Cours: Théorie des langages & Compilation

Chapitre 3: Langages réguliers et Automates finis (partie B)

Réalisé par:
Dr. Sakka Rouis Taoufik

https://sourceforge.net/projects/tla-compilation/files/

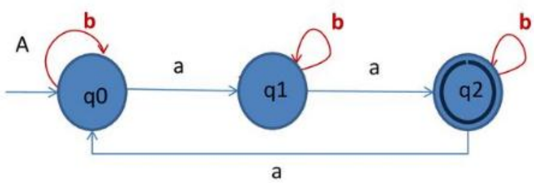
1

B.I. Langage accepté par un automate fini

Définition : Le langage reconnu par un automate est l'ensemble des chaînes qui permettent de passer de l'état initial à un état terminal.

Définition : Une configuration est un couple (q, ω) où q est l'état courant et ω est le reste du mot à lire.

Pour l'automate suivant, des configurations possibles sont:



(q0, aa) (q1, a) (q2, ε)

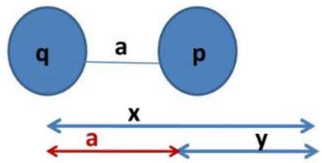
(q0, bba) (q0, ba) (q0, a) (q1, ε)

2

B.I. Langage accepté par un automate fini

Définition: Configuration successeur
Une configuration (p, y) est successeur d'une configuration (q, x) qu'on note:
(q, x) |-- (p, y)

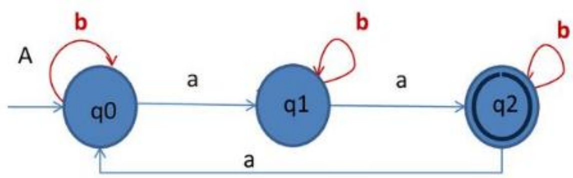
ssi



∃ un mot a / x = a y   et   δ(q, a) = {p}

B.I. Langage accepté par un automate fini

Pour l'automate suivant, des configurations possibles sont:



(q0, aa) |-- (q1, a) |-- (q2, ε)

(q0, bba) |-- (q0, ba) |-- (q0, a) |-- (q1, ε)

B.I. Langage accepté par un automate fini

Définition: Configuration k ^{ème} successeur

Une configuration (p, y) est k ^{ème} successeur d'une configuration (q, x) qu'on note:

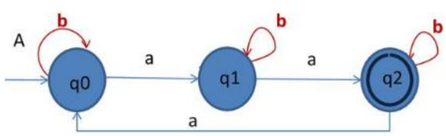
$(q, x) \vdash^{k} (p, y)$ ssi

$(q,x) \vdash (q_1, x_1) \vdash (q_2, x_2) \vdash \ldots (q_k, x_k) = (p, y)$

$(q, x) \vdash^{*} (p, y)$ ssi $\exists\, k >= 0 \,/\, (q,x) \vdash^{k} (p, y)$

B.I. Langage accepté par un automate fini

Pour l'automate suivant, nous avons :



$(q0, aa) \vdash^{*} (q2, \varepsilon)$ puisque: $\exists\, k \geq 0 \,/\, (q0, aa) \vdash^{k} (q2, \varepsilon)$

$(q0, aa) \vdash (q1, a) \vdash (q2, \varepsilon)$ $(k = 2)$

$(q0, bba) \vdash^{3} (q1, \varepsilon)$

En effet, $(q0, bba) \vdash (q0, ba) \vdash (q0, a) \vdash (q1, \varepsilon)$

B.I. Langage accepté par un automate fini

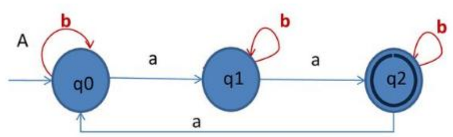
Définition : Soit A = (Q, X, q0, δ, F) un automate fini.
Le langage accepté (ou reconnu par A) est noté L(A) /

$$L(A) = \{\omega / (q0, \omega) \mid\overset{*}{--} (qf, \varepsilon),\ qf \in F\}$$

ω est accepté par A ssi $\omega \in L(A)$

B.I. Langage accepté par un automate fini

Exercice : Soit l'automate fini A suivant :



1) Montrer que aa ∈ L(A) et que bba ∉ L(A)

Corrigé:

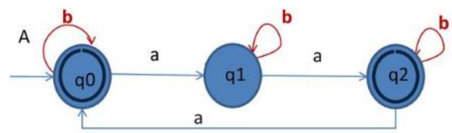
aa ∈ L(A) par ce que : $(q0, aa) \mid\overset{*}{--} (q2, \varepsilon)$ puisque:
$(q0, aa) \mid-- (q1, a) \mid-- (q2, \varepsilon)$ (k = 2)

bba ∉ L(A) par ce que : $\neg\exists\, k / (q0, bba) \mid\overset{k}{--} (q2, \varepsilon)$
En effet, $(q0, bba) \mid-- (q0, ba) \mid-- (q0, a) \mid-- (q1, \varepsilon)$

Ch3: Langages réguliers et Automates finis

B.I. Langage accepté par un automate fini

Exercice: Considérons l'automate A suivant :



1) Quel est le langage reconnu par A

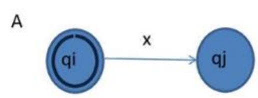
2) Est-ce que le mot ε est accepté par A

9

Ch3: Langages réguliers et Automates finis

B.I. Langage accepté par un automate fini

Lemme d'Arden 1: Soit A un automate fini sur un vocabulaire X Avec $x \in X$



On dénote par Li, le langage reconnu par l'automate A en considérant que qi est l'état initial. Ainsi, L0 est le langage reconnu par A

$\omega_j \in Lj \Leftrightarrow \exists\, \omega_i \in Li \,/\, \omega_i = x\,\omega_j$ en conséquence Li = xLj

Si qi est un état final alors Li = x Lj + {ε} par ce que si qi est un état initial et final alors le langage Li contient le mot ε

Lemme d'Arden 2: Soient A et B deux langages sur un vocabulaire X,

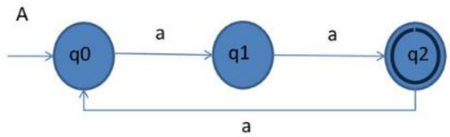
Les équations L= AL + B et L = LA + B admettent respectivement comme solution minimale L=A*B et L=BA*. Cette solution est unique si ε ∉ A.

10

B.I. Langage accepté par un automate fini

Exercice : Soit A un automate fini sur un vocabulaire X= {a,b}.

Chercher le langage L0 reconnu par l'automate A



Corrigé :

L0 = {a} L1 (1) L1 = {a} L2 (2)

L2 = {a} L0 + {ε} (3)

(1) + (2) $\Rightarrow$ L0 = {a}{a}L2 = {aa}L2 (4)

(4) + (3) $\Rightarrow$ L0 = {aa} L2 = {aa}({a}L0 + {ε}) $\Rightarrow$ L0 = {aaa}L0 + {aa}

$\Rightarrow$ L0 = {aaa}*{aa} = (aaa)*aa

D'après le lemme d'Arden

Ch3: Langages réguliers et Automates finis

B.I. Langage accepté par un automate fini

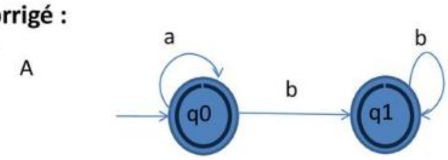
Exercice:

1) Construire un automate qui accepte le langage a*b*

2) Vérifier que l'automate construit reconnaît le langage a*b*

Corrigé :

1)



2) Calculons L0

L0 = {a} L0 + {b}L1 + {ε} (1)

L1 = {b} L1 + {ε} (2)

(2) $\Rightarrow$ L1 = {b}*{ε} d'après le lemme d'Arden $\Rightarrow$ L1 = b* (3)

(1) + (3) $\Rightarrow$ L0 = {a}L0 + {b} b* + {ε}

$\Rightarrow$ L0 = {a}L0 + b^+ + {ε}

$\Rightarrow$ L0 = {a}L0 + b^* car b^++{ε} = b*

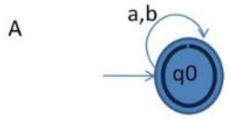
$\Rightarrow$ **L0** = {a}*b* = **a*b***d'après le lemme d'Arden

L(A) = L0 = a*b* donc l'automate A reconnaît le langage a*b*

B.I. Langage accepté par un automate fini

Exercice:
1) Construire une automate qui accepte le langage (a+b)*
2) Vérifier que l'automate construit reconnaît le langage (a+b)* (L0 est le langage accepté par l'automate puisque c'est le langage où q0 est l'état initial)

Corrigé:
1)



2) $L0 = \{a\} L0 + \{b\} L0 + \{\varepsilon\}$ (1)

