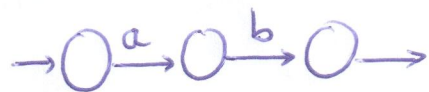
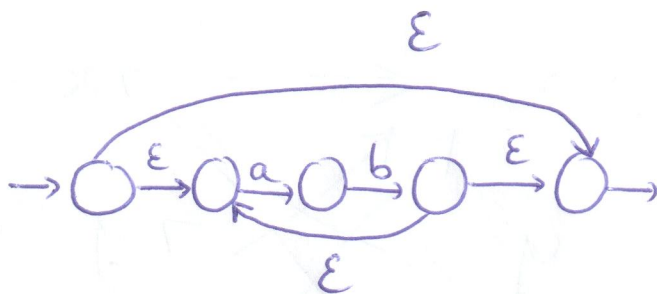


①

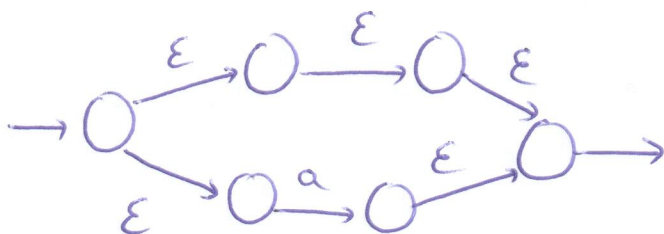
automate pour l'expression ab :



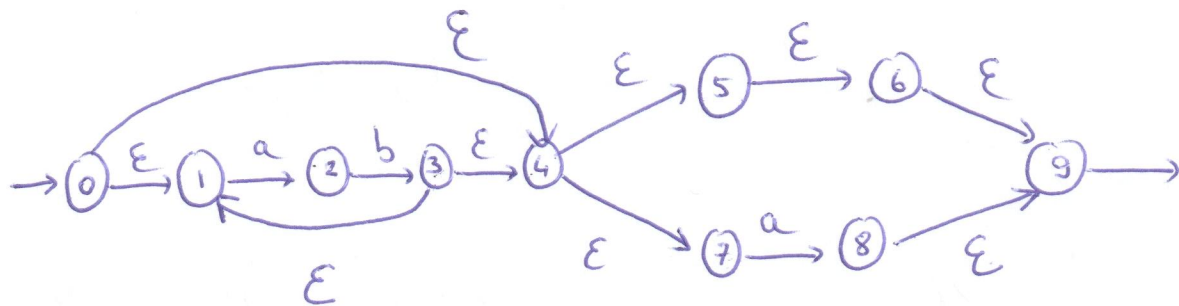
automate pour l'expression $(ab)^*$



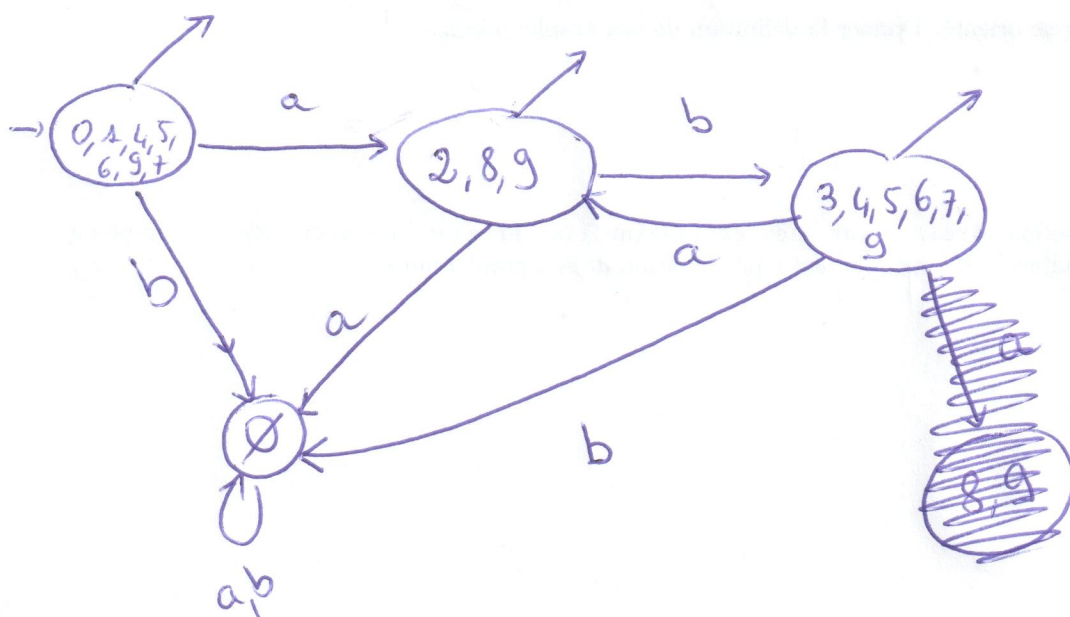
automate pour l'expression $(\epsilon|a)$



On en déduit, en appliquant l'opération de concaténation sur les deux derniers automates obtenus:



② il est plus simple de prendre $K(\{0\})$ comme état initial plutôt que $\{0\}$



③

linéarisation:

$$(a_1 b_2)^* (\varepsilon | a) ?$$

⚠ l'expression contient le symbole ε ce qui n'est pas valide avec la méthode il faut donc commencer par réécrire ε en une expression équivalente de la forme $e' | \varepsilon$:

$$\begin{aligned} \mathcal{L}(e) &= \mathcal{L}((ab)^* | (ab)^* a) = \mathcal{L}(\varepsilon | ab(ab)^* | (ab)^* a) \\ &= \mathcal{L}(\varepsilon | (ab)^* (ab|a)) \end{aligned}$$

Ainsi on applique l'algo à $e' = (ab)^* (ab|a)$ puis on ajoutera ε à la main comme mot accepté par l'automate.

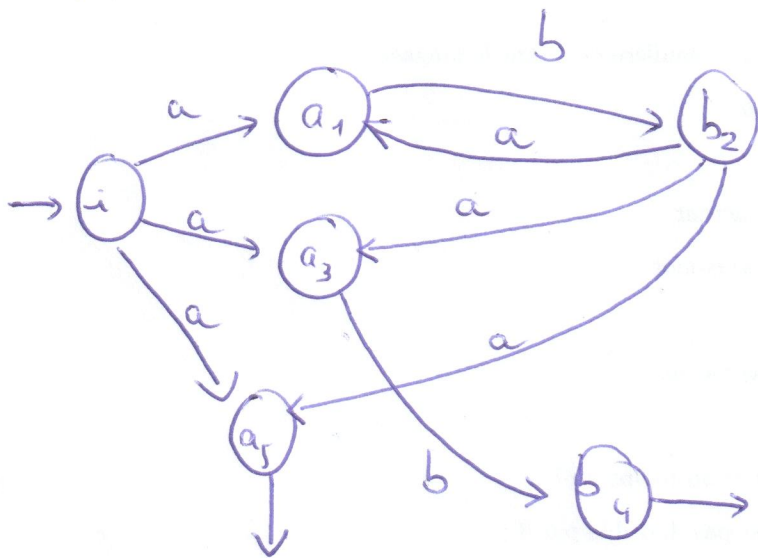
linéarisation de e' : $(a_1 b_2)^* (a_3 b_4 | a_5)$

