Automates et langages : Fiche

Sommaire

---

1. Introduction

1.1. Les trois niveaux d'analyse

- **Analyse lexicale** : reconnaitre les éléments en entrée et les catégoriser
- **Analyse syntaxique** : vérifier que le code est conforme aux règles définies
- **Analyse sémantique** : donner du sens

1.2. Alphabet

> Alphabet (Σ) : ensemble non vide dont les éléments sont appelés lettres ou symbole.

- Exemple : $\Sigma_1 = \{a, b, c\}$
- Un mot u constitué de n symboles : $|u| = n$
- Ensemble de tous les mots définis sur Σ : Σ^*

- Σ^+ = Σ^* \ $\{\epsilon\}$: ensemble de tous les mots non vides sur Σ
- Σ^i : ensemble de tous les mots de longueur i

1.3. Concaténation

Soient deux mots $u = x_1\ x_2 ...\ x_p$ et $v = y_1\ y_2 ...\ y_q$, on appelle produit ou concaténation de u et c, noté uv ou u.v le mot $x_1\ x_2 ...\ x_p\ y_1\ y_2 ...\ y_q$

- $u.\epsilon = u$

---

2. Langage

> Un langage L sur un alphabet Σ est un sous ensemble quelconque de Σ^* $(L \subseteq \Sigma^*)$, un ensemble de mots définis sur un même aplhabet

- $L_1 \cup L_2$: tous les éléments des deux ensembles
- $L_1 \cap L_2$: éléments communs des deux ensembles
- Puissance :
  - $L^0 = \{\epsilon\}$
  - $L^1 = L$
  - $L^2 = L.L$
  - $L^{n+1} = L.L^n$
- Etoile :
  A compléter

2.1 Langage régulier

> Un langage est dit régulier ssi on peut le construire à partir d'un nombre fini d'application d'opérations régulières à partir de langages finis.

- Langages finis :
  - $\emptyset, \epsilon$
  - $x \in \Sigma$
  - Soient $a, b \in \Sigma$:
    - $a + b = (a + b)$: a ou b
    - $a.b = ab = (ab)$: a et b
    - a^* : N'importe quel mot qui est une puissance de a.
      - $(a + b)^* = \{\epsilon, ab, abbaa, aaaaaa, bbbb, ...\}$
      - $(a.b)^* = \{\epsilon, ab, abab, ababab, ...\}$
    - $(ab + b)(ab + b)$ = $abab$ ou abb ou bab ou bb

2.2. Exercice

- Sur l'alphabet $\Sigma = \{a, b\}$:

Langage	Mot	Appartient ?
a^*b	b	oui $(a^0.b = b)$
a^*b	ϵ	non, pas de puissance de b
a^*b	ab	oui $(a^0.b = ab)$
a^*b	$aaab$	oui $(a^3.b = aaab)$
ab^*	b	non, pas de puissance de a
ab^*	ab	oui $(ab^1 = ab)$
ab^*	$abab$	non, pas de puissance de a
ab^*	$abbbb$	oui $(a.b^4 = abbbb)$
$(a+b)^*$	$abbbb$	oui $(a^1.b^4 = abbbb)$
$(a+b)^*$	ϵ	oui $(a^0.b^0 = \epsilon)$
$(a+b)^*$	$bbabaabb$	oui $(b^2.a.b.a^2.b^2 = bbabaabb)$

- Remarque : $(a+b)^* = \Sigma^*$

Langage	Combinaisons
b^*	$\{\epsilon, b, bb, bbb, ...\}$
$(a+\epsilon)^*$	$\{\epsilon, a, aa, aaa, ...\} = a^*$
$(ab)^*a$	$\{a, aba, ababa, ...\}$
$(aba)^*$	$\{\epsilon, aba, abaaba, ...\}$
$a(a+b)^*a$	$\{aa, aaa, aba, aaba, aaaaaaa, ...\}$

