Théorie des Langages 1

Cours 2 : Opérations sur les langages, automates finis

L. Rieg

Grenoble INP - Ensimag, 1re année

Année 2023-2024

Opérations sur les langages

Définition

Soient L et M deux langages sur V (L et $M \subseteq V^*$).
Par analogie avec les opérations sur les mots, on définit :

$$L \cup M \stackrel{\text{def}}{=}$$

$$L.M \stackrel{\text{def}}{=}$$

$$\forall i > 0,\ L^i \stackrel{\text{def}}{=}$$

$$L^0 \stackrel{\text{def}}{=}$$

$$L^* \stackrel{\text{def}}{=} \qquad \text{et} \qquad L^+ \stackrel{\text{def}}{=}$$

Opérations sur les langages

Définition

Soient L et M deux langages sur V (L et $M \subseteq V^*$).
Par analogie avec les opérations sur les mots, on définit :

$$L \cup M \stackrel{\text{def}}{=} \{w \in V^* \mid w \in L \text{ ou } w \in M\}$$

$$L.M \stackrel{\text{def}}{=}$$

$$\forall i > 0,\ L^i \stackrel{\text{def}}{=}$$

$$L^0 \stackrel{\text{def}}{=}$$

$$L^* \stackrel{\text{def}}{=} \qquad \text{et} \qquad L^+ \stackrel{\text{def}}{=}$$

Opérations sur les langages

Définition

Soient L et M deux langages sur V (L et $M \subseteq V^*$).
Par analogie avec les opérations sur les mots, on définit :

$$
\begin{aligned}
L \cup M &\stackrel{\text{def}}{=} \{w \in V^* \mid w \in L \text{ ou } w \in M\} \\
L.M &\stackrel{\text{def}}{=} \{w_1 w_2 \mid w_1 \in L \text{ et } w_2 \in M\} \\
\forall i > 0,\ L^i &\stackrel{\text{def}}{=} \\
L^0 &\stackrel{\text{def}}{=} \\
L^* &\stackrel{\text{def}}{=} \qquad\qquad \text{et} \qquad L^+ \stackrel{\text{def}}{=}
\end{aligned}
$$

Opérations sur les langages

Définition

Soient L et M deux langages sur V (L et $M \subseteq V^*$).
Par analogie avec les opérations sur les mots, on définit :

$$L \cup M \stackrel{\text{def}}{=} \{w \in V^* \mid w \in L \text{ ou } w \in M\}$$

$$L.M \stackrel{\text{def}}{=} \{w_1 w_2 \mid w_1 \in L \text{ et } w_2 \in M\}$$

$$\forall i > 0,\ L^i \stackrel{\text{def}}{=} L.L^{i-1} \qquad \text{concaténation de } i \text{ mots de } L$$

$$L^0 \stackrel{\text{def}}{=}$$

$$L^* \stackrel{\text{def}}{=} \qquad \text{et} \qquad L^+ \stackrel{\text{def}}{=}$$

Opérations sur les langages

Définition

Soient L et M deux langages sur V (L et $M \subseteq V^*$).
Par analogie avec les opérations sur les mots, on définit :

$$
\begin{aligned}
L \cup M &\stackrel{\text{def}}{=} \{w \in V^* \mid w \in L \text{ ou } w \in M\} \\
L.M &\stackrel{\text{def}}{=} \{w_1 w_2 \mid w_1 \in L \text{ et } w_2 \in M\} \\
\forall i > 0,\ L^i &\stackrel{\text{def}}{=} L.L^{i-1} \qquad\qquad \text{concaténation de } i \text{ mots de } L \\
L^0 &\stackrel{\text{def}}{=} \{\varepsilon\} \\
L^* &\stackrel{\text{def}}{=} \qquad\qquad \text{et} \qquad L^+ \stackrel{\text{def}}{=}
\end{aligned}
$$

Opérations sur les langages

Définition

Soient L et M deux langages sur V (L et $M \subseteq V^*$).
Par analogie avec les opérations sur les mots, on définit :

$$
\begin{aligned}
L \cup M &\stackrel{\text{def}}{=} \{w \in V^* \mid w \in L \text{ ou } w \in M\} \\
L.M &\stackrel{\text{def}}{=} \{w_1 w_2 \mid w_1 \in L \text{ et } w_2 \in M\} \\
\forall i > 0,\ L^i &\stackrel{\text{def}}{=} L.L^{i-1} \qquad \text{concaténation de } i \text{ mots de } L \\
L^0 &\stackrel{\text{def}}{=} \{\varepsilon\} \\
L^* &\stackrel{\text{def}}{=} \bigcup_{i \geq 0} L^i \qquad \text{et} \qquad L^+ \stackrel{\text{def}}{=}
\end{aligned}
$$

Opérations sur les langages

Définition

Soient L et M deux langages sur V (L et $M \subseteq V^*$).
Par analogie avec les opérations sur les mots, on définit :

$$
\begin{aligned}
L \cup M &\stackrel{\text{def}}{=} \{w \in V^* \mid w \in L \text{ ou } w \in M\} \\
L.M &\stackrel{\text{def}}{=} \{w_1 w_2 \mid w_1 \in L \text{ et } w_2 \in M\} \\
\forall i > 0,\ L^i &\stackrel{\text{def}}{=} L.L^{i-1} \qquad\qquad \text{concaténation de } i \text{ mots de } L \\
L^0 &\stackrel{\text{def}}{=} \{\varepsilon\} \\
L^* &\stackrel{\text{def}}{=} \bigcup_{i \geq 0} L^i \qquad \text{et} \qquad L^+ \stackrel{\text{def}}{=} \bigcup_{i > 0} L^i
\end{aligned}
$$

Opérations sur les langages

Définition

Soient L et M deux langages sur V (L et $M \subseteq V^*$).
Par analogie avec les opérations sur les mots, on définit :

$$\begin{aligned}
L \cup M &\stackrel{\text{def}}{=} \{w \in V^* \mid w \in L \text{ ou } w \in M\} \\
L.M &\stackrel{\text{def}}{=} \{w_1 w_2 \mid w_1 \in L \text{ et } w_2 \in M\} \\
\forall i > 0,\ L^i &\stackrel{\text{def}}{=} L.L^{i-1} \qquad \text{concaténation de } i \text{ mots de } L \\
L^0 &\stackrel{\text{def}}{=} \{\varepsilon\} \\
L^* &\stackrel{\text{def}}{=} \bigcup_{i \geq 0} L^i \qquad \text{et} \qquad L^+ \stackrel{\text{def}}{=} \bigcup_{i > 0} L^i
\end{aligned}$$

Notation : on pourra noter LM au lieu de $L.M$.

