Théorie des langages
Grammaires et langages algébriques

TELECOM Nancy (1A)

2019-2020

Introduction

- Les langages algébriques (ou hors-contexte ou context free) sont les langages engendrés par les grammaires algébriques. Dans la hiérarchie (de Chomsky) des langages, ils correspondent à la classe immédiatement supérieure à celle des langages réguliers.
- Les langages algébriques ont des applications dans le domaine des langages de programmation (analyse sntaxique, compilation) et en linguistique.
- A TELECOM Nancy, on parle des langages algébriques :
  - dans le module de 1A "Mathématiques pour l'Informatique" : théorie des langages (MAI1), l'analyse syntaxique descendante (MAI2).
  - dans dans le module de Traduction 2A : analyse syntaxique ascendante, utilisation d'outils dédiés à l'analyse syntaxique, compilation des langages.
- En linguistique, les grammaires algébriques sont utilisées dans le traitement automatique des langues (TAL).

Plan

- Définitions et exemples (grammaire algébrique, dérivation, arbre syntaxique, mot généré, langage engendré)
- Ambiguïté d'une grammaire algébrique, d'un langage algébrique
- Expressions arithmétiques : notations fonctionnelle, préfixée, postfixée, infixée
- Langages réguliers et langages algébriques
  - Grammaires régulières à droite
  - Lemmes de l'étoile
- Propriétés des langages algébriques

Définition

On appelle grammaire algébrique un quadruplet $G = (N, T, \rightarrow, X)$ tel que :

- N est un vocabulaire (**fini**) auxiliaire de la grammaire appelé l'ensemble des **symboles non terminaux**. Intuitivement chaque élément de N doit se comprendre comme le nom d'une catégorie syntaxique de mots sur le vocabulaire T de la grammaire.
- T est un autre vocabulaire (**fini**) disjoint de N, les éléments de T s'appellent des **terminaux**. C'est sur ce vocabulaire que sont définis les mots engendrés par la grammaire.
- $\rightarrow$ est une relation binaire **finie** de N vers $(N \cup T)^*$, chaque élément de la relation est une règle de la grammaire G et est notée $A \rightarrow \alpha$ où $A \in N$ et $\alpha \in (N \cup T)^*$.
- X est un élément distingué de N appelé l'axiome de la grammaire et représentant intuitivement la catégorie syntaxique la plus large du langage que l'on veut engendrer.

Remarques

- Pour chaque règle $A \to \alpha$, A est un non terminal appelé, membre gauche de la règle, α est le membre droit de la règle.
- Par convention, les lettres majuscules dénotent généralement les non terminaux, et les lettres minuscules les terminaux. On adopte cette convention dans ce cours.
- Lorsque l'on a plusieurs règles avec le même membre gauche, il est courant de les écrire en factorisant le membre gauche commun. Par exemple si l'on a les trois règles suivantes :

$$A \to \alpha_1 \qquad A \to \alpha_2 \qquad A \to \alpha_3$$

on écrit

$$A \to \alpha_1 \mid \alpha_2 \mid \alpha_3$$

la barre verticale "$\mid$" dénote alors l'alternative.

Exemple 1 : grammaire (très simplifiée) de la langue française

- $N = \{ \quad < \textit{phrase} >, < \textit{groupe nominal} >, < \textit{groupe verbal} >,$
  $< \textit{groupe-verbal-etre} >, < \textit{groupe complement} >,$
  $< \textit{nom} >, < \textit{article} >, < \textit{verbe} >, < \textit{adjectif} >,$
  $< \textit{preposition} >, < \textit{attribut} >, \ldots\}$
- $T =$
  $\{\textit{le}, \textit{la}, \textit{chat}, \textit{souris}, \textit{table}, \textit{est}, \textit{mange}, \textit{dans}, \textit{sur}, \textit{beau}, \textit{belle}, \ldots\}$
  Le vocabulaire T (l'ensemble des terminaux) de la grammaire est l'ensemble de tous les mots de la langue française.
- $X =< \textit{phrase} >$, l'axiome est le concept syntaxique le plus général de la grammaire, ici c'est la phrase, correspondant au symbole $< \textit{phrase} >$.

Exemple 1 (suite et fin)

Les règles de grammaire engendrant la langue française sont très nombreuses (quelques milliers à quelques dizaines de milliers). Donnons quelques exemples caractéristiques.

$$
\begin{array}{lcl}
< \textit{phrase} > & \rightarrow & < \textit{groupe nominal} >< \textit{groupe verbal} > \\
 & & \qquad < \textit{groupe complement} > | \\
 & \rightarrow & < \textit{groupe nominal} >< \textit{groupe-verbal-etre} > \\
 & & \qquad < \textit{attribut} > | \ldots \\
< \textit{groupe nominal} > & \rightarrow & < \textit{article} >< \textit{adjectif} >< \textit{nom} > | \\
 & \rightarrow & < \textit{article} >< \textit{nom} >< \textit{adjectif} > | \\
 & \rightarrow & < \textit{article} >< \textit{nom} > | \ldots \\
< \textit{article} > & \rightarrow & \textit{le} \mid \textit{la} \mid \textit{les} \mid \textit{des} \mid \ldots \\
< \textit{adjectif} > & \rightarrow & \textit{beau} \mid \textit{belle} \mid \textit{petit} \mid \textit{mort} \mid \textit{noir} \mid \ldots \\
< \textit{nom} > & \rightarrow & \textit{chat} \mid \textit{souris} \mid \textit{table} \mid \ldots \\
< \textit{preposition} > & \rightarrow & \textit{dans} \mid \textit{sur} \mid \ldots \\
< \textit{groupe verbal} > & \rightarrow & \textit{mange} \mid \textit{est} \mid \ldots \\
< \textit{groupe-verbal-etre} > & \rightarrow & \textit{est} \mid \textit{sont} \mid \ldots \\
< \textit{attribut} > & \rightarrow & < \textit{adjectif} > \mid \ldots \\
< \textit{groupe complement} > & \rightarrow & < \textit{groupe nominal} > \mid \\
 & \rightarrow & < \textit{preposition} >< \textit{groupe nominal} > \mid \ldots
\end{array}
$$

