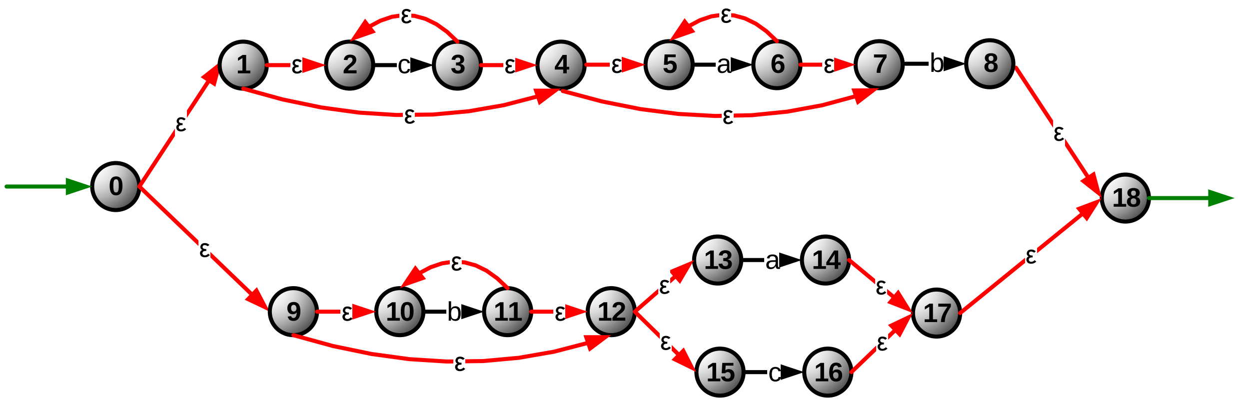
DE : Mathématiques pour l'informatique, version 2

Solutions

Exercice 1.

Construire, en respectant à la lettre les règles données en cours, l'automate asynchrone dont le langage est donné par l'expression rationnelle suivante : c*a*b + b*(a+c)

Solution



Exercice 2. Déterminiser l'AF suivant :

		a	b
	A	E	C
E	B	B	D
S	C	F	A
S	D	D	F
E	E	D	C
	F	F	E

Le résultat est attendu <u>sous forme d'un tableau de transitions</u> avec les entrées et les sorties indiquées.

Solution

		a	b
E	BE	BD	CD
S	BD	BD	DF
S	CD	DF	AF
S	DF	DF	EF
	AF	EF	CE
	EF	DE	CE
S	CE	DF	AC
S	AC	EF	AC

Exercice 3

Minimiser l'automate suivant, en détaillant le processus de minimisation (partitions successives). Le résultat est attendu sous forme d'un tableau de transitions avec les entrées et les sorties indiquées.

	état	a	b	c
	A	G	–	–
E	B	C	E	–
S	C	A	F	E
	D	C	E	–
S	E	D	G	C
	F	C	E	–
	G	–	C	–

'E' = état initial, ou "d'entrée"
'S' = état terminal, ou "de sortie"

Le résultat est attendu sous forme d'un tableau de transitions avec les entrées et les sorties indiquées.

Solution

Tout d'abord il faut compléter l'automate qui est déjà déterministe :

	état	a	b	c
	A	G	P	P
E	B	C	E	P
S	C	A	F	E
	D	C	E	P
S	E	D	G	C
	F	C	E	P
	G	P	C	P
	P	P	P	P

Θ_0={T, NT} avec T={C,E}, NT={A,B,D,F,G,P}. 1re itération :

	a	b	c	sous Θ_0			
C	A	F	E	NT	NT	T	pas de séparation
E	D	G	C	NT	NT	T	

A	G	P	P	NT	NT	NT	séparation en trois groupes
B	C	E	P	T	T	NT	
D	C	E	P	T	T	NT	
F	C	E	P	T	T	NT	
G	P	C	P	NT	T	NT	
P	P	P	P	NT	NT	NT	

Θ_1={T, (B,D,F), (A,P), G}. Nommons les sous-groupes en fonction de leur contenu : CE,BDF,AP, G.

