UFR d'Informatique
L3 Informatique et Math-Info

Grammaires et analyse syntaxique
Année 2021-2022

TD n°5

LL(1), le retour

Dans tous les exercices qui suivent, les minuscules sont les terminaux, les majuscules sont les non-terminaux.

Exercice 1 *On considère la grammaire suivante :*

$Z \to S\#$
$S \to D \mid XA \mid X \mid \epsilon$
$X \to bX \mid YVWV$
$Y \to aX \mid \epsilon$
$W \to c \mid d$
$V \to v \mid \epsilon$
$D \to DE$
$E \to e \mid Ee$
$F \to f$

On considère deux méthodes pour réduire la grammaire :

1. *On détermine les non-terminaux non-productifs. On les enlève de la grammaire. On détermine les non-terminaux non-accessibles de la grammaire ainsi obtenu et on les enlève.*
2. *On détermine les non-terminaux non-accessibles. On les enlève de la grammaire. On détermine les non-terminaux non-productifs de la grammaire ainsi obtenu et on les enlève.*

— *Appliquer les deux méthodes.*
— *Avec laquelle des deux méthodes on obtient une grammaire réduite ?*

Pour la suite on considère la grammaire réduite obtenue.

— *Calculer l'ensemble de non-terminaux annulables* EPS.
— *Calculer l'ensemble* FIRST_1 *de chaque non-terminal.*
— *Calculer l'ensemble* FOLLOW_1 *de chaque non-terminal.*
— *Est-ce que la grammaire est* LL(1) *?*

Exercice 2 *On considère la grammaire suivante :*

$E \to E \vee T \mid T$
$T \to T \wedge F \mid F$
$F \to m \mid (E)$

— *Existe-t-il un k tel que cette grammaire soit* LL(k) *? Pourquoi ?*
— *Donner une grammaire LL(1) qui génère le même langage et montrer qu'elle est LL(1).*

Exercice 3 *On considère la grammaire suivante :*

$Z \to S\#$
$S \to X \mid Yc \mid aL \mid T$
$X \to a \mid \epsilon$
$L \to Ua$
$U \to aXaLb$
$Y \to Sb \mid d \mid \epsilon$
$R \to ax \mid Yb$
$T \to XYX$

— *Donner les non-terminaux productifs et les non-terminaux accessibles.*
— *Est-ce que la grammaire est réduite ? si non, la réduire.*
— *Donner les non-terminaux effaçables.*

Exercice 4 *On considère la grammaire suivante :*

$Z \to S\#$
$S \to X \mid Yc$
$X \to a \mid \epsilon$
$Y \to Sb \mid d$

Justifier qu'elle n'est LL(k) *pour aucun* k.

Exercice 5 *(Facultatif) Le langage des palindromes sur* $\{0, 1\}$ *peut-il être engendré par une grammaire* LL(1) *? Justifier.*

Exercice 6 *On souhaite construire un analyseur grammatical des expressions arithmétiques (avec − et +) bien parenthésées engendré par la grammaire*

$S \to n \mid (S) \mid S + S \mid S - S$

1. *Donner une grammaire* LL(1) *avec axiome* S_1 *pour le langage qu'on appellera* L_1
2. *Ajouter la possibilité de faire des opérations avec des variables 'v'. (On appellera* L_2 *ce langage)*
3. *A présent on veut définir le langage* L_3 *"let* $v = a_1$ *and* $v = a_2$ *… and* $v = a_k$ *in* b*" avec les* a_i *dans* L_1*, et* b *dans* L_2*. (On appelle ce langage* L_3*)*
4. *Faire la table d'analyse (c'est-à-dire un tableau avec les non-terminaux en ordonné et les terminaux en abscisse), qui indique à chaque fois quelle règle on est censé appliquer.*
5. *Modifier le fichier* `parser.ml` *pour qu'il analyse la grammaire établie.*