Correction du DS 1

Laure Danthony et Laurent Fousse

3 novembre 2002

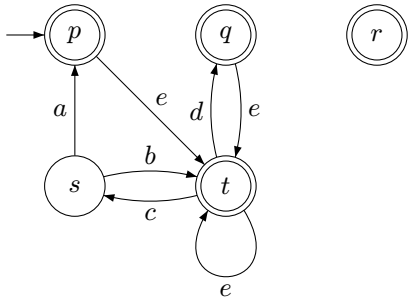
1 Kleene Revisited

1.1 Mettre ou ne pas mettre des ε-transitions

1. (a) On effectue ce que l'on appelle une "fermeture avant" de l'automate. Dans un premier temps, on prend la clôture transitive des ε-transitions, c'est-à-dire si $(q_i, q_j) \in T_\varepsilon$ et $(q_j, q_k) \in T_\varepsilon$, alors on ajoute (q_i, q_k) dans T_ε. A ce stade, on peut toujours supposer que dans un un chemin, deux flèches successives ne sont jamais étiquetées par ε. Dans un deuxième temps, on réalise la fermeture avant proprement dite : à l'automate $\mathcal{A} =< Q, A, T, I, F, T_\varepsilon >$ fermé par ε-transition on associe l'automate $< Q, A, T', I, F' >$ où T' est obtenu à partir de T_ε par l'opération suivante : si $p \xrightarrow{\varepsilon} q$ et $q \xrightarrow{a} q'$ alors on ajoute $p \xrightarrow{a} q'$ et on enleve $q \xrightarrow{\varepsilon} q'$. Un état p est état final du nouvel automate si et seulement si il est un état final de $\mathcal{A}$ ou bien si il est l'origine d'une transition spontanée dont l'extrémité est un état final de $\mathcal{A}$. Le nouvel automate est bien équivalent à l'automate de départ (vérification rapide à l'aide des calculs réussis).

   (b) Si on considère l'automate clos par ε-transitions, le nombre de nouvelles transitions crées à partir de p est clairement linéaire en le nombre de transitions spontanées partant de p. D'où un algorithme en $O(nt)$ où $n = |Q|$ et t le nombre de ces transitions spontanées. Au total, l'algorithme est polynomial en le nombre de sommets de l'automate (ce nombre n'a pas varié). Les grincheux vérifieront que la clôture est quadratique.

   (c) On obtient assez facilement l'automate suivant : [1]



2. (a) On ajoute à l'automate $\mathcal{A}$ de départ ($L(\mathcal{A}) = L$) toutes les transitions spontanées $p \xrightarrow{\varepsilon} q$ pour $(p, q) \in Q^2$. Ensuite, on réalise la fermeture avant de cet automate. On obtient un automate $\mathcal{B}$ qui reconnait l'ensemble des sous-mots de mots de L.

   (b) Un automate classique fournit rapidement un automate standard reconnaissant le même langage : On crée un nouvel état initial $i \notin Q$ et pour tout élément de I, on ajoute une flèche de i vers cet élément, étiquetée par ε. Le seul état initial de ce nouvel automate étant i, on a bien $|I'| = 1$ et aucune flèche ne revient vers i par contruction. On a bien l'équivalence des langages. Dans l'autre sens, un automate standard fournit un automate classique équivalent, en utilisant la construction éliminant les éventuelles transitions spontanées.

[1] On remarquera sur les copies que la fermeture arrière (réalisée de façon symétrique) fournit en général bien plus de nouvelles transitions. Alors forcément, la fermeture avant et arrière...