Itération 2 :

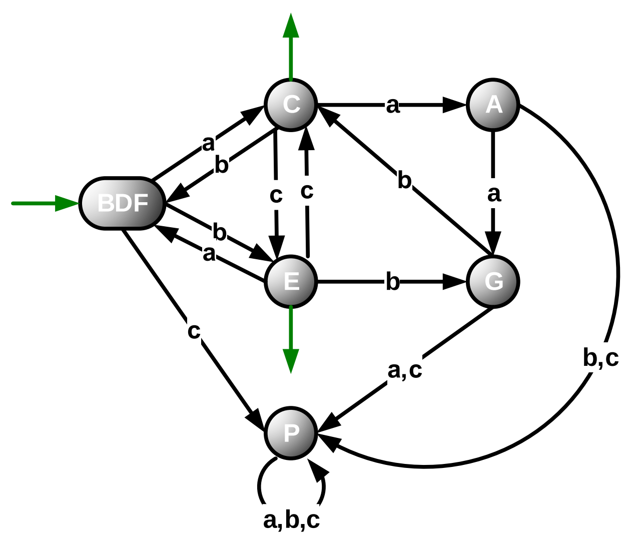
	a	b	c	sous Θ_1			
C	A	F	E	AP	BDF	CE	séparation en 2 groupes
E	D	G	C	BDF	AP	CE	
B	C	E	P	CE	CE	AP	pas de séparation
D	C	E	P	CE	CE	AP	
F	C	E	P	CE	CE	AP	
A	G	P	P	G	AP	AP	séparation en 2 groupes
P	P	P	P	AP	6P	6P	

Θ_2={(B,D,F), A, C, E, G, P}.

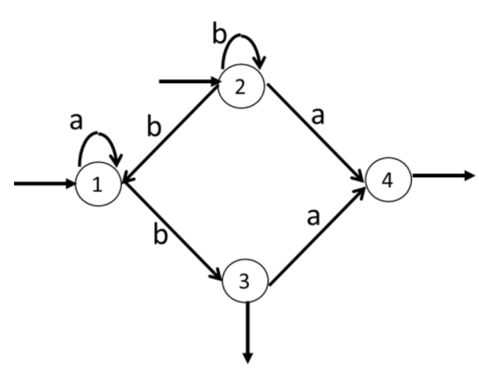
Pour le seul groupe composé de plus d'un état restant BDF il est impossible qu'il se sépare car les transitions de ses composantes sont les mêmes. Donc $\Theta_2 = \Theta_{fin}$.

La table de transitions de l'automate minimal (et on donne ici le schéma même s'il n'a pas été demandé):

		a	b	c
E	BDF	C	E	P
S	C	A	BDF	E
S	E	BDF	G	C
	A	G	P	P
	G	P	C	P
	P	P	P	P



Exercice 4. Obtenir sous forme d'une expression rationnelle le langage de l'automate ci-contre :



Vous pouvez utiliser, au choix :

- Soit la méthode d'élimination des états (en éliminant les états dans l'ordre que vous voulez)
- Soit la méthode de l'arrivée ou celle du départ, avec les équations.

Quelle que soit la méthode utilisée, toutes les étapes intermédiaires doivent apparaître dans votre réponse.

Solution

(i) Méthode de l'arrivée

$1 = \varepsilon + 1a + 2b$ (eq 1)
$2 = \varepsilon + 2b$ (eq 2)
$3 = 1b$ (eq 3)
$4 = 2a + 3a$ (eq 4)

$L = 3 + 4$

Eq 2 $\rightarrow$ $2 = \varepsilon b^* = b^*$. (Lemme d'Arden)
Eq 1 $\rightarrow$ $1 = \varepsilon + 1a + b^*b$ et (Lemme d'Arden) $1 = (\varepsilon + b^*b)a^* = b^*a^*$ car $\varepsilon + XX^* = X^*$
$3 = ab = b^*a^*b$
$4 = 2a+3a = b^*a + b^*a^*ba$

$L = 3 + 4 = b^*a + b^*a^*(b + ba)$

(ii) Méthode du départ (pas donné dans les groupes A,B,C,D)

