

## Cours sur le traitement automatique des langues (II)

Violaine Prince  
Université de Montpellier 2  
LIRMM-CNRS

## L'analyse syntaxique

- Le but : fournir une structure interprétable d'un texte.
  - ◆ si le langage est artificiel, il faut en plus vérifier la correction
- L'outil principal de l'analyse syntaxique
  - ◆ la grammaire
    - ✦ hors-contexte (langage artificiel)
    - ✦ ou dépendante du contexte

## Grammaires hors contexte

- Rappel sur les grammaires
  - ◆  $G = \{V_n, V_t, S, P\}$ 
    - ✦  $V_n$  : vocabulaire auxiliaire
    - ✦  $V_t$  : vocabulaire terminal
    - ✦  $S \in V_n$  : axiome
    - ✦  $P$  ensemble fini de productions
      - $p \in P$  si
      - $p \in (V_n \cup V_t)^* x (V_n \cup V_t)$  et  $p = (w_1, w_2)$ .

- Le système défini par  $\{V_t^*, S, P\}$  est un système formel
- Le langage  $L$  engendré par  $G$  est l'ensemble des théorèmes du système formel
- En langage naturel :
  - ◆  $V_n$  : ensemble des **catégories grammaticales** (étiquettes) et des catégories syntaxiques.
    - ✦ Ex : groupe verbal est une catégorie syntaxique. verbe est une catégorie grammaticale.

- ◆  $V_t$  : ensemble des mots de la langue (dictionnaire)
- ◆  $P$  : ensemble des règles de grammaire
- ◆  $S$  : axiome unitaire (phrase)
- ◆ Structure syntaxique : arborescence mémorisant la démonstration permettant d'obtenir un mot (dérivation).

correspond à  $A \rightarrow w_1 w_2$

### Hors contexte et dépendant du contexte

- $G$  est dite hors contexte si, pour toute production du type
  - ◆  $A \rightarrow w$ ,  $A \in V_n$ , et  $w \in V_n$  ou  $V_t$ .
- $G$  est dite dépendante du contexte sinon.
- Exemples
  - + une règle du type  $GN \rightarrow \text{Nom préposition Nom}$  est hors contexte
  - + une règle du type  $GN \rightarrow \text{Nom « de » Nom}$  est dépendante du contexte

### Les grammaires normées de Chomsky

- Une grammaire normée de Chomsky est un ensemble de règles de réécriture (ou production) de la forme :
  - $S \rightarrow S_1 S_2$ 
    - ◆ avec  $S \in V_n$ ,  $S_1 \in V_n$ ,  $S_2 \in V_n$
  - Ou  $S \rightarrow A$ 
    - ◆ Avec  $S \in V_n$ ,  $A \in V_t$ .

### Les grammaires normées de Chomsky

- $S$ ,  $S_1$  et  $S_2$  sont des symboles non terminaux (dans l'ensemble  $V_n$  des étiquettes grammaticales et syntaxiques)
- $A$  est un symbole terminal (dans l'ensemble  $V_t$  des lexèmes)
- Exemple :
  - ◆  $PH \rightarrow GN GV$
  - ◆  $GV \rightarrow V GN \mid V GNPREP$
  - ◆  $GN \rightarrow DET SN$
  - ◆  $GNPREP \rightarrow PREP GN$
  - ◆ Sont des règles valides sur des symboles non terminaux

### Les grammaires normées de Chomsky

- Les règles suivantes permettent d'inclure des symboles terminaux :
  - DET → Le
  - SN → chat
  - SN → souris
  - SN → sourire
  - V → manger
  - V → bondir
  - V → sourire
  - PREP → sur
  - PREP → à

On supposera que l'analyse morphologique est capable de reconnaître les flexions et les conjugaisons.

