Université de Paris
Automates et Analyse Lexicale

UFR Informatique
Année 2021–2022

Handout 5

L'algorithme de Glushkov

1 Algorithme de Glushkov

Cet algorithme transforme une expression rationnelle directement en un automate non-déterministe, sans de produire des ϵ-transitions.

1.1 Linéariser l'expression rationnelle

Une expression rationnelle r sur l'alphabet Σ est *linéaire* quand chaque symbole de Σ paraît au plus une fois dans r. Si une expression rationnelle n'est pas linéaire on peut la linéariser en passant à un alphabet *plus grand*, appelé l'*alphabet linéarisé*. Le plus simple est de numéroter toutes les occurrences des lettres dans r à partir de 1.

Exemple : $(ab+b)^*(bb+a^*)$ sur l'alphabet $\{a,b\}$ devient $(a_1b_2+b_3)^*(b_4b_5+a_6^*)$ sur l'alphabet linéarisé $\{a_1,b_2,b_3,b_4,b_5,a_6)$.

1.2 Analyser l'expression rationnelle

Calculer les informations suivantes sur l'expression rationnelle r :

- est-ce que $\epsilon \in \mathcal{L}(r)$?
- $first(r) = \{a \in \Sigma \mid \exists w \in \Sigma^* : aw \in \mathcal{L}(r)\}$
- $last(r) = \{a \in \Sigma \mid \exists w \in \Sigma^* : wa \in \mathcal{L}(r)\}$
- $next(r) = \{f \in \Sigma^2 \mid \exists v, w \in \Sigma^* : v \cdot f \cdot w \in \mathcal{L}(r)\}$

Souvent on se contente de dire qu'on peut "voir" ces informations. Des définitions formelles sont données à la section 1.6

1.3 Construire un automate déterministe

L'automate est défini comme suit :

- l'alphabet Σ comme donnée pour l'expression rationnelle (linéarisée !)
- ensemble d'états $Q = \Sigma \cup \{0\}$
- état initial $q_0 = 0$
- états acceptants : $F = last(r)$, plus l'état 0 dans le cas où $\epsilon \in \mathcal{L}(r)$
- transitions :

$$\delta(q,b) = \begin{cases} b & \text{si } q = 0, b \in \mathit{first}(r) \\ b & \text{si } q = a, ab \in \mathit{next}(r) \end{cases}$$

Il s'agit donc d'un automate déterministe et possiblement incomplet sur l'alphabet linéarisé.

1.4 Revenir à l'alphabet original

Remplacer dans Q, F, δ les symboles de l'alphabet linéarisé par les symboles d'origine. On obtient un automate *non*-déterministe.

1.5 Déterminiser

Comme vu au cours 3.

1.6 Définition formelle de l'analyse de l'expression rationnelle

- La fonction *eps*: $\textsc{Rat} \to \{\texttt{true}, \texttt{false}\}$

$$
\begin{array}{rcll}
\mathit{eps}(\epsilon) & = & \texttt{true} & \\
\mathit{eps}(a) & = & \texttt{false} & a \in \Sigma \\
\mathit{eps}(\emptyset) & = & \texttt{false} & \\
\mathit{eps}(r_1 + r_2) & = & \mathit{eps}(r_1) \vee \mathit{eps}(r_2) & \\
\mathit{eps}(r_1 r_2) & = & \mathit{eps}(r_1) \wedge \mathit{eps}(r_2) & \\
\mathit{eps}(r^*) & = & \texttt{true} &
\end{array}
$$

- La fonction *first*: $\textsc{Rat} \to 2^{\Sigma}$

$$
\begin{array}{rcll}
\mathit{eps}(\epsilon) & = & \emptyset & \\
\mathit{first}(a) & = & \{a\} & a \in \Sigma \\
\mathit{first}(\emptyset) & = & \emptyset & \\
\mathit{first}(r_1 + r_2) & = & \mathit{first}(r_1) \cup \mathit{first}(r_2) & \\
\mathit{first}(r_1 r_2) & = & \left\{ \begin{array}{ll} \mathit{first}(r_1) & \text{si } \neg \mathit{eps}(r_1) \\ \mathit{first}(r_1) \cup \mathit{first}(r_2) & \text{si } \mathit{eps}(r_1) \end{array} \right. & \\
\mathit{first}(r^*) & = & \mathit{first}(r) &
\end{array}
$$

- La fonction *last*: $\textsc{Rat} \to 2^{\Sigma}$

$$
\begin{array}{rcll}
\mathit{last}(\epsilon) & = & \emptyset & \\
\mathit{last}(a) & = & \{a\} & a \in \Sigma \\
\mathit{last}(\emptyset) & = & \emptyset & \\
\mathit{last}(r_1 + r_2) & = & \mathit{last}(r_1) \cup \mathit{last}(r_2) & \\
\mathit{last}(r_1 r_2) & = & \left\{ \begin{array}{ll} \mathit{last}(r_2) & \text{si } \neg \mathit{eps}(r_2) \\ \mathit{last}(r_2) \cup \mathit{last}(r_1) & \text{si } \mathit{eps}(r_2) \end{array} \right. & \\
\mathit{last}(r^*) & = & \mathit{last}(r) &
\end{array}
$$

- La fonction *next*: $\textsc{Rat} \to 2^{(\Sigma^2)}$

$$
\begin{array}{rcll}
\mathit{next}(\epsilon) & = & \emptyset & \\
\mathit{next}(a) & = & \emptyset & a \in \Sigma \\
\mathit{next}(\emptyset) & = & \emptyset & \\
\mathit{next}(r_1 + r_2) & = & \mathit{next}(r_1) \cup \mathit{next}(r_2) & \\
\mathit{next}(r_1 r_2) & = & \mathit{next}(r_1) \cup \mathit{next}(r_2) \cup \mathit{last}(r_1)\mathit{first}(r_2) & \\
\mathit{next}(r^*) & = & \mathit{next}(r) \cup \mathit{last}(r)\mathit{first}(r) &
\end{array}
$$