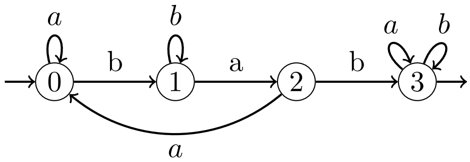
Solution - TD Feuille 2 - Automates finis et expressions rationnelles

Informatique Théorique 2 - Unité J1INPW11
Licence 3 - Université Bordeaux 1

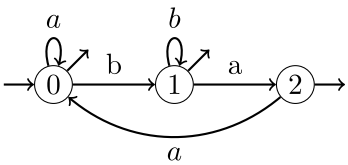
Solution de l'exercice 1 :

Pour tout l'exercice, on note $A = \{a, b\}$

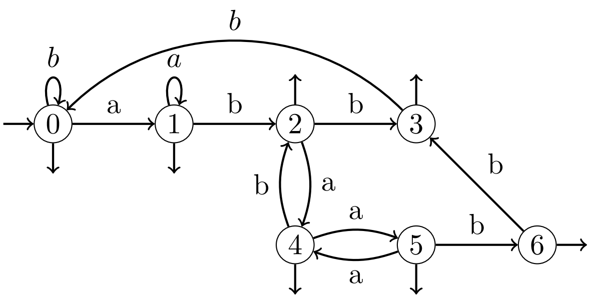
1. Expression régulière : $A^*(bab)A^*$.
   Automate :



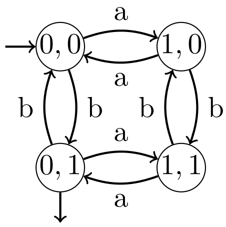
2. Expression régulière : $a^* + a^*b((aa^+)^*b)^*a^*$.
   Automate :



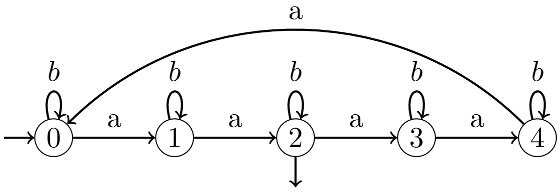
3. Automate :



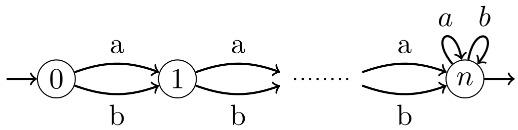
4. Automate :



5. Expression régulière : $b^*ab^*a(b^*ab^*ab^*ab^*ab^*a)^*b^*$.
   Automate :



6. Expression régulière : $(a+b)^n(a+b)^*$.
   Automate :



7. Le langage L des mots contenant autant de a que de b n'est pas pas régulier. On va le prouver en utilisant une technique de pompage (c'est aussi ce type de technique qui est utilisée dans la preuve du lemme de l'étoile) :
   On procède par l'absurde. Imaginons qu'il existe un automate $\mathcal{A} = (Q, q_I, F, \delta)$ pour L, on appelle N son nombre d'états. Considérons maintenant le mot $w = a^{N+1}b^{N+1}$, par définition de L, w est accepté par notre automate. Il existe donc une séquence d'états qui correspond à un execution de l'automate pour le mot w et qui est acceptante, c'est-à-dire une séquence $q_0, \ldots, q_{2N+2} \in Q$ telle que :
   – $q_0 = q_I$ l'état intial.
   – Pour tout $i \in \{0, N\}$: $(q_i, a) \to q_{i+1} \in \delta$.
   – Pour tout $i \in \{N+1, 2N+1\}$: $(q_i, b) \to q_{i+1} \in \delta$.
   – $q_{2N+2} \in F$ l'ensemble des états finaux.
   Or on sait par hypothèse qu'il existe exactement N états dans Q, donc il existe au moins deux indices $i_1 < i_2 \in \{0, N\}$ (ensemble de taille $N+1$) tels que $q_{i_1} = q_{i_2}$
   Il en découle que la séquence d'états $q_1, \ldots, q_{i_1}, q_{i_2+1}, \ldots, q_{2N+2}$ est acceptante pour le mot $a^{N+1-(i_2-i_1)}b^{N+1} \notin L$. Donc $\mathcal{A}$ accepte un mot qui n'est pas dans L, c'est une contradiction puisque $\mathcal{A}$ est censé être un automate qui reconnaît L.