### Les grammaires normées de Chomsky

- La grammaire ainsi définie, est capable de générer (ou de reconnaître) les phrases suivantes :
  - Le chat mange la souris.
  - Le chat bondit sur la souris.
  - Le chat sourit à la souris.

```

    graph TD
      PH --- GN1[GN]
      PH --- GV[GV]
      GN1 --- DET1[DET]
      GN1 --- SN1[SN]
      DET1 --- Le[Le]
      SN1 --- chat[chat]
      GV --- V[V]
      GV --- GN2[GN]
      V --- manger[manger]
      GN2 --- DET2[DET]
      GN2 --- SN2[SN]
      DET2 --- le[le]
      SN2 --- souris[souris]
    
```

### Génération et analyse

- Les grammaires de Chomsky sont des grammaires génératives (puissance en génération)
- L'analyse est vue comme un cas particulier d'appariement entre :
  - Une phrase fournie
  - Une phrase que l'on peut engendrer par la grammaire
- En raison de l'ambiguïté des étiquettes, la combinatoire en analyse est élevée, et l'élimination des combinaisons « fausses » est une tâche importante.

### Combinatoire en analyse

- Exemple :
  - Le chat mange la souris.

On cherchera à construire deux arbres

(arborescence précédente)

```

    graph TD
      subgraph Tree1 [Tree 1]
        DET1[DET] --- Le1[Le]
        SN1[SN] --- chat1[chat]
        V1[V] --- manger1[manger]
      end
      subgraph Tree2 [Tree 2]
        DET2[DET] --- le2[le]
        SN2[SN] --- souris2[souris]
      end
      subgraph Tree3 [Tree 3]
        GN3[GN] --- DET3[DET]
        GN3 --- SN3[SN]
        DET3 --- Le3[Le]
        SN3 --- chat3[chat]
        V3[V] --- manger3[manger]
        DET4[DET] --- le4[le]
        V4[V] --- souris4[souris]
      end
    
```

### Combinatoire en analyse

- On dit que l'analyse a réussi si et seulement si, à partir d'une phrase donnée, on construit par les règles de grammaire une arborescence de nœud PH (racine).
- Dans l'exemple précédent, avec la valeur V pour « souris » la phrase ne mène pas à une arborescence de nœud PH. Elle doit donc être éliminée.

### Difficultés et « backtracking »

- Plus la grammaire est importante, plus la combinatoire est élevée.
- Exemple :
  - Supposons que l'on rajoute la règle terminale :
    - GN → Le
  - Pour rendre compte des pronoms personnels compléments d'objet.

### Exemple de construction d'arborescences

Le chat mange la souris.

DET SN V DET SN  
GN GN V

6 arborescences seront construites avec comme base

DET SN V DET SN (c'est la bonne)  
 GN SN V DET SN |= GN SN V GN |= GN SN GV  
 DET SN V GN SN |= GN V GN SN |= GN GV SN |= PH SN  
 DET SN V GN V |= GN V GN V |= GN GV V |= PH V  
 GN SN V GN SN |= GN SN GV SN  
 GN SN V GN V |= GN SN GV V  
 GN SN V DET V

### Algorithme d'analyse

- Il existe un algorithme, nommé algorithme de Cock qui permet de savoir si une phrase est analysable par une grammaire normée de Chomsky, avec une complexité en  $O(n^3)$ . Fourni en TD.
- Une phrase peut ne pas être analysable par une grammaire de Chomsky parce que:
  - La grammaire est non couvrante
  - La phrase n'est pas grammaticale (mal formée).
  - Il y a des mots inconnus.

