Automate des occurrences

Leçons : 907 ; 909 ; 921

Références : CORMEN–LEISERSON–RIVEST–STEIN, *Introduction à l'algorithmique* (p.915) [?] et LESESVRE–MONTAGNON–LE BARBENCHON–PIERRON, *131 développements pour l'oral* (p. 763) [?]

Définition 1. Un automate complet et déterministe qui reconnait un langage L est dit minimal, s'il a un nombre minimal d'états parmi tous les automates complets et déterministes qui reconnaissent L.

Prérequis :

- Un automate complet et déterministe, dont le nombre d'état vaut le nombre de résiduels, est minimal.

Introduction :

On va présenter ici un automate qui permet de trouver les occurrences d'un motif dans un texte, il est notamment utile dans l'algorithme de Morris Pratt. Cet algorithme peut être amélioré en l'algorithme de Knuth–Morris–Pratt qui ne retient pas l'automate des occurrences mais seulement la longueur des bords des préfixes du motif m.

But : Trouver un algorithme qui cherche un motif dans un texte

Soit Σ un alphabet fini, soit $m \in \Sigma^*$ le motif que l'on veut chercher. On veut construire un automate fini qui reconnait les mots se finissant par m.

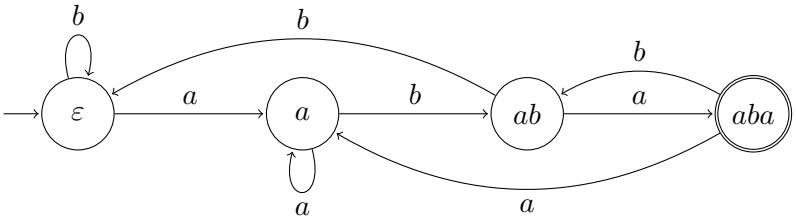
On pose $\mathcal{P}$ l'ensemble des préfixes du mot m. Pour tout $u \in \Sigma^*$, on pose $\sigma_m(u)$ le plus grand suffixe de u qui est dans $\mathcal{P}$ [1].

On va construire l'automate $\mathcal{A}_m$ de la manière suivante :

$$\mathcal{A}_m = (\mathcal{P}, \{\varepsilon\}, \{m\}, \delta, \Sigma)$$

où δ est définie telle que, pour tout $p \in \mathcal{P}$, $a \in \Sigma$, on ait $\boxed{\delta(p,a) = \sigma_m(pa)}$.

Exemple : Pour $m = aba$, l'automate $\mathcal{A}_m$ est de la forme



Théorème 1. L'automate $\mathcal{A}_m$ reconnait le langage $\Sigma^* m$ et cet automate est minimal.

1. i.e. qui est aussi suffixe de m

Démonstration. On prolonge la fonction δ aux mots. [2]

Étape 1 : *Montrons que* $\delta(p,u) = \sigma_m(pu)$

Pour tout $p = p_1 \dots p_k \in \mathcal{P}$, on va prouver par récurrence sur la longueur du mot u, que $\delta(p,u) = \sigma_m(pu)$.

Initialisation : $|u| = 1$, par définition de δ, on a $\delta(p,u) = \sigma_m(u)$ car $u \in \Sigma$.

Hérédité : Supposons que le résultat est vrai pour w avec $|w| \geqslant 1$. Soit $a \in \Sigma$, on veut prouver le résultat pour aw.

Par définition de δ, on a $\delta(p,aw) = \delta(\delta(p,a),w) = \delta(\sigma_m(pa),w) = \sigma_m(\sigma_m(pa)w)$ (par hypothèse de récurrence). Il faut donc prouver que $\sigma_m(\sigma_m(pa)w) = \sigma_m(paw)$.

— Si $\sigma_m(pa) \neq \varepsilon$, il existe donc $i \in \{1,\dots,k\}$ tel que $\sigma_m(pa) = p_i \dots p_k a$ et pour tout $j < i$, $p_j \dots p_k a$ n'est pas préfixe de m. Ainsi

$$\sigma_m(\sigma_m(pa)w) = \sigma_m(p_i \dots p_k aw) = \sigma_m(p_1 \dots p_k aw) \text{ [3]} = \sigma_m(paw)$$

— Si $\sigma_m(pa) = \varepsilon$, aucun suffixe de pa n'est préfixe de m, donc les suffixes de paw qui sont préfixes de m sont exactement les suffixes de w qui sont préfixes de m. Ainsi

$$\sigma_m(\sigma_m(pa)w) = \sigma_m(w) = \sigma_m(paw)$$

Ainsi on a prouvé que

$$\sigma_m(\sigma_m(pa)w) = \sigma_m(paw)$$

Conclusion : Pour tout mot $u \in \Sigma^*$, $\delta(p,u) = \sigma_m(pu)$.

