Théorie des langages et des automates

François Lemieux

Département d'informatique et de mathématique
Université du Québec à Chicoutimi

1 Minimisation d'automates

La figure 1 représente le graphe de transition d'un automate fini déterministe M_1. Dans cette section nous nous posons la question suivante: Est-il possible de réduire le nombre d'états dans cet automate? De façon plus générale: Comment peut-on trouver le plus petit AFD possible qui reconnaisse un langage régulier donné?

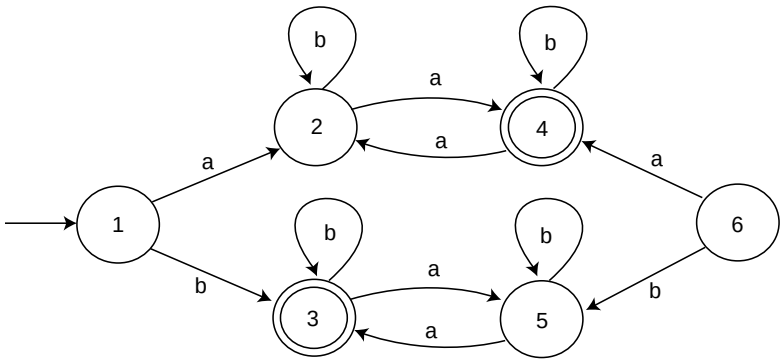

Figure 1: L'AFD $M_1 = (\{1,2,3,4,5,6\}, \{a,b\}, 1, \{1,3,4\}, \delta)$

Après avoir examiné M_1, on remarque que l'état 6 est *inaccessible* à partir de l'état initial. En d'autres termes, il n'existe aucun mot $w \in \{a,b\}^*$ tel que $\delta^*(1,w) = 6$. Si on enlève cet état, on obtient l'automate M_2 de la figure 2. On observe que M_2 est déterministe et reconnaît le même langage que M_1 puisque l'état que nous avons enlevé était, à toutes fins pratiques, inutile. On peut faire de cette observation une règle générale:

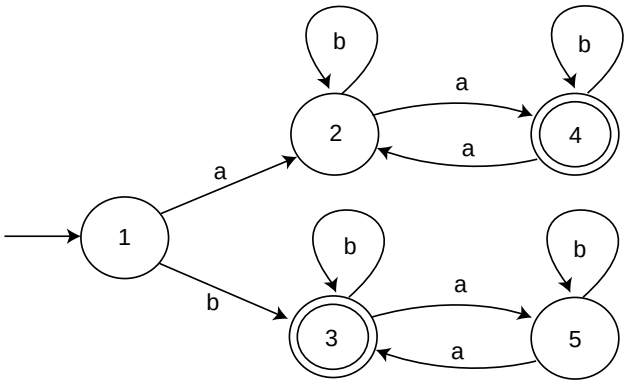

Figure 2: L'AFD M_2 obtenu en enlevant les états inacessibles de M_1

Règle 1 : Soit M un AFD et soit M' l'AFD obtenu en enlevant tous les états inacessibles de M. Alors $L(M) = L(M')$.

Maintenant, après avoir appliqué la règle 1, peut-on encore réduire le nombre d'états de M_2? La réponse est encore oui! La figure 3 représente le graphe de transition de l'AFD M_3. Cet automate est le plus petit AFD reconnaissant $L(M_1)$.

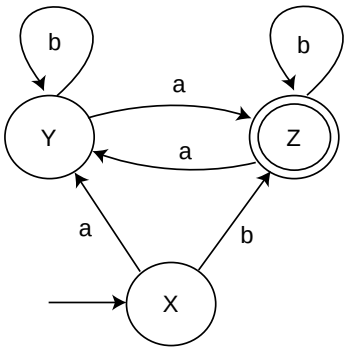

Figure 3: L'AFD minimal M_3 équivalent à M_1

Comment a-t-on obtenu cet automate? Pour répondre à cette question il est nécessaire de faire quelques observations sur l'AFD M_2 de la figure 2.

