Théorie des Langages

Automates

Claude Moulin

Université de Technologie de Compiègne

Printemps 2013

Sommaire

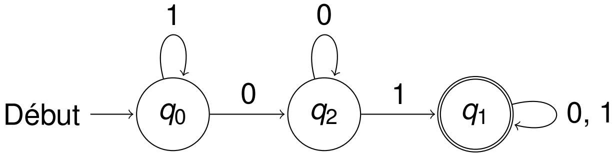
Sommaire

Introduction

- Question : Déterminer des outils permettant de reconnaître si une chaîne appartient à un langage
- Les automates sont des modèles mathématiques qui prennent en entrée une chaîne de symboles et qui effectuent un algorithme de reconnaissance de la chaîne.
- Si l'algorithme se termine dans certaines conditions, l'automate est dit accepter la chaîne correspondante.
- Le langage reconnu par un automate est l'ensemble des chaînes qu'il accepte.

Exemple

- Ensemble des chaînes construites sur $\{0, 1\}$ comportant une suite de caractères 01.
- 3 Cas de figure
  - Le premier 0 n'est pas encore apparu
  - Le premier 0 est apparu et on attend le premier 1 suivant
  - La séquence 01 a été rencontrée



Définition

- Un automate fini déterministe (AFD) est un modèle mathématique qui consiste en :
  - Q un ensemble d'états.
  - Σ un ensemble de symboles d'entrée (alphabet).
  - δ une fonction de transition qui prend comme argument un état et un symbole d'entrée et qui retourne un état.
    $\delta : Q \times \Sigma \longrightarrow Q$.
  - q_0, un état initial appartenant à Q.
  - F, un ensemble d'états finals ou états d'acceptation, $F \subset Q$.
- $A = (Q, \Sigma, \delta, q_0, F)$

Exemple

- Langage construit sur {0,1} dont les chaînes ont un nombre pair de 0 et un nombre pair de 1.

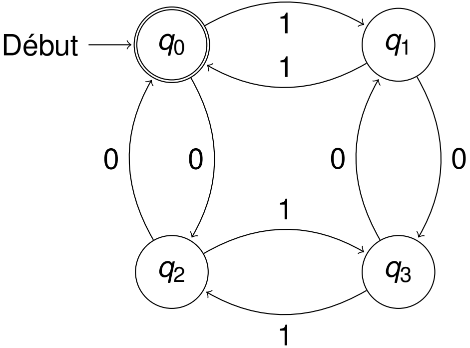
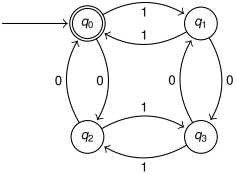
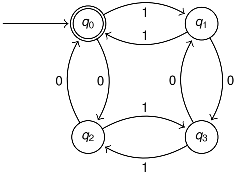


Diagramme de transition



- Pour chaque état $q \in Q$, il existe un nœud étiqueté q.
- Pour chaque état q et chaque symbole a de Σ tel que $\delta(q, a) = p$, il existe un arc du nœud q vers le nœud p étiqueté a.
- Les nœuds correspondant aux états de satisfaction (états appartenant à F) sont représentés par un cercle double.

Table de transition

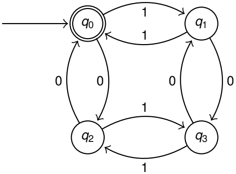


Exemples :
$\delta(q_0, 0) = q_2$
$\delta(q_1, 0) = q_3$

	0	1
$* \rightarrow q_0$	q_2	q_1
q_1	q_3	q_0
q_2	q_0	q_3
q_3	q_1	q_2

$\rightarrow$ désigne l'état initial
* désigne les états finals

Chemins



- $\hat{\delta}(q_0, 0) = q_2$
  car $\delta(q_0, 0) = q_2$
- $\hat{\delta}(q_0, 0101) = q_0$ car
  - $\hat{\delta}(q_0, 010) = q_1$
  - et $\delta(q_1, 1) = q_0$

Fonction de transition étendue

La fonction de transition étendue appelée $\hat{\delta}$ décrit ce qui se passe lorsqu'on se positionne dans un état quelconque de l'automate et que l'on suit une séquence quelconque d'entrées.