Solution de l'exercice 2 :

Nous allons construire l'automate qui reconnait l'ensemble des représentations binaires des entiers positifs divisibles par 3. En lisant les mot de la gauche vers la droite.
Imaginons que l'automate ait déjà lu le mot w. Notons e_w l'entier représenté par w. Lorsque l'automate lit un nouvelle lettre, le mot lu devient alors :

$$\begin{cases} w.0 & \text{si la lettre lue est } 0; \\ w.1 & \text{si la lettre lue est } 1. \end{cases}$$

Les entiers représentés par ces deux mots sont alors égaux à :

$$\begin{cases} e_{w.0} = 2e_w + 0; \\ e_{w.1} = 2e_w + 1. \end{cases}$$

On cherche à determiner la divisibilité par 3 de e_w, pour cela, on écrit e_x sous la forme, $3.k$, $3.k+1$ et $3.k+2$.
Lorsque l'on lit un lettre, la forme de l'entier représenté par le mot change et son évolution est décrite par le tableau suivant :

e_w	$e_{w.0}$	$e_{w.1}$
$3.k+0$	$2.(3.k+0)+0 = 3k'$	$2.(3.k+0)+1 = 3k'+1$
$3.k+1$	$2.(3.k+1)+0 = 3k'+2$	$2.(3.k+1)+1 = 3k'+0$
$3.k+2$	$2.(3.k+2)+0 = 3k'+1$	$2.(3.k+2)+1 = 3k'+2$

Si l'on code chaque forme par un état, on obtient alors l'automate suivant :

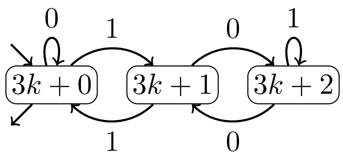


Figure 1 – Lecture de gauche à droite des entiers en base 2

Comme le mot de départ est le mot vide ϵ, il code l'entier 0 qui est de la forme $3.k+0$. L'état initial de l'automate est donc l'état $3.k+0$. Comme l'automate doit reconnaîtres les entiers divisibles par 3, il doit donc accepter les mots w dont l'entier e_w est de la forme $3.k+0$. L'automate a donc un seul état términal qui est l'état $3.k+0$.

Nous allons construire l'automate qui reconnait l'ensemble des représentations binaires des entiers positifs divisibles par 3. En lisant les mot de la droite vers la gauche.
Imaginons que l'automate ait déjà lu le mot w. Notons e_w l'entier représenté par w. Lorsque l'automate lit un nouvelle lettre, le mot lu devient alors :

$$\begin{cases} 0.w & \text{si la lettre lue est 0;} \\ 1.w & \text{si la lettre lue est 1.} \end{cases}$$

Les entiers représentés par ces deux mots sont alors égaux à :

$$\begin{cases} e_{0.w} = e_w + 0; \\ e_{1.w} = e_w + 2^{|w|}. \end{cases}$$

Nous nous interessons à la divisibilité par 3 de e_w, $e_{0.w}$ et $e_{1.w}$. Pour mener à bien cette étude, il faut étudier celle de $2^{|w|}$. On peut vérifier que $2^{|w|} = 3k+1$ si et seulement si $|w|$ est *pair* et $2^{|w|} = 3k+2$ si et seulement si $|w|$ est *impair*. On obtient le tableau récapitulatif suivant :

w	$0.w$	$1.w$
$e_w = 3.k + 0$ $\|w\|$ pair	$e_{0.w} = (3.k + 0) + 0 = 3k$ $\|0.w\|$ impair	$e_{1.w} = (3.k + 0) + (3k' + 1) = 3k'' + 1$ $\|1.w\|$ impair
$e_w = 3.k + 0$ $\|w\|$ impair	$e_{0.w} = (3.k + 0) + 0 = 3k$ $\|0.w\|$ pair	$e_{1.w} = (3.k + 0) + (3k' + 2) = 3k'' + 2$ $\|1.w\|$ pair
$e_w = 3.k + 1$ $\|w\|$ pair	$e_{0.w} = (3.k + 1) + 0 = 3k + 1$ $\|0.w\|$ impair	$e_{1.w} = (3.k + 1) + (3k' + 1) = 3k'' + 2$ $\|1.w\|$ impair
$e_w = 3.k + 1$ $\|w\|$ impair	$e_{0.w} = (3.k + 1) + 0 = 3k + 1$ $\|0.w\|$ pair	$e_{1.w} = (3.k + 1) + (3k' + 2) = 3k'' + 0$ $\|1.w\|$ pair
$e_w = 3.k + 2$ $\|w\|$ pair	$e_{0.w} = (3.k + 2) + 0 = 3k + 2$ $\|0.w\|$ impair	$e_{1.w} = (3.k + 2) + (3k' + 1) = 3k'' + 0$ $\|1.w\|$ impair
$e_w = 3.k + 2$ $\|w\|$ impair	$e_{0.w} = (3.k + 2) + 0 = 3k + 2$ $\|0.w\|$ pair	$e_{1.w} = (3.k + 2) + (3k' + 2) = 3k'' + 1$ $\|1.w\|$ pair

