TP4 : Analyse syntaxique

Cours d'initiation au TAL - L2 MIASHS

17 mars 2014

Pour chacun des exercices, sauvegardez le script Python dans un fichier qui a comme nom *exo < numéro de l'exercice >.py* puis mettre tous les fichiers dans une archive .zip qui a comme nom *TP4_<nom de l'étudiant>_<prénom de l'étudiant >.zip*. A la fin de la séance, envoyez l'archive en fichier attaché à l'adresse *perrier@loria.fr* en mettant comme objet du message *Initiation au TAL : TP4*.

A l'aide d'un navigateur, affichez le tutoriel NLTK en allant à l'URL : *nltk.org/book* Ouvrez le tutoriel au chapitre 7 « Extracting Information from Text ». Lisez attentivement les passages qui vous seront recommandés en exécutant les exemples dans une fenêtre *Idle* ouverte en parallèle.

1 Arbre syntaxique

Allez directement à la section 7.4 « Recursion in Linguistic Structure »et au second paragraphe « Trees ». Ce paragraphe introduit la notion d'arbre. Le constructeur *nltk.Tree (etiquette, liste_fils)* permet de construire un arbre. Son premier argument *etiquette* est une chaîne de caractères qui étiquettera la racine de l'arbre. Le second argument est une liste des sous-arbres fils de l'arbre construit. Certains éléments de la liste peuvent être des chaînes de caractères. Ces chaînes représentent alors les étiquettes de feuilles de l'arbre.

Exercice 1.1 *Ecrivez un programme Python qui crée et affiche tous les différents arbres syntaxiques des phrases ci-dessous (toutes sont ambigües donc elles ont au moins deux arbres syntaxiques).*

Le joueur de football américain connaît Marie.
Marie arrive et repart avec Jean.
Les filles de Marie et Jean arrivent.

Vous utiliserez les catégories grammaticales suivantes :

symbole	*nom complet*
P	*phrase*
COMPL	*proposition complétive*
GN	*groupe nominal*
GV	*groupe verbal*
GP	*groupe prépositionnel*
ADJ	*adjectif*
CC	*conjonction de coordination*
CS	*conjonction de subordination*
DET	*déterminant*
NC	*nom commun*
NP	*nom propre*
PREP	*préposition*
VC	*verbe transitif avec complétive pour objet*
VI	*verbe intransitif*
VT	*verbe transitif avec groupe nominal pour objet*

Il faut aussi tenir compte du fait qu'une conjonction de coordination coordonne toujours deux syntagmes de même type X pour former un syntagme de type X.

Mettez le programme dans un fichier TP4exo1.1.py.

2 Grammaires algébriques et analyse syntaxique

Passez au chapitre 8 « Analyzing Sentence Structure »en allant directement à la section 8.3 « Context Free Grammar » qui présente le formalisme des grammaires algébriques, tel qu'il est appliqué à la modélisation de la syntaxe des langues.

Le paragraphe « A simple Grammar » illustre ce qu'est une grammaire algébrique à l'aide d'un exemple linguistique et une démonstration d'analyse descendante. L'exemple a été choisi pour montrer la notion d'ambiguïté syntaxique.

Après l'étude de ce paragraphe, allez à la section 8.4 « Parsing With Context Free Grammar » qui aborde l'analyse proprement dite à l'aide d'une grammaire algébrique. Différentes stratégies sont proposées. La première exposée dans le paragraphe « Recursive Descent Parsing » est la méthode descendante récursive. Elle consiste à construire l'arbre syntaxique à partir de la racine S, qui représente la phrase globale, en descendant jusqu'aux feuilles qui représentent les mots de la phrase. Une démonstration qui peut être lancée avec l'instruction *nltk.app.rdparser()* vous permet de mieux la comprendre. Vous pouvez ignorer le paragraphe.

La seconde méthode exposée dans le paragraphe « Shift-Reduce Parsing » est une méthode ascendante qui utilise une pile. Au fur et à mesure de la lecture des mots de la phrase, ceux-ci sont empilés (*shift*). Dès que sur la pile, apparaît une suite de symboles représentant la partie droite d'une règle, cette suite est remplacée par la partie gauche de la règle (*reduce*). A la fin, il ne doit rester que le symbole de départ sur la pile. Etudiez avec soin la démonstration réalisée par l'instruction *nltk.app.srparser()*.

Exercice 2.1 *Voici une suite de phrases grammaticalement correctes :*

Le beau bébé dort dans le berceau.
Jean voit le bébé dans le berceau dans la chambre.
Jean pense que dans le berceau le bébé dort.

1. *Ecrivez un programme Python qui permette d'analyser (en mode trace) ces phrases avec une grammaire algébrique que vous concevrez, en utilisant la stratégie* shift-reduce. *Pour écrire cette grammaire, vous pourrez utiliser les non terminaux de l'exercice précédent mais vous avez la liberté de modifier et d'enrichir ces non terminaux. Vous mettrez le programme dans un fichier* TP4exo2.1.py.
2. *Exécutez le programme précédent et enregistrez la sortie du programme dans un fichier* TP4exo2.1.txt. *Si vous avez rencontré des problèmes dans l'analyse de certaines phrases, cherchez la cause de ces problèmes et notez vos conclusions dans le même fichier.*