ROYAUME DU MAROC
UNIVERSITE ABDELMALEK ESSAADI
Ecole Nationale des Sciences Appliquées

Tanger

Année universitaire 2019-2020

Cours : Théorie des Automates / Chapitre II

Automates à nombre d'états Fini

Version 2.0

Introduction

En informatique théorique, l'objectif de la théorie des automates est de proposer des modèles de mécanismes mathématiques qui formalisent les méthodes de calcul.

Cette théorie est le fondement de plusieurs branches importantes de l'informatique théorique, comme :

1. La calculabilité par le modèle des machines de Turing,
2. Les automates finis, et leurs variantes, qui sont utilisées dans :
    a. L'analyse des langues naturelles,
    b. La traduction des programmes,
    c. Divers algorithmes de manipulation de texte
        (Comme Les algorithmes de recherche de sous-chaîne),
    d. Ou la vérification automatique du fonctionnement de circuits logiques
        (Cette partie concerne les ingénieurs GSEA)
3. La théorie de la complexité des algorithmes
4. Le modèle «Checking» qui sert à établir la conformité des programmes à leurs spécifications.

La théorie des automates est l'étude des machines abstraites qui permettent de formaliser les méthodes de calcul.

Donc un automate n'est pas un robot comme on pourrait l'imaginer, c'est juste des objets mathématiques, des machines abstraites qui modélisent des systèmes

L'étude des automates a commencé depuis les années 50 (1950). Avant même l'apparition de l'informatique et les premiers ordinateurs.

L'objet traité par un automate est un mot d'un langage. Pour arriver à la généralité souhaitée on convertit :

- un « problème » en un **langage**
- Et la résolution du problème, en l'analyse d'un élément de ce langage.

On représente chaque instance d'un « problème » par **un mot**.

Ils sont utilisés en :

- Informatique,
- Linguistique,
- Biologie (Génomique),
- Mathématiques,
- Mécanique : machine à café

I. Les Automates à nombre d'états fini déterministes :

a. Définition :

Un automate fini déterministe (ou AFD) est la donnée d'un quintuple :

$$A = (Q, \Sigma, \delta, Q_0, Q_m) \quad \text{où}$$

- Q : ensemble d'états de l'automate;
- Σ : ensemble d'événements, ou symboles, alphabet de l'automate;
- δ : fonction de transition de A;
- Q_0 : ensemble des états initiaux;
- Q_m : ensemble des états marqués ou états finaux

Soit Σ un alphabet fini. Un automate sur Σ est **fini** si et seulement si son ensemble d'états Q est **fini**.

Il y a, à partir d'un état, un seule flèche pour chaque transition→ Un automate fini **déterministe**

Nous supposerons que, δ est une **fonction totale**, i.e., que δ est définie pour tout couple $(q, \sigma) \in Q \times \Sigma$ (on parle alors d'AFD complet).

Un automate à états finis est un formalisme caractérisé par les états et les événements :

- Un **état** représente une activité ou un état de fonctionnement du système,
- un **événement** est un signal Permettant de passer d'un état à un autre.

b. Représentation graphique :

Nous représentons un AFD A de la manière suivante :

- Les états de A sont les sommets d'un graphe orienté et sont représentés par des **cercles**.
- Les transitions par des arcs orientés (**flèches**), avec leurs étiquettes inscrites à proximité.
- Les états initiaux sont repérés par des courtes flèches entrantes sans label.
- Les états finaux par des flèches sortantes sans label ou un double cercle.

c. Exemple :

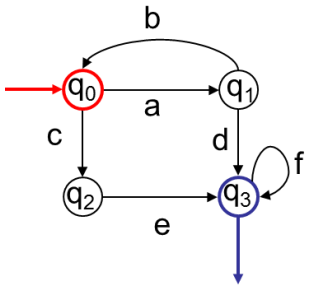


Figure II.1 : Exemple d'Automate à nombre d'états fini déterministe

- $Q = \{q_0, q_1, q_2, q_3\}$;
- $\Sigma = \{a, b, c, d, e, f\}$;
- $Q_0 = \{q_0\}$;
- $Q_m = \{q_3\}$;

$\delta : Q \times \Sigma \rightarrow \mathbf{Q}$

$(q_i, \sigma) \rightarrow q_j$

Dans notre exemple :

$\delta\ (q_0, a) = q_1$

$\delta\ (q_1, d) = q_3$

Une transition représentée par un seul arc et plusieurs labels :

Si $(q_i, \sigma) \rightarrow q_j$ Et $(q_i, \sigma') \rightarrow q_j$ Alors : $q_i \xrightarrow{\sigma,\sigma'} q_j$

d. Exemple

Soit l'automate : $A = (Q, q_0, F, \Sigma, \delta)$

- $Q = \{1, 2, 3\}$;
- $\Sigma = \{a, b\}$;
- $q_0 = 1$;
- $F = \{1,2\}$;

Et où la fonction de transition δ est donnée par :

δ	a	b
1	1	2
2	1	3
3	3	2

L'automate est représenté à la figure 2.2.

