Chapitre 7

Automates

Option informatique MP1, MP2 & MP3

Ce chapitre met en place des langages caractérisés par le fait qu'il existe une méthode simple pour tester l'appartenance d'un mot à ce langage. On verra que ces langages sont en fait les langages rationnels.

I Automates déterministes

I.1 Transitions

La première étape va être de donner un cadre aux commandes et états.

Définition 1 - Machine

Une machine est un triplet $(\mathcal{A}, S, \delta)$ où

- $\mathcal{A}$ est un ensemble fini, l'alphabet, ses éléments seront les lettres,
- S est un ensemble fini, l'ensemble des états,
- δ est une fonction de $S \times \mathcal{A}$ vers S, la fonction de transition.

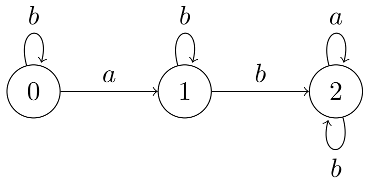
Une machine est caractérisée par la fonction de transition ; en effet l'alphabet et l'ensemble des états sont les composantes de son ensemble de départ. Voici un exemple :

δ	0	1	2
a	1	1	2
b	0	2	2

Définition 2 - Transition

Une transition d'une machine $(\mathcal{A}, S, \delta)$ est un triplet (s, a, s') tel que $\delta(s, a) = s'$.
On notera aussi $s.a = s'$ ou $s \xrightarrow{a} s'$.
s est l'origine de la transition, s' en est l'extrémité et a son étiquette.

Une représentation peut être faite par un graphe valué par $\mathcal{A}$: S est l'ensemble des sommets et les arêtes sont les transitions.
Contrairement aux graphes vus jusqu'à présent les arêtes d'un sommet vers lui-même sont possibles et il peut y avoir plusieurs arêtes d'un sommet s vers un sommet s' : elles auront alors des étiquettes distinctes.
On peut noter que, si n est le cardinal de $\mathcal{A}$, il y a n arêtes sortantes depuis tout sommet, étiquetées par les n lettres de l'alphabet.



Une suite finie de commandes est une suite de lettres donc un mot.
On peut associer une transition à un mot en composant les transition des lettres : on fait le choix de lire les lettres de la gauche vers la droite.

Définition 3 - Action d'un mot

Si $M = (\mathcal{A}, S, \delta)$ est une machine sur le langage $\mathcal{A}$ on prolonge la fonction de transition à $\mathcal{A}^*$ tout entier par la fonction δ^* de $S \times \mathcal{A}^*$ vers S avec

- $\delta^*(s, \varepsilon) = s$ pour tout $s \in S$
- $\delta^*(s, u.x) = \delta\big(\delta^*(s, u), x\big)$ pour $u \in \mathcal{A}^*$ et $x \in \mathcal{A}$.

On notera encore $\delta^*(s, u) = s.u$ ou $s \xrightarrow{u} s'$, si $s' = s.u$.
On peut donc écrire $s.(u.x) = (s.u).x$ pour $u \in \mathcal{A}^*$ et $x \in \mathcal{A}$. Pour $u = x_1 x_2 \cdots x_n$, on a $s.u = \big((\cdots((s.x_1).x_2).\cdots.x_{n-1}).x_n\big)$.
Si on pose $s_1 = \delta(s, x_1)$, $s_2 = \delta(s_1, x_2)$, $\ldots$, $s' = s_n = \delta(s_{n-1}, x_n)$ alors $(s, s_1, s_2, \ldots, s_n)$ est le **parcours** (ou **calcul**) de u à partir de s.
On notera ce parcours sous la forme

$$s \xrightarrow{x_1} s_1 \xrightarrow{x_2} s_2 \xrightarrow{x_3} \quad \cdots \quad \xrightarrow{x_{n-1}} s_{n-1} \xrightarrow{x_n} s_n$$

Théorème 1 - Produit

$s.(u.v) = (s.u).v$ pour tous $u, v \in \mathcal{A}^*$.

Exercice 1 - Démonstration
Prouver le théorème 1

I.2 Automates déterministes complets

Pour l'instant les parcours de la machine se font sans condition de point de départ et tous les parcours ont la même valeur. Nous allons enrichir la machine en lui ajoutant un point de départ et des sommets d'arrivée souhaités.

Définition 4 - Automate déterministe complet

Un automate fini déterministe ou A.F.D. (DFA en anglais)
est un quintuplet $Q = (\mathcal{A}, S, \delta, s_0, T)$ où

- $\mathcal{A}$ est un alphabet (fini),
- S est un ensemble fini d'états,
- δ est une application de $S \times \mathcal{A}$ vers S : la fonction de transition.
- s_0 est un élément de S, l'état initial,
- T est une partie de S, l'ensemble des états finaux,
  on dit aussi accepteurs, terminaux voire même finals !

Les 3 premiers élément du quintuplet forment donc une machine.

Un automate va servir à tester les mots : a-t-on $s_0.u \in T$?
On dit alors que u est **reconnu** par l'automate.

Définition 5 - Langage reconnu

- Le langage reconnu (ou accepté) par un automate $Q = (\mathcal{A}, S, \delta, s_0, T)$ est l'ensemble $L(Q)$ de tous les mots reconnus par Q.

$$L(Q) = \{u \in \mathcal{A}^* \ ; \ s_0.u \in T\}$$

- Un langage L est **reconnaissable** s'il existe un a.f.d. complet Q tel que $L = L(Q)$.

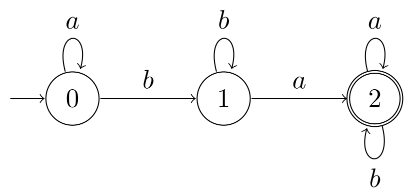
Exercice 2 - Premières propriétés
$Q = (\mathcal{A}, S, \delta, s_0, T)$ est un automate.

- Si $T = \emptyset$ alors $L(Q) = \emptyset$.
- Si $T = S$ alors $L(Q) = \mathcal{A}^*$.
- $L(Q)$ contient ε si et seulement si s_0 appartient à T.

Pour représenter graphiquement un automate

- l'état initial est indiqué par une flèche rentrante
- un état terminal est entouré d'un double cercle ou est marqué par une flèche sortante.

Dans l'exemple donné ci-dessus,on pose $s_0 = 0$ et $T = \{2\}$.

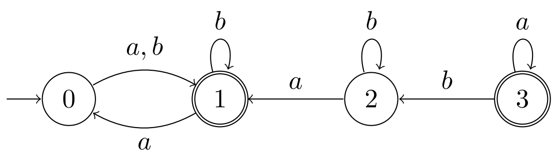


Le langage reconnu est l'ensemble des mots contenant ba.

I.3 Émondage

Nous allons dans la suite déterminer des automates différents en conservant le langage. La première possibilité est d'enlever les états non utilisés.

Dans l'automate ci-dessous on voit qu'aucun parcours ne passera par les états 2 ou 3.



On va essayer de les retirer.

Définition 6 - Accessibilité

$Q = (\mathcal{A}, S, \delta, s_0, T)$ est un automate déterministe complet.
Un état $s \in S$ est dit accessible s'il existe $u \in \mathcal{A}^*$ tel que $s = s_0.u$.

Les états accessibles sont les seuls états visibles depuis s_0.

Théorème 2 - États accessibles

Si S' est l'ensemble des états accessibles d'un automate $Q = (\mathcal{A}, S, \delta, s_0, T)$ alors, pour tout $s \in S'$ et pour tout $a \in \mathcal{A}$, $\delta(s, a)$ appartient à S'.

Exercice 3 - Démonstration
Prouver le théorème 2

Définition 7 - Automate émondé

Si S' est l'ensemble des états accessibles d'un automate $Q = (\mathcal{A}, S, \delta, s_0, T)$, l'automate émondé associé à Q est l'automate déterministe complet

$$Q' = (\mathcal{A}, S', \delta, s_0, T \cap S')$$

Théorème 3 - Conservation du langage

On a $L(Q) = L(Q')$.

Exercice 4 - Démonstration
Prouver le théorème 3

I.4 Opérations ensemblistes

Nous allons prouver ici quelques stabilités des langages reconnaissables.

On rappelle que le complémentaire d'un langage L sur $\mathcal{A}$ est $\overline{L} = \mathcal{A}^* \setminus L$.

Théorème 4 - Complémentaire

Le complémentaire d'un langage reconnaissable est reconnaissable.

Exercice 5 - Démonstration
Prouver le théorème 4 ; on pourra remplacer T par $S \setminus T$.

Pour l'union et le produit nous auront besoin d'une construction de machine.

Définition 8 - Produit de machines

Si $M_1 = (\mathcal{A}, S_1, \delta_1)$ et $M_2 = (\mathcal{A}, S_2, \delta_2)$ sont deux machines sur le même alphabet leur produit $M_1 \odot M_2$ est la machine
$(\mathcal{A}, S_1 \times S_2, \delta_1 \times \delta_2)$ avec $\delta_1 \times \delta_2\big((s,t), a\big) = \big(\delta_1(s,a), \delta_2(t,a)\big)$.

