Grammaires Hors-contexte
Grammaires : Séance 2

Marie-Laure Potet

Grenoble INP-Ensimag

2020-2021

Notes

Summary

Notes

Grammaire hors-contexte

Soit $G =< V_T, V_N, S, R >$ une grammaire. G est dite hors-contexte si et seulement si les règles sont de la forme :

- $A \rightarrow w$

avec $A \in V_N$ et $w \in (V_T \cup V_N)^*$.

Un langage L est dit hors-contexte si il existe une grammaire hors-contexte G telle que $L(G) = L$.

⇒ Une classe qui fournit un bon compromis Expressivité/décidabilité :

- Permet de décrire la syntaxe des langages de programmation
- Des algorithmes efficaces de reconnaissance (TL2)

Notes

Exemples

Exemple 1 : Une grammaire d'expressions arithmétiques :
$V_T = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, +, -, *, (,)\}$

$$exp \rightarrow num \mid exp + exp \mid exp - exp \mid exp * exp \mid (exp)$$
$$num \rightarrow 0 \mid \ldots \mid 9 \mid 0\ num \mid \ldots \mid 9\ num$$

⇒ On verra une " meilleure " grammaire plus tard.

Exemple 2 : les instructions :

statement	→	compound_statement | if_statement
		| assignment_statement
compound_statement	→	statement **;**
		| statement **;** compound_statement
assignment_statement	→	var **=** exp **;**
if_statement	→	**if (** exp **) then** statement suite **end ;**
suite	→	ϵ | **else** statement

Ici les terminaux sont en **gras**.

Notes

Summary

Notes

Rappels

Soit $G =< V_T, V_N, S, R >$ une grammaire et $V = V_T \cup V_N$:

- $xuy \Longrightarrow xvy$ ssi $u \to v \in R$ avec $x, y, u, v \in V^*$
- $\Longrightarrow^p$ les dérivations de longueur p (avec $\Longrightarrow^1 = \Longrightarrow$)
- $\Longrightarrow^*$ les dérivations de longueur quelconque

Soit $w \in V^*$. Le langage engendré par w est défini par :

$$L(w) = \{x \in V_T^* \mid w \Longrightarrow^* x\}$$

et :

$$L(G) = \{x \in V_T^* \mid S \Longrightarrow^* x\}$$

Notes

Composition des dérivations

✓ Propriété de composition des dérivations :

$$\frac{u_1 \Longrightarrow^{p_1} v_1 \qquad u_2 \Longrightarrow^{p_2} v_2}{u_1 u_2 \Longrightarrow^{p_1+p_2} v_1 v_2}$$

Notes

Théorème de décomposition des dérivations

Soit $G =< V_T, V_N, S, R >$ une grammaire hors-contexte et $V = V_T \cup V_N$ et

$$u_1 ... u_n \Longrightarrow^{p} w$$

avec $u_i \in V^*$ et $w \in V^*$ alors $\exists\ v_1\ ...\ \exists\ v_n$ avec $v_i \in V^*$ tels que :

1. $w = v_1 ... v_n$
2. $u_i \Longrightarrow^{p_i} v_i$
3. $\sum_{i=1}^{n} p_i = p$

Preuve par induction sur p (dans poly)

Notes

Preuve du théorème de décomposition

Cas $p = 0$ $u_1 ... u_n \Longrightarrow^0 w$.

On a :

1. $w = u_1 ... u_n$
2. $u_i \Longrightarrow^0 u_i$
3. $\sum_{i=1}^{n} 0 = 0$

Notes

Preuve du théorème de décomposition (suite)

Cas $p + 1$: $u_1 ... u_n \Longrightarrow^{p+1} w$.

Supposons que la première règle est appliquée à u_k ($u_k \rightarrow t$). On a donc :

$$u_1 ... u_k ... u_n \Longrightarrow u_1 ... u_{k-1} t u_{k+1} ... u_n \Longrightarrow^p w$$

Par hypothèse d'induction $\exists\, v_1 \ldots \exists\, v_n$ avec $v_i \in V^*$ tels que :

