Langages et Automates
Langages réguliers et Grammaires

Engel Lefaucheux

Prépas des INP

Rappels et exercices

- $L_1 \cdot L_2 = \{w_1 \cdot w_2 \mid w_1 \in L_1 \text{ et } w_2 \in L_2\}$
- $L_1^n = \underbrace{L_1 \cdot L_1 \ldots L_1 \cdot L_1}_{n \text{ times}}$
- $L^* = L^0 \cup L^1 \cup L^2 \cup L^3 \cup \ldots$

Les mots $\{\varepsilon, a, babar\}$ appartiennent-ils aux langages suivants

- $(a + b)^* \cdot r$
- $((a + b)^* \cdot r)^*$
- $((aa + bb + ab)^* \cdot r)^*$

Lemma

Deux mots u et v avec $|u| = |v|$ commutent si et seulement si $u = v$.

Expression régulière pour la machine à café

Rappelons l'exemple de la machine à café :

- Si à l'arrêt, on peut cliquer sur un bouton pour l'activer.
- En cours de fonctionnement, la machine fait du bruit pendant une durée aléatoire.
- Si la machine a été activé, elle va éventuellement produire du café et s'éteindre.

Quel expression régulière représente son comportement ?

Encore une modélisation

Un système de contrôle d'accès à un bâtiment fonctionne selon les règles suivantes :

- Une personne commence par badger son badge pour s'identifier.
- Si le badge est valide, elle peut entrer immédiatement. Sinon, elle doit effectuer une validation manuelle auprès du personnel.
- Après validation (automatique ou manuelle), elle doit ouvrir la porte pour accéder au bâtiment.
- La porte peut rester ouverte temporairement pour d'autres personnes déjà validées, mais elle finit toujours par se refermer automatiquement.

Modélisez les séquences possibles d'interactions avec ce système en utilisant une expression régulière.

Le cas des regexp

En pratique (sur ordinateur)

- l'alphabet est implicite (tous les caractères)
- raccourcis syntaxiques:

`.`	pour un caractère quelconque
`[a-z]`	pour $a\|b\|c\|\ldots\|z$
`[^abc]`	pour un caractère autre que `a`, `b` ou `c`
`\s`	pour les caractères d'espacement
`a?`	pour $\epsilon\|a$ (au plus un `a`)
`a+`	pour aa^* (au moins un `a`)
`e{50}`	pour $\mathcal{L}(e)^{50}$
`^` et `$`	pour début et fin de ligne

- possibilité de nommage de blocs et substitution de texte

Exemples

- `[0-9]{10}` dénote l'ensemble des numéros de téléphone
- `[a-z]+@[a-z]+[.][a-z]{3}` dénote des courriels
- `<[^>]*>` dénote les balises html (e.g. `<h1 class="first">`)

- https://regex101.com/

Outline

1 Critères de régularité

2 Construire une grammaire et son langage

3 Type de grammaires

Une règle d'or

Si des relations existent entre les exposants apparaissant dans la description du langage, alors celui-ci n'est pas régulier.

- $\{a^n b^n \mid n \in \mathbb{N}\}$
- $\{a^n b^m c^k \mid n, m, k \in \mathbb{N} \wedge k \geq n + m\}$

ne sont pas réguliers.

Plusieurs critères formels de non-régularité

- Théorème de Myhill-Nerode (complexe)
- Lemme de l'étoile (simple, mais ne marche pas tout le temps)

Lemme de l'étoile

Theorem

Soit L un langage régulier. Il existe un entier N tel que tout mot w de L de longueur $|w| \geq N$ possède une factorisation $w = xyz$ avec $0 < |y|$ telle que

1. $0 < |xy| \leq N$ *et*
2. $xy^n z \in L$ *pour tout entier* $n \geq 0$.