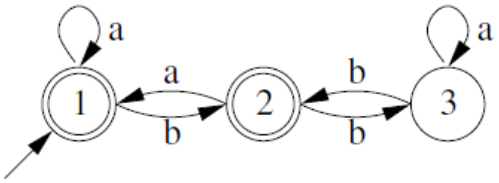


Figure II.2 : un A.F.D.

e. Définition :

Soit l'automate : $A = (Q, q_0, F, \Sigma, \delta)$ un AFD. On étend naturellement la fonction de transition : δ à $Q \times \Sigma^*$ de la manière suivante : $\delta(q,\varepsilon) = q$ Et

$$\delta(q, \sigma\omega) = \delta(\delta(q,\sigma), \omega), \quad \sigma \in \Sigma\ \omega \in \Sigma^*$$

Le langage accepté par A est alors : $L(A) = \{\omega \in \Sigma^* \mid \delta(q_0, \omega) \in F\}$.

Si $\omega \in L(A)$, on dit encore que : A accepte le mot ω (ou ω est accepté par A).

Ainsi le rôle fondamental d'un automate est d'accepter ou de rejeter des mots. Un automate partitionne l'ensemble des mots sur Σ en deux sous-ensembles :

$$L(A) \text{ et } \Sigma^* \setminus L(A)$$

f. Exemple :

Prenant l'exemple précédent :

L'automate A de la figure 1 accepte le mot : abbab car on a, partant de l'état initial, le parcours suivant au sein de A :

$$1 \xrightarrow{a} 1 \xrightarrow{b} 2 \xrightarrow{b} 3 \xrightarrow{a} 3 \xrightarrow{b} 2 \ \in F$$

Par contre , bba n'est pas accepé par A car :

$$1 \xrightarrow{b} 2 \xrightarrow{b} 3 \xrightarrow{a} 3 \ \notin F$$

g. Exemple :

L'Automate A représenté à la figure II.2 accepte exactement le langage formé des mots sur l'alphabet {a,b} et contenant un nombre impair de a.

$$L(A) = \{\omega \in \{a, b\}^* \quad | \quad |\omega|_a = 1 \ (\text{mod } 2)\}.$$

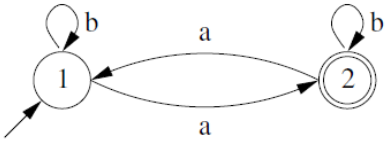


Figure II.2.2 : Un Automate fini déterministe

h. Remarque :

Pour simplifier les notations, on s'autorise à écrire q.ω

A lieu de δ(q, ω) si aucune confusion n'est possible.

i. Définition :

A tout mot ω de Σ^* correspond une unique suite d'états de A correspondant au chemin parcouru lors de la lecture de ω dans A. Cette suite s'appelle l'**exécution** de ω sur A.

Ainsi, si $\omega = \omega_0 \ldots \omega_k$, avec $\omega_i \in \Sigma$, alors l'exécution de ω sur A est :

$$(q_0, q_0.\omega_0, q_0.\omega_0\omega_1, \ldots, q_0.\omega_0 \ldots \omega_{k-1}, q_0.\omega_0 \ldots \omega_k).$$

j. Remarque :

On aurait pu aussi introduire des automates « infinis » en n'imposant pas la restriction à l'ensemble d'états Q d'être fini (mais nous supposerons quand même que l'alphabet reste fini),

Les notions d'exécution ou de langage accepté se transposent sans peine à ce cadre plus général.

II. Les Automates à nombre d'états fini non déterministes

Le modèle d'automate fini non déterministe généralise le cas des AFD. Comme nous le verrons bientôt, le non-déterminisme permet une plus grande souplesse bien utile dans certaines situations.

a. Définition :

Un automate fini non déterministe (AFND) est la donnée d'un quintuple :

$$A = (Q, I, F, \Sigma, \Delta) \quad \text{où}$$

- Q : est un ensemble fini d'états de l'automate;
- I C Q: ensemble des états initiaux;
- Σ : ensemble d'événements, ou symboles, alphabet de l'automate;
- Δ C Q x Σ^* x Q : est une **relation** de transition (qu'on supposera finie);
- F C Q : ensemble des états finaux.

Différence entre AFD et AFND :

- Dans le cas AFND, il est possible d'avoir **plus d'un** état initial : l'ensemble I;
- Les labels des arcs ne sont plus nécessairement **des lettres** mais bien **des mots** de Σ^*
- Et enfin, on n'a plus **une fonction de transition** mais une **relation de transition**.