Exemples

Exemple

Soient $L = \{ab, cd\}$ et $M = \{ab, bba\}$. Alors

$$L \cup M = \{ab, cd, bba\} \text{ et } LM = \{abab, abbba, cdab, cdbba\}$$

Exemples

Exemple

Soient $L = \{ab, cd\}$ et $M = \{ab, bba\}$. Alors

$$L \cup M = \{ab, cd, bba\} \text{ et } LM = \{abab, abbba, cdab, cdbba\}$$

Exemple

Soient $L = \{a^n b^n \mid n \geq 0\}$ et $M = \left\{c^n d^{2n} \;\middle|\; n \geq 0\right\}$. Alors

$$\begin{aligned} LM &= \\ L^* &= \end{aligned}$$

Exemples

Exemple

Soient $L = \{ab, cd\}$ et $M = \{ab, bba\}$. Alors

$$L \cup M = \{ab, cd, bba\} \text{ et } LM = \{abab, abbba, cdab, cdbba\}$$

Exemple

Soient $L = \{a^n b^n \mid n \geq 0\}$ et $M = \left\{c^n d^{2n} \mid n \geq 0\right\}$. Alors

$$\begin{aligned} LM &= \left\{a^n b^n c^p d^{2p} \mid n, p \geq 0\right\} \\ L^* &= \end{aligned}$$

Exemples

Exemple

Soient $L = \{ab, cd\}$ et $M = \{ab, bba\}$. Alors

$$L \cup M = \{ab, cd, bba\} \text{ et } LM = \{abab, abbba, cdab, cdbba\}$$

Exemple

Soient $L = \{a^n b^n \mid n \geq 0\}$ et $M = \left\{c^n d^{2n} \mid n \geq 0\right\}$. Alors

$$\begin{aligned} LM &= \left\{a^n b^n c^p d^{2p} \mid n, p \geq 0\right\} \\ L^* &= \{a^{n_1} b^{n_1} a^{n_2} b^{n_2} \cdots a^{n_k} b^{n_k} \mid k \geq 0 \wedge n_1, \ldots, n_k \geq 0\} \end{aligned}$$

Exemples

Exemple

Soient $L = \{ab, cd\}$ et $M = \{ab, bba\}$. Alors

$$L \cup M = \{ab, cd, bba\} \text{ et } LM = \{abab, abbba, cdab, cdbba\}$$

Exemple

Soient $L = \{a^n b^n \mid n \geq 0\}$ et $M = \left\{c^n d^{2n} \;\middle|\; n \geq 0\right\}$. Alors

$$\begin{aligned} LM &= \left\{a^n b^n c^p d^{2p} \;\middle|\; n, p \geq 0\right\} \\ L^* &= \{a^{n_1} b^{n_1} a^{n_2} b^{n_2} \cdots a^{n_k} b^{n_k} \mid k \geq 0 \wedge n_1, \ldots, n_k \geq 0\} \end{aligned}$$

Question

Si L est un langage, peut-on avoir $\varepsilon \in L^+$?

Exemples

Exemple

Soient $L = \{ab, cd\}$ et $M = \{ab, bba\}$. Alors

$$L \cup M = \{ab, cd, bba\} \text{ et } LM = \{abab, abbba, cdab, cdbba\}$$

Exemple

Soient $L = \{a^n b^n \mid n \geq 0\}$ et $M = \left\{c^n d^{2n} \mid n \geq 0\right\}$. Alors

$$\begin{aligned} LM &= \left\{a^n b^n c^p d^{2p} \mid n, p \geq 0\right\} \\ L^* &= \{a^{n_1} b^{n_1} a^{n_2} b^{n_2} \cdots a^{n_k} b^{n_k} \mid k \geq 0 \wedge n_1, \ldots, n_k \geq 0\} \end{aligned}$$

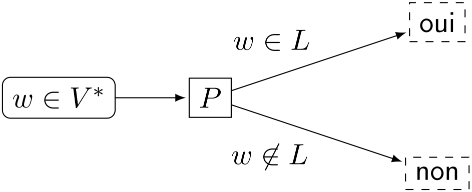
Question

Si L est un langage, peut-on avoir $\varepsilon \in L^+$? Oui, ssi $\varepsilon \in L$

Exemple : $L = \{\varepsilon, a\}$, $L^2 = \{\varepsilon, a, aa\}$, $L^+ = \{a^n \mid n \geq 0\} = L^*$

Les automates finis

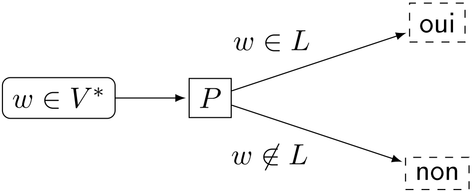
On s'intéresse à définir des « programmes » qui reconnaissent des langages.



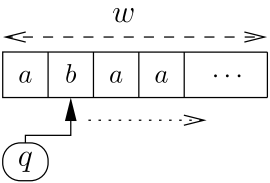
Les programmes les plus « simples » sont les automates finis.

Les automates finis

On s'intéresse à définir des « programmes » qui reconnaissent des langages.



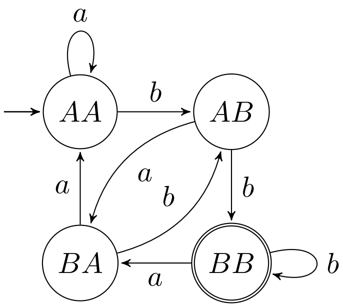
Les programmes les plus « simples » sont les automates finis.



À chaque pas d'exécution, l'automate peut changer d'état et/ou lire un symbole et se positionner sur le symbole suivant.

Exemple d'automate fini

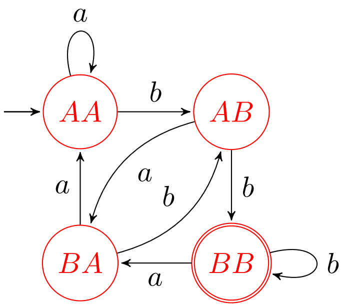
Voici un automate fini.
Il contient :



Exemple d'automate fini

Voici un automate fini.
Il contient :

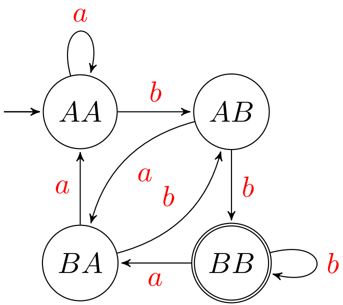
- des états



Exemple d'automate fini

Voici un automate fini.
Il contient :

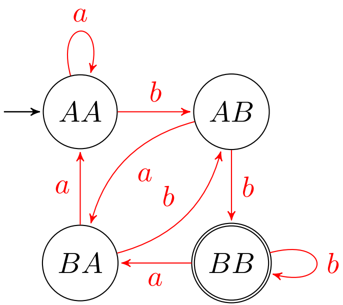
- des états
- le vocabulaire



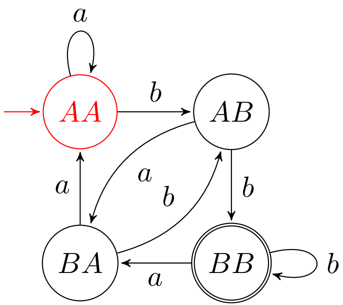
Exemple d'automate fini

Voici un automate fini.
Il contient :

- des états
- le vocabulaire
- des transitions



Exemple d'automate fini



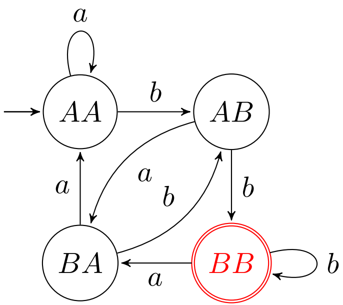
Voici un automate fini.
Il contient :

- des états
- le vocabulaire
- des transitions
- des états initiaux

Exemple d'automate fini

Voici un automate fini.
Il contient :

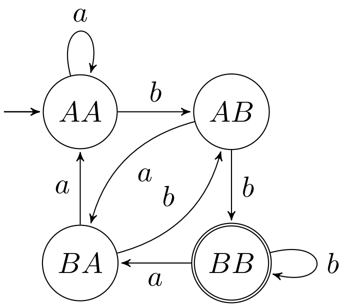
- des états
- le vocabulaire
- des transitions
- des états initiaux
- des états acceptants



Exemple d'automate fini

Voici un automate fini.
Il contient :

- des états
- le vocabulaire
- des transitions
- des états initiaux
- des états acceptants



