5.12.3 Clôture des déterministes par complémentaire

Le but de cette section est de démontrer que :

Theorem 25 *le complémentaire d'un langage algébrique déterministe est déterministe.*

Soit L un langage algébrique déterministe. Il existe un automate à pile déterministe $\mathcal{A}$ tel que $L = L(\mathcal{A})$.

L'idée générale est d'échanger les états finaux et les états non-finaux de l'automate $\mathcal{A}$:

$\mathcal{A}$ $\overline{\mathcal{A}}$

En général, $L(\overline{\mathcal{A}}) \neq \overline{L(\mathcal{A})}$. Cette idée fonctionne presque modulo quelques transformations de l'automate. Il y a deux problèmes :

Pb1. L'automate peut ne pas lire tout le mot. Dans ce cas, le mot n'est pas accepté par l'automate $\mathcal{A}$. Mais il ne l'est pas non plus par $\overline{\mathcal{A}}$.

Pb2. Quand bien même l'automate lit tout le mot, l'exécution peut se poursuivre avec un nombre arbitraire de transitions qui ne lisent plus rien sur le mot d'entrée, c'est à dire des transitions étiquettées par $\overset{\epsilon}{\underset{\text{push}(**)}{\text{pop}(*)}}$. Supposons que l'exécution est acceptante et que l'on atteigne un état final 

. On peut poursuivre l'exécution avec des transitions de la forme $\overset{\epsilon}{\underset{\text{push}(**)}{\text{pop}(*)}}$ et atteindre un état non final 

. Mais alors cette nouvelle exécution est acceptante dans $\overline{\mathcal{A}}$ car q' est final dans $\overline{\mathcal{A}}$. Ce mot est donc aussi accepté par $\overline{\mathcal{A}}$...

Avant de démontrer le théorème 25, on va transformer l'automate initial $\mathcal{A}$ pour se débarasser de Pb1.

Lemma 12 *On peut transformer l'automate à pile déterministe $\mathcal{A}$ en un automate à pile déterministe $\mathcal{B}$ équivalent qui lit toujours tout son mot d'entrée. De plus en prenant une ϵ-transition depuis une configuration acceptante, on reste dans une configuration acceptante.*

PROOF.
Il y a deux raisons qui empêchent l'automate $\mathcal{A}$ de lire tout son mot d'entrée.

1. La première est que l'on peut arriver dans une configuration (q, ϵ) où la pile est vide et toutes les transitions demandent à dépiler un symbole sur la pile. L'automate est bloqué.

2. Une autre raison est que l'exécution peut être infinie : on prend *ad vitam æternam* des ϵ-transitions ϵ / pop($*$) / push($**$) qui ne lisent rien sur le mot d'entrée sans finir la lecture du mot.

Avant tout, on va régler le problème suivant :
0. On veut qu'en prenant une ϵ-transition depuis un configuration acceptante, on reste dans une configuration acceptante.

Pour cela, on modifie l'automate $\mathcal{A}$ comme montré dans la figure 5.1. D'abord, on copie tous les états de $\mathcal{A}$: on obtient la partie copiée (en bleu). Dans la partie originale, on ne conserve que les transitions qui partent d'un état non final et/ou qui lisent une lettre. Les ϵ-transitions depuis un état final sont redirigés vers la copie de l'état destination. Dans la partie copiée (en bleu), tous les états sont déclarés comme finaux et on a copié uniquement les ϵ-transitions. Une transition qui devrait lire depuis un état copié est redirigé vers l'état destination dans la partie originale. On obtient l'automate $\mathcal{A}'$.

TODO: définir formellement

1. Pour résoudre le problème 1., on va utiliser un symbole de "fond de pile" (\$). De l'automate $\mathcal{A}'$, on construit l'automate $\mathcal{A}''$ comme montré dans la figure 5.2. On ajoute un nouvel état i', initial et on ajoute une transition de i' à l'(ancien) état initial i où on empile le symbole de fond de pile (\$). Ensuite, on ajoute deux états $refuse$ et *accepte*. *accepte* est final. On relie un état de $\mathcal{A}'$ non final à $refuse$ en dépilant (\$) (sans lire le mot) et un état final de $\mathcal{A}'$ à *accepte* en dépilant (\$) (sans lire le mot). Dans l'état $refuse$, on peut continuer à finir la lecture du mot. Dans *accepte*, si on lit une lettre supplémentaire, on va dans l'état $refuse$.