Pour représenter les AFND, nous utilisons les mêmes conventions que pour les AFD.

b. Exemple :

L'automate de la figure II.3 est un AFND ayant :

- 1 et 3 comme états **initiaux**,
- 2 comme état **final** et
- La relation de **transition** est :

$$\Delta = \{(1,a,1), (1,a,3), (1,ba,2), (2,a,1), (2,b,3), (3,b,2),\}$$

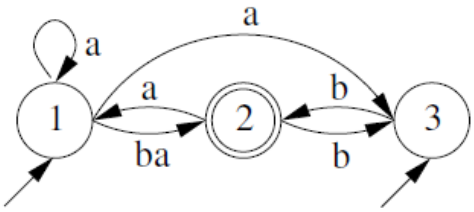


Figure II.3 : Un AFND

c. Définition :

Un mot $\omega = \omega_1 \ldots \omega_k$ est accepté par un AFND A = (Q, I, F, Σ, Δ)

- s'il existe $q_0 \in I$, $\ell \in \mathbf{N} \setminus \{0\}$, $u_1 \ldots u_\ell \in \Sigma^*$, $q_{1,\ldots,}q_\ell \in Q$ tels que :
- $(q_0, u_1, q_1), (q_1, u_2, q_2), \ldots, (q_{\ell-1}, u_\ell, q_\ell) \in \Delta$
- $\omega = u_1 \ldots u_\ell$
- Et $q_\ell \in F$

En d'autres termes, cette condition signifie qu'il existe **un chemin** dans le graphe associé à A débutant dans un **état initial**, de **label** ω et se terminant dans un état **final**.

Naturellement, le langage accepté par un AFND A est l'ensemble des mots acceptés par A et se note encore L(A).

Enfin, deux AFND A et B sont dits **équivalents** si L(A) = L(B).

d. Exemple

Si nous poursuivons l'exemple de la figure II.3, **le mot ab est accepté** car :

- 1 ϵ I,
- (1,a,3) ϵ Δ
- (3,b,2) ϵ Δ
- Et 2 ϵ F

Ceci se note schématiquement :

$$1 \xrightarrow{a} 3 \xrightarrow{b} 2$$

A un mot, il peut correspondre plus d'un chemin. Par exemple, au mot baa, il correspond les chemins :

$$3 \xrightarrow{b} 2 \xrightarrow{a} 1 \xrightarrow{a} 1$$

$$3 \xrightarrow{b} 2 \xrightarrow{a} 1 \xrightarrow{a} 3$$

Et

$$1 \xrightarrow{ba} 2 \xrightarrow{a} 1 .$$

Ce sont les 3 seules possibilité partant d'un état initial. **Le mot baa n'est donc pas accepté** par l'automate.

e. Remarque :

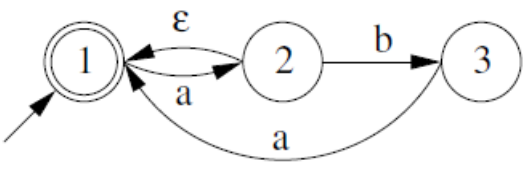
Dans la définition d'un AFND, rien n'empêche d'avoir des **transitions vides** du type :

$$(q,\text{Ɛ},q') \in \Delta$$

On parle parfois de **Ɛ –transitions**. En particulier, on suppose implicitement que pour tout état q d'un AFND, on a : $(q,\text{Ɛ},q) \in \Delta$

f. Exemple :

Considérons l'AFND suivant : [2]



II.4 : Un AFND avec Ɛ –transitions.

Cet automate **accepte le mot a** car on a le chemin :

$$1 \xrightarrow{a} 2 \xrightarrow{\text{Ɛ}} 1 \in F.$$

g. Définition : AFND élémentaire

Un AFND A = (Q,I,F,Σ,Δ) est qualifié d'élémentaire si pour tout (q,ω,q') ϵ Δ $|\omega| \leq 1$.

Comme le montre le lemme suivant, on peut dans le cadre des AFND se restreindre au cas d'automates élémentaires. En effet, tout AFND est équivalent à un AFND élémentaire.

Lemme : Tout langage accepté par un AFND est accepté par un AFND élémentaire.

h. Exemple : AFND élémentaire

Soit l'AFND non élémentaire A représenté à la figure : II.5 :

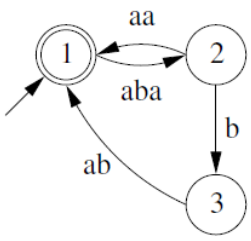
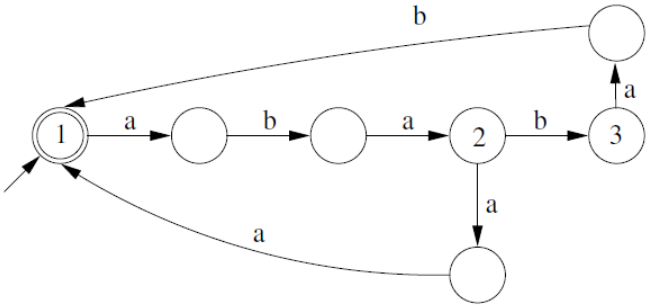


Figure II.5 : Un AFND non élémentaire.

