Automates finis

UE LU2IN005 – Mathématiques discrètes

Nathalie Sznajder

SCIENCES SORBONNE UNIVERSITÉ

Rappels sur les langages (1/2)

- Un *alphabet* est un ensemble fini (et non vide) de symboles.
- Un *mot* sur un alphabet A est une séquence finie de symboles de A.

Exemple

- Sur l'alphabet $A = \{a, b, c, \ldots, z\}$, on peut définir les mots $u = bonjour$, $v = soleil$, $w = rekltjvuis$,...
- Sur l'alphabet $A = \{0, 1\}$, on peut définir les mots $x = 001$, $y = 0101$, $z = 11$.

- La séquence vide, ne contenant aucun symbole de l'alphabet, est un mot appelé le mot vide, et noté ε.
- L'ensemble de tous les mots sur l'alphabet A est un ensemble infini noté A^* (qui contient *toujours* ε).

Rappels sur les langages (2/2)

- Un *langage* L sur A, aussi appelé un langage de A^*, est un sous-ensemble de A^* :$L \subseteq A^*$.

Exemple

Sur l'alphabet $A = \{a, b, c, \ldots, z\}$, l'ensemble L_1 des mots du dictionnaire de langue française est un langage. Sur l'alphabet $A = \{0, 1\}$, l'ensemble L_2 des mots terminant par 0 est un langage.

Remarque: Le langage L_1 est un ensemble fini, et le langage L_2 est un ensemble *infini*.

Opérations sur les langages

Soient L_1 et L_2 deux langages sur un alphabet A.

- Les langages étant des ensembles, on peut utiliser les définitions ensemblistes connues : $L_1 \cup L_2$, $L_1 \cap L_2$, $\overline{L_1}$, $L_1 \setminus L_2$.
- On définit aussi le *langage produit* (ou *concaténation*) par $L_1 \cdot L_2 = \{w \in A^* \mid w = u \cdot v, u \in L_1, v \in L_2\}$.
- On définit l'*étoile* d'un langage L. Tout d'abord, on définit la suite de langages suivantes : $L^0 = \{\varepsilon\}$, puis, pour tout $n \geq 1$, $L^{n+1} = L \cdot L^n$. L'étoile de L est donnée par $L^* = \bigcup_{n \geq 0} L^n$. On définit également $L^+ = \bigcup_{n \geq 1} L^n$.

Exemple

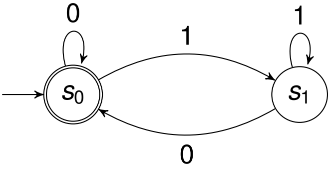
Si $L_1 = \{a, ab\}$ et $L_2 = \{c, bc\}$, alors
$L_1 \cdot L_2 = \{ac, abc, abbc\} = \{ac, abc, abbc\}$. $L_2^2 = \{aa, aab, aba, abab\}$.

Automates et langages

- Pour un problème de *décision*, les mots sont une façon de représenter les *données* du problème. Un langage, étant un ensemble de mots, permet de représenter les *solutions* de ce problème. Ainsi, à tout problème de décision P, on peut associer sur un certain alphabet, le langage L_P des solutions de P.
- Le problème du mot : étant donné un langage $L \subseteq A^*$ et un mot $u \in A^*$, est-ce que $u \in L$?
- Un *automate* est un modèle de programme simple permettant de résoudre le problème du mot.
- Un automate comporte des états (en nombre *fini* !), reçoit en entrée un mot qu'il lit lettre à lettre, et change d'état en fonction de ces entrées. À la fin de son exécution, l'état dans lequel se trouve l'automate détermine si le mot lu en entrée appartient au langage recherché ou non.

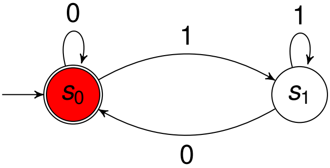
Automates et langages - Exemple

Sur l'alphabet $A = \{0, 1\}$, on peut représenter l'ensemble des nombres écrits en base 2. Le langage $L_{pair} = \{$l'ensemble des nombres pairs$\}$ est donc constitué de l'ensemble des mots de A^* se terminant par 0.



