Théorie des langages formels – complément de TD

Romuald THION

7 décembre 2017

Résumé

Complément de TD aux supports de LIF "Théorie des langages formels".

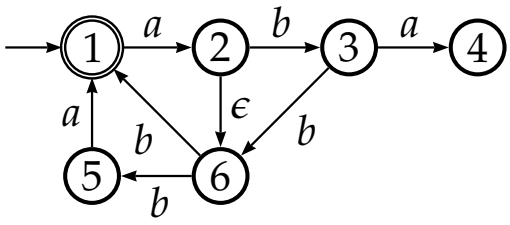
Table des matières

1 Déterminisation

À partir d'un automate non-déterministe dont on connait $\Delta : K \times (\Sigma \cup \{\epsilon\}) \rightarrow \wp(K)$ [1], le principe de la déterminisation consiste à calculer, jusqu'à l'obtention d'un point fixe, l'ensemble des ensembles d'états accessibles en lisant un mot. Intuitivement, il s'agit de représenter l'état global d'un système où plusieurs automates non-déterministes s'exécuteraient en parallèle et de façon synchrone, chacun essayant de prendre un chemin différent pour lire le mot.

1. en premier lieu, on calcule la ϵ fermeture transitive $E^+ \subseteq K \times K$ de la relation d'accessibilité E entre paires d'états en lisant seulement le mot vide $E = \{(x, y) | y \in \Delta(x, \epsilon)\}$;
2. on initialise avec $S = \{\{i\} \cup \{x \mid (i, x) \in E^+\}\}$ l'ensemble des états ϵ accessibles depuis l'état initial. Noter qu'il s'agit bien d'un *ensemble d'ensembles* d'états ;
   — $S' := S$;
   — pour chaque $s \in S'$
      — pour chaque lettre $a \in \Sigma$;
         — calculer s_a l'ensemble des états a accessibles soit $s_a := \bigcup\{\Delta(x, a) \mid x \in s\}$;
         — fermer l'ensemble par ϵ transitions $s_a := \{x \mid \exists y \in S_a \wedge (y, x) \in E^+\} \cup s_a$;
         — ajouter s_a à l'ensemble des états de l'automate déterministe $S' := S' \cup \{s_a\}$;
         — définir $\delta(s, a) = s_a$ pour la fonction de transition déterministe ;
   — tant que $S \subseteq S'$ (on a atteint de nouveaux états) relancer l'algorithme avec S' ;
3. pour chaque $s \in S$, si $s \cap F \neq \emptyset$, alors ajouter s dans l'ensemble des finaux.

1.1 TD3.B.2



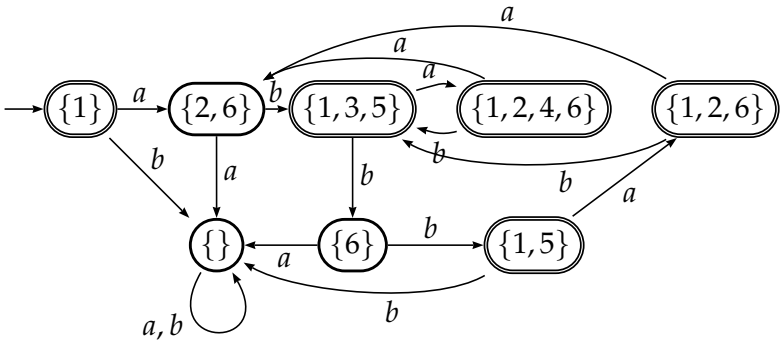
Ici, la seule ϵ transition est de 2 à 6, $E+ = \{(x, x) \mid x \in K\} \cup \{(2, 6)\}$, informellement si on peut atteindre 2 alors on peut aussi atteindre 6.