1. $w = v_1 ... v_n$
2. $u_i \Longrightarrow^{p_i} v_i$ pour $i \in 1..k-1 \cup k+1..n$
3. $t \Longrightarrow^{p_k} v_k$
4. $\sum_{i=1}^{n} p_i = p$

et donc $u_k \Longrightarrow^{p_k+1} v_k$

Notes

Implications du théorème

- L'ordre d'application des règles n'a pas d'importance. Soit les 2 règles $A \to u$ et $B \to v$ et la chaine $w_1 A w_2 B w_3$. On peut indifféremment faire les réécritures suivantes :

$$w_1 A w_2 B w_3 \Longrightarrow w_1 u w_2 B w_3 \Longrightarrow w_1 u w_2 v w_3$$
$$\text{ou} \quad w_1 A w_2 B w_3 \Longrightarrow w_1 A w_2 v w_3 \Longrightarrow w_1 u w_2 v w_3$$

- Utilisation d'une dérivation canonique (par exemple réécriture systématique du non-terminal le plus à gauche)
- Preuve par " morceau "

Non vrai pour les grammaires sous-contexte ou générales. Exemple : $aA \to b$ et $Ba \to a$ et BaA.

Notes

Summary

Notes

Dérivations canoniques

On s'intéresse généralement aux dérivations les plus à gauche :

- $xuy \Longrightarrow xvy$ ssi $u \rightarrow v \in R$ avec $x, y, u, v \in V^*$
- $xuy \Longrightarrow_{gauche} xvy$ ssi $u \rightarrow v \in R$ avec $x \in V_T^*$ et $y, u, v \in V^*$

Exemple :
Soit la grammaire suivante sur le vocabulaire $\{a, b\}$:

$$S \rightarrow SS \mid aSb \mid \epsilon$$

Deux dérivations équivalentes :

$$(1) S \Longrightarrow SS \Longrightarrow aSbS \Longrightarrow abS \Longrightarrow abaSb \Longrightarrow abab$$

$$(2) S \Longrightarrow SS \Longrightarrow SaSb \Longrightarrow Sab \Longrightarrow aSbab \Longrightarrow abab$$

Deux dérivations différentes :

$$(1) S \Longrightarrow SS \Longrightarrow aSbS \Longrightarrow abS \Longrightarrow abaSb \Longrightarrow abab$$

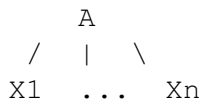
$$(2) S \Longrightarrow SS \Longrightarrow SSS \Longrightarrow aSbSS \Longrightarrow abSS \Longrightarrow abaSbS \Longrightarrow ababS \Longrightarrow abab$$

Notes

Arbre de dérivation

Autre représentation : les arbres de dérivation

- les noeuds sont des éléments de $V_T \cup V_N \cup \{\epsilon\}$
- La racine est étiquetée par l'axiome
- les feuilles sont des éléments de $V_T \cup \{\epsilon\}$
- l'application d'une règle A $\rightarrow$ X1...Xn avec Xi $\in V_T \cup V_N \cup \{\epsilon\}$ produit le sous-arbre :



Notes

Exemple

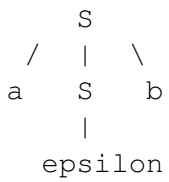
Soit la grammaire suivante sur le vocabulaire $\{a, b\}$:

$$S \rightarrow SS \mid aSb \mid \epsilon$$

On a plusieurs représentations canoniques pour la chaîne ab :

Dérivation 1 :

$$S \Longrightarrow aSb \Longrightarrow ab$$



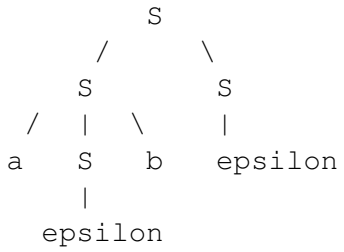
$\Rightarrow$ Mot associé à un arbre de dérivation = mot des feuilles.

Notes

Exemple (suite)

Dérivation 2 :

$$S \Longrightarrow SS \Longrightarrow aSbS \Longrightarrow abS \Longrightarrow ab$$



mot des feuilles : $a\epsilon b\epsilon = ab$

Notes

Summary

Notes

Grammaire ambiguë

Définition: (Grammaire ambiguë)

Une grammaire G est dite ambiguë ssi il existe au moins un mot de L(G) pour lequel il existe plusieurs dérivations canoniques.

$\Rightarrow$ La grammaire précédente est ambiguë. Autre mot possible ϵ.

Notes

Ambiguïté

Pourquoi on n'aime pas l'ambiguïté ?