Lemme de l'étoile

Theorem

Soit L un langage régulier. Il existe un entier N tel que tout mot w de L de longueur $|w| \geq N$ possède une factorisation $w = xyz$ avec $0 < |y|$ telle que

1. $0 < |xy| \leq N$ *et*
2. $xy^n z \in L$ *pour tout entier* $n \geq 0$.

Quid de $\{a^n b^n \mid n \in \mathbb{N}\}$?

Exercice

Les langages suivants sont-ils réguliers ?

- $\{a^n b^m \mid n \geq m\}$
- $\{a^n \mid n$ est un nombre premier$\}$
- $(ab)^* \cap \{w \mid |w|_a = |w|_b\}$
- $ab(a+b)^* \cap \{w \mid |w|_a = |w|_b\}$

Theorem

Soit L un langage régulier. Il existe un entier N tel que tout mot w de L de longueur $|w| \geq N$ possède une factorisation $w = xyz$ avec $0 < |y|$ telle que

1. $0 < |xy| \leq N$ *et*
2. $xy^n z \in L$ *pour tout entier* $n \geq 0$.

Dépasser les expressions régulières

Les langages réguliers reconnaissent

- des mots issus de lexiques
- des structures simples.

Ils sont insuffisants pour

- les structures de type $\{a^n b^n \mid n \in \mathbb{N}\}$
- le langage naturel
- l'analyse de programmes

Outline

1 Critères de régularité

2 Construire une grammaire et son langage

3 Type de grammaires

Définition formelle

Une grammaire est un quadruplet (T, N, R, S)

- T : symboles terminaux
  $\longrightarrow$ l'alphabet du langage que nous voulons créer
- N : symboles non-terminaux / temporaires
  $\longrightarrow$ symboles utilisés au cours de la dérivation
- R : ensemble des règles de la dérivation
- S : Axiome
  $\longrightarrow$ symbole de départ

$(\{a, b\}, \{S, A\}, R, S)$ où R est l'ensemble de règles suivant

$S \rightarrow AA$

$A \rightarrow AAA \mid b\ A \mid A\ b \mid a$

Arbre de dérivation

Représentation graphique de la dérivation

- Racine = axiome
- Noeud = Symbole non-terminal
- Feuille = symbole terminal
- Relation parent-enfant = règle

Arbre de dérivation

Représentation graphique de la dérivation

- Racine = axiome
- Noeud = Symbole non-terminal
- Feuille = symbole terminal
- Relation parent-enfant = règle

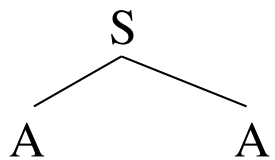
$S \rightarrow AA$
$A \rightarrow AAA \mid b\ A \mid A\ b \mid a$

Arbre de dérivation

Représentation graphique de la dérivation

- Racine = axiome
- Noeud = Symbole non-terminal
- Feuille = symbole terminal
- Relation parent-enfant = règle

$S \rightarrow AA$
$A \rightarrow AAA \mid b\ A \mid A\ b \mid a$

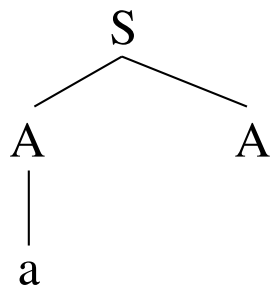


Arbre de dérivation

Représentation graphique de la dérivation

- Racine = axiome
- Noeud = Symbole non-terminal
- Feuille = symbole terminal
- Relation parent-enfant = règle

$S \rightarrow AA$
$A \rightarrow AAA \mid b\ A \mid A\ b \mid a$



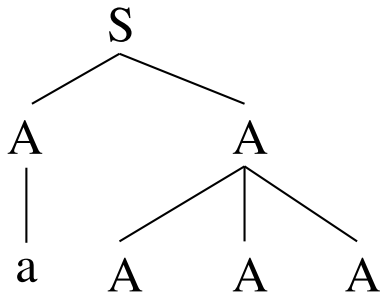
Arbre de dérivation

Représentation graphique de la dérivation

- Racine = axiome
- Noeud = Symbole non-terminal
- Feuille = symbole terminal
- Relation parent-enfant = règle