$$P = \{a_1, a_3, a_5\}$$

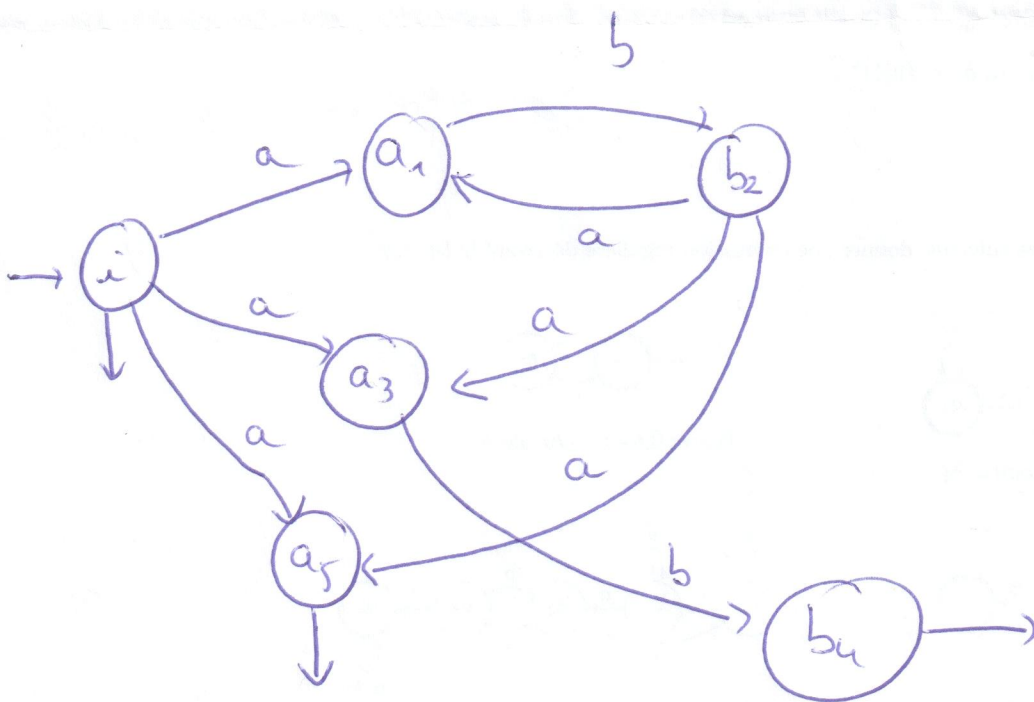
$$S = \{b_4, a_5\}$$

$$F = \{a_1 b_2, b_2 a_1, b_2 a_3, b_2 a_5, a_3 b_4\}$$

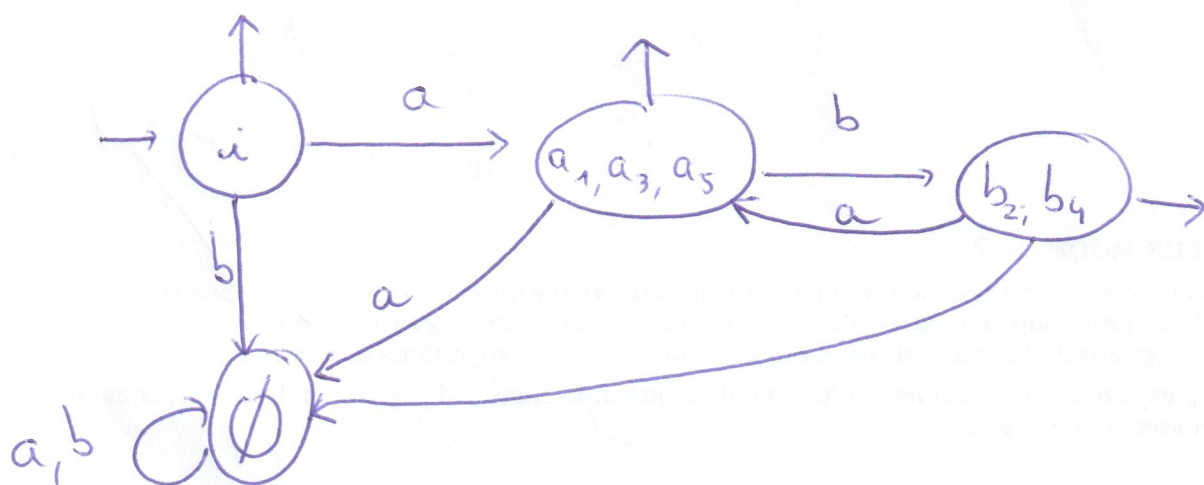
On obtient l'automate suivant non déterministe:
pour e'



on rajoute i comme état final pour obtenir:



Déterminisation



On retrouve exactement le même automate

- ④ les deux méthodes permettent d'obtenir un AFND en ~~le~~ espace linéaire, Thompson se fait en temps linéaire et BS en temps quadratique à cause des calculs de P, S et F.
- ⑤ Thompson est plus simple techniquement à mettre en oeuvre et a une complexité en tps plus réduite mais le résultat contient beaucoup d' ϵ -transitions donc la déterminisation de l'automate obtenu, si souhaitée est plus coûteuse à cause des calculs de clôtures par ϵ -transitions.