1. on commence avec $S = \{\{1\}\}$
   — pour $\{1\}$
      — pour a on atteint le nouvel état $\{2, 6\}$, c'est-à-dire $\{2\}$ une fois fermé par ϵ (car $\Delta(1, a) = \{2\}$ et $(2, 6) \in E^+$) ;
      — pour b on atteint le nouvel état $\{\}$;
2. on continue avec $S = \{\{1\}, \{2, 6\}, \{\}\}$;
   — pour $\{2, 6\}$
      — pour a on atteint $\{\}$;
      — pour b on atteint $\{3\}$ via 2, et $\{1, 5\}$ via 6, le nouvel état est $\{3\} \cup \{1, 5\} = \{1, 3, 5\}$;

1. noter que $R \subseteq A \times B$ est essentiellement le même objet que $R : A \rightarrow \wp(B)$: toute relation entre A et B peut être représentée par une fonction de A dans l'ensemble des sous-ensembles de B et réciproquement.

— comme $\{\}$ est vide, $\forall a \in \Sigma$ on a toujours $\delta(\{\}, a) = \{\}$, c'est l'état puit des mots qui ne sont pas lus en entier ;

3. on continue avec $S = \{\{1\}, \{2,6\}, \{\}, \{1,3,5\}\}$;
   — pour $\{1,3,5\}$
      — pour a on atteint $\{1,2,4,6\}$ car on atteint $\{2,6\}$ via 1, $\{4\}$ via 3 et $\{1\}$ via 5 ;
      — pour b on atteint $\{6\}$;
4. on continue avec $S = \{\{1\}, \{2,6\}, \{\}, \{1,3,5\}, \{1,2,4,6\}, \{6\}\}$;
   — pour $\{1,2,4,6\}$
      — pour a on atteint $\{2,6\}$
      — pour b on atteint $\{1,3,5\}$;
   — pour $\{6\}$
      — pour a on atteint $\{\}$
      — pour b on atteint $\{1,5\}$;
5. on continue avec $S = \{\{1\}, \{2,6\}, \{\}, \{1,3,5\}, \{1,2,4,6\}, \{6\}, \{1,5\}\}$;
   — pour $\{1,5\}$
      — pour a on atteint $\{1,2,6\}$
      — pour b on atteint $\{\}$;
6. on continue avec $S = \{\{1\}, \{2,6\}, \{\}, \{1,3,5\}, \{1,2,4,6\}, \{6\}, \{1,5\}, \{1,2,6\}\}$;
   — pour $\{1,2,6\}$
      — pour a on atteint $\{2,6\}$
      — pour b on atteint $\{1,3,5\}$;
7. à cette étape, *il n'y a plus de nouveaux états rencontrés*, on s'arrête.



2 D'automate à expression

Pour déterminer l'expression rationnelle décrivant le langage reconnu par un automate, on va considérer des automates *enrichis* dont les étiquettes des arêtes sont des *expressions rationnelles* et non plus seulement des lettres.

On réduit l'automate de départ en supprimant progressivement les noeuds intermédiaires entre l'état de départ et l'état final. Si on a plusieurs états finaux on « prépare » l'automate en ajoutant un état F, avec une ϵ transition depuis chaque état final vers F. Voir construction sur le dernier exemple.

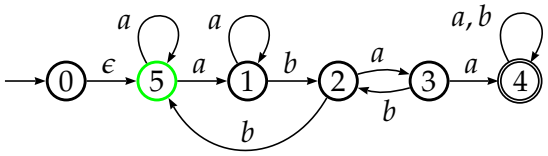
Lors de la suppression du noeud j, on regarde tous les chemins i à k passant par j en considérant chaque transition de i (source) vers j et chaque transition de j vers k (cible). On ajoute

aux chemins existants une transition de i vers kk pour ne rien perdre lors de la suppression. Par exemple avec $i \xrightarrow{\alpha} j$, $j \xrightarrow{\beta} k$ et $j \xrightarrow{\gamma} j$, on va compléter le chemin de i à k avec une transition de la forme $\alpha\gamma^*\beta$.

Progressivement, on arrive à un automate ne comportant qu'un seul chemin entre l'état initial et l'unique état final : c'est une expression rationnelle du langage reconnu. On peut obtenir plusieurs expressions syntaxiquement différentes mais qui décrivent le même langage selon l'ordre dans lequel on supprime les noeuds intermédiaires.