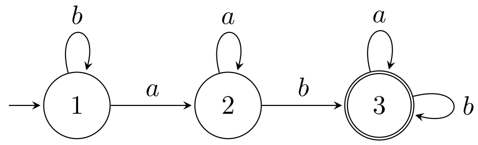
Ainsi, en notant de la même façon l'action des lettres dans M_1, M_2 et $M_1 \odot M_2$,
on a $(s,t).x = (s.x, t.x)$ pour toute lettre de $\mathcal{A}$ d'où, par récurrence,
pour tout mot de $\mathcal{A}^*$, $(s,t).u = (s.u, t.u)$.

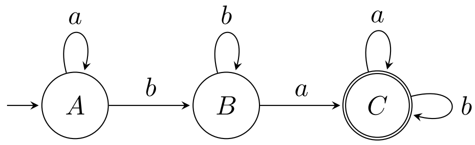
Théorème 5 - Intersection et union

L'intersection de deux langages reconnaissables est reconnaissable.
L'union de deux langages reconnaissables est reconnaissable.

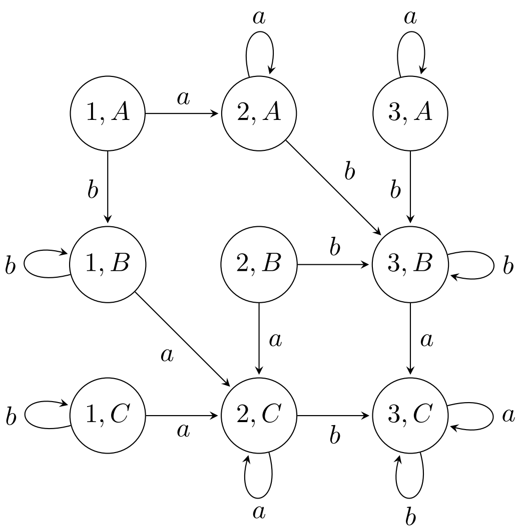
Exercice 6 - Démonstration
Prouver le théorème 5 ; on choisira T selon le cas.

Exemple Les automates suivants reconnassent respectivement L_1, ensemble des mots contenant ab et L_2, ensemble des mots contenant ba.

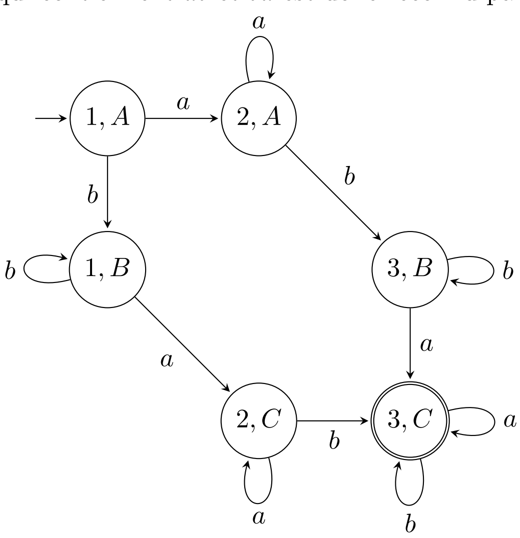




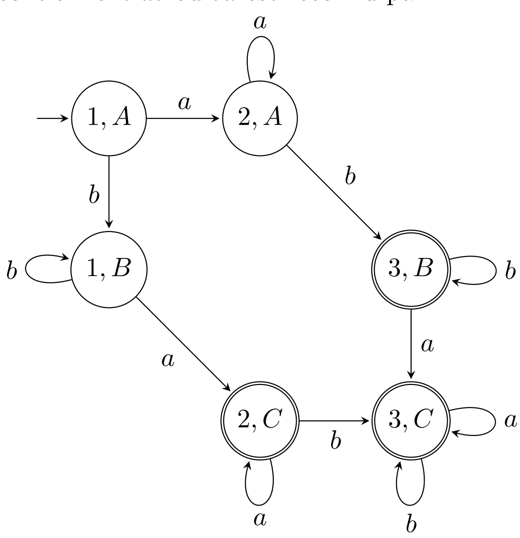
Le produit des machines associées est



Après élagage, le langage ($L_1 \cap L_2$) des mots qui contiennent ab et ba est donc reconnu par

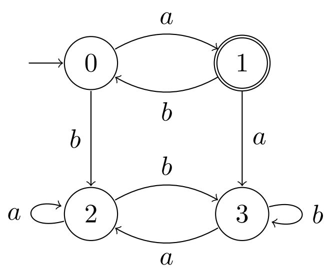


et le langage ($L_1 \cup L_2$) des mots qui contiennent ab ou ba est reconnu par



I.5 Automates déterministes incomplets

Nous avons vus qu'on pouvait enlever les états invisibles depuis l'état initial. Il y a d'autres états qu'on peut souhaiter ignorer.



Les états 2 et 3 représentent une sorte de piège : dès qu'on y parvient on ne pourra jamais revenir à un état terminal. La tentation de les enlever se heurte au fait qu'alors il existerait des transitions manquantes.

Définition 9 - Automate déterministe incomplet

Un automate fini déterministe incomplet est un quintuplet $Q = (\mathcal{A}, S, \delta, s_0, T)$

- où $\mathcal{A}$ est un alphabet (fini),
- S est un ensemble fini d'états,
- δ est une application définie sur une **partie** de $S \times \mathcal{A}$ vers S : la fonction de transition.
- s_0 est un élément de S, l'état initial,
- T est une partie de S, l'ensemble des états finaux,

Définition 10 - Mots bloquants

$Q = (\mathcal{A}, S, \delta, s_0, T)$ est un automate déterministe incomplet.
$u = u_1 u_2 \cdots u_p$ est un mot sur $\mathcal{A}$.
Si $s_0.u_1 = s_1$, $s_1.u_2 = s_2$, $\ldots$, $s_{k-1}.u_k = s_k$ sont définis mais $\delta(s_k, u_{k+1})$ n'est pas défini (avec $k < p$) on dit que u est un **mot bloquant**.
Le langage reconnu par un automate incomplet est l'ensemble des mots u non bloquants tels que $s_0.u$ appartient à T.

Définition 11 - Co-accessibilité

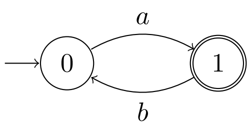
$Q = (\mathcal{A}, S, \delta, s_0, T)$ est un automate déterministe.
Un état $s \in S$ est dit co-accessible s'il existe $u \in \mathcal{A}^*$ non bloquant tel que $s.u \in T$.

Théorème 6 - Conservation du langage

Si S' est l'ensemble des états co-accessibles de Q alors
$Q' = (\mathcal{A}, S', \delta, s_0, T \cap S')$ est un automate déterministe incomplet tel que $L(Q) = L(Q')$.

Exercice 7 - Démonstration
Prouver le théorème 6

Dans l'exemple ci-dessus on aboutit ainsi à l'automate



Comme on a élargi la définition des automates on peut imaginer que l'on a aussi augmenté le nombre de langages reconnaissables. En fait il n'en est rien.
L'idée est d'ajouter un état-puits qui va recevoir toutes les transitions manquantes et qui reste stable par toutes les lettres. On construit ainsi un automate complet.

Définition 12 - Complétion

$Q = (\mathcal{A}, S, \delta, s_0, T)$ est un automate incomplet.
On choisit p n'appartenant pas à S et on définit $S' = S \cup \{p\}$.
On prolonge δ en δ' défini sur $S' \times \mathcal{A}$ par

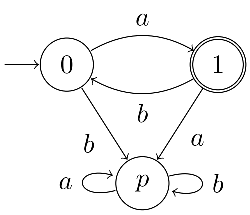
- $\delta'(s, x) = \delta(s, x)$ pour $s \in S$ si $\delta(s, x)$ existe,
- $\delta'(s, x) = p$ pour $s \in S$ si $\delta(s, x)$ n'est pas défini et
- $\delta'(p, x) = p$

Théorème 7 - Conservation du langage

$Q' = (\mathcal{A}, S', \delta', s_0, T)$ est un automate déterministe complet tel que $L(Q) = L(Q')$.

Exercice 8 - Démonstration
Prouver le théorème 7

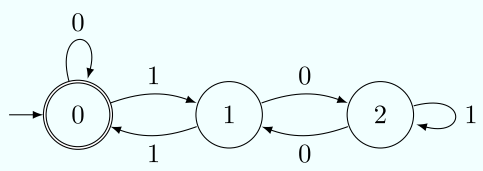
L'exemple ci-dessus donne finalement l'automate



I.6 Exercices

Exercice 9 - 3 occurences
$\mathcal{A}$ désigne l'alphabet $\{a, b\}$ et L l'ensemble des mots qui contiennent au moins 3 occurrences de la lettre a : $L = \{u \in \Sigma^* / |u|_a \geqslant 3\}$.
Donner une expression rationnelle dénotant L.
Décrire un automate fini reconnaissant L.