- $\hat{\delta} : Q \times \Sigma* \longrightarrow Q$ est définie par induction sur la longueur de la chaîne de symboles d'entrée par :
  - $\hat{\delta}(q, \epsilon) = q$
  - Soit $w = xa$, a le dernier symbole de w et x la chaîne début de w,
  - si $\hat{\delta}(q, x) = p$, alors $\hat{\delta}(q, w) = \delta(p, a) = \delta(\hat{\delta}(q, x), a)$

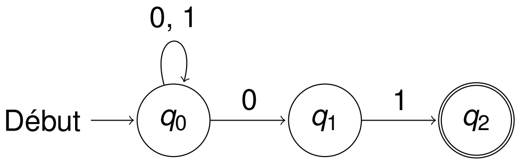
Langage engendré par un automate

Le langage d'un automate à états finis déterministe $A = (Q, \Sigma, \delta, q_0, F)$ est défini par :

- l'ensemble des chaînes obtenues sur tout chemin du graphe partant du nœud initial et s'achevant sur un nœud final.
- $L(A) = \{w \mid \hat{\delta}(q_0, w) \in F\}$

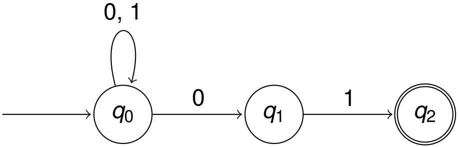
Sommaire

Automate non déterministe



- L'automate non déterministe n'accepte que les chaînes composées de 0 et de 1 et terminées par la séquence 01.
- Il reste en l'état q_0 tant qu'il n'a pas deviné que l'ultime séquence n'a pas commencé.
- $\delta(q_0, 0) = \{q_0, q_1\}$

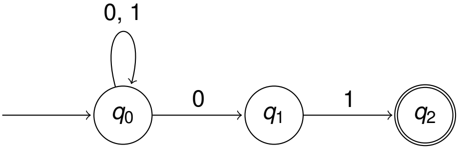
Transition

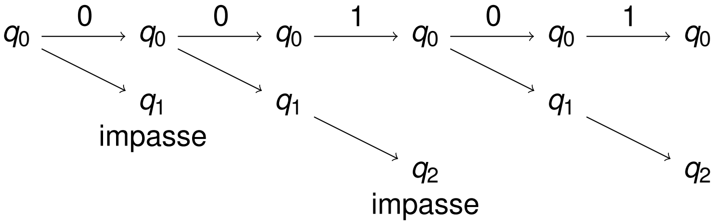


$$A = (\{q_0, q_1, q_2\}, \{0, 1\}, \delta, q_0, \{q_2\})$$

	0	1
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\emptyset$	$\{q_2\}$
$*q_2$	$\emptyset$	$\emptyset$

Reconnaissance de la chaîne 00101



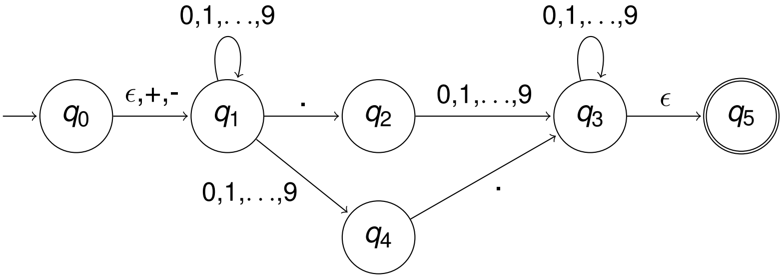


Sommaire

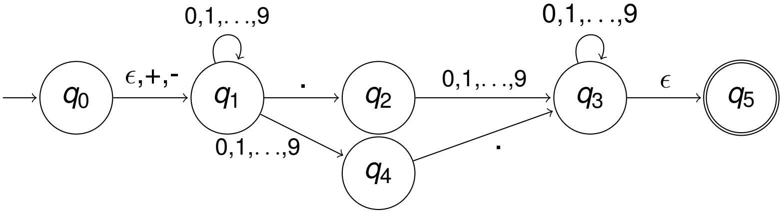
Automate avec ϵ-transitions (ϵ-AFN)