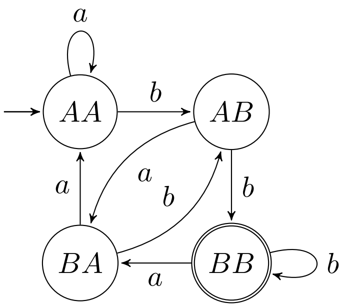
Utilisation d'un automate :

1. commencer dans un état initial
2. lire les lettres d'un mot en suivant les transitions
3. à la fin du mot, regarder si l'état est acceptant

Exemple d'automate fini

Voici un automate fini.
Il contient :

- des états
- le vocabulaire
- des transitions
- des états initiaux
- des états acceptants



$\mathcal{L}(\mathcal{A}) = V^*.\{bb\}$

Utilisation d'un automate :

1. commencer dans un état initial
2. lire les lettres d'un mot en suivant les transitions
3. à la fin du mot, regarder si l'état est acceptant

Définition formelle

Définition

Un automate fini (AF) est un quintuplet $\langle Q, V, \delta, I, F \rangle$, où :

- Q est un ensemble fini d'états
- V est le vocabulaire d'entrée
- $\delta \subseteq Q \times (V \cup \{\varepsilon\}) \times Q$ est la relation de transition
- $I \subseteq Q$ est l'ensemble des états initiaux
- $F \subseteq Q$ est l'ensemble des états acceptants (ou finaux ou finals)

Définition formelle

Définition

Un automate fini (AF) est un quintuplet $\langle Q, V, \delta, I, F \rangle$, où :

- Q est un ensemble fini d'états
- V est le vocabulaire d'entrée
- $\delta \subseteq Q \times (V \cup \{\varepsilon\}) \times Q$ est la relation de transition
- $I \subseteq Q$ est l'ensemble des états initiaux
- $F \subseteq Q$ est l'ensemble des états acceptants (ou finaux ou finals)

Relation de transition

- Pour $a \in V$, si $(p, a, q) \in \delta$, alors étant dans l'état p et lisant un a, l'automate peut passer dans l'état q et avancer dans le mot.
- Si $(p, \varepsilon, q) \in \delta$, alors étant dans l'état p, l'automate peut passer à l'état q sans avancer dans le mot.

Exemple et représentation graphique

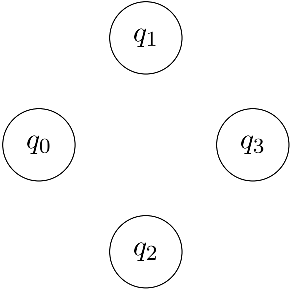
Soit A l'automate défini par :

$$\begin{aligned}
Q &= \{q_0, q_1, q_2, q_3\}\,,\; V = \{a, b\}\,,\\
\delta &= \{(q_0, b, q_1), (q_0, a, q_2), (q_1, a, q_1), (q_1, a, q_3), (q_2, b, q_1), (q_2, \varepsilon, q_3)\}\\
I &= \{q_0, q_2\}\,,\; F = \{q_2, q_3\}
\end{aligned}$$

Exemple et représentation graphique

Soit A l'automate défini par :

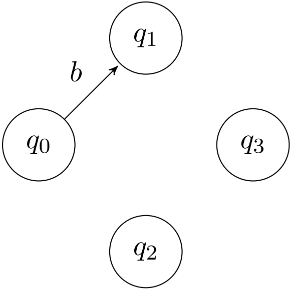
$$
\begin{aligned}
Q &= \{q_0, q_1, q_2, q_3\}\,,\ V = \{a, b\}\,, \\
\delta &= \{(q_0, b, q_1), (q_0, a, q_2), (q_1, a, q_1), (q_1, a, q_3), (q_2, b, q_1), (q_2, \varepsilon, q_3)\} \\
I &= \{q_0, q_2\}\,,\ F = \{q_2, q_3\}
\end{aligned}
$$



Exemple et représentation graphique

Soit A l'automate défini par :

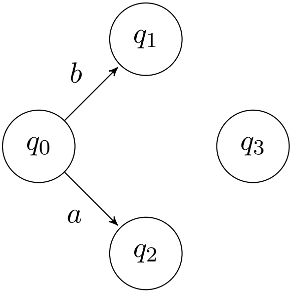
$$
\begin{array}{rcl}
Q & = & \{q_0, q_1, q_2, q_3\}\,,\ V \ = \ \{a, b\}\,, \\
\delta & = & \{(q_0, b, q_1), (q_0, a, q_2), (q_1, a, q_1), (q_1, a, q_3), (q_2, b, q_1), (q_2, \varepsilon, q_3)\} \\
I & = & \{q_0, q_2\}\,,\ F \ = \ \{q_2, q_3\}
\end{array}
$$



Exemple et représentation graphique

Soit A l'automate défini par :

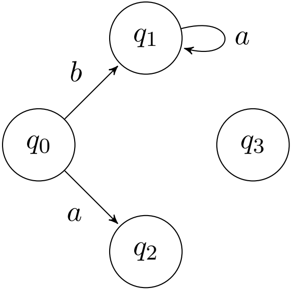
$$
\begin{aligned}
Q &= \{q_0, q_1, q_2, q_3\}\,,\ V = \{a, b\}\,,\\
\delta &= \{(q_0, b, q_1), {\color{red}(q_0, a, q_2)}, (q_1, a, q_1), (q_1, a, q_3), (q_2, b, q_1), (q_2, \varepsilon, q_3)\}\\
I &= \{q_0, q_2\}\,,\ F = \{q_2, q_3\}
\end{aligned}
$$



Exemple et représentation graphique

Soit A l'automate défini par :

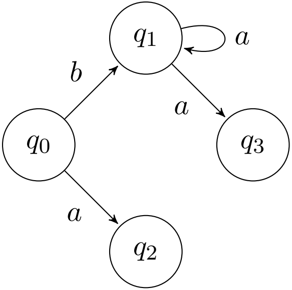
$$\begin{aligned} Q &= \{q_0, q_1, q_2, q_3\}\,,\ V = \{a, b\}\,, \\ \delta &= \{(q_0, b, q_1), (q_0, a, q_2), {\color{red}(q_1, a, q_1)}, (q_1, a, q_3), (q_2, b, q_1), (q_2, \varepsilon, q_3)\} \\ I &= \{q_0, q_2\}\,,\ F = \{q_2, q_3\} \end{aligned}$$



Exemple et représentation graphique

Soit A l'automate défini par :

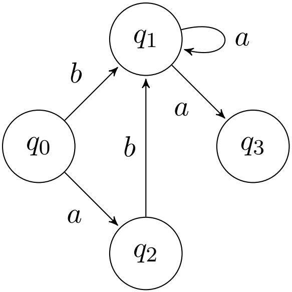
$$\begin{aligned} Q &= \{q_0, q_1, q_2, q_3\}\,,\ V \ = \ \{a, b\}\,, \\ \delta &= \{(q_0, b, q_1), (q_0, a, q_2), (q_1, a, q_1), \color{red}{(q_1, a, q_3)}, (q_2, b, q_1), (q_2, \varepsilon, q_3)\} \\ I &= \{q_0, q_2\}\,,\ F \ = \ \{q_2, q_3\} \end{aligned}$$



Exemple et représentation graphique

Soit A l'automate défini par :

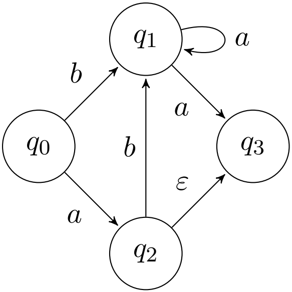
$$\begin{aligned} Q &= \{q_0, q_1, q_2, q_3\}\,,\ V = \{a, b\}\,, \\ \delta &= \{(q_0, b, q_1), (q_0, a, q_2), (q_1, a, q_1), (q_1, a, q_3), \color{red}{(q_2, b, q_1)}, (q_2, \varepsilon, q_3)\} \\ I &= \{q_0, q_2\}\,,\ F = \{q_2, q_3\} \end{aligned}$$