## Théorème d'équivalence

- Toute grammaire de type réécriture, hors contexte, peut être réécrite sous forme de grammaire normée de Chomsky.
- Exemple :
  - ◆  $S \rightarrow S1 S2 S3$
  - ◆  $S3 \rightarrow S5$
  - ◆  $S2 \rightarrow k$
- Peut se réécrire sous forme :
  - ◆  $S \rightarrow S1 S4$
  - ◆  $S4 \rightarrow S2 S3$
  - ◆  $S2 \rightarrow k$

## Intégrité de la grammaire

- Une grammaire est intègre en génération si tout symbole non terminal peut se réduire, par une série de réécritures, en un (ou plusieurs) symbole(s) terminal(ux).
- Pas de symboles « vides ».
  - ◆ Ex:  $GV \rightarrow V \text{GNPREP}$
  - ◆  $\text{GNPREP} \rightarrow \text{PREP GN}$
  - ◆ Il faut qu'il existe une règle terminale avec  $\text{PREP} \rightarrow \text{xxx}$

## Les grammaires normées de Chomsky

- Pour le langage naturel :
  - ◆ Analyse en constituants
- Des défauts :
  - ◆ Rechercher une grammaire couvrante, intègre.
  - ◆ Ne traite pas les phrases agrammaticales
  - ◆ Multiplication des règles de génération pour les phrases particulières (nominales, subordonnées, relatives, etc...).

## Les grammaires normées de Chomsky

- Pour une première approche des problèmes peut servir d'accroche « pédagogique » :
  - ◆ Implémentation de l'algorithme de Cock
  - ◆ Recherche de la couverture grammaticale d'un ensemble de phrases données (bien formées).
- Marche mal sur du tout venant, et sur des grandes masses de données.

## Exemple d'un analyseur syntaxique théorique : SYGMART

- Analyse morphosyntaxique
- Moteur de réécriture sur des objets cibles : structures syntaxiques
- Extension des algorithmes de Markov
- Puissance de la machine de Turing

## Eléments de base

- Arborescences
  - ◆ Structures neutres.
  - ◆ Représentation :
    - ✦ chaîne de caractères sur le vocabulaire  $V = \{(\,,)\}$
- Propriétés
  - ◆ transformation d'arborescence : transformation de chaîne
  - ◆ Élément structuré : ensemble d'arborescences.
  - ◆ Identification d'arborescence : numéro (dimension)

### ■ Etiquettes

- ◆ Définies par une valeur
- ◆ Variables typées : arithmétique (entière), flottant, chaîne, à valeur définie exclusive et non exclusive, potentielle (valeur définie mais non explicitée), référence (pointeur sur une autre étiquette).

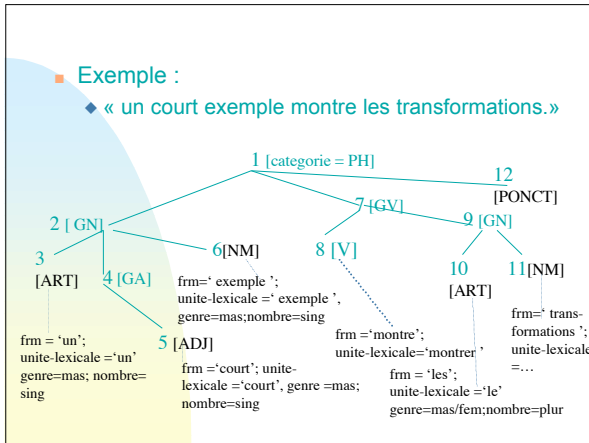
### ■ Fonction d'étiquetage :

- ◆ application d'un noeud d'une arborescence à une étiquette associée.
  - ✦ étiquetage morphologique : catégories grammaticales, nombre, genre