- Les automates finis possédant des ϵ-transitions sont des automates pouvant faire spontanément des transitions.
- Les arcs correspondants sont étiquetés par le symbole ϵ et ne consomment aucun caractère de la chaîne d'entrée.
- Un automate non déterministe avec ϵ-transitions est un quintuplet : $A = (Q, \Sigma, \delta, q_0, F)$
- δ, la fonction de transition, prend en argument un état et un élément de $\Sigma \cup \{\epsilon\}$ et retourne un ensemble d'états.
- $\delta : Q \times \Sigma \cup \{\epsilon\} \longrightarrow Q$.

Automate avec ϵ-transitions (ϵ-AFN)



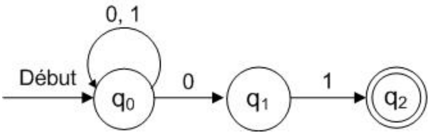
(ϵ-AFN)- Table de transition



	ϵ	+, -	.	0, 1, . . . , 9
$\rightarrow q_0$	$\{q_1\}$	$\{q_1\}$	$\emptyset$	$\emptyset$
q_1	$\emptyset$	$\emptyset$	$\{q_2\}$	$\{q_1, q_4\}$
q_2	$\emptyset$	$\emptyset$	$\emptyset$	$\{q_3\}$
q_3	$\{q_5\}$	$\emptyset$	$\emptyset$	$\{q_3\}$
q_4	$\emptyset$	$\emptyset$	$\{q_3\}$	$\emptyset$
$*q_5$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$

$F = \{q_5\}$

Transformation : AFN $\longrightarrow$ AFD



Question : Déterminer un AFD reconnaissant le même langage que cet AFN ?

Algorithme

1. Soit $E = \{q_0\}$.
2. Construire $E^{(1)}(x) = \delta(E, x) = \bigcup_{q \in E} \delta(q, x)$ pour tout $x \in \Sigma$.
3. Recommencer 2 pour toute transition et pour chaque nouvel ensemble $E^{(i)}(a)$.
4. Tous les ensembles d'états $E^{(i)}(a)$ contenant au moins un état final deviennent des états finals.
5. Renuméroter les ensembles d'états en tant que simples états.

Exemple

état	a	b
$\rightarrow q_0$	q_0, q_2	q_1
q_1	q_3	q_0, q_2
$* \ q_2$	q_3, q_4	q_2
$* \ q_3$	q_2	q_1
q_4	$\emptyset$	q_3

q_0 état initial ; $F = \{q_2, q_3\}$

Table de l'AFD correspondant

état	a	b
q_0	q_0, q_2	q_1
q_0, q_2	q_0, q_2, q_3, q_4	q_1, q_2
q_1	q_3	q_0, q_2
q_0, q_2, q_3, q_4	q_0, q_2, q_3, q_4	q_1, q_2, q_3
q_1, q_2	q_3, q_4	q_0, q_2
q_3	q_2	q_1
q_1, q_2, q_3	q_2, q_3, q_4	q_0, q_1, q_2
q_3, q_4	q_2	q_1, q_3
q_2	q_3, q_4	q_2
q_2, q_3, q_4	q_2, q_3, q_4	q_1, q_2, q_3
q_0, q_1, q_2	q_0, q_2, q_3, q_4	q_0, q_1, q_2
q_1, q_3	q_2, q_3	q_0, q_1, q_2
q_2, q_3	q_2, q_3, q_4	q_1, q_2

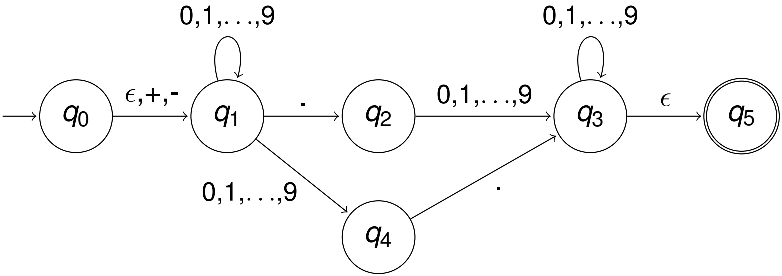
Transformation : ϵ-AFN $\longrightarrow$ AFD

- L'algorithme reste similaire au précédent à la différence que chaque ensemble d'états considéré doit être clos par ϵ-transition.
- L'ϵ-fermeture d'un état q est l'ensemble des états obtenus en suivant des $\epsilon-$transitions à partir de q.