Une grammaire d'expressions arithmétiques :

$V_T = \{0,1,2,3,4,5,6,7,8,9,+,-,*,(,)\}$

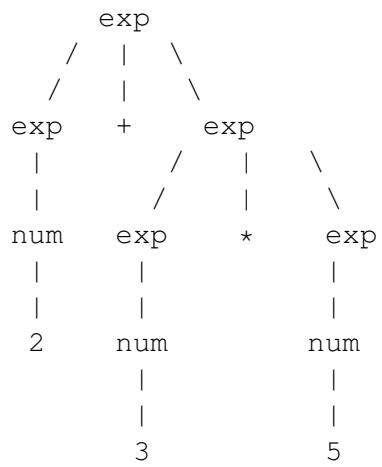
$$exp \rightarrow num \mid exp + exp \mid exp - exp \mid exp * exp \mid (exp)$$
$$num \rightarrow 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9$$

2 interprétations pour : $2 + 3 * 5$

$\Rightarrow$ On verra une " meilleure " grammaire plus tard.

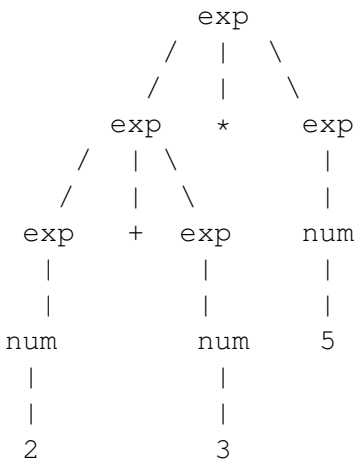
Notes

Arbre 1



Notes

Arbre 2



Notes

Langage ambigu

Définition: (Langage ambigu)

Un langage L est dit (intrinséquement) ambigu ssi toute grammaire G qui le décrit (i.e telle que $L(G) = L$) est ambiguë.

Il existe des langages intrinséquement ambigus : $a^n b^n c^i \cup a^n b^i c^i$ (intuitivement $a^n b^n c^n$ n'est pas un langage hors-contexte).

Là on se distingue des langages réguliers.

Notes

Exemple

Une grammaire ambiguë:
Soit la grammaire suivante G_1 sur le vocabulaire $\{a, b\}$:

$$S \rightarrow SS \mid aSb \mid \epsilon$$

On peut décrire le même langage de manière non ambiguë, par la grammaire G_2 suivante :

$$\begin{aligned} S &\rightarrow AS \mid \epsilon \\ A &\rightarrow aSb \end{aligned}$$

Mais bon, est-ce le même langage ? Comment on est sûr que cette grammaire n'est pas ambiguë ?

Notes

Condition suffisante pour qu'une grammaire ne soit pas ambiguë

Théorème:

Une condition suffisante pour qu'une grammaire G ne soit pas ambiguë est que les deux propositions ci-dessous soient vérifiées :

Condition 1 :

pour tout couple de règles $(A \rightarrow \alpha, A \rightarrow \beta)$ *de G tel que* $\alpha \neq \beta$, $L(\alpha) \cap L(\beta) = \emptyset$;

Condition 2 :

pour toute règle de la forme $A \rightarrow X_1 X_2 ... X_n$, *où* $X_i \in V_T \cup V_N$, $\forall w \in V_T^*$ *tel que* $X_1 X_2 ... X_n \Longrightarrow^* w$, $\exists!(w_1, w_2, ...w_n)$ *tel que* $w_i \in V_T^*$, $w = w_1 w_2 ... w_n$ *et* $\forall i$, $X_i \Longrightarrow^* w_i$.

Notes

Application à l'exemple : G1

La grammaire G_1 ($S \to SS \mid aSb \mid \epsilon$) ne vérifie pas les conditions :

- Condition 1 violée :
  $L(SS)$ contient $L(S)$ (règle $S \to \epsilon$) qui contient $L(aSb)$
- Condition 2 violée :
  la chaîne abababb peut être découpée de plusieurs manières différentes par SS : le premier S produit ab et le second abab ou l'inverse.

Notes

Application à l'exemple : G2

La grammaire G_2 ($S \to AS \mid \epsilon$, $A \to aSb$) vérifie les conditions 1 et 2 :

- Condition 1 OK : $L(\epsilon) = \{\epsilon\}$, $L(AS)$ des mots de longueur supérieure à 2
- Condition 2 OK :
  il faut montrer une seule possibilité d'appliquer la règle $S \to AS$.