Dénotons par L_i $(i = 1, 2, 3, 4, 5)$ l'ensemble des mots menant à un état final à partir de l'état i. Plus formellement, définissons

$$L_i = \{w \in \{a, b\}^* \mid \delta^*(i, w) \in F\}$$

On observe que

- $L_2 = \{w \in \{a,b\}^* \mid w \text{ contient un nombre impair de } a\}$
- $L_3 = \{w \in \{a,b\}^* \mid w \text{ contient un nombre pair de } a\}$
- $L_4 = \{w \in \{a,b\}^* \mid w \text{ contient un nombre pair de } a\}$
- $L_5 = \{w \in \{a,b\}^* \mid w \text{ contient un nombre impair de } a\}$

On a donc

$$L_2 = L_5 \text{ et } L_3 = L_4$$

On peut interpréter cette observation de la façon suivante:

- À chaque fois que l'on se retrouve dans l'état 2, on pourrait poursuivre l'exécution à partir de l'état 5 sans rien changer au résultat.
- À chaque fois que l'on se retrouve dans l'état 3, on pourrait poursuivre l'exécution à partir de l'état 4 sans rien changer au résultat.
- À chaque fois que l'on se retrouve dans l'état 4, on pourrait poursuivre l'exécution à partir de l'état 3 sans rien changer au résultat.
- À chaque fois que l'on se retrouve dans l'état 5, on pourrait poursuivre l'exécution à partir de l'état 2 sans rien changer au résultat.

Cette observation est très importante pour notre propos et elle suggère la définition suivante:

Définition: Deux états i et j sont *équivalents* si $L_i = L_j$. En d'autres termes, i et j sont équivalents si pour tout $w \in A^*$, on a

$$\delta^*(i,w) \in F \text{ si et seulement si } \delta^*(j,w) \in F$$

Ainsi, dans l'AFD M_2 les états 2 et 5 sont équivalents et les états 3 et 4 sont équivalents. L'état 1 n'est équivalent à aucun autre état.

Remarque: La définition précédente est une équivalence au sens mathématique du terme. Rappelons qu'une relation d'équivalence doit satisfaire trois conditions:

1. (Réflexivité:) Chaque état est équivalent à lui-même.
2. (Symétrie:) Si i est équivalent à j alors j est équivalent à i.
3. (Transitivité:) Si i est équivalent à j et si j est équivalent à k alors i est équivalent à k.

De façon générale, si i et j sont deux états équivalents d'un automate donné, alors *à chaque fois que l'on se retrouve dans l'état i, on pourrait poursuivre l'exécution à partir de l'état j sans rien changer au résultat.* Cela conduit à l'observation suivante:

Observation : Il est toujours possible d'ajouter des transitions ϵ entre deux états équivalents sans rien changer au langage reconnu.

Si l'on applique cette observation sur l'AFD de la figure 2, on obtient l'automate M_4 dont le graphe de transition est représenté à la figure 4.

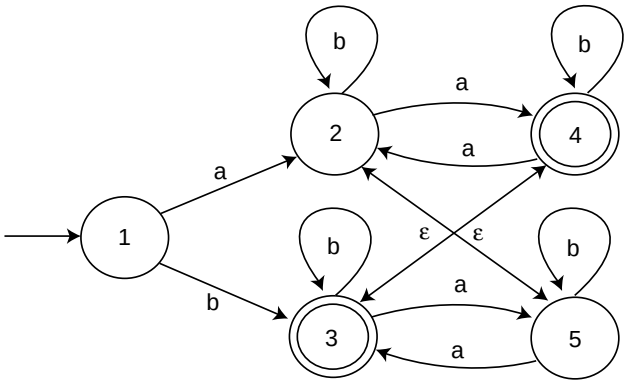

Figure 4: L'AFN M_4 obtenu à partir de M_2 en ajoutant des transitions ϵ entre les paires d'états équivalents.

Remarquez que l'automate obtenu n'est plus déterministe. Cependant, on peut obtenir un AFD M_5 à partir de L'AFN M_4 en utilisant la méthode vue précédemment. Le graphe de transition de l'AFD M_5 obtenue est illustré à la figure 5.

On remarque que l'AFD M_5 est identique à l'automate M_3, l'automate minimal qui reconnaît le langage $L(M_1)$. On remarque surtout que la transformation de M_4 à M_5 n'a pas augmenté le nombre d'états. Bien au contraire.