On obtient alors un automate élémentaire équivalent à A représenté à la figure II.6, les états supplémentaires ne portant pas de numéro.



FigureII.6 : Un AFND élémentaire équivalent à A

Remarque :

Soit A = (Q,I,F,Σ,Δ) un AFND. Si R C Q et ω ϵ Σ^*, on note : R.ω l'ensemble des états atteints à partir des états de R en lisant ω.

Par exemple : avec l'automate de la figure II.4 :

{1}.a = {1,2} car :

$$1 \xrightarrow{a} \mathbf{2}$$

$$1 \xrightarrow{a} 2 \xrightarrow{\varepsilon} \mathbf{1}$$

On a aussi pour cet automate, {1}.b = Ø.

Dans le cas particulier de ω = Ɛ, on a toujours : R. Ɛ ⊃ R

Si L est un langage sur Σ, on pose :

$$R.L = \bigcup_{\omega \epsilon L} R.\omega$$

Le résultat suivant stipule que tout AFND est équivalent à un AFD. En d'autres termes, lorsqu'on s'intéresse à l'acceptation des mots d'un langage, un AFND n'est pas « plus puissant » qu'un AFD.

Proposition (Rabin et Scott) : **Tout langage accepté par un AFND est accepté par un AFD**.

i. Construction d'Automate par sous-ensembles :

Nous allons montrer un exemple d'automate construit à partir des sous-automates.

Des fois il est inutile de considérer tous les parties de Q. Seuls les états qui peuvent être atteints depuis Q_0 méritent d'être considérés.

Exemple :

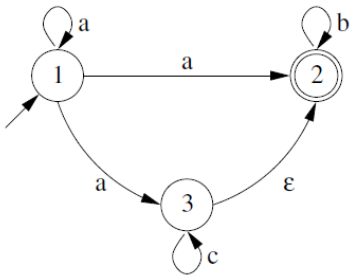


Figure II.8 : Un automate fini non déterministe

Soit un AFND représenté à la figure II.8. On remarque qu'il est élémentaire (sinon il faudrait le rendre élémentaire d'abord).

La construction du tableau des **sous-ensembles** peut être facilitée en établissant au préalable la table de transition de l'automate :

q	***q.a***	***q.b***	***q.c***
1	{1, 2, 3}		
2		2	
3		2	{2, 3}

On construit le tableau des sous-ensembles suivant de proche en proche, en l'initialisant avec I. Ɛ qui est ici :

$$\{1\}.Ɛ = \{1\}$$

A chaque étape, pour un sous-ensemble X d'états non encore traités, on détermine les valeurs de X.σ pour tout $\sigma \in \Sigma$.

La construction se termine une fois que tous les sous-ensembles d'états apparaissant ont été pris en compte.

X	***X.a***	***X.b***	***X.c***		***Nom***		
{1}	{1, 2, 3}	Ø	Ø		A		
{1, 2, 3}	{1, 2, 3}	{2}	{2, 3}		B		
Ø	Ø	Ø	Ø		C		
{2}	Ø	{2}	Ø		D		
{2, 3}	Ø	{2}	{2, 3}		E		

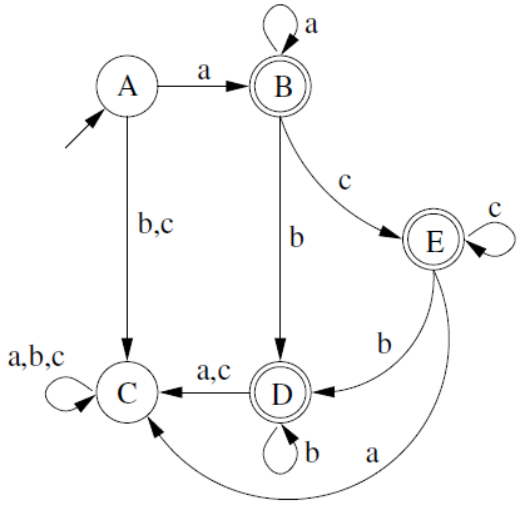


Figure II.9 Un AFD équivalent à l'AFND de la figure II.8

Si on pose A = {1}, B = {1, 2, 3}, C = Ø, D = {2} et E = {2,3},

Alors on a l'automate représenté à la figure II.9. Bien évidemment, les états finals de cet automate sont les sous-ensembles d'états de A qui contiennent 2 (le seul état final de A).

j. Exemple :

Considérons un AFND (élémentaire) acceptant, comme il est facile de le vérifier, le langage : a(ba)* $\cup$ a^*.