Exemple de phrases engendrées par la grammaire précédente :

- *le petit chat est mort*
- *le chat mange sur le table*
- *les souris sont beaux*
- *la table mange la souris*

Exemple 2 : le langage de programmation Pascal

$$
\begin{aligned}
N = \{ \quad & < \textit{programme} >, < \textit{partie declaration} >, \\
& < \textit{partie instruction} >, < \textit{declaration} >, < \textit{instruction} >, \\
& < \textit{affectation} >, < \textit{identificateur} >, \\
& < \textit{conditionnelle} >, < \textit{iteration} >, < \textit{procedure} >, \ldots\} \\
T = \quad & \{A,\ B,\ \ldots,\ Z,\ a, \ldots,\ z,\ (,\ ),\ [,\ ],\ +,\ *,\ /,\ ;,\ \ldots\} \\
X = \quad & < \textit{programme} >
\end{aligned}
$$

$$
\begin{aligned}
< \textit{programme} > \quad & \rightarrow \quad \textit{PROGRAM} < \textit{identificateur} >; \\
& \qquad\qquad\qquad\qquad < \textit{partie declaration} >; \\
& \qquad < \textit{partie instruction} > \\
< \textit{identificateur} > \quad & \rightarrow \quad \ldots \\
< \textit{partie declaration} > \quad & \rightarrow \quad \ldots \\
< \textit{partie instruction} > \quad & \rightarrow \quad < \textit{declaration} > \ | \\
& \rightarrow \quad < \textit{declaration} >; < \textit{partie instruction} > \ | \\
& \rightarrow \quad \ldots
\end{aligned}
$$

Définition

Soit $G = (N,\ T,\ \rightarrow,\ X)$ une grammaire algébrique. On définit sur $(N \cup T)^*$ les relations de réécriture $\rightarrowtail$, de dérivation $\overset{*}{\rightarrowtail}$ et de dérivation stricte $\overset{+}{\rightarrowtail}$ de la manière suivante :

1. $\alpha \rightarrowtail \beta$ si et seulement si, il existe $\alpha_1 \in (N \cup T)^*$, $\alpha_2 \in (N \cup T)^*$, $A \in N$, $\gamma \in (N \cup T)^*$ tels que $\alpha = \alpha_1 A \alpha_2$ et $\beta = \alpha_1 \gamma \alpha_2$ et $A \rightarrow \gamma$ est une règle de G.
2. $\alpha \overset{*}{\rightarrowtail} \beta$ si et seulement si, il existe un entier n et une suite finie $(\alpha_i)_{0 \leq i \leq n}$ telle que $\alpha = \alpha_0$ et $\beta = \alpha_n$ et pout tout i de [0, n-1], $\alpha_i \rightarrowtail \alpha_{i+1}$.
3. $\alpha \overset{+}{\rightarrowtail} \beta$ si et seulement si, il existe un entier n non nul et une suite finie $(\alpha_i)_{0 \leq i \leq n}$ telle que $\alpha = \alpha_0$ et $\beta = \alpha_n$ et pout tout i de [0, n-1], $\alpha_i \rightarrowtail \alpha_{i+1}$.

La suite $(\alpha_i)_{0 \leq i \leq n}$ s'appelle une dérivation de α à β (ou une dérivation au sens large si $n > 0$).

Exemple

G définie par $N = \{X,\ A,\ B\}$, $T = \{a, b\}$, X l'axiome de la grammaire, les règles sont les suivantes :

$$X \to \varepsilon \mid aB \mid bA$$
$$A \to aX \mid bAA$$
$$B \to bX \mid aBB$$

Exemples de réécritures :

$aBbA \rightarrowtail aaBBbA$

$aaBBbA \rightarrowtail aaBbXbA$

$aaBbXbA \rightarrowtail aaBb\varepsilon bA = aaBbbA$

Exemple de dérivation :

$aBbA \rightarrowtail aaBBbA \rightarrowtail aaBbXbA \rightarrowtail aaBbbA$ d'où $aBbA \overset{*}{\rightarrowtail} aaBbbA$

Mot et langage engendrés par une grammaire algébrique

Définition

Soit $G = (N,\ T,\ \rightarrow,\ X)$ une grammaire algébrique.

- On dit qu'un mot α de T^* est engendré par la grammaire G à partir de A ($A \in N$) si et seulement si $A \overset{*}{\rightarrowtail} \alpha$.
- On dit qu'un mot α de T^* est engendré par la grammaire G si et seulement si $X \overset{*}{\rightarrowtail} \alpha$ (α est engendré par G à partir de l'axiome X de la grammaire).
- Le langage engendré par la grammaire G à partir de A ($A \in N$) est l'ensemble des mots engendrés par G à partir de A, on le note $L(G,\ A)$.
- Le langage engendré par la grammaire G est le langage engendré par la grammaire à partir de l'axiome X de la grammaire, on le note $L(G)$. On a $L(G) = L(G,\ X) = \{\alpha\ ;\ \alpha \in T^*\ et\ X \overset{*}{\rightarrowtail} \alpha\}$.
- Un langage engendré par une grammaire algébrique s'appelle un langage algébrique (ou hors-contexte).