Si l'on code chaque forme par un état, on obtient alors l'automate suivant :

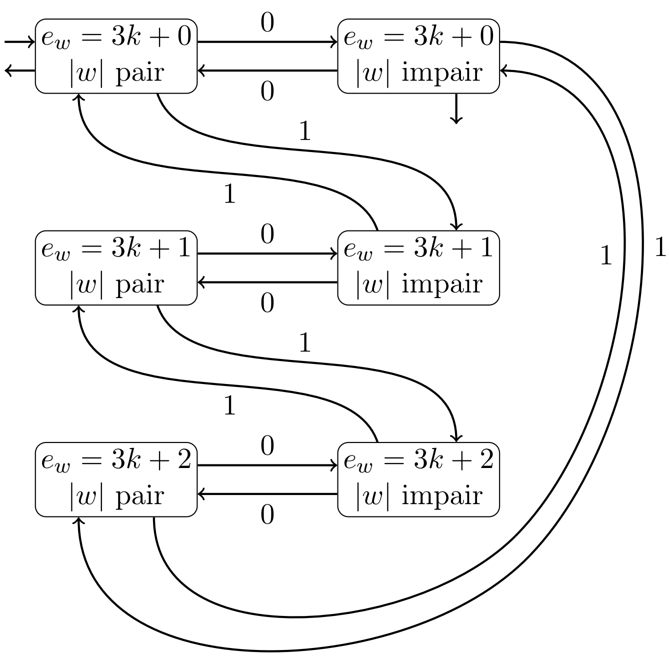


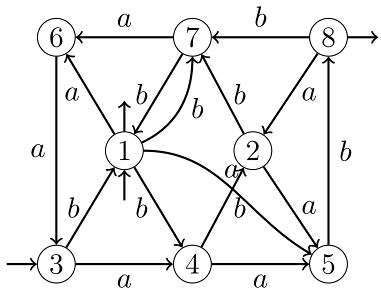
Figure 2 – Lecture de droite à gauche des entiers en base 2

Solution de l'exercice 3 :

1. On a :
   - $L_1 \cup L_2 = \{aa, baa, ab, aba, bbb, a\}$.
   - $L_1 \cap L_2 = \{ab\}$.
   - $L_1 \bullet L_2 = \{aabbb, aaa, aaab, baabbb, baaa, baaab, abbbb, aba, abab, ababbb, abaa, abaab\}$.
   - $L_1 \setminus L_2 = \{aa, baa, aba\}$.
   - $(L_2)^2 = L_2 \bullet L_2 = \{bbbbbb, bbba, bbbab, abbb, aa, aab, abbbb, aba, abab\}$.

- $L_2^* = (bbb + a + ab)^*$.
- $(a+b)^* \setminus L_1 = \epsilon + a + b + ba + bb + bab + bba + bbb + aaa + aab + abb + (a+b)^4(a+b)^*$

2. On donne un automate équivalent sans ϵ-transitions :



3. On construit un automate pour $L(\mathcal{A}_1) \cup L(\mathcal{A}_2)$:

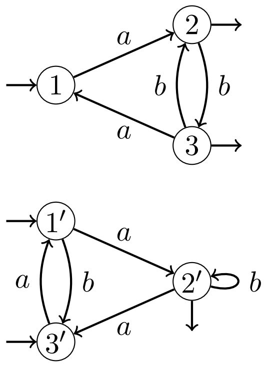


FIGURE 3 – Automate pour $L(\mathcal{A}_1) \cup L(\mathcal{A}_2)$

On calcule l'automate produit pour $L(\mathcal{A}_1) \cap L(\mathcal{A}_2)$:

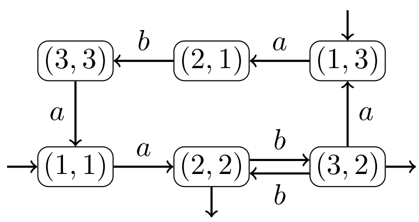


FIGURE 4 – Automate pour $L(\mathcal{A}_1) \cap L(\mathcal{A}_2)$

Pour $L(\mathcal{A}_1) \bullet L(\mathcal{A}_2)$, on commence par faire une version avec ϵ-transitions :

