Info Spés - 22 Octobre 2002 **DS 1 - 3 Heures**

Langages et automates

Ce sujet est constitué de deux problèmes totalement indépendants. Le premier concerne le théorème de Kleene (on y voit des points de vue légèrement différents de ceux pris en cours). Le second, plus ardu, concerne les facteurs itérants d'un langage.

Je suggère à ceux voulant se frotter à des questions plus compliquées de commencer par le second problème (et d'y rester !). Les autres commenceront par le premier problème, et auront normalement largement le temps de commencer le second, dont la première partie est plus aisée.

Quoi qu'il en soit, les deux problèmes seront rédigés sur des copies distinctes.

Chaque problème sera noté avec son propre barème, donnant lieu à deux "sigma" intermédiaires. La note sur 20 sera issue d'un barycentrage tant mystérieux que subtil et pertinent[1]...

1 Kleene revisited

On rappelle qu'un automate (fini) classique est un quintuplet $\mathcal{A} =< Q, A, T, I, F >$ avec Q un ensemble fini (d'états), A un ensemble fini (de lettres), T un ensemble de transitions (une partie de $Q \times A \times Q$), $I \subset Q$ est l'ensemble des états initiaux, et F l'ensemble des états acceptants, ou terminaux, ou finaux, ou... Cet automate sera dit déterministe lorsque I est un singleton et lorsque pour tout $(q, \alpha) \in Q \times A$, il existe au plus une transition $(q, \alpha, q') \in T$.

Un calcul de $\mathcal{A}$ est une suite de transitions $q_0 \xrightarrow{\alpha_1} q_1 \xrightarrow{\alpha_2} \cdots \xrightarrow{\alpha_n} q_n$ (avec pour tout i : $(q_{i-1}, \alpha_i, q_i) \in T$) : un tel calcul est dit étiqueté par $\alpha_1 ... \alpha_n$. Il est déclaré réussi lorsque $q_0 \in I$ et $q_n \in F$. Le langage reconnu par $\mathcal{A}$ est l'ensemble des étiquettes des calculs réussis. Un tel langage sera dit reconnaissable sur A. Deux automates seront déclarés équivalents lorsqu'ils reconnaissent le même langage.

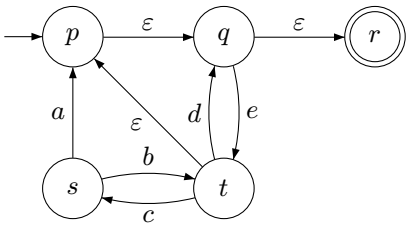
$RatA^*$ désigne l'ensemble des langages rationnels sur un aphabet A, et $RecA^*$ l'ensemble des langages reconnaissables sur ce même alphabet.

1.1 Mettre ou ne pas mettre des ε-transitions...

Un automate "avec transitions spontanées" est un sextuplet $\mathcal{A} =< Q, A, T, I, F, T_\varepsilon >$: la seule différence avec un automate usuel est la présence d'"ε-transitions", reliant différents états par le mot vide. $T_\varepsilon \subset Q \times Q$ est l'ensemble des (q_1, q_2) tels qu'existe une transition $q_1 \xrightarrow{\varepsilon} q_2$. La définition d'un calcul reste la même que dans l'introduction, si ce n'est qu'on peut avoir $\alpha_i = \varepsilon$. Le reste des définitions reste inchangé.

Un automate (avec transitions spontanées) est dit standard lorsqu'il comporte un seul état initial i, et qu'aucune transition n'arrive sur cet état, i.e. : il n'existe pas de transition $(q, \alpha, i) \in T$ ou $(q, i) \in T_\varepsilon$.

1. (a) Proposer un moyen pour construire un automate classique (pas forcément déterministe) équivalent à un automate avec transitions spontanées donné, et ceci sans changer l'ensemble des états.
   (b) Evaluer la complexité de votre algorithme (en terme de ... et en fonction de ...). Comparer à l'algorithme de détermination par la méthode des parties.
   (c) Illustrer l'algorithme sur l'automate suivant :



[1] Cette même note sur 20 sera elle même barycentrée avec la seconde pour donner une moyenne de trimestre très lisible !