Exemple de langage engendré par une grammaire algébrique

- Soit la grammaire suivante $G = (\{X\},\ \{a, b\},\ \rightarrow,\ X)$, où les règles sont définies par $X \rightarrow aXb \mid \varepsilon$
  On a $L(G) = \{a^n b^n\ ;\ n \geq 0\}$.
- Soit la grammaire suivante $G = (\{X\},\ \{a, b\},\ \rightarrow,\ X)$ où les règles sont $X \rightarrow aXa \mid bXb \mid \varepsilon$.
  Le langage engendré par G est $L(G) = \{w\widetilde{w}\ ;\ w \in \{a, b\}^*\}$ ($\widetilde{w}$ est le mot obtenu en renversant le mot w).

Exemples de dérivations

Soit la grammaire algébrique $G = (\{X\},\ \{a, b\},\ \rightarrow,\ X)$, où $\rightarrow$ est défini par

$$X \rightarrow XX\ |1|\ aXa\ |2|\ bXb\ |3|\ \varepsilon|4|$$

Soit le mot *aabaab*, on veut montrer que $aabaab \in L(G)$

$X \overset{1}{\rightarrowtail} XX \overset{2}{\rightarrowtail} aXaX \overset{4}{\rightarrowtail} aaX \overset{3}{\rightarrowtail} aabXb \overset{2}{\rightarrowtail} aabaXab \overset{4}{\rightarrowtail} aabaab$

$X \overset{1}{\rightarrowtail} XX \overset{3}{\rightarrowtail} XbXb \overset{2}{\rightarrowtail} XbaXab \overset{4}{\rightarrowtail} XbaXab \overset{2}{\rightarrowtail} aXabaab \overset{4}{\rightarrowtail} aabaab$

il y 8 autres dérivations possibles pour générer le mot *aabaab* à partir de l'axiome X.

Remarque : le fait que l'ordre dans lequel les règles sont appliquées n'est pas important est une caractéristique des grammaires algébriques, aussi appelées grammaires hors-contexte (car un non-terminal peut être remplacé indépendamment de la chaîne de caractères qui l'entoure).

Définition

Soit $G = (N,\ T,\ \rightarrow,\ X)$, un arbre syntaxique pour la grammaire G est un arbre étiqueté par les éléments de $N \cup T \cup \{\varepsilon\}$ qui satisfait les conditions suivantes :

- la racine de l'arbre est étiquetée par X, l'axiome de G.
- chaque nœud interne est étiqueté par un élément de N. Chaque feuille est étiquetée par un élément de T ou par ε.
- pour tout nœud interne, si son étiquette est un non-terminal A et si ses descendants immédiats sont les nœuds $n_1, \ldots, n_k$ ayant respectivement pour étiquettes $X_1, \ldots, X_k$ alors $A \rightarrow X_1 \ldots X_k$ doit être une règle de G.
- si un nœud est étiqueté par ε, alors ce nœud est le seul descendant immédiat de son prédécesseur (cette dernière règle évite l'introduction d'instances inutiles de ε dans l'arbre syntaxique).

Mot généré par un arbre syntaxique

Définition

Le mot généré par un arbre syntaxique est celui obtenu par la concaténation des feuilles de l'arbre prises de gauche à droite.

Exemple

Soit la grammaire algébrique $G = (\{X\},\ \{a, b\},\ \rightarrow,\ X)$, où $\rightarrow$ est défini par

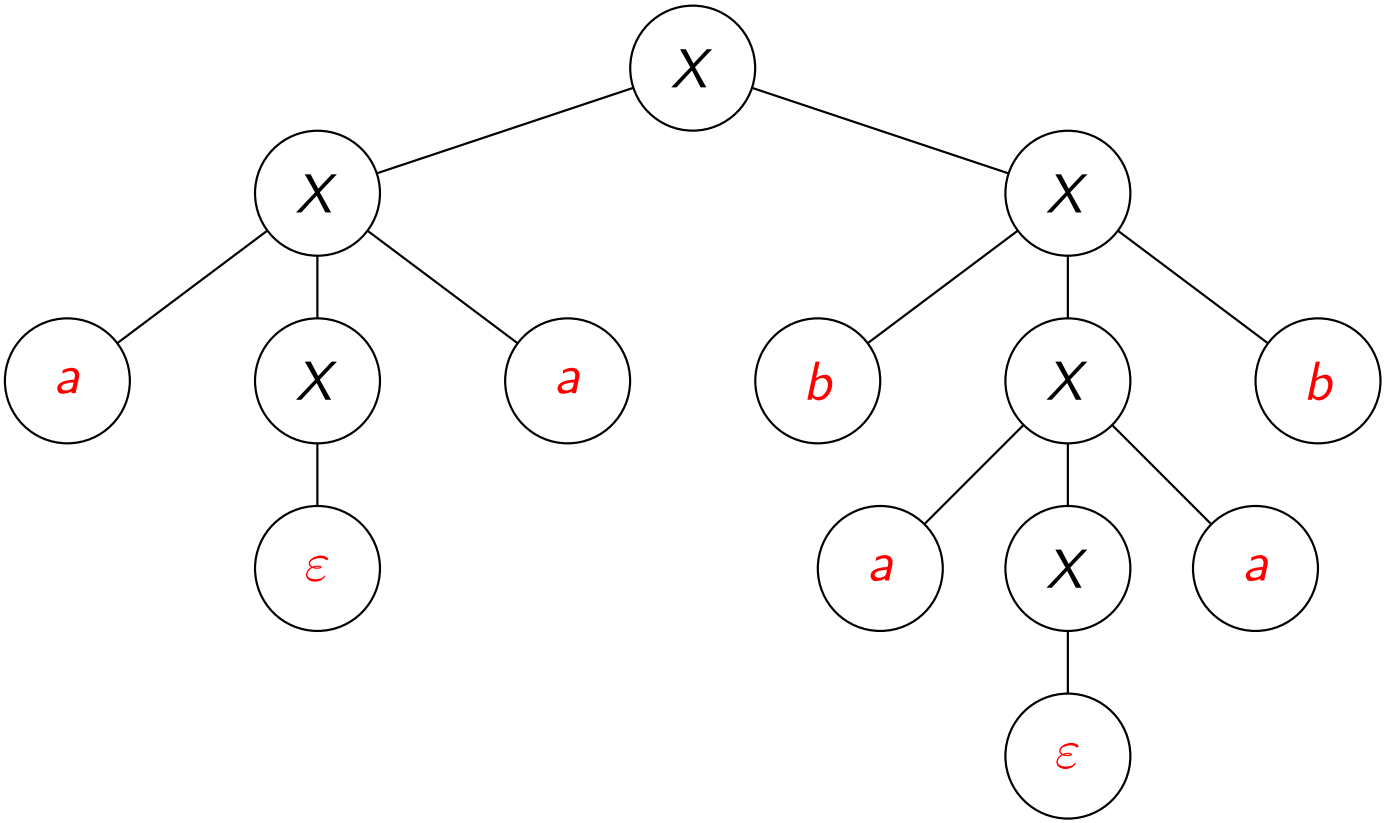
$$X \rightarrow XX \mid aXa \mid bXb \mid \varepsilon$$

