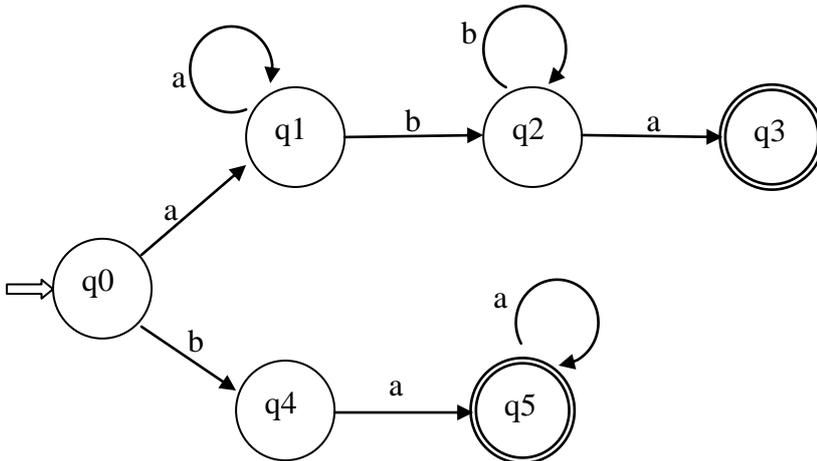


CORRIGÉ ABREGÉ DE LA SÉRIE D'EXERCICES n° 2 de ThL

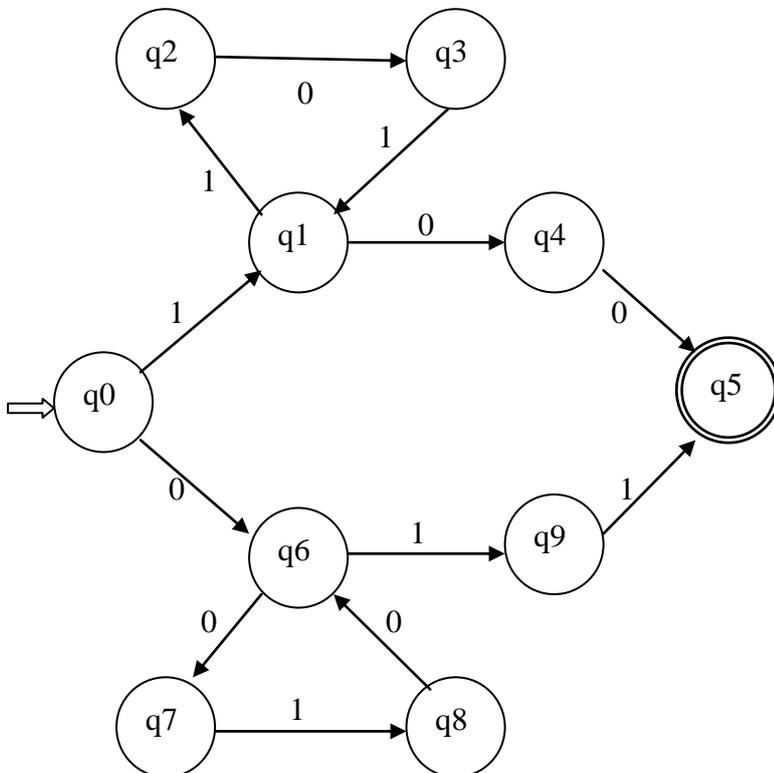
par : S. Khemliche, M.S. Habet, Y. Yesli

EXERCICE 1 :

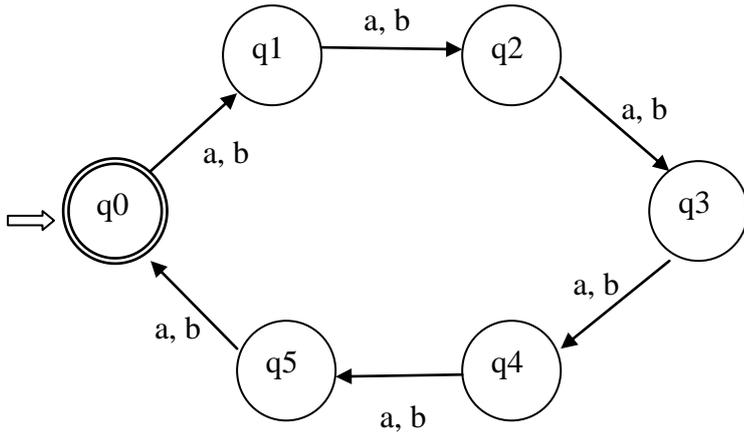
a) $L_1 = \{ w \in \{a, b\}^* / w = a^n b^m a \text{ ou } w = b a^n ; n, m \geq 1 \}$:



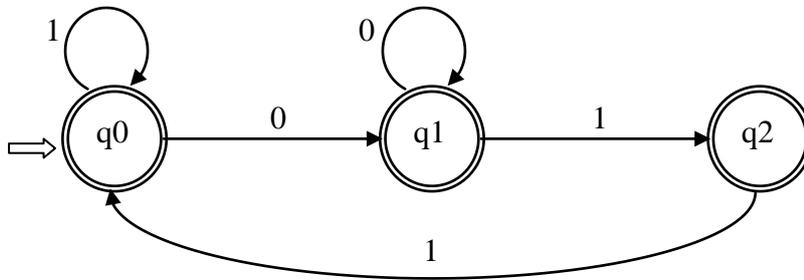
b) $L_2 = \{ w \in \{0, 1\}^* / w = 1(101)^n 00 \text{ ou } w = 0(010)^n 11, n \geq 0 \}$:



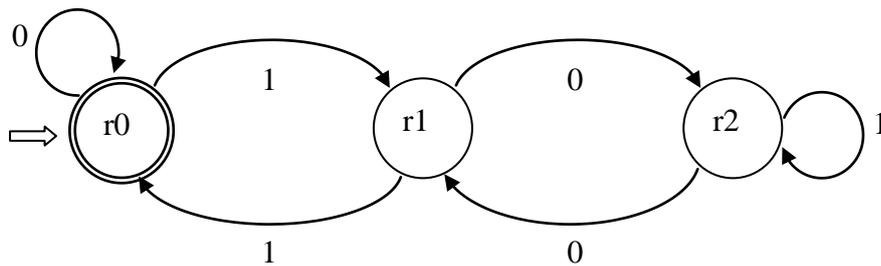
c) $L_3 = \{ w \in \{a, b\}^* / |w| \equiv 0 [6] \}$:



d) $L_4 = \{ w \in \{0, 1\}^* / w \text{ ne contient pas la sous chaîne «010» } \}$:

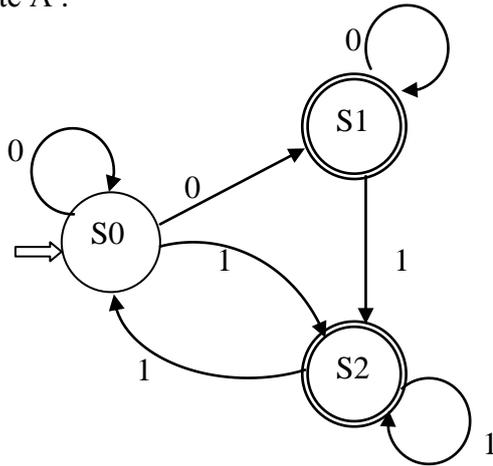


e) $L_5 =$ ensemble des mots de $\{0, 1\}^*$ représentant les nombres divisibles par 3 (dans le système de numération binaire naturel) :