---

3. Automates

3.1. Reconnaître les mots d'un langage

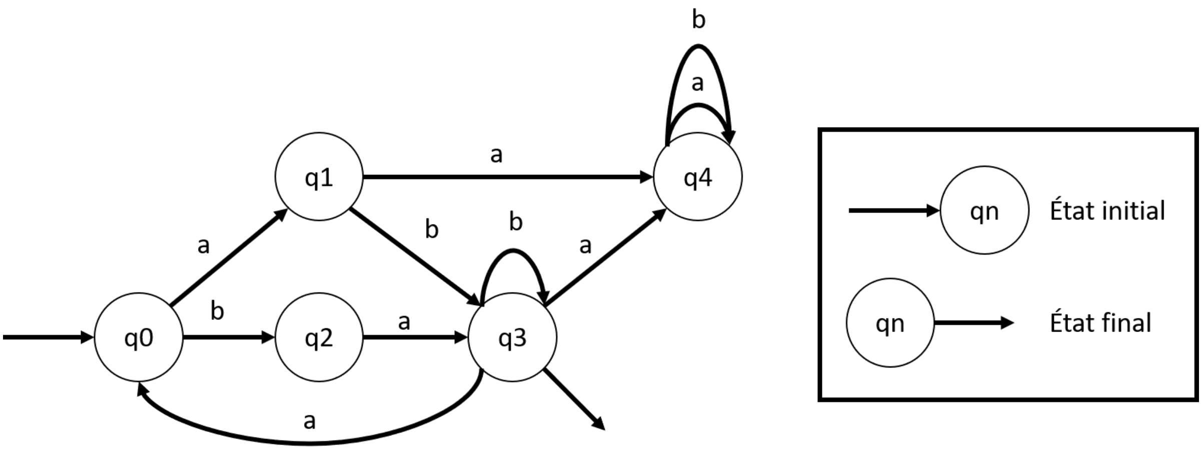
- Idée : créer un programme cpable de reconnaître les mots du langage
- Formalisme utilisé : les automates

> Un automate à états finis (AF) est défini par A = $< \Sigma, Q, I, F, \Delta >$

- Σ : alphabet fini de l'automate (ex : $\{A, B\}$)
- Q : ensemble d'états (ex : $\{q_0, q_1, q_2, q_3, q_4\}$)
- I : ensemble des états initiaux (ex : $\{q_0\}$)

- F : ensemble des états finaux (ex : $\{q_3\}$)
- Δ : relation de transition (ex : δ)

3.2. Représentation graphique



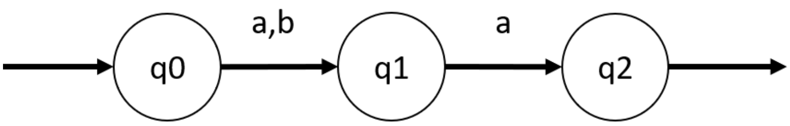
3.3. Reconnaissance des mots

- Les mots sont reconnus s'ils arrivent à l'état final. Exemple :
  - $ba : q_0 \rightarrow^b q_2 \rightarrow^a q_3$
  - $babb : q_0 \rightarrow^b q_2 \rightarrow^a q_3 \rightarrow^b q_3 \rightarrow^b q_3$
- Mots non reconnus :
- $aa : q_0 \rightarrow^a q_1 \rightarrow^a q_4$: q_4 n'est pas un état final

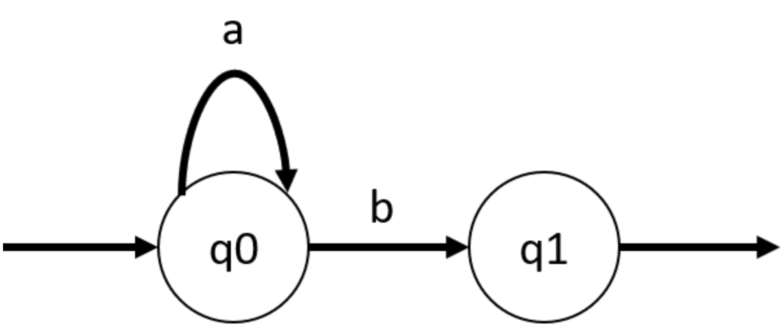
Un même langage peut être reconnu par des automates différents

3.4 De l'expression régulière à l'automate

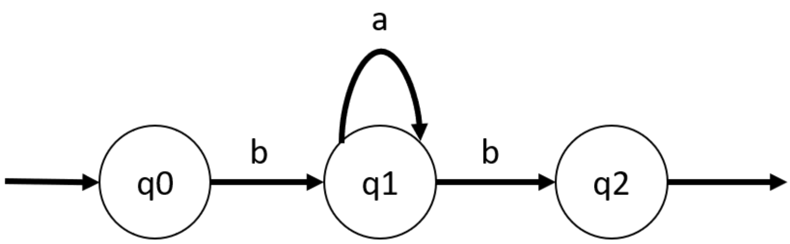
$(a+b)a$:



a^*b :



ba^*b :



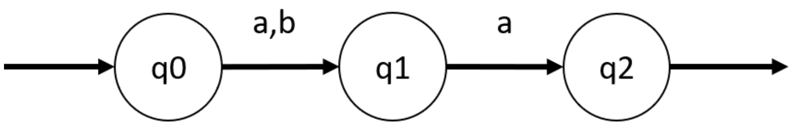
3.5 De l'automate à l'expression régulière