Proposition

Soit G une grammaire algébrique, un mot w est généré par G si et seulement si il existe un arbre syntaxique de la grammaire G qui génère w.

Exemple d'arbre syntaxique

Un arbre syntaxique du mot *aabaab*



Ambiguïté d'une grammaire algébrique et d'un langage algébrique

Définition

Une grammaire G est ambiguë si et seulement si il existe un mot $w \in L(G)$ engendré par deux arbres syntaxiques différents.

Définition

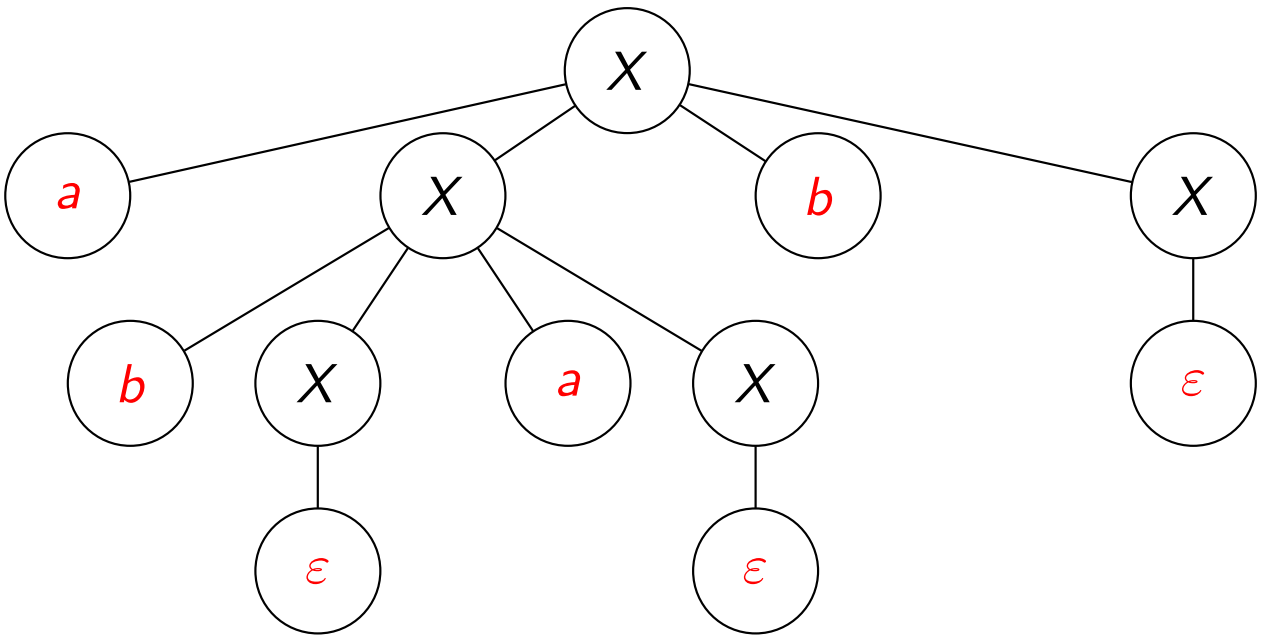
Soit L un langage algébrique. On dit que L possède une ambiguïté inhérente si et seulement si toute grammaire algébrique engendrant L est ambiguë.

Exemple

Soit $G = (\{X\},\ \{a,\ b\},\ \rightarrow,\ X)$, la grammaire définie par les règles suivantes : $X \rightarrow aXbX\ |bXaX\ |\varepsilon$, est ambiguë.
Le langage $L(G)$ ne possède pas d'ambiguïté inhérente (on peut montrer qu'il existe au moins une grammaire non ambiguë qui l'engendre).