Notons que dans la preuve il faut suivre l'ordre des états tels que numérotés. Ce n'est en fait pas une contrainte car on peut toujours trouver une nouvelle numérotation (bijective !) et changer les labels des états.

2.1 TD5.A.a

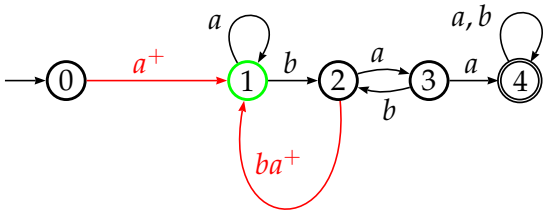


Pour supprimer l'état 5 (en vert sur l'automate précédent) :

— comme on peut aller de 0 à 5 (en lisant ϵ) et de 5 à 1 (en lisant a), en passant par 5 autant de fois que l'on veut (en lisant a) : on ajoute la transition $\epsilon a^* a = a^+$ de 0 à 1 ;

— comme on peut aller de 2 à 5 (en lisant b) et de 5 à 1 (en lisant a), en passant par 5 autant de fois que l'on veut (en lisant a) : on ajoute la transition $ba^* a = ba^+$ de 2 à 1 ;

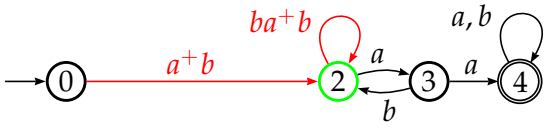
On obtient ainsi (changements en rouge).



Pour supprimer l'état 1 :

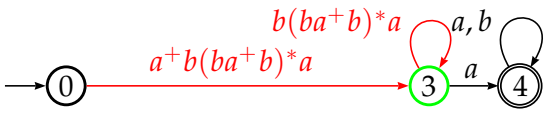
— comme on peut aller de 0 à 2 en passant par 1 autant de fois que l'on veut : on ajoute $a^* a a^* b = a a^* b = a^+ b$ de 0 à 2 ;

— comme on peut aller de 2 à 2 en passant par 1 autant de fois que l'on veut : on ajoute une boucle $baa^* a^* b = ba^+ b$ de 2 à 2 ;

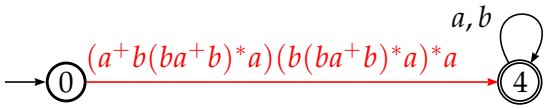


Pour supprimer l'état 2 :

— comme on peut aller de 0 à 3 en passant par 2 autant de fois que l'on veut : on ajoute $a^+b(ba^+b)^*a$ de 0 à 3 ;
— comme on peut aller de 3 à 3 en passant par 2 autant de fois que l'on veut : on ajoute une boucle $b(ba^+b)^*a$ de 3 à 3 ;



Pour supprimer l'état 3 :
— comme on peut aller de 0 à 4 en passant par 3 autant de fois que l'on veut : on ajoute $a^+b(ba^+b)^*a(b(ba^+b)^*a)^*a$ de 0 à 4 ;

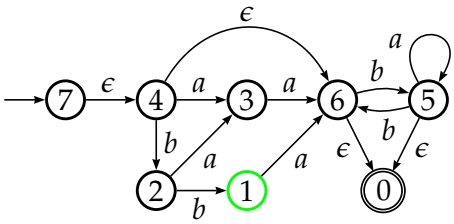


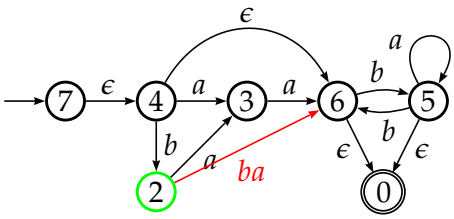
Il ne reste plus qu'à supprimer la dernière boucle pour obtenir l'expression :

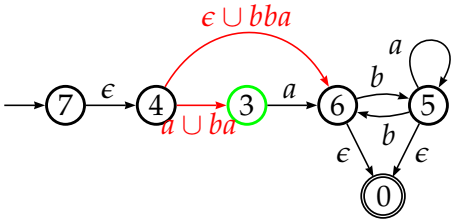
$$a^+b(ba^+b)^*a(b(ba^+b)^*a)^*a(a \cup b)^* = a^+(b(ba^+b)^*a)^+a(a \cup b)^*$$