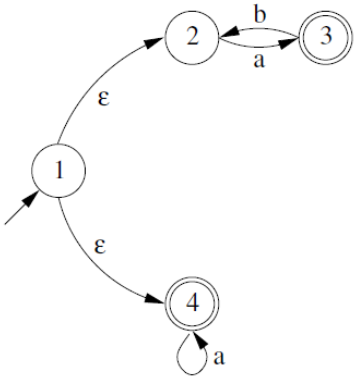


Figure II.10 un AFND acceptant a(ba)* $\cup$ a^*

Cet automate est représenté à la figure II.10. Ici, {1}.Ɛ = {1, 2, 4}.

Ainsi, la construction du tableau donne :

X	***X.a***	***X.b***
A= {1, 2, 4}	{3,4}	Ø
B = {3, 4}	{4}	{2}
C = Ø	Ø	Ø
D = {4}	{4}	Ø
E = {2}	{3}	Ø
F = {3}	Ø	{2}

Et on trouve l'automate de la figure II.11.

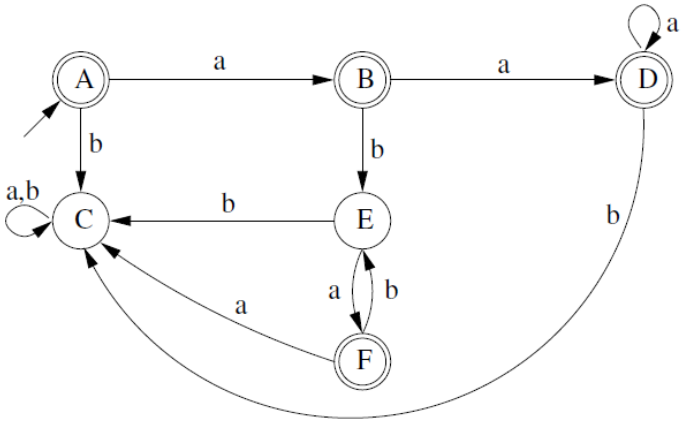


Figure II.11 un AFD acceptant a(ba)* ⋃ a^*

k. Remarque :

Il est clair que tout AFD est un cas particulier d'un AFND. Par conséquent, tout langage accepté par un AFD A est trivialement accepté par un AFND (à savoir A lui-même).

Puisque tout langage accepté par un AFND est accepté par un AFD, autrement dit les langages acceptés par les AFD et les AFND coïncident. Nous pourrons dès lors parler d'un langage accepté par un automate fini (sans autre précision).

Le nombre d'états peut croître de manière exponentielle lorsqu'on rend déterministe un AFND. Dans certaines situations, cette explosion du nombre d'états est inévitable et ce, même dans le cas d'un alphabet unaire.

l. Définition :

On définit un AFND A_k sur un alphabet unaire {a} comme suit. Cet automate possède un unique état initial à partir duquel on peut se déplacer dans k boucles disjointes.

Pour i = 1,…,k, la i-ième boucle est un cycle de longueur p_i où pi représente le i-ème nombre premier. Les états finals sont l'état initial et un état par cycle, de manière telle que le langage accepté par A_k soit

$$L(A_k) = \{a^n | n \in N.p_1 \cup N.p_2 \cup \ldots \cup N.p_k\} =: L_k$$

Un exemple, dans le cas k = 3, est représenté à la figure II.12

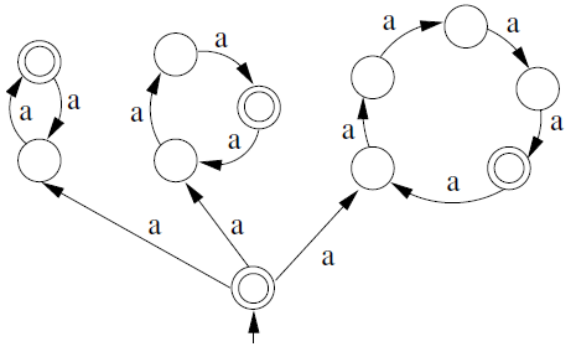


Figure II.12 L'Automate A_3.

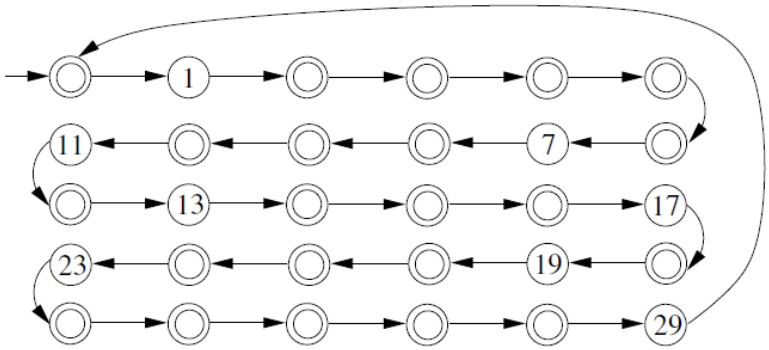


Figure II.13 Un AFD acceptant L_3

m. Proposition :

Le langage L_k accepté par l'AFND A_k possédant $1+p_1+p_2+\ldots+p_k$ états est accepté par un AFD ayant : $N = p_1p_2\ldots p_k$ états et aucun AFD ayant moins de N états n'accepte ce langage.

