Travaux Pratiques –n°7

Modèles de langage à base de règles avec NLTK

Objectif du TP (sous Fedora) :

Comprendre comment écrire des grammaires et les utiliser pour analyser une suite de mots et en extraire une information.

Nous utiliserons la boite à outils python NLTK (Natural Language ToolKit http://www.nltk.org/) a priori déjà installée sous **`fedora.`** Il s'agit de la version 2, qui comporte certaines limites par rapport à la version 3 décrite dans un ouvrage en ligne http://www.nltk.org/book/

Récupérer les données et les scripts mis à disposition sous moodle.

CONSIGNE :
Pour chacune des étapes du TP, récupérer les traces des exécutions des commandes dans un fichier texte nommé comme suit : **`VOTRE_NOM_TP_NLTK.txt`** Répondre également de manière claire et lisible aux questions posées dans le sujet. **Ce fichier sera à déposer sur moodle en fin de séance.**

1) Se familiariser avec le format .cfg

L'extension **`cfg`** *correspond aux grammaires hors contexte.*

1.1) Editer la grammaire **`GR_COMMANDE_V0_aip.cfg`** et étudier son contenu. Trouver 3 exemples de phrases correspondant à cette grammaire.

1.2) Editer le script python **`script_question1_aip.py`**, étudier le contenu de ce script, puis l'exécuter.

2) Grammaire et analyseur

2.1) Editer la grammaire **`GR_COMMANDE_V1_aip.cfg`** et étudier son contenu.

2.2) Utiliser ensuite le script python **`script_question2_aip.py`** pour analyser une série de commandes en utilisant le **mode trace** pour étudier le comportement de l'analyseur. Quel est l'algorithme implémenté ? Quel type de résultats obtient-on et quelles sont les limites de cette grammaire ?

2.3) Modifier la grammaire pour prendre en compte des commandes comme :

```
pose la télécommande sur la table
porte le verre dans la cuisine
```


2.4) Modifier le script et le renommer pour inclure votre nom dans le nom du script (ex : **`script_question2_aip_DUPOND.py)`**. Analyser plusieurs commandes pour tester la grammaire modifiée.

3) Grammaires probablisées

L'extension **`pcfg`** *correspond aux grammaires probabilisées.*

3.1) Editer et étudier la grammaire **`GR_COMMANDE_PROB_aip.pcfg`**

3.2) Utiliser le script python **`script_question3_proba_aip.py`** pour analyser une série de commandes. Quel type de résultats obtient-on ?

4) Grammaires et structures de traits : contrôle et interprétation

L'extension **`fcfg`** *correspond aux grammaires incluant des structures de traits (spécification d'attributs et construction d'un résultat structuré).*

4.1) Editer et étudier la grammaire **`IHR1_COMMANDE_SDT_aip.fcfg`**

4.2) Utiliser le script python **`script_question4.py`** pour traiter plusieurs commandes et tester la validité des contrôles mis en place. Quel type de résultat obtient-on ?

4.3) Faire évoluer la grammaire pour prendre en compte les mêmes phrases que précédemment (question 2.3) et intégrer la partie interprétation associée. Il s'agit, par exemple pour la phrase **`'prends le verre bleu dans la cuisine'`** d'obtenir une interprétation du type :

```
ORDRE[INTERP=[ACTION='pick_up', OBJET =[ [TYPE_OBJET='glass'],
                                         [COLOR='blue']]],
                                LOC =  [ [TYPE_LOC = 'room'],
                                         [VAL_LOC = 'kitchen']]]
```

5) Grammaires et structures de traits : transformation en commande

5.1) Editer et étudier la grammaire **`IHR1_COMMANDE_EXEC10_aip.fcfg`**

5.2) Utiliser le script python **`script_question5_aip.py`** pour traiter plusieurs commandes et comparer les résultats obtenus avec ceux obtenus dans la question 4.2)

Attention : il n'y a pas de documentation pour la version 2 de nltk disponible sur fedora.

Annexe –extrait de la documentation NLTK version 3:
http://www.nltk.org/howto/parse.html
http://www.nltk.org/howto/grammar.html
http://www.nltk.org/_modules/nltk/parse/generate.html