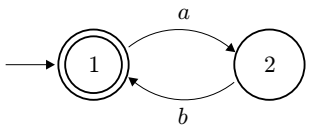
IN 101 - TD 06
14 octobre 2011

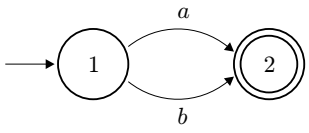
Énoncé

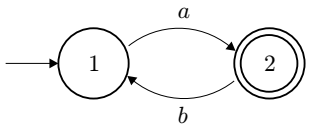
ENSTA ParisTech

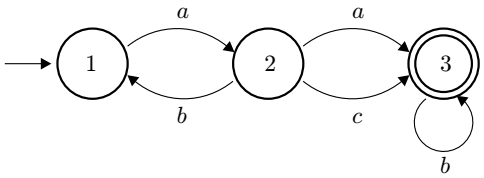
Exercice 1 : Automates finis

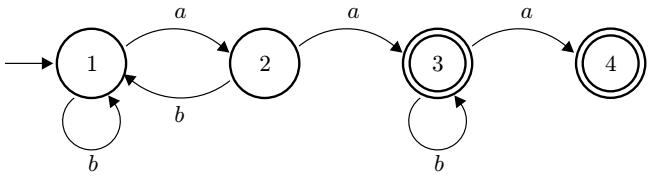
1.a] Quels sont les langages reconnus par les automates suivants (on rappelle que le langage reconnu est l'ensemble *complet* des mots reconnus, pas juste un sous-ensemble) ?

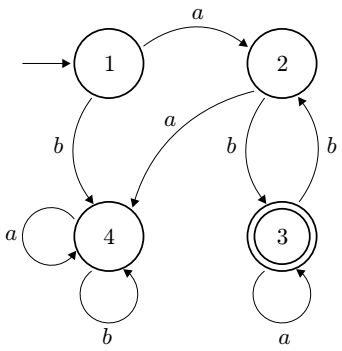
1.


2.


3.


4.


5.


6.


1.b] Construire des automates reconnaissant respectivement les langages suivants :

1. $(ba|a)^*$
2. $(ba|b)^*$
3. $(a|aa|aaa)$
4. $b^*(a|aa|aaa)b^*$

1.c] Construire des automates (avec pour alphabet les chiffres de 0 à 9) reconnaissant :

1. les nombres pairs,
2. les multiples de 3,
3. les multiples de 4.

1.d] On veut construire un automate reconnaissant un nombre complexe (à coefficients entiers). L'alphabet contient les caractères '+', '-', et 'i' et les chiffres de 0 à 9. On veut reconnaître les nombres de la forme '+3', '2i', '-4+5i', '3i-2', 'i+2'... Les nombres des formes suivantes ne doivent en revanche pas être reconnus : '-+3', 'i4+5', '2i+i', '5-3', '-'... Écrivez le langage correspondant à ces nombres complexes et construisez un automate déterministe reconnaissant ce langage. Pour simplifier, on reconnaitra aussi des entiers commençant par 0, par exemple, +05-0i sera reconnu.

Exercice 2 : Automates non-déterministes

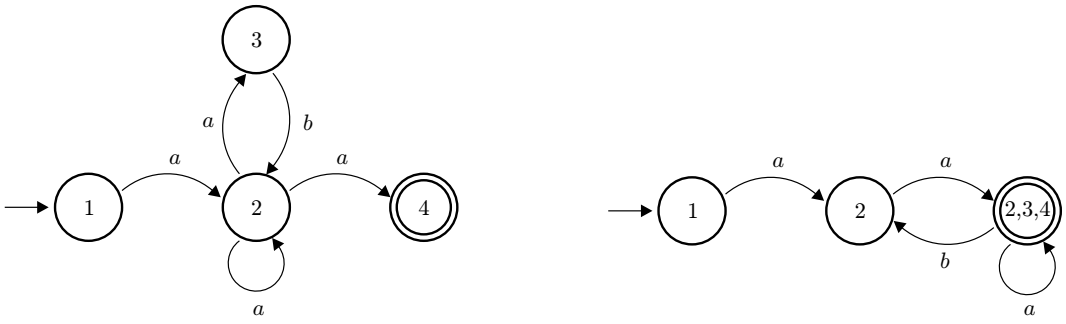
Un automate fini non-déterministe est tel que partant d'un même état et pour un même symbole, plusieurs états peuvent être accessibles. Graphiquement, cela signifie qu'un sommet peut être origine de plusieurs flèches étiquetées par le même symbole. Un mot est reconnu par un automate non-déterministe s'il existe une suite de transitions qui mènent à un état final. L'avantage de tels automates est que pour certains langages, il sont faciles à construire, alors que la conception d'un automate déterministe n'est pas toujours évidente.

2.a] Construire un automate non déterministe reconnaissant le langage $L = (a|b)^*abb$.

L'utilisation d'un automate non-déterministe est en revanche plus chère : il est nécessaire de maintenir en permanence un ensemble d'états possibles (tous les états dans lesquels le mot lu peut avoir amené l'automate). Il est donc préférable de déterminiser un automate non-déterministe avant de l'utiliser, et il existe pour cela un algorithme assez simple. Pour construire un automate déterministe $\mathcal{D}$ à partir d'un automate non déterministe $\mathcal{A}$:

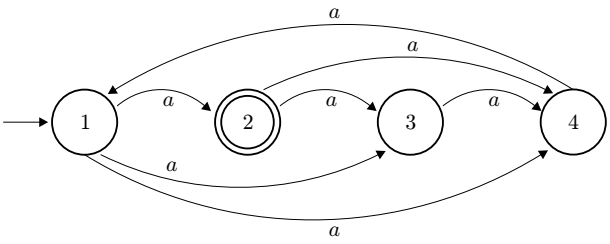
- l'alphabet Σ reste identique,
- si Q est l'ensemble des états de $\mathcal{A}$, l'ensemble des états possibles de $\mathcal{D}$ est l'ensemble des parties de Q (donc si $\mathcal{A}$ a n états, $\mathcal{D}$ peut en avoir jusqu'à 2^n),
- si l'état initial de $\mathcal{A}$ est p_0, l'état initial de $\mathcal{D}$ est $\{p_0\}$,
- si l'ensemble des états finaux de $\mathcal{A}$ est F, l'ensemble des états finaux de $\mathcal{D}$ est l'ensemble des états f tels que $f \cap F \neq \emptyset$ (si f contient un état final de $\mathcal{A}$, c'est un état final de $\mathcal{D}$),
- si la fonction de transition de $\mathcal{A}$ fait passer de l'état $p_i \in Q$ à l'ensemble d'états $q_i^s \subseteq Q$ en lisant le symbole s, la fonction de transition δ de $\mathcal{D}$ fait passer l'automate déterministe de l'état $\{p_0, p_1, \ldots, p_k\}$ à l'état $\bigcup_{i=0}^{k} q_i^s$ en lisant le symbole s.

Exemple. Ci-dessous, l'automate non-déterministe de gauche reconnaît le langage $L = a(a|ab)^*a$ et l'automate déterministe de droite a été obtenu par l'algorithme que nous venons de décrire.



2.b] Construire un automate déterministe reconnaissant le langage $L = (a|b)^*abb$ (en passant par un automate non-déterministe).

2.c] Quel est le langage reconnu par l'automate suivant ? Quels mots ne sont pas reconnus par cet automate ?



Exercice 3 : *Pattern-matching* à base d'automates finis

3.a] Construire l'automate de recherche du motif *abaabbaaa* (comme vu en cours, vous pouvez commencer par construire l'arrête centrale, puis construire le reste des transitions en utilisant la partie de l'automate déjà construite).

3.b] On note Σ^* une suite quelconque d'éléments de Σ, Σ^+ une suite quelconque non vide d'éléments de Σ et Σ un élément quelconque de Σ. Dessinez les automates reconnaissants les langages Σ^*, Σ^+, et Σ.

3.c] On veut construire les automates permettant de reconnaître les motifs $ab\Sigma^*cd$ et $ab\Sigma^+cd$. Reconnaître l'un de ces motifs consiste à reconnaître un premier motif ab puis, dès que ce motif a été reconnu, de chercher à reconnaître le motif cd. En revanche, comme n'importe quel texte peut se trouver entre les deux motifs, une fois que le premier motif a été trouvé on ne revient plus jamais en arrière (avant le dernier état de l'automate reconnaissant le premier motif). Déduisez-donc des automates reconnaissant ab et cd les 2 automates reconnaissant les motifs $ab\Sigma^*cd$ et $ab\Sigma^+cd$ (on note $[\hat{}ab]$ un caractère quelconque autre que a ou b).

3.d] On veut construire l'automate reconnaissant le motif $ab\Sigma cd$. Contrairement aux deux motifs précédents, on cherche les deux motifs ab et cd séparés d'un seul caractère. Donc, une fois que l'on a vu le premier motif, si le motif cd n'apparaît pas un caractère après on repart du tout début. Attention, le caractère Σ peut aussi être un a, modifiant la façon dont on repart au début. Vérifiez en particulier que votre automate reconnaît le texte *ababzcd*. Pour simplifier, vous pourrez commencer par construire un automate non-déterministe, puis vous le déterminiserez.