2. Deux applications...
   (a) Montrer que l'ensemble des sous-mots d'un langage reconnaissable est reconnaissable.
   (b) Montrer que tout automate standard est équivalent à un automate classique, et réciproquement. En utilisant impérativement des constructions sur les automates standards, montrer que $RatA^* \subseteq RecA^*$.

L'objectif des deux parties qui suivent est de montrer $RecA^* \subseteq RatA^*$, d'une autre façon que dans le cours, c'est-à-dire sans utiliser le lemme d'Arden (la résolution de l'équation aux langages $L = KL + M$ d'inconnue L lorsque $\varepsilon \notin K$).

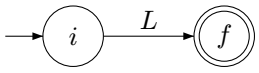
1.2 L'algorithme BMC...

... pour Brzozowski et Mc Cluskey. *"méthode d'élimination"*

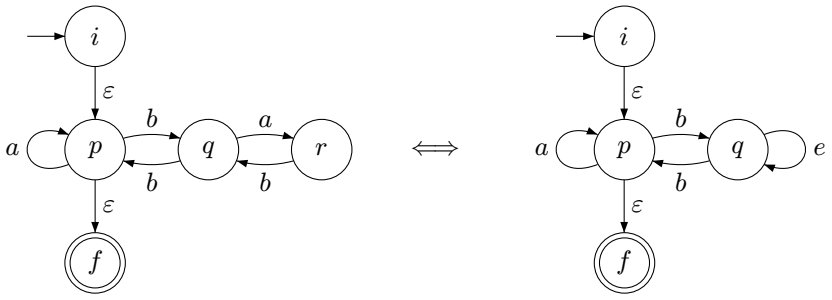
On introduit les automates généralisés $\mathcal{A} =< Q, A, T, I, F >$ dans lesquels les étiquettes des transitions ne sont plus des lettres de A, mais des parties de A^* (*i.e.* des ensembles de mots) : T est maintenant une partie de $Q \times \mathcal{P}(A^*) \times Q$. L'étiquette d'un calcul est, comme habituellement, le mot formé en concaténant les étiquettes des transitions du chemin (c'est donc un langage !). Le langage reconnu par un tel automate est l'union des étiquettes des calculs réussis, c'est-à-dire des calculs allant d'un état initial à un état final.

1. Montrer que l'on peut toujours supposer que quels que soient les états p, q de Q, il existe au plus une transition de p à q. C'est ce qu'on fera par la suite.

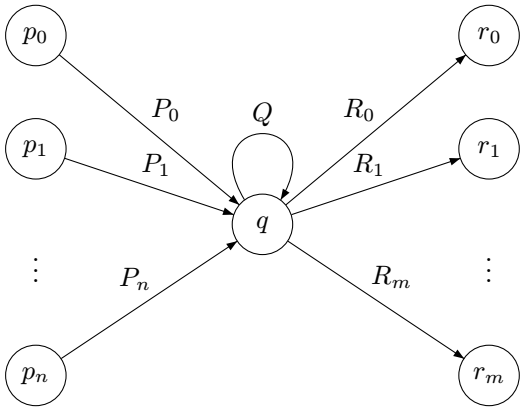
   Idée : On va construire à partir de $\mathcal{A}$ considéré comme généralisé un automate $\mathcal{B}$ équivalent, mais n'ayant que deux états i (initial) et f (final), reliés par une seule transition, étiquetée par L, dénoté sous forme de langage associé à une expression rationnelle. On aura alors bien montré l'inclusion voulue, avec en plus un algorithme effectif pour obtenir une expression du langage reconnu par $\mathcal{A}$.

   - *Première phase* : On construit un automate généralisé "standard" $\mathcal{A}'$ à partir de $\mathcal{A}$ en ajoutant deux nouveaux états i (seul état initial) et f (seul état final), et en considérant les étiquettes des transitions comme des ensembles à un élément.
   - *Deuxième phase* : On supprime un à un les éléments de Q en modifiant l'automate pour que les langages soient équivalents.
2. Détailler la première phase, pour avoir $\mathcal{A}$ et $\mathcal{A}'$ équivalents.
3. Montrer comment remplir l'étiquette e dans le cas suivant :

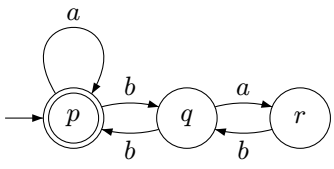
   

   pour que les deux automates représentés soient équivalents.
4. Que fait-on dans le cas général suivant, lorsqu'il s'agit de supprimer le sommet q ?