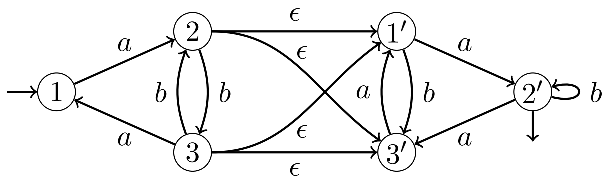


Figure 5 – Automate pour $L(\mathcal{A}_1) \bullet L(\mathcal{A}_2)$ (avec ϵ-transitions)

On élimine ensuite les ϵ-transitions :

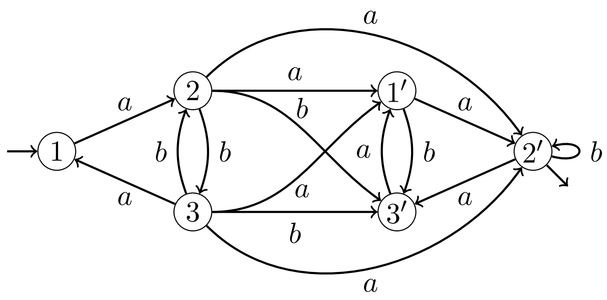


Figure 6 – Automate pour $L(\mathcal{A}_1) \bullet L(\mathcal{A}_2)$ (sans ϵ-transitions)

On fait de même pour $(L(\mathcal{A}_2))^2$, on commence par faire une version avec ϵ-transitions :

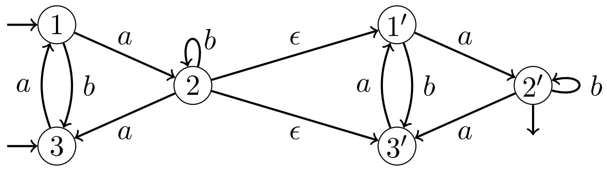


Figure 7 – Automate pour $(L(\mathcal{A}_2))^2$ (avec ϵ-transitions)

On élimine ensuite les ϵ-transitions :

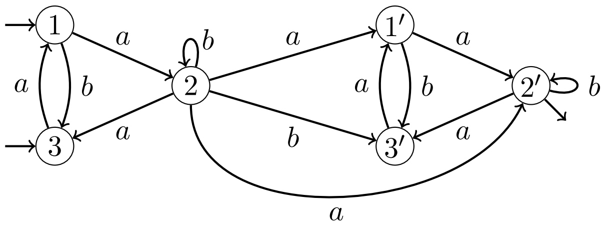


Figure 8 – Automate pour $(L(\mathcal{A}_2))^2$ (sans ϵ-transitions)

On fait de même pour $(L(\mathcal{A}_2))^*$, on commence par faire une version avec ϵ-transitions :

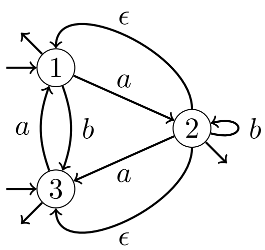


Figure 9 – Automate pour $(L(\mathcal{A}_2))^*$ (avec ϵ-transitions)

On élimine ensuite les ϵ-transitions :

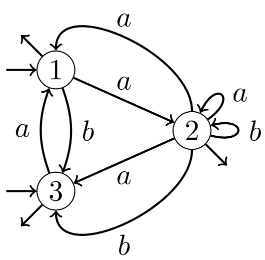


Figure 10 – Automate pour $(L(\mathcal{A}_2))^*$ (sans ϵ-transitions)

Pour $L(\mathcal{A}_1) \setminus L(\mathcal{A}_2)$, on doit construire un automate pour le complémentaire de $\mathcal{A}_2$ puis en faire l'intersection avec $\mathcal{A}_1$. On commence par déterminiser et compléter $\mathcal{A}_2$:
On en déduit un automate pour $(a + b)^* \setminus L(\mathcal{A}_2)$:

