

TD 13 : Grammaires $LL(k)$

Exercice 1 (Motifs Objective Caml). Voici un extrait de la syntaxe des motifs d'Objective Caml :

$$\begin{aligned}
 \langle pat \rangle &\rightarrow VN \\
 &| - \\
 &| \langle pat \rangle :: \langle pat \rangle \\
 &| \langle pat \rangle \text{ as } VN \\
 &| [\langle patl \rangle] \\
 \langle patl \rangle &\rightarrow \langle pat \rangle \\
 &| \langle patl \rangle ; \langle pat \rangle
 \end{aligned}$$

Cette grammaire permet de générer des motifs tels que

`[a; _; b] :: t as l`

1. Donner une grammaire $SLL(1)$ équivalente à ce fragment.
2. Préciser les ensembles $First_1$ et $Follow_1$ calculés.
3. Écrire un analyseur récursif descendant dans votre langage favori pour votre grammaire.

Exercice 2 (Calcul efficace de $First_1$ et $Follow_1$). On appelle *expression relationnelle* un terme décrit par la syntaxe abstraite

$$e ::= R \mid e^* \mid e^{-1} \mid e \cdot e \mid e \cup e$$

où R est une relation "atomique". Une expression relationnelle définit une relation $R(e)$. La *taille* $|e|$ d'une expression relationnelle e est la somme des tailles des relations atomiques qui la composent. Pour une expression relationnelle e de taille finie, on peut calculer une image par sa relation $R(e)$ en temps $O(|e|)$. On souhaite calculer les ensembles $First_1$ et $Follow_1$ d'une grammaire G à l'aide d'expressions relationnelles de taille $O(|G|)$.

1. Proposer une expression relationnelle `first` sur $N \times \Sigma$ de taille $O(|G|)$ telle que

$$First_1(A) = R(\text{first})(A) \cup \{\varepsilon \mid A \Rightarrow^* \varepsilon\}.$$

2. Proposer une expression relationnelle `follow` sur $N \times \Sigma$ telle que

$$Follow_1(A) = R(\text{follow})(A) \cup \{\varepsilon \mid \exists \delta \in (N \uplus \Sigma)^*, S \Rightarrow^* \delta A\}.$$

Au moins dans un premier temps, on pourra se contenter d'une expression de taille $O(|G|^2)$.

Exercice 3 (Langages $LL(k)$).

1. Montrer qu'une grammaire est $LL(1)$ si et seulement si elle est $SLL(1)$.
2. Montrer que si une grammaire est $LL(k)$, alors il existe une grammaire $SLL(k)$ équivalente.
3. Appliquer cette construction à la grammaire

$$S \rightarrow aAab \mid bAb$$

$$A \rightarrow cAB \mid \varepsilon \mid a$$

$$B \rightarrow \varepsilon$$

Il existe une hiérarchie stricte des langages $LL(k)$ pour chaque valeur de k . Il existe aussi des langages déterministes qui ne sont pas $LL(k)$.

Exercice 4 (Récursivité gauche, GNF, langages simples). Un symbole non terminal $A \in N$ est *récuratif gauche* s'il existe une dérivation $A \Rightarrow^+ A\alpha$ avec α dans $(N \cup \Sigma)^*$. Une grammaire réduite est récurative gauche s'il existe au moins un non terminal A récuratif gauche.

1. Montrer que si une grammaire réduite est récurative gauche, alors elle n'est pas LL .
2. Montrer que si une grammaire est simple, alors elle est $SLL(1)$.
3. Montrer que si une grammaire sans ε -production est $LL(k)$, alors on peut construire une grammaire $LL(k)$ équivalente sous forme normale de GREIBACH.
4. Montrer que si une grammaire sans ε -production est $LL(1)$, alors elle engendre un langage simple.
5. Donner une grammaire $SLL(1)$ qui n'engendre pas un langage simple.