Étape 2 : $L(\mathcal{A}_m) = \Sigma^* m$

Par double inclusion,

$\boxed{\subset}$ Si $w \in \mathrm{L}(\mathcal{A}_m)$, alors $\delta(\varepsilon,w) \in \{m\}$, d'où par l'étape 1, $\sigma_m(w) = m$. Donc, par définition, m est un suffixe de w, autrement dit, $w \in \Sigma^* m$.

$\boxed{\supset}$ Si $w \in \Sigma^* m$, alors $\sigma_m(w) = m$ car m est suffixe de w. Or $\sigma_m(w) = \delta(\varepsilon,w) = m$, donc $w \in \mathrm{L}(\mathcal{A}_m)$.

Étape 3 : *Montrons que* $\mathcal{A}_m$ *est minimal*

On note $L = \Sigma^* m$. Comme $\mathcal{A}_m$ est déterministe, complet et reconnait L, il suffit de prouver que $|\mathcal{P}|$ est égal au nombre de résiduels. On notera $\{u^{-1}L\}$ l'ensemble des résiduels du langage L. On sait que le nombre de résiduels est inférieur au nombre d'états de $\mathcal{A}_m$ [4] On a donc déjà

$$|\{u^{-1}L\}| \leqslant |\mathcal{P}|$$

2. pour un mot aw (avec $a \in \Sigma$ et $w \in \Sigma^*$) et $p \in \mathcal{P}$, on pose $\tilde{\delta}(p,aw) = \begin{cases} \delta(p,a) & \text{si } w = \varepsilon \\ \tilde{\delta}(\delta(p,a),w) & \text{sinon} \end{cases}$ bien définie car l'automate est déterministe et qu'on applique l'hypothèse de récurrence sur un mot de longueur strictement plus petite. On continuera de noter cette fonction δ

3. car pour tout $j < i$, $p_j \dots p_k a$ n'est pas préfixe de m donc $p_j \dots p_k aw$ n'est pas préfixe de m non plus

4. car pour tout $u \in \Sigma^*$, on note q l'unique (car déterministe) état dans lequel l'automate sera après avoir lu u (existe car complet), on a

$$u^{-1}L = \{v \in Q, q \xrightarrow{v} f, f \in F\}$$

Donc, $u^{-1}L$ ne dépend que de q, ainsi le nombre de résiduel est inférieur au nombre d'états de l'automate. Plus rigoureusement, l'application

$$\varphi : \left\{ \begin{array}{ccc} \{u^{-1}L\} & \to & Q \\ u^{-1}L & \mapsto & \delta(i,u) \end{array} \right.$$

est injective (où i est l'état initial de l'automate).

D'autre part, l'application

$$f : \begin{cases} \mathcal{P} & \to & \{u^{-1}L\} \\ p & \mapsto & p^{-1}L \end{cases}$$

est injective, car si $m_1 \dots m_i \neq m_1 \dots m_j$ avec $i < j$, alors $m_{j+1} \dots m_{|m|} \in (m_1 \dots m_j)^{-1}L$, or $m_{j+1} \dots m_{|m|} \notin (m_1 \dots m_i)^{-1}L$.[5] Ainsi $(m_1 \dots m_i)^{-1}L \neq (m_1 \dots m_j)^{-1}L$. Donc f est bien injective et l'on a

$$|\mathcal{P}| \leqslant |\{u^{-1}L\}|$$

Donc

$$|\mathcal{P}| = |\{u^{-1}L\}|$$

Ce qui permet de conclure sur le fait que $\mathcal{A}_m$ est minimal. □

Astuces de l'agrégatif :

On pourrait initialiser la récurrence à 0.

Il faut bien comprendre ce que sont les résiduels d'un langage et la construction de l'automate des résiduels pour maitriser ce développement. On a utilisé de manière cruciale que le nombre de résiduels d'un langage est inférieur au nombre d'états de tout automate déterministe complet reconnaissant le langage.

5. car on a lu moins de $|m|$ caractères donc on ne peut pas atteindre l'état final