Automates et langages - Exemple

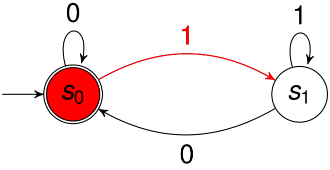
Sur l'alphabet $A = \{0, 1\}$, on peut représenter l'ensemble des nombres écrits en base 2. Le langage $L_{pair} = \{$l'ensemble des nombres pairs$\}$ est donc constitué de l'ensemble des mots de A^* se terminant par 0.



110

Automates et langages - Exemple

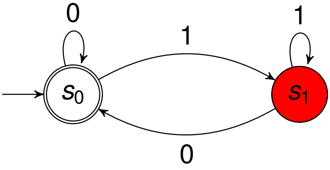
Sur l'alphabet $A = \{0, 1\}$, on peut représenter l'ensemble des nombres écrits en base 2. Le langage $L_{pair} = \{$l'ensemble des nombres pairs$\}$ est donc constitué de l'ensemble des mots de A^* se terminant par 0.



110

Automates et langages - Exemple

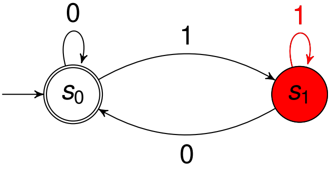
Sur l'alphabet $A = \{0, 1\}$, on peut représenter l'ensemble des nombres écrits en base 2. Le langage $L_{pair} = \{$l'ensemble des nombres pairs$\}$ est donc constitué de l'ensemble des mots de A^* se terminant par 0.



110

Automates et langages - Exemple

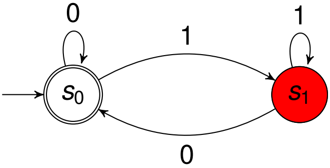
Sur l'alphabet $A = \{0, 1\}$, on peut représenter l'ensemble des nombres écrits en base 2. Le langage $L_{pair} = \{$l'ensemble des nombres pairs$\}$ est donc constitué de l'ensemble des mots de A^* se terminant par 0.



110

Automates et langages - Exemple

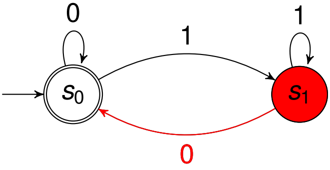
Sur l'alphabet $A = \{0, 1\}$, on peut représenter l'ensemble des nombres écrits en base 2. Le langage $L_{pair} = \{$l'ensemble des nombres pairs$\}$ est donc constitué de l'ensemble des mots de A^* se terminant par 0.



110

Automates et langages - Exemple

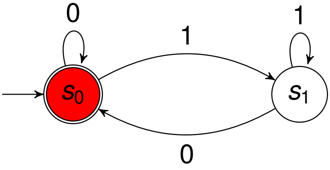
Sur l'alphabet $A = \{0, 1\}$, on peut représenter l'ensemble des nombres écrits en base 2. Le langage $L_{pair} = \{$l'ensemble des nombres pairs$\}$ est donc constitué de l'ensemble des mots de A^* se terminant par 0.



110

Automates et langages - Exemple

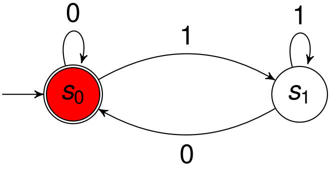
Sur l'alphabet $A = \{0, 1\}$, on peut représenter l'ensemble des nombres écrits en base 2. Le langage $L_{pair} = \{$l'ensemble des nombres pairs$\}$ est donc constitué de l'ensemble des mots de A^* se terminant par 0.



110

Automates et langages - Exemple

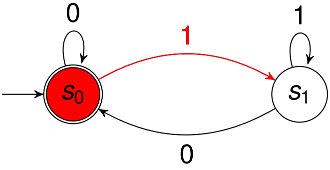
Sur l'alphabet $A = \{0, 1\}$, on peut représenter l'ensemble des nombres écrits en base 2. Le langage $L_{pair} = \{$l'ensemble des nombres pairs$\}$ est donc constitué de l'ensemble des mots de A^* se terminant par 0.



1101

Automates et langages - Exemple

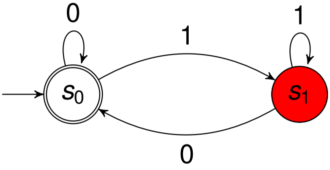
Sur l'alphabet $A = \{0, 1\}$, on peut représenter l'ensemble des nombres écrits en base 2. Le langage $L_{pair} = \{$l'ensemble des nombres pairs$\}$ est donc constitué de l'ensemble des mots de A^* se terminant par 0.