### ■ VARIABLES UTILISEES

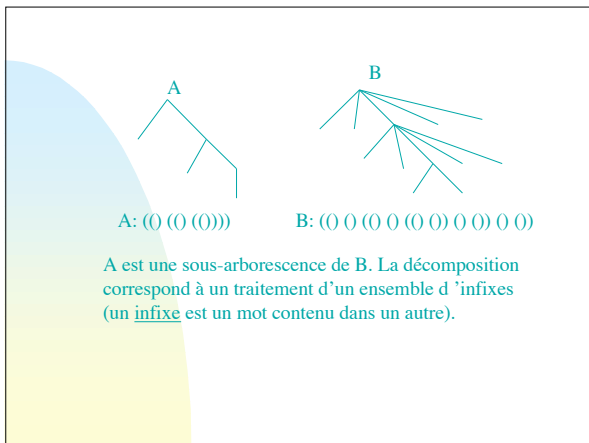
#### ■ DEFINIT Analyse

- ◆ DECLARE variables\_lexicales
  - ✦ CHAINE : fm.
  - ✦ POT: unite-lexicale.
- ◆ FIN variables\_lexicales.
- ◆ DECLARE variables\_grammaticales
  - ✦ EXC : categorie (PH,GN,GV,ART,ADJ,NM,PONCT)
  - ✦ NEX:nombre(sing,plur); genre(mas,fem).
- ◆ FIN variables\_grammaticales
- FIN analyse.



### Transformations d'éléments structurés

- Schéma
  - ◆ ensemble d'éléments structurés recherché dans l'élément traité. Chaque élément a globalement la même structure, les variations portent sur l'étiquetage, la présence (ou non) de certaines parties, l'ordre entre les éléments.
- schéma d'arborescence
  - ◆ soient deux arborescences A et B. A est une sous-arborescence de B s'il existe une projection de A dans B.



### Contraintes sur la décomposition

- Dépendance : immédiate ou généralisée
  - ◆ dans la reconnaissance d'une sous-arborescence, chaque point différent de la racine, dépend directement d'un autre point.

■ **Contrainte de continuité**

S'il y a une contrainte de continuité entre X et Y, l'arborescence de gauche ne peut pas être une sous-arborescence de l'arborescence de droite.

■ **Contrainte d'ordre**

■ **Contrainte de présence**

- ◆ un noeud (et ses descendants) peut être déclaré optionnel
- ◆ Le schéma A sera reconnu dans l'arborescence B si le noeud X peut être déclaré optionnel.

schéma A

schéma B

■ **Schéma d'élément structuré**

- ◆ Il est avant tout défini par des conditions sur les étiquettes associées aux noeuds par la fonction d'étiquetage.
- ◆ Conditions :
  - + expressions booléennes portant sur les valeurs des variables.
  - + deux types
    - conditions propres : l'expression booléenne ne peut faire référence qu'à une seule étiquette du schéma
    - conditions inter-sommets : elle peut faire référence à tout ou partie de l'ensemble des noeuds du schéma.

■ **Exemple de schéma**

■ (0(1(2)3(4)))

genre(2)&genre(4) != 0  
Le nom et l'adjectif doivent être du même genre



- Identification d'un schéma
  - ◆ lorsqu'un schéma est reconnu dans une arborescence, il identifie un ensemble de noeuds. Cette identification sépare les différents éléments de l'arborescence en listes.

- liste d'arborescence
  - ◆  $A \langle B, C \rangle$  : ensemble des éléments qui se trouvent sous le point associé à A, à droite du point associé à B et à gauche du point associé à C. A est obligatoire, B et C sont facultatifs.

### Réalisation des transformations

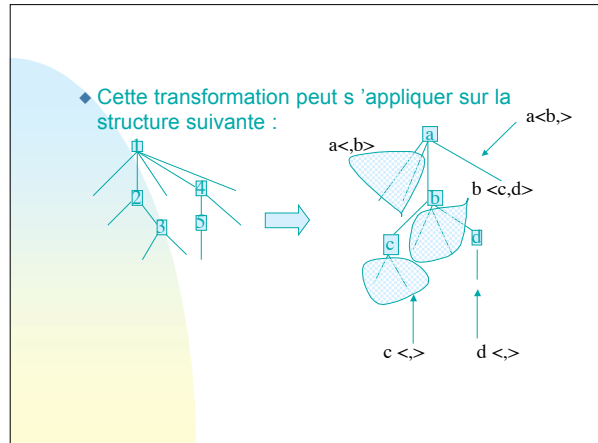
- Transformation d'arborescence
  - ◆ on remplace la sous-arborescence, reconnue par le schéma de reconnaissance, par l'arborescence de transformation.
- Transformation d'élément structuré.
  - ◆ Une transformation est définie par un quadruplet  $(A, B, f, O)$ 
    - + A est un schéma de reconnaissance
    - + B est un schéma de transformation
    - + f est une fonction de l'ensemble des listes de A dans l'ensemble des listes de B.
    - + O est une relation d'ordre partielle telle que :
      - x et y deux listes de A telles que  $f(x)=f(y)$ , alors  $O(x,y)$  est défini.