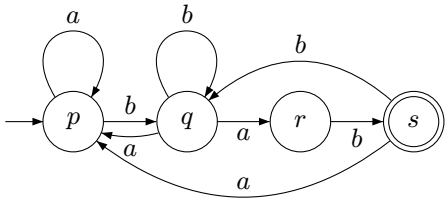


On n'a pas représenté d'éventuelles transitions (p_i, S, r_j).

5. Montrer soigneusement que l'opération précédente réalise bien l'équivalence désirée.
6. Jouer l'algorithme sur les automates $\mathcal{A}_1$ et $\mathcal{A}_2$ suivants :



Automate $\mathcal{A}_1$



Automate $\mathcal{A}_2$

7. Montrer qu'il n'y a pas unicité de l'expression du langage ainsi obtenue.

1.3 L'algorithme MNY...

... pour Mc Naughton et Yamada.

Dans cette dernière partie, on considère un automate $\mathcal{A} =< Q, A, T, I, F >$ sans ε-transitions. On suppose : $Q = [\![1, n]\!]$. Pour $1 \leq p, q \leq n$, on définit $R_{p,q}$ l'ensemble des calculs (indexé par $w \neq \varepsilon$) dans $\mathcal{A}$ dont l'origine est p et l'extrémité est q.

A k fixé dans $[\![0, n]\!]$, on note $R^{(k)}_{p,q}$ l'ensemble des étiquettes des calculs qui vont de p à q sans passer par des états intermédiaires $> k$ (au sens strict : par exemple, dans un chemin réduit à une transition $p \xrightarrow{\alpha} q$, il n'y a pas d'état intermédiaire). On convient de représenter, à k fixé, $R^{(k)}_{p,q}$ en position (p, q) d'une matrice de taille $|Q| \times |Q|$.

1. Que représente $R^{(0)}_{p,q}$? Donner sa forme matricielle pour les automates $\mathcal{A}_1$ et $\mathcal{A}_2$.
2. Exprimer $R^{(k+1)}_{p,q}$ en fonction de $R^{(k)}_{p,k+1}$, $R^{(k)}_{k+1,k+1}$ et $R^{(k)}_{k+1,q}$. *On demande d'accompagner le résultat d'un dessin et d'une preuve sommaire.*
3. Exprimer le langage reconnu par $\mathcal{A}$ en fonction des $R_{p,q}$ puis des $R^{(k)}_{p,q}$, et conclure.
4. Traiter en exemple les automates $\mathcal{A}_1$ et $\mathcal{A}_2$, en faisant des simplifications "au vol". Montrer que là encore, il n'y a pas unicité de l'expression obtenue.

2 Facteurs itérants d'un langage

2.1 Automates réguliers à gauche

Soit $\mathcal{A}$ un automate déterministe complet : l'ensemble de transitions peut alors être avantageusement remplacé par la fonction de transition $\delta : Q \times A \to Q$, qui s'étend en une fonction de $Q \times A^*$ dans Q. On note ainsi : $\mathcal{A} = \langle Q, A, \delta, q_0, F\rangle$. On dit que $\mathcal{A}$ est régulier à gauche lorsque

$$\forall f, g \in A^* \qquad \Big(\delta(q_0, f) = \delta(q_0, g) \quad \Longrightarrow \quad [\ \forall u \in A^*,\ \delta(q_0, uf) = \delta(q_0, ug)\ ]\Big)$$

On va montrer le résultat suivant :

Tout automate déterministe complet est équivalent à un automate déterministe complet régulier à gauche.