Règles d'appartenance :

- q $\in$ ϵ-fermeture(q)
- si p $\in$ ϵ-fermeture(q) et s'il existe une ϵ-transition entre p et un état r
  alors r $\in$ ϵ-fermeture(q).

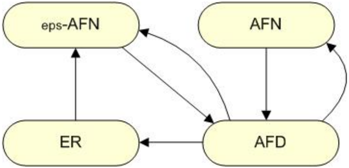
Application



Sommaire

Sommaire

Automate $\leftrightarrow$ Expressions Régulières



AFD $\longrightarrow$ ER
ER $\longrightarrow$ ϵ-AFN
AFD $\longrightarrow$ ϵ-AFN

ϵ-AFN $\longrightarrow$ AFD
AFN $\longrightarrow$ AFD

Théorèmes

- Tout langage défini par un automate est aussi défini par une expression régulière
- Si L(A) est le langage défini par un AFD, A, alors il existe une expression régulière R telle que :

$$L(R) = L(A)$$

- Tout langage défini par une expression régulière est aussi défini par un automate
- Si L(R) est le langage défini par une expression régulière, alors il existe un automate A tel que :

$$L(A) = L(R)$$

Démonstration

- A un AFD ayant n états.
- {1, 2, . . . , n} l'ensemble de ses états.
- $R_{ij}^{(k)}$ le nom de l'expression régulière dont le langage est l'ensemble des chaînes *w* construites sur un chemin allant de l'état i à l'état j qui ne passe pas par un état dont le rang est supérieur à k.

k = 0

- 2 possibilités :
  - $i = j$
  - $i \neq j$
- si $i = j$ on ne considère que le nœud i lui-même
- Chemins légaux :
  - le chemin de longueur 0 étiqueté ϵ
  - les boucles sur l'état i.
- $R_{ii}^{(0)} = \epsilon$: pas de transition
- $R_{ii}^{(0)} = \epsilon \mid a$: 1 transition étiquetée a
- $R_{ii}^{(0)} = \epsilon \mid a_1 \mid a_2 \mid \ldots \mid a_p$: p transitions

k = 0 ; $i \neq j$

- Arcs reliant le nœud *i* au nœud *j*.
  - $R_{ij}^{(0)} = \emptyset$: pas de transition
  - $R_{ij}^{(0)} = a$: 1 transition étiquetée *a*
  - $R_{ij}^{(0)} = a_1 \mid a_2 \mid \ldots \mid a_p$: p transitions étiquetées de a_1 à a_p

Récurrence : k > 0

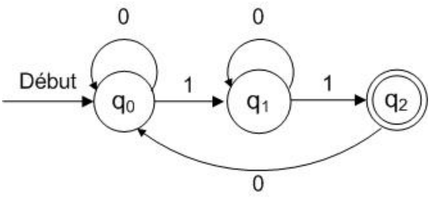
- Chemins allant du nœud *i* au nœud *j* passant par des nœuds dont l'indice est $\leq$ k.
- Un chemin ne passe pas par le nœud k
  - Son étiquette est décrite par $R_{ij}^{(k-1)}$
- Un chemin passe par le nœud *k* au moins une fois, repasse éventuellement par ce même nœud puis termine sur le nœud *j*.
  - son étiquette satisfait l'expression régulière :
  - $R_{ik}^{(k-1)}(R_{kk}^{(k-1)}) * R_{kj}^{(k-1)}$

Conclusion

- $R_{ij}^{(k)} = R_{ij}^{(k-1)} \mid R_{ik}^{(k-1)}(R_{kk}^{(k-1)}) * R_{kj}^{(k-1)}$
- $R_{ij}^{(k)}$ ne dépend que d'expressions de degré inférieur à k
- L'expression régulière décrivant un AFD est donnée par :
  - $R_{1p_1}^{(n)} \mid R_{1p_2}^{(n)} \mid \ldots \mid R_{1p_m}^{(n)}$
  - les nœuds $p_1, p_2, \ldots, p_m$ sont les nœuds d'acceptation.