III. Stabilité des langages acceptés par automate

Nous montrons tout d'abord que l'ensemble des langages acceptés par un automate fini est stable pour les opérations rationnelles (c.-à-d., l'union, la concaténation et l'étoile de Kleene).

a. Proposition : Stabilité pour l'union

Si L(A) et L(B) sont les langages acceptés par deux automates finis A et B.

Alors $L(A) \cup L(B)$ est aussi accepté par un automate fini.

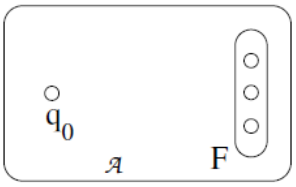


Figure II.15 Représentation symbolique d'un automate.

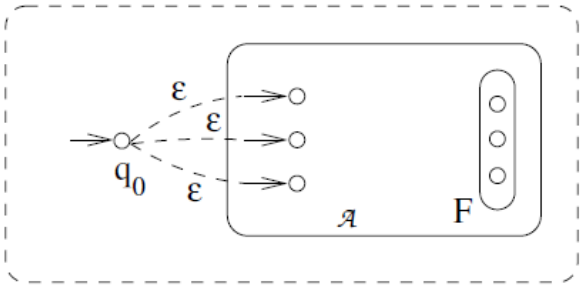


Figure II.16 Considérer un seul état initial n'est pas 1 restriction

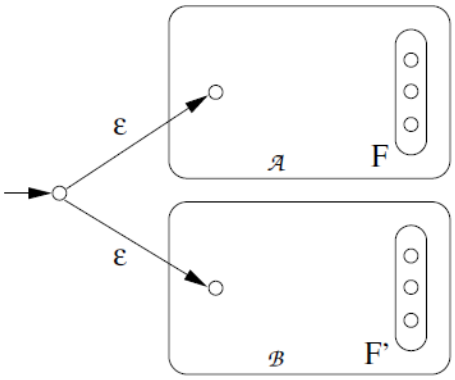


Figure II.17 Automate acceptant $L(A) \cup L(B)$

b. Proposition : Stabilité pour la concaténation et Kleene

Si L(A) et L(B) sont les langages acceptés par deux automates finis A et B.

Alors $L(A)\ L(B)$ est aussi accepté par un automate fini.

Et $L^*(A)$ l'est aussi.

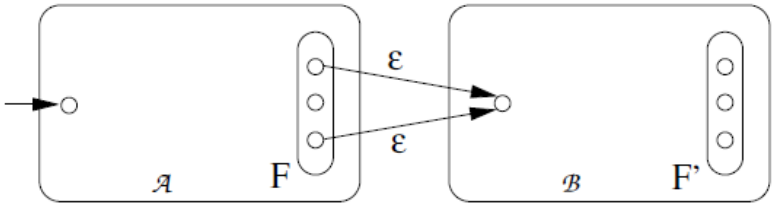


Figure II.17 Automate acceptant $L(A)\ L(B)$

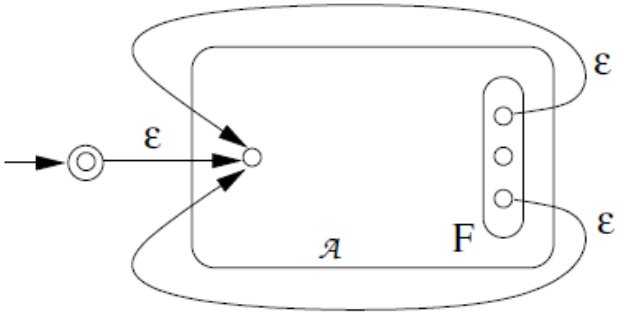


Figure II.17 Automate acceptant $L(A)^*$

c. Proposition : Stabilité pour opération de morphisme

Soit $L \subseteq \Sigma^*$ un langage accepté par un automate fini A.

Alors le langage $\Sigma^* \setminus L$ est aussi accepté par un automate fini.

d. Proposition

Soit $L \subseteq \Sigma^*$ un langage accepté par un automate fini A.

Et $f : \Sigma^* \to \Gamma^*$ *un morphisme de monoïdes*.