2. Maintenant, on va s'occuper des boucles infinis où on ne lit rien sur le mot d'entrée dans $\mathcal{A}''$. On va construire $\mathcal{A}'''$ où cela n'arrive plus. Considérons un état q qui n'est ni *accepte* ni $refuse$. Dans cet état, la pile n'est jamais vide. On dit (q, x) où $q \in Q$ et $x \in \Gamma$ (Γ contient (\$) désormais) est une situation à boucle si quand on est dans une configuration où on est dans l'état q avec x sur le sommet de pile, on ne dépile jamais plus loin que x et on ne lit plus jamais rien sur le mot d'entrée :

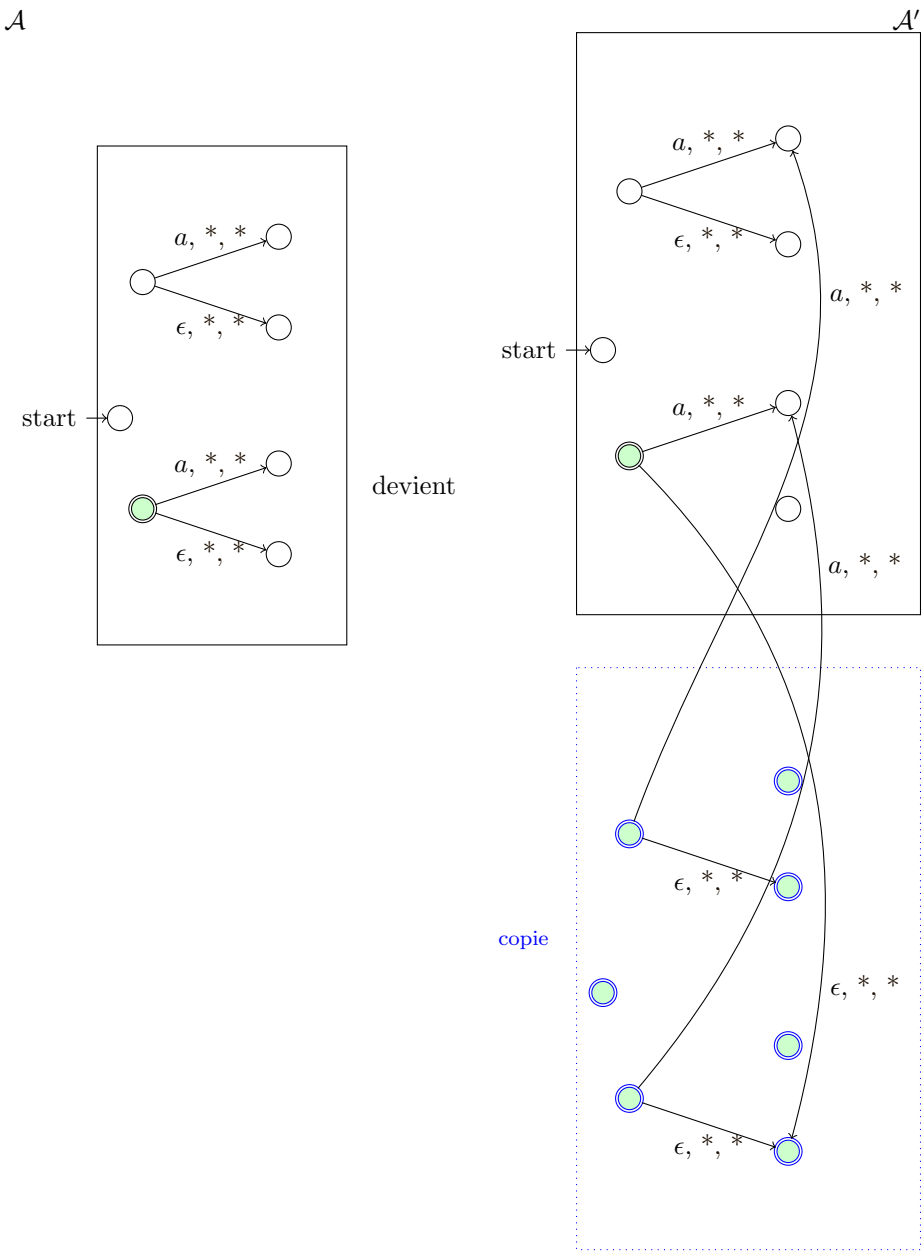


Figure 5.1: Transformation 0.

