Conversion d'une expression régulière en automate déterministe

Léo Gayral

2017-2018

ref : Aho, Ullman – Compilateurs, Principes, techniques et outils – p.160

Remarque 1. Une façon théorique de montrer la réduction d'une expression rationnelle en un *automate fini déterministe* (AFD) consiste à convertir l'expression en AFN *avec ϵ-transitions*, d'éliminer ces transitions puis de considérer le déterminisé de l'AFN obtenu. La taille de cet AFD étant exponentielle en celle de l'expression initiale, l'algorithme de conversion induit est naturellement exponentiel en temps.

Malheureusement, en toute généralité, cette borne est optimale. En effet, si on considère par exemple l'expression $(a|b)^*a(a|b)^{n-1}$, un mot w est accepté lorsque $|w| \geq n$ et que la n-ième lettre en partant de la fin est un a. Un état de l'automate doit donc garder une trace des n dernières lettres lues. On considère l'automate défini par $Q = (a|b)^n$, $q_0 = b^n$, $F = a(a|b)^{n-1}$ et muni des transitions $\delta : Q \times (a|b) \to Q$ suivantes :

$$\forall w \in (a|b)^{n-1}, \forall x, y \in (a|b), \delta(xw, y) = wy\,.$$

On peut montrer que cet automate est minimal pour le langage considéré, d'où une borne inférieure exponentielle en la taille de l'expression rationnelle.

En pratique, sur un mot w, on va au plus visiter $|w|$ états de l'automate. On aimerait donc pouvoir générer un AFD partiel, au fur et à mesure de la lecture de w, en temps linéaire.

Quitte à garder en mémoire les transitions déjà calculées, cela permettrait d'amortir le coût de génération de l'AFD au cours de son utilisation effective, et de limiter la complexité spatiale sur des petits exemples.

Théorème 1. Soit E une expression rationnelle. Sans perte de généralité, quitte à ajouter un nouveau symbole # à l'alphabet, on considère par la suite l'expression $E\#$.

Après un prétraitement sur l'arbre syntaxique d'une telle expression rationnelle, on peut simuler l'AFD visité par un mot w en temps $O\left(|w| \times |E|^2\right)$.

Idée de l'algorithme.
L'idée générale de la preuve est de construire tous les outils permettant décrire l'AFN induit, mais sans jamais le construire explicitement, en se déplaçant à la volée au sein des parties de cet automate.

On commence par définir plusieurs fonctions sur les sous-arbres syntaxiques, par induction depuis les feuilles énumérées par $(x_i)_{1\leq i\leq r}$:

T	$Nul(T)$	$Fst(T)$
$x_i \in \Sigma$	$\bot$	$\{i\}$
$(*, T)$	$\top$	$Fst(T)$
$(\vert, T_1, T_2)$	$Nul\left(T_1\right) \vee Nul\left(T_2\right)$	$Fst\left(T_1\right) \cup Fst\left(T_2\right)$
$(\bullet, T_1, T_2)$	$Nul\left(T_1\right) \wedge Nul\left(T_2\right)$	si $Nul\left(T_1\right)$ alors $Fst\left(T_1\right) \cup Fst\left(T_2\right)$ sinon $Fst\left(T_1\right)$

où Nul détermine si ϵ est dans le langage associé à l'expression rationnelle induite par le sous-arbre T, et Fst la liste des feuilles correspondant à la première lettre d'un mot du langage. On définit également Lst – correspondant à la dernière lettre d'un mot – sensiblement comme Fst, la seule différence étant la dernière ligne, où :

$$Lst(T) = \text{si } Nul\left(T_2\right) \text{ alors } Lst\left(T_1\right) \cup Lst\left(T_2\right) \text{ sinon } Lst\left(T_2\right) .$$

On veut maintenant définir $Suiv(i) \subset [\![1, r]\!]$ comme l'ensemble des feuilles de l'arbre pouvant correspondre à la lettre suivante d'un mot du langage. Pour avoir $j \in Suiv(i)$, pour pouvoir aller directement de x_i à x_j au sein de l'expression rationnelle, il faut nécessairement les relier en avançant dans une concaténation, ou en revenant en arrière dans une étoile. Autrement dit :

$$Suiv(i) = \bigcup_{\substack{(\bullet, T_1, T_2) \\ i \in Lst(T_1)}} Fst\left(T_2\right) \bigcup_{\substack{(*, T) \\ i \in Lst(T)}} Fst(T) .$$

Ces étapes correspondent au pré-traitement annoncé, qu'on peut faire en $O\left(|E|^2\right)$ en temps et en espace.

On peut enfin simuler l'AFD sur les états $Q = \mathcal{P}(\llbracket 1, r \rrbracket)$, en partant de $q_0 = Fst(T)$ (T désigne ici l'arbre syntaxique complet), en terminant dans $F = \{P \in Q, r \in P\}$ (avec $x_r = \#$), selon les transitions définies par :

$$\delta(P, c) = \bigcup_{i \in P, x_i = c} Suiv(i)$$

et en majorant $|P| = O(|E|)$ et $|Suiv(i)| = O(|E|)$, on peut garantir que le calcul de $\delta(P, c)$ aboutit toujours en $O\left(|E|^2\right)$ opérations. □