

Linguistique computationnelle

Séance 2 - Mardi 3 mai 2005

Philippe Blache

LPL, CNRS-Univ. de Provence

pb@lpl.univ-aix.fr

Jean-Philippe Prost

Macquarie University, Sydney

et LPL, CNRS-Univ. de Provence

jpprost@ics.mq.edu.au

Plan général

- Introduction
- Les grammaires syntagmatiques
- les grammaires logiques
- Grammaires syntagmatiques de haut niveau
- Les contraintes

Plan général

- Introduction
- **Les grammaires syntagmatiques**
- les grammaires logiques
- Grammaires syntagmatiques de haut niveau
- Les contraintes

plan cours 2

- Représenter l'information syntaxique
- L'analyse syntaxique
- Les grammaires context-free
- Stratégies d'analyse

L'analyse syntaxique

- **La grammaire :**

description des structures d'un domaine de connaissance

- **La syntaxe :**

description de la façon dont les mots sont arrangés ensembles.

Grammaire générative

Grammaire permettant la génération de *toutes et seulement* les chaînes acceptables d'une langue naturelle.

Constituance

- **Constituant** : groupe de mots se comportant comme une unité
- Ex. : Le Syntagme Nominal (ou Groupe Nominal)
 - Le livre rouge
 - Harry le cheval
 - Ils
 - les raisons pour lesquelles il est parti

Grammaires context-free (CFG)

- Grammaire syntagmatique
Phrase-Structure Grammar
- Une CFG est décrite par les 4 composants suivants :
 - Un ensemble de *symboles terminaux* (i.e. les mots)
 - Un ensemble de *symboles non-terminaux* (SN, SV, P, ...)
 - Un ensemble de *règles de production*
 - Un symbole non-terminal de départ (S, par convention)

Règles de production

1. SN \rightarrow Dét Nominal
2. SN \rightarrow NomPropre
3. Nominal \rightarrow Nom | Nom Nominal

Règles de production

1. Dét -> le

2. Dét -> un

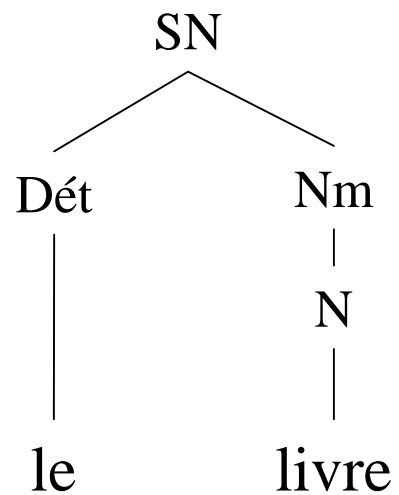
3. Nom -> livre

Dérivation

1. SN -> Dét Nominal
 2. SN -> NomPropre
 3. Nominal -> Nom | Nom Nominal
 4. Dét -> le
 5. Dét -> un
 6. Nom -> livre
- La chaîne *le livre* **dérive** du non-terminal SN
 - dérivation : séquence de règles d'expansion, représentée par un **arbre syntaxique**

Arbre syntaxique

Parse tree



Grammaire Context-Free

- **Double rôle :**
 - **affecter une structure à la phrase**
 - **générer les phrases du langage**

Grammaticalité

- **phrase générée / reconnue par la grammaire**
- **notion binaire**
- **caractérise tous les langages formels**
- **très réducteur pour ce qui est de la modélisation du langage naturel**

Compléments du nom et clause relative

- **Les vols [pour Sydney]**
- **Les vols [pour Sydney] [avant sept heures [du matin]]**
- **Les vols [qui arrivent avant midi]**
- **Celui [dont je t'ai parlé hier]**

Compléments du nom et clause relative

- 1. Nominal -> Nominal SP (SP) (SP)**
- 2. Nominal -> Nominal ClauseRel**
- 3. ClauseRel -> (PronomRel) SV**
- 4. PronomRel -> qui | que | quoi | dont | où**

Coordination

- **Rappelez-moi** [_{SN} [_{SN} les vols] et [_{SN} les coûts]]
- **Vous passez par** [_{SN} [_{SN} Singapour] ou [_{SN} Bangkok]]
- [_S [_S Je pense] donc [_S je suis]]
- **Il** [_{SV} [_{SV} ne marche pas] mais [_{SV} court]]