Alors le langage $f(L) \subseteq \Gamma^*$ est aussi accepté par un automate fini.

e. Proposition:

Si L est un langage accepté par un automate fini, alors L^R est aussi accepté par un automate fini.

f. Proposition

Si L est un langage accepté par un automate fini, alors Pref(L) est aussi accepté par un automate fini.

g. Proposition

Si L est un langage accepté par un automate fini, alors Suff(L) est aussi accepté par un automate fini.

h. …

IV. Produit d'Automates

a. Proposition :

Si L et M sont des langages acceptés par 2 automates finis, alors $L \cap M$ est aussi accepté par un automate fini.

b. Démonstration :

Supposons que les automates A et B possèdent le même alphabet Σ. Ainsi,

$$A = \left(Q^{(a)}, q_0^{(a)}, F^{(a)}, \Sigma, \delta^{(a)}\right) et\ B = (Q^{(b)}, q_0^{(b)}, F^{(b)}, \Sigma, \delta^{(b)})$$

Considérons l'automate P ayant :

- $Q^{(a)}\ x\ Q^{(b)}$ comme ensemble fini d'états,
- $(q_0^{(a)}, q_0^{(b)})$ comme état initial,
- $F^{(a)}\ x\ F^{(b)}$ comme ensemble d'états finals

Et dont la fonction de transition π est définie par :

Tel que π : $(Q^{(a)}\ x\ Q^{(b)})\ x\ \Sigma\ \rightarrow\ (Q^{(a)}\ x\ Q^{(b)})$

$$((q\ x\ q'), \sigma)\ \rightarrow (\delta^{(a)}(q, \sigma), \delta^{(b)}(q', \sigma))$$

Les mots acceptés par P sont exactement les mots $\omega \in \Sigma^*$ tels que :

$$\pi\left(\left(q_0^{(a)}, q_0^{(b)}\right), \omega\right)\ \in\ F^{(a)}\ x\ F^{(b)}$$

Ceci est équivalent à :

$$\delta^{(a)}\left(q_0^{(a)}, \omega\right) \in\ F^{(a)}\ \text{ et}$$

$$\delta^{(b)}\left(q_0^{(b)}, \omega\right)\ \in\ F^{(b)}$$

Et signifie que le langage accepté par P est $L(A)\ \cap\ L(B)$

c. Proposition :

Si L et M sont des langages acceptés par 2 automates finis, alors L Ш M est aussi accepté par un automate fini.

d. Démonstration

Supposons que les automates A et B possèdent des alphabets différents $\Sigma^{(a)}$ et $\Sigma^{(b)}$ tel quel :

$$\Sigma^{(a)} \cap\ \Sigma^{(b)} =\ \emptyset.$$

Ainsi,

$$A = \left(Q^{(a)}, q_0^{(a)}, F^{(a)}, \Sigma^{(a)}, \delta^{(a)}\right) et\ B = (Q^{(b)}, q_0^{(b)}, F^{(b)}, \Sigma^{(b)}, \delta^{(b)})$$

Considérons l'automate P ayant :

- $Q^{(a)}\ x\ Q^{(b)}$ comme ensemble fini d'états,
- $(q_0^{(a)}, q_0^{(b)})$ comme état initial,
- $F^{(a)}\ x\ F^{(b)}$ comme ensemble d'états finals,

- $\Sigma^{(a)} \cup \Sigma^{(b)}$ comme alphabet.

Et dont la fonction de transition π est définie par :

Tel que π : $(Q^{(a)} \, x \, Q^{(b)}) \, x \, \Sigma^{(a)} \cup \Sigma^{(b)} \rightarrow (Q^{(a)} \, x \, Q^{(b)})$

$$\pi : ((q\,, q'), \sigma) \rightarrow \begin{cases} \left(\delta^{(a)}(q, \sigma), q'\right) si \; \sigma \in \Sigma^{(a)} \\ \left(q, \delta^{(b)}(q', \sigma)\right) si \; \sigma \in \Sigma^{(b)} \end{cases}$$

Par construction, il est clair que $L(P) = L(A)$Ш L(B).

En effet, en lisant un mot ω, on ne peut atteindre un état final de P que si après avoir lu :

- Toutes les lettres de $\Sigma^{(a)}$ apparaissant dans ω, on se trouve dans un état de P dont la première composante est dans $F^{(a)}$.
- Et toutes les lettres de $\Sigma^{(b)}$ apparaissant dans ω, on se trouve dans un état de P dont la seconde composante est dans $F^{(b)}$.

Pour considérer le cas où les alphabets ne sont pas disjoints c.-à-d. : $\Sigma^{(a)} \cap \Sigma^{(b)} \neq \emptyset$.

Dans cette situation, on peut remplacer, par exemple, $\Sigma^{(b)} = \{\sigma_1, \dots, \sigma_n\}$

Par un nouvel alphabet, $\underline{\Sigma}^{(b)} = \{\underline{\sigma_1}, \dots, \underline{\sigma_n}\}$ de telle manière que $\Sigma^{(a)} \cap \Sigma^{(b)} = \emptyset$.