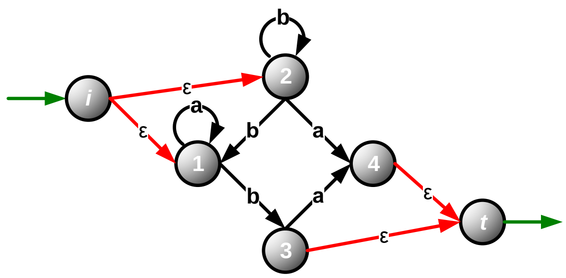
$1 = a1+b3$ (eq 1')
$2 = b2 + b1 + a4$ (eq 2')
$3 = a4 + \varepsilon$ (eq 3')
$4 = \varepsilon$ (eq 4')

$L = 1 + 2$

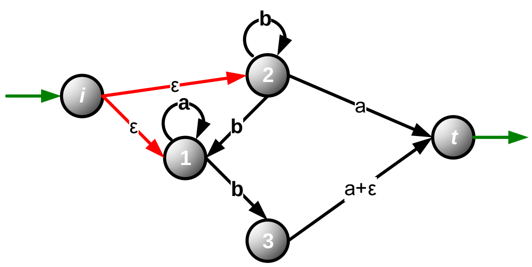
$3 = a4 + \varepsilon = a + \varepsilon$
$1 = a1 + b3 = a1 + b(a + \varepsilon) = a1 + ba + b$ et (Lemme d'Arden) $1 = a^*(ba + b)$
$2 = b1 + b2 + a4 = b2 + ba^* (ba + b) + a$ et (Lemme d'Arden) $2 = b^*(ba^* (ba + b) + a) = b^*ba^*(ba + b) + b^*a$
$L = 1 + 2 = a^*(ba + b) + b^*ba^*(ba + b) + b^*a = b^*a^*(ba + b) + b^*a$.

(iii) Méthode d'élimination d'états

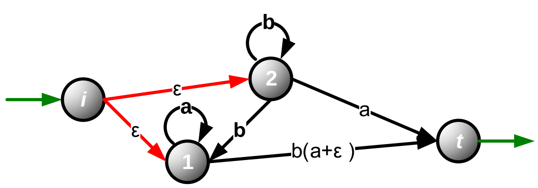
1) Initialisation :



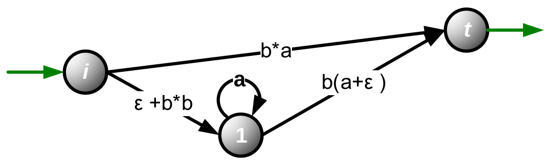
2a) élimination de 4 :



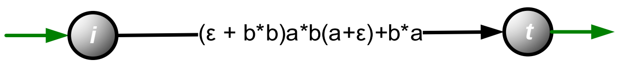
2b) élimination de 3 :



2c) élimination de 2 :



2d) élimination de 1 :



En utilisant ε + b*b=b* (car ∀ X, ε +X*X=X*), nous obtenons
L=b*a*(ba+b) + b*a

Exercice 5. Obtenir un AF dont le langage est le complément du langage **reconnu par** l'AF suivant :

		a	b
E/S	A	B	C
S	B	--	D
S	C	E	--
S	D	--	E
S	E	D	B

Résultat : sous forme <u>d'une table de transitions</u> avec les entrées et les sorties indiquées.

Solution :

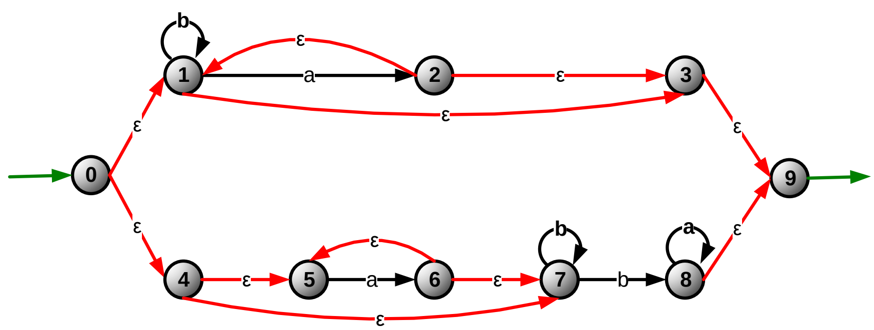
D'abord il faut compléter cet AD :

