Automate asynchrone

Souvenons-nous que jusqu'à ce point nous avons évité **le mot vide** comme étiquette de transition**.**

Un automate fini dans lequel certaines flèches sont étiquetées par le mot vide (« ***ε*-*transitions*** ») s'appelle **automate asynchrone**. L'ensemble des flèches vérifie donc
$\boldsymbol{E \subset Q \times (A \cup \varepsilon) \times Q}$.

Proposition : Pour tout automate asynchrone, il existe un automate fini ordinaire (c.à.d. sans ε-transitions) qui reconnaît le même langage.
Comme tout automate non déterministe est équivalent à un automate déterministe, il en suit que **<u>tout automate asynchrone est équivalent à un automate déterministe</u>**.

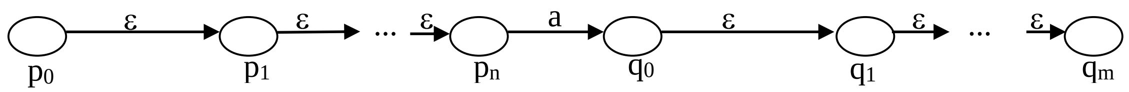
Il existe un algorithme de suppressions de ε-transitions.

Pour un automate asynchrone $\boldsymbol{C = (Q, I, T, E)}$ avec $\boldsymbol{E \subset Q \times (A \cup \varepsilon) \times Q}$ nous allons construire un automate $\boldsymbol{B=(Q, I, T, F)}$ qui ne diffère de $\boldsymbol{C}$ que par l'ensemble de ses flèches ($\boldsymbol{F}$) et ce, suivant une règle bien déterminée :

par définition on a $\boldsymbol{(p.a.q) \in F}$ s'il existe un chemin

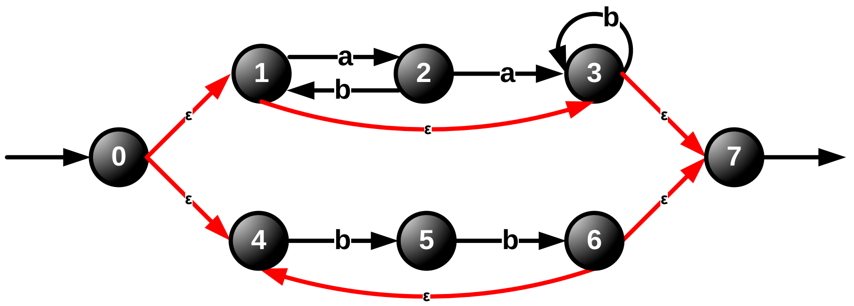
$$c : \quad p_0 \xrightarrow{\varepsilon} p_1 \xrightarrow{\varepsilon} \dots \xrightarrow{\varepsilon} p_n \xrightarrow{a} q_0 \xrightarrow{\varepsilon} q_1 \xrightarrow{\varepsilon} \dots \xrightarrow{\varepsilon} q_m$$

et $\quad p_0 = p,\ q_m = q$



Le nombre de ε-transitions à droite ou à gauche de la transition $\boldsymbol{p_n \xrightarrow{a} q_0}$ peut être nul (dans ce cas $\boldsymbol{p_n = p_0}$ ou $\boldsymbol{q_0 = q_m}$).
(Clarification : on identifie donc tout **chemin** de l'automate ***Ci*** n'impliquant qu'une seule transition étiquetée par un caractère autre que ε, à une transition de l'automate B marquée par ce caractère).

<u>Ex</u>. Prenons un automate asynchrone :



(A1)

1a) Elimination des ε-transitions

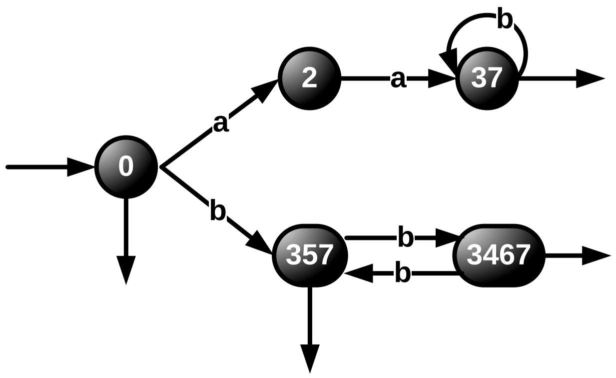
	état	a	b
ES	0	2	3,5,7
S	1	2	3,7
	2	3,7	--
S	3	--	3,7
	4	--	5
	5	--	4,6,7
S	6	--	5
S	7	--	--