Sommaire

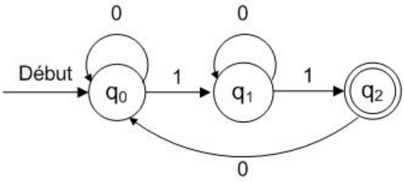
Elimination d'états



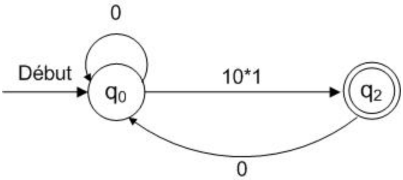
- Elimination de l'état q_1
- Le chemin q_0q_2 doit être remplacé par un arc d'étiquette : 10*1
- On obtient un graphe dont les arcs ont pour étiquette des expressions régulières

Résultat

- A partir de :



- On obtient le graphe :



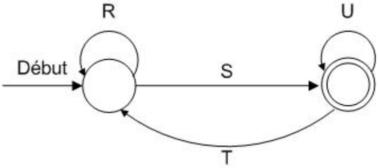
Expression régulière : (0 | 10*10)* 10*1

Méthode

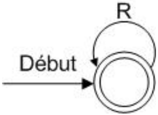
- Chaque chemin q, s, p produit une expression régulière qui vient s'ajouter (par la réunion) à l'expression du chemin q, p.
- Pour chaque état d'acceptation, on applique la réduction précédente pour produire un automate équivalent.
- On élimine tous les états s qui ne sont ni un état d'acceptation ni l'état initial.

Fin de la méthode

- En fin de méthode, on obtient :



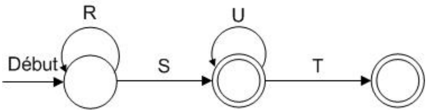
(R | SU*T)* SU*



R*

Exemple

- Situation : l'expression du chemin *q*, *p*.



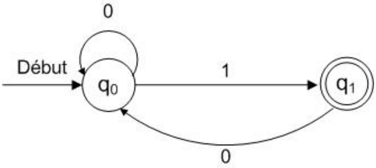
- R*SU* | R*SU*T

Sommaire

Autre méthode

- On appelle L_i le langage que reconnaîtrait l'automate si q_i était son état initial.
- On établit un système d'équation liant tous les L_i
- Pour chaque transition telle que $\delta(q_i, a) = q_j$: $L_{i,a} = aL_j$
- $L_i = L_{i,a_1} \mid L_{i,a_2} \ldots \mid L_{i,a_n}$
- $L_i = a_1 L_{j_1} \mid a_2 L_{j_2} \ldots \mid a_n L_{j_n}$
- Si q_i est terminal sans transition : $L_i = \epsilon$
- si $\delta(q_i, a) = q_i$ alors $L_i = aL_i = a*$
- $L = \alpha L \mid \beta$ se simplifie en : $L = \alpha * \beta$
- Il s'agit de calculer L_0

Exemple



- Equations
  - $L_0 = 0L_0 \mid 1L_1$ et $L_1 = 0L_0 \mid \epsilon$
- Résolution :
  - $L_0 = 0L_0 \mid 10L_0 \mid 1$
  - $L_0 = (0|10)L_0 \mid 1$
  - $L_0 = (0|10) * 1$

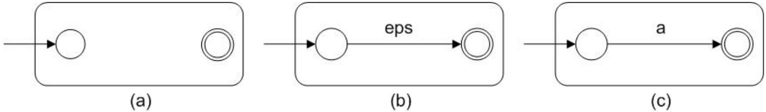
Sommaire

ER $\longrightarrow$ ϵ-*AFN*

- La preuve est construite par récurrence à partir des expressions régulières de base
- Tous les automates construits sont des ϵ-*AFN* ayant un unique état d'acceptation
- Il faut combiner les automates pour construire des automates composites acceptant les opérations :
  - Union (|)
  - Concaténation
  - Fermeture (*)
- Si L(R) est le langage dénoté par une expression régulière R, on démontre qu'il existe un ϵ-*AFN*, A, tel que L(A) = L(R)

