DM2

Le théorème de Kleene affirme l'équivalence entre reconnaissabilité et rationalité. Il existe des preuves constructives de ce théorème. Une telle preuve fournit un algorithme associant à un automate $\mathcal{A}$ une expression régulière e qui décrit le langage reconnu par $\mathcal{A}$, ou, réciproquement, associant à une expression régulière e un automate $\mathcal{A}$ qui reconnaît le langage décrit par e.

Cette deuxième construction, d'utilisation fréquente, est l'objet de ce problème. La méthode la plus connue et aussi la plus facile à expliquer, repose sur l'utilisation des automates de Thompson, publiée en 1968. Bien plus tôt, R. McNaughton et H. Yamada (1960) et V. M. Glushkov (1961) avaient, indépendamment, publié une méthode d'apparence très différente : elle consiste à numéroter les occurrences de chaque lettre de l'expression régulière pour obtenir une nouvelle *expression dite linéaire*. La construction de l'automate fini associé est alors aisée et se termine en *dénumérotant* les états puis en déterminisant l'automate.

Gérard Berry et Ravi Sethi ont donné en 1986 une présentation rigoureuse de cette méthode, et mis en évidence sa relation avec les dérivées d'une expression régulière (notion due à Janus Brzozowski). Jean Berstel et Jean-Éric Pin ont proposé en 1995 une interprétation très simple de cette méthode, en faisant appel aux *langages locaux*.

Préliminaires

Soit L un langage sur l'alphabet Σ. On note $P(L)$ l'ensemble des préfixes de longueur 1 des mots de L, $S(L)$ l'ensemble des suffixes de longueur 1 des mots de L et $F(L)$ l'ensemble des facteurs de longueur 2 des mots de L. On a donc :

$$\begin{aligned} x \in P(L) &\iff x \in \Sigma \text{ et } x\Sigma^* \cap L \neq \emptyset \\ x \in S(L) &\iff x \in \Sigma \text{ et } \Sigma^* x \cap L \neq \emptyset \\ x \in F(L) &\iff x \in \Sigma^2 \text{ et } \Sigma^* x \Sigma^* \cap L \neq \emptyset \end{aligned}$$

On définit également $N(L) = \Sigma^2 \setminus F(L)$.

Soit $\mathcal{A}$ un automate fini. On note $L_{\text{af}}(\mathcal{A})$ le langage reconnu par $\mathcal{A}$. De même, si e est une expression régulière, on note $L_{\text{reg}}(e)$ le langage décrit par e. Soient X et Y deux alphabets. Un *morphisme* de X^* vers Y^* est une application $\phi : X^* \mapsto Y^*$ vérifiant $\phi(uv) = \phi(u)\phi(v)$ quels que soient les mots u et v pris dans X^*. Un tel morphisme est parfaitement défini par les images des éléments de X. Un automate fini déterministe $\mathcal{A} = (\Sigma, Q, q_0, F, \delta)$ est dit *standard* si aucune de ses transitions ne mène à l'état initial q_0. L'objectif de ce problème est de décrire un algorithme calculant une fonction A qui, à une expression régulière e, associe un automate fini $\mathcal{A}$ tel que $L_{\text{af}}(\mathcal{A}) = L_{\text{reg}}(e)$. On aura donc $L_{\text{af}} \circ A = L_{\text{reg}}$.

Question 1. Soit $\mathcal{A} = (\Sigma, Q, q_0, F, \delta)$ un automate fini déterministe. Décrire un algorithme construisant un automate fini déterministe standard $\mathcal{A}'$ reconnaissant le même langage que $\mathcal{A}$. Nous dirons que $\mathcal{A}'$ est le standardisé de $\mathcal{A}$.

Langages locaux

Un langage L sur l'alphabet Σ est dit local si on a $L \setminus \{\varepsilon\} = (P(L)\Sigma^* \cap \Sigma^* S(L)) \setminus \Sigma^* N(L) \Sigma^*$. Ceci revient à dire qu'un mot appartient à L ssi sa première lettre est dans $P(L)$, sa dernière lettre est dans $S(L)$, et si aucun de ses facteurs de longueur 2 n'appartient à $N(L)$.

Question 2. Montrer que tout langage local est régulier.

Question 3. Montrer que le langage décrit par l'expression régulière $(abc)^*$ est local.

Question 4. Montrer que le langage décrit par l'expression régulière a^*ba n'est pas local.

Question 5. Exhiber deux langages locaux L_1 et L_2, dont la réunion $L_1 \cup L_2$ n'est pas un langage local.

Question 6. Exhiber deux langages locaux L_1 et L_2, dont la concaténation $L_1 L_2$ n'est pas un langage local.

Question 7. Montrer que l'image d'un langage régulier par un morphisme est également un langage régulier.

Question 8. Soit $\mathcal{A} = (\Sigma, Q, I, F, \delta)$ un automate fini (déterministe ou non). Montrer que l'ensemble des calculs réussis de $\mathcal{A}$ est un langage local sur l'alphabet $Q \times \Sigma \times Q$.

Question 9. Un morphisme $\phi : X^* \mapsto Y^*$ est *alphabétique* si $|\phi(x)| \leqslant 1$ pour toute lettre $x \in \Sigma$; il est strictement alphabétique si $|\phi(x)| = 1$ pour toute lettre $x \in \Sigma$. Montrer que tout langage régulier est l'image d'un langage local par un morphisme strictement alphabétique.

Automates locaux

Un automate fini déterministe $\mathcal{A} = (\Sigma, Q, q_0, F, \delta)$ est dit *local* si les transitions étiquetées par une lettre x donnée arrivent toutes dans un même état, qui ne dépend donc que de x.

Question 10.
□ **10.1.** Montrer que tout langage local sur un alphabet Σ peut être reconnu par un *automate local standard* dont l'ensemble des états est $\Sigma \cup \{\varepsilon\}$.
□ **10.2.** Réciproquement, montrer que le langage reconnu par un automate local est lui-même local.

Question 11. L_1 et L_2 sont deux langages locaux sur des alphabets X et Y disjoints.
□ **11.1.** Montrer que $L_1 \cup L_2$ est un langage local.
□ **11.2.** Montrer que $L_1 \cdot L_2$ est un langage local.
□ **11.3.** Montrer que L_1^* est un langage local.

Algorithme de McNaughton, Yamada et Glushkov

Une expression régulière e sur l'alphabet Σ est dite linéaire si chaque lettre de Σ possède au plus une occurrence dans e. Soit e une expression régulière. On obtient une version linéaire e' de e en remplaçant les lettres qui apparaissent dans e par des lettres deux à deux distinctes ; plus précisément, les diverses occurrences dans e d'une même lettre a seront remplacées par $a_1, a_2, \dots, a_{|e|_a}$. Ainsi, une version linéaire de $e = (a|b)^*aba$ sera $e' = (a_1|b_1)^*a_2b_2a_3$.

Question 12. Soit e une expression régulière linéaire. Montrer que $L_{\text{reg}}(e)$ est un langage local.

Question 13. Que penser de la réciproque ?

Question 14. Définir, par induction structurelle, la fonction λ qui, à l'expression régulière e, associe $\{\varepsilon\} \cap L_{\text{reg}}(e)$.

Question 15. Définir, toujours par induction structurelle, une fonction π qui, à l'expression régulière linéaire e, associe $\pi(e) = P(L_{\text{reg}}(e))$. Nous aurons donc $\pi = P \circ L_{\text{reg}}$.

Question 16. Donner de même des définitions par induction structurelle des fonctions $\sigma = S \circ L_{\text{reg}}$ et $\varphi = F \circ L_{\text{reg}}$.

Question 17. L'algorithme de McNaughton-Yamada-Glushkov permet d'associer à une expression régulière e un automate fini reconnaissant le langage $L_{\text{reg}}(e)$. Il consiste en quatre étapes.

- Construire une version linéaire e' de l'expression régulière e, en mémorisant l'encodage des lettres qui apparaissent dans e.
- Construire les ensembles $\pi(e')$, $\sigma(e')$ et $\varphi(e')$.
- Construire un automate fini déterministe $\mathcal{A}'$ reconnaissant $L_{\text{reg}}(e')$.
- Décoder les étiquettes des transitions de $\mathcal{A}'$ pour obtenir un automate fini reconnaissant $L_{\text{reg}}(e)$.

Appliquer l'algorithme de McNaughton-Yamada-Glushkov à l'expression régulière $((ab(ac)^*|ca)^*b)^*$.

Question 18. Expliquer la phrase suivante qui conclut l'article dont est tiré ce sujet.

> « Berry and Sethi have given an unusual proof of a well-known result, namely that every rational language is the homomorphic image of a local language. »