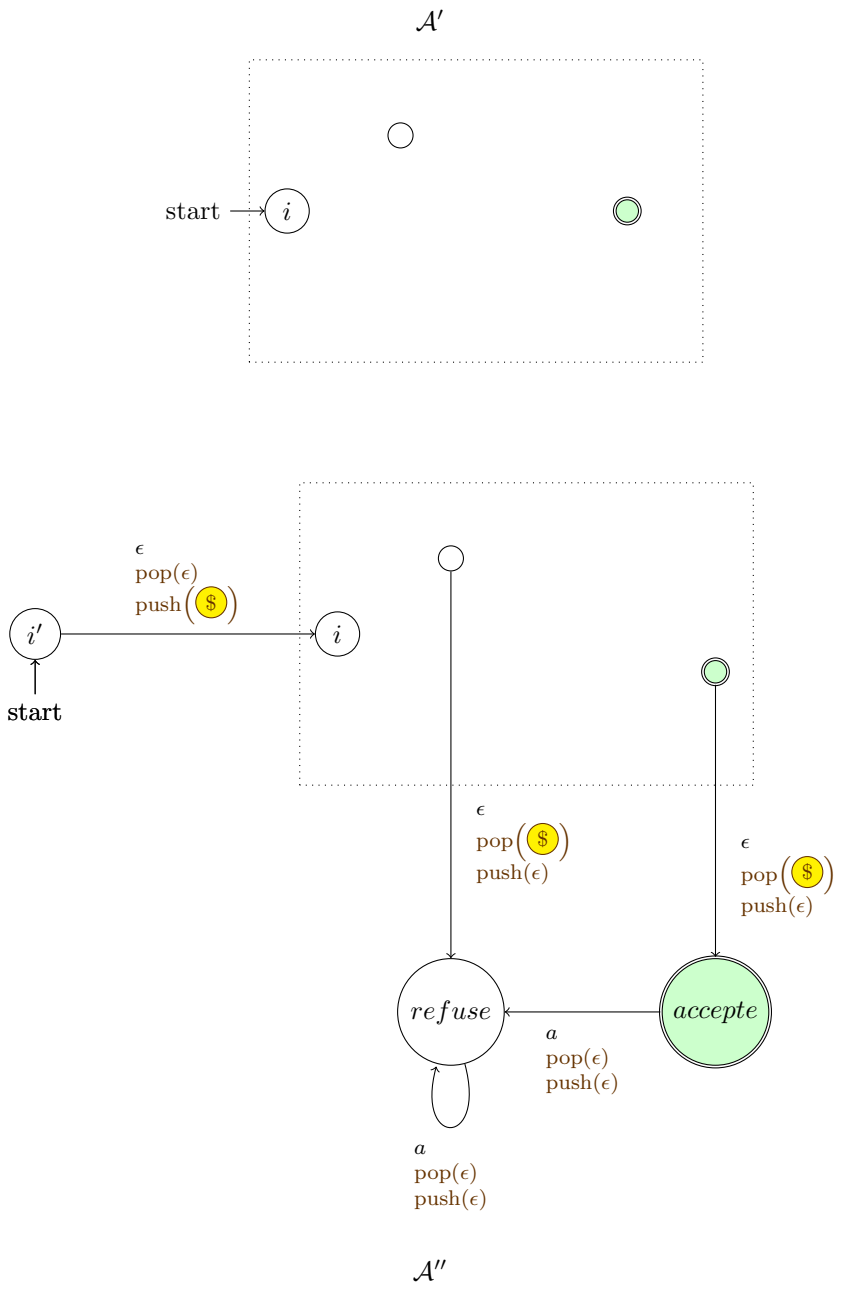
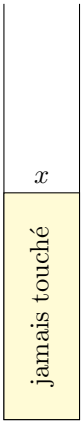


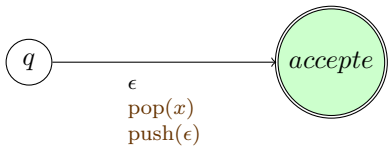
Figure 5.2: Introduction d'un symbole de "fond de pile"



Remarque : on demande bien à ne pas dépiler au delà de x car si on dépile au delà et que l'on boucle, on finit un jour par atteindre une autre situation boucle ou à tout dépiler jusqu'à arriver au fond de pile $\$$ et on finit sur $refuse$ ou *accepte*. TODO: à formaliser mieux

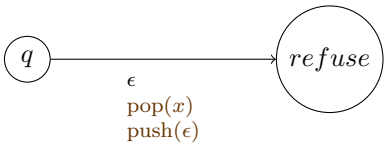
Une situation à boucle (q, x) est dite acceptante si l'on rencontre un état final sur l'unique chemin de ϵ-transitions. Sinon, (q, x) est dite refusante.

