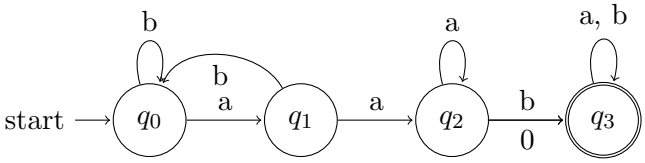
Trouver un automate déterministe pour chacun des langages suivants :

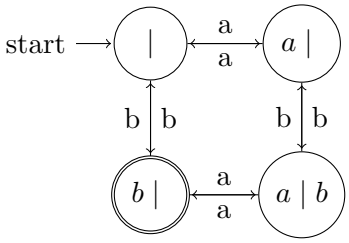
Exercice 1. Les mots sur l'alphabet $\{a, b\}$ contenant le facteur aab ou $aaab$.

Solution 1. On remarque qu'un mot contient aab ou $aaab$ s'il contient $aa(a^*)b$ donc :



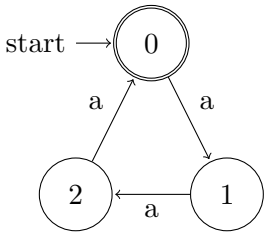
Exercice 2. Les mots sur l'alphabet $\{a, b\}$ contenant un nombre pair de a et impair b.

Solution 2. Selon la parité du nombre de a et de b vus on a 4 états :



Exercice 3. Les mots sur l'alphabet $\{a\}$ de longueur multiple de 3.

Solution 3. On maintient la taille modulo 3 du nombre de lettres lues :



Exercice 4. Pour chaque $d \in \mathbb{N}$ les mots sur l'alphabet $\{a\}$ de longueur multiples de d.

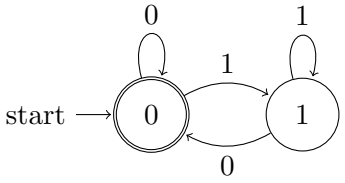
Solution 4. On maintient la taille modulo d du nombre de lettres lues. L'automate est

$\mathcal{A} =< \{a\}, \mathcal{Q}, \delta, q_0, \{q_0\} >$ avec $\mathcal{Q} = \{q_0, \dots q_{d-1}\}$ et $\delta(q_i, a) = \begin{cases} q_{i+1} & \text{si} i \neq d-1 \\ q_0 \end{cases}$.

On retrouve bien l'automate de la question précédente pour $d = 3$.

Exercice 5. Les représentations binaires d'entiers pairs. Ici entier est entendu au sens positif et les nombres sont donnés dans l'ordre gros-boutien (c'est à dire l'ordre normal de lecture des nombres, 1 puis 0 puis 1 puis 0 puis 1 puis 0 c'est le nombre binaire 101010 soit 42 en décimal).

Solution 5. Un entier binaire positif est pair si, et seulement si, il finit par un 0. On maintient donc dans l'état le dernier bit lu.



Exercice 6. Pour chaque $d \in \mathbb{N}$, les représentations binaires des entiers multiples de d.

Solution 6. On note $a\%b$ le reste de la division entière de a par b.

Tout d'abord le nombre $b_0 \dots b_k$ correspond à $\sum_{i \leq k+1} 2^{k-i} \times b_i$. Notons r_k le reste modulo d du nombre représenté par les $k+1$ premiers bits. On a $r_{k+1} = \sum_{i \leq k+1} 2^{k+1-i} \times b_i \% d = (2 \times r_k + v_{k+1})\%d$. Notre automate est donc $\mathcal{A} =< \{0,1\}, \mathcal{Q}, \delta, r_0, \{r_0\} >$ avec $\mathcal{Q} = \{r_0, \dots, r_{d-1}\}$ and $\delta(r_i, b) = r_{(2i+b)\%d}$.

Exercice 7. Pour chaque $d, c \in \mathbb{N}^2$, les représentations binaires des entiers de la forme $c + k \times d$ pour $k \in \mathbb{N}$.

Solution 7. Cette fois-ci il faut d'un côté un automate capable de détecter qu'un nombre est plus grand que c et de l'autre un automate qui reconnaisse les nombres de la forme $c + k \times d$ pour $k \in \mathbb{Z}$. Ensuite il suffit de faire l'intersection.

Pour faire un automate qui reconnaisse les nombres plus grands que c, il suffit de compter jusqu'à c. Pour cela On a $c+1$ états $q_0, \dots q_{c-1}$ et q_c qui stocke tous les nombres $\geq c$. La fonction de transition $\eta(q_i, b) = q_{min(2i+b,c)}$ convient.

Pour l'automate qui reconnaisse les nombres n tels que à $n \equiv c \pmod d$, on change juste l'état final de l'exercice précédent.

Tout cela nous donne l'automate $\mathcal{A} =< \{0,1\}, \mathcal{Q}, \delta, (0,0), \{(c, c\%d)\} >$ avec $\mathcal{Q} = \{0, \dots, d-1\} \times \{0, \dots, c\}$ et $\delta((i,j), b) = (min(2i+b, c), (2j+b)\%d)$.