1101

Automates et langages - Exemple

Sur l'alphabet $A = \{0, 1\}$, on peut représenter l'ensemble des nombres écrits en base 2. Le langage $L_{pair} = \{$l'ensemble des nombres pairs$\}$ est donc constitué de l'ensemble des mots de A^* se terminant par 0.



1101

Automates - Définition

Définition

Sur un alphabet A, un automate fini est donné par $\mathcal{A} = (S, T, I, F)$ où

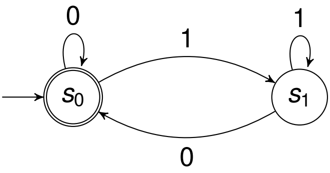
- S est un ensemble fini (non vide) d'*états*,
- $T \subseteq S \times A \times S$ est une *relation de transition*,
- $I \subseteq S$ est l'ensemble (non vide) des *états initiaux*,
- $F \subseteq S$ est l'ensemble des *états finaux*

Automates - Définition

Définition

Sur un alphabet A, un automate fini est donné par $\mathcal{A} = (S, T, I, F)$ où

- S est un ensemble fini (non vide) d'*états*,
- $T \subseteq S \times A \times S$ est une *relation de transition*,
- $I \subseteq S$ est l'ensemble (non vide) des *états initiaux*,
- $F \subseteq S$ est l'ensemble des *états finaux*

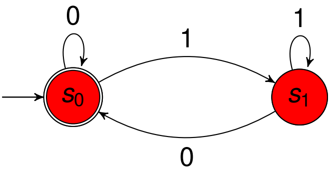


Automates - Définition

Définition

Sur un alphabet A, un automate fini est donné par $\mathcal{A} = (S, T, I, F)$ où

- S est un ensemble fini (non vide) d'*états*,
- $T \subseteq S \times A \times S$ est une *relation de transition*,
- $I \subseteq S$ est l'ensemble (non vide) des *états initiaux*,
- $F \subseteq S$ est l'ensemble des *états finaux*

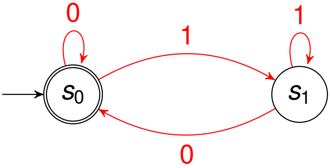


Automates - Définition

Définition

Sur un alphabet A, un automate fini est donné par $\mathcal{A} = (S, T, I, F)$ où

- S est un ensemble fini (non vide) d'*états*,
- $T \subseteq S \times A \times S$ est une *relation de transition*,
- $I \subseteq S$ est l'ensemble (non vide) des *états initiaux*,
- $F \subseteq S$ est l'ensemble des *états finaux*

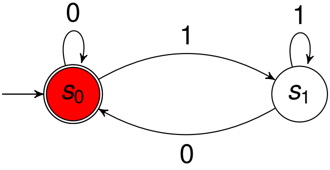


Automates - Définition

Définition

Sur un alphabet A, un automate fini est donné par $\mathcal{A} = (S, T, I, F)$ où

- S est un ensemble fini (non vide) d'*états*,
- $T \subseteq S \times A \times S$ est une *relation de transition*,
- $I \subseteq S$ est l'ensemble (non vide) des *états initiaux*,
- $F \subseteq S$ est l'ensemble des *états finaux*

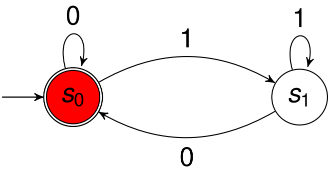


Automates - Définition

Définition

Sur un alphabet A, un automate fini est donné par $\mathcal{A} = (S, T, I, F)$ où

- S est un ensemble fini (non vide) d'*états*,
- $T \subseteq S \times A \times S$ est une *relation de transition*,
- $I \subseteq S$ est l'ensemble (non vide) des *états initiaux*,
- $F \subseteq S$ est l'ensemble des *états finaux*



Automates - Exécutions

Définition

Sur un alphabet A, un automate fini est donné par $\mathcal{A} = (S, T, I, F)$ où

- S est un ensemble fini d'*états*,
- $T \subseteq S \times A \times S$ est une *relation de transition*,
- $I \subseteq S$ est l'ensemble des *états initiaux*,
- $F \subseteq S$ est l'ensemble des *états finaux*

Une exécution de $\mathcal{A}$ est une séquence finie $s_0 a_1 s_1 a_2 \dots a_n s_n$, souvent notée $s_0 \xrightarrow{a_1} s_1 \xrightarrow{a_2} s_2 \dots s_{n-1} \xrightarrow{a_n} s_n$, telle que :

- $s_0 \in I$, est un état initial,
- pour tout $0 \leq i < n$, $(s_i, a_{i+1}, s_{i+1}) \in T$ est une transition autorisée par la relation de transition.