ϵ-*AFN* pour des expressions de base

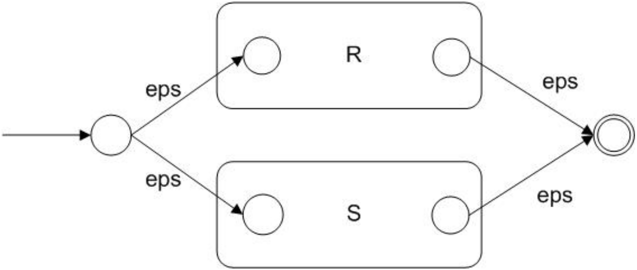
- Automate construit à partir de : $\emptyset$, ϵ, un symbole



- a) - L = $\emptyset$
- b) - L = $\{\epsilon\}$
- c) - L = $\{a\}$

R | S $\longrightarrow$ ϵ-*AFN*

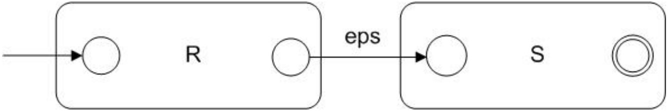
- Automate construit à partir de la disjonction



- $L = L(R) \cup L(S)$

RS $\longrightarrow$ ϵ-*AFN*

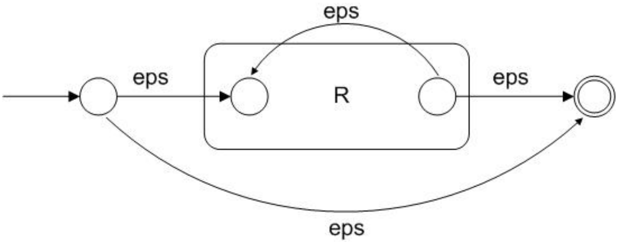
- Automate construit à partir de la concaténation



- $L = L(R)L(S)$

R* $\longrightarrow$ ϵ-AFN

- Automate construit à partir de l'expression R*



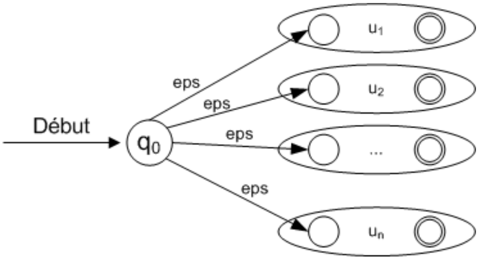
- $L = L(R)*$

Langages non réguliers

- Le langage : $L_{01} = \{0^n 1^n \mid n \geq 1\}$ est non régulier
- Raisonnement par l'absurde : soit un AFD représentant ce langage.
  - Il existe deux nombres différents i et j tels que après avoir lu les préfixes 0^i et 0^j, l'AFD soit dans le même état.
  - A partir de cet état, l'AFD lit un certain nombre de 1 et arrive dans un état final.
  - On aurait $0^i 1^p \in L_{01}$ et $0^j 1^p \in L_{01}$ avec $i \neq j$
- Les expressions régulières ne sont pas suffisantes pour représenter les langages de programmation

Analyse lexicale

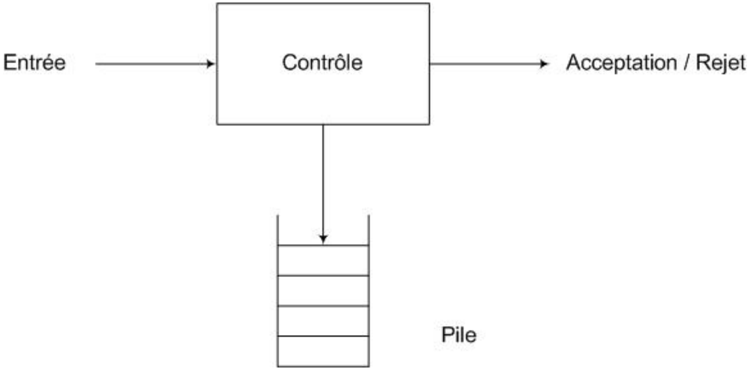
- Les unités lexicales sont définies par des expressions régulières.
- A chaque unité lexicale $u_1, u_2, \ldots u_n$ est associé un AFN.
- Un AFN général est bâti à partir de ces AFN de base par réunion.