$S \rightarrow AA$

$A \rightarrow AAA \mid b\ A \mid A\ b \mid a$



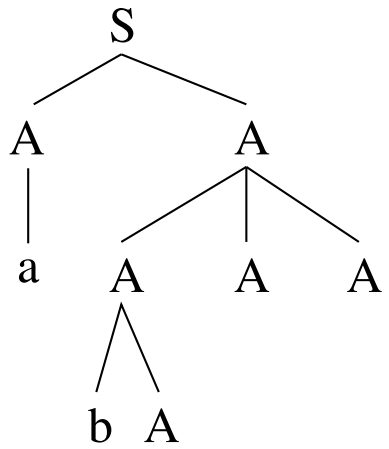
Arbre de dérivation

Représentation graphique de la dérivation

- Racine = axiome
- Noeud = Symbole non-terminal
- Feuille = symbole terminal
- Relation parent-enfant = règle

$S \rightarrow AA$

$A \rightarrow AAA \mid b\ A \mid A\ b \mid a$



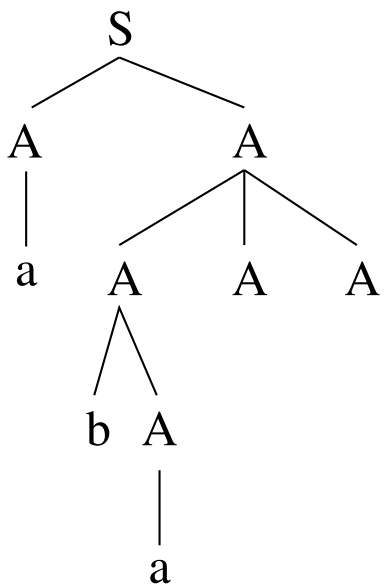
Arbre de dérivation

Représentation graphique de la dérivation

- Racine = axiome
- Noeud = Symbole non-terminal
- Feuille = symbole terminal
- Relation parent-enfant = règle

$S \rightarrow AA$

$A \rightarrow AAA \mid b\ A \mid A\ b \mid a$



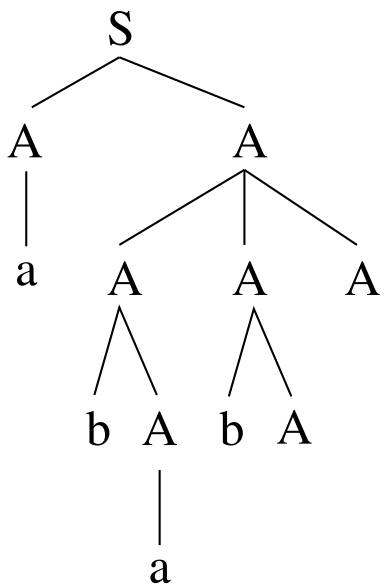
Arbre de dérivation

Représentation graphique de la dérivation

- Racine = axiome
- Noeud = Symbole non-terminal
- Feuille = symbole terminal
- Relation parent-enfant = règle

$S \rightarrow AA$

$A \rightarrow AAA \mid b\ A \mid A\ b \mid a$



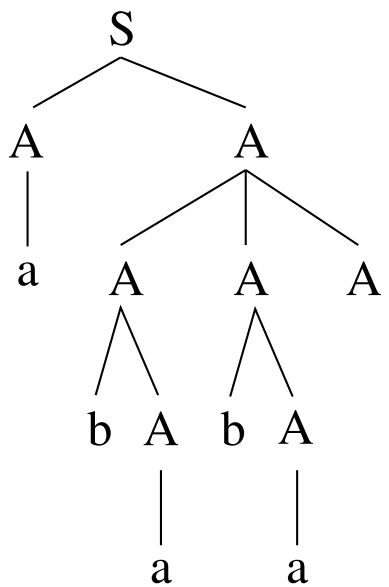
Arbre de dérivation

Représentation graphique de la dérivation

- Racine = axiome
- Noeud = Symbole non-terminal
- Feuille = symbole terminal
- Relation parent-enfant = règle