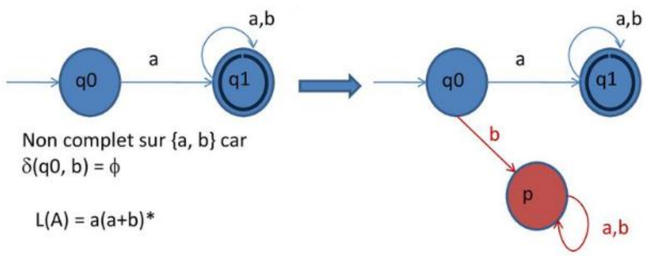
(1) $\Rightarrow L0 = \{a,b\} L0 + \{\varepsilon\}$

$\Rightarrow L0 = \{a,b\}^*\{\varepsilon\}$ d'après le lemme d'Arden

$\Rightarrow L0 = \{a,b\}^* = (a+b)^*$

B.II. Déterminisation d'un automate à états fini

Cas 1. Automate fini non complet
Si l'automate fini est non déterministe par ce qu'il est non complet, alors pour le rendre déterministe, il suffit d'ajouter un état puis ajouter toutes les transitions manquantes vers cet état.



Non complet sur {a, b} car $\delta(q0, b) = \phi$

$L(A) = a(a+b)^*$

Complet sur {a, b}

B.II. Déterminisation d'un automate à états fini

Cas 2. Automate fini ambigu

Si l'automate fini est non déterministe par ce qu'il est ambigu alors on doit construire un automate en groupant les états visités par un même symbole à partir d'un état, ces états groupés forment les nouveaux états de l'automate déterministe.

La construction de l'automate déterministe suit les étapes suivantes:

Etape 1. Définir les nouveaux groupes d'états

Etape 2. Renommer les nouveaux groupes et définir les états finaux.
Un nouvel état (Groupe d'état) est un état final ssi il contient un ancien état final.

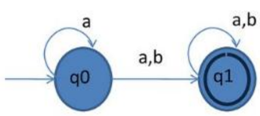
Etape 3. Construire l'automate déterministe

B.II. Déterminisation d'un automate à états fini

Cas 2. Automate fini ambigu

Exemple1:

Etape 1. Définir les nouveaux groupes d'états



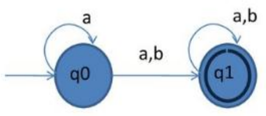
$L(A) = a^*(a+b)^+$

	a	b
{q0}	{q0, q1}	{q1}
{q0, q1}	{q0, q1}	{q1}
{q1}	{q1}	{q1}

B.II. Déterminisation d'un automate à états fini

Cas 2. Automate fini ambigu

Etape 2. Renommer les nouveaux groupes et définir les états finaux.

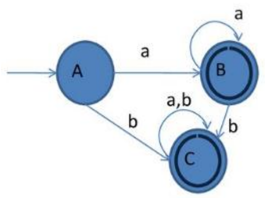


	a	b
{q0} A	{q0, q1} B	{q1} C
{q0, q1} B	{q0, q1} B	{q1} C
{q1} C	{q1} C	{q1} C

B.II. Déterminisation d'un automate à états fini

Cas 2. Automate fini ambigu

Etape 3. Construire l'automate déterministe

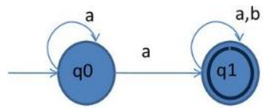


	a	b
{q0} A	{q0, q1} B	{q1} C
{q0, q1} B	{q0, q1} B	{q1} C
{q1} C	{q1} C	{q1} C

B.II. Déterminisation d'un automate à états fini

Cas 2. Automate fini ambigu

Exercice : Soit l'automate A suivant.



Construire un automate fini déterministe qui reconnaît le même langage

Correction :

Etape 1. Définir les nouveaux groupes d'états

$L(A) = a^+(a+b)^*$

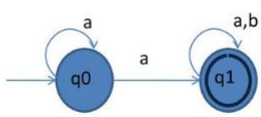
	a	b
{q0}	{q0, q1}	ϕ
{q0, q1}	{q0, q1}	{q1}
{q1}	{q1}	{q1}

Ch3: Langages réguliers et Automates finis

B.II. Déterminisation d'un automate à états fini

Cas 2. Automate fini ambigu

Etape 2. Renommer les nouveaux groupes et définir les états finaux.