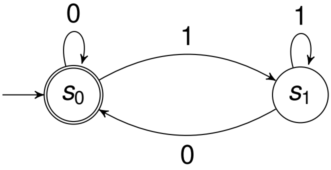
Automates - Exécutions

Une exécution de $\mathcal{A}$ est une séquence finie $s_0 a_1 s_1 a_2 \ldots a_n s_n$, souvent notée $s_0 \xrightarrow{a_1} s_1 \xrightarrow{a_2} s_2 \ldots s_{n-1} \xrightarrow{a_n} s_n$, telle que :

- $s_0 \in I$, est un état initial,
- pour tout $0 \leq i < n$, $(s_i, a_{i+1}, s_{i+1}) \in T$ est une transition autorisée par la relation de transition.

La séquence $a_1 a_2 \ldots a_n$ est un mot de A, qui *étiquette* l'exécution. Une exécution est *acceptante* si $s_n \in F$. Un mot $u \in A^*$ est *accepté par* $\mathcal{A}$ s'il est étiquette d'une exécution acceptante.

Automates - Exécutions



Exécution sur le mot 110 : $s_0 \xrightarrow{1} s_1 \xrightarrow{1} s_1 \xrightarrow{0} s_0$. C'est une exécution *acceptante*, 110 est accepté par l'automate.

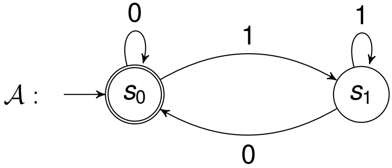
Exécution sur le mot 1101 : $s_0 \xrightarrow{1} s_1 \xrightarrow{1} s_1 \xrightarrow{0} s_0 \xrightarrow{1} s_1$. C'est une exécution *non acceptante*, 1101 est rejeté par l'automate.

Langages reconnaissables

- Soit $\mathcal{A}$ un automate fini sur l'alphabet A. Le *langage de* $\mathcal{A}$ est donné par $L(\mathcal{A}) = \{u \in A^* \mid u \text{ est accepté par } \mathcal{A}\}$.
- On dit que deux automates $\mathcal{A}$ et $\mathcal{B}$ sont *équivalents* si $L(\mathcal{A}) = L(\mathcal{B})$.
- Un langage $L \subseteq A^*$ est *reconnaissable* s'il existe un automate fini $\mathcal{A}$ sur l'alphabet A tel que $L = L(\mathcal{A})$.

Langages reconnaissables

- Soit $\mathcal{A}$ un automate fini sur l'alphabet A. Le *langage de* $\mathcal{A}$ est donné par $L(\mathcal{A}) = \{u \in A^* \mid u \text{ est accepté par } \mathcal{A}\}$.
- On dit que deux automates $\mathcal{A}$ et $\mathcal{B}$ sont *équivalents* si $L(\mathcal{A}) = L(\mathcal{B})$.
- Un langage $L \subseteq A^*$ est *reconnaissable* s'il existe un automate fini $\mathcal{A}$ sur l'alphabet A tel que $L = L(\mathcal{A})$.



$$
\begin{aligned}
L(\mathcal{A}) &= \{w \in A^* \mid w \text{ se termine par } 0\} \\
&= \{w \in A^* \mid w \text{ est un nombre pair codé en base } 2\} = L_{\text{pair}}
\end{aligned}
$$

L_{pair} est un langage reconnaissable

Automates non déterministes

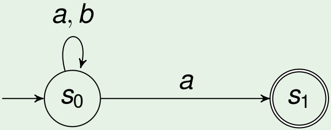
Exemple

Le langage $L = \{w \in \{a,b\}^* \mid w \text{ se termine par } a\}$ est reconnaissable.

Automates non déterministes

Exemple

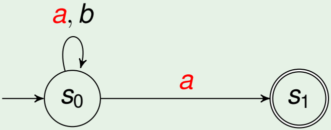
Le langage $L = \{w \in \{a, b\}^* \mid w \text{ se termine par } a\}$ est reconnaissable.