	état	a	b
ES	0	2	357
	2	37	--
S	357	--	3467
S	37	--	37
S	3467	--	357

0ε1ε3ε7 signifie que 0 est une sortie 0ε1a2 → 0a2 0ε1ε3b3→ 0b3 0ε1ε3b3ε7→ 0b7 0ε4b5→ 0b5	1ε3ε7 signifie que 1 est une sortie 1a2 reste tel quel 1ε3b3→ 1b3 1ε3b3ε7→ 1b7	2a3 reste tel quel 2a3ε7 →2a7	3ε7 signifie que 3 est une sortie 3b3 reste tel quel 3b3ε7 →3b7
4b5 reste tel quel	5b6 reste tel quel 5b6ε4→ 5b4 5b6ε7→5b7	6ε7 signifie que 6 est une sortie 6ε4b5→ 6b5	7 : pas de transitions

On obtient une TT :

1b) Déterminisation (sans compléter):



	état	a	b
ES	0’	2’	3’5’
	2’	3’	--
S	3’5’	--	3’6’
S	3’	--	3’
S	3’6’	--	3’5’

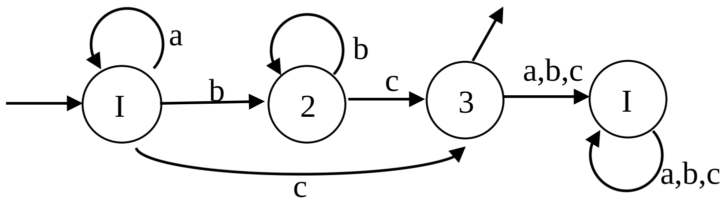
2) Déterminisation en un seul pas :
ε-clôtures:
de 0: **0**1347
de 2 : 2
de 3: **3**7
de 5: 5
de 6 : 4**6**7

	état	a	b
ES	0	2	357
	2	37	--
S	357	--	3467
S	37	--	37
S	3467	--	357

Notons 0’=01347, 2’=2, 3’=37, 5’=5 , 6’=467 (notation utile mais pas obligatoire).

Dans les termes de ces ε-clôtures qui deviennent des états composés, nous pouvons déterminiser tout de suite :

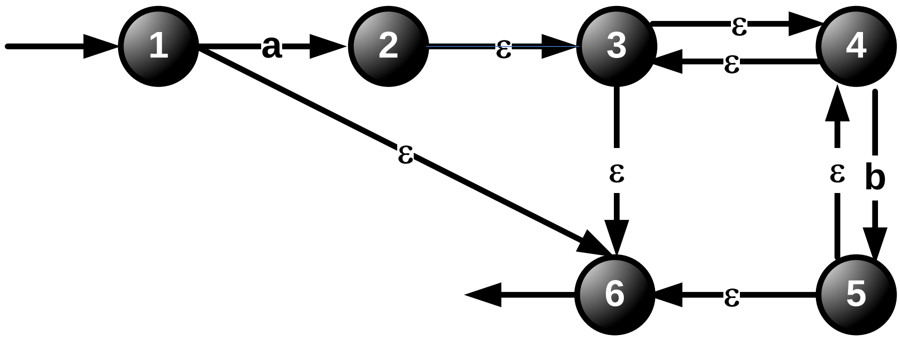
Ce qui est exactement la même chose que la TT à droite



ce qui est le même automate que le résultat de compléter l'automate (A5) ! (en minimisant l'automate (A5), la seule chose qu'il reste à faire c'est de le compléter, autrement il est déjà minimal).

Exemple plus compliqué

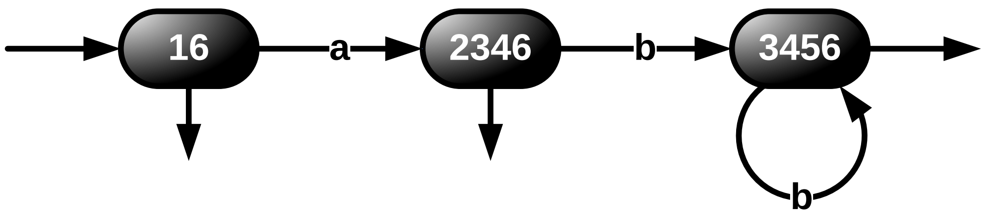
L'exemple qu'on vient de considérer n'est pas assez général. Prenons un exemple plus complexe :



Déterminisons cet automate en considérant tous les états qu'on peut atteindre en lisant un caractère comme un état composé ; l'état initial c'est l'état où on peut arriver en ne lisant que le mot vide, donc ici c'est l'état composé (1,6). En lisant ***a*** à partir de l'état 1, on arrive en 2, mais en lisant ***aε=a***, on arrive en 3, en lisant ***aεε=a***, on arrive en 4 et en 6, donc on a l'état composé (2,3,4,6) etc.

état	**a**	**b**
(1,6)	(**2**,3,4,6)	--
		(A5)
(2,3,4,6)	--	(3,4,**5**,6)
(3,4,5,6)	--	(3,4,**5**,6)

Tous les états composés contenant l'état terminal 6 sont terminaux, donc ici tous les trois états sont terminaux.
On obtient :



Il existent d'autres techniques équivalentes pour éliminer les ε-transitions[1].

[1] Par exemple : Patrice Séébold, Théorie des automates, Vuibert Informatique 1999, pp. 165-167.