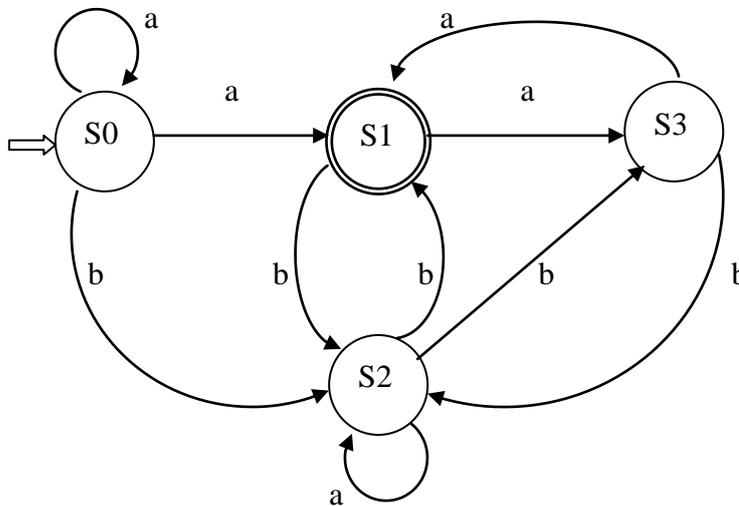


EXERCICE 2 :

1) L'automate A :



L'automate B :



2) Grammaire régulière à droite équivalente à A :

$G_A = (\{0, 1\}, \{S, A, B\}, S, P_A)$, où P_A contient les règles :

$S \rightarrow 0S \mid 0A \mid 1B$; $A \rightarrow 0A \mid 1B \mid \varepsilon$; $B \rightarrow 1S \mid 1B \mid \varepsilon$

Grammaire régulière à droite équivalente à B :

$G_B = (\{a, b\}, \{S, A, B, C\}, S, P_B)$, où P_B contient les règles :

$S \rightarrow aS \mid aA \mid bB$; $A \rightarrow bB \mid aC \mid \varepsilon$; $B \rightarrow bA \mid aB \mid bC$; $C \rightarrow aA \mid bB$

3) Table de transition de l'automate déterministe équivalent à A : (les états soulignés sont des états finaux)

	0	1
$\langle S0 \rangle = q0$	$\langle S0, S1 \rangle$	$\langle S2 \rangle$
$\langle S0, S1 \rangle = q1$	$\langle S0, S1 \rangle$	$\langle S2 \rangle$
$\langle S2 \rangle = q2$	/	$\langle S0, S2 \rangle$
$\langle S0, S2 \rangle = q3$	$\langle S0, S1 \rangle$	$\langle S0, S2 \rangle$

L'automate déterministe équivalent à A :

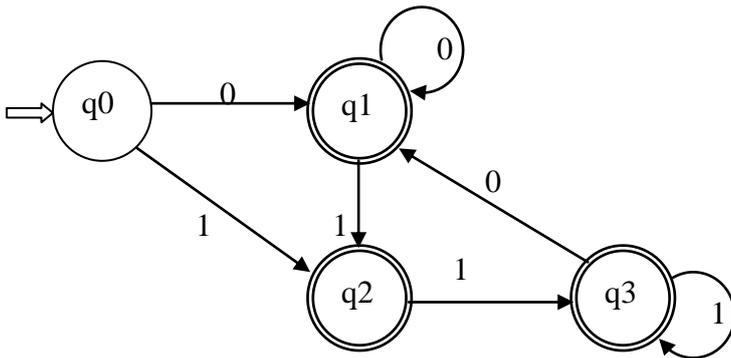
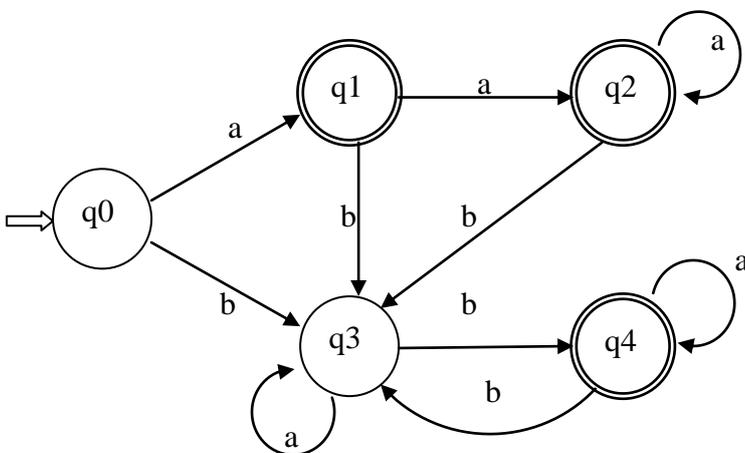


Table de transition de l'automate déterministe équivalent à B :

	a	b
$\langle S0 \rangle = q0$	$\langle S0, S1 \rangle$	$\langle S2 \rangle$
$\langle S0, S1 \rangle = q1$	$\langle S0, S1, S3 \rangle$	$\langle S2 \rangle$
$\langle S0, S1, S3 \rangle = q2$	$\langle S0, S1, S3 \rangle$	$\langle S2 \rangle$
$\langle S2 \rangle = q3$	$\langle S2 \rangle$	$\langle S1, S3 \rangle$
$\langle S1, S3 \rangle = q4$	$\langle S1, S3 \rangle$	$\langle S2 \rangle$

L'automate déterministe équivalent à B :



EXERCICE 3 :

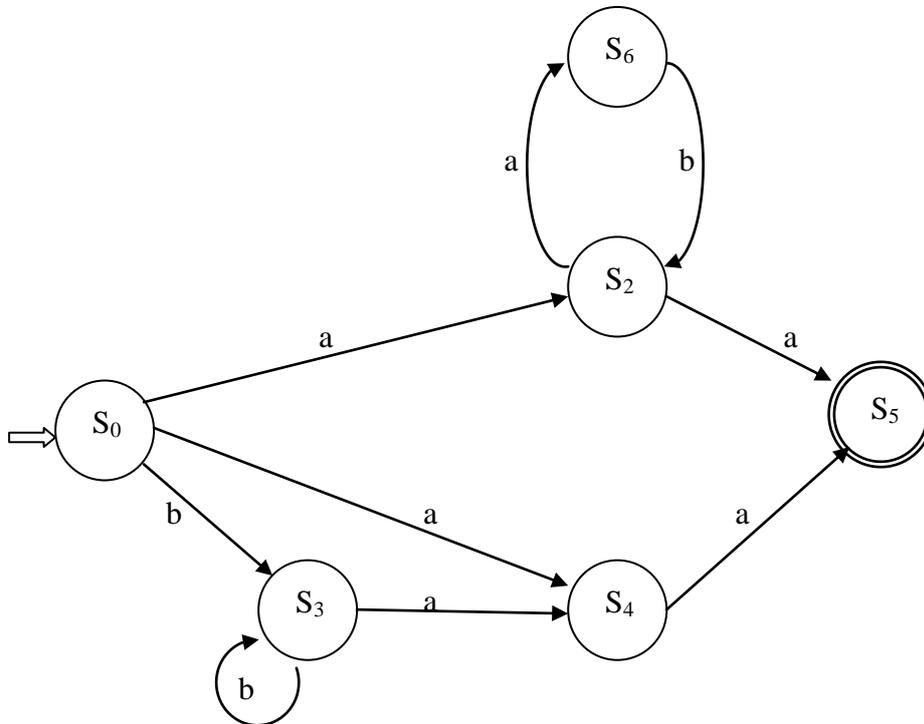
D'abord on transforme l'automate généralisé A en automate partiellement généralisé A_p en décomposant les transitions qui se font sur des mots. C'est le cas, pour A, de la transition (ab, S_2, S_2) ; pour la décomposer on ajoute un nouvel état S_6 et on la remplace par les transitions (a, S_2, S_6) et (b, S_6, S_2) .