Montrons maintenant que les langages rationnels sont reconnaissables par des constructions sur les automates standards. On construit assez facilement un automate standard reconnaissant $\{\varepsilon\}$, un automate standard reconnaissant $\emptyset$, un automate standard reconnaissant les lettres de l'alphabet A, il s'agit ensuite, à partir d'automates standards reconnaissant L (automate $\mathcal{A}$) et L' (automate $\mathcal{B}$), de construire des automates standards reconnaissant $L + L'$, $L.L'$ et L^* : (au passage, une construction sans dessin n'est *théoriquement* pas lue...)

i. pour l'union : si $\mathcal{A}$ et $\mathcal{B}$ sont standards, on vérifie facilement que l'automate qui réalise l'union est standard.

ii. pour la concaténation : on ajoute une transition spontanée de chaque état final de $\mathcal{A}$ vers l'unique état initial de $\mathcal{B}$, on supprime la qualité finale des états terminaux de $\mathcal{A}$ et la qualité initiale de l'état initial de $\mathcal{B}$.

iii. pour l'étoile : on ajoute une transition spontanée de chaque état final de $\mathcal{A}$ vers l'unique état initial. Tous les états finaux de $\mathcal{A}$ perdent leur qualité d'état final, et le seul étant initial devient final. On supprime ensuite les transitions spontanées aboutissant à cet état par fermeture avant.

1.2 Algo BMC

1. Si il existe deux transitions étiquetées par A et B de p à q, on les remplace par une seule transition étiquetée par $A \cup B$. On peut donc toujours ce ramener au cas d'une seule transition.
2. On ajoute un nouvel état i seul état initial et des transitions spontanées de i vers les anciens états initiaux de $\mathcal{A}$. On fait de même dans l'autre sens pour les états terminaux. On obtient ainsi un nouvel automate $\mathcal{A}'$. On remplace ensuite les étiquettes e (une lettre) des transitions par des étiquettes $\{e\}$. Voilà !
3. On remplace e par ab. Notons que $(ab)^*$ convient aussi, mais l'écriture de l'expression rationnelle s'en trouvera modifiée (et compliquée).
4. La transition (p_i, S, r_j) est remplacée par la transition (p_i, S', r_j) avec $S' = S \cup P_i Q^* R_j$. On fait de même pour toutes les transitions de ce type. On supprime ensuite l'état Q.
5. On vérifie aisément (l'écrire !) la légalité de la transformation [2]
6. Pour le premier automate, en enlevant successivement les états r, q puis p, on obtient l'expression $L(\mathcal{A}_1) = \big(a + (b(ab)^*b)\big)^*$ et pour le deuxième, en enlevant dans l'ordre r, s, q, p, on obtient $L(\mathcal{A}_2) = \big(a + b(abb + b)^*(a + ba)\big)^*.\big(b(b + abb)^*ab\big)$, expression que l'on peut simplifier ...
7. Dans le premier automate, si on commence par enlever q, puis r, puis p, on obtient l'expression : $L(\mathcal{A}_1) = \big(a + bb + ba(ba)^*bb\big)^*$.

1.3 Algorithme MNY

1. $R^{(0)}_{p,q}$ représente l'étiquette de la transition p, q, si elle existe. On obtient pour $\mathcal{A}_1$ la matrice $\begin{pmatrix} a & b & \emptyset \\ b & \emptyset & a \\ \emptyset & b & \emptyset \end{pmatrix}$ et pour $\mathcal{A}_2$ la matrice $\begin{pmatrix} a & b & \emptyset & \emptyset \\ a & b & a & \emptyset \\ \emptyset & \emptyset & \emptyset & b \\ a & b & \emptyset & \emptyset \end{pmatrix}$.
2. On trouve facilement en faisant un dessin :

$$R^{(k+1)}_{p,q} = R^{(k)}_{p,q} + R^{(k)}_{p,k+1}\big(R^{(k)}_{k+1,k+1}\big)^* R^{(k)}_{k+1,q}$$

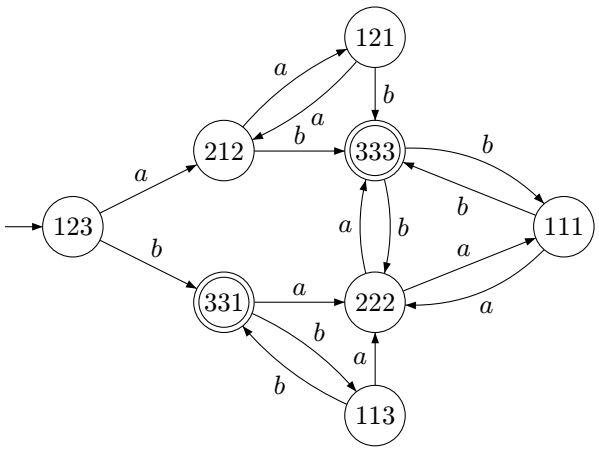
.