Étant donné un langage rationnel $\mathcal{L}$ prouver que les langages suivants sont rationnels :

Exercice 8. $Init(\mathcal{L}) = \{u \mid \exists v : uv \in \mathcal{L}\}$

Solution 8. Soit $\mathcal{A} =< \Sigma, \mathcal{Q}, \delta, i, \mathcal{F} >$ un automate qui reconnaît $\mathcal{L}$, soit $Co - Access$ l'ensemble des états co-accessibles, c'est à dire les états $q \in \mathcal{Q}$ tel qu'il existe $c_1, \dots, c_n$ et $\delta(\dots \delta(q, c_1) \dots, c_n) \in \mathcal{F}$ alors $< \Sigma, \mathcal{Q}, \delta, i, CoAccess >$ reconnaît $Init(\mathcal{L})$. En effet, si $< \Sigma, \mathcal{Q}, \delta, i, CoAccess >$ accepte v alors il existe $c_1, \dots, c_n$ tel que $vc_1 \dots c_n$ soit accepté par $\mathcal{A}$ et donc $v \in Init(\mathcal{L})$.

Exercice 9. $Min(\mathcal{L}) = \{w \in \mathcal{L} \mid \nexists u \in \mathcal{L} : u \text{ préfixe propre de } w\}$

Solution 9. Soit $\mathcal{A} =< \Sigma, \mathcal{Q}, \delta, i, \mathcal{F} >$ un automate déterministe qui reconnaît $\mathcal{L}$. On supprime juste toutes les transitions sortants des états de $\mathcal{F}$. Ce nouvel automate reconnaît bien les mots de $Min(\mathcal{L})$: si un mot n'a pas de préfixe propre dans $\mathcal{L}$ il suivra le même parcours dans l'automate ; si un mot a un préfixe propre, il sera bloqué au premier préfixe propre. Il est important ici que l'automate soit déterministe car cela amène l'unicité du chemin d'un mot et donc l'absence d'état final sur le parcours (sauf le dernier) assure l'absence de préfixe propre dans le langage.

Exercice 10. $Max(\mathcal{L}) = \{w \in \mathcal{L} \mid wu \in \mathcal{L} \Rightarrow u = \epsilon\}$

Solution 10. Soit $\mathcal{A} =< \Sigma, \mathcal{Q}, \delta, i, \mathcal{F} >$ un automate déterministe qui reconnaît $\mathcal{L}$. On limite les états finaux de $\mathcal{A}$ aux états finaux qui ne sont pas co-accessibles. Encore une fois le déterminisme de $\mathcal{A}$ est important, sinon un mot pourrait être reconnu par deux chemins dont l'un fini sur un état co-accessible l'autre pas.

Exercice 11. $Cycle(\mathcal{L}) = \{uv \mid vu \in \mathcal{L}\}$

Solution 11. Soit $\mathcal{A} =< \Sigma, \mathcal{Q}, \delta, init, \mathcal{F} >$ un automate qui reconnaît $\mathcal{L}$ avec $\mathcal{Q} = \{q_1, \ldots, q_n\}$.

Notre construction va rajouter aux états de $\mathcal{Q}$ l'information de si nous avons déjà cyclé de quel état nous sommes parti. Notre automate est donc $< \Sigma, \mathcal{Q} \times \{0,1\} \times \mathcal{Q}, \eta, \mathcal{I}, \mathcal{F}' >$ avec

- Les états seront donc $\mathcal{Q} \times \{0,1\} \times \{\mathcal{Q}\}$.
- Les états initiaux seront tous les états qui n'ont pas cyclés et qui sont marqué eux mêmes comme initiaux $\mathcal{I} = \{(q_j, 0, q_j)\}$.
- On définit la transition de la façon suivante $\eta((q_i, b, q_j), c) = (\delta(q_i, c), b, q_j)$ mais $\eta(q_i, 0, q_j)$ contient aussi $i, 1, q_j$ si $q_i \in \mathcal{F}$.
- Enfin les états finaux sont $\mathcal{F}' = \{(q_j, 1, q_j)\}$.

Cet automate reconnaît bien le bon langage car à tout chemin acceptant peut être découpé en une partie d'un état $(q_j, 0, q_j)$ à un état $(q_f, 0, q_j)$ puis une deuxième partie $(init, 1, q_j)$ à $(q_j, 1, q_j)$ donc on retrouve un chemin dans l'automate initial. de $init$ à q_j puis de q_j à $q_f \in \mathcal{F}$.

Exercice 12. $\frac{1}{2}\mathcal{L} = \{u \mid \exists v : uv \in \mathcal{L} \text{ et } |v| = |u|\}$

Solution 12. Soit $\mathcal{A} =< \Sigma, \mathcal{Q}, \delta, init, \mathcal{F} >$ un automate déterministe qui reconnaît $\mathcal{L}$. L'idée c'est de faire un automate dérivé de $\mathcal{A}$ qui en lisant un mot m maintient d'un côté l'éventuel état $\delta(init, m)$ et de l'autre l'ensemble des états e tel que $\exists v : |v| = |u| \wedge \delta(e, v) \in \mathcal{F}$.

Notre automate est $< \Sigma, \mathcal{Q} \times 2^{\mathcal{Q}}, \eta, init', \mathcal{F}' >$ où :

- $init' = init, \mathcal{F}$
- $\mathcal{F}' = \{e, v \in \mathcal{Q} \times 2^{\mathcal{Q}} \mid e \in v\}$
- $\eta((e, v), c) = (\delta(e, c), \{q \mid \exists c : \delta(q, c) \in v\})$.