$S \rightarrow AA$

$A \rightarrow AAA \mid b\ A \mid A\ b \mid a$



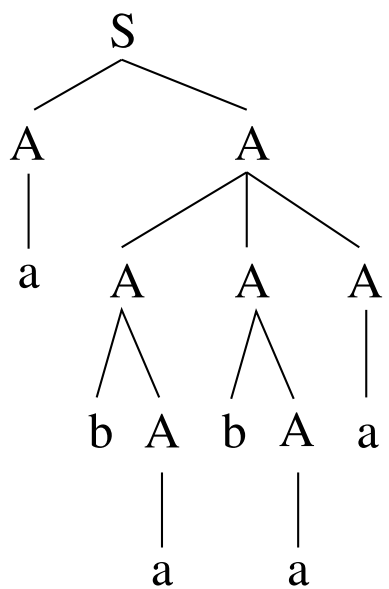
Arbre de dérivation

Représentation graphique de la dérivation

- Racine = axiome
- Noeud = Symbole non-terminal
- Feuille = symbole terminal
- Relation parent-enfant = règle

$S \rightarrow AA$

$A \rightarrow AAA \mid b\ A \mid A\ b \mid a$



Petit exemple de grammaire

On veut construire une PHRASE simple.

$$
\begin{aligned}
\mathit{PHRASE} &\rightarrow \mathit{SUJET}\ \ \mathit{VERBE}\ \ \mathit{COMPLEMENT} \\
\mathit{SUJET} &\rightarrow \mathit{ludovic} \mid \mathit{pierre} \mid \mathit{nicolas} \\
\mathit{VERBE} &\rightarrow \mathit{mange} \mid \mathit{porte} \\
\mathit{COMPLEMENT} &\rightarrow \mathit{ARTICLE}\ \ \mathit{NOM}\ \ \mathit{ADJECTIF} \\
\mathit{ARTICLE} &\rightarrow \mathit{un} \mid \mathit{le} \\
\mathit{NOM} &\rightarrow \mathit{livre} \mid \mathit{plat} \mid \mathit{chat} \\
\mathit{ADJECTIF} &\rightarrow \mathit{delicieux} \mid \mathit{rouge} \mid \mathit{doux}
\end{aligned}
$$

On part de PHRASE, et on remplace les termes en grâce aux règles

Petit exemple de grammaire

Construction d'une *IFSTRUC*

$$
\begin{aligned}
IFSTRUC &\rightarrow if\ \ TEST\ \ then\ BLOCK\ \ else\ BLOCK \\
TEST &\rightarrow VAR\ \leq\ INT \mid TEST\ \ et\ \ TEST \\
BLOCK &\rightarrow INSTANCE \mid INSTANCE\ \ BLOCK \\
VAR &\rightarrow x \mid y \\
INT &\rightarrow 0 \mid 1 \mid INT\ \ INT \\
INSTANCE &\rightarrow incr\ \ VAR \mid decr\ \ VAR \mid IFSTRUC \mid return\ \ VAR
\end{aligned}
$$

Exercices

Quel langage pour les grammaires suivantes commençant par S

1. $S \to \varepsilon \mid T$
   $T \to a\ b \mid a\ T\ b$
2. $S \to \varepsilon \mid A$
   $A \to a\ B\ C \mid a\ A\ B\ C$
   $C\ B \to B\ C$
   $A\ B \to A\ b$
   $a\ B \to A\ b$
   $B\ B \to B\ b$
   $C \to c$

Construisez une grammaire pour le langage des palindromes sur $\Sigma = \{a, b\}$