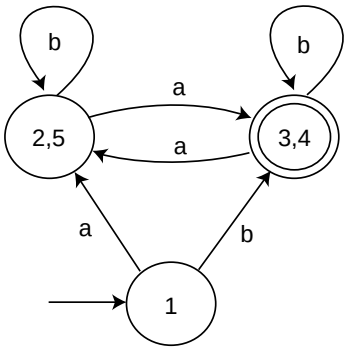

Figure 5: L'AFD M_5 obtenu à partir de l'AFN M_4

Les états de M_5 sont des ensembles d'états de M_4 mais ces ensembles sont disjoints. En d'autres termes, chaque état de M_4 se retrouve dans un unique état de M_5. Cela nous permet d'énoncer une seconde règle générale:

Règle 2 : Soit $M = (Q, A, q_0, F, \delta)$ un AFD et soit $M' = (Q', A, S_0, F', \delta')$ l'automate minimal qui reconnaît $L(M)$. Alors les éléments de Q' sont des sous-ensembles disjoints de Q. De plus, deux états de Q appartiennent au même sous-ensemble dans Q' si et seulement s'ils sont équivalents.

2 Définition

Nous allons maintenant définir de façon plus formelle ce qu'est l'automate minimal d'un langage. Soit $M = (Q, A, 1, F, \delta)$ un AFD dont tous les états sont accessibles à partir de l'état initial. Pour chaque état $i \in Q$, on défini le sous-ensemble $[i] \subseteq Q$ de la façon suivante:

$$[i] = \{k \in Q \mid k \text{ est équivalent à } i\}$$

Observation : Si i et j sont deux états dans Q alors $[i]$ et $[j]$ sont soit égaux ou soit disjoints.

L'automate minimal reconnaissant le langage $L(M)$ est l'AFD $M' = (Q', A, [1], F', \delta')$ tel que

- $Q' = \{[i] \mid i \in Q\}$

- $[1] \in Q'$ est l'état initial.
- $F' = \{[i] \in Q' \mid i \in F\}$ est l'ensemble des états acceptants.
- $\delta' : Q' \times A \to Q'$ est la fonction de transition définie par

$$\delta'([i], a) = [\delta(i, a)]$$

3 Trouver les états équivalents

Comme nous venons de le voir, une fois que nous avons déterminé quels sont les états équivalents, il devient facile de construire l'automate minimal. Le problème se réduit donc à trouver effectivement les états équivalents. L'exemple que nous avons vu était relativement simple mais en général ce problème apparaît beaucoup plus complexe. Heureusement, il existe un algorithme simple pour résoudre ce problème. En fait, plutôt que de chercher les paires d'états équivalents, nous allons déterminer quels sont les paires d'états qui ne le sont pas. Nous dirons que deux états sont *distincts* s'ils ne sont pas équivalents.

La première étape de l'algorithme consiste à construire un graphe dirigé $G(M)$ à partir de l'AFD $M = (Q, A, 1, F, \delta)$. Afin de ne pas confondre ce graphe avec le graphe de transition nous l'appellerons *graphe d'équivalence*. Les noeuds du graphe d'équivalence sont des paires d'états de Q, c'est-à-dire des sous-ensembles de Q contenant exactement deux éléments. Il y a un arc entre le noeud $\{i, j\}$ et le noeud $\{k, l\}$ si et seulement si $\{k, l\} = \{\delta(i, a), \delta(j, a)\}$ pour au moins une lettre $a \in A$.

Exemple: La figure 6 donne le graphe d'équivalence de l'AFD de la figure 2. Les étiquettes associées aux arcs indiquent quelles lettres permettent de passer d'une paire d'états à une autre paire d'états. Par exemple, puisque $\delta(2, a) = 4$ et $\delta(5, a) = 3$ alors il y a un arc étiqueté avec la lettre a partant du noeud $\{2, 5\}$ vers le noeud $\{3, 4\}$.

Observation 1: S'il existe un chemin entre les noeuds $\{i, j\}$ et $\{k, l\}$ alors il existe un mot $w \in A^*$ tel que $\{k, l\} = \{\delta^*(i, w), \delta^*(j, w)\}$.

Exemple: Considérant la figure 6, il existe un chemin entre les noeuds $\{1, 2\}$ et $\{4, 5\}$. En utilisant le mot ba on a bien que $\{4, 5\} = \{\delta^*(1, ab), \delta^*(2, ab)\}$.