Automates non déterministes

Exemple

Le langage $L = \{w \in \{a, b\}^* \mid w \text{ se termine par } a\}$ est reconnaissable.

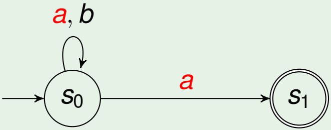


non déterminisme

Automates non déterministes

Exemple

Le langage $L = \{w \in \{a,b\}^* \mid w \text{ se termine par } a\}$ est reconnaissable.



non déterminisme

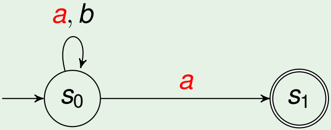
Exécution sur *aba* :

$s_0 \xrightarrow{a} s_0 \xrightarrow{b} s_0 \xrightarrow{a} s_0$

Automates non déterministes

Exemple

Le langage $L = \{w \in \{a, b\}^* \mid w \text{ se termine par } a\}$ est reconnaissable.



non déterminisme

Exécution sur *aba* :

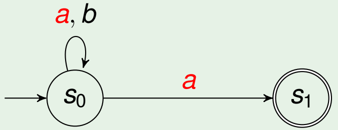
$s_0 \xrightarrow{a} s_0 \xrightarrow{b} s_0 \xrightarrow{a} s_0$

$s_0 \xrightarrow{a} s_1 \xrightarrow{b} \dots$

Automates non déterministes

Exemple

Le langage $L = \{w \in \{a, b\}^* \mid w \text{ se termine par } a\}$ est reconnaissable.



non déterminisme

Exécution sur *aba* :

$s_0 \xrightarrow{a} s_0 \xrightarrow{b} s_0 \xrightarrow{a} s_0$

$s_0 \xrightarrow{a} s_1 \xrightarrow{b} \ldots$

$s_0 \xrightarrow{a} s_0 \xrightarrow{b} s_0 \xrightarrow{a} s_1$

Automates - Retour sur les exécutions

Une exécution de $\mathcal{A}$ est une séquence finie de transitions $(s_0, a_1, s_1)(s_1, a_2, s_2)\ldots(s_{n-1}, a_{n-1}, s_n)$, souvent notée $s_0 \xrightarrow{a_1} s_1 \xrightarrow{a_2} s_2 \ldots s_{n-1} \xrightarrow{a_n} s_n$, telle que :

- $s_0 \in I$, est un état initial,
- pour tout $0 \leq i < n$, $(s_i, a_{i+1}, s_{i+1}) \in T$ est une transition autorisée par la relation de transition.

La séquence $a_1 a_2 \ldots a_n$ est un mot de A, qui *étiquette* l'exécution. Une exécution est *acceptante* si $s_n \in F$. Un mot $u \in A^*$ est *accepté par* $\mathcal{A}$ s'il est étiquette d'une exécution acceptante.

Automates - Retour sur les exécutions

Une exécution de $\mathcal{A}$ est une séquence finie de transitions $(s_0, a_1, s_1)(s_1, a_2, s_2)\ldots(s_{n-1}, a_{n-1}, s_n)$, souvent notée $s_0 \xrightarrow{a_1} s_1 \xrightarrow{a_2} s_2 \ldots s_{n-1} \xrightarrow{a_n} s_n$, telle que :

- $s_0 \in I$, est un état initial,
- pour tout $0 \leq i < n$, $(s_i, a_{i+1}, s_{i+1}) \in T$ est une transition autorisée par la relation de transition.

La séquence $a_1 a_2 \ldots a_n$ est un mot de A, qui *étiquette* l'exécution. Une exécution est *acceptante* si $s_n \in F$. Un mot $u \in A^*$ est *accepté par* $\mathcal{A}$ s'il est étiquette d'au moins une exécution acceptante.

Automates complets

Définition

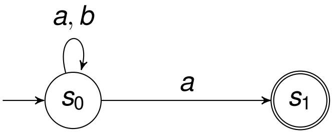
Un automate $\mathcal{A} = (S, T, I, F)$ sur un alphabet A est *complet* si, pour tout $s \in S$, pour tout $a \in A$, il existe $s' \in S$ et $(s, a, s') \in T$.

Remarque: Dans un automate complet, tout mot est étiquette d'au moins une exécution.