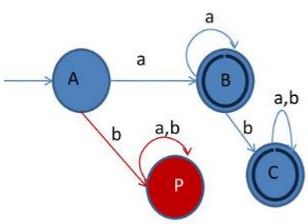


	a	b
{q0} A	{q0, q1} B	ϕ P
{q0, q1} B	{q0, q1} B	{q1} C
{q1} C	{q1} C	{q1} C

B.II. Déterminisation d’un automate à états fini

Cas 2. Automate fini ambigu

Etape 3. Construire l’automate déterministe



	a	b
{q0} A	{q0, q1} B	ϕ P
{q0, q1} B	{q0, q1} B	{q1} C
{q1} C	{q1} C	{q1} C

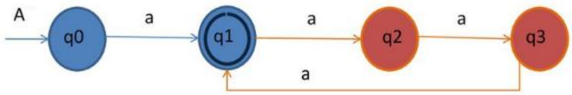
B.III. Minimisation d’un automate fini déterministe

Soit A un automate fini déterministe A = (Q, X, q0, δ, F)
Pour obtenir un automate A minimal en nombre d’états, suivre les étapes suivantes:

1) Construire une partition initiale π_0 composée de deux groupes: Groupe des états non finaux G1et Groupe des états finaux G2
2) Répéter Jusqu’à $\pi_{i+1} = \pi_i$ / i >= 0
•Pour obtenir π_{i+1}, Partitionner chaque groupe de π_i en mettant ensemble des états p et q si pour chaque symbole s du vocabulaire, p et q transitent vers des états d’un même groupe d’états.
•Construire les nouveaux groupes
3) Associer à chaque groupe un nom
4) Construire les transitions des groupes en utilisant les transitions des états des groupes.
5) Un groupe qui contient un état final est un état final dans l’automate minimal

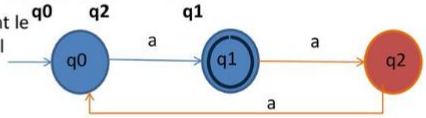
B.III. Minimisation d'un automate fini déterministe

Exercice1 : Soit l'automate A suivant. Construire un automate minimal qui accepte le même langage (Minimiser le nombre d'états de l'automate A)



Corrigé:

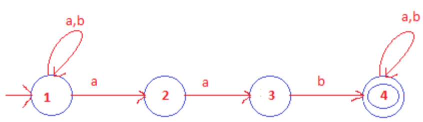
	G1		G2
Π0	(q0, q2, q3)		(q1)
	G1	G2	G3
Π1	(q0, q3)	(q2)	(q1)
Π2	(q0, q3)	(q2)	(q1)
On obtient le A minimal	q0	q2	q1



(q2, a, q0) est dans A minimal car (q2, a, q3) était dans A et que le group d'états (q0, q3) est renommé q0

B.III. Minimisation d'un automate fini déterministe

Exercice 2 : Soit l'automate A1 suivant.

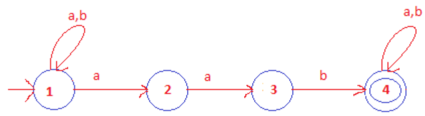


1) Construire un automate fini déterministe D qui reconnaît le même langage

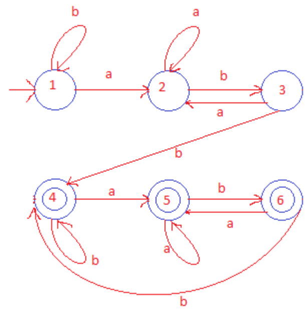
2) Construire un automate fini déterministe minimal M qui accepte le même langage (Minimiser le nombre d'états de l'automate D)

B.III. Minimisation d’un automate fini déterministe

Correction: automate fini déterministe D



δ	a	b
{1}	{1,2}	{1}
{1,2}	{1,2}	{1,3}
{1,3}	{1,2}	{1,4}
{1,4}	{1,2,4}	{1,4}
{1,2,4}	{1,2,4}	{1,3,4}
{1,3,4}	{1,2,4}	{1,4}



Ch3: Langages réguliers et Automates finis

B.III. Minimisation d’un automate fini déterministe

Correction: automate fini minimal M

	1	2	3	4	5	6
π_0	I	I	I	II	II	II
a	I	I	I	II	II	II
b	I	I	II	II	II	II
π_1	I	I	II	III	III	III
a	I	I	I	III	III	III
b	I	II	III	III	III	III
π_2	I	II	III	IV	IV	IV
a	II	II	II	IV	IV	IV
b	I	III	III	IV	IV	IV
π_3	I	II	III	IV	IV	IV