[2] la flemme gagne l'auteur !

3. On a $R_{p,q} = R^{(n)}_{p,q}$ si $p \neq q$ et $R_{p,p} = R^{(n)}_{p,p} + \varepsilon$. L'égalité précédente prouve que $R^{(k+1)}_{p,q}$ pour peu que les $R^{(k)}_{p,q}$ le soient. Comme $L(\mathcal{A}) = \bigcup_{p\in I, q\in T} R_{p,q}$, on obtient $L(\mathcal{A})$ rationnel et une expression du langage.
4. Pour le premier automate, on obtient, en renommant les états p, q, r par 1, 2, 3 (dans cet ordre), les matrices $R^{(1)} = \begin{pmatrix} a^+ & a^*b & a^+ \\ ba^* & ba^*b & a+ba^* \\ a^+ & a^*b & a^* \end{pmatrix}$,
$R^{(2)} = \begin{pmatrix} (a^*bb)^*a^* & (a^*bb)^*a^*b & a+(a^*b)(ba^*b)^*(a+ba^*) \\ (ba^*b)^*(ba^*) & (ba^*b)^+ & (ba^*b)^*(a+ba^*) \\ a^+ + (a^*bb)^+a^* & (a^*b)(ba^*b)^* & a^* + a^*b(ba^*b)^*(a+ba^*) \end{pmatrix}$ donc $L(\mathcal{A}_1) = R^{(3)}_{1,1} = R^{(2)}_{1,1} + R^{(2)}_{1,3}(R^{(2)}_{3,3})^*R^{(2)}_{3,1} = \ldots$ [3](qui contient ε), et pour le deuxième automate, on obtient une expression encore plus monstrueuse [4]

2 Facteurs itérants d'un langage

2.1 Automates réguliers à gauche

1. Pour $r \in \mathcal{V}(Q)$, $a \in A$, chaque état du vecteur r a un unique successeur dans $\mathcal{A}$ par a car $\mathcal{A}$ est déterministe complet, donc r a bien un unique successeur par a dans $\mathcal{A}_g$. $\mathcal{A}_g$ est donc déterministe complet. NB : on se fiche de l'éventuelle injectivité de η.
2. Soit $u = \alpha_1\alpha_2\ldots\alpha_n \in L(\mathcal{A})$. Le calcul dans $\mathcal{A}$
$$q_0 \xrightarrow{\alpha_1} q_1 \xrightarrow{\alpha_2} q_2 \xrightarrow{\alpha_3} q_3 \ldots q_{n-1} \xrightarrow{\alpha_n} q_n$$
$q_n \in T$ devient dans $\mathcal{A}_g$ un calcul qui contient le calcul précédent en tant que première composante des états-vecteurs, terminant dans un état de U par définition de ce dernier, et donc $u \in L(\mathcal{A}_g)$.
Réciproquement, soit $u = \alpha_1\alpha_2\ldots\alpha_n \in L(\mathcal{A}_g)$. La calcul dans $\mathcal{A}_g$
$$i = r_0 \xrightarrow{\alpha_1} r_1 \xrightarrow{\alpha_2} r_2 \xrightarrow{\alpha_3} r_3 \ldots r_{n-1} \xrightarrow{\alpha_n} r_n$$
avec $r_n \in U$ devient en projetant suivant la première composante un calcul réussi dans $\mathcal{A}$, et donc $u \in \mathcal{A}$.
3. Soit $f, g \in A^*$ tels que $i.f = i.g$, et $u \in A^*$.[5] Soit également $p \in Q$.
Par définition de i et de $\mathcal{A}_g$, on obtient, en lisant la "p-ième composante de $i.f$", $p.f = p.g$, mais aussi $(p.u).f = (p.u).g$, soit encore $p.(uf) = p.(ug)$. Ceci étant vrai quel que soit p, on obtient finalement, dans $\mathcal{A}_g$: $i.uf = i.ug$.
4. 