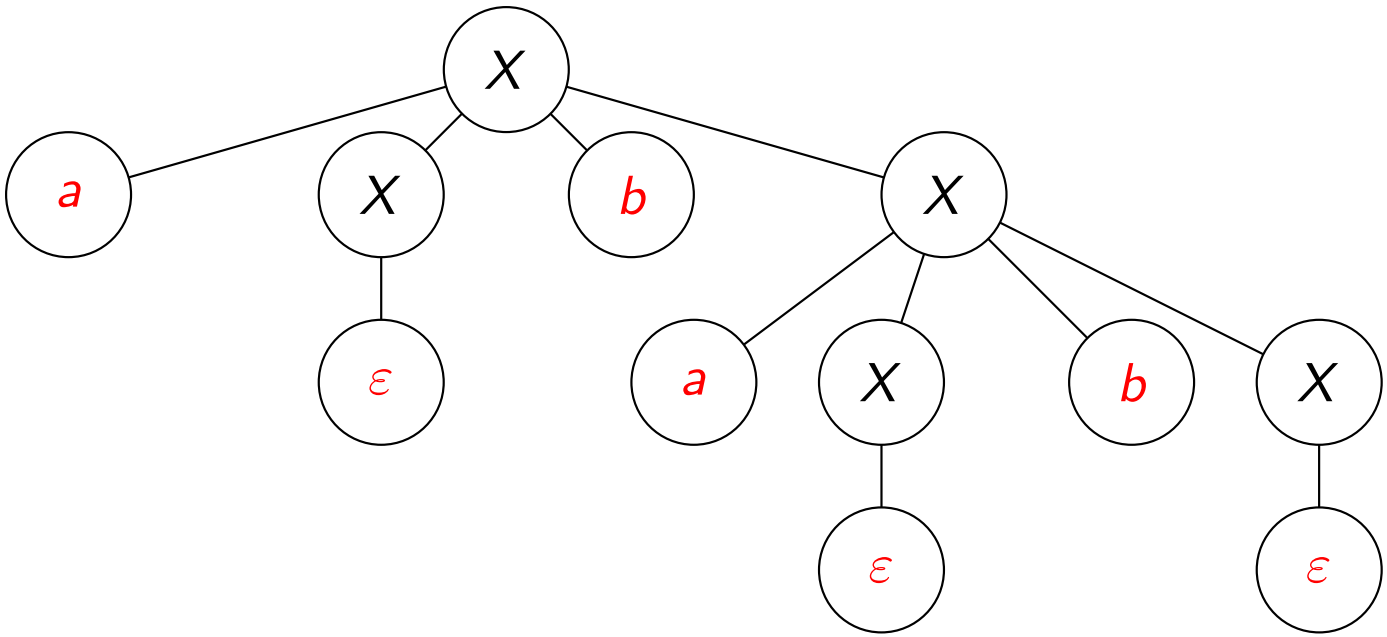
Exemple : ambiguïté

Soit $G = (\{X\},\ \{a,\ b\},\ \rightarrow,\ X)$, la grammaire définie par les règles $X \rightarrow aXbX\ |bXaX\ |\varepsilon$ est ambiguë. Arbre syntaxique du mot *abab* :



Exemple : ambiguïté (suite)

Deuxième arbre syntaxique de *abab*



Le mot *abab* a deux arbres syntaxiques, il est ambigu, la grammaire G est donc ambiguë.

Les expressions arithmétiques

Les langages de programmation classiques (PASCAL, LISP, FORTRAN, C, ...) sont en première approximation des expressions arithmétiques compliquées...
On s'intéresse ici à la **syntaxe** des expressions arithmétiques. Pour construire les expressions arithmétiques on a besoin de noms de fonctions. Les symboles de fonctions sont classés selon leur **arité** (nombre d'arguments de la fonction) :

- $\{a,\ b,\ c\}$ les symboles de constantes (d'arité 0).
- $\{s,\ f,\ g\}$ les symboles d'arité 1.
- $\{+,\ *,\ /, -\}$ les opérateurs binaires.
- $\{\psi,\ \varphi\}$ les opérateurs d'arité 3.

On dispose d'un ensemble de symboles de variables
$X = \{x,\ y,\ z,\ x',\ y',\ z'\}$.
L'ensemble des symboles terminaux est constitué de l'ensemble des symboles de fonctions et des symboles de variables (plus éventuellement des symboles de ponctuation et des parenthèses selon les cas).

Notation fonctionnelle

- $G = (\{X,\ A,\ B,\ C,\ D,\ V\},\ T,\ \rightarrow,\ X)$ telle que
  - $T = \{x,\ y,\ z,\ x',\ y',\ z'\} \cup \{a,\ b,\ c\} \cup \{s,\ f,\ g\} \cup \{+,\ *,\ /, -\} \cup \{\psi,\ \varphi\} \cup \{(,\ ),\ ,\}$
  - $\rightarrow$ est définie par :

$$
\begin{array}{l}
X \rightarrow V \mid A \mid B(X) \mid C(X,\ X) \mid D(X,\ X,\ X) \\
V \rightarrow x \mid y \mid z \mid x' \mid y' \mid z' \\
A \rightarrow a \mid b \mid c \\
B \rightarrow s \mid f \mid g \\
C \rightarrow + \mid * \mid / \mid - \\
D \rightarrow \psi \mid \varphi
\end{array}
$$

- $*(\phi(s(x),\ f(+(a, y)), b), /(\psi(a, z, +(y, c)), *(x, y)))$ est un mot engendré par G

Remarque : les parenthèses et les virgules séparent les opérandes des fonctions, elles permettent une meilleure lisibilité des formules arithmétiques.