Exemple et représentation graphique

Soit A l'automate défini par :

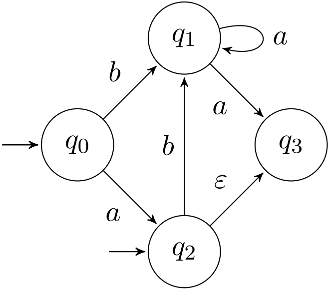
$$\begin{aligned} Q &= \{q_0, q_1, q_2, q_3\}\,,\ V = \{a, b\}\,, \\ \delta &= \{(q_0, b, q_1), (q_0, a, q_2), (q_1, a, q_1), (q_1, a, q_3), (q_2, b, q_1), (q_2, \varepsilon, q_3)\} \\ I &= \{q_0, q_2\}\,,\ F = \{q_2, q_3\} \end{aligned}$$



Exemple et représentation graphique

Soit A l'automate défini par :

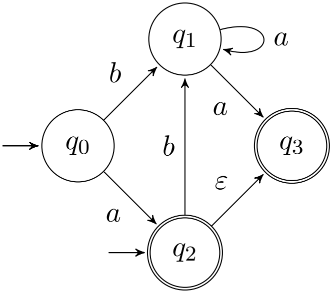
$$\begin{aligned}
Q &= \{q_0, q_1, q_2, q_3\}\,,\ V = \{a, b\}\,,\\
\delta &= \{(q_0, b, q_1), (q_0, a, q_2), (q_1, a, q_1), (q_1, a, q_3), (q_2, b, q_1), (q_2, \varepsilon, q_3)\}\\
I &= \{q_0, q_2\}\,,\ F = \{q_2, q_3\}
\end{aligned}$$



Exemple et représentation graphique

Soit A l'automate défini par :

$$\begin{aligned} Q &= \{q_0, q_1, q_2, q_3\}\,,\; V \;=\; \{a, b\}\,, \\ \delta &= \{(q_0, b, q_1), (q_0, a, q_2), (q_1, a, q_1), (q_1, a, q_3), (q_2, b, q_1), (q_2, \varepsilon, q_3)\} \\ I &= \{q_0, q_2\}\,,\; \color{red}{F \;=\; \{q_2, q_3\}} \end{aligned}$$



Langage reconnu par un automate et chemins

$\mathcal{L}(A)$
$=$ « Ens. des mots permettant de passer d'un état initial à un état final »

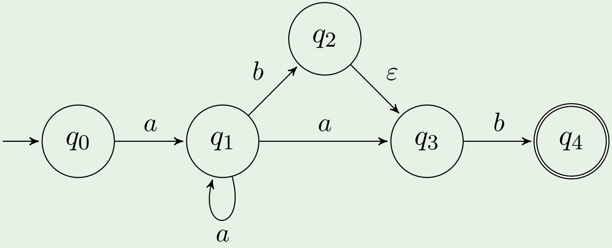
Langage reconnu par un automate et chemins

$\mathcal{L}(A)$
$=$ « Ens. des mots permettant de passer d'un état initial à un état final »
$=$ « Les mots qui étiquettent un chemin d'un état initial à un état final »

Langage reconnu par un automate et chemins

$\mathcal{L}(A)$
$=$ « Ens. des mots permettant de passer d'un état initial à un état final »
$=$ « Les mots qui étiquettent un chemin d'un état initial à un état final »

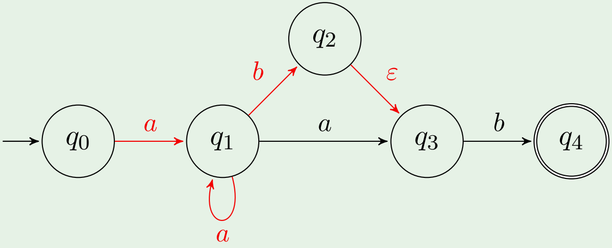
Exemple (Chemins pour aab)



Langage reconnu par un automate et chemins

$\mathcal{L}(A)$
= « Ens. des mots permettant de passer d'un état initial à un état final »
= « Les mots qui étiquettent un chemin d'un état initial à un état final »

Exemple (Chemins pour aab)

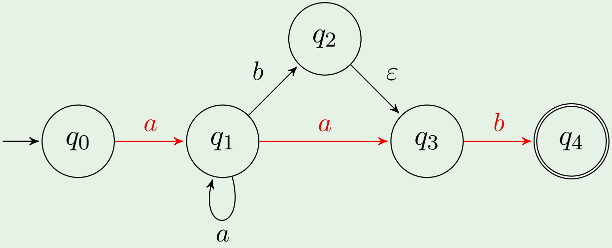


$$\chi_1 \quad = \quad (q_0, a, q_1)(q_1, a, q_1)(q_1, b, q_2)(q_2, \varepsilon, q_3)$$

Langage reconnu par un automate et chemins

$\mathcal{L}(A)$
$=$ « Ens. des mots permettant de passer d'un état initial à un état final »
$=$ « Les mots qui étiquettent un chemin d'un état initial à un état final »

Exemple (Chemins pour aab)



$$\chi_1 = (q_0, a, q_1)(q_1, a, q_1)(q_1, b, q_2)(q_2, \varepsilon, q_3)$$
$$\chi_2 = (q_0, a, q_1)(q_1, a, q_3)(q_3, b, q_4)$$

Langage reconnu par un automate et chemins

$\mathcal{L}(A)$
= « Ens. des mots permettant de passer d'un état initial à un état final »
= « Les mots qui étiquettent un chemin d'un état initial à un état final »

Définition (Chemin)

Soit $A = \langle Q, V, \delta, I, F \rangle$ un automate. L'ensemble des chemins dans A est défini inductivement de la façon suivante :

Base Pour tout $p \in Q$, $()$ est un chemin (vide) dans A de p à p ;

Induction Pout tous $p, q, q' \in Q$ et $a \in V \cup \{\varepsilon\}$,
si $(p, a, q) \in \delta$ et χ est un chemin dans A de q à q',
alors $(p, a, q).\chi$ est un chemin dans A de p à q'.

Convention

Dans un chemin non vide, on ne note en général pas le « () » final.

Trace, longueur d'un chemin

Définition

Soit $(q_0, a_1, q_1)(q_1, a_2, q_2) \cdots (q_{n-1}, a_n, q_n)()$ un chemin dans A.
Ce chemin est de longueur n et de trace $a_1 a_2 \cdots a_n$.

Trace, longueur d'un chemin

Définition

Soit $(q_0, a_1, q_1)(q_1, a_2, q_2) \cdots (q_{n-1}, a_n, q_n)()$ un chemin dans A.
Ce chemin est de longueur n et de trace $a_1 a_2 \cdots a_n$.

Exemples

$\chi_1 = (q_0, a, q_1)(q_1, a, q_1)(q_1, b, q_2)(q_2, \varepsilon, q_3)()$	longueur :	trace :
$\chi_2 = (q_0, a, q_1)(q_1, a, q_3)(q_3, b, q_4)()$	longueur :	trace :

Trace, longueur d'un chemin

Définition

Soit $(q_0, a_1, q_1)(q_1, a_2, q_2) \cdots (q_{n-1}, a_n, q_n)()$ un chemin dans A.
Ce chemin est de longueur n et de trace $a_1 a_2 \cdots a_n$.