Sommaire

Sommaire

Principe



Description

$L_{wcw^R} = \{wcw^R; w \in (0|1)*\}$

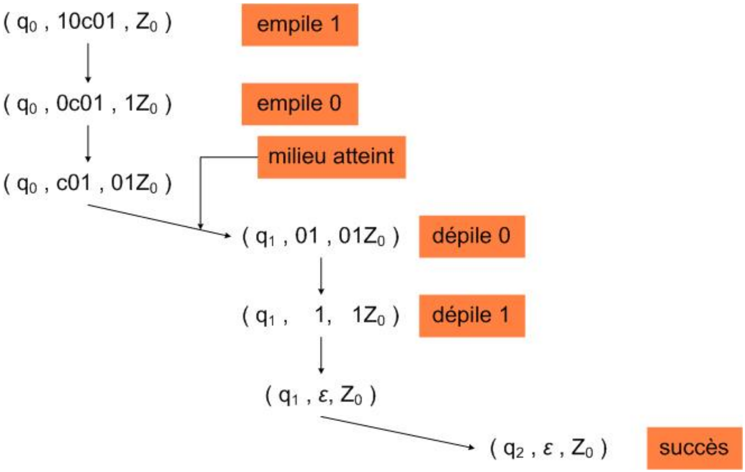
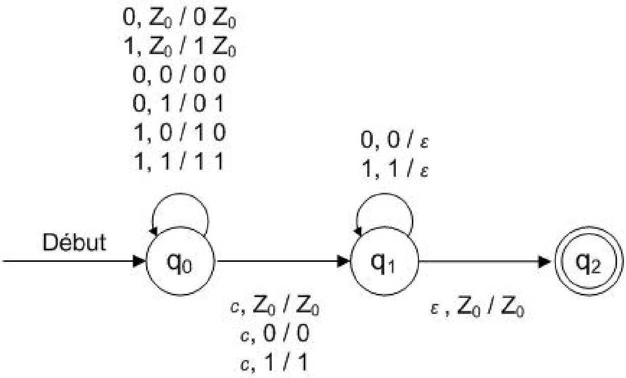


Diagramme de transition



Définition

$P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$

- Q est un ensemble fini d'états
- Σ est un ensemble fini de symboles d'entrée
- Γ est un ensemble fini dit alphabet de pile. C'est l'ensemble des symboles qui peuvent être empilés
- δ est la fonction de transition.
- q_0 est l'état initial
- Z_0 est le symbole de départ de la pile
- F est l'ensemble des états finals

Fonction de transition

- δ, la fonction de transition prend en entrée trois paramètres :
  - q un état de Q
  - a un symbole d'entrée ou ϵ
  - X un symbole de pile
- Elle renvoie un ensemble fini de couples (p, γ) où p est le nouvel état et γ est la nouvelle chaîne de symboles qui remplace X au sommet de la pile.

Règle de transition

Lors d'une transition l'automate :

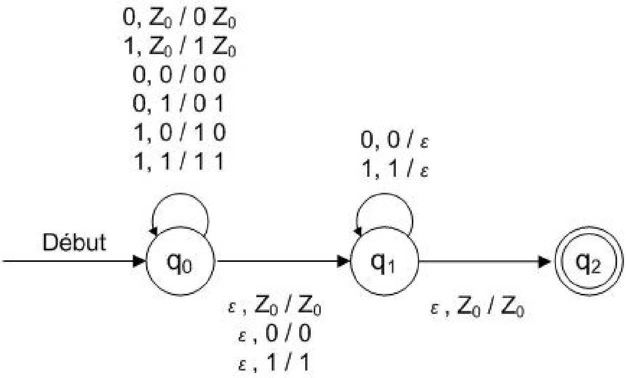
- consume le symbole utilisé dans la transition. Si ϵ est utilisé aucun symbole d'entrée n'est consumé.
- passe dans un nouvel état.
- remplace le symbole au sommet de la pile par une chaîne de symboles :
  - ϵ ; dépilement (pop).
  - le même symbole que le sommet actuel ; aucun changement dans la pile.
  - un autre symbole (pop du sommet suivi de push).
  - une suite de symboles ; dépilement du sommet (pop) et empilement (push) des symboles.