		a	b
E/S	A	B	C
S	B	P	D
S	C	E	P
S	D	P	E
S	E	D	B
	P	P	P

Maintenant on peut transformer les états terminaux en non terminaux et vice versa pour obtenir un automate reconnaissant le langage complémentaire :

		a	b
E	A	B	C
	B	P	D
	C	E	P
	D	P	E
	E	D	B
S	P	P	P

Exercice 6. Déterminiser l'automate asynchrone suivant :



Résultat : sous forme d'une table de transitions avec les entrées et les sorties indiquées

Solution :

Déterminisation :

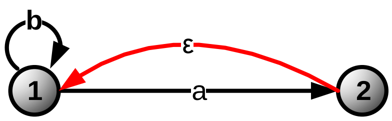
ε-clôtures:	
0' = 0134579	terminal
1' = 139	terminal
2' = 1239	terminal
6' = 567	
7' = 7	
8' = 8	terminal

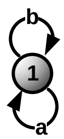
		a	b
E/S	0'	2'6'	1'7'8'
S	2'6'	2'6'	1'7'8'
S	1'7'8'	2'8'	1'7'8'
S	2'8'	2'8'	1'
S	1'	2'	1'
S	2'	2'	1'

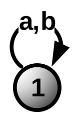
sans utiliser les ε-clôtures			
		a	b
E/S	0134579	1235679	13789
S	1235679	1235679	13789
S	13789	12389	13789
S	12389	12389	139
S	139	1239	139
S	1239	1239	139

Analyse du résultat (non demandé !)

Nous avons obtenu un ADC dont tous les états sont terminaux, donc l'automate d'origine reconnait A*=(a+b)*. Cela parait étrange ; pourquoi ?
Regardons de près la branche supérieure de l'automate asynchrone. Un de ses éléments est



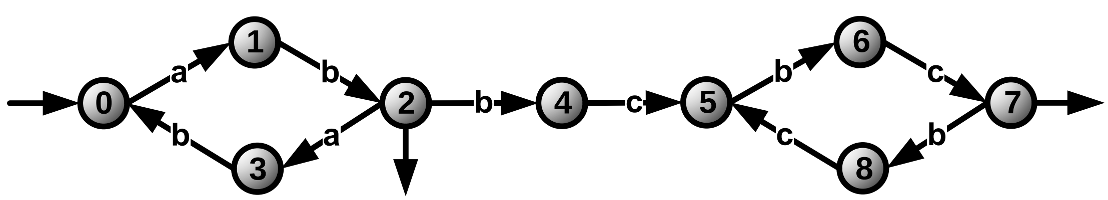
Mais vu que aε = a, c'est équivalent à 

et donc à 

. Or, l'état 1 est sur un chemin en ε entre l'entrée 0 et la sortie 9. Donc peu importe ce qu'est reconnu sur la branche inférieure, l'automate reconnait déjà A*.

Exercice 7. Construire un AF sur l'alphabet {*a,b,c*} reconnaissant le langage dont les mots commencent par un nombre impaire (y compris 0) de '*ab*' suivis d'un nombre paire (y compris 0) de '*bc*'. Par exemple, les mots *'abbcbc', 'ab', 'ababab', 'abababbcbc',* appartiennent au langage, tandis que les mots *'bcbc', 'bc', 'ababbc', 'bcabab'* etc., non.
Attention : le '*b*' ne peut pas appartenir simultanément à '*ab*' et à '*bc*', de façon que '*abababc*' ne satisfait pas à la condition.
Solution (une version parmi d'autres)

Remarque. Une grosse erreur de copier-coller a été faite. La phrase « un nombre impaire (y compris 0) » n'a aucun sens : 0 n'est pas un nombre impaire, il est paire. (Et, évidemment, il y a une contradiction entre cette possibilité de ne pas faire des 'ab' et la phrase « 'bcbc' n'appartient pas au langage). Cela veut dure qu'il faut que nous acceptions et les solutions où on a bien compris que c'était une erreur de copier-coller, et celles où le mot vide fait partie du langage, ainsi que tous les autres mots du type $(bc)^{paire}$.