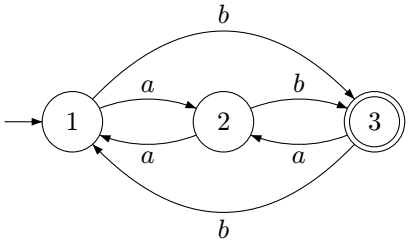
Soit donc $\mathcal{A} = \langle Q, A, \delta, q_1, F\rangle$ un automate déterministe complet avec $|Q| = N$. On note $q_1, q_2, \ldots, q_N$ les éléments de Q et $\mathcal{V}(Q)$ les N-uplets d'éléments de Q. On définit $i = (q_1, q_2, \ldots, q_N)$ et, pour $r \in \mathcal{V}(Q)$, on note r_p la p^{ieme} composante de r. On a donc pour tout $p \in [\![1, N]\!]$: $i_p = q_p$ par définition de i.

On définit l'automate $\mathcal{A}_g = \langle \mathcal{V}(Q), A, \eta, i, U\rangle$ en prenant :

$$U = \{r \in \mathcal{V}(Q) \mid r_1 \in T\}$$

et pour chaque $r \in \mathcal{V}(Q)$ et $a \in A$, $\eta(r, a) = s$, avec pour tout $p \in [\![1, N]\!]$, $s_i = \delta(r_i, a)$.
On ne conserve de l'automate ainsi défini que les états effectivement accessibles (cf. la déterminisation des automates).

1. Vérifier que $\mathcal{A}_g$ est bien déterministe complet.
2. Vérifier que $\mathcal{A}_g$ est équivalent à $\mathcal{A}$.
3. Vérifier que $\mathcal{A}_g$ est régulier à gauche.
4. Appliquer ce procédé de régularisation à l'automate $\mathcal{B} = \langle\{1, 2, 3\}, \{a, b\}, \delta, 1, \{3\}\rangle$ suivant :



Automate $\mathcal{B}$

5. Montrer que la propriété "régulier à gauche" ne s'étend pas à d'autres états qu'à l'état initial (i.e. en remplaçant q_0 par un autre état dans la définition de la régularité à gauche).
6. Montrer que si $\mathcal{A} = \langle Q, A, \delta, q_0, T\rangle$ est régulier à gauche, alors

$$\forall f, g \in A^*, \Big(\delta(q_0, f) = \delta(q_0, g) \quad \Longrightarrow \quad \forall k \in \mathbb{N},\ \delta(q_0, f^k) = \delta(q_0, g^k)\Big)$$

2.2 Définition des facteurs itérants

On appelle facteur itérant d'un langage L sur A^* tout mot v (non vide) vérifiant :

$$\exists\ u, w \in A^* \mid uv^*w \subseteq L$$

et on note $Iter(L)$ l'ensemble des facteurs itérants du langage L. Pour v facteur itérant on dira que le mot u (resp. w) intervenant dans la définition précédente est un contexte gauche (resp. droite) de v.

1. Montrer que $Iter(L)$ est non vide si et seulement si L est infini.
2. Soient
$$L_1 = \{u \in \{a,b\}^* \mid |u|_a = |u|_b\}$$
et
$$L_2 = \{u \in a^*c^*b^* \mid |u|_a = |u|_b\}$$
   (a) Montrer que L_1 n'est pas rationnel.
   (b) Montrer : $Iter(L_1) = L_1 \setminus \{\varepsilon\}$.
   (c) Montrer que L_2 n'est pas rationnel. Calculer $Iter(L_2)$. Est-il rationnel ?
3. Soit maintenant L un langage sur A^* reconnu par un automate $\mathcal{A} = \langle Q, A, \delta, q_0, T\rangle$ déterministe émondé : tous les états sont accessibles (depuis q_0) et co-accessibles (il existe un mot étiquetant un chemin jusqu'à un état final). On note $N = |Q|$.
Pour chaque état p de Q on note $\mathcal{A}_p = \langle Q, A, \delta, p, \{p\}\rangle$ l'automate obtenu en remplaçant l'ensemble des états finaux par $\{p\}$ et l'état initial par p.
   (a) Montrer que pour tout p dans Q, $L(\mathcal{A}_p) \subseteq Iter(L)$.
   (b) Montrer que pour tout facteur itérant v de L, il existe un entier $k \leq N$ et un état p dans Q tel que v^k appartient à $L(\mathcal{A}_p)$, et qu'il existe u contexte gauche de v tel que $\delta(q_0, u) = p$.