Ensuite, à partir de l'automate partiellement généralisé A_p , on construit l'automate simple A_s en éliminant les ϵ -transitions (qu'on appelle aussi les transitions spontanées) suivant les règles :

- si $(\epsilon, S_i, S_j) \in I$ et S_j est un état final alors S_i deviendra lui aussi état final ;

- si (ϵ, S_i, S_j) et $(a, S_j, S_k) \in I$ alors ajouter la transition : (a, S_i, S_k) à I . À la fin des ajouts, éliminer la transition (ϵ, S_i, S_j) de I .

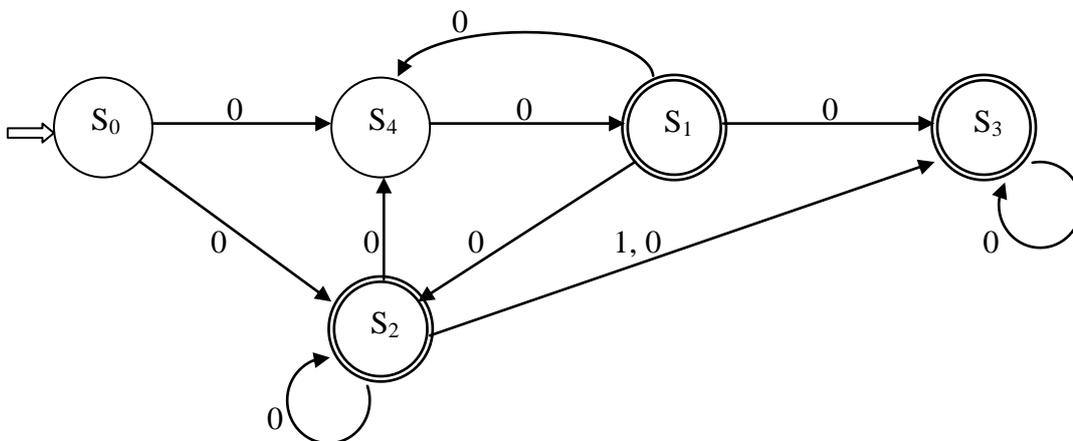
Ainsi, on obtient l'automate A_s suivant :



Remarque : On n'a pas représenté l'état S_1 car il est devenu inaccessible (c'est-à-dire qu'on ne peut pas l'atteindre à partir de l'état initial S_0) ; on l'a donc supprimé de A_s .

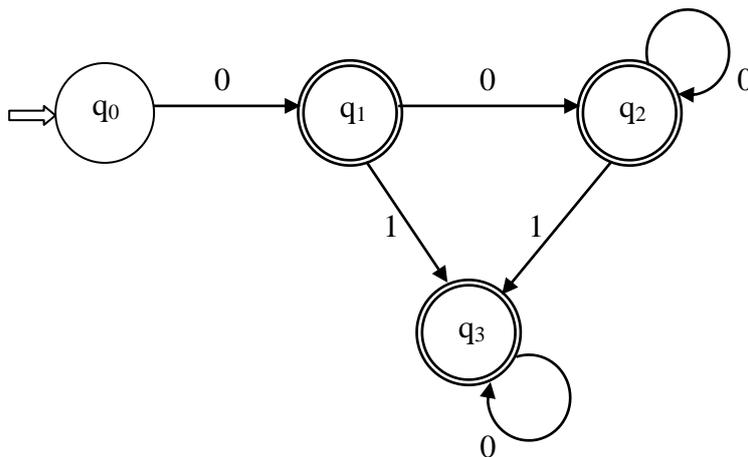
EXERCICE 4 :

1) On construit d'abord l'automate simple A_s équivalent à A , en procédant de la même manière que dans l'exercice 3 précédent. On obtient l'automate simple (non déterministe) A_s suivant :



Ensuite, on transforme A_s en automate déterministe, en procédant de la même manière que dans l'exercice 2.

On obtient l'automate déterministe Ad suivant :



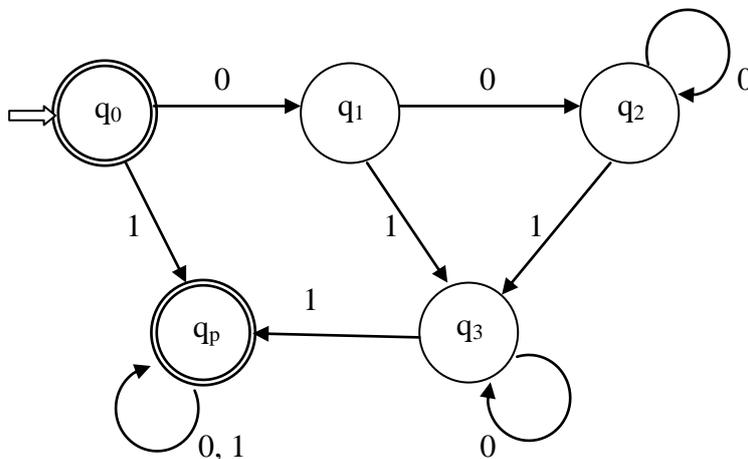
2) Pour obtenir l'automate du complémentaire de $L(A)$ (c'est-à-dire du complémentaire de $L(Ad)$, car A et Ad sont équivalents), on procède comme suit :

I) on prend l'automate simple déterministe Ad , obtenu en 1) ;

II) on complète Ad : on ajoute un état puits – non final – q_p , qu'on raccorde aux états qui ont des transitions manquantes (ici, le raccordement se fait en ajoutant les transitions $(1, q_0, q_p)$ et $(1, q_3, q_p)$ – ne pas oublier les transitions $(0, q_p, q_p)$ et $(1, q_p, q_p)$) ;

III) on inverse les états finaux et non finaux : les états non finaux vont devenir finaux et vice versa.

On obtient ainsi :



EXERCICE 5 :

On va traiter les deux questions 1) et 2) pour les deux grammaires g_1 et g_2 .

Pour construire un automate équivalent à une grammaire régulière, on procède comme suit :

i-) on associe à chaque non terminal de la grammaire, un état de l'automate. L'état associé à l'axiome sera l'état initial de l'automate ;

ii-) à toute règle de production $A \rightarrow \epsilon$ de la grammaire, l'état associé au non terminal A sera final ;

- iii-) à toute règle $A \rightarrow wB$, où w est un mot, on ajoutera la transition (w, E_A, E_B) dans l'automate (E_A et E_B sont les états associés à A et B respectivement) ;
- iv-) à toute règle $A \rightarrow w$, où w est un mot, on ajoutera la transition (w, E_A, F) dans l'automate (E_A est l'état associé à A , F est un nouvel état final).

Ainsi, en appliquant les étapes i-).iv-), avec la grammaire g_1 :

$g_1 = \langle \pi, N_1, S, P_1 \rangle$ où $\pi = \{a, b, c\}$; $N_1 = \{S, A, B\}$;

$P_1 = \{ S \rightarrow abS \mid bA \mid \varepsilon ; A \rightarrow bB \mid B ; B \rightarrow aB \mid b \mid cA \}$;

on obtient l'automate A_1 suivant, qui est équivalent à g_1 :

$A_1 = \langle V, S, F, S_0, I \rangle$ où $V = \{a, b, c\}$; $S = \{S_0, S_1, S_2, S_3\}$; $F = \{S_0, S_3\}$; S_0 état initial

$I = \{ (ab, S_0, S_0) ; (b, S_0, S_1) ; (b, S_1, S_2) ; (\varepsilon, S_1, S_2) ; (a, S_2, S_2) ; (b, S_2, S_3) ; (c, S_2, S_1) \}$;

On transforme ensuite cet automate généralisé A_1 en automate simple :

- automate partiellement généralisé :

$A_{1p} = \langle V, S', F, S_0, I \rangle$ où $V = \{a, b, c\}$; $S' = \{S_0, S_1, S_2, S_3, S_4\}$; $F = \{S_0, S_3\}$; S_0 état initial