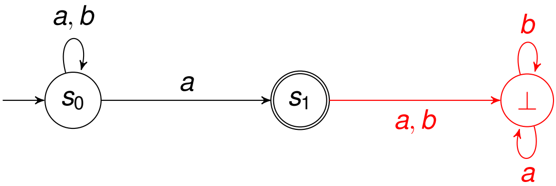
Proposition

Tout automate fini est équivalent à un automate complet.

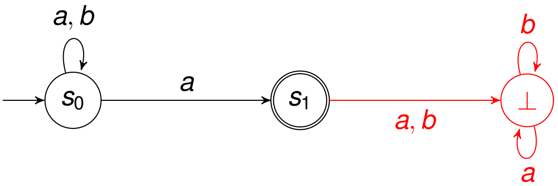
Compléter un automate



Compléter un automate



Compléter un automate



Contruction: Soit $\mathcal{A} = (S, T, I, F)$ un automate fini sur un alphabet A. On construit $\mathrm{comp}(\mathcal{A}) = (S \uplus \{\perp\}, T', I, F)$ avec
$T' = T \uplus \{(s, a, \perp) \mid s \in S \uplus \{\perp\}, a \in A, \text{ et pour tout } s' \in S, (s, a, s') \notin T\}$.

- $\mathrm{comp}(\mathcal{A})$ est complet.
- $L(\mathrm{comp}(\mathcal{A})) = L(\mathcal{A})$.

Automates déterministes

Définition

Un automate fini déterministe sur un alphabet A est un automate donné par $\mathcal{A} = (S, T, i, F)$ avec i *unique* état initial, et T tel que pour tout $s \in S$, pour tout $a \in A$, si $(s, a, s') \in T$ et $(s, a, s'') \in T$, alors $s = s''$.

Remarque: Dans un automate fini déterministe, T est en fait une *fonction* qui peut être notée $T : S \times A \to S$. On note parfois $T(s, a) = s'$ pour $(s, a, s') \in T$ dans un automate déterministe.

Remarque: Dans un automate fini déterministe, tout mot est étiquette d'au plus une exécution.

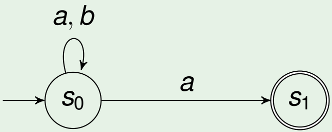
Automates déterministes

Définition

Un automate fini déterministe sur un alphabet A est un automate donné par $\mathcal{A} = (S, T, i, F)$ avec i *unique* état initial, et T tel que pour tout $s \in S$, pour tout $a \in A$, si $(s, a, s') \in T$ et $(s, a, s'') \in T$, alors $s' = s''$.

Exemple

$\mathcal{A}$:

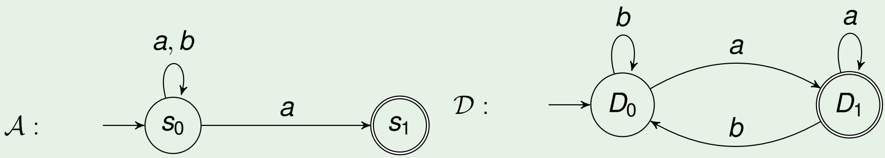


Automates déterministes

Définition

Un automate fini déterministe sur un alphabet A est un automate donné par $\mathcal{A} = (S, T, i, F)$ avec i *unique* état initial, et T tel que pour tout $s \in S$, pour tout $a \in A$, si $(s, a, s') \in T$ et $(s, a, s'') \in T$, alors $s' = s''$.

Exemple



$L(\mathcal{A}) = L(\mathcal{D})$

Automates déterministes

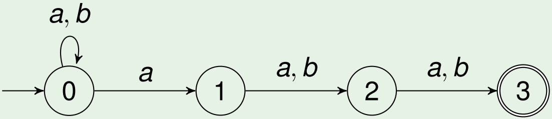
Exemple

Automate sur l'alphabet $A = \{a, b\}$ reconnaissant les mots dont la troisième lettre avant la fin est un a ?

Automates déterministes

Exemple

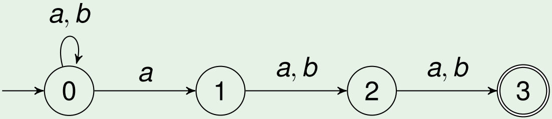
Automate sur l'alphabet $A = \{a, b\}$ reconnaissant les mots dont la troisième lettre avant la fin est un a?



Automates déterministes

Exemple

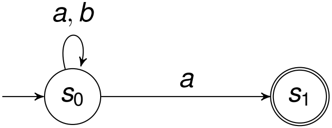
Automate sur l'alphabet $A = \{a, b\}$ reconnaissant les mots dont la troisième lettre avant la fin est un a ?