Lemme d'Arden :

- Soient 3 langages K, L et X tels que $\epsilon \notin K$
  Si $X = KX + L$ alors $X = K^*L$

Exemple :

$(a+b)a$:

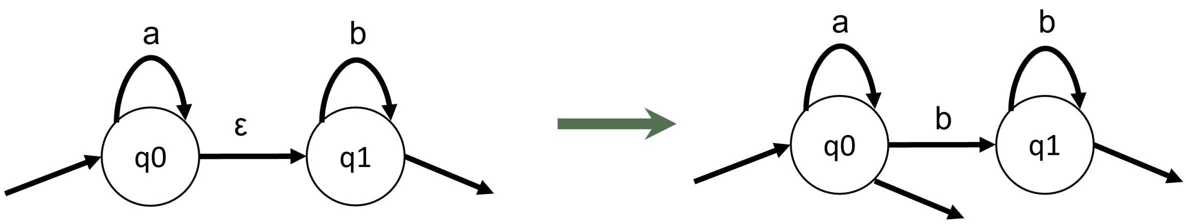


- Expression régulière :
  - $L = L_1$
  - $L_1 = aL_2 + bL_2$
  - $L_2 = aL_3$
  - $L_3 = \epsilon$

---

4. Automates asynchrones

- Contiennent au moins un mot vide
- Pour tout automate asynchrone, il existe un automate sans ϵ-transition équivalent :



Suppression des transitions vides :
$L = a^*b^*$

$\epsilon.C(q_0) = \{q_0, q_1\}$
$\epsilon.C(q_1) = \{q_1\}$

$F = \{q_1\}$
$Fr = \{q_0, q_1\}$

$q_i = \frac{q_0}{q_j = q_1}$

Ajouter (q_0, b, q_1) à Δ

4.1. Automate complet

Méthode pour construire un automate complet déterministe A' équivalent à A:

1. Emonder l'automate

- **Automate émondé** : chaque état est **accessible** et **co-accessible**
  - q est **accessible** : on peut atteindre l'état q à partir de l'état initial
  - q est **co-accessible** : on peut atteindre l'état final à partir de l'état q
- On garde uniquement les états (et les transitions correspondantes) qui sont accessibles et co-accessibles

2. Déterminiser l'automate

- **Automate déterministe** : possède un seul état initial et la relation de transition est telle que :
  - D'un état donné, il existe au plus une seule transition partant de cet état avec une lettre donnée
  - Il n'existe pas d'ϵ-transition

5. Grammaire

5.1. Méthodes de définition d'un langage

- En **extension** : on donne un ensemble de mots du langage
- Par **expression régulière** : on donne une expression régulière qui reconnaît le langage
- Par une **propriété** : on donne une propriété qui caractérise le langage
- En utilisant des **notations ensemblistes** : on donne une définition ensembliste du langage
- Par une **grammaire** : on donne une grammaire qui reconnaît le langage

5.2. Définition d'une grammaire

> Une grammaire est définie par G = $< \Sigma, V, S, R >$

- Σ : alphabet de la grammaire (ex : $\{a, b\}$)
- V : ensemble fini des variables (ex : $\{S, A, B\}$)

- S : variable initiale, **racine** ou **axiome** (ex : S)
- R : ensemble fini des règles de production (ex : $\{S \to aA, A \to bB, B \to \epsilon\}$)

5.3. Règle

- **Règle de production** : $A \to \alpha$
  - A : variable de la grammaire
  - α : mot formé à partir de Σ et de V

5.4. Exemple

$G_1 =< \Sigma, V, S, R >$

- $\sigma = \{a, b\}$
- $V = \{S, T\}$
- $R = \{S \to a, S \to T, T \to bT, T \to \epsilon\}$

Règles :

- $S \to a|T$ (a ou T)
- $T \to bT|\epsilon$

Donc on peut écrire :

$L_1 = L(G_1) = \{u \in \Sigma^*|S \to^* u\} = \{a, \epsilon, b, bb, bbb, ...\}$

5.5 Dérivation

- **Dérivation** : on remplace une variable par une règle de production

5.6. Exemple

$G =< \Sigma, V, S, R >$

- $\sigma = \{a, b\}$
- $V = \{S, T\}$
- $R = \{S \to abST, S \to \epsilon, T \to Ta, T \to \epsilon\}$

Dérivation de S :

$S \to abST \to ababSTT \to ababTT \to ababTaT \to ababaT \to ababa$