compte concernent essentiellement les noms d'attributs. Pour prendre un exemple simple, dans la séquence *les deux chars*, la notion de quantité peut être directement déduite du fait de la présence du déterminant *deux* dans le syntagme. Pour une séquence comme *les chars kaki*, la base de connaissance contient l'information comme quoi *kaki* est une couleur. Si le mot *kaki* est inconnu, l'analyse échoue car la valeur ne peut pas être reliée à un attribut.

Variantes syntaxiques

Un ensemble de variantes syntaxiques est enfin intégré au système afin de permettre l'analyse de séquence comme *Je veux déplacer le char du point alpha au point gamma*. La plupart des variantes libres peuvent être enregistrées et simplement supprimées de l'énoncé. On ramène alors celui-ci à une séquence reconnue par la grammaire.

6 MISE EN ŒUVRE

Le modèle proposé peut être entièrement dynamique. Les énoncés sont alors découpés, analysés en fonction des règles génériques d'analyse et la structure logique élaborée « à la volée » en fonction des éléments contenus dans la base de connaissances.

Cependant, en pratique, les analyseurs exigent une base de règles statiques. Il s'avère également utile de générer la grammaire correspondante pour pouvoir obtenir un ensemble de règles valides et modifiables. La génération de l'ensemble des règles d'analyse peut cependant poser les problèmes mentionnés dans l'introduction, si des modifications y sont apportées manuellement. En effet, ces corrections seront perdues si une nouvelle version de la grammaire est générée automatiquement (ce qui est nécessaire si le modèle de l'application évolue).

Soulignons enfin que la stratégie proposée permet d'obtenir des grammaires indépendantes de l'analyseur. Il suffit de réécrire le module de génération en respectant le format exigé par l'analyseur pour obtenir un résultat utilisable, la base de connaissances et les règles d'analyse ne changeant pas. La confrontation de la grammaire avec un ensemble d'énoncés typiques fournis par les utilisateurs finaux permet de s'assurer de la validité des représentations obtenues.

Il est enfin possible de vérifier que le résultat correspond bien à ce qu'aurait permis une grammaire écrite entièrement manuellement. Une comparaison approfondie reste à mener mais on peut d'ores et déjà souligner que l'automatisation du processus rend le résultat plus homogène et cohérent en termes de description linguistique.

7 CONCLUSION

Nous avons présenté un système d'analyse automatique de commandes vocales. Le système se fonde sur le modèle de l'application et sur un ensemble de règles locales. Les représentations logiques correspondantes sont construites conjointement à l'analyse, ce qui permet de résoudre dynamiquement les cas d'ambiguïté. Le processus d'analyse sémantique est essentiellement compositionnel, ce qui semble adapté au sous-langage décrit par les grammaires de commande.

BIBLIOGRAPHIE

A. Bonnet. « Les grammaires sémantiques, outil puissant pour interroger les bases de données en langage naturel ». In *R.A.I.R.O. Informatique.* vol. 14. n°2. 1980. pp. 137–148.

N. Carbonell et P. Dauchy. Empirical data on the use of speech and gestures in a multimodal human-computer environment. *Proceedings HCI International'99*. Munich. 1999. pp. 446-450.

Jensen et Nillson « Ontology-Based Semantics for Prepositions ». *Proceedings of the ACL-SIGSEM workshop on the syntax and semantics of prepositions and computational linguistics applications*, Toulouse. 2003.

F.A. Mathieu, S. Surcin et C. Sedogbo. « Un système de commande vocale multimodale : Thom-Speaker". *Technique et science informatiques*. Vol. 20. n°3. 2001, p.337-368.

J. Pomian. *Mémoire d'entreprise*. Sapientia. Gentilly. 1998.

D. Scapin. *Guide ergonomique de conception des interfaces humain-machine*. INRIA. 1986.

A.R. Toth, T.K. Harris, J. Sanders, S. Shriver, R. Rosenfeld. Towards every-citizen's speech interface : an application generator for speech interfaces to databases. *ICSLP 2002*. Denver. 2002. pp. 1497-1500.