  On va montrer qu'un mot de $L(A)$ n'a pas de préfixe propre dans $L(A)$. On aura alors $AS \Longrightarrow^* w$ décompose w en $u_1 u_2$ de manière unique avec $A \Longrightarrow^* u_1$ et $S \Longrightarrow^* u_2$.

Notes

Argumentation

$$S \to AS \mid \epsilon \qquad A \to aSb$$

On peut montrer les propriétés suivantes :

- tout mot de $L(S)$ et de $L(A)$ a autant de a que de b
- tout préfixe de $L(S)$ et de $L(A)$ a autant ou plus de a que de b.
  $v \in L(S) \cup L(A) \Rightarrow |v|_a \geq |v|_b$.

Donc tout préfixe propre de $L(A)$ s'écrit av avec v préfixe d'un mot de $L(S)$ et on a $|av|_a > |v|_b$. Donc $av \notin L(A)$.

Notes

Summary

Notes

Principe

- Soit une grammaire hors-contexte $G = (V_T, V_N, S, R)$
- Soit L la définition d'un langage de la forme $L = \{w \mid w \in V_T^* \wedge P(w)\}$, avec P un prédicat.

La preuve de $L(G) = L$ se fait en 2 temps :

1. la preuve de $L(G) \subseteq L$ est appelée **la correction** de la grammaire : tout mot produit par la grammaire vérifie la propriété attendue.
2. la preuve de $L \subseteq L(G)$ est appelée **la complétude** de la grammaire : tout mot vérifiant la propriété attendue peut être produit par la grammaire.

$\Rightarrow$ on va voir comment conduire ces preuves de manière systématique

Notes

Exemple

Soit $V_T = \{a, b\}$ et $L = \{a^i b^p \mid i \geq p\}$ et G la grammaire suivante :

(1) $S \rightarrow aS$ (2) $S \rightarrow aSb$ (3) $S \rightarrow \epsilon$

- Pour la correction on va montrer que tous les mots qui sont produits à partir de S sont de la forme $a^i b^p$: on va raisonner sur les dérivations.
- Pour la complétude on va montrer que tout mot de la forme $a^i b^p$ peut être produit à partir de S : on va raisonner sur les mots. Tout mot de L s'écrit $a^n a^p b^p$. On peut les construire de la manière suivante :
  1. A partir de S appliquer n fois la règle (1) : on obtient $S \Longrightarrow^* a^n S$
  2. puis appliquer p fois la règle (2) : on obtient $S \Longrightarrow a^n a^p S b^p$ S
  3. puis appliquer 1 fois la règle (3) : on obtient $S \Longrightarrow a^n a^p b^p$

Que se passe t-il si on remplace (1) par $S \rightarrow aaS$? Si on ajoute $S \rightarrow Sb$?

Notes

Preuve de correction

$L(G) \subseteq L$: le but est de montrer que tout mot w produit par la grammaire ($S \Longrightarrow^* w$), vérifie P (est tel que $P(w)$ est vrai).

- La preuve se fait par induction sur la longueur des dérivations : $\Longrightarrow^n$
- Demande généralement de caractériser les langages intermédiaires $L(A)$ associés à chaque non-terminal :

$$L_A = \{w \mid w \in V_T^* \wedge P_A(w)\}$$

On va donc prouver, pour tout $A \in V_N$ et tout $w \in V_T*$:

$$(A \Longrightarrow^n w) \Rightarrow P_A(w)$$

Avec l'hypothèse d'induction, pour tout $A \in V_N$:

$$\forall k < n \,.\, (A \Longrightarrow^k w) \Rightarrow P_A(w)$$

Notes

Preuve de Complétude

$L \subseteq L(G)$: le but est de montrer que tout mot vérifiant P (tout mot w tel que $P(w)$ est vrai) peut être produit par la grammaire ($S \Longrightarrow^* w$).

- La preuve se fait généralement par induction sur une mesure associée à w (sa longueur, le nombre d'occurrences d'un certain symbole...), l'ordre choisi dépendant du prédicat P.
- On sera généralement amené à montrer qu'on sait produire les éléments des langages intermédiaires $L(A)$, pour tout $A \in V_N$.