Exemples

$\chi_1 = (q_0, a, q_1)(q_1, a, q_1)(q_1, b, q_2)(q_2, \varepsilon, q_3)()$	longueur : 4	trace :
$\chi_2 = (q_0, a, q_1)(q_1, a, q_3)(q_3, b, q_4)()$	longueur :	trace :

Trace, longueur d'un chemin

Définition

Soit $(q_0, a_1, q_1)(q_1, a_2, q_2) \cdots (q_{n-1}, a_n, q_n)()$ un chemin dans A.
Ce chemin est de longueur n et de trace $a_1 a_2 \cdots a_n$.

Exemples

$\chi_1 = (q_0, a, q_1)(q_1, a, q_1)(q_1, b, q_2)(q_2, \varepsilon, q_3)()$	longueur : 4	trace : aab
$\chi_2 = (q_0, a, q_1)(q_1, a, q_3)(q_3, b, q_4)()$	longueur :	trace :

Trace, longueur d'un chemin

Définition

Soit $(q_0, a_1, q_1)(q_1, a_2, q_2) \cdots (q_{n-1}, a_n, q_n)()$ un chemin dans A.
Ce chemin est de longueur n et de trace $a_1 a_2 \cdots a_n$.

Exemples

$\chi_1 = (q_0, a, q_1)(q_1, a, q_1)(q_1, b, q_2)(q_2, \varepsilon, q_3)()$	longueur : 4	trace : aab
$\chi_2 = (q_0, a, q_1)(q_1, a, q_3)(q_3, b, q_4)()$	longueur : 3	trace : aab

Trace, longueur d'un chemin

Définition

Soit $(q_0, a_1, q_1)(q_1, a_2, q_2) \cdots (q_{n-1}, a_n, q_n)()$ un chemin dans A.
Ce chemin est de **longueur** n et de **trace** $a_1 a_2 \cdots a_n$.

$$\mathsf{lgr} : \begin{cases} () & \mapsto \quad 0 \\ (p, a, q)\chi & \mapsto \quad 1 + \mathsf{lgr}(\chi) \end{cases} \qquad \mathsf{tr} : \begin{cases} () & \mapsto \quad \varepsilon \\ (p, a, q)\chi & \mapsto \quad a \cdot \mathsf{tr}(\chi) \end{cases}$$

Exemples

$\chi_1 = (q_0, a, q_1)(q_1, a, q_1)(q_1, b, q_2)(q_2, \varepsilon, q_3)()$ longueur : 4 trace : aab

$\chi_2 = (q_0, a, q_1)(q_1, a, q_3)(q_3, b, q_4)()$ longueur : 3 trace : aab

Trace, longueur d'un chemin

Définition

Soit $(q_0, a_1, q_1)(q_1, a_2, q_2)\cdots(q_{n-1}, a_n, q_n)()$ un chemin dans A.
Ce chemin est de longueur n et de trace $a_1 a_2 \cdots a_n$.

$$\mathsf{lgr} : \begin{cases} () & \mapsto \quad 0 \\ (p,a,q)\chi & \mapsto \quad 1 + \mathsf{lgr}(\chi) \end{cases} \qquad \mathsf{tr} : \begin{cases} () & \mapsto \quad \varepsilon \\ (p,a,q)\chi & \mapsto \quad a \cdot \mathsf{tr}(\chi) \end{cases}$$

Exemples

$\chi_1 = (q_0, a, q_1)(q_1, a, q_1)(q_1, b, q_2)(q_2, \varepsilon, q_3)()$	longueur : 4	trace : aab
$\chi_2 = (q_0, a, q_1)(q_1, a, q_3)(q_3, b, q_4)()$	longueur : 3	trace : aab

Définition (Mot reconnu par un automate)

Un mot w est reconnu par A si et seulement si
il existe un chemin dans A d'un état initial à un état final, de trace w.

Exercices

Exercice 1

Construire un automate fini qui reconnaît le langage

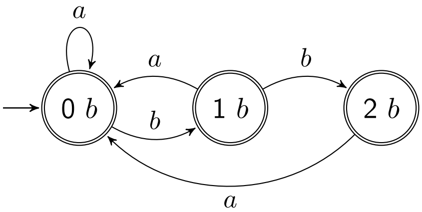
$$L = \{w \in \{a,b\}^* \mid w \text{ ne contient } \textbf{pas plus de} \text{ deux } b \text{ consécutifs}\}$$

Exercices

Exercice 1

Construire un automate fini qui reconnaît le langage

$$L = \{w \in \{a,b\}^* \mid w \text{ ne contient } \textbf{pas plus de} \text{ deux } b \text{ consécutifs}\}$$



Exercices

Exercice 2

Construire un automate fini qui reconnaît le langage

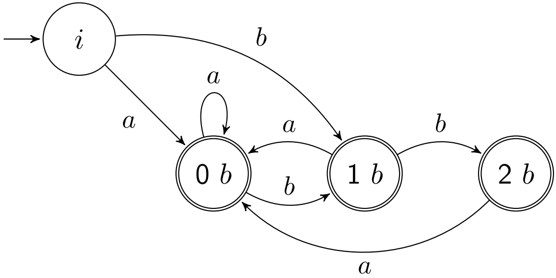
$$L = \left\{ w \in \{a, b\}^+ \;\middle|\; w \text{ ne contient } \textbf{pas plus de} \text{ deux } b \text{ consécutifs} \right\}$$

Exercices

Exercice 2

Construire un automate fini qui reconnaît le langage

$$L = \{w \in \{a, b\}^+ \mid w \text{ ne contient } \textbf{pas plus de} \text{ deux } b \text{ consécutifs}\}$$



Automate non-déterministe

Un AF $\langle Q, V, \delta, I, F\rangle$ est dit non-déterministe si

1. Card$(I) > 1$ (plus d'un état intial), et/ou
2. $\exists\,(q, a, p)$ et $(q, a, r) \in \delta$ avec $p \neq r$, et/ou
3. $\exists\,(q, \varepsilon, p) \in \delta$

Automate non-déterministe

Un AF $\langle Q, V, \delta, I, F \rangle$ est dit non-déterministe si

1. $\text{Card}(I) > 1$ (plus d'un état intial), et/ou
2. $\exists\,(q, a, p)$ et $(q, a, r) \in \delta$ avec $p \neq r$, et/ou
3. $\exists\,(q, \varepsilon, p) \in \delta$

Dans les trois cas, « on ne sait pas quoi faire » :

1. « où dois-je commencer ? »
2. « je suis en q, je vois le symbole a, où vais-je ? »
3. « je suis en q, $\forall$ symbole je peux choisir de passer en p ou non »

Automate non-déterministe

Un AF $\langle Q, V, \delta, I, F \rangle$ est dit non-déterministe si

1. $\text{Card}(I) > 1$ (plus d'un état intial), et/ou
2. $\exists\,(q, a, p)$ et $(q, a, r) \in \delta$ avec $p \neq r$, et/ou
3. $\exists\,(q, \varepsilon, p) \in \delta$

Dans les trois cas, « on ne sait pas quoi faire » :

1. « où dois-je commencer ? »
2. « je suis en q, je vois le symbole a, où vais-je ? »
3. « je suis en q, $\forall$ symbole je peux choisir de passer en p ou non »

Non-déterminisme :

- ne donne pas immédiatement un « programme » reconnaisseur

Automate non-déterministe

Un AF $\langle Q, V, \delta, I, F \rangle$ est dit non-déterministe si

1. $\text{Card}(I) > 1$ (plus d'un état intial), et/ou
2. $\exists\,(q, a, p)$ et $(q, a, r) \in \delta$ avec $p \neq r$, et/ou
3. $\exists\,(q, \varepsilon, p) \in \delta$

Dans les trois cas, « on ne sait pas quoi faire » :

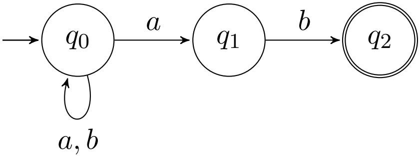
1. « où dois-je commencer ? »
2. « je suis en q, je vois le symbole a, où vais-je ? »
3. « je suis en q, $\forall$ symbole je peux choisir de passer en p ou non »

Non-déterminisme :

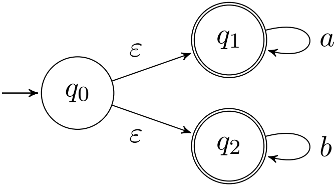
- ne donne pas immédiatement un « programme » reconnaisseur
- mais facilite la définition des automates !

