Analyse syntaxique automatique

2ème semestre 2000–2001

Laura Kallmeyer

laura.kallmeyer@linguist.jussieu.fr

Page web du cours : http://talana.linguist.jussieu.fr/~lkallmey/LI035.html

Bibliographie : voir page web.

1 Introduction

1.1 La notion d'un parser

Parsing = Analyse syntaxique automatique

Parser =

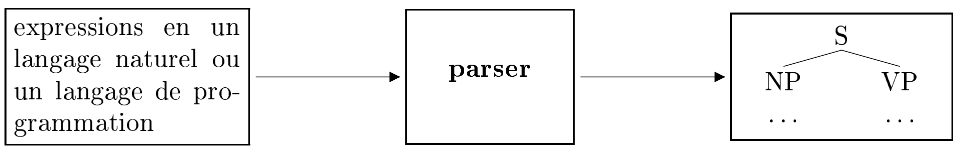
1. parfois programme (implémentation) qui teste pour une expression et une langue donnée si l'expression fait partie de la langue, et si oui, le parser donne une déscription de la structure de l'expression
2. parfois l'algorithme sur lequel un tel programme est basé.

En plus, dépendant de l'analyse donnée par un parser, on peut distinguer entre

- parser syntaxique
- parser sémantique
- parser phonologique
- parser morphologique
- parser de texte (discours)

Nous allons considérer des algorithmes (pas des implémentations concrètes) en traitant l'analyse syntaxique des langages naturels.

Schema d'un parser:



Des parsers sont utilisés

- dans des compilateurs: un compilateur fait entre autres une analyse syntaxique
- en psychologie cognitive: le parser doit modeler la façon de laquelle l'homme analyse des expressions de langage naturel
- en linguistique machinelle: analyse syntaxique automatique d'une expression de langage naturel, n'exige pas nécessairement une plausibilité cognitive

  Aspects importants: compléxité et efficacité

Quelques problèmes qui se posent avec langages naturels:

- syntaxe plus complexe que dans le cas des langages de programmations → plus grande variation de structures
- beaucoup d'ambiguités → parsing non-déterministique

1.2 Grammaires pour l'analyse automatique des langages naturels

Pour permettre une analyse automatique, il faut représenter les connaissances linguistiques nécessaires sous forme d'une grammaire formelle.

Définition 1 (Grammaire hors contexte, CFG)

Une grammaire hors contexte (CFG) $G = \langle V, T, S, P \rangle$ consiste en

1. *un alphabète V de symboles non-terminaux (variables),*
2. *un alphabète T de symboles terminaux*
3. *un symbole initiale $S \in V$.*
4. *un ensemble P de productions dont chaque élément est de la forme $A \to \alpha$ avec $A \in V$, $\alpha \in (V \cup T)^*$.*

Définition 2 (Dérivation dans une grammaire hors contexte)

$G = \langle V, T, S, P \rangle$ soit une grammaire hors contexte, $\gamma, \gamma' \in (V \cup T)^*$.

1. *$\gamma \Rightarrow \gamma'$ ssi il existe une production $A \to \alpha \in P$ et il existe des $\alpha_1, \alpha_2 \in (V \cup T)^*$ tel que $\gamma = \alpha_1 A \alpha_2$ et $\gamma' = \alpha_1 \alpha \alpha_2$.*
2. *$\gamma \stackrel{n}{\Rightarrow} \gamma'$ pour $n \geq 1$ ssi il existe $\gamma_1, \ldots, \gamma_{n-1} \in (V \cup T)^*$ tel que $\gamma \Rightarrow \gamma_1 \Rightarrow \cdots \Rightarrow \gamma_{n-1} \Rightarrow \gamma'$.*
3. *$\gamma \stackrel{*}{\Rightarrow} \gamma'$ ssi $\gamma = \gamma'$ ou il existe un n tel que $\gamma \stackrel{n}{\Rightarrow} \gamma'$.*

Définition 3 (Langage d'une grammaire hors contexte)

$G = \langle V, T, S, P \rangle$ soit une grammaire hors contexte.

1. *un $w \in T^*$ s'appelle une phrase.*
2. *$L(G) = \{w \in T^* \mid S \stackrel{*}{\Rightarrow} w\}$.*

Avantages des grammaires hors contexte pour langages naturels:

1. elles sont suffisamment puissantes pour décrire la plupart des phénomènes linguistiques.
2. elles sont facile à développer et à modifier
3. la question si une phrase w fait partie du langage $L(G)$ généré par une CFG G est décidable
4. elles permettent de developper des parsers efficaces.

Pour ces raisons, beaucoup de formalismes utilisés pour langages naturels sont basés sur des grammaires hors contexte (souvent enrichi de traits etc.).

Le cours va traiter des algorithmes de parsing pour des grammaires hors contexte.

1.3 Classification des algorithmes de parsing

Les algorithmes de parsing se distinguent à l'ègard de trois critères:

1. La direction dans laquelle la phrase est inspectée (de gauche à droite, de droite à gauche ou bidirectionnel).
2. La direction de l'analyse: Quels sont les objets de départ/les objets cible de l'analyse? Unidirectionnel: top-down (chercher des dérivations) ou bottom-up (chercher des réductions), approche mixte.
3. Qu'est-ce qu'on fait en cas de plusieures analyses possibles? Parsing déterministique ou non-déterministique.

1.4 Compléxité et efficacité

La question de compléxité se pose pour des algorithmes tandis que l'efficacité n'est considéré que pour des implémentations concrètes.

Compléxité: ici on caractérise le besoin de temps/de mémoire d'un algorithme dans le cas le plus mauvais. Cette caractérisation tient compte de la taille de la grammaire et elle se fait en fonction de la longueur n de la phrase à analyser.

1.5 La spécification des algorithmes

Nous adoptons la forme suivante pour les déscriptions d'algorithmes ou de procédures:

Une telle déscription consiste de

- **une tête** : La tête donne le nom de l'algorithme/de la procédure plus d'éventuels attributs (p.e. non-déterministique).
- **une partie de spécification** : Cette partie spécifie des données globales utilisées, l'input (paramètres et leurs domaines), l'output et les variables locales (avec leurs domaines).
- **une partie opérationnelle** : Décrit la méthode. Structures de contrôle utilisées ici:
  Itérations:
  - **repeat:** $\langle$action$\rangle^+$
  - **repeat until** $\langle$condition$\rangle$ $\langle$action$\rangle^+$
  - **for** $\langle$variable$\rangle$ = $\langle$valeur$_1\rangle$, ..., $\langle$valeur$_k\rangle$: $\langle$action$\rangle^+$
  - **for all** $\langle$objet$\rangle$ **from** $\langle$ensemble$\rangle$: $\langle$action$\rangle^+$

  Actions avec conditions:
  - **if** $\langle$condition$\rangle$ **then** $\langle$action$\rangle^+$
  - **if** $\langle$condition$\rangle$ **then** $\langle$action$\rangle^+$ **else** $\langle$action$\rangle^+$

I Algorithmes élémentaires

Les algorithmes présentés dans les trois parties suivantes sont les algorithmes les plus simples. Ils se distinguent l'un de l'autre à l'égard de leur direction d'analyse: top-down, bottom-up et left-corner (une stratégie entre les deux).

Un trait caractéristique de ces trois algorithmes est que des résultats intermédiaire ne sont pas sauvegardés. Il peut alors arriver qu'une partie de la phrase est analysée plusieures fois de la même façon.

Exemple 1

G soit une grammaire contenant (entre autres) les règles suivantes:

$S \rightarrow NP\ Aux\ VP$
$S \rightarrow NP\ VP\ PP$
$S \rightarrow NP\ VP$

Avec un algorithme élémentaire on va peut-être analyser le NP sujet plusieures fois avant de trouver la bonne production pour réécrire S.

Tous les trois algorithmes sont d'une compléxité polynomielle.