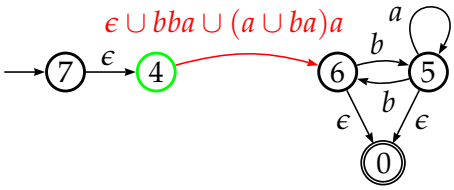
2.2 TD5.A.b

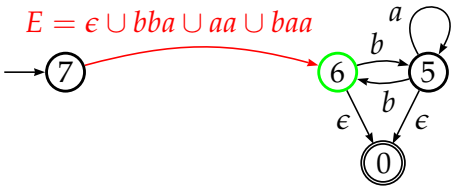
On va supprimer les états dans cet ordre : 1, 2, 3, 4, 6, 5.







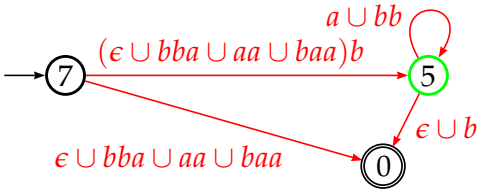




Pour supprimer l'état 6 :

— il faut regarder les transitions partant de 7 (expression E lue) et allant soit vers 5 (Eb lue) soit vers 0 ($E\epsilon$, rien à lire de plus, 6 était final) ;

— il faut regarder les transitions partant de 5 (b lu) et allant soit vers 5 (bb lue, on rajoute une boucle à celle existante) soit vers 0 ($b\epsilon$) ;



On obtient alors l'expression finale :

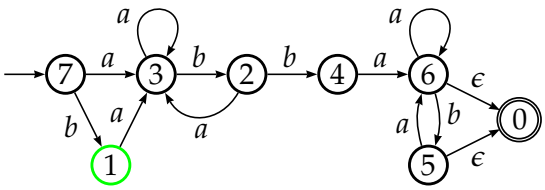
$$(\epsilon \cup bba \cup aa \cup baa) \cup (\epsilon \cup bba \cup aa \cup baa)b(a \cup bb)^*(\epsilon \cup b)$$

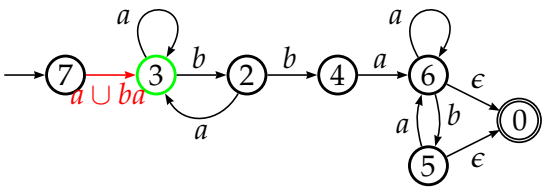
Soit

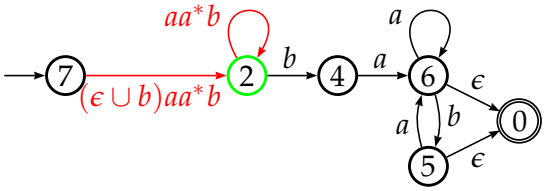
$$(\epsilon \cup bba \cup aa \cup baa)(\epsilon \cup b(a \cup bb)^*(\epsilon \cup b))$$

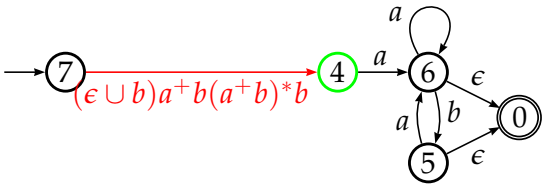
2.3 TD5.A.c

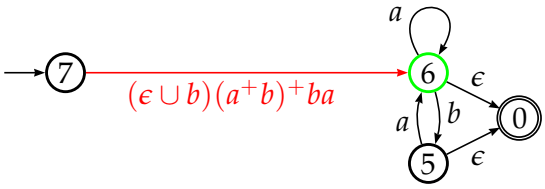
On prépare l'automate en ajoutant un état final unique et en supprimant l'état puits non-acceptant.