### Exemple

- On définit la transformation suivante :
  - ◆ schéma de reconnaissance

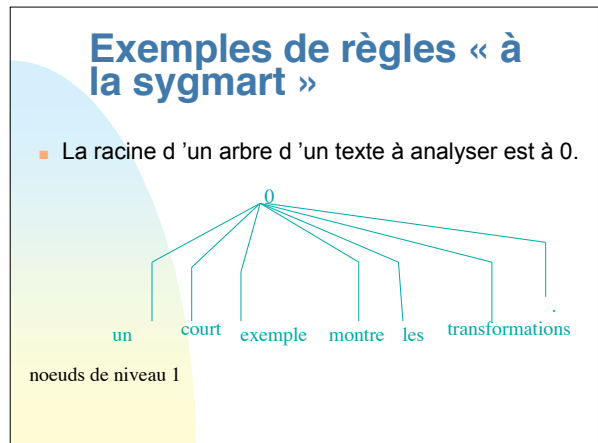
- ◆ Schéma de transformation

- Des applications permettant de définir f et O:
  - ◆  $1<,2> \rightarrow a<,b>$
  - ◆  $1<2,3> \rightarrow a<,b>$
  - ◆  $1<3,> \rightarrow a<b,>$
  - ◆  $2<,> \rightarrow b<c,d>$
  - ◆  $3<,> \rightarrow b <c,d>$
  - ◆  $4 <,> \rightarrow c <,>$
  - ◆  $5 <,> \rightarrow d <,>$
  - ◆ soit :
  - ◆  $(1(2(3) 4(5))) \rightarrow (a[* 1<,2> *, * 1<2,3>*]( b$   
 $([*1<3,>*]c [*2<,>*, *3<,>*, *4 <,>* ] d [2<,>*,$   
 $*3<,>*, *5 <,>*])))$



## Grammaire

- Algorithme de Markov étendu aux éléments structurés.
  - ◆ un ensemble ordonné de règles de transformation.
  - ◆ modes de fonctionnement :
    - ✦ itératif (mode des algorithmes de Markov)
    - ✦ exhaustif (lorsqu'une règle est appliquée elle est éliminée de la grammaire)
    - ✦ unitaire (mode itératif borné par une valeur numérique)



■ **Éléments de la règle (transformation)**

◆ **forme de la structure à transformer :**

- ✦ noeud1 (noeud2)
- ✦ noeud1, noeud 2

• les noeuds doivent être contigus (contrainte de continuité)

- ✦ noeud1, \*, noeud2

◆ **conditions sur la structure à transformer :**  
condition noeud1;condition noeud2;...;condition noeud n

◆ **schéma de transformation:**

- ✦ insertion de noeud,
- ✦ transformation de hiérarchie.

■ **insertion :**

◆ noeud-père: noeud-père (propriétés noeud-fils)

■ **modification de hiérarchie**

- R1 : 0 (1), 0 : categorie = 0, 1 : categorie = ADJ => 0:0: categorie = GA (insertion d'un noeud sous 0)
- R2 : 0 (1), /0 : categorie = 0, 1 : categorie = NM=>0:0: categorie = GN
- R3 : 0 (1), 0 : categorie = 0, 1 : categorie = V=> 0:0 : categorie = GV