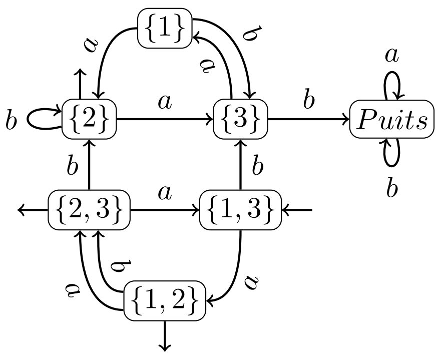


Figure 11 – Automate déterministe complet pour $L(\mathcal{A}_2)$

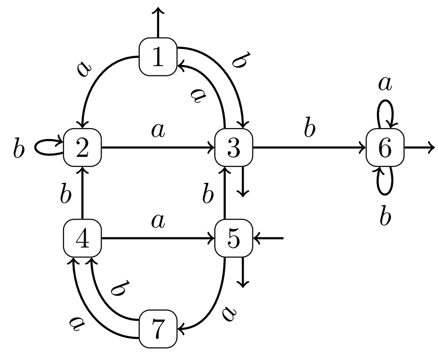


Figure 12 – Automate (déterministe complet) pour $(a+b)^* \setminus L(\mathcal{A}_2)$

Il nous reste maintenant à faire l'intersection avec $\mathcal{A}_1$ pour avoir $L(\mathcal{A}_1) \setminus L(\mathcal{A}_2)$

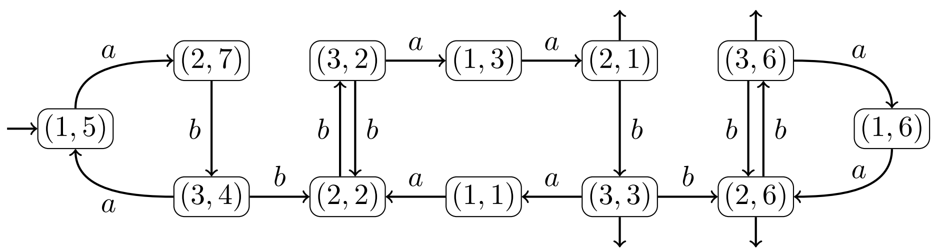


Figure 13 – Automate pour $L(\mathcal{A}_1) \setminus L(\mathcal{A}_2)$

Solution de l'exercice 4 :

On prouve la clôture par intersection. Soit L_1, L_2 deux langages réguliers, on montre que $L_1 \cap L_2$ est aussi régulier. Puisque L_1, L_2 sont réguliers on dispose de $\mathcal{A}_1 = (Q_1, I_1, F_1, \delta_1)$ et $\mathcal{A}_1 = (Q_2, I_2, F_2, \delta_2)$ des automates qui acceptent respectivement L_1 et L_2. On construit à partir de $\mathcal{A}_1$ et $\mathcal{A}_2$, un nouvel automate $\mathcal{B}$ qui accepte $L_1 \cap L_2$.

$\mathcal{B} = (Q, I, F, \delta)$ est l'automate produit défini comme suit : $Q = Q_1 \times Q_2, I = I_1 \times I_2, F = F_1 \times F_2$ et $\delta = \{((q_1, q_2), a) \to (r_1, r_2) \mid (q_1, a) \to r_1 \in \delta_1$ **et** $(q_2, a) \to r_2 \in \delta_2\}$. Il reste à montrer que $\mathcal{B}$ accepte bien $L_1 \cap L_2$. On prouve la double inclusion (i.e. $L_1 \cap L_2 \subseteq L(\mathcal{B})$ et $L(\mathcal{B} \subseteq L_1 \cap L_2))$.

Soit $w = a_1 \cdots a_n$ un mot. Par définition de $\mathcal{B}$: $w \in L(\mathcal{B})$ si et seulement si il existe une séquence d'états de Q : $(q_1^0, q_2^0), (q_1^1, q_2^1), \ldots, (q_1^n, q_2^n)$ telle que :

- $(q_1^0, q_2^0) \in I$
- $(q_1^n, q_2^n) \in F$
- Pour tout $i < n$, $((q_1^i, q_2^i), a_{i+1}) \to (q_1^{i+1}, q_2^{i+1}) \in \delta$.

En utilisant les définitions de I, F, δ on obtient que cette propriété est équivalente à l'existence de de deux séquences : $q_1^0, q_1^1, \ldots, q_1^n$ sur Q_1 et $q_2^0, q_2^1, \ldots, , q_2^n$ sur Q_2 telles que :

- $q_1^0 \in I_1$ et $q_2^0 \in I_2$
- $q_1^n \in F_1$ et $q_2^n \in F_2$
- Pour tout $i < n$, $(q_1^i, a_{i+1}) \to q_1^{i+1} \in \delta_1$ et $(q_2^i, a_{i+1}) \to q_2^{i+1} \in \delta_2$.

Autrement dit $w \in L(\mathcal{B})$ si et seulement si $w \in L(\mathcal{A}_1)$ et $w \in L(\mathcal{A}_2)$, on en déduit $L(\mathcal{B}) = L_1 \cap L_2$