Notation polonaise préfixée

- $G = (\{X,\ A,\ B,\ C,\ D,\ V\},\ T,\ \rightarrow,\ X)$ telle que
  - $T =$
    $\{x,\ y,\ z,\ x',\ y',\ z'\} \cup \{a,\ b,\ c\} \cup \{s,\ f,\ g\} \cup \{+,\ *,\ /, -\} \cup \{\psi,\ \varphi\}$
  - $\rightarrow$ est définie par :

$$
\begin{array}{l}
X \rightarrow V \mid A \mid BX \mid CXX \mid DXXX \\
V \rightarrow x \mid y \mid z \mid x' \mid y' \mid z' \\
A \rightarrow a \mid b \mid c \\
B \rightarrow s \mid f \mid g \\
C \rightarrow + \mid * \mid / \mid - \\
D \rightarrow \psi \mid \varphi
\end{array}
$$

- $*\phi sxf + ayb/\psi az + yc * xy$ est un mot engendré par G

Remarque : notation minimale qui consiste à écrire les opérateurs avant les opérandes. Cette notation est bien adaptée si l'on veut démontrer des résultats théoriques sur les expressions arithmétiques.

Notation polonaise inverse ou notation postfixée

- $G = (\{X,\ A,\ B,\ C,\ D,\ V\},\ T,\ \rightarrow,\ X)$ telle que
  - $T = \{x,\ y,\ z,\ x',\ y',\ z'\} \cup \{a,\ b,\ c\} \cup \{s,\ f,\ g\} \cup \{+,\ *,\ /, -\} \cup \{\psi,\ \varphi\}$
  - $\rightarrow$ est définie par :

$$
\begin{aligned}
&X \rightarrow V \mid A \mid XB \mid XXC \mid XXXD \\
&V \rightarrow x \mid y \mid z \mid x' \mid y' \mid z' \\
&A \rightarrow a \mid b \mid c \\
&B \rightarrow s \mid f \mid g \\
&C \rightarrow + \mid * \mid / \mid - \\
&D \rightarrow \psi \mid \varphi
\end{aligned}
$$

- $xsay + fb\phi azyc + \psi xy * /*$ est un mot engendré par G

Remarque : les opérandes sont écrits avant les opérateurs. Cette notation est pratique si l'on veut évaluer les expressions, elle a été utilisée par certaine marque de calculatrice.

La notation infixée

C'est la notation usuelle qui consiste à noter les opérateurs binaires sous forme infixée, elle nécessite d'utiliser des parenthèses et des virgules. Les opérateurs d'arité 1 ou supérieures à 2 sont sous forme préfixée.
L'écriture de ces grammaires se fera en exercice.

Exemple de mot : $\phi(s(x),\ f(a+y),\ b)*(\psi(a,\ z,\ y+c)/(x*y))$

Démonstration de L(G) = E : des exemples

- Soit la grammaire suivante $G = (\{X\},\ \{a, b\},\ \rightarrow,\ X)$, où les règles de G sont définies par $X \rightarrow aXb \mid \varepsilon$. On a $L(G) = \{a^n b^n\ ;\ n \geq 0\}$.
- Soit la grammaire suivante $G = (\{X\},\ \{a, b\},\ \rightarrow,\ X)$ où les règles sont $X \rightarrow aXa \mid bXb \mid \varepsilon$. Montrons que ce langage engendré par G est $L(G) = \{w\widetilde{w}\ ;\ w \in \{a, b\}^*\}$ ($\widetilde{w}$ est le mot obtenu en renversant le mot w).

  Preuve :
  Soit un mot $u \in \{a, b\}^*$ de $L(G)$, on va montrer que u s'écrit $w\widetilde{w}$.
  $u \in L(G)$ donc $X \overset{*}{\rightarrowtail} u$, on effectue un raisonnement par récurrence sur la longueur n de la dérivation.
  - si $n = 1$ alors $X \rightarrowtail \varepsilon$ d'où $u = \varepsilon$ et $u = \varepsilon\widetilde{\varepsilon}$
  - si $n > 1$ alors on a l'une des deux dérivations suivantes :
    $X \rightarrowtail aXa \rightarrowtail \ldots \rightarrowtail u$ ou $X \rightarrowtail bXb \rightarrowtail \ldots \rightarrowtail u$
    La première dérivation implique qu'il existe un mot v de $\{a,\ b\}^*$ tel que $u = ava$, on a donc $X \rightarrowtail aXa \rightarrowtail \ldots \rightarrowtail ava$, et la dérivation $X \rightarrowtail \ldots \rightarrowtail v$ a comme longueur $(n-1)$, par hypothèse de récurrence il existe donc un mot s sur $\{a, b\}$ tel que $v = s\widetilde{s}$ et par conséquent $u = as\widetilde{s}a$, comme $\widetilde{as} = \widetilde{s}a$, en posant $w = as$, on a $u = w\widetilde{w}$.
    Même raisonnement avec la deuxième dérivation...

Exemples (suite)

Réciproque : soit un mot u de la forme $w\widetilde{w}$, montrons qu'il est engendré par G, on raisonne par récurrence sur la longueur du mot u.

- si $|u| = 0$ alors $u = \varepsilon$, ε est engendré par G car $X \rightarrowtail \varepsilon$
- si $|u| > 0$ alors $u = av\widetilde{v}a$ ou $u = bv\widetilde{v}b$, on suppose que $u = av\widetilde{v}a$ (l'autre cas se traite de façon identique),
  comme $|v\widetilde{v}| < |u|$ par hypothèse de récurrence $v\widetilde{v}$ est engendré par G,
  donc il existe une dérivation de la forme $X \rightarrowtail \ldots \rightarrowtail v\widetilde{v}$
  et on a $X \rightarrowtail aXa \rightarrowtail \ldots \rightarrowtail av\widetilde{v}a = u$, d'où $u \in L(G)$

Remarque

Si G est une grammaire algébrique, démontrer $L(G) = E$ revient à montrer les **deux** inclusions $L(G) \subset E$ et $E \subset L(G)$.