Exercice 10 - Divisibilité par trois
Prouver que l'automate suivant teste la divisibilité par trois d'un nombre exprimé en binaire.



Que se passe-t-il si on change l'état final ?

Exercice 11 - Petits automates
Énumérer tous les automates déterministes complets à 2 états sur $\mathcal{A} = \{a, b\}$ et le langage qu'ils reconnaissent.

Exercice 12 - Alphabet à une lettre
Déterminez la forme des automates déterministes sur un alphabet à une lettre.
En déduire la structure des langages reconnaissables sur un alphabet à une lettre.

Exercice 13 - États co-accessibles
À quelle condition sur le langage reconnu un automate complet n'a-t-il que des états co-accessibles ?

II Automates non-déterministes

Les automates vus jusqu'à présent envoyait un état vers un autre état bien déterminé (ou vers aucun état) sous l'action d'une lettre.
Nous allons maintenant lever cette contrainte d'un état unique et considérer qu'une lettre peut déterminer plusieurs transitions depuis un état.
Ce non-déterminisme sera à l'œuvre aussi au départ : on autorisera plusieurs états initiaux.

II.1 Définitions

Définition 13 - Automate fini non-déterministe

Un automate fini non-déterministe ou AFND (NFA en anglais)
est un quintuplet $(\mathcal{A}, S, \Delta, I, T)$ composé de :

- $\mathcal{A}$, un alphabet,
- S, l'ensemble fini des états de l'automate,
- Δ une application de $S \times \mathcal{A}$ dans $\mathcal{P}(S)$,
  c'est la fonction de transition.
- $I \subset S$, l'ensemble des états initiaux,
- $T \subset S$, l'ensemble des états acceptants,

Un automate déterministe peut ainsi être considéré comme un cas particulier d'automate non–déterministe. Dans le cas d'un automate déterministe, pour tout couple $(s, x) \in S \times \mathcal{A}$, $\Delta(s, x)$ admet exactement un élément dans le cas des automates complets ou au plus un élément dans le cas des automates incomplets.

On peut remplacer la fonction de transition par son graphe $G_\Delta = \big\{(s, x, s')\ ;\ s' \in \Delta(s, x)\big\}$.
G_Δ est une partie de $S \times \mathcal{A} \times S$, c'est l'ensemble des transitions.
Δ est alors défini par $\Delta(s, x) = \big(\{s\} \times \{x\} \times S\big) \cap G_\Delta = \big\{s' \in S\ ;\ (s, x, s') \in G_\Delta\big\}$.

On notera encore la transition $(s, x, s') \in G_\Delta$ par $s \xrightarrow{x} s'$.

Définition 14 - Calcul

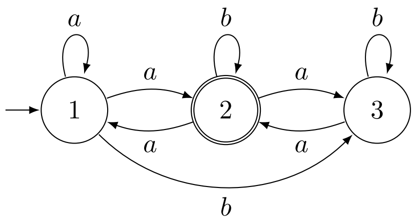
Un calcul de l'automate est une suite (finie) de transitions telle que l'origine de chacune est l'extrémité de la précédente. Il sera noté

$$s \xrightarrow{x_1} s_1 \xrightarrow{x_2} s_2 \xrightarrow{x_3} \quad \cdots \quad \xrightarrow{x_{n-1}} s_{n-1} \xrightarrow{x_n} s_n$$

- s est l'origine du calcul,
- s_n est l'extrémité du calcul,
- $u = x_1 x_2 \cdots x_n$ est l'étiquette du calcul.

La principale différence avec les automates déterministe est qu'il y avait une bijection entre les parcours et leur étiquette.
Dans le cas d'un automate non-déterministe chaque calcul a une étiquette unique mais un même mot peut être associé à plusieurs calculs depuis une même origine.

Exemple : dans l'automate



$1 \xrightarrow{a} 1 \xrightarrow{a} 1 \xrightarrow{b} 3$,
$1 \xrightarrow{a} 1 \xrightarrow{a} 2 \xrightarrow{b} 2$,
$1 \xrightarrow{a} 2 \xrightarrow{a} 1 \xrightarrow{b} 3$,
$1 \xrightarrow{a} 2 \xrightarrow{a} 3 \xrightarrow{b} 3$
sont les calculs d'étiquette aab.

Définition 15 - Langage associé

- Un calcul est **réussi** si son origine appartient à I et son extrémité appartient à T.
- Un mot est **reconnu** (ou accepté) par l'automate s'il est l'étiquette d'au moins un calcul réussi.
- Le **langage reconnu** (ou langage accepté) par un automate M est l'ensemble, $L(M)$, des mots reconnus par l'automate.

II.2 Déterminisation

Comme les automates déterministes sont des cas particuliers d'automates non-déterministes tout langage reconnaissable est reconnu par un automate non-déterministe.
Par contre il n'est pas évident qu'un langage reconnu par un automate non-déterministe soit reconnaissable (par un automate déterministe). C'est pourtant le cas ; la démonstration s'appuie sur la transformation d'un automate non-déterministe en un automate déterministe en conservant le langage.

Définition 16 - Automate des parties

$Q = (\mathcal{A}, S, \Delta, I, T)$ est un automate non-déterministe.
L'automate des parties associé à Q est $Q' = (\mathcal{A}, \mathcal{P}(S), \delta, I, \mathcal{F})$ avec

- $\delta(E, x) = \bigcup\limits_{s \in E} \Delta(s, x) = \{s' \in S \ ; \ \exists s \in E, \ (s, x, s') \in G_\Delta\}$.
- $\mathcal{F} = \{E \subset S \ ; \ E \cap T \neq \emptyset\}$

Théorème 8 - Conservation du langage

- Q' est un automate déterministe complet.
- $L(Q') = L(Q)$.

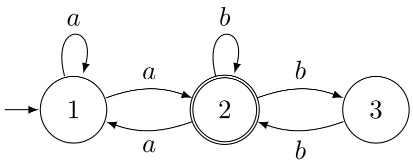
On dit aussi que Q' est le déterminisé de Q.

Exercice 14 - Démonstration
Prouver le théorème 8

Si l'automate initial est non-déterministe avec n états, le déterminisé admet 2^n états.
Dans le calcul pratique on ne calculera les images des parties de S que pour les parties accessibles : on part de I on calcule $\delta(I, x)$ pour les différentes lettres de Σ puis les images de ces derniers ensembles et ainsi de suite : on calcule seulement l'automate émondé.

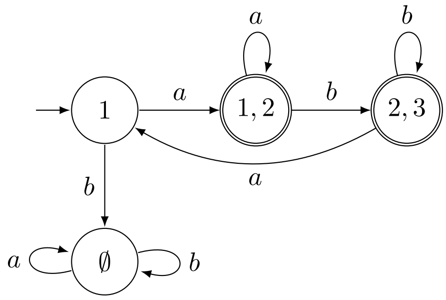
Exemple :

On part de l'automate non déterministe



On calcule seulement les transitions depuis les états utiles (les états accessibles).

Q'	$\{1\}$	$\{1,2\}$	$\{2,3\}$	$\emptyset$
a	$\{1,2\}$	$\{1,2\}$	$\{1\}$	$\emptyset$
b	$\emptyset$	$\{2,3\}$	$\{2,3\}$	$\emptyset$



II.3 Exercices

Exercice 15 - Construction
Construire un automate reconnaissant tous les mots qui finissent par aba.
Déterminiser l'automate obtenu.

Exercice 16 - Le pire peut arriver
Donner un automate (non-déterministe) à $n+1$ états qui reconnaît L, le langage sur $\mathcal{A} = \{a, b\}$ des mots de n lettres au moins qui finissent par $a.u$ avec u de longueur $n-1$.
Prouver que tout automate déterministe qui reconnaît L admet au moins 2^n états.

Exercice 17 - Langage transposé
$Q = (\mathcal{A}, S, \Delta, I, T)$ est un automate non déterministe.
On définit $Q^T = (\mathcal{A}, S, \Delta^T, T, I)$ avec Δ^T tel que $G_{\Delta^T} = \big\{(s, x, s')\ ;\ (s', x, s) \in G_\Delta\big\}$.
Q^T est l'automate obtenu en inversant les transitions.
Pour tout mot $u = u_1 u_2 \cdots u_n$ le miroir de u est $u^T = u_n u_{n-1} \cdots u_1$.
Pour tout langage L le le langage miroir de L, L^T, est $L^T = \{u^T\ ;\ u \in L\}$.

Prouver que $L(Q^T) = \big(L(Q)\big)^T$.