[3]sauf erreur
[4]Note du vérificateur : Mouais...
[5]je passe en notation multiplicative pour ne pas m'encombrer du η, c'est-à-dire que $\eta(s,m)$ est noté $s.m$.

5. Dans l'automate construit précédemment, on vérifie que l'état $(3,3,3)$ n'est pas régulier à gauche : En effet, on a $(3,3,3).a = (3,3,3).ba = (2,2,2)$ et $(3,3,3).aa = (1,1,1) \neq (2,2,2) = (3,3,3).aba$.
6. Soient $f, g \in A^*$ tels que $q_0.f = q_0.g$ et on montre la propriété par récurrence sur k : rien à montrer pour $k = 0$ et $k = 1$.
   Maintenant, supposons $q_0.f^n = q_0.g^n$. Alors $q_0.ff^n = q_0.fg^n$ car $\mathcal{A}$ est régulier à gauche. Donc $q_0.ff^n = q_0.gg^n$: Gagné !

2.2 Facteurs itérants

1. BIEN ENTENDU, l'énoncé tel quel est faux : considérer $L = \{a^n b^n | n \in \mathbb{N}\}$ ($Iter(L) = \emptyset$ alors que L est infini). Il fallait (suffisait...) ajouter l'hypothèse "L rationnel" pour obtenir un énoncé correct, à défaut d'être subtil.
2. (a) L_1 non rationnel : lemme de l'étoile, séparation d'états, réduction, etc...
   (b) Si $w \in L_1 \setminus \{\varepsilon\}$, on a $\varepsilon.w^*.\varepsilon \subseteq L_1$. Donc $w \in Iter(L_1)$. Réciproquement, si $v \in Iter(L_1)$, alors il existe $u, w \in A^*$ tels que $uv^*w \subseteq L_1$, donc $|uw|_a = |uw|_b$ et $|uvw|_a = |uvw|_b$ d'où l'on tire $|v|_a = |v|_b$, soit $v \in L_1 \setminus \{\varepsilon\}$.
   Ainsi, $Iter(L_1) = L_1$ n'est pas rationnel : le langage des facteurs itérants d'un langage non rationnel peut ne pas être rationnel.
   (c) L_2 non rationnel : gnagnagna... Ensuite, $c^+ \subseteq Iter(L_2)$ de façon évidente. Réciproquement, si $v \in Iter(L_2)$, alors $|v|_a = |v|_b$. Considérons $v = a^n c^m b^n$. Si $n \neq 0$, alors $v^2 = a^n c^m b^n a^n c^m b^n$ et peu importe le choix de u et w, $uv^*w \not\subseteq a^*b^*c^*$.
   Donc $v \in c^+$. Ainsi, $Iter(L_2) = c^+$ est rationnel : le langage des facteurs itérants d'un langage non rationnel peut être rationnel.
3. (a) Soit $p \in Q$. Comme $\mathcal{A}$ est supposé émondé, il existe $u, w \in A^*$ tels que
   $$q_0 \xrightarrow{u} p \xrightarrow{w} t$$
   et $t \in T$.
   Alors pour $v \in L(\mathcal{A}_p)$, $uv^*w \in L(\mathcal{A})$ donc $v \in Iter(L)$.
   (b) Soit $v \in Iter(L)$. Il existe $u, w \in A^* \mid uv^*w \in L$.
   On considère le calcul (réussi) dans $\mathcal{A}$ du mot uv^Nw :
   $$q_0 \xrightarrow{u} p_0 \xrightarrow{v} p_1 \ldots p_{N-1} \xrightarrow{v} p_N \xrightarrow{w} t$$
   Comme $|Q| = N$, on a alors l'existence de $i < j$ vérifiant $p_i = p_j$, et on a alors $v^k \in L(\mathcal{A}_p)$ où $k = j - i$.
   De plus $q_0.uv^i = p$. Donc uv^i est un contexte gauche de v.
4. On rédige pour $\mathcal{R}_2(L)$: le calcul $i \xrightarrow{u} s$ dans $\mathcal{A}_g$ nous donne le calcul $1 \xrightarrow{u} s_1$ dans $\mathcal{A}$, mais aussi $s_1 \xrightarrow{u} s_{s_1}$ dans $\mathcal{A}$, donc le calcul $1 \xrightarrow{u^2} s_{s_1}$. Pour les états finaux, on met s final si s_{s_1} est final dans $\mathcal{A}$.
5. Une inclusion est immédiate, l'autre découle directement de la question 3.
6. $\mathcal{J}_{p,k} \subseteq L(\mathcal{A}_{p,k})$ par définition de $\mathcal{A}_{p,k}$; montrons donc l'autre inclusion.
   Soit $w \in L(\mathcal{A}_{p,k})$. Il existe donc $r \in W_{p,k}$ tel que $i \xrightarrow{w} r$. Comme $r \in W_{p,k}$, il existe $v \in \mathcal{J}_{p,k}$ tel que $i.v = r$. De $i.v = i.w$ et comme l'automate $\mathcal{A}_{p,k}$ est régulier à gauche, on obtient d'après la question 1.6 le fait que pour tout $j \geq 0$, $i.w^j = i.v^j$. Mais $v \in \mathcal{J}_{p,k}$, donc il existe un mot $x \in A^*$ tel que pour tout $j \geq 0$, $p.v^j x \in T$ avec de plus $p.v^k = p$.
   En projetant les chemins obtenus dans $i.w^j = i.v^j$ suivant leur première composante, on en déduit que pour tout $g \geq 0$, $p.w^j x = p.v^j x \in T$ et d'autre part que $p.w^k = p.v^k = p$.
   Ainsi, $w \in \mathcal{J}_{p,k}$: gagné.
7. $Iter(L)$ est une union finie de langages rationnels, donc est rationnel.