- $L(G) \subset E$ se démontre par récurrence sur la longueur de la dérivation d'un mot α de $L(G)$
- $E \subset L(G)$ se démontre par récurrence sur la longueur d'un mot α de E

Définition

Soit $G = (N,\ T,\ \rightarrow,\ X)$ une grammaire algébrique, on dit que G est une grammaire régulière à droite si toutes ses règles sont de la forme :

$$A \rightarrow aB \quad ou \quad A \rightarrow a \quad ou \quad A \rightarrow \varepsilon$$

où $A,\ B \in N$ et $a \in T$.

Remarque

Les membres droits des règles de grammaires régulières à droite contiennent au plus un non terminal qui est situé à droite.

Proposition

1. Soit $\mathcal{A} = (A,\ Q,\ q_0,\ \delta,\ T)$ un automate fini déterministe, le langage régulier $L(\mathcal{A})$ reconnu par $\mathcal{A}$ est engendré par la grammaire algébrique suivante $G = (Q,\ A,\ \rightarrow,\ q_0)$ où la relation $\rightarrow$ est définie par :
   - $q \rightarrow aq'$ si et seulement si $q' = \delta(q,\ a)$.
   - $q \rightarrow \varepsilon$ si et seulement si q est un état terminal.
2. Soit $\mathcal{A} = (A,\ Q,\ I,\ E,\ T)$ un automate fini non déterministe, le langage régulier $L(\mathcal{A})$ reconnu par $\mathcal{A}$ est engendré par la grammaire algébrique suivante $G = (Q \cup \{q_N\},\ A,\ \rightarrow,\ q_N)$ où q_N est un nouvel état et la relation $\rightarrow$ est définie par :
   - $q \rightarrow aq'$ si et seulement si $(q,\ a,\ q') \in E$.
   - $q \rightarrow q'$ si et seulement si $(q, \varepsilon, q') \in E$.
   - $q \rightarrow \varepsilon$ si et seulement si $q \in T$.
   - $q_N \rightarrow q_i$ si et seulement si $q_i \in I$.

Corollaire

Tout langage régulier est un langage algébrique.

Preuve de la proposition

On démontre la première partie de la proposition. Par récurrence sur la longueur du mot $\alpha \in A^*$ on montre l'équivalence suivante :

$$\delta(q,\ \alpha) = q' \textbf{ si et seulement si } q \overset{*}{\rightarrowtail} \alpha q'$$

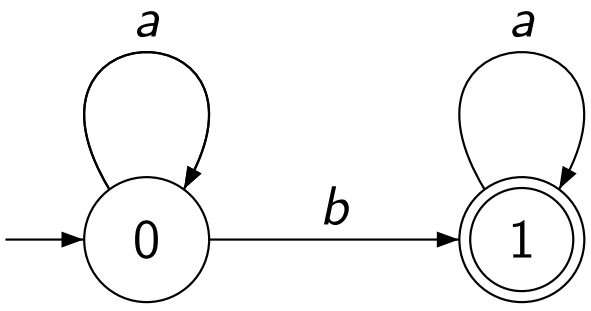
pour tout mot α et tout état (ou non terminal) q, q'.

1. Si le mot α est de longueur 0, $\alpha = \varepsilon$, le résultat est trivial, puisque $\delta(q, \varepsilon) = q$ et $q \overset{*}{\rightarrowtail} q$ sont vraies.
2. Hypothèse de récurrence : $\forall (q, q') \in Q^2\ \delta(q,\ \alpha) = q' \Leftrightarrow q \overset{*}{\rightarrowtail} \alpha q'$ pour $\alpha \in A^*$
   1. Si $\delta(q,\ \alpha a) = \delta(\delta(q,\ \alpha), a) = q''$, si l'on pose $\delta(q,\ \alpha) = q'$ en appliquant l'hypothèse de récurrence on déduit (i) $q \overset{*}{\rightarrowtail} \alpha q'$, comme d'autre part $\delta(q', a) = q''$ en appliquant l'hypothèse de récurrence on a (ii) $q' \rightarrowtail a q''$. Des deux dérivations (i) et (ii), on déduit $q \overset{*}{\rightarrowtail} \alpha a q''$.
   2. Si $q \overset{*}{\rightarrowtail} \alpha a q''$, comme la grammaire G est récursive à droite, la lettre **a** a été produite par une règle de la forme $q' \rightarrow a q''$, la dérivation peut alors s'écrire $q \overset{*}{\rightarrowtail} \alpha q' \rightarrowtail \alpha a q''$, par hypothèse de récurrence on a donc $\delta(q, \alpha) = q'$ et $\delta(q', a) = q''$, d'où l'on peut déduire $\delta(q,\ \alpha a) = q''$.