   On a donc montré :
$$\left[\bigcup_{p\in Q} L(\mathcal{A}_p)\right] \subseteq Iter(L)$$
   et
$$Iter(L) \subseteq \bigcup_{p\in Q}\left(\bigcup_{1\leq k\leq N} \mathcal{R}_k(L(\mathcal{A}_p))\right) \qquad (I)$$
   où $\mathcal{R}_k(L)$ désigne le langage racine k^{ieme} de L, c'est-à-dire l'ensemble des $w \in A^*$ tels que $A^k \in L$.
4. En utilisant les automates régularisés, montrer que si L est rationnel, alors $\mathcal{R}_k(L)$ l'est aussi.
   On va maintenant restreindre les $\mathcal{R}_k(L(\mathcal{A}_p))$ dans (I) pour avoir une égalité, tout en gardant des langages rationnels.
   Pour un état p de $\mathcal{A}$ et un entier k fixé, on appelle $\mathcal{J}_{p,k}$ l'ensemble des facteurs itérants de L vérifiant ces conditions.
$$\mathcal{J}_{p,k} = \left\{v \in Iter(L) \mid v^k \in L(\mathcal{A}_p) \text{ et il existe } u \in A^* \text{ contexte gauche pour } v \text{ tel que } \delta(q_0, u) = p\right\}$$
5. Montrer que l'on a alors :
$$Iter(L) = \bigcup_{p\in Q}\left(\bigcup_{1\leq k\leq N} \mathcal{J}_{p,k}\right)$$
   Nous allons maintenant construire un automate reconnaissant $\mathcal{J}_{p,k}$.
6. Pour un état p de Q et un entier k fixé :
   - on construit l'automate $\mathcal{A}'$ obtenu à partir de $\mathcal{A}$ en remplaçant l'état initial par p.
   - on complète si nécessaire cet automate, puis on le régularise avec la méthode vue dans la première partie de ce problème. On obtient ainsi l'automate $\mathcal{A}_g = \langle \mathcal{V}(Q), A, \eta, i, U\rangle$, où i est le vecteur état contenant tous les états de $\mathcal{A}$ et dont la première composante est l'état p.
   - on définit
$$W_{p,k} = \{q \in \mathcal{V}(Q) \mid \exists v \in \mathcal{J}_{p,k};\quad q = \eta(p, v)\}$$
     et on remplace dans $\mathcal{A}_g$ l'ensemble des états finaux U par $W_{p,k}$.

   L'automate finalement obtenu est appelé $\mathcal{A}_{p,k}$.
   Montrer : $L(\mathcal{A}_{p,k}) = \mathcal{J}_{p,k}$.
7. Conclure.

2.3 Construction de l'automate reconnaissant $Iter(L)$

La définition de $W_{p,k}$ n'est malheureusement pas constructive, *a priori*. Nous allons montrer cependant qu'il est possible de calculer $W_{p,k}$ par un algorithme, donc de calculer l'automate qui reconnait $Iter(L)$ à partir de L.

1. Montrer que si $v \in \mathcal{J}_{p,k}$, alors il existe $u, w \in A^*$ tels que :
   - $uv^*w \subseteq L$
   - $\delta(q_0, u) = p$
   - $|u| \leq N$
   - $|w| \leq N^k$

   On pourra, par exemple, considérer les k chemins partant de l'état p et étiquetés w, vw, ..., $v^{k-1}w$ pour montrer le dernier point.
2. Montrer de manière analogue que si $q \in W_{p,k}$, alors il existe $v \in \mathcal{J}_{p,k}$ tel que $\delta(p, w) = q$ et $|v| \leq N^N$.
3. En déduire un algorithme en $\mathcal{O}(N^{k+N})$ qui calcule $W_{p,k}$ à partir de l'automate $\mathcal{A}$ reconnaissant L.
4. Ecrire finalement un algorithme qui calcule un automate reconnaissant $Iter(L)$ à partir d'un automate qui reconnait L. Quelle est sa complexité ? *On donnera un majorant "raisonnable" au vu de la construction ; par raisonnable, vu les ordres de grandeur, cela peut vouloir dire à un facteur polynômial (en N) près !*