Exercice 18 - Le barman aveugle
On dispose de 4 jetons, chacun ayant une face noire et une face blanche. Un joueur (le barman) a les yeux bandés. Son but est de retourner les 4 jetons sur la même couleur (dès que les 4 jetons sont retournés la partie s'arrête et le barman a gagné). Pour cela, il peut retourner à chaque tour 1, 2 ou 3 jetons. Un autre joueur perturbe le jeu en tournant le plateau sur lequel reposent les jetons d'un quart de tour, d'un demi-tour ou de trois quarts de tour entre chaque opération du barman.
Montrer que le barman a une stratégie gagnante, c'est–à–dire que quoi que fasse celui qui tourne le plateau, le barman gagnera.

III L'algorithme de Berry-Sethi

Les langages rationnels sont facilement définissables globalement et les langages reconnaissables admettent un test d'appartenance simple. Ces deux aspects sont en fait utilisables en même temps.

Théorème 9 - Théorème de Kleene

Les langages rationnels sont les langages reconnaissables.

Nous allons démontrer dans cette partie une implication : tout langage reconnaissable et rationnel. La réciproque, plus facile, est hors-programme et sera démontrée dans le chapitre suivant.

Nous utiliserons un résultat du chapitre précédent :

Théorème 10 - rappel

Si e est une expression régulière linéaire (c'est-à-dire dont toutes les lettres employées sont distinctes) alors le langage qu'elle dénote est un langage local.

III.1 Automate local

L est un langage local sur $\mathcal{A}$ défini par (P, F, S, b).

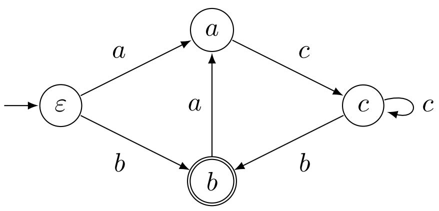
Définition 17 - Automate de Glushkov

L'automate de Glushkov associé à L est l'automate incomplet
$Q = (\mathcal{A}, \mathcal{A} \cup \{\varepsilon\}, \delta, \varepsilon, S')$ où δ est définie par

- $\delta(\varepsilon, x)$ existe si et seulement si $x \in P$, alors $\delta(\varepsilon, x) = x$,
- $\delta(x, y)$ existe si et seulement si $xy \in F$, alors $\delta(x, y) = y$.

et $S' = S \cup \{\varepsilon\}$ si b vaut vrai c'est-à-dire si $\varepsilon \in L$, $S' = S$ sinon.

Exemple L est le langage local sur $\mathcal{A} = \{a, b, c\}$ défini par $P = \{a, b\}$, $S = \{b\}$, $F = \{ac, ba, cb, cc\}$ et b vaut faux.



On remarquera que toutes les transitions de même étiquette ont la même extrémité.

Définition 18 - Automate local

Un automate déterministe $Q = (\mathcal{A}, S, \delta, s_0, T)$ est local si, pour tout $x \in \mathcal{A}$, $\delta(s, x)$ est indépendant de s.

L'automate de Glushkov est donc un automate local

Exercice 19 - Langage reconnu par un automate local
Prouver que le langage reconnu par un automate local est un langage local.

Théorème 11 - Théorème de Glushkov

Si Q est l'automate de Glushkov associé à L alors Q reconnaît L.

Exercice 20 - Démonstration
Prouver le théorème 11

On peut synthétiser ce résultat, ici r est une expression régulière linéaire

$$\begin{tikzcd} L_r \arrow[r, leftrightarrow, "="] & L(Q) \\ r \arrow[u] \arrow[r] & Q \arrow[u] \end{tikzcd}$$

III.2 Morphismes

On considère une fonction f d'un alphabet $\mathcal{B}$ vers un alphabet $\mathcal{A}$.

Prolongement aux mots

Les mots sont des produits de lettres : on peut donc définir une fonction f^* de $\mathcal{B}^*$ vers $\mathcal{A}^*$ par $f^*(\varepsilon) = \varepsilon$ (le premier ε est le mot vide de $\mathcal{B}^*$, le second celui de $\mathcal{A}^*$)
et $f^*(u.x) = f^*(u).f(x)$ pour $u \in \mathcal{B}^*$ et $x \in \mathcal{B}$.

Exercice 21 - Morphisme de monoïdes
Prouver que si f est une fonction de $\mathcal{B}$ vers $\mathcal{A}$ alors
$f^*(u.v) = f^*(u).f^*(v)$ pour tous mots u et v de $\mathcal{B}^*$.

On prolonge f^* sur les langages sans changer le nom :
si L est un langage sur $\mathcal{B}$, $f^*(L) = \left\{f^*(u)\ ;\ u \in L\right\}$ est un langage sur $\mathcal{A}$.

Exercice 22 - Opération rationnelles
Prouver que si L_1 et L_2 sont des langages sur $\mathcal{B}$ alors

1. $f^*(L_1 \cup L_2) = f^*(L_1) \cup f^*(L_2)$,
2. $f^*(L_1.L_2) = f^*(L_1).f^*(L_2)$
3. $f^*(L_1^*) = \left(f^*(L_1)\right)^*$.
4. A-t-on toujours $f^*(L_1 \cap L_2) = f^*(L_1) \cap f^*(L_2)$?

Prolongement aux expression régulières

De même on peut prolonger f de manière inductive sur l'ensemble des expressions régulières sur $\mathcal{B}$: `f(∅ ) = ∅` , `f(ϵ) = ϵ`, `f(a) = b` avec $b = f(a)$,
`f(r1 + r2) = (f(r1) + f(r’)`, `f(r1.r2) = (f(r1).f(r2))` et `f(r*) = (f(r))*`.

Théorème 12 - Transport des expressions régulières

Si f est une fonction de $\mathcal{B}$ vers $\mathcal{A}$ alors $L\big[\texttt{f(r)}\big] = f^*\big(L[\texttt{r}]\big)$
pour toute expression régulière `r` sur $\mathcal{B}$.

On a donc un transport depuis les expressions régulières vers les langages.

$$\begin{tikzcd} L[\texttt{r}] \arrow[r, "f^*"] & L[\texttt{r1}] \\ \texttt{r} \arrow[u] \arrow[r, "\texttt{f}"'] & \texttt{r1} \arrow[u] \end{tikzcd}$$

Dans ce diagramme les deux chemins de $\texttt{r}$ à $L[\texttt{r1}]$ donnent le même résultat.

Exercice 23 - Démonstration
Prouver le théorème 12

Prolongement aux automates

f peut servir aussi à transformer les automates (non déterministes) sur l'alphabet $\mathcal{B}$.
Si $Q = (\mathcal{B}, S, \Delta, I, T)$ on note $\widehat{f}(Q) = (\mathcal{A}, S, \Delta', I, T)$ où le graphe de Δ' est défini par $G_{\Delta'} = \left\{(s, f(x), s')\ ;\ (s, x, s') \in G_\Delta\right\}$.
C'est l'automate obtenu en changeant les lettres par leur image par f.

Théorème 13 - Transport des automates

Si f est une fonction de $\mathcal{B}$ vers $\mathcal{A}$ alors $L\big(\widehat{f}(Q)\big) = f^*\big(L(Q)\big)$.
pour toute expression régulière r sur $\mathcal{B}$.

On a donc un transport depuis les automates vers les langages.

$$\begin{tikzcd} L(Q) \arrow[r, "f^*"] & L(Q') \\ Q \arrow[u] \arrow[r, "\widehat{f}"'] & Q' \arrow[u] \end{tikzcd}$$

Dans ce diagramme les deux chemins de Q à $L(Q')$ donnent le même résultat.

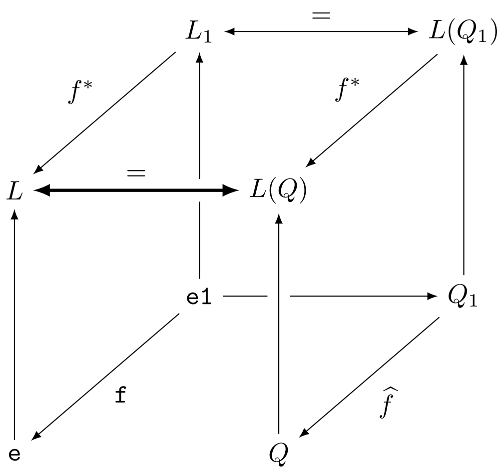
Exercice 24 - Démonstration
Prouver le théorème 13

III.3 Construction de l'automate

- On part d'un langage rationnel non vide L.
  dénoté par une expression régulière `e`, $L = L[\mathtt{e}]$.
- On transforme `e` en une expression linéaire `e1` en changeant uniquement les lettres par des lettres **distinctes**.
  Si on note $\mathcal{A}_1$ l'alphabet utilisé dans `e1`, on note f la fonction de $\mathcal{A}_1$ vers $\mathcal{A}$ qui donne, pour chaque lettre de $\mathcal{A}_1$, la lettre de $\mathcal{A}$ qu'elle remplace.
  Avec la notation ci-dessus on a `e = f(e1)`.
- On note $L_1 = L[\mathtt{e1}]$, L_1 est local car `e1` est linéaire.
- On associe à `e1` son automate de Glushkov, Q_1.
- On transforme Q_1 en Q en remplaçant les étiquettes à l'aide de f : $Q = \widehat{f}(Q_1)$.
- On peut, si besoin, déterminiser Q.

Les résultats prouvés ci-dessus donnent alors :

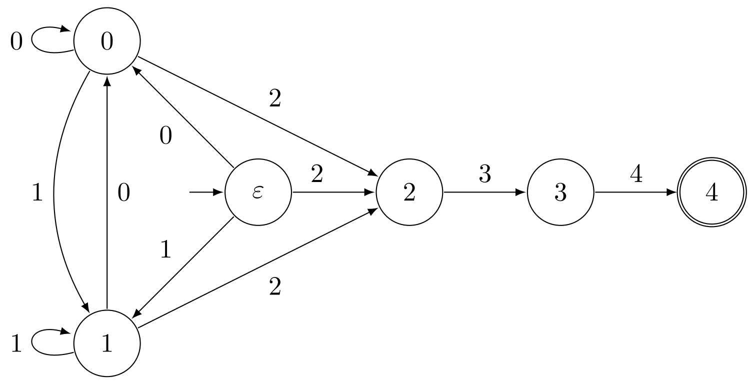
- $L = L[\mathtt{e}] = L[\mathtt{f(e1)}] = f^*\big(L[\mathtt{e1}]\big) = f^*\big(L_1\big)$ (transport des expressions régulières)
- $L_1 = L(Q_1)$ (théorème de Glushkov)
- $L(Q) = L\big(\widehat{f}(Q_1)\big) = f^*\big(L(Q_1)\big)$ (transport des automates)
- Ainsi $L = f^*\big(L_1\big) = f^*\big(L(Q_1)\big) = L(Q)$.



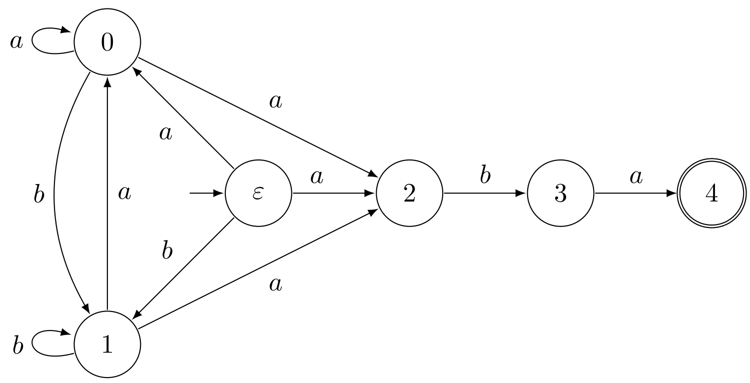
La construction de Q à partir de `r` est l'algorithme de Berry-Sethy.

III.4 Un exemple

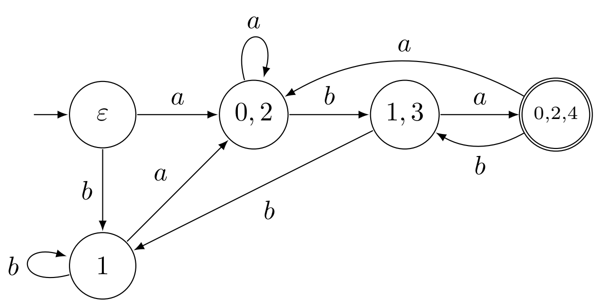
- Soit L dénoté par `e = (a+b)*.a.b.a`.
- On pose `e1 = (0+1)*.2.3.4` : on définit donc f par le tableau `[|a; b; a; b; a|]`.
- $L_1 = L[$`e1`$]$ ne content pas ε.
  On calcule $P = \{0, 1, 2\}$, $S = \{4\}$, $F = \{00, 01, 10, 11, 02, 12, 23, 34\}$.
- L'automate Q_1 est calculé.



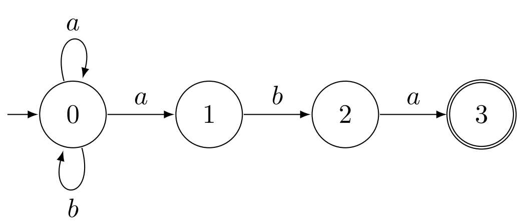
- On traduit en l'automate Q, non déterministe.

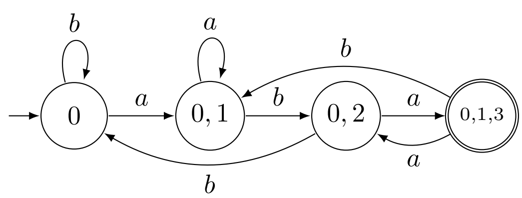


- On peut le déterminiser.



On pouvait aussi trouver directement un automate non déterministe puis le déterminiser.





Solutions

Solution de l'exercice 1 - Démonstration
La propriété se démontre par récurrence sur $|v|$.

- Elle est triviale pour $|v| = 0$ car alors $v = \varepsilon$ et $s.(u.\varepsilon) = s.u = (s.u).\varepsilon$.
- On suppose qu'elle est vraie pour les mots de longueur n.
  Si v est de longueur $n+1$ on écrit $v = v'.x$ avec v' de longueur n.
  On a $s.(u.v') = (s.u).v'$ d'après l'hypothèse de récurrence. Alors, en notant $s' = s.u$,
  $s.(u.v) = s.(u.v'.x) = \big(s.(u.v')\big).x = \big((s.u).v'\big).x = \big(s'.v'\big).x = s'.(v'.x) = s'.v = (s.u).v$.
- la propriété est donc vraie pour tout n.

Solution de l'exercice 2 - Premières propriétés

- Si $T = \emptyset$ alors $s_0.u$ ne peut jamais appartenir à T donc $L(Q) = \emptyset$.
- Si $T = S$ alors $s_0.u \in S = T$ pour tout u donc $L(Q) = \mathcal{A}^*$.
- Si $L(Q)$ contient ε alors $s_0 = s_0.\varepsilon \in T$: s_0 appartient à T.
  Inversement si $s_0 \in T$ alors $s_0.\varepsilon = s_0 \in T$ donc $L(Q)$ contient ε.

Solution de l'exercice 3 - Démonstration
Tout élément s de S' est de la forme $s = s_0.v$ donc $\delta(s, a) = s.a = (s_0.u).a = s_0.(u.a) \in S'$.
Ainsi la restriction de δ à $S' \times \mathcal{A}$ est à valeur dans S', on la note encore δ.
On obtient ainsi un nouvel automate

Solution de l'exercice 4 - Démonstration

- Si u appartient à $L(Q')$ alors $s_0.u \in T \cap S' \subset T$ donc $u \in L(Q)$: $L(Q') \subset L(Q)$.
- Si u appartient à $L(Q)$ alors $s_0.u \in T$.
  Les sommets atteints depuis s_0 appartiennent à S' donc le chemin est un chemin dans S'.
  En particulier $s_0.u \in S'$ donc $s_0.u \in T \cap S'$ et $u \in L(Q')$.
  Ainsi $L(Q') \subset L(Q)$ donc $L(Q') = L(Q)$.

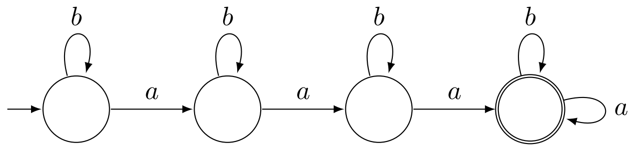
Solution de l'exercice 5 - Démonstration
Soit L reconnaissable. On considère un automate $Q = (\mathcal{A}, S, \delta, s_0, T)$ qui le reconnaît. On a a alors $u \in \overline{L}$ équivalent à $s_0.u \notin T$, c'est-à-dire à $s_0.u \in \overline{T} = S \setminus T$.
Ainsi $\overline{L}$ est reconnu par l'automate $(\mathcal{A}, S, s_0, \delta, \overline{T})$.

Solution de l'exercice 6 - Démonstration
L'état initial est $(s_{1,0}, s_{2,0})$. On remarque qu'un mot u est reconnu par Q_1 (resp. Q_2) si et seulement si
$(s_{1,0}, s_{2,0}).u \in T_1 \times S_2$ (resp. $(s_{1,0}, s_{2,0}).u \in S_1 \times T_2$).
Si L_1 est reconnu par $(\mathcal{A}, S_1, \delta_1, s_{1,0}, T_1)$ et L_2 par $(\mathcal{A}, S_2, \delta_2, s_{2,0}, T_2)$ alors
$L_1 \cup L_2$ est reconnu par $(\mathcal{A}, S_1 \times S_2, \delta_1 \times \delta_2, (s_{1,0}, s_{2,0}), T_1 \times S_2 \cup S_1 \times T_2)$ et
$L_1 \cap L_2$ est reconnu par $(\mathcal{A}, S_1 \times S_2, \delta_1 \times \delta_2, (s_{1,0}, s_{2,0}), T_1 \times T_2)$.

Solution de l'exercice 7 - Démonstration
Tout mot reconnu par Q' est reconnu par Q.
De plus tout mot reconnu par Q est non bloquant et ne passe que par des états co-accessibles donc appartenant à S', il est donc reconnu par Q'.

Solution de l'exercice 8 - Démonstration
Démonstration :les mots bloquants deviennent alors des mots qui arrivent à l'état-puits et qui y restent donc ne sont pas reconnus alors que les mots non-bloquants gardent leur parcours et sont reconnus ou non en même temps dans les deux automates.

Solution de l'exercice 9 - 3 occurences
L est dénoté par `(b*.a).(b*.a).(b*.a).(a+b)*` est reconnu par

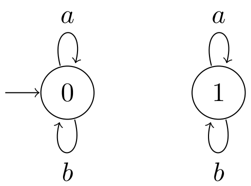
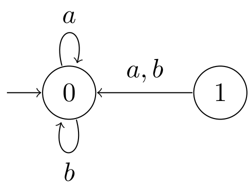
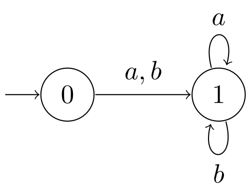
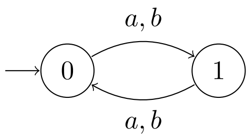


Solution de l'exercice 10 - Divisibilité par trois
Si on veut calculer le reste de la division par 3 d'un nombre écrit en binaire, à chaque nouveau bit lu le nombre est multiplié par 2 et on ajoute le bit :

Reste	0	1	2
0	0	2	1
1	1	0	2

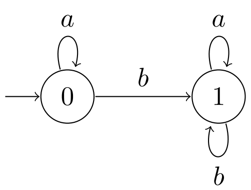
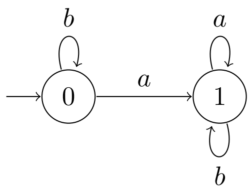
$0.n = i$ est donc équivalent à `n mod 3 = i`.

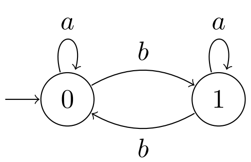
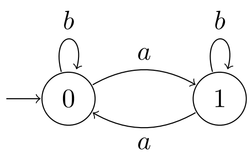
Solution de l'exercice 11 - Petits automates
On note $S = \{0, 1\}$, il y a deux choix pour chaque transition $0.a$, $0.b$, $1.a$ et $1.b$ donc il y a 16 machines possibles.
On choisira 0 pour état initial ; il reste à choisir T. $T = \emptyset$ ou $T = \{0, 1\}$ donnent des langages triviaux ($\emptyset$ ou $\mathcal{A}^*$), on aura deux langages complémentaires en choisissant $T = \{0\}$ ou $T = \{1\}$.
On commence par les 4 machines symétriques en a et b.

-  ou  $L = \mathcal{A}^*$ si $T = \{0\}$ et $L = \emptyset$ si $T = \{1\}$.
-  $L = \{\varepsilon\}$ si $T = \{0\}$ et $L = \mathcal{A}^+$ si $T = \{1\}$.
-  L est l'ensemble des mots de longueur paire si $T = \{0\}$.

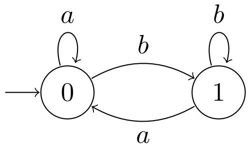
  L est l'ensemble des mots de longueur impaire si $T = \{1\}$.

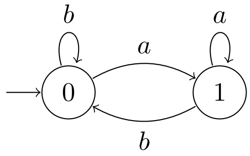
Dans les autres cas on peut regrouper les automates par paires en échangeant le rôle de a et de b.

-  ou 

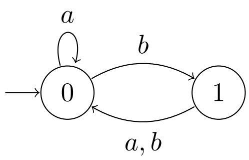
  $L = \{a\}^*$ (ou $L = \{b\}^*$) si $T = \{0\}$.
  L est l'ensemble des mots contenant b (ou contenant a) si $T = \{1\}$.
-  ou 

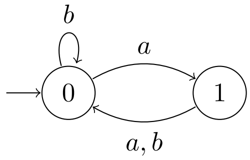
L est l'ensemble des mots de longueur paire en b (resp. en a) si $T = \{0\}$
L est l'ensemble des mots de longueur impaire en b (resp. en a) si $T = \{1\}$

- 

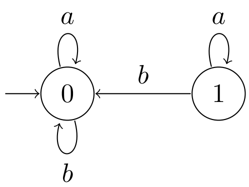
ou


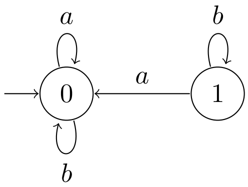
L est l'ensemble des mots vides ou finissant par a (resp. par b) si $T = \{0\}$
L est l'ensemble des mots non vides finissant par b (resp. par a) si $T = \{1\}$

- 

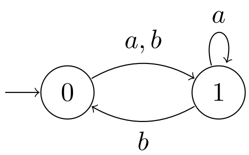
ou


L est l'ensemble des mots finissant par un nombre pair de b (resp. de a) si $T = \{0\}$
L est l'ensemble des mots finissant par un nombre impair de b (resp. de a) si $T = \{1\}$

- 

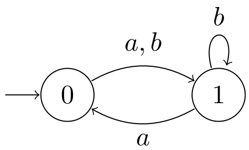
ou


$L = \mathcal{A}^*$ si $T = \{0\}$ et $L = \emptyset$ si $T = \{1\}$.

- 

$L = \{b^{2n}\ ;\ n \in \mathbb{N}\} \cup \{u.ab^{2n+1}\ ;\ u \in \mathcal{A}^*, n \in \mathbb{N}\}$ si $T = \{0\}$.

$L = \{b^{2n+1}\ ;\ n \in \mathbb{N}\} \cup \{u.ab^{2n}\ ;\ u \in \mathcal{A}^*, n \in \mathbb{N}\}$ si $T = \{1\}$.



$L = \{a^{2n}\ ;\ n \in \mathbb{N}\} \cup \{u.ba^{2n+1}\ ;\ u \in \mathcal{A}^*, n \in \mathbb{N}\}$ si $T = \{0\}$.

$L = \{a^{2n+1}\ ;\ n \in \mathbb{N}\} \cup \{u.ba^{2n}\ ;\ u \in \mathcal{A}^*, n \in \mathbb{N}\}$ si $T = \{1\}$.

Solution de l'exercice 12 - Alphabet à une lettre
On part de l'état initial, noté 1, et on numérote les états tant que l'image par δ est un nouvel état. Le dernier état accessible est envoyé dans un état déjà visité.

$$1 \xrightarrow{a} 2 \xrightarrow{a} \cdots \xrightarrow{a} n \xrightarrow{a} p \text{ avec } 1 \leqslant p \leqslant n$$

On obtient une forme de poêle.
Si les états finaux sont $f_1 < f_2 < \cdots < f_r$ avec $f_s < p \leqslant f_{s+1}$
alors le langage reconnu est $\{a^n\ ;\ n \in A\}$ avec, en notant $T = n - p + 1$ et
$A = \{f_1, f_2, \ldots, f_s\} \cup \{f_{s+1} + Tn, f_{s+2} + Tn, \ldots, f_r + Tn\ ;\ n \in \mathbb{N}\}$.

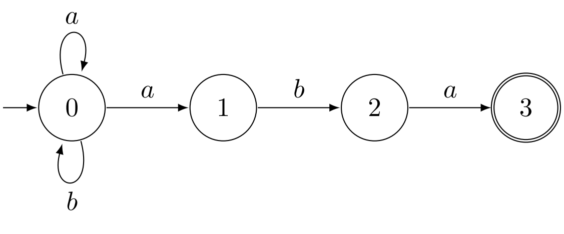
Solution de l'exercice 13 - États co-accessibles

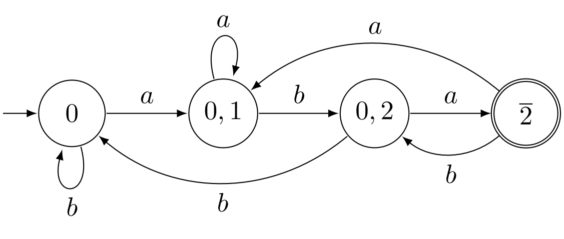
- Si tous les états sont co-accessibles alors, pour tout u, $s_0.u$ est co-accessible donc il existe v tel que $(s_0.u).v \in T$ donc $u.v$ est un mot reconnu. On a prouvé que tout mot est un préfixe d'un mot reconnu.
- On suppose que tout mot est un préfixe d'un mot reconnu.
  Pour tout état accessible s, on peut écrire $s = s_0.u$. u est un préfixe d'un mot reconnu u', $u' = u.v$. On a $s_0.u' \in T$ donc $s.v = (s_0.u).v \in T$.
  Tout état accessible est donc co-accessible.

Solution de l'exercice 14 - Démonstration
I est une partie de S donc $I \in \mathcal{P}(S)$.
De même $\delta(E, x) \in \mathcal{P}(S)$ pour tout $E \in \mathcal{P}(S)$ et pour tout $x \in \mathcal{A}$.
Q' est bien un automate déterministe complet.

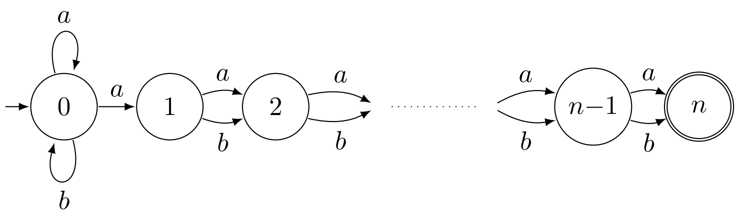
1. Si $u = x_1x_2\cdots x_n$ appartient à $L(Q)$ alors il existe un calcul réussi
   $s_0 \xrightarrow{x_1} s_1 \xrightarrow{x_2} s_2 \xrightarrow{x_3} \quad \cdots \quad \xrightarrow{x_{n-1}} s_{n-1} \xrightarrow{x_n} s_n$ avec $s_0 \in I$ et $s_n \in T$.
   On a alors $s_1 \in \Delta(s_0, x_1)$ avec $s_0 \in I$ donc $s_1 \in \delta(I, x_1) = E_1$.
   De même $s_2 \in \Delta(s_0, x_1)$ avec $s_1 \in E_1$ d'où $s_2 \in \delta(E_1, x_2) = E_2$.
   On construit ainsi le parcours de u dans Q' : $I \xrightarrow{x_1} E_1 \xrightarrow{x_2} E_2 \xrightarrow{x_3} \quad \cdots \quad \xrightarrow{x_{n-1}} E_{n-1} \xrightarrow{x_n} E_n$
   Or $s_n \in E_n$ et $s_n \in T$ donc $S_n \cap T \neq \emptyset$ d'où $S_n \in \mathcal{T}$.
   Ainsi $I.u \in \mathcal{T}$ dans Q' donc $u \in L(Q')$: on a prouvé $L(Q) \subset L(Q')$.
2. Si $\varepsilon \in L(Q)$ alors il existe un état s de S qui est initial et final à la fois. Ainsi $I \cap T \neq \emptyset$.
   Dans Q' on a donc $I \in \mathcal{T}$ donc $\varepsilon \in L(Q')$.
3. Inversement si $u \neq \varepsilon \in L(Q')$ alors son parcours est de la forme
   $I = E_0 \xrightarrow{x_1} E_1 \xrightarrow{x_2} \cdots \xrightarrow{x_{n-1}} E_{n-1} \xrightarrow{x_n} E_n$ avec $E_n \in \mathcal{T}$ donc il existe $s_n \in E_n \cap T$.
   La transition $E_{n-1} \xrightarrow{x_n} E_n$ implique qu'il existe un élément s_{n-1} dans E_{n-1} et une transition $s_{n-1} \xrightarrow{x_n} s_n$
   On définit ainsi, en descendant, un élément $s_i \in E_i$ pour tout i avec un calcul
   $s_0 \xrightarrow{x_1} s_1 \xrightarrow{x_2} \cdots \xrightarrow{x_{n-1}} s_{n-1} \xrightarrow{x_n} s_n$
   $s_0 \in E_0 = I$, $s_n \in T$ donc u est reconnu par Q.
   On a alors $L(M') \subset L(M)$ d'où l'égalité.
4. Si $\varepsilon \in L(Q')$, alors l'état initial, I, est final donc il existe un état s dans $I \cap T$. s est initial et final dans M donc $\varepsilon \in L(Q)$.

Solution de l'exercice 15 - Construction





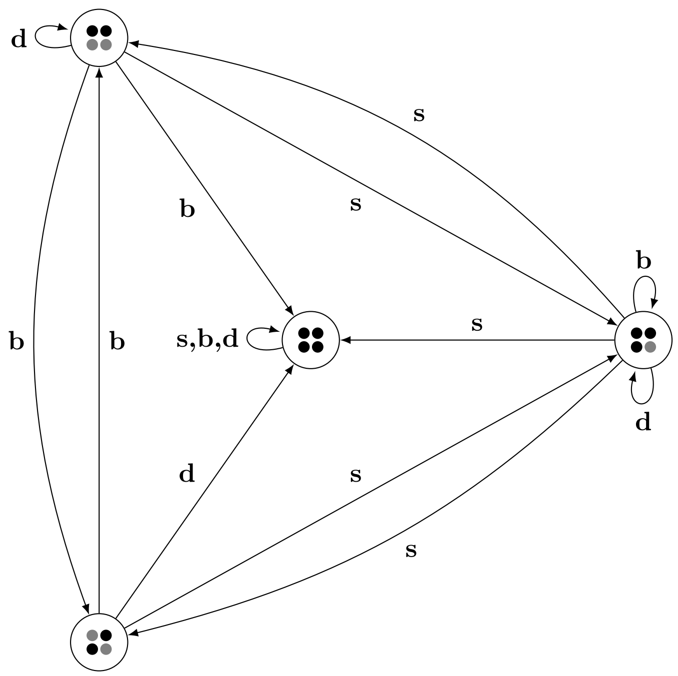
Solution de l'exercice 16 - Le pire peut arriver



Soit $Q = (\mathcal{A}, S, \delta, s_0, T)$ un automate déterministe qui reconnaît $\mathcal{A}$.
On définit φ de $\mathcal{A}^n$ vers S par $\varphi(u) = s_0.u$.
On montre que φ est injective. On suppose qu'on a $\varphi(u) = \varphi(u')$.
Si on a $u \neq u'$ on considère la première position k telle que $u_k \neq u'_k$: on a $1 \leqslant k \leqslant n$, $u = v.a.w$, $u' = v.b.w'$ (ou l'inverse) avec $|v| = k - 1$ et $|w| = |w'| = n - k$.
On a $u.a^{k-1} = v.a.(w.a^{k-1})$ qui appartient à L car $|w.a^{k-1}| = n - 1$ mais $u'.a^{k-1}$ n'appartient pas à L.
Ainsi $\varphi(u).a^{k-1} = s_0.(u.a^{k-1}) \in T$ mais $\varphi(u').a^{k-1} = s_0.(u'.a^{k-1}) \notin T$ ce qui rend impossible l'égalité $\varphi(u) = \varphi(u')$.
On doit donc avoir $u = u'$: φ est injective.
On a une injection de $\mathcal{A}^n$ dans S donc $|S| \geqslant |\mathcal{A}^2| = 2^n$.

Solution de l'exercice 17 - Langage transposé
Les calculs dans Q^T sont les inverses des calculs dans Q.

Solution de l'exercice 18 - Le barman aveugle
Voici un automate décrivant les mouvements :



- **s** (pour singleton) consiste à retourner 1 jeton (ou 3 jetons)
- **b** (pour bord) consiste à retourner 2 jeton adjacents
- **d** (pour diagonale) consiste à retourner 2 jeton en diagonale.
- Depuis l'état de 4 couleurs identiques les transformations renvoie au même état car on est dans un état gagnant.

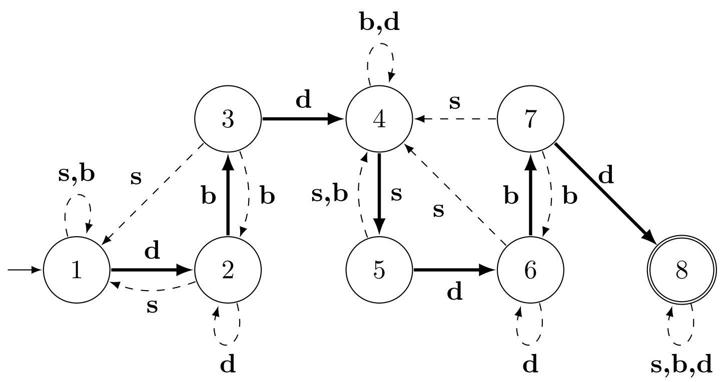
On note

- T l'état dans lequel tous les jetons sont de la même couleur,
- U l'état dans lequel un jeton a une couleur et les 3 autres ont l'autre couleur,
- B l'état dans lequel 2 jetons côte-à-côte ont une couleur et les 2 autres ont l'autre couleur,
- D l'état dans lequel 2 jetons en diagonale ont une couleur et les 2 autres ont l'autre couleur.

La déterminisation, avec T, U, P et D états initiaux, donne

	U,B,D,T	U,B,T	U,D,T	U,T	B,D,T	B,T D,T	T
s	U,B,D,T	U,B,D,T	U,B,D,T	B,D,T	U,T	U,T U,T	T
b	U,B,D,T	U,D,T	U,B,T	U,T	U,T	D,T B,T	T
d	U,B,T	U,B,T	U,T	U,T	B,T	B,T T	T
état	1	2	3	4	5	6 7	8

L'état final qui nous intéresse est $\{T\}$: on veut être certain d'arriver.



La suite de mouvement **dbdsdbd** permet d'arriver au résultat.

Solution de l'exercice 19 - Langage reconnu par un automate local

$Q = (\mathcal{A}, S, \delta, s_0, T)$ est un automate local. On définit

- $P = \{x \in \mathcal{A}\ ;\ \delta(s_0, x) \text{ est défini}\}$,
- $S = \{x \in \mathcal{A}\ ;\ \exists s \in S,\ \delta(s, x) \in T\}$,
- $T = \{xy \in \mathcal{A}^2\ ;\ \exists s, t \in S,\ \delta(s, x) = t \text{ et } \delta(t, y) \text{ est défini}\}$.

On note L le langage local défini par (P, S, F) qui contient ou non ε selon que s_0 est final ou non.

1. Par construction ε appartient à L et $L(Q)$ ou a aucun des deux.
2. Si un mot non vide est reconnu par Q alors sa première lettre est dans P, sa dernière lettre est dans S et ses facteurs (s'il y en a) sont dans F, par construction.
   Ainsi $L(Q) \setminus \{\varepsilon\}$ est inclus dans le, noté L.
3. Inversement supposons que u appartienne au langage local défini par (P, S, F) avec $u \neq \varepsilon$.
   On écrit $u = x_1 x_2 \cdots x_n$.
   $x_1 \in P$ donc $\delta(s_0, x_1)$ est défini : on note $s_1 = \delta(s_0, x_1)$.
   $x_1 x_2 \in F$ donc il existe une transition $s \xrightarrow{x_1} t \xrightarrow{x_2} r$. On a donc $t = \delta(s, x_1)$ et $r = \delta(t, x_2)$.
   Comme $\delta(s, x_1)$ et $\delta(s_0, x_1)$ sont définis dans l'automate local, ils sont égaux donc $t = s_1$ et on peut définir $s_2 = r = \delta(s_1, x_2)$.
   On peut ainsi définir un chemin dans Q d'étiquette u qui aboutit dans T car on a $x_n \in S$.
   Ainsi u est reconnu par Q : $L \setminus \{\varepsilon\} \subset L(Q)$.

On a donc $L = L(Q)$.

Solution de l'exercice 20 - Démonstration
Les mots non vides de L, $u = x_1 x_2 \cdots x_n$, sont associés aux mots reconnus par Q selon le chemin $\varepsilon \xrightarrow{x_1} x_1 \xrightarrow{x_2} x_2 \xrightarrow{x_3} \quad \cdots \quad \xrightarrow{x_{n-1}} x_{n-1} \xrightarrow{x_n} x_n$.
Inversement un tel chemin a une étiquette u dans L car la transition $\varepsilon \xrightarrow{x_1} x_1$ implique $x_1 \in P$, les $x_{i-1} \xrightarrow{x_i} x_i \xrightarrow{x_{i+1}} x_{i+1}$ impliquent $x_i x_{i+1} \in F$ et x_n terminal implique $x_n \in T$.
Le mot vide est reconnu si et seulement ε est un état final.

Solution de l'exercice 21 - Morphisme de monoïdes
On montre par récurrence sur n que, pour tout mot $u \in \mathcal{B}^*$ et pour tout mot $v \in \mathcal{B}^*$ de longueur n, $f^*(u.v) = f^*(u).f^*(v)$.

Solution de l'exercice 22 - Opération rationnelles

1. On a $L_1 \subset L_1 \cup L_2$ donc $f^*(L_1) \subset f^*(L_1 \cup L_2)$.
   De même $f^*(L_2) \subset f^*(L_1 \cup L_2)$ donc $f^*(L_1) \cup f^*(L_2) \subset f^*(L_1 \cup L_2)$.
   Inversement si $v \in f^*(L_1 \cup L_2)$ alors il existe $u \in L_1 \cup L_2$ tel que $v = f^*(u)$. Si $u \in L_i$ alors $v \in f^*(L_i) \subset f^*(L_1) \cup f^*(L_2)$ d'où l'égalité.
2. $f^*(L_1.L_2) = \big\{f^*(u.v)\ ;\ u \in L_1, v \in L_2\big\} = \big\{f^*(u).f^*(v)\ ;\ u \in L_1, v \in L_2\big\}$ puis
   $f^*(L_1.L_2) = \big\{f^*(u)\ ;\ u \in L_1\big\}.\big\{f^*(v)\ ; v \in L_2\big\} = f^*(L_1).f^*(L_2)$.
3. On en déduit par récurrence sur n que $f^*(L_1^n) = \big(f^*(L_1)\big)^n$ puis
   $$f\left(L_1^*\right) = f\left(\bigcup_{n\in\mathbb{N}} L_1^n\right) = \bigcup_{n\in\mathbb{N}} f\left(L_1^n\right) = \bigcup_{n\in\mathbb{N}} \big(f^*(L_1)\big)^n = \big(f^*(L_1)\big)^*.$$
4. On a $L_1 \cap L_2 \subset L_1$ donne $f^*(L_1 \cap L_2) \subset f^*(L_1)$ et, de même, $f^*(L_1 \cap L_2) \subset f^*(L_2)$ donc $f^*(L_1 \cap L_2) \subset f^*(L_1) \cap f^*(L_2)$.
   Cependant l'inclusion peut être stricte. Pour $\mathcal{B} = \mathcal{A} = \{a, b\}$, $f(a) = f(b) = a$, $L_1 = \{a\}^*$ et $L_2 = \{b\}^*$ on a $L_1 \cap L_2 = \{\varepsilon\}$ mais $f^*(L_1) = f^*(L_2) = \{a\}^*$ donc
   $f^*(L_1 \cap L_2) = \{\varepsilon\} \neq \{a\}^* = f^*(L_1) \cap f^*(L_2)$.

Solution de l'exercice 23 - Démonstration
La démonstration se fait par induction structurelle.

1. Les cas de base sont immédiats.
2. Pour $r = x$, $L\big[\widehat{\varphi}(x)\big] = L\big[\varphi(x)\big] = \{\varphi(x)\} = \varphi^*\big(\{x\}\big) = \varphi^*\big(L[x]\big)$.
3. On suppose que $L\big[\widehat{\varphi}(e)\big] = \varphi^*\big(L[e]\big)$ et $L\big[\widehat{\varphi}(f)\big] = \varphi^*\big(L[f]\big)$.
   $L\big[\widehat{\varphi}(e+f)\big] = L\big[\widehat{\varphi}(e)+\widehat{\varphi}(f)\big] = L\big[\widehat{\varphi}(e)\big] \cup L\big[\widehat{\varphi}(f)\big] = \varphi^*\big(L[e] \cup \varphi^*\big(L[f]\big) = \varphi^*\big(L[e] \cup L_[f]\big) = \varphi^*\big(L[(e+f)]\big)$
   De même pour le produit et l'étoile.

Solution de l'exercice 24 - Démonstration
Tout mot de $L(Q)$ est l'étiquette d'un calcul réussi dans Q ; on note $u = x_1 x_2 \cdots x_n$,
$s_0 \xrightarrow{x_1} s_1 \xrightarrow{x_2} s_2 \xrightarrow{x_3} \quad \cdots \quad \xrightarrow{x_{n-1}} s_{n-1} \xrightarrow{x_n} x_n$.
Dans $\widehat{f}(Q)$ le chemin devient
$s_0 \xrightarrow{f(x_1)} s_1 \xrightarrow{f(x_2)} s_2 \xrightarrow{f(x_3)} \quad \cdots \quad \xrightarrow{f(x_{n-1})} s_{n-1} \xrightarrow{f(x_n)} x_n$
il reste réussi car les ensembles initiaux et terminaux sont inchangés ;
ainsi $f(x_1)f(x_2)\cdots f(x_n) = f^*(u)$ appartient à $L\big(\widehat{f}(Q)\big)$ d'où $f^*\big(L(Q)\big) \subset L\big(\widehat{f}(Q)\big)$.

De même tout calcul réussi dans $\widehat{f}(Q)$ peut se "remonter" en un calcul réussi dans Q car les étiquettes sont des $f(x_i)$. Le mot associé est donc l'image par f^* d'un mot de $L(Q)$ d'où $L\big(\widehat{f}(Q)\big) \subset f^*\big(L(Q)\big)$
On a ainsi $L\big(\widehat{f}(Q)\big) = f^*\big(L(Q)\big)$.