Exemples

1. Non-déterministe, sans ε-transition



2. Non-déterministe, avec ε-transition



Automate déterministe

À l'inverse, un AF $\langle Q, V, \delta, I, F\rangle$ est dit déterministe si

1. Card$(I) = 1$ (exactement un état intial), et
2. Si (q, a, p) et $(q, a, r) \in \delta$, alors $p = r$, et
3. $\nexists\, (q, \varepsilon, p) \in \delta$

Automate déterministe

À l'inverse, un AF $\langle Q, V, \delta, I, F \rangle$ est dit déterministe si

1. Card$(I) = 1$ (exactement un état intial), et
2. Si (q, a, p) et $(q, a, r) \in \delta$, alors $p = r$, et
3. $\nexists\, (q, \varepsilon, p) \in \delta$

Ainsi, « on sait toujours quoi faire » : les transitions possibles sont uniques.
Sera en particulier utilisé en architecture/CEP et en TL2

Automate déterministe

À l'inverse, un AF $\langle Q, V, \delta, I, F \rangle$ est dit déterministe si

1. $\text{Card}(I) = 1$ (exactement un état intial), et
2. Si (q, a, p) et $(q, a, r) \in \delta$, alors $p = r$, et
3. $\nexists\, (q, \varepsilon, p) \in \delta$

Ainsi, « on sait toujours quoi faire » : les transitions possibles sont uniques.
Sera en particulier utilisé en architecture/CEP et en TL2

Conséquences de la définition

- L'automate a un seul état initial
-
-
-

Automate déterministe

À l'inverse, un AF $\langle Q, V, \delta, I, F \rangle$ est dit déterministe si

1. $\mathrm{Card}(I) = 1$ (exactement un état intial), et
2. Si (q, a, p) et $(q, a, r) \in \delta$, alors $p = r$, et
3. $\nexists\, (q, \varepsilon, p) \in \delta$

Ainsi, « on sait toujours quoi faire » : les transitions possibles sont uniques.
Sera en particulier utilisé en architecture/CEP et en TL2

Conséquences de la définition

- L'automate a un seul état initial
- δ est une fonction partielle : $Q \times V \rightharpoonup Q$:
  Si $(p, a, q) \in \delta$, on pourra noter $\delta(p, a) = q$
-
-

Automate déterministe

À l'inverse, un AF $\langle Q, V, \delta, I, F \rangle$ est dit déterministe si

1. $\mathrm{Card}(I) = 1$ (exactement un état intial), et
2. Si (q, a, p) et $(q, a, r) \in \delta$, alors $p = r$, et
3. $\nexists\, (q, \varepsilon, p) \in \delta$

Ainsi, « on sait toujours quoi faire » : les transitions possibles sont uniques.
Sera en particulier utilisé en architecture/CEP et en TL2

Conséquences de la définition

- L'automate a un seul état initial
- δ est une fonction partielle : $Q \times V \rightharpoonup Q$:
  Si $(p, a, q) \in \delta$, on pourra noter $\delta(p, a) = q$
- Donne directement un « programme » reconnaisseur
-

Automate déterministe

À l'inverse, un AF $\langle Q, V, \delta, I, F \rangle$ est dit déterministe si

1. Card$(I) = 1$ (exactement un état intial), et
2. Si (q, a, p) et $(q, a, r) \in \delta$, alors $p = r$, et
3. $\nexists\, (q, \varepsilon, p) \in \delta$

Ainsi, « on sait toujours quoi faire » : les transitions possibles sont uniques.
Sera en particulier utilisé en architecture/CEP et en TL2

Conséquences de la définition

- L'automate a un seul état initial
- δ est une fonction partielle : $Q \times V \rightharpoonup Q$:
  Si $(p, a, q) \in \delta$, on pourra noter $\delta(p, a) = q$
- Donne directement un « programme » reconnaisseur
- Mais certaines transitions peuvent manquer !

Automate complet

Définition

Un automate est complet si de chaque état et chaque symbole, une transition est toujours possible : $\forall (q, a) \in Q \times V, \exists p \in Q, (q, a, p) \in \delta$.

Pour un AF déterministe complet, δ est une fonction totale : $Q \times V \to Q$.

Automate complet

Définition

Un automate est complet si de chaque état et chaque symbole, une transition est toujours possible : $\forall (q, a) \in Q \times V, \exists p \in Q, (q, a, p) \in \delta$.

Pour un AF déterministe complet, δ est une fonction totale : $Q \times V \to Q$.
Un automate peut être non-déterministe mais complet !

Automate complet

Définition

Un automate est complet si de chaque état et chaque symbole, une transition est toujours possible : $\forall (q, a) \in Q \times V, \exists p \in Q, (q, a, p) \in \delta$.

Pour un AF déterministe complet, δ est une fonction totale : $Q \times V \to Q$.
Un automate peut être non-déterministe mais complet !

Comment compléter un automate (sans changer son langage) ?

Automate complet

Définition

Un automate est complet si de chaque état et chaque symbole, une transition est toujours possible : $\forall (q, a) \in Q \times V, \exists p \in Q, (q, a, p) \in \delta$.

Pour un AF déterministe complet, δ est une fonction totale : $Q \times V \to Q$.
Un automate peut être non-déterministe mais complet !

Comment compléter un automate (sans changer son langage) ?

1. Ajouter un état puits : un état non acceptant qui boucle pour tous les symboles
2. Ajouter toutes les transitions manquantes vers cet état.

Automate complet

Définition

Un automate est complet si de chaque état et chaque symbole, une transition est toujours possible : $\forall (q, a) \in Q \times V, \exists p \in Q, (q, a, p) \in \delta$.

Pour un AF déterministe complet, δ est une fonction totale : $Q \times V \to Q$.
Un automate peut être non-déterministe mais complet !

Comment compléter un automate (sans changer son langage) ?

1. Ajouter un état puits : un état non acceptant qui boucle pour tous les symboles
2. Ajouter toutes les transitions manquantes vers cet état.

Questions : Veut-on toujours un automate déterministe complet ?
Est-ce toujours mieux ?