Outline

1 Critères de régularité

2 Construire une grammaire et son langage

3 Type de grammaires

Hiérarchie de Chomsky

Type 0 Pas de restrictions sur les règles

Type 1 règles de la forme
$u\ A\ v \rightarrow u\ w\ v$ avec $A \in N$ et $u, v, w \in (N \cup T)^*$

Type 2 règles de la forme
$A \rightarrow w$ avec $A \in N$ et $w \in (N \cup T)^*$

Type 3 Toutes les règles sont soit de la forme
$A \rightarrow a\ B$ ou $A \rightarrow a$ (grammaire à droite)
soit de la forme
$A \rightarrow B\ a$ ou $A \rightarrow a$ (grammaire à gauche)

Hiérarchie de Chomsky

Type 0 Pas de restrictions sur les règles

Type 1 règles de la forme
$u\ A\ v \rightarrow u\ w\ v$ avec $A \in N$ et $u, v, w \in (N \cup T)^*$

Type 2 règles de la forme
$A \rightarrow w$ avec $A \in N$ et $w \in (N \cup T)^*$

Type 3 Toutes les règles sont soit de la forme
$A \rightarrow a\ B$ ou $A \rightarrow a$ (grammaire à droite)
soit de la forme
$A \rightarrow B\ a$ ou $A \rightarrow a$ (grammaire à gauche)

Grammaire de type 0 $\Longleftrightarrow$ Machine de Turing

Hiérarchie de Chomsky

Type 0 Pas de restrictions sur les règles

Type 1 règles de la forme
$u\ A\ v \rightarrow u\ w\ v$ avec $A \in N$ et $u, v, w \in (N \cup T)^*$

Type 2 règles de la forme
$A \rightarrow w$ avec $A \in N$ et $w \in (N \cup T)^*$

Type 3 Toutes les règles sont soit de la forme
$A \rightarrow a\ B$ ou $A \rightarrow a$ (grammaire à droite)
soit de la forme
$A \rightarrow B\ a$ ou $A \rightarrow a$ (grammaire à gauche)

Grammaire de type 0 $\Longleftrightarrow$ Machine de Turing
Grammaire de type ? $\Longleftrightarrow$ Expression régulière

Hiérarchie de Chomsky

Type 0 Pas de restrictions sur les règles

Type 1 règles de la forme
$u\ A\ v \rightarrow u\ w\ v$ avec $A \in N$ et $u, v, w \in (N \cup T)^*$

Type 2 règles de la forme
$A \rightarrow w$ avec $A \in N$ et $w \in (N \cup T)^*$

Type 3 Toutes les règles sont soit de la forme
$A \rightarrow a\ B$ ou $A \rightarrow a$ (grammaire à droite)
soit de la forme
$A \rightarrow B\ a$ ou $A \rightarrow a$ (grammaire à gauche)

Grammaire de type 0 $\Longleftrightarrow$ Machine de Turing
Grammaire de type 3 $\Longleftrightarrow$ Expression régulière

Exercices

Représenter les langages suivant avec une grammaire de type 3

- $baab^*$
- $b(aab)^*$

De quel type est la grammaire
$S \to aU \mid c$
$U \to Sb \mid d$

Quel est son langage ?

L_1 et L_2 langages de grammaire G_1 et G_2
Informellement, comment construire une grammaire pour
$L_1 \cup L_2$, $L_1 \cdot L_2$ et L_1^*

Exercices

Représenter les langages suivant avec une grammaire de type 3

- $baab^*$
- $b(aab)^*$

De quel type est la grammaire
$S \to aU \mid c$
$U \to Sb \mid d$

Quel est son langage ? $\{a^n c b^n, a^{n+1} d b^n \mid n \in \mathbb{N}\}$

L_1 et L_2 langages de grammaire G_1 et G_2
Informellement, comment construire une grammaire pour
$L_1 \cup L_2$, $L_1 \cdot L_2$ et L_1^*