Si (q, x) est une situation à boucle acceptante, on supprime la transition où on dépile x de q et on la remplace par



où *accepte* est introduit dans l'étape 1.

Si (q, x) est une situation à boucle refusante, on supprime la transition où on dépile x de q et on la remplace par



où $refuse$ est introduit dans l'étape 1.

■

Maintenant, on s'attaque à la démonstration du théorème 25. Soit L un langage algébrique déterministe. Grâce au lemme 12, il existe un automate $\mathcal{B}$ qui reconnait L, qui lit tout le mot d'entrée. De plus si on est dans une configuration acceptante, on reste dans une configuration acceptante en prenant une ϵ-transition. En particulier, si un mot est accepté, quitte à prendre suffisamment d'ϵ-transitions à la fin, on atteint une configuration acceptante puis toutes les ϵ-transitions nous laisse dans une configuration acceptante :

$$(q_1,\gamma_1) \to^\epsilon (q_2,\gamma_2) \to^\epsilon ((q_3),\gamma_3) \to^\epsilon ((q_4),\gamma_4) \to^\epsilon ((q_5),\gamma_5) \to^\epsilon \dots$$

Si on échange états finaux/non finaux, on obtient l'automate $\overline{\mathcal{B}}$ alors après la lecture du mot dans $\mathcal{B}$ on a :

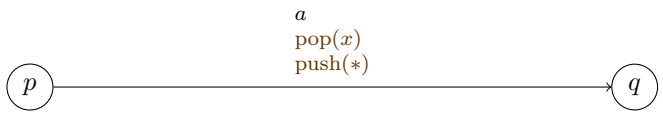
$$((q_1),\gamma_1) \to^\epsilon ((q_2),\gamma_2) \to^\epsilon (q_3,\gamma_3) \to^\epsilon (q_4,\gamma_4) \to^\epsilon (q_5,\gamma_5) \to^\epsilon \dots$$

et le mot est aussi accepté par $\mathcal{B}$. Ce n'est toujours pas bon. Nous n'avons toujours pas $\overline{L(\mathcal{B})} = L(\overline{\mathcal{B}})$. Bref, on a toujours le problème Pb2.

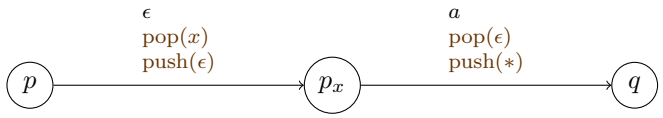
La solution consiste à ne s'intéresser qu'aux états qui lisent effectivement une lettre, que l'on va appeler *états lecteurs* que l'on va représenter dans les schémas en violet gras ○.

Un état est lecteur s'il y part une transition $\overset{a}{\underset{\substack{\text{pop}(\epsilon)\\ \text{push}(*)}}{\longrightarrow}}$ où $a \in \Sigma$. C'est à dire on lit une lettre a, on ne dépile rien et on peut empiler des choses. D'un état lecteur, il n'y part aucune ϵ-transition car l'automate est déterministe. On va introduire une transformation pour éviter les états 'hybrides' (desquels on peut lire effectivement des lettres et aussi tirer des ϵ-transitions).

4. On transforme les transitions de la forme



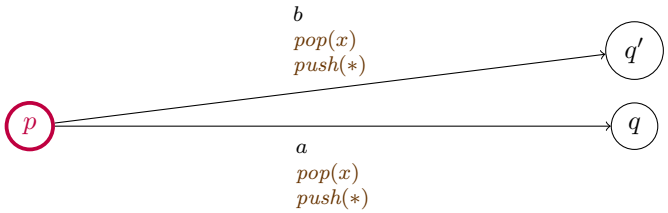
en



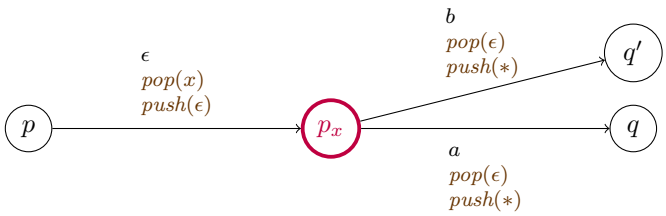
où p_x est un nouvel état. Dans la transformation précédente, on rend p_x final si, et seulement si p est final.