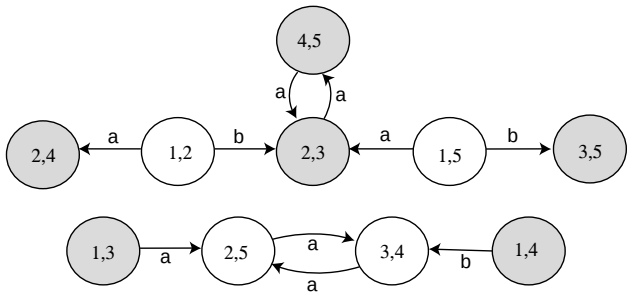

Figure 6: Le graphe d'équivalence de l'AFD M_2

Similairement, il existe un chemin entre les noeuds $\{1,4\}$ et $\{2,5\}$. En utilisant encore le mot ba on voit que $\{2,5\} = \{\delta^*(1,ab), \delta^*(4,ab)\}$.

Observation 2: Deux états $i, j \in Q$ sont distincts s'il existe un chemin dans le graphe d'équivalence entre le noeud $\{i,j\}$ et un noeud $\{k,l\}$ tel qu'un seul des états k et l est acceptant.

Cette observation suggère l'algorithme suivant pour trouver les paires d'états équivalents:

```
Étant donné un AFD M:
1) Calculer le graphe d'équivalence G(M).
2) Marquer tous les noeuds possédant exactement un état
   acceptant.
3) Marquer tous les noeuds  {i,j} tel qu'il existe un chemin
   entre {i,j} et un noeud déjà marqué.
4) Tous les noeuds qui n'ont pas été marqués représentent des
   paires d'états équivalents.
```

Exemple: La figure 6 représente l'état du graphe d'équivalence après la seconde étape de l'algorithme. Les noeuds marqués sont de couleur grise. Après l'étape 3, deux nouveaux noeuds seront marqués: les noeuds $\{1,2\}$ et $\{1,5\}$. Les noeuds $\{2,5\}$ et $\{3,4\}$ demeureront non marqués, ce qui indique que l'état 2 est équivalent à l'état 5 et que l'état 3 est équivalent à l'état 4.

Lorsque le graphe d'équivalence est de grande taille, il est plus difficile de déterminer s'il existe un chemin entre deux noeuds. Pour cette raison,

l'étape 3 de l'algorithme précédent mérite d'être détaillée d'avantage. Une des façons de procéder consiste à effectuer une boucle où, à chaque itération, on marque les noeuds possédant un arc vers un noeud marqué à l'itération précédente. On termine lorqu'aucun nouveau noeud ne peut être marqué. Nous obtenons donc l'algorithme suivant:

```
Étant donné un AFD M:
1) Calculer le graphe d'équivalence G(M).
2) Marquer tous les noeuds possédant exactement un état
   acceptant.
3) Répéter:
       Marquer tous les noeuds {i,j} qui n'ont pas déjà été
       marqués et tel qu'il existe un arc entre {i,j} et un
       noeud marqué à l'itération précédente.
   Tant qu'il reste des noeuds à marquer
4) Tous les noeuds qui n'ont pas été marqués représentent
   des paires d'états équivalents.
```

Exemple: La figure 7 représente un AFD à 10 états. Le graphe d'équivalence de cet automate est représenté à la figure 8. La figure 8(a) donne l'état du graphe après la seconde étape de l'algorithme. Les figures 8(b), 8(c) et 8(d) indiquent quels noeuds seront marqués après chacune des trois itérations nécessaires à la complétion de l'algorithme. La figure 9 donne l'AFD minimal résultant.

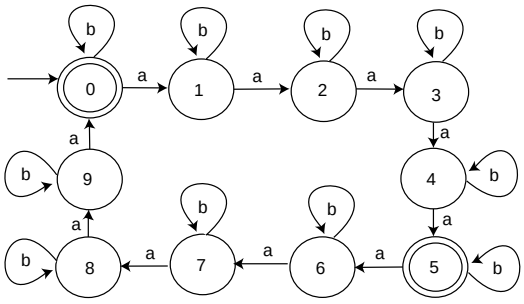

Figure 7:

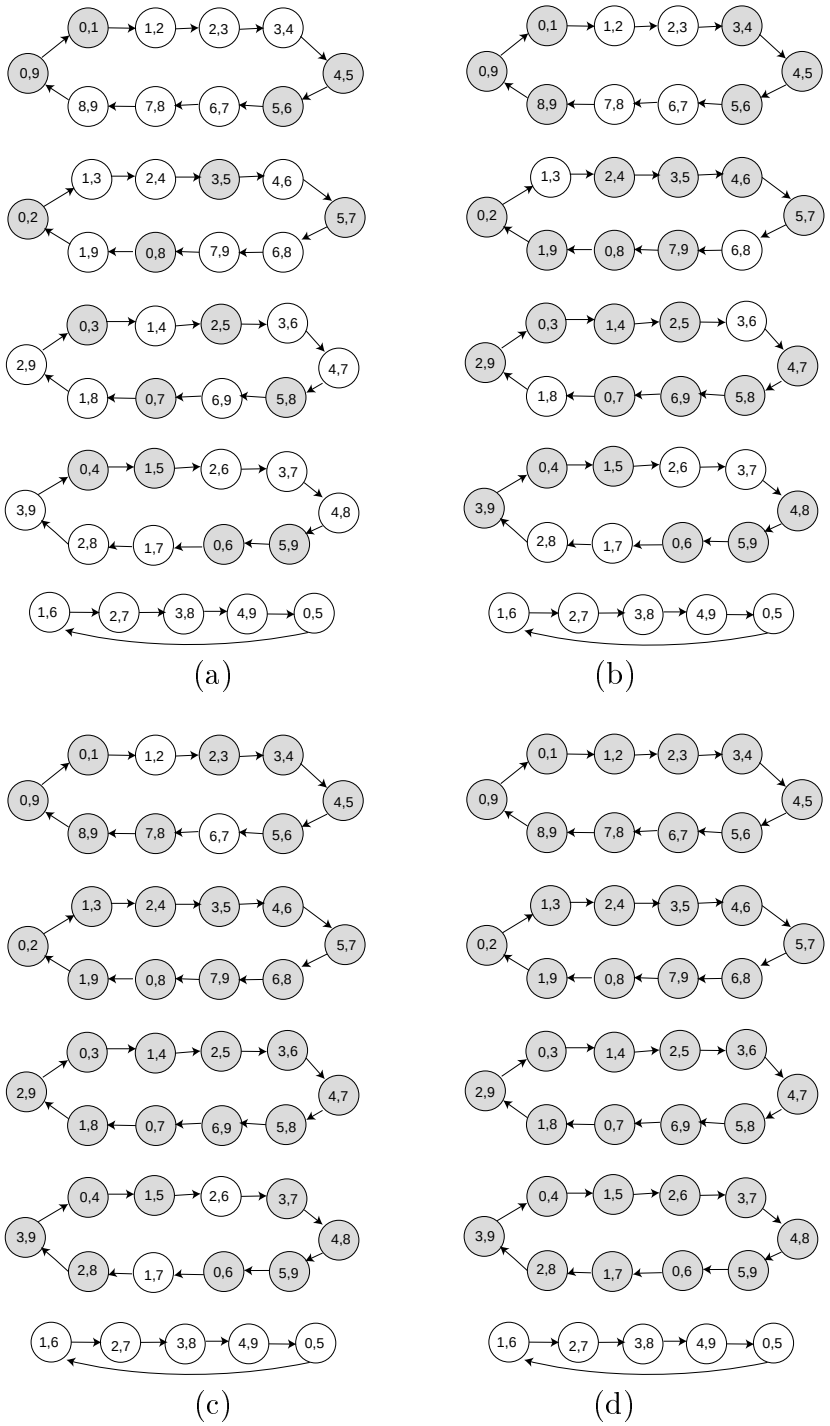
0,1 1,2 2,3 3,4 4,5 5,6 6,7 7,8 8,9 0,9
1,3 2,4 3,5 4,6 5,7 6,8 7,9 0,8 1,9 0,2
0,3 1,4 2,5 3,6 4,7 5,8 6,9 0,7 1,8 2,9
0,4 1,5 2,6 3,7 4,8 5,9 0,6 1,7 2,8 3,9
1,6 2,7 3,8 4,9 0,5
(a)
(b)
(c)
(d)

Figure 8:

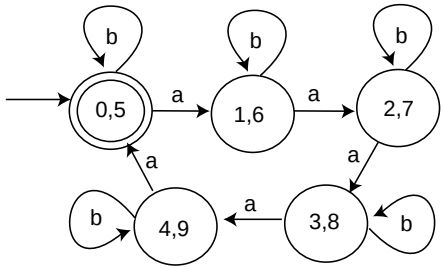

Figure 9: