Langages, Automates et Compilation

Dr Mouhamadou GAYE

UFR Sciences Et Technologies
Département d'Informatique
Licence 3 en Informatique Option Génie Logiciel

23 septembre 2020

Automates à états finis

Automate fini

Définition

Un automate fini est défini par un 5-uplet $A = (\Sigma, E, E_o, F, \delta)$, où

- Σ est l'ensemble fini des symboles
- E est un ensemble fini : l'ensemble des états
- $E_o \subset E$ est le sous-ensemble des états initiaux
- $F \subset E$ est le sous-ensemble des états finaux
- δ est un ensemble fini de transitions : une transition est un triplet (i, a, j), où i et j sont des états et a est un symbole.

Une paire (q, w), où $q \in E_o$ est un état, et $w \in \Sigma^*$ est un mot de l'alphabet Σ est appelée une configuration.

Reconnaissance d'un mot

Définition

La reconnaissance d'un mot w par un automate A est la suite des configurations :

$(q_o, w) \rightarrow (q_1, w_1) \rightarrow (q_2, w_2) \rightarrow ... \rightarrow (q_n, \varepsilon)$

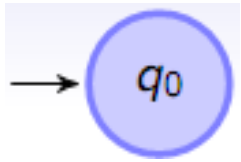
Langage reconnu par un automate

Définition

Le langage reconnu par un automate A est l'ensemble de toutes les chaînes reconnues. Il est noté L(A) .

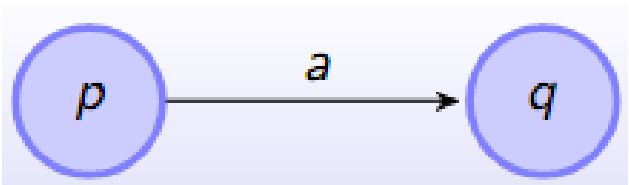
Représentation graphique

- Etat initial



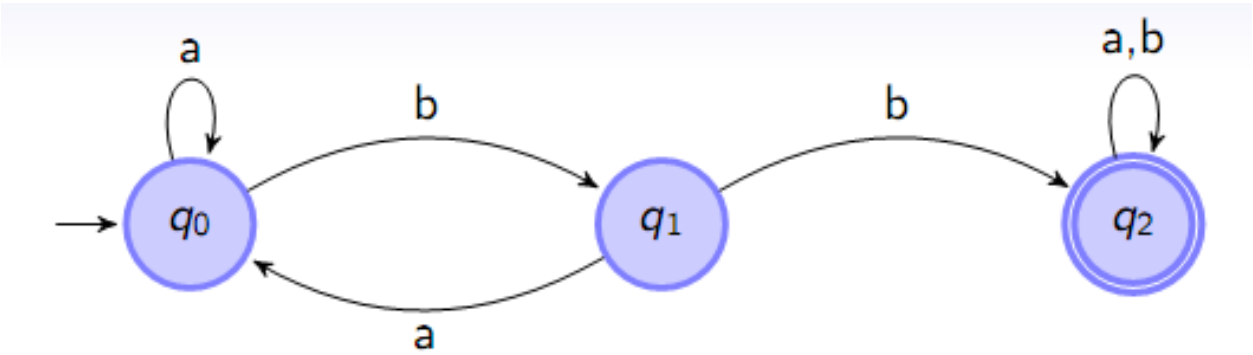
- Etat final

- Transition entre deux états p et q : $\delta(p, a) = q$



Représentation graphique

- **Exemple :**



Automates déterministes

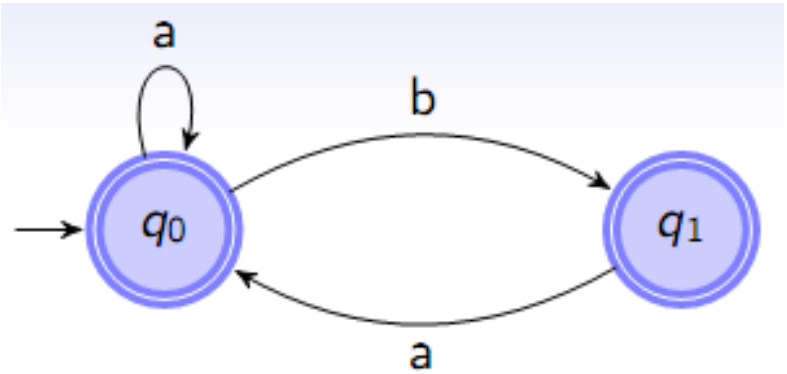
Définition

Un automate déterministe (AFD) est un automate :

- avec un unique état initial
- à partir de chaque état, il y a au plus une transition d'étiquette donnée.

Automates déterministes

- **Exemple :**



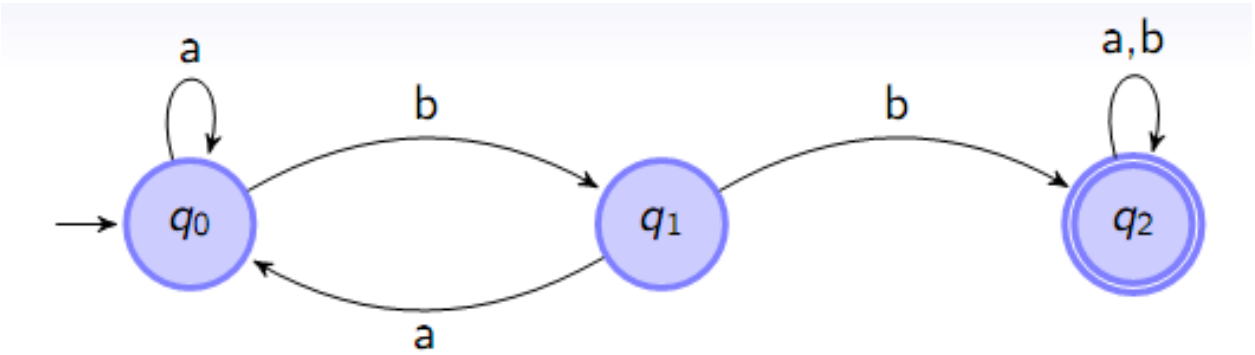
Automates déterministes complets

Définition

Un automate déterministe complet (AFDC) est un automate déterministe pour lequel il y a, à partir de chaque état, une et une seule transition pour chaque étiquette possible de Σ.

Automates déterministes complets

- Exemple :



Automates déterministes

Théorème

Tout automate peut être transformé en un automate déterministe reconnaissant le même langage.

Preuve :

L'idée est qu'il faut regrouper les états auxquels on peut arriver en lisant un même symbole.

Théorème

Tout automate déterministe peut être transformé en un automate déterministe complet reconnaissant le même langage.

Preuve :

L'opération est élémentaire : si l'AFD n'est pas déjà complet,

- lui rajouter un nouvel état ("état poubelle")
- rajouter les transitions d'étiquettes manquantes en les dirigeant toutes vers cet état poubelle P ; ne pas oublier les transitions de P lui-même vers P.

Automates déterministes

Théorème

Si L un langage reconnu par un automate, alors le langage complémentaire Σ^* - L est aussi reconnu par un automate.

Preuve :

Partir d'un automate A reconnaissant L ; le transformer en un AFD, puis en un AFDC.

Automates non déterministes

Les automates non déterministes sont obtenus en enrichissant le fonctionnement des AFD de deux nouvelles capacités :

- on autorise des transitions sur le mot vide (i.e., qui se font sans déplacement de la tête de lecture sur le mot d'entrée),
- on permet qu'à une lettre et un état fixés correspondent plusieurs transitions. Ainsi, lorsqu'il est dans un état p et qu'il lit une lettre a, l'automate peut choisir de manière non déterministe d'évoluer vers différents nouveaux états.

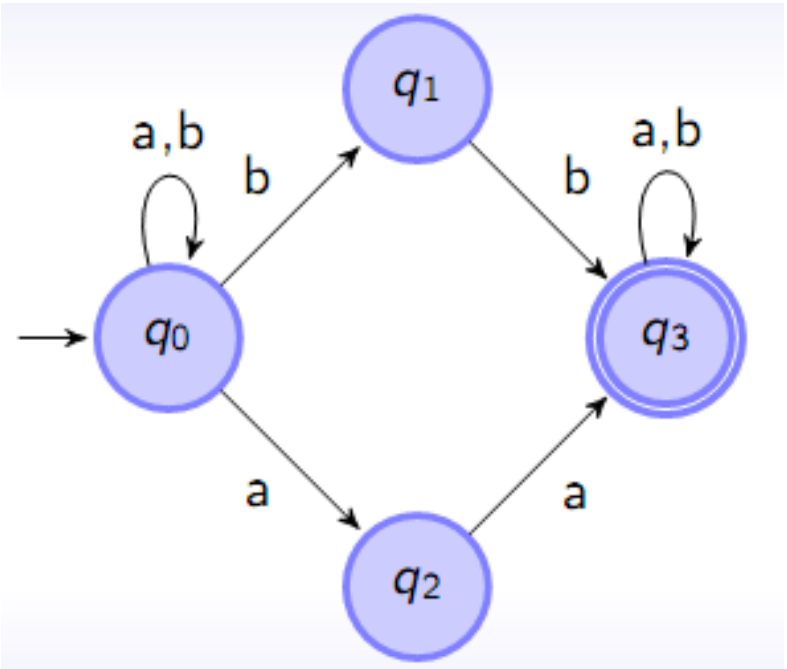
Automates non déterministes

Définition

Les définitions des AFN et des AFD sont similaires sauf concernant la nature de la fonction de transition. Dans un automate non déterministe, la fonction prend un couple constitué d'un état et d'une lettre ou du mot vide et lui associe un ensemble d'états q $\subseteq$ Q accessibles en une transition.

Automates non déterministes

- **Exemple :**



Automates non déterministes

Théorème

Tout automate non déterministe est équivalent à un automate déterministe.

Preuve :

Construction de l'AFD

- Soit A = (Σ, E, E_o, F, δ) l'automate initial donné.
- Le nouvel automate, déterministe, sera noté A' = (Σ , E', $\{s'_o\}$, F', δ') .
- Les états de ce nouvel automate sont des groupes d'états de l'ancien automate, c'est-à-dire des sous-ensembles de l'ensemble E .
- L'état initial s'_o est unique.

Construction de l'AFD

Algorithme de détermination

1. initialiser : $s'_o = E_o$, $E' = \{s'_o\}$
2. répéter, jusqu'à ce que l'ensemble E' soit stationnaire (c'est-à-dire que plus aucun élément nouveau ne s'y rajoute) :
   pour chaque état $A' \epsilon$ E' venant d'être construit, et pour chaque symbole $a \epsilon \Sigma$,
   - considérer (dans A) toutes les transitions d'étiquette a issues d'un état $A \epsilon A'$, et regrouper leurs états d'arrivée
   - si ce groupe d'états n'est pas encore un élément de E', on crée ce nouvel état en le rajoutant à l'ensemble E'
   - rajouter la transition d'étiquette a issue de A' vers cet état.
3. définir les états finaux de A' : ce sont tous les états qui contiennent au moins un état final de A.

Automates non déterministes

- **Proposition**
  L'ensemble expressions régulières est clos pour l'union, la concaténation des langages et la fermeture de Kleene. Autrement dit, si L1 et L2 sont reconnaissables, alors L1∪L2, L1L2 et L1* sont reconnaissables.
  Preuve :

Des automates aux expressions régulières

- Soit $\mathcal{A}$ =(V, $\mathcal{Q}$, δ, q_o, F) un automate fini déterministe et p ϵ $\mathcal{Q}$. Si les transitions d'origine p s'écrivent p $\rightarrow$ p_1, ..., p $\rightarrow$ p_k, où p_1, ..., p_k ϵ $\mathcal{Q}$ et V = $\{a_1, ..., a_k\}$, alors les expressions régulières qui dénotent les langages L_p, L_{p1}, ..., L_{pk} sont liées par l'équation :

$$X_p = a_1 X_{p1} + ... + a_k X_{pk} \text{ si p n'est pas dans F ;}$$

ou par l'équation :

$$X_p = a_1 X_{p1} + ... + a_k X_{pk} + \varepsilon \text{ si p } \epsilon \text{ F.}$$

Le système d'équations associé à $\mathcal{A}$ est le système obtenu en écrivant les équations correspondant à chaque état p ϵ $\mathcal{Q}$.

Des automates aux expressions régulières

- **Lemme** (Lemme d'Arden)
  Soient A, B deux langages sur V tels que ε n'est pas dans A.
  Alors l'équation

$$X = AX \cup B$$

  admet A*B comme unique solution.
  Preuve :

Minimisation d'un AFD

- Etant donné un automate fini déterministe, on peut toujours construire un automate fini déterministe équivalent contenant un nombre minimum d'états. Ce processus est appelé **minimisation**.

Minimisation d'un AFD

- Algorithme de minimisation d'un automate A
  1. Supprimer tous les états de A inaccessibles à partir de q_0 ;
  2. Rendre l'automate complet ;
  3. Construire le tableau de marquage ;
  4. En déduire l'automate minimal.

Minimisation d'un AFD

Construction du tableau de marquage

1. Construire un tableau de taille N x N (N étant le nombre d'états de l'automate). Chaque case de ce tableau correspond à un couple d'états (p, q) et on garde seulement la partie triangulaire (supérieure ou inférieure) du tableau.
2. Marquer chaque case (p, q) correspondant à un état final et un état non final ($p \in F$ et $q \notin F$ ou $p \notin F$ et $q \in F$).
3. Parcourir toutes les cases vides en les modifiant de la manière suivante :
   - pour chaque case (p, q) et une étiquette a, calculer $\delta(p, a) = p'$ et $\delta(q, a) = q'$;
   - marquer la case (p, q) si et seulement si la case (p' , q') est déjà marquée. Répéter le balayage jusqu'à ce que le tableau ne puisse être modifié.
4. A la fin du balayage, toute case (p, q) non marquée signifie que les états p et q sont équivalents.