$I = \{ (a, S_0, S_4) ; (b, S_4, S_0) ; (b, S_0, S_1) ; (b, S_1, S_2) ; (\varepsilon, S_1, S_2) ; (a, S_2, S_2) ; (b, S_2, S_3) ; (c, S_2, S_1) \}$;

- automate simple :

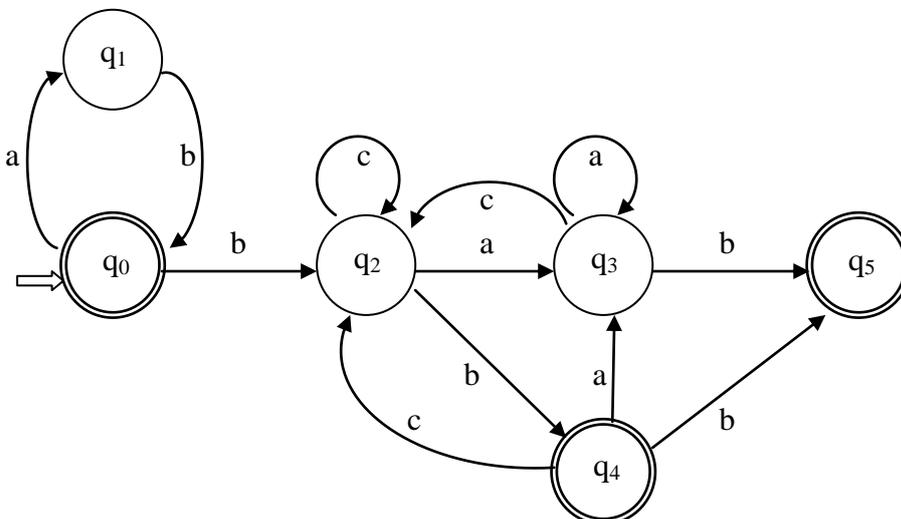
$A_{1s} = \langle V, S', F, S_0, I \rangle$ où $V = \{a, b, c\}$; $S' = \{S_0, S_1, S_2, S_3, S_4\}$; $F = \{S_0, S_3\}$; S_0 état initial

$I = \{ (a, S_0, S_4) ; (b, S_4, S_0) ; (b, S_0, S_1) ; (b, S_1, S_2) ; (a, S_2, S_2) ; (b, S_2, S_3) ; (c, S_2, S_1) ; (a, S_1, S_2) ; (b, S_1, S_3) ; (c, S_1, S_1) \}$;

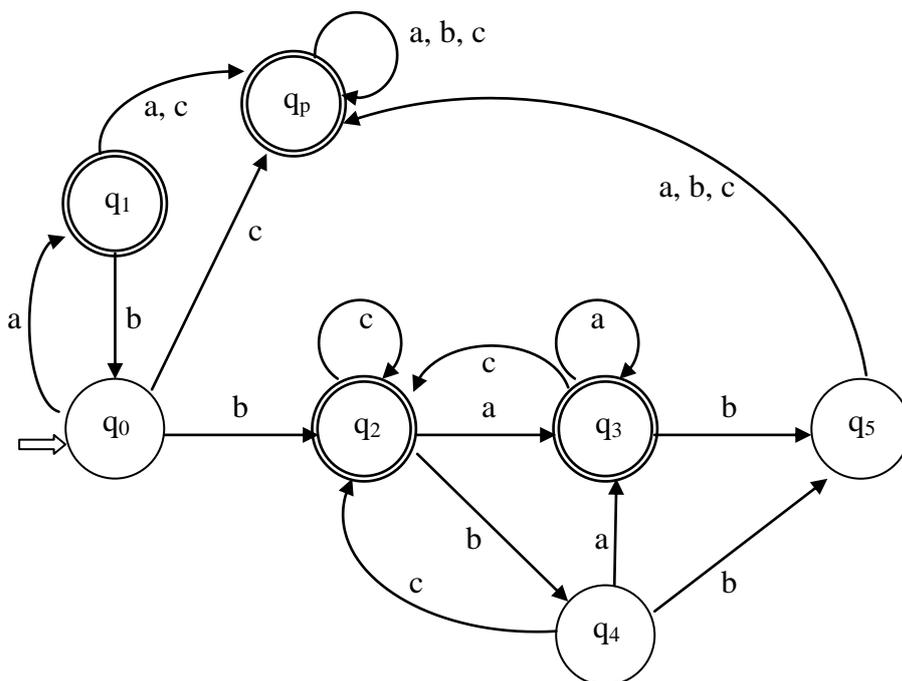
- table de transition de l'automate simple déterministe :

	a	b	c
$\langle S_0 \rangle = q_0$	$\langle S_4 \rangle$	$\langle S_1 \rangle$	/
$\langle S_4 \rangle = q_1$	/	$\langle S_0 \rangle$	/
$\langle S_1 \rangle = q_2$	$\langle S_2 \rangle$	$\langle S_2, S_3 \rangle$	$\langle S_1 \rangle$
$\langle S_2 \rangle = q_3$	$\langle S_2 \rangle$	$\langle S_3 \rangle$	$\langle S_1 \rangle$
$\langle S_2, S_3 \rangle = q_4$	$\langle S_2 \rangle$	$\langle S_3 \rangle$	$\langle S_1 \rangle$
$\langle S_3 \rangle = q_5$	/	/	/

- automate simple déterministe A_{1d} :



On procède comme c'est fait dans l'exercice 4, pour obtenir l'automate du complémentaire, c'est à dire :
 I) on prend l'automate simple déterministe A1d, obtenu précédemment ;
 II) on complète A1d ;
 III) on inverse les états finaux et non finaux. On obtient :



Avec la grammaire g_2 :

$g_2 = \langle \pi, N_2, S, P_2 \rangle$ où $\pi = \{a, b, c\}$; $N_2 = \{S, A\}$;

$P_2 = \{ S \rightarrow aS \mid bS \mid cS \mid abcA ; A \rightarrow aA \mid bA \mid cA \mid \epsilon \}$.

Remarquons d'abord que $L(g_2) =$ ensemble des mots de $\{a, b, c\}^*$ qui **contiennent** la sous-chaîne «**abc**».

Ainsi son complémentaire sera :

l'ensemble des mots de $\{a, b, c\}^*$ qui **ne contiennent pas** la sous-chaîne «**abc**».

En appliquant les étapes *i-iv-*, avec g_2 , on obtient l'automate A_2 suivant, qui est équivalent à g_2 :

$A_2 = \langle V, S, F, S_0, I \rangle$ où $V = \{a, b, c\}$; $S = \{S_0, S_1\}$; $F = \{S_1\}$; S_0 état initial

$I = \{ (a, S_0, S_0) ; (b, S_0, S_0) ; (c, S_0, S_0) ; (abc, S_0, S_1) ; (a, S_1, S_1) ; (b, S_1, S_1) ; (c, S_1, S_1) \}$;

On transforme ensuite cet automate généralisé A_2 en automate simple :

- automate partiellement généralisé :

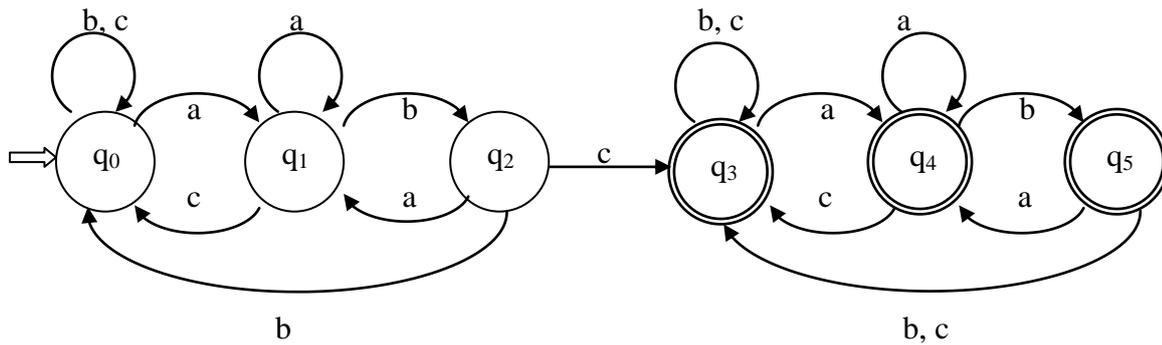
$A_{2p} = \langle V, S, F, S_0, I \rangle$ où $V = \{a, b, c\}$; $S = \{S_0, S_1, S_2, S_3\}$; $F = \{S_1\}$; S_0 état initial

$I = \{ (a, S_0, S_0) ; (b, S_0, S_0) ; (c, S_0, S_0) ; (a, S_0, S_2) ; (b, S_2, S_3) ; (c, S_3, S_1) ; (a, S_1, S_1) ; (b, S_1, S_1) ; (c, S_1, S_1) \}$

Cet automate A_{2p} est simple, car l'automate initial A_2 ne contient pas de transitions spontanées. Mais il n'est pas déterministe. Construisons la table de transitions de l'automate déterministe équivalent :

	a	b	c
$\langle S_0 \rangle = q_0$	$\langle S_0, S_2 \rangle$	$\langle S_0 \rangle$	$\langle S_0 \rangle$
$\langle S_0, S_2 \rangle = q_1$	$\langle S_0, S_2 \rangle$	$\langle S_0, S_3 \rangle$	$\langle S_0 \rangle$
$\langle S_0, S_3 \rangle = q_2$	$\langle S_0, S_2 \rangle$	$\langle S_0 \rangle$	$\langle S_0, S_1 \rangle$
$\langle S_0, S_1 \rangle = q_3$	$\langle S_0, S_2, S_1 \rangle$	$\langle S_0, S_1 \rangle$	$\langle S_0, S_1 \rangle$
$\langle S_0, S_2, S_1 \rangle = q_4$	$\langle S_0, S_2, S_1 \rangle$	$\langle S_0, S_3, S_1 \rangle$	$\langle S_0, S_1 \rangle$
$\langle S_0, S_3, S_1 \rangle = q_5$	$\langle S_0, S_2, S_1 \rangle$	$\langle S_0, S_1 \rangle$	$\langle S_0, S_1 \rangle$

L'automate déterministe A2d est le suivant :

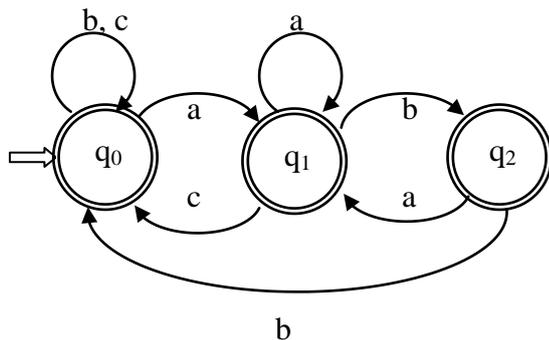


On procède comme précédemment, pour obtenir l'automate du complémentaire, c'est à dire :

- I) on prend l'automate simple déterministe A2d ;
- II) on complète A2d (remarquons que A2d est déjà complet) ;
- III) on inverse les états finaux et non finaux.

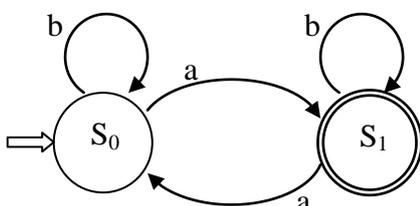
Dans la représentation graphique de l'automate du complémentaire de A2d, on ne représentera pas les états q3, q4 et q5 car ils sont devenus, après l'étape iii), improductifs (il n'y a aucun chemin qui mène de q3, q4 ou q5 vers un des états finaux de l'automate du complémentaire).

On aura donc :

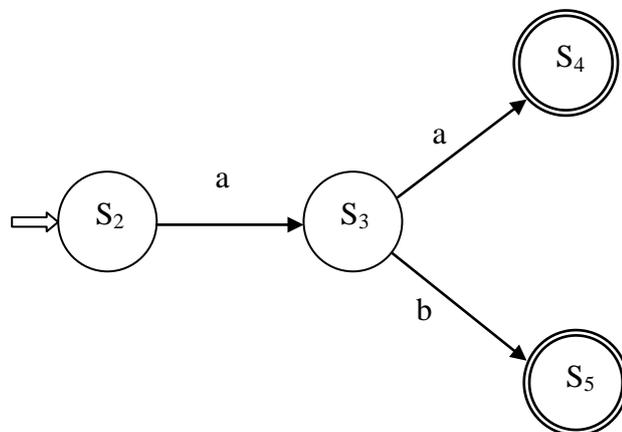


EXERCICE 6 :

1) Automate pour L_1 :



2) Automate pour L_2 :

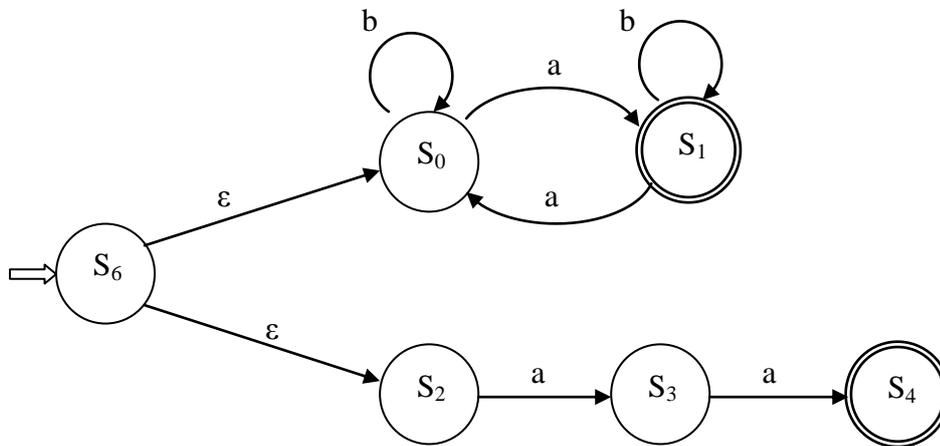


3) Automate pour $L_1 \cup L_2$:

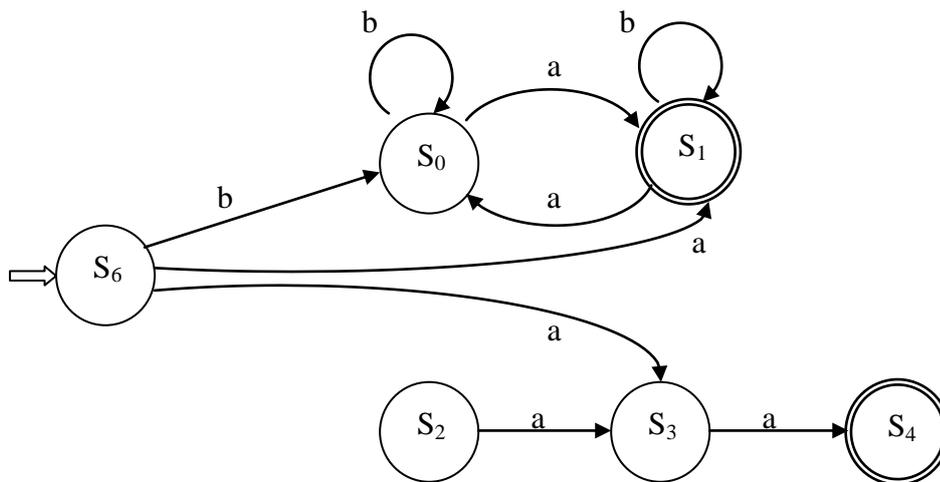
Pour construire l'automate simple de l'union, il suffit d'ajouter un nouvel état initial S_6 , qui sera relié aux états initiaux des automates de L_1 et L_2 par des transitions spontanées et de le rendre ensuite simple.

Mais remarquons d'abord que $ab \in L_1$, ce qui fait que $L_1 \cup L_2 = L_1 \cup \{aa\}$.

- Automate partiellement généralisé de l'union :



- Automate simple de l'union (après élimination des ϵ -règles) :



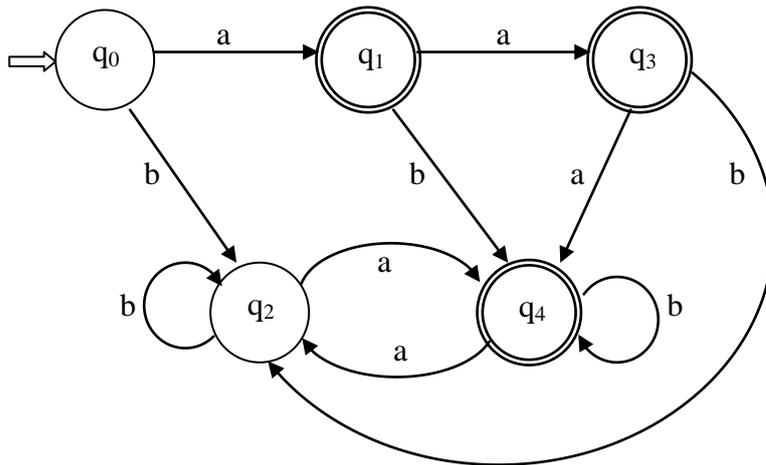
Remarque : l'état S_2 est devenu inaccessible, on peut le supprimer.

4) L'automate de 3) est non-déterministe. Construisons la table de transition de l'automate déterministe :

	a	b
$\langle S_6 \rangle = q_0$	$\langle S_1, S_3 \rangle$	$\langle S_0 \rangle$
$\langle S_1, S_3 \rangle = q_1$	$\langle S_0, S_4 \rangle$	$\langle S_1 \rangle$
$\langle S_0 \rangle = q_2$	$\langle S_1 \rangle$	$\langle S_0 \rangle$
$\langle S_0, S_4 \rangle = q_3$	$\langle S_1 \rangle$	$\langle S_0 \rangle$
$\langle S_1 \rangle = q_4$	$\langle S_0 \rangle$	$\langle S_1 \rangle$

Les états soulignés sont des états finaux.

Automate déterministe de l'union :



EXERCICE 7 :

1) Automate généralisé équivalent à g :

$Ag = \langle V, S, F, S_0, I \rangle$ où $V = \{a, b, c\}$; $S = \{S_0, S_1, S_2, S_3\}$; $F = \{S_2, S_3\}$; S_0 état initial
 $I = \{ (aa, S_0, S_1) ; (ab, S_0, S_2) ; (a, S_1, S_2) ; (a, S_1, S_3) ; (b, S_2, S_2) \}$.

À partir de Ag, on construit l'automate généralisé qui accepte $L(g)^+$: on ajoute des ϵ -transitions des états finaux de Ag (S_2 et S_3) vers l'état initial de celui-ci.

Puis, on construit l'automate partiellement généralisé :

$Apg = \langle V, S, F, S_0, I \rangle$ où $V = \{a, b, c\}$; $S = \{S_0, S_1, S_2, S_3, S_4, S_5\}$; $F = \{S_2, S_3\}$; S_0 état initial
 $I = \{ (a, S_0, S_4) ; (a, S_4, S_1) ; (a, S_0, S_5) ; (b, S_5, S_2) ; (a, S_1, S_2) ; (a, S_1, S_3) ; (b, S_2, S_2) ;$
 $(\epsilon, S_2, S_0) ; (\epsilon, S_3, S_0) \}$.

On procède ensuite à la construction de l'automate simple équivalent :

$As = \langle V, S, F, S_0, I \rangle$ où $V = \{a, b, c\}$; $S = \{S_0, S_1, S_2, S_3, S_4, S_5\}$; $F = \{S_2, S_3\}$; S_0 état initial
 $I = \{ (a, S_0, S_4) ; (a, S_4, S_1) ; (a, S_0, S_5) ; (b, S_5, S_2) ; (a, S_1, S_2) ; (a, S_1, S_3) ; (b, S_2, S_2) ;$
 $(a, S_2, S_4) ; (a, S_2, S_5) ; (a, S_3, S_4) ; (a, S_3, S_5) \}$.

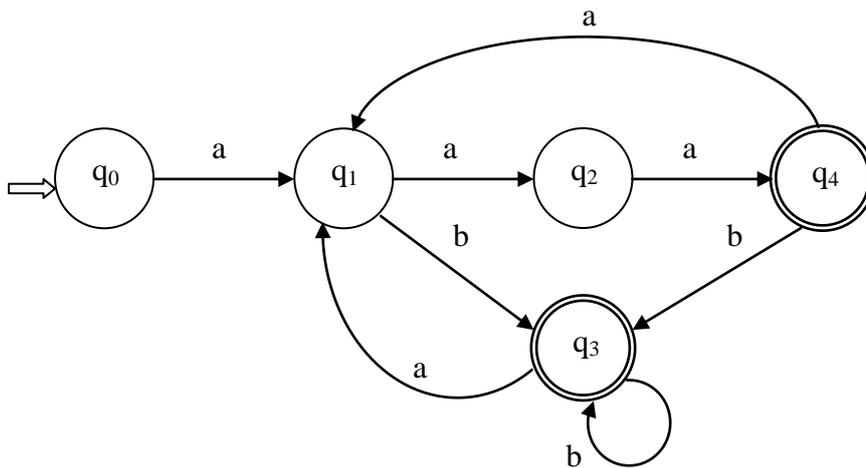
Puis on fait la détermination :

	a	b
$\langle S_0 \rangle = q_0$	$\langle S_4, S_5 \rangle$	/
$\langle S_4, S_5 \rangle = q_1$	$\langle S_1 \rangle$	$\langle S_2 \rangle$
$\langle S_1 \rangle = q_2$	$\langle S_2, S_3 \rangle$	/
$\langle S_2 \rangle = q_3$	$\langle S_4, S_5 \rangle$	$\langle S_2 \rangle$
$\langle S_2, S_3 \rangle = q_4$	$\langle S_4, S_5 \rangle$	$\langle S_2 \rangle$

L'automate déterministe :

$Ad = \langle V, S, F, S_0, I \rangle$ où $V = \{a, b, c\}$; $S = \{q_0, q_1, q_2, q_3, q_4\}$; $F = \{q_3, q_4\}$; q_0 état initial
 $I = \{ (a, q_0, q_1) ; (a, q_1, q_2) ; (a, q_2, q_4) ; (b, q_1, q_3) ; (a, q_3, q_1) ; (a, q_4, q_1) ; (b, q_3, q_3) ;$
 $(b, q_4, q_3) \}$.

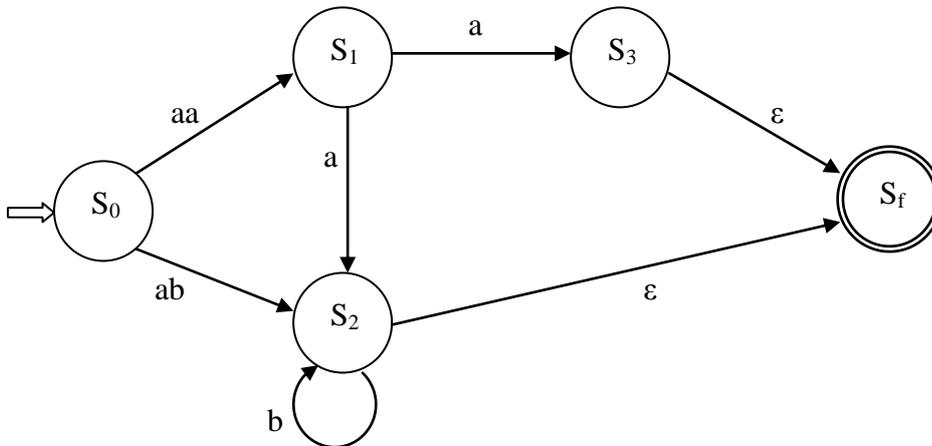
Représentation graphique de Ad :



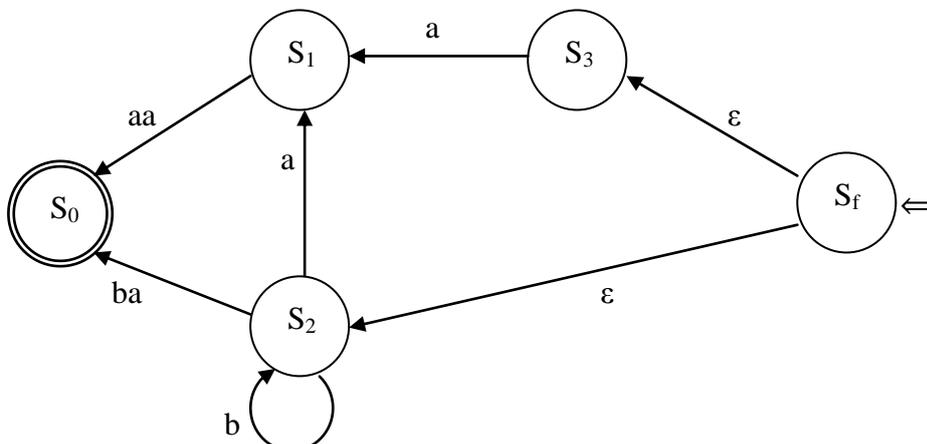
2) Pour trouver un automate (pas forcément simple et déterministe) Ar qui accepte le reflet miroir de $L(g)$ (c'est-à-dire : $L(g)^R$) :

- on prend un automate acceptant $L(g)$, par exemple Ag (de la question précédente) ;
- on le rend avec un seul état final : on ajoute le nouvel état final S_f et on ajoute aussi des ϵ -transitions des anciens états finaux de Ag vers S_f , typiquement les transitions : (ϵ, S_2, S_f) et (ϵ, S_3, S_f) ;
- chaque transition (w, S_i, S_j) de Ag devient une transition (w^R, S_j, S_i) de Ar (cela revient à inverser le sens des flèches, ainsi que les étiquettes, de Ag) ;
- l'état initial de Ar est S_f , l'état final sera S_0 (l'état initial de Ag).

Ag avec un seul état final :



Ar :



EXERCICE 8 :

1) Grammaire régulière à droite g_1 pour L_1 :

Terminaux : a, b.

Non-terminaux : S, A

axiome : S

Règles de production : $S \rightarrow bS \mid aA \mid \varepsilon$; $A \rightarrow aA \mid \varepsilon$

Grammaire régulière à droite g_2 pour L_2 :

Terminaux : a, b.

Non-terminaux : S, A

axiome : S

Règles de production : $S \rightarrow aA$; $A \rightarrow baA \mid b$

2) Automate A_1 pour $L(g_1)$:

$A_1 = \langle V, S, F, S_0, I \rangle$ où $V = \{a, b\}$; $S = \{S_0, S_1\}$; $F = \{S_0, S_1\}$; S_0 état initial

$I = \{ (b, S_0, S_0) ; (a, S_0, S_1) ; (a, S_1, S_1) \}$.

Automate A_2 pour $L(g_2)$:

$A_2 = \langle V, S, F, S_2, I \rangle$ où $V = \{a, b\}$; $S = \{S_2, S_3, S_4\}$; $F = \{S_4\}$; S_2 état initial

$I = \{ (a, S_2, S_3) ; (ba, S_3, S_3) ; (b, S_3, S_4) \}$.

3) Pour construire l'automate de $L_1.L_2^*$, on va d'abord construire l'automate de L_2^* . Celui-ci s'obtient à partir de A_2 en :

- faisant de l'état initial S_2 un état final (car il n'y a pas de transitions entrantes sur S_2) ;
- ajoutant des ε -transitions des états finaux de A_2 vers l'état initial de A_2 .

Soit A_{2it} l'automate de l'itération obtenu :

$A_{2it} = \langle V, S, F, S_2, I \rangle$ où $V = \{a, b\}$; $S = \{S_2, S_3, S_4\}$; $F = \{S_4, S_2\}$; S_2 état initial

$I = \{ (a, S_2, S_3) ; (ba, S_3, S_3) ; (b, S_3, S_4) ; (\varepsilon, S_4, S_2) \}$.

Pour construire l'automate A_c de la concaténation, on procède comme suit :

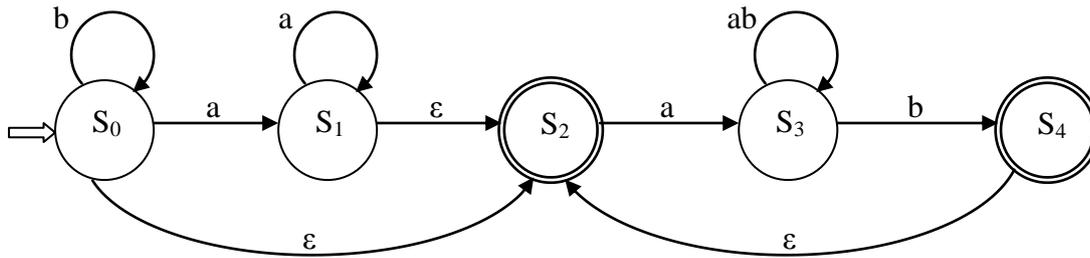
- on prend les deux automates A_1 et A_{2it} ;
- l'état initial de A_c sera l'état initial de A_1 (c'est-à-dire S_0) ;
- les états finaux de A_c seront ceux de A_{2it} (c'est-à-dire S_4 et S_2) ;
- on ajoute des ε -transitions des états finaux de A_1 (S_0, S_1) vers l'état initial de A_{2it} (S_2).

On obtient donc :

$A_c = \langle V, S, F, S_0, I \rangle$ où $V = \{a, b\}$; $S = \{S_0, S_1, S_2, S_3, S_4\}$; $F = \{S_4, S_2\}$; S_0 état initial

$I = \{ (b, S_0, S_0) ; (a, S_0, S_1) ; (a, S_1, S_1) ; (a, S_2, S_3) ; (ba, S_3, S_3) ; (b, S_3, S_4) ; (\varepsilon, S_4, S_2) ; (\varepsilon, S_0, S_2) ; (\varepsilon, S_1, S_2) \}$.

Graphiquement :



EXERCICE 9 :

Il faut montrer que $Mel(L,L')$ est régulier sachant que L et L' sont réguliers.

Soient $A = \langle V_1, S_1, F_1, S_{01}, I_1 \rangle$ l'automate qui accepte L , et $B = \langle V_2, S_2, F_2, S_{02}, I_2 \rangle$ l'automate qui accepte L' .

L'automate qui accepte $Mel(L,L')$ est : $P = \langle V, S, F, S_0, I \rangle$ où :

$V = V_1 \cup V_2$, $S = S_1 \times S_2$, $F = F_1 \times F_2$, $S_0 = (S_{01}, S_{02})$ et I est tel que :

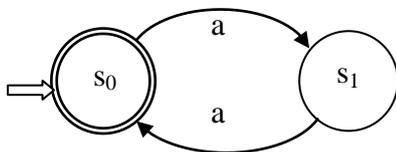
$(a, (S_{i1}, S_{j2}), (S_{k1}, S_{l2})) \in I$ si et seulement si :

$(a, S_{i1}, S_{k1}) \in I_1$ et $S_{j2} = S_{l2}$; ou bien $(a, S_{j2}, S_{l2}) \in I_2$ et $S_{i1} = S_{k1}$.

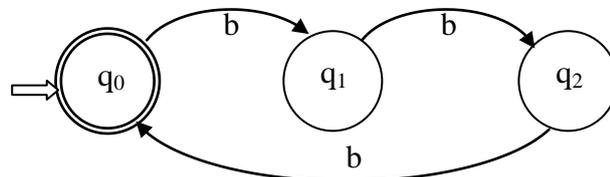
Prenons maintenant l'exemple : $L = \{ (aa)^n / n \geq 0 \}$; $L' = \{ (bbb)^n / n \geq 0 \}$.

Les automates de L et L' sont :

A :



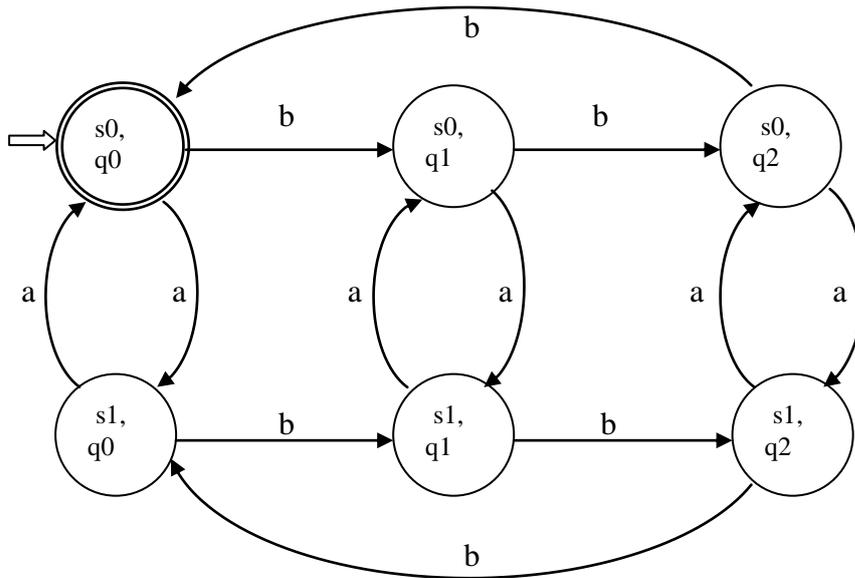
B :



La table de transition de P :

	a	b
(s0,q0)	(s1,q0)	(s0,q1)
(s1,q0)	(s0,q0)	(s1,q1)
(s0,q1)	(s1,q1)	(s0,q2)
(s1,q1)	(s0,q1)	(s1,q2)
(s0,q2)	(s1,q2)	(s0,q0)
(s1,q2)	(s0,q2)	(s1,q0)

Le graphe de P :



EXERCICE 10 :

On donnera les grandes lignes des algorithmes de reconnaissance. Le codage en Pascal est laissé aux soins de l'étudiant.

1) Dans le cas où l'automate est déterministe, on le représentera par un tableau à deux dimensions :

Var Trans : Tableau['a'..'z', 0..nmax] de entier ;

où nmax est le nombre maximum d'états pour l'automate.

Ici on traitera des automates qui travaillent sur les lettres 'a'..'z'. Dans le cas général, pour travailler sur des caractères quelconques la 1ère dimension de Trans sera un type énuméré, et il faudra une table de correspondance entre ce type énuméré et les caractères en question.

On conviendra que si $\text{Trans}[c,s] = -1$ alors il n'y a pas de transition à partir de l'état s avec la lettre c.

On utilisera aussi une variable ensemble F comprenant les états finaux de l'automate.

L'algorithme de reconnaissance, qui aura à tester si une chaîne de caractères C est acceptée par l'automate, parcourra la chaîne C avec l'indice i, de gauche à droite, tout en effectuant des transitions entre les états (pour cela on utilisera la variable s, qui sera initialisée à 0 : l'état initial).

Si on arrive à la fin du parcours de la chaîne C, et on se trouve dans un état final (c'est-à-dire $s \in F$) alors on dira que la chaîne C est acceptée par l'automate ; sinon elle n'est pas acceptée.

Algorithme test_det

Entrée : une chaîne de caractères C

Sortie : 'oui' si C est acceptée par l'automate, 'non' sinon

Début

s := 0 ; (* on suppose que l'état initial est 0 *)

bool := VRAI ;

i := 1 ;

Tant que (i <= longueur(C)) et bool faire

 car := C[i] ;

 i := i+1 ;

 si non (car dans ['a'..'z']) alors

 bool := FAUX

 sinon

 s := Trans[car,s] ;

 si s=-1 alors bool := FAUX finsi ;

 finsi ;

FinTantque

si bool et (s dans F) alors

 écrire ('la chaîne est acceptée')

sinon

 écrire ('la chaîne n'est pas acceptée')

finsi

Fin.

2) Dans le cas où l'automate est non déterministe, alors tableau qui le représentera sera :

 Var Trans : Tableau['a'..'z',0..nmax] de ensemble de 0..nmax ;

Dans ce cas, il n'y a pas de transition à partir de l'état s avec la lettre c lorsque Trans[c,s]={}.

La variable ensemble F comprendra les états finaux de l'automate.

On utilisera aussi une variable tableau S d'ensembles d'états.

Algorithme test_non-det

Entrée : une chaîne de caractères C dans {'a'..'z'}*

Sortie : 'oui' si C est acceptée par l'automate, 'non' sinon

Début

 n := longueur(C) ;

 Pour i := 0 à n faire

 si i=0 alors S[i] := {0} (* on affecte à la variable ensemble S[0] l'état initial *)

 sinon S[i] := $\cup_{e \in S[i-1]} \text{Trans}(C[i],e)$

 finsi ;

 finPour

 si (S[n] \cap F \neq {}) alors écrire ('la chaîne est acceptée')

 sinon écrire ('la chaîne n'est pas acceptée') finsi ;

Fin

EXERCICE 11 :

On va représenter un automate d'états finis simple déterministe par un algorithme.

Dans ce cas on associera des étiquettes (*label*), aux états de l'automate, dans l'algorithme.

Considérons l'exemple l'automate de e) de l'exercice 1 de cette série.

Algorithme Exo1e

Entrée : une chaîne de caractères C dans $\{ '0', '1' \}^*$

Sortie : 'oui' si C est acceptée par l'automate, 'non' sinon

Procédure lex ;

Début

 car := C[k] ;

 k := k+1 ;

Fin ;

Début (* de l'algorithme *)

 n := longueur(C) ; k :=1 ; bool := FAUX

 r0 :

 si k>n alors

 bool := VRAI ; aller_à stop

 sinon

 lex ;

 si car='0' alors aller_à r0

 sinon aller_à r1 finsi

 finsi ;

 r1 :

 si k>n alors

 bool := FAUX ; aller_à stop

 sinon

 lex ;

 si car='0' alors aller_à r2

 sinon aller_à r0 finsi

 finsi ;

 r2 :

 si k>n alors

 bool := FAUX ; aller_à stop

 sinon

 lex ;

 si car='0' alors aller_à r1

 sinon aller_à r2 finsi

 finsi ;

 stop :

 si bool alors écrire ('la chaîne est acceptée') sinon écrire ('la chaîne n''est pas acceptée') finsi

Fin.

----- Fin du corrigé de la série 2 de ThL -----