Coordination

1. **SN -> SN Coord SN**
2. **SV -> SV Coord SV**
3. **S -> S Coord S**
4. **Coord -> mais | ou | et | donc | or | ni |
car**

Accord

- **Do any flights stop in Chicago?**
- **Does this flight stop in Dallas?**

Accord

1. **S -> 3sgAux 3sgSN SV**
2. **S -> Non3sgAux Non3sgSN SV**
3. **3sgAux -> does | has | can | ...**
4. **Non3sgAux -> do | have | can | ...**

Sous-catégorisation

1. **SV -> Verb** **disappear**
2. **SV -> Verbe SN** **prefer a morning flight**
3. **SV -> Verbe SN SP** **leave Brisbane in the morning**
4. **SV -> Verbe SP** **leaving on Thursday**
5. **SV -> Verbe S** **I [_{SV} [_V think [_S I will take the next one]]**

Exercices

- **Décrire l'arbre syntaxique pour les phrases suivantes :**
 - *Does Qantas have a flight between five a.m. and six a.m.*
 - *I would like to flight on Air France.*
 - *Please repeat that.*
 - *Does Qantas 487 have a first class section?*
 - *I need to fly between Sydney and Perth.*
- **Décrire une grammaire générative pour le français**

plan cours 2

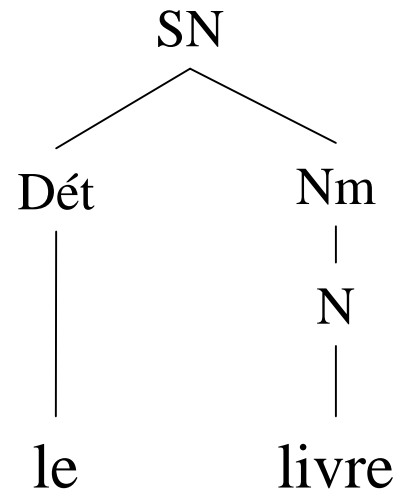
- Représenter l'information syntaxique
- L'analyse syntaxique
- Les grammaires context-free
- **Stratégies d'analyse**

Analyse syntaxique

Parsing

- Reconnaissance (grammaticalité de la phrase)
- Affectation d'une structure (arbre syntaxique)

Structure syntaxique



Stratégie d'analyse descendante

Top-down Parsing

- Construction de l'arbre à partir de la racine S
- Construction simultanée des différentes possibilités

Analyse descendante

- **Développement de tous les sous-arbres de racine S**
- **Pour chaque constituant non-terminal, développer tous les sous-arbres possibles**
- **Réitérer jusqu'à atteindre les catégories lexicales en bas de l'arbre**
- **Rejeter les arbres pour lesquels les feuilles ne correspondent pas aux mots de la phrase à analyser**

Une mini-grammaire pour l'anglais

1. S -> SN SV
2. S -> Aux SN SV
3. S -> SV
4. SN -> Dét Nominal
5. Nominal -> N
6. Nominal -> N Nominal
7. SN -> Npropre
8. SV -> V
9. SV -> V SN
10. Nominal -> Nominal | SP
11. SP -> Prép SN
12. Dét -> that | this | a
13. N -> book | flight | meal | money
14. V -> book | include | prefer
15. Aux -> does
16. Prép -> from | to | on
17. Npropre -> Sydney | Qantas

Stratégie d'analyse montante

Bottom-Up Parsing

- Commencer par les mots de la phrase à analyser
- Développer les arbres possibles par application successive des règles de grammaire
- Un arbre syntaxique est obtenu lorsqu'on a construit un arbre de racine S

Une mini-grammaire pour l'anglais

1. S -> SN SV
2. S -> Aux SN SV
3. S -> SV
4. SN -> Dét Nominal
5. Nominal -> N
6. Nominal -> N Nominal
7. SN -> Npropre
8. SV -> V
9. SV -> V SN
10. Nominal -> Nominal | SP
11. SP -> Prép SN
12. Dét -> that | this | a
13. N -> book | flight | meal | money
14. V -> book | include | prefer
15. Aux -> does
16. Prép -> from | to | on
17. Npropre -> Sydney | Qantas

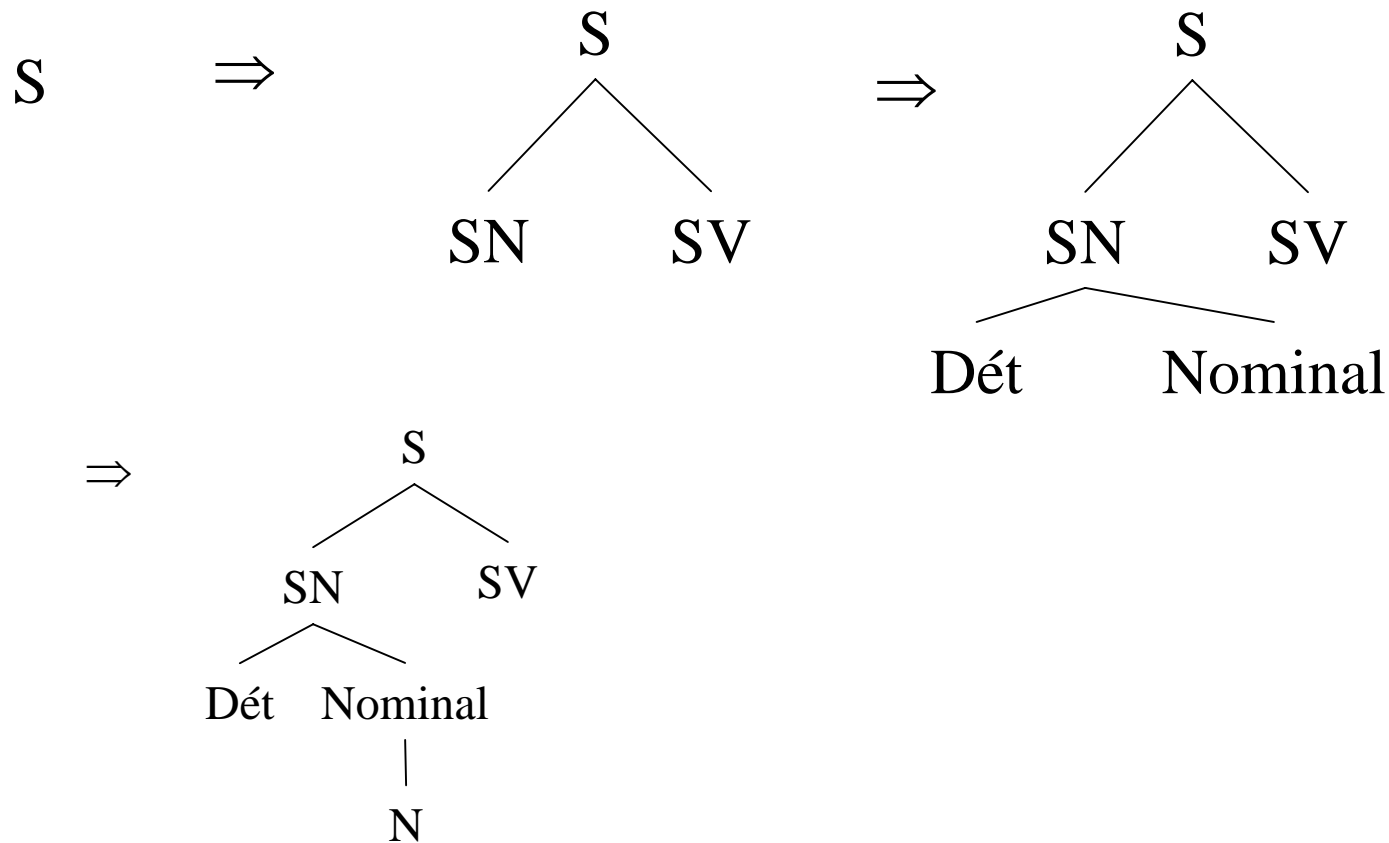
Ascendant vs. descendant

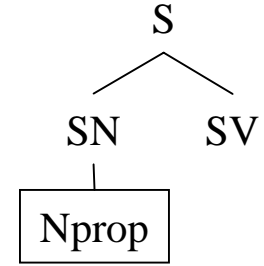
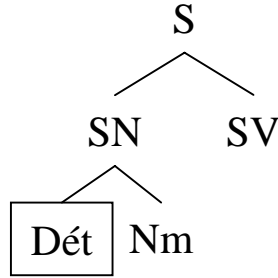
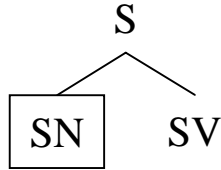
- La stratégie descendante n'explore que des arbres de racine S , i.e. aucun sous-arbre ne menant pas à S n'est exploré (contrairement à la stratégie ascendante)
- La stratégie descendante génère inutilement une multitude de sous-arbres non pertinents, i.e. sans rapport avec la phrase à analyser.

Analyseur descendant avec filtrage ascendant

- Préférer une recherche en profondeur (depth-first strategy) à une exploration systématique de tous les sous-arbres en parallèle à chaque niveau :
 - chaque sous-arbre est développé en profondeur jusqu'à obtenir soit un arbre complet, soit un échec
 - en cas d'échec, la recherche reprend au sous-arbre le plus récemment généré mais non encore exploré

Exemple



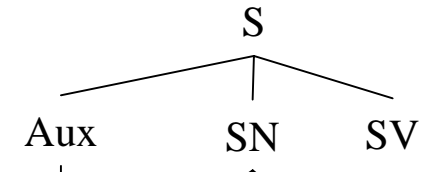
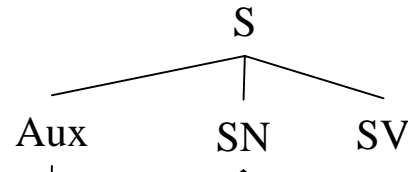
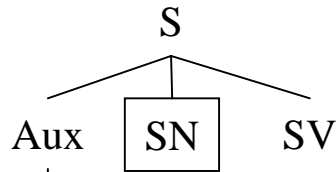
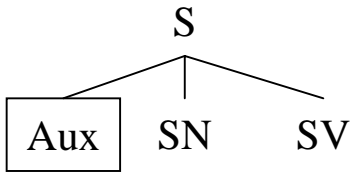


[Does]

[Does]

[Does]

[Does]



[Does]

Does [this]

Does [this]

Does this

```

function TOP-DOWN-PARSE (input, grammar) returns a parse tree
  agenda <- (initial S tree, beginning of input)
  current-search-state <- POP(agenda)
  loop
    if SUCCESSFUL-PARSE?(current-search-state) then
      return TREE(current-search-state)
    else
      if CAT(NODE-TO-EXPAND(current-search-state)) is a POS then
        if CAT(node-to-expand)
           $\subset$ 
            POS(CURRENT-INPUT(current-search-state)) then
              PUSH(APPLY-LEXICAL-RULE(current-search-state), agenda)
            else
              return reject
          else
            PUSH(APPLY-RULES(current-search-state, grammar), agenda)
      if agenda is empty then
        return reject
      else
        current-search-state <- NEXT(agenda)
  end

```

Analyseur descendant avec filtrage ascendant

- Quelle feuille développer en premier ?
- Quelle règle de grammaire appliquer en premier ?

Coin gauche

- L'analyseur développe chaque nouveau non-terminal en commençant par la branche gauche
- En conséquence, l'analyseur ne doit pas considérer une règle de la grammaire si l'entrée courante (mot) ne peut pas servir de coin gauche d'une dérivation.

Filtrage ascendant

Does this flight include a meal?

1. $S \rightarrow SN SV$
2. $S \rightarrow Aux SN SV$
3. $S \rightarrow SV$

*Does ne peut servir de coin gauche que
pour (2).*

Algorithme de Earley

- Programmation dynamique : stockage des solutions locales intermédiaires dans un tableau (chart parsing).
- Règle le problème de la récursivité à gauche
- Stratégie descendante
- au pire $O(N^3)$

Robustesse

- Comment augmenter un parseur de sorte qu'il puisse traiter des entrées incorrectes, qui incluent par exemple des erreurs d'orthographe, ou des mots non-reconnus ?