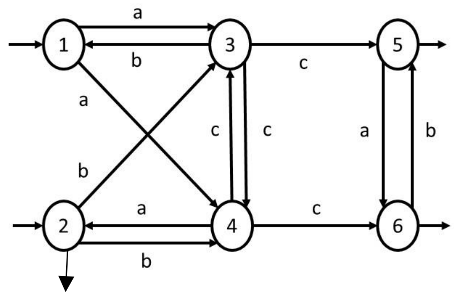
Je remarque au passage que tant que beaucoup d'élèves ont inclut 0 fois 'ab' dans leur solution, ***pas un seul n'a écrit que compter 0 parmi les nombres impaires est une erreur !***

Ceci est un automate tel que nous l'avions envisagé. Si on veut un automate prenant en compte l'erreur de frappe comme si elle n'était pas une erreur, il faut ajouter une flèche d'entrée sur l'état 2.

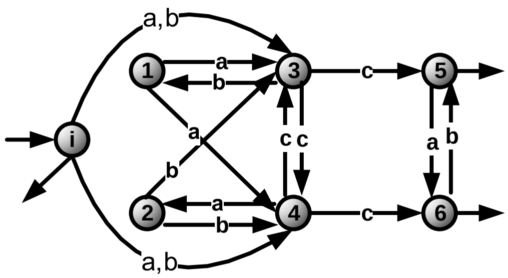
Remarque : certains élèves ont construit cet automate avec des ε-transitions. Ce n'est pas du tout interdit par l'énoncé , pourquoi pas.

Exercice 8

a) Standardiser l'automate suivant :

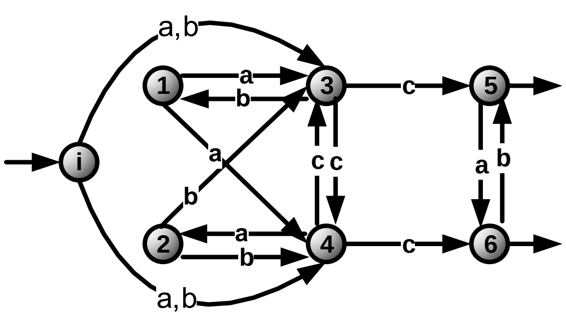


Solution :



b) Est-ce que votre automate standardisé reconnaît le mot vide ? (Expliquer la réponse !).
Si la réponse est « oui », donner un automate reconnaissant le même langage à l'exception du mot vide.
Si elle est « non », donner un automate reconnaissant le même langage plus le mot vide.

Solution : Oui, car l'état *I* est une sortie (ce qui est la consequence du fait que l'automate d'origine reconnait le mot vide car l'entrée 2 est une sortie).
L'AF reconnaissant le même langage à l'exception du mot vide:



9. Questions de cours :

a) Soit un automate déterministe complet non minimal A. Pour obtenir l'automate déterministe complet minimal reconnaissant le langage complémentaire à celui reconnu par A (choisir la réponse correcte) :

- **on doit** effectuer l'opération T↔NT (transformer chaque état terminal en état non terminal, et chaque état non terminal, en état terminal) sur A et puis minimiser l'automate obtenu
- **on doit** minimiser A et puis effectuer l'opération T↔NT sur l'automate obtenu
- **on peut faire l'une ou l'autre des deux méthodes et** on obtient la même chose dans les deux cas

Réponse : on obtient la même chose dans les deux cas.

b) Quel est l'automate minimal équivalent à l'automate suivant :

		a	b	c	d
E/S	0	3	2	1	2
S	1	0	2	3	1
S	2	3	3	0	0
S	3	2	1	1	1

Donner une réponse immédiate, **sans effectuer la procédure de minimisation par des partitions successives** mais avec une explication (une phrase ou deux) !

Réponse : c'est un ADC dont tous les états sont terminaux. Donc, l'automate minimale est :