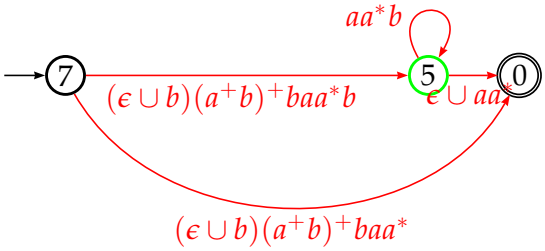








Pour éliminer 6, on remarque que l'on peut aller :
— de 7 à F ou de 7 à 5 en bouclant une infinité de fois sur 6 ;
— de 5 à F ou de 5 à 5 en bouclant une infinité de fois sur 6 ;



Après avoir supprimé 5 on obtient :

$$(\epsilon \cup b)(a^+b)^+baa^* \cup (\epsilon \cup b)(a^+b)^+baa^*b(aa^*b)^*(\epsilon \cup aa^*)$$

Soit

$$(\epsilon \cup b)(a^+b)^+ba^+ \cup (\epsilon \cup b(a^+b)^*a^*)$$

3 Théorème de Myhill–Nerode et minimisation

Pour un langage $L \subseteq \Sigma^*$ et un mot u on définit le *contexte à droite* relativement à L noté $u^{-1}L$, dit aussi *quotient à droite*, par l'équation $u^{-1}L = \{z \in \Sigma^* | uz \in L\}$. Informellement, $u^{-1}L$ désigne l'ensemble des mots qui peuvent être lus pour compléter u en un mot de L.

Le Théorème de Myhill–Nerode établit qu'un langage est régulier *si et seulement si* l'ensemble des quotients $Q = \{u^{-1}L | u \in \Sigma^*\}$ est fini. Un élément $u^{-1}L \in Q$ est appelé *classe* (d'équivalence) du mot u.

3.1 Calcul d'automate minimal par les classes d'équivalences

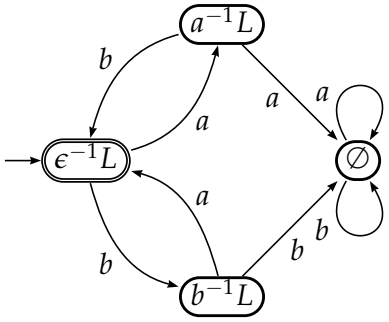
Le Théorème de Myhill–Nerode donne lieu à la construction de l'automate déterministe de Nerode $\mathcal{A} = \langle Q, \Sigma, \delta, L, F \rangle$ définit comme suit. Cet automate est *minimal* en son nombre d'état et est unique à un renommage des états près.

— les états sont les différents quotients Q
— l'état initial est le langage $\epsilon^{-1}L = L$
— les états finaux sont les langages $F = \{u^{-1}L | u \in L\} \subseteq Q$
— la transition définie par $\delta(u^{-1}L, a) = a^{-1}(u^{-1}L) = (ua)^{-1}L$

Considérons $L = (ab \cup ba)^*$, on va calculer « à la main » les classes successives de L en commençant par L lui-même, c'est-à-dire $\epsilon^{-1}L$, puis en « lisant » des mots de plus en plus longs pour parcourir l'automate de Nerode $\mathcal{A}$:

$$\begin{aligned}
\epsilon^{-1}L &= L = L_0 \\
a^{-1}L_0 &= b(ab \cup ba)^* = L_1 \\
b^{-1}L_0 &= a(ab \cup ba)^* = L_2 \\
a^{-1}L_1 &= \emptyset \\
b^{-1}L_1 &= (ab \cup ba)^* = L \\
a^{-1}L_2 &= (ab \cup ba)^* = L \\
b^{-1}L_2 &= \emptyset
\end{aligned}$$

L'automate de Nerode $\mathcal{A}$ ainsi obtenu est le suivant :



Le Théorème de Myhill–Nerode assure que le processus de calcul des classes s'arête si et seulement si L est régulier. Pour vous en convaincre, calculer les classes du langage irrégulier archétypique $I = \{a^n b^n | n \in \mathbb{N}\}$ ou du langage $\{a^n | n \in \mathbb{N} \wedge \text{n premier}\}\ldots$