Extension de la fonction de transition aux mots

Définition

Soit $A = \langle Q, V, \delta, I, F \rangle$ un AFD complet. On définit la fonction $\delta^* : Q \times V^* \to Q$ par induction de la façon suivante : pour tout $p \in Q$,

Extension de la fonction de transition aux mots

Définition

Soit $A = \langle Q, V, \delta, I, F \rangle$ un AFD complet. On définit la fonction $\delta^* : Q \times V^* \to Q$ par induction de la façon suivante : pour tout $p \in Q$,

- $\delta^*(p, \varepsilon) = p$

Extension de la fonction de transition aux mots

Définition

Soit $A = \langle Q, V, \delta, I, F \rangle$ un AFD complet. On définit la fonction $\delta^* : Q \times V^* \to Q$ par induction de la façon suivante : pour tout $p \in Q$,

- $\delta^*(p, \varepsilon) = p$
- $\delta^*(p, aw) = \delta^*(\delta(p, a), w)$

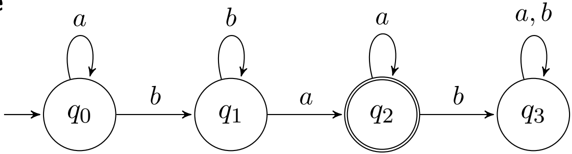
Extension de la fonction de transition aux mots

Définition

Soit $A = \langle Q, V, \delta, I, F \rangle$ un AFD complet. On définit la fonction $\delta^* : Q \times V^* \to Q$ par induction de la façon suivante : pour tout $p \in Q$,

- $\delta^*(p, \varepsilon) = p$
- $\delta^*(p, aw) = \delta^*(\delta(p, a), w)$

Exemple



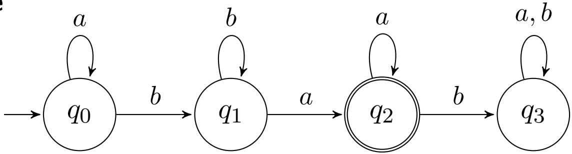
Extension de la fonction de transition aux mots

Définition

Soit $A = \langle Q, V, \delta, I, F \rangle$ un AFD complet. On définit la fonction $\delta^* : Q \times V^* \to Q$ par induction de la façon suivante : pour tout $p \in Q$,

- $\delta^*(p, \varepsilon) = p$
- $\delta^*(p, aw) = \delta^*(\delta(p, a), w)$

Exemple



$\delta^*(q_0, aab) = q_1$ et $\delta^*(q_0, abba) = q_2$

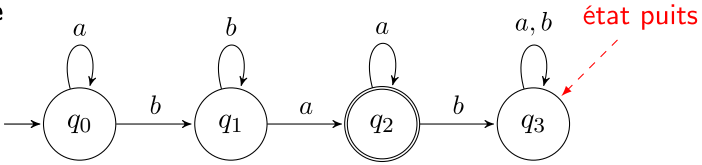
Extension de la fonction de transition aux mots

Définition

Soit $A = \langle Q, V, \delta, I, F \rangle$ un AFD complet. On définit la fonction $\delta^* : Q \times V^* \to Q$ par induction de la façon suivante : pour tout $p \in Q$,

- $\delta^*(p, \varepsilon) = p$
- $\delta^*(p, aw) = \delta^*(\delta(p, a), w)$

Exemple



$\delta^*(q_0, aab) = q_1$ et $\delta^*(q_0, abba) = q_2$

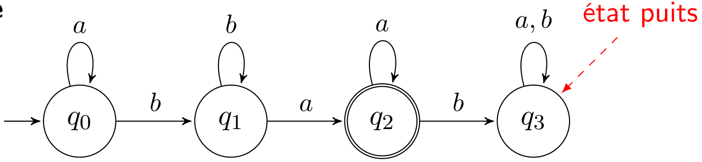
Extension de la fonction de transition aux mots

Définition

Soit $A = \langle Q, V, \delta, I, F \rangle$ un AFD complet. On définit la fonction $\delta^* : Q \times V^* \to Q$ par induction de la façon suivante : pour tout $p \in Q$,

- $\delta^*(p, \varepsilon) = p$
- $\delta^*(p, aw) = \delta^*(\delta(p, a), w)$

Exemple

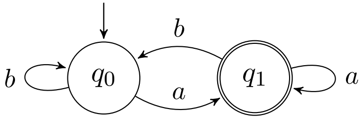


$$\delta^*(q_0, aab) = q_1 \text{ et } \delta^*(q_0, abba) = q_2$$

L'extension de la fonction de transition est parfois notée δ au lieu de δ^*.

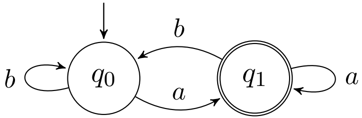
Test d'appartenance pour les AFD complets

$L = \{a, b\}^* \{a\}^+$



Test d'appartenance pour les AFD complets

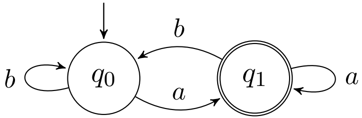
$$L = \{a, b\}^* \{a\}^+$$



$bbaba \in L$

Test d'appartenance pour les AFD complets

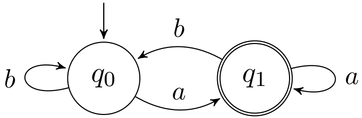
$$L = \{a, b\}^* \{a\}^+$$



$$bbaba \in L \iff \delta^*(q_0, bbaba) \in F$$

Test d'appartenance pour les AFD complets

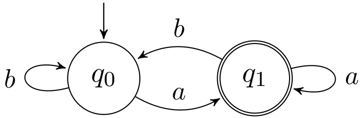
$$L = \{a, b\}^* \{a\}^+$$



$$bbaba \in L \iff \delta^*(q_0, bbaba) \in F \iff \delta^*(q_0, baba) \in F$$

Test d'appartenance pour les AFD complets

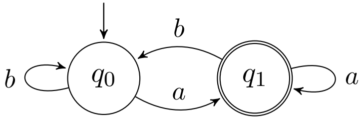
$$L = \{a, b\}^* \{a\}^+$$



$$bbaba \in L \iff \delta^*(q_0, bbaba) \in F \iff \delta^*(q_0, baba) \in F$$
$$\iff \delta^*(q_0, aba) \in F$$

Test d'appartenance pour les AFD complets

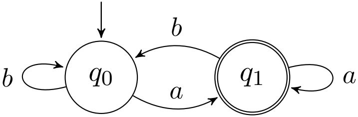
$L = \{a, b\}^* \{a\}^+$



$$\begin{aligned} bbaba \in L &\iff \delta^*(q_0, bbaba) \in F \iff \delta^*(q_0, baba) \in F \\ &\iff \delta^*(q_0, aba) \in F \quad \iff \delta^*(q_1, ba) \in F \end{aligned}$$

Test d'appartenance pour les AFD complets

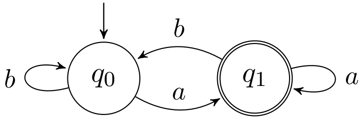
$$L = \{a, b\}^* \{a\}^+$$



$$\begin{aligned} bbaba \in L &\iff \delta^*(q_0, bbaba) \in F &&\iff \delta^*(q_0, baba) \in F \\ &\iff \delta^*(q_0, aba) \in F &&\iff \delta^*(q_1, ba) \in F \\ &\iff \delta^*(q_0, a) \in F \end{aligned}$$

Test d'appartenance pour les AFD complets

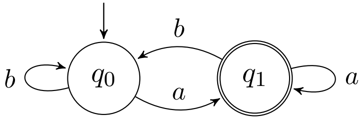
$L = \{a,b\}^* \{a\}^+$



$$\begin{aligned} bbaba \in L &\iff \delta^*(q_0, bbaba) \in F &&\iff \delta^*(q_0, baba) \in F \\ &\iff \delta^*(q_0, aba) \in F &&\iff \delta^*(q_1, ba) \in F \\ &\iff \delta^*(q_0, a) \in F &&\iff \delta^*(q_1, \varepsilon) \in F \end{aligned}$$