Existe-t-il un automate déterministe équivalent ?

Déterminiser un automate - Exemple

$\mathcal{A}$:



Déterminiser un automate

Théorème

Tout automate fini est équivalent à un automate fini déterministe.

Remarque: La démonstration est constructive : pour un automate $\mathcal{A}$, on peut construire $\mathrm{det}(\mathcal{A})$.

Contruction: Soit $\mathcal{A} = (S, T, I, F)$ un automate fini non déterministe sur un alphabet A. On définit $\mathrm{det}(\mathcal{A}) = (\mathcal{P}(S), T', I, F')$ avec

- Pour $X \in \mathcal{P}(S)$, pour $a \in A$,
  $T'(X, a) = \{s' \in S \mid \text{il existe } s \in X \text{ et } (s, a, s') \in T\}$.
- $F' = \{X \subseteq S \mid X \cap F \neq \emptyset\}$

Déterminiser un automate

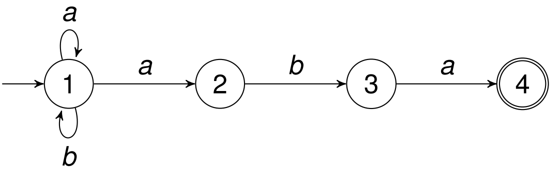
Théorème

Tout automate fini est équivalent à un automate fini déterministe.

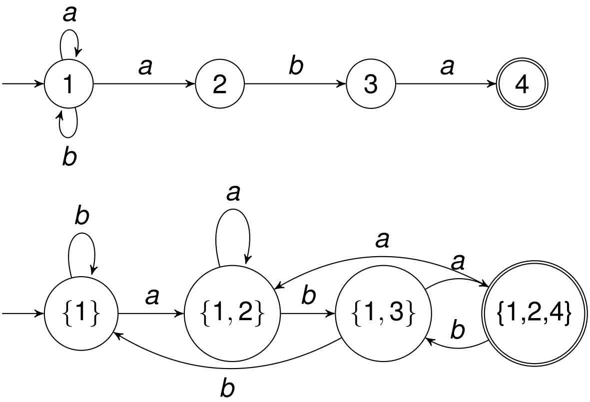
Remarque: La démonstration est constructive : pour un automate $\mathcal{A}$, on peut construire $\mathrm{det}(\mathcal{A})$.
Contruction: Soit $\mathcal{A} = (S, T, I, F)$ un automate fini non déterministe sur un alphabet A. On définit $\mathrm{det}(\mathcal{A}) = (\mathcal{P}(S), T', I, F')$ avec

- Pour $X \in \mathcal{P}(S)$, pour $a \in A$,
  $T'(X, a) = \{s' \in S \mid \text{il existe } s \in X \text{ et } (s, a, s') \in T\}$.
- $F' = \{X \subseteq S \mid X \cap F \neq \emptyset\}$
- $\mathrm{det}(\mathcal{A})$ est déterministe.
- $L(\mathcal{A}) = L(\mathrm{det}(\mathcal{A}))$.
- Remarque : I, état initial de $\mathrm{det}(\mathcal{A})$, est un *ensemble* d'états, comme tout état de $\mathrm{det}(\mathcal{A})$. L'état initial de $\mathrm{det}(\mathcal{A})$ peut aussi s'écrire $\{i \in I\}$.

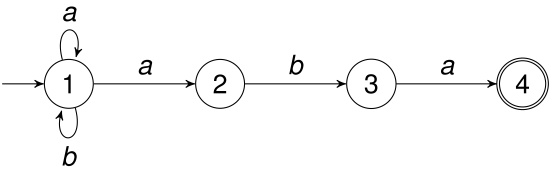
Déterminiser un automate - Exemple

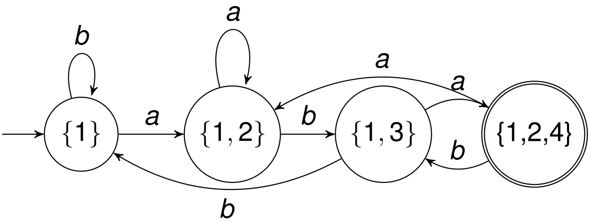


Déterminiser un automate - Exemple



Déterminiser un automate - Exemple