Sommaire

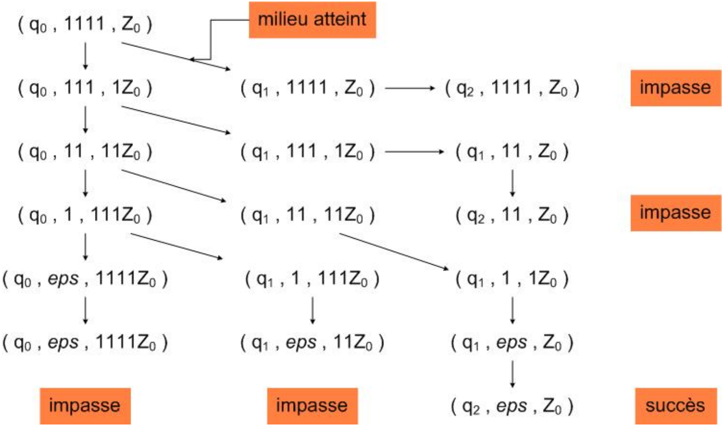
Automate déterministe

- Un automate à pile est déterministe s'il n'existe pas d'alternative de déplacement dans aucune situation.
- $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ est déterministe, si et seulement si :
  - $\delta(q, a, X)$ contient au plus un élément, pour tout état $q \in Q$, $a \in \Sigma$ ou $a = \epsilon$ et $X \in \Gamma$.
  - si $\delta(q, a, X)$ est non vide pour un élément $a \in \Sigma$, alors $\delta(q, \epsilon, X)$ doit être vide.

$L_{ww^R} = \{ww^R; w \in (0|1)*\}$



Simulation : chaîne 1111



Transition

- Une configuration d'un AFP est un triplet (q, v, γ) :
  - q est l'état de la configuration.
  - v est la chaîne d'entrée restant à analyser.
  - γ est le contenu de la pile.
- $(q, aw, X\beta) \vdash (p, w, \alpha\beta)$
  - le symbole d'entrée a est consumé.
  - l'automate passe de l'état q à l'état p.
  - X au sommet de la pile est remplacé par α.
- Notation $\overset{*}{\vdash}$ pour un ou plusieurs pas de l'automate.
- $I \overset{*}{\vdash} I$
- si $I \overset{*}{\vdash} K$ et $K \vdash J$ alors $I \overset{*}{\vdash} J$

Acceptation

- $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$
- Le langage accepté par P par état final est :
  - $L(P) = \{w \mid (q_0, w, Z_0) \overset{*}{\underset{P}{\vdash}} (q, \epsilon, \alpha)\}$
    - q est un état final
    - Le buffer d'entrée est vide
    - α une suite quelconque de symboles.
- Le langage accepté par P par pile vide est :
  - $N(P) = \{w \mid (q_0, w, Z_0) \overset{*}{\underset{P}{\vdash}} (q, \epsilon, \epsilon)\}$
    - le buffer d'entrée est vide
    - q est un état quelconque

Equivalence

- Si $L = N(P_N)$ pour $P_N = (Q, \Sigma, \Gamma, \delta_N, q_0, Z_0, F)$
  - alors il existe un automate à pile P_F tel que :
  - $L = L(P_F)$.
- Si $L = L(P_F)$ pour $P_F = (Q, \Sigma, \Gamma, \delta_F, q_0, Z_0, F)$
  - alors il existe un automate à pile P_N tel que :
  - $L = N(P_N)$.
- Soit G, une grammaire hors contexte. il est possible de construire un automate à pile P tel que $N(P) = L(G)$.
- Soit $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ un AFP. Il existe une grammaire hors contexte G telle que $L(G) = N(P)$.

Equivalence