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

1. Soit $v \in \mathcal{J}_{p,k}$. Alors il existe u, w tels que $uv^*w \subseteq L$, et $\delta(q_0, u) = p$ et $p.v^k = p$ par définition de $\mathcal{J}_{p,k}$. Sur le chemin de q_0 à p étiqueté par u on ne peut rencontrer plus de N états distincts, et donc, quitte à raccourcir u on peut supposer $|u| \leq N$.

   On considère maintenant les k états non nécessairement distincts $p, p.v, p.v^2, \ldots, p.v^{k-2}, p.v^{k-1}$, que l'on écrit comme vecteur d'états de Q.

   Lisant le mot w à partir de ces k états simultanément, on fait évoluer le vecteur :

$$\begin{pmatrix} p \\ p.v \\ \vdots \\ p.v^{k-1} \end{pmatrix} \xrightarrow{w} \begin{pmatrix} p.w \\ p.vw \\ \vdots \\ p.v^{k-1}w \end{pmatrix}$$

   Il y a au plus N^k vecteurs différents sur ce chemin, quitte à raccourcir w en w' on peut donc supposer $|w'| \leq N^k$.

   On vérifie facilement qu'on a toujours $uv^jw' \subseteq L$ pour $j \leq k$ et la propriété $p.v^k = p.v$ donne le résultat $uv^*w' \in L$.

2. La réponse à cette question utilise simplement le fait que $|\mathcal{V}(Q)| \leq N^N$.

3. Pour p et k donné,

   - Construire l'automate régulier à gauche $\mathcal{A}_{p,k}$ avec $W_{p,k}$ initialisé à vide.
   - Enumérer les mots $v \in A^*$ vérifiant $|v| \leq N^N$.
   - Si $p.v^k = p$, alors énumerer les mot $w \in A^*$ vérifiant $|w| \leq N^k$, et tester si $p.v^jw \in T$ pour $0 \leq j \leq k-1$. Si oui, alors ajouter $i.v$ à $W_{p,k}$.

   A priori, la construction de l'automate est en $\mathcal{O}(N^N)$ donc la complexité totale pour calculer $W_{p,k}$ est la même que celle pour calculer l'automate $\mathcal{A}_{p,k}$ en entier.

4. Pour tout $p \in Q$ et $1 \leq k \leq N$, construire $\mathcal{A}_{p,k}$. Un automate qui reconnait $Iter(L)$ est donné par l'union de ces automates, que l'on construit de façon usuelle.

   En recollant les morceaux, on peut donc borner la complexité par une fonction cauchemardesque du genre $\mathcal{O}(N^{2N+10})$.