On applique dès lors la construction présentée ci-dessus.

Pour terminer, il suffit d'appliquer le morphisme f :

$$(\Sigma^{(a)} \cup \underline{\Sigma}^{(b)})^* \rightarrow (\Sigma^{(a)} \cup \Sigma^{(b)})^*$$

Défini par :

$$f\left(\underline{\sigma_i}\right) = \sigma_i \; \forall \, \underline{\sigma_i} \in \underline{\Sigma}^{(b)} et \; f(\sigma) = \sigma, \forall \, \sigma \in \Sigma^{(a)}$$

On conclut en utilisant la proposition concernant le morphisme de monoïdes vue ci-dessus.

e. Exemple :

Considérons les langages a^*b^* et $(cd)^*$ acceptés respectivement par les deux AFD de la figure II.21.

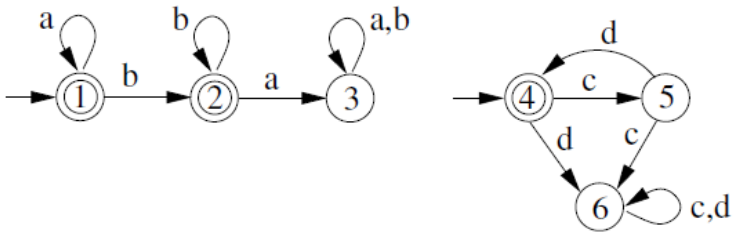


Figure II.21 AFD acceptant a^*b^* et $(cd)^*$

Les tables de transition sont :

q	$q.a$	$q.b$
1	1	2
2	3	2
3	3	3

q	$q.c$	$q.d$
4	5	6
5	6	4
6	6	6

Recherchons la table de transition de l'automate acceptant le langage : a^*b^* Ш $(cd)^*$,

q	$q.a$	$q.b$	$q.c$	$q.d$
$(1,4)$	$(1,4)$	$(2,4)$	$(1,5)$	$(1,6)$
$(1,5)$	$(1,5)$	$(2,5)$	$(1,6)$	$(1,4)$
$(1,6)$	$(1,6)$	$(2,6)$	$(1,6)$	$(1,6)$
$(2,4)$	$(3,4)$	$(2,4)$	$(2,5)$	$(2,6)$
$(2,5)$	$(3,5)$	$(2,5)$	$(2,6)$	$(2,4)$
$(2,6)$	$(3,6)$	$(2,6)$	$(2,6)$	$(2,6)$
$(3,4)$	$(3,4)$	$(3,4)$	$(3,5)$	$(3,6)$
$(3,5)$	$(3,5)$	$(3,5)$	$(3,6)$	$(3,4)$
$(3,6)$	$(3,6)$	$(3,6)$	$(3,6)$	$(3,6)$

Les états finals sont (1,4) et (2,4), l'état initial est (1,4).Si on renumérote les états de 1 à 9 dans l'ordre du tableau, on a l'AFD repris à la figure II.22

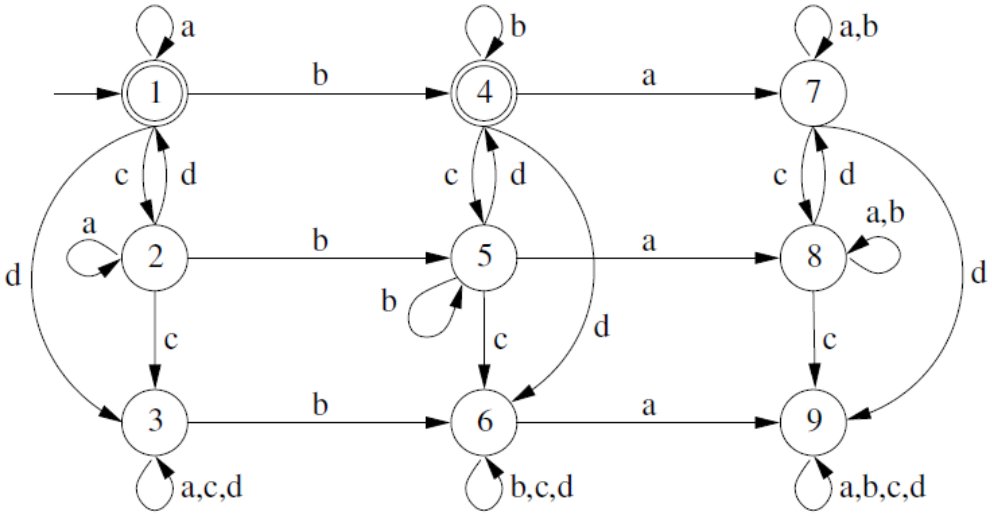


Figure II.22 AFD Shuffle