Automates déterministes, automates complets et langages complémentaires

Automates déterministes et automates complets

Un automate $M = (\Sigma, Q, I, F, \delta)$ est dit *déterministe* s'il possède un unique état initial et si par chaque état et pour chaque lettre, il existe au plus une transition :

$|I| = 1$
$\forall p \in Q,\ \forall a \in \Sigma,\ |\delta(p, a)| \leq 1$
Exemple 1 :

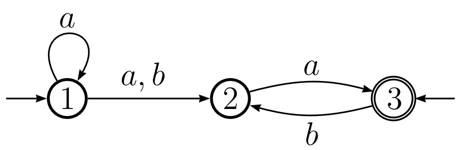


Figure 1 – Automate M_1

M_1 est un automate non déterministe ; il a deux états initiaux 1 et 3 et de plus, il y a deux flèches qui sortent de 1 par la lettre a.

Exemple 2 :

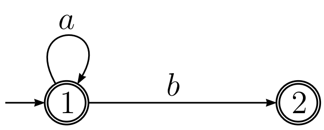


Figure 2 – Automate M_2

M_2 est un automate déterministe ; il a un unique état initial 1 et de 1 il y a une flèche par a et une flèche par b, de 2 il n'y a aucune flèche ni par a ni par b.
On dit que deux automates sont équivalents s'ils reconnaissent le même langage.

Théorème : Tout automate admet un automate déterministe équivalent.

L'intérêt du déterminisme est que, à chaque étape, il existe au plus un choix possible et donc au plus un chemin dans l'automate étiqueté par un mot donné. En conséquence, ce sont des objets particulièrement efficaces et simples à programmer.

On dit que M est *complet* s'il est déterministe et que de plus, par chaque état et pour chaque lettre, il existe exactement une transition :

$|I| = 1$
$\forall p \in Q,\ \forall a \in \Sigma,\ |\delta(p, a)| = 1$

Exemple 3 :

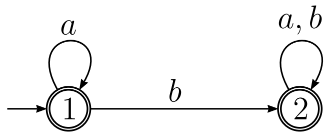


FIGURE 3 – Automate M_3

M_3 est un automate complet ; il a un unique état initial 1 et de 1 et de 2, il y a exactement une flèche par a et une flèche par b.

Dans le cas d'un automate complet, il existe exactement un chemin partant de l'état initial étiqueté par un mot donné.

Algorithme déterminiser
Entrée : $M = (\Sigma, Q, I, F, \delta)$ un automate quelconque
Sortie : $M'_a = (\Sigma, Q'_a, I'_a, F'_a, \delta'_a)$ automate complet équivalent à M
Début

```
    Q'_a ⟵ ∅
    δ'_a ⟵ ∅
    I'_a ⟵ {I}
    F'_a ⟵ ∅
    T ⟵ file_vide
    T ⟵ enfiler( I, T )
    Tant que T est non vide faire
        P ⟵ premier(T)
        T ⟵défiler(T)
        Si P n'est pas dans Q'_a alors
            Q'_a ⟵ Q'_a ∪ {P}
            Pour tout a de Σ faire
                R ⟵ ⋃_{p∈P} δ(p, a)
                δ'_a ⟵ δ'_a ∪ {(P, a, R)}
                T ⟵ enfiler( R, T )
            Fpour
        Fsi
    Ftq
    Pour tout P de Q'_a faire
        Si P ∩ F ≠ ∅ alors
            F'_a ⟵ F'_a ∪ {P}
        Fsi
    M'_a ⟵ (Σ, Q'_a, I'_a, F'_a, δ'_a)
    Retourner M'_a
Fin
```

Exemple 4 :

Reprenons l'automate M_1. Sa table de transitions est donnée par

	a	b
$\rightarrow 1$	$\{1,2\}$	$\{2\}$
2	$\{3\}$	$\emptyset$
$\leftrightarrow 3$	$\emptyset$	$\{2\}$

la table de transition du déterminisé :

	a	b
$\leftrightarrow \{1,3\}$	$\{1,2\}$	$\{2\}$
$\{1,2\}$	$\{1,2,3\}$	$\{2\}$
$\{2\}$	$\{3\}$	$\emptyset$
$\leftarrow \{1,2,3\}$	$\{1,2,3\}$	$\{2\}$
$\leftarrow \{3\}$	$\emptyset$	$\{2\}$
$\emptyset$	$\emptyset$	$\emptyset$

qui correspond après renumérotation des états à l'automate :

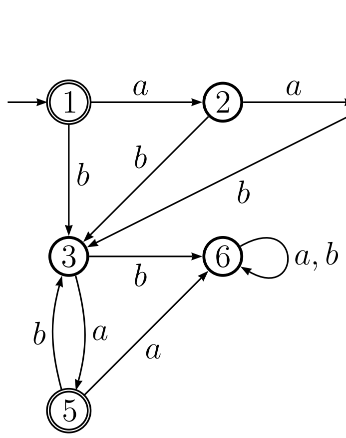


FIGURE 4 – Automate déterminisé de M_1

L'automate obtenu est non seulement déterministe, mais aussi complet. Ceci est dû au fait que l'algorithme construit les flèches de chaque état atteint.

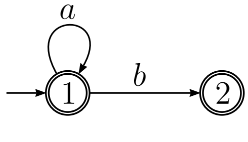
Algorithme de complétion :

Théorème : Tout automate admet un automate équivalent complet.

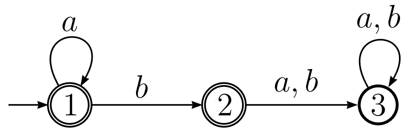
Soit M un automate. Si M n'est pas déterministe, l'algorithme de déterminisation retourne un automate complet équivalent à M. Si M est un automate déterministe non complet, on rajoute un nouvel état à M et on envoie toutes les flèches manquantes sur cet état. On appelle cet état un *état puits*. L'automate obtenu est complet.

Exemple 5 :

de l'automate



on passe à l'automate :

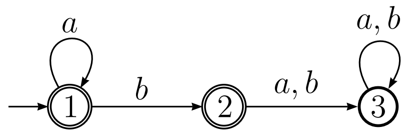


Complémentaire

Théorème : $Rec(\Sigma^*)$ est fermé par passage au complémentaire.

Soit $M = (\Sigma, Q, I, F, \delta)$ un automate complet qui reconnait L. L'automate $M' = (\Sigma, Q, I, Q \setminus F, \delta)$ reconnait $\Sigma^* \setminus L$.

Exemple 6 :



est un automate complet, son complémentaire est :