On va donc prouver, pour tout $A \in V_N$ et tout $w \in V_T*$:

$$(P_A(w) \wedge |w| = n) \Rightarrow (A \Longrightarrow^* w)$$

Avec l'hypothèse d'induction, pour tout $A \in V_N$:

$$\forall \alpha \,.\, (P_A(\alpha) \wedge |\alpha| < n) \Rightarrow (A \Longrightarrow^* \alpha)$$

Notes

Exemple Poly (1)

(1) $S \to aSb$
(2) $S \to bSa$
(3) $S \to SS$
(4) $S \to \epsilon$

On veut montrer:

$$L(G) = L$$
avec
$$L = \{w \in \{a,b\}^* : |w|_a = |w|_b\}$$

Notes

Correction

On va prouver $\forall w . ((S \Longrightarrow^n w) \Rightarrow (w \in L))$ par induction sur n.
Hypothèse :

$$\forall \alpha . \forall k . ((k < n \wedge S \Longrightarrow^k \alpha) \Rightarrow (\alpha \in L))$$

- $n = 1$
  1. $S \Longrightarrow^1 \epsilon$:
     on a bien $|\epsilon|_a = |\epsilon|_b = 0$
- $n > 1 : S \Longrightarrow^n w$
  1. $S \Longrightarrow^1 SS \Longrightarrow^{n-1} w$: Par le théorème de décomposition il existe w_1 et w_2 tels que $w = w_1 w_2$ et $S \Longrightarrow^{n1} w_1$ et $S \Longrightarrow^{n2} w_2$ et $n1 + n2 = n - 1$. On a $n1 < n$ et $n2 < n$ et donc $|w_1|_a = |w_1|_b$ et $|w_2|_a = |w_2|_b$. On en déduit $|w_1 w_2|_a = |w_1 w_2|_b$ et donc $w \in L$.
  2. $S \Longrightarrow^1 aSb \Longrightarrow^{n-1} w$: ...
  3. $S \Longrightarrow^1 bSa \Longrightarrow^{n-1} w$: ...

Notes

Complétude

On va prouver $\forall w\ .\ ((w \in L) \Rightarrow (S \Longrightarrow^* w))$ par induction sur $n = |w|$.
Hypothèse :

$$\forall \alpha\ .\ ((|\alpha| < n \wedge\ \alpha \in L) \Rightarrow (S \Longrightarrow^* \alpha))$$

- $n = 0$, i.e $w = \epsilon$. On a bien $S \Longrightarrow^* w$ par la règle $S \to \epsilon$.
- $|w| > 0$. On considère les 4 cas suivants :
  1. $w = aub$
  2. $w = bua$
  3. $w = aua$
  4. $w = bub$

qui couvre tous les cas de mots w de L tel que $|w| > 0$.

Notes

Exemple (suite)

1. $w = aub$ (similaire pour $w = bua$)
2. $w = aua$ (similaire pour $w = bub$)

Cas 1 :

on a $|u| < |w|$ et $u \in L$ (u a autant de a que de b) donc il existe une dérivation $S \Longrightarrow^* u$ (par hypothèse d'induction) et donc
$S \Longrightarrow aSb \Longrightarrow^* aub = w$.

Notes

Exemple (suite 2)

Cas 2 : on va prouver que w a un préfixe propre (different de ϵ et de w) qui est dans L. On aura donc $w = w1w2$ avec $w1 \in L$ et $w2 \in L$ et $|w1| < |w|$ et $|w2| < |w|$. On peut donc construire $w1$ et $w2$ à partir de S ($S \Longrightarrow^* w1$ et $S \Longrightarrow^* w2$). On peut donc construire w par la règle $S \rightarrow SS$.

Comme $w = aua$ et $w \in L$ alors w a nécessairement un préfixe propre ayant plus de b que de a. Soit x le plus petit de ces préfixes ($|x|_b > |x|_a$).
Il s'écrit ayb car w commence par a et que si x ne finit pas par b ce n'est pas le plus petit préfixe ayant plus de b. On a alors $|ay|_a = |x|_a$ et $|ay|_b = |x|_b - 1$.
On ne peut pas avoir $|x|_b - 1 > |x|_a$ sinon ay est un préfixe plus petit que x qui vérifie $|ay|_b > |ay|_a$. On a donc : $|x|_b - 1 = |x|_a$ et donc ay est dans L.

Notes

Exemple plus complexe

L : nombre pair de a

(1) $P \rightarrow bP$
(2) $P \rightarrow aI$
(3) $P \rightarrow \epsilon$
(4) $I \rightarrow bI$
(5) $I \rightarrow aP$

$\Rightarrow$ Ici il faut faire les preuves à la fois sur P et I.

Notes