Il reste à montrer que les mots reconnus par l'automate $\mathcal{A}$ sont exactement ceux générés par la grammaire G. α reconnu par $\mathcal{A}$ **ssi** $\delta(q_0, \alpha) = q$ avec $q \in T$ **ssi** $q_0 \overset{*}{\rightarrowtail} \alpha q$ avec $q \in T$ **ssi** $q_0 \overset{*}{\rightarrowtail} \alpha$ car comme $q \in T$ il existe dans G une règle $q \rightarrow \varepsilon$. CQFD

Exemple

Soit $\mathcal{A} = (\{a, b\},\ \{0,\ 1\}, 0,\ \delta,\ \{1\})$ tel que $\delta(0, a) = 0$, $\delta(0, b) = 1$, $\delta(1, a) = 1$



La grammaire correspondant à cet automate est
$G = (\{X, A\},\ \{a, b\},\ \rightarrow,\ X)$ est
X correspond à l'état 0 et A à l'état 1.
Les règles sont :

$$
\begin{array}{lcll}
X & \rightarrow & aX & (\delta(0, a) = 0) \\
X & \rightarrow & bA & (\delta(0, b) = 1) \\
A & \rightarrow & aA & (\delta(1, a) = 1) \\
A & \rightarrow & \varepsilon & (1 \textit{ est un état terminal})
\end{array}
$$

Proposition

Une grammaire $G = (N,\ A,\ \rightarrow,\ X)$ régulière à droite, engendre le langage régulier reconnu par l'automate fini suivant :
$\mathcal{A} = (A,\ N,\ \{X\},\ E,\ T)$ tel que :

1. $(A, a, B) \in E$ ssi $A \rightarrow aB$.
2. $A \in T$ pour toute règle de la forme $A \rightarrow \varepsilon$.
3. S'il existe dans G des règles $A \rightarrow a$, ajouter à N et T un nouvel état q_N et dans E les transitions (A, a, q_N) (l'état q_N étant le même pour toutes les règles $A \rightarrow a$).

Cette proposition est la réciproque de la proposition précédente, on peut conclure qu'un langage est régulier si et seulement s'il peut être engendré par une grammaire régulière à droite.

Lemme de l'étoile (première version)

Soit L un langage régulier sur l'alphabet A, il existe un entier p tel que, pour tout mot α de L de longueur supérieure à p, il existe une factorisation de α en uvw $(u, v, w \in A^*)$ avec $v \neq \varepsilon$, telle que pour tout n dans $\mathbb{N}$, $uv^n w$ est dans L.
Ce qui se formalise par :
$(\forall \alpha \in L)\ |\alpha| \geq p \Rightarrow [(\exists u, v, w \in A^*)\ \alpha = uvw \text{ et } v \neq \varepsilon \text{ et } (\forall n \in \mathbb{N}) uv^n w \in L]$

Démonstration

1. Si L est fini, on pose $p = \max_{\beta \in L} |\beta| + 1$, comme il n'existe pas de mot de longueur supérieure à p la prémisse de l'implication est fausse donc l'implication est trivialement vraie.
2. Si L est infini, soit p le nombre d'états d'un automate reconnaissant le langage L, soit α un mot de L de longueur supérieure (ou égale) à p, il existe un calcul partant de l'état initial de l'automate et aboutissant à un état final et dont l'étiquette est le mot α, comme $|\alpha| \geq p$ et que le nombre d'états de l'automate est p, il existe dans ce calcul un état q_i apparaissant deux fois et il existe donc un calcul d'origine q_i et d'extrémité q_i, soit v l'étiquette de ce calcul, α peut se décomposer en $\alpha = uvw$ avec $v \neq \varepsilon$ et $uv^n w \in L$ (car on peut "tourner" dans la boucle allant de q_i et menant à q_i un nombre quelconque de fois).

Lemme de l'étoile (deuxième version)

Soit L un langage régulier sur l'alphabet A, il existe un entier p tel que pour tout mot α de L et toute factorisation de la forme $\alpha = \gamma\beta\delta$ telle que $|\beta| \geq p$, il existe une factorisation $\beta = uvw$ avec $v \neq \varepsilon$ et telle que pour tout n de $\mathbb{N}$, le mot $\gamma uv^n w\delta$ appartient à L.

Démonstration

La démonstration est analogue à celle du premier lemme de l'étoile.

Remarque

Les lemmes de l'étoile énoncent des propriétés vérifiées par les langages réguliers, mais les propriétés décrites par ces lemmes ne sont pas des propriétés *caractéristiques* des langages réguliers, c'est-à-dire que certains langages peuvent vérifier ces propriétés sans être réguliers.
Par contre si un langage ne possède pas ces propriétés alors on peut déduire qu'il n'est pas régulier. C'est dans ce sens que l'on utilise souvent les lemmes de l'étoile.

Exemple d'utilisation du lemme de l'étoile

Soit le langage $L = \{a^i b^i,\ i \in \mathbb{N}\}$, montrons que L n'est pas un langage régulier en appliquant le lemme de l'étoile (version 2).

Soit p l'entier dont il est question dans le lemme de l'étoile, considérons le mot $a^p b^p$ et le facteur $\beta = a^p$, il existe une factorisation $\beta = uvw$ avec $v \neq \varepsilon$ cette factorisation est de la forme $a^q a^r a^s$ avec $q + r + s = p$, $r > 0$ et pour tout n de $\mathbb{N}$ $a^q (a^r)^n a^s b^p \in L$, cette dernière condition ne peut être vérifiée puisque par exemple pour $n = 2$, $q + 2r + s \neq p$. L n'est donc pas un langage régulier.

Proposition

Si L et L' sont des langages algébriques alors $L \cup L'$, LL', L^* sont des langages algébriques.

Remarques

- Pour montrer la proposition il suffit de construire des grammaires algébriques engendrant les langages $L \cup L'$, LL' et L^* à partir de grammaires engendrant L et L' (à voir en TD).
- L'intersection de deux langages algébriques n'est généralement pas un langage algébrique. De même le complémentaire d'un langage algébrique n'est généralement pas un langage algébrique.
- On peut montrer que l'intersection d'un langage régulier et d'un langage algébrique est un langage algébrique. (Mathématiques Discrètes. Pierre Marchand. Dunod).