Test d'appartenance pour les AFD complets

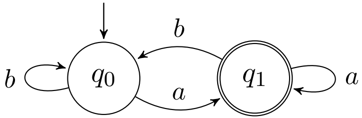
$L = \{a, b\}^* \{a\}^+$



$$
\begin{aligned}
bbaba \in L &\iff \delta^*(q_0, bbaba) \in F \iff \delta^*(q_0, baba) \in F \\
&\iff \delta^*(q_0, aba) \in F \quad \iff \delta^*(q_1, ba) \in F \\
&\iff \delta^*(q_0, a) \in F \qquad \iff \delta^*(q_1, \varepsilon) \in F \iff q_1 \in F
\end{aligned}
$$

Test d'appartenance pour les AFD complets

$L = \{a,b\}^* \{a\}^+$



$$\begin{aligned} bbaba \in L &\iff \delta^*(q_0, bbaba) \in F \iff \delta^*(q_0, baba) \in F \\ &\iff \delta^*(q_0, aba) \in F \iff \delta^*(q_1, ba) \in F \\ &\iff \delta^*(q_0, a) \in F \iff \delta^*(q_1, \varepsilon) \in F \iff q_1 \in F \end{aligned}$$

fonction reconnaître(q : état, w : mot) **renvoie Booléen** =

  tant que $w \neq \varepsilon$ **faire**

    $s \leftarrow$ premier_symbole(w)

    $w \leftarrow$ reste_mot(w)

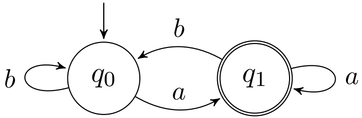
    $q \leftarrow \delta(q, s)$

  fin tant que

  renvoyer $(q \in F)$

Test d'appartenance pour les AFD complets

$$L = \{a, b\}^* \{a\}^+$$



$$\begin{aligned} bbaba \in L &\iff \delta^*(q_0, bbaba) \in F \iff \delta^*(q_0, baba) \in F \\ &\iff \delta^*(q_0, aba) \in F \iff \delta^*(q_1, ba) \in F \\ &\iff \delta^*(q_0, a) \in F \iff \delta^*(q_1, \varepsilon) \in F \iff q_1 \in F \end{aligned}$$

fonction reconnaître(q : état, w : mot) **renvoie Booléen** =
    tant que $w \neq \varepsilon$ **faire**
        $s \leftarrow$ premier_symbole(w)
        $w \leftarrow$ reste_mot(w)
        $q \leftarrow \delta(q, s)$
    fin tant que
    renvoyer $(q \in F)$

$\forall w \in V^*$, $w \in \mathcal{L}(A)$ si et seulement si reconnaître$(q_0, w) =$ **vrai**

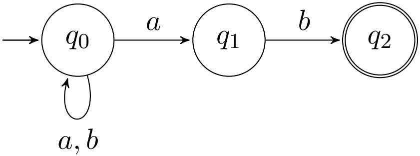
Automates équivalents

Deux automates A et A' sont équivalents ssi $\mathcal{L}(A) = \mathcal{L}(A')$.

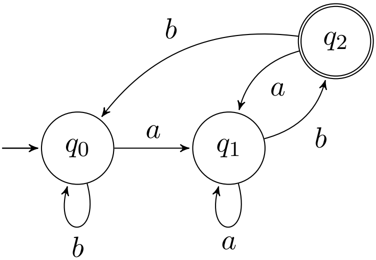
Automates équivalents

Deux automates A et A' sont équivalents ssi $\mathcal{L}(A) = \mathcal{L}(A')$.

- Automate A :



- Automate A' :



Langages réguliers – États accessibles

Définition

On appelle langage régulier tout langage reconnu par un automate fini.

Langages réguliers – États accessibles

Définition

On appelle langage régulier tout langage reconnu par un automate fini.

Définition

Soit $A = \langle Q, V, \delta, I, F \rangle$ un automate.
Un état $p \in Q$ est accessible si on peut passer d'un état $q_0 \in I$ à p en se servant des transitions de δ.
Un automate est initialement connecté si tous ses états sont accessibles.

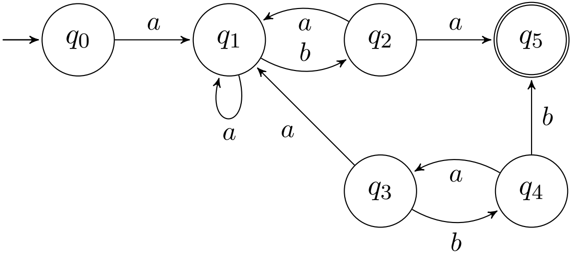
Langages réguliers – États accessibles

Définition

On appelle langage régulier tout langage reconnu par un automate fini.

Définition

Soit $A = \langle Q, V, \delta, I, F \rangle$ un automate.
Un état $p \in Q$ est accessible si on peut passer d'un état $q_0 \in I$ à p en se servant des transitions de δ.
Un automate est initialement connecté si tous ses états sont accessibles.



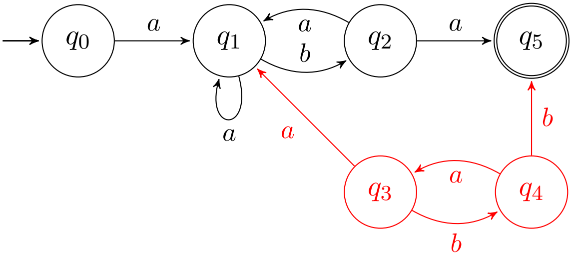
Langages réguliers – États accessibles

Définition

On appelle langage régulier tout langage reconnu par un automate fini.

Définition

Soit $A = \langle Q, V, \delta, I, F \rangle$ un automate.
Un état $p \in Q$ est accessible si on peut passer d'un état $q_0 \in I$ à p en se servant des transitions de δ.
Un automate est initialement connecté si tous ses états sont accessibles.



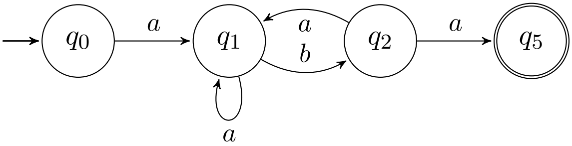
Langages réguliers – États accessibles

Définition

On appelle langage régulier tout langage reconnu par un automate fini.

Définition

Soit $A = \langle Q, V, \delta, I, F \rangle$ un automate.
Un état $p \in Q$ est accessible si on peut passer d'un état $q_0 \in I$ à p en se servant des transitions de δ.
Un automate est initialement connecté si tous ses états sont accessibles.



Propriétés

Définitions équivalentes

Soit un AFD complet $A = \langle Q, V, \delta, \{q_0\}, F\rangle$

- Langage reconnu par A

Propriétés

Définitions équivalentes

Soit un AFD complet $A = \langle Q, V, \delta, \{q_0\}, F \rangle$

- Langage reconnu par A

$$\mathcal{L}(A) \;=\; \{w \in V^* \mid \delta^*(q_0, w) \in F\}$$

Propriétés

Définitions équivalentes

Soit un AFD complet $A = \langle Q, V, \delta, \{q_0\}, F \rangle$

- Langage reconnu par A

$$\mathcal{L}(A) \ = \ \{w \in V^* \mid \delta^*(q_0, w) \in F\}$$

- L'automate A est initialement connecté si et seulement si

Propriétés

Définitions équivalentes

Soit un AFD complet $A = \langle Q, V, \delta, \{q_0\}, F \rangle$

- Langage reconnu par A

$$\mathcal{L}(A) \;=\; \{w \in V^* \mid \delta^*(q_0, w) \in F\}$$

- L'automate A est initialement connecté si et seulement si

$$\forall p \in Q,\; \exists w \in V^*,\, \delta^*(q_0, w) = p$$