Notons que p_x est un état lecteur. Notons que p n'est pas un état lecteur. Toujours, il n'y a pas d'ϵ-transitions qui partent d'un état lecteur.

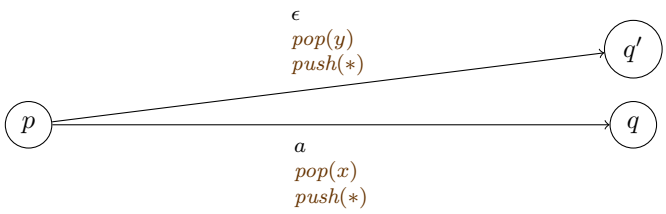
Example 86 *Par exemple,*



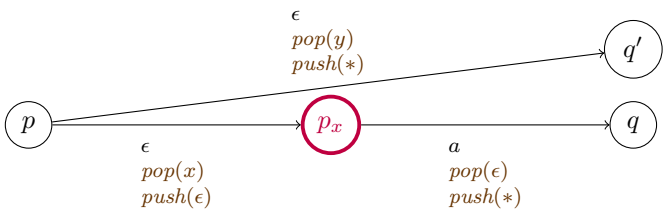
devient



Example 87 *Par exemple,*

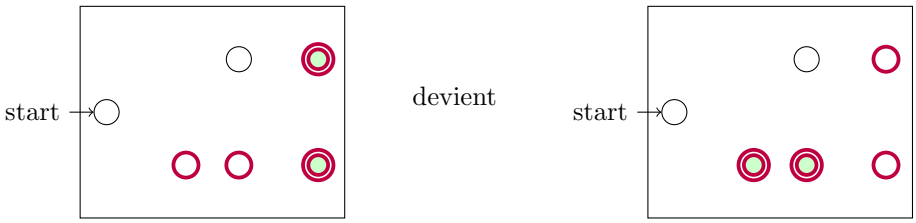


devient



Maintenant, on déclare tous les états non lecteurs comme non-finaux. On appelle $\mathcal{B}'$ l'automate obtenu. Cela n'affecte pas le langage accepté. En effet, s'il y a un état final qui n'est pas lecteur alors tous les états accessibles depuis cet état par des ϵ-transitions sont finaux par le lemme 12. Soit on arrive sur un état lecteur directement qui sera, du coup, final, soit on arrive sur un état p qui aura subi la transformation 4 et donc on atteint un un état lecteur p_x final (sinon, on aurait une exécution infinie de ϵ-transitions, ce qui est interdit par le lemme 12 TODO: à améliorer).

On inverse les états finaux et non-finaux des états lecteurs seulement :



On obtient un automate $\mathcal{B}'^{\bullet}$ qui reconnait le complémentaire, c'est à dire $L(\mathcal{B}'^{\bullet}) = \overline{L(\mathcal{B}')}$.

Example 88 *L'exécution (maximale) de $\mathcal{B}'$ sur le mot ab pourrait ressembler à*

$(i', \epsilon) \to^{\epsilon} (\textcircled{\scriptsize q_2}, \gamma_2) \to^{a} (q_3, \gamma_3) \to^{\epsilon} (q_4, \gamma_4) \to^{\epsilon} (\textcircled{\scriptsize q_5}, \gamma_5) \to^{b} (q_6, \gamma_6) \to^{\epsilon} (\textcircled{\scriptsize q_7}, \gamma_7)$

Dans l'automate $\mathcal{B}'^{\bullet}$ où on a échangé les finaux/non finaux pour les états lecteurs, l'exécution (maximale) est :

$(i', \epsilon) \to^{\epsilon} (\textcircled{\scriptsize q_2}, \gamma_2) \to^{a} (q_3, \gamma_3) \to^{\epsilon} (q_4, \gamma_4) \to^{\epsilon} (\textcircled{\scriptsize q_5}, \gamma_5) \to^{b} (q_6, \gamma_6) \to^{\epsilon} (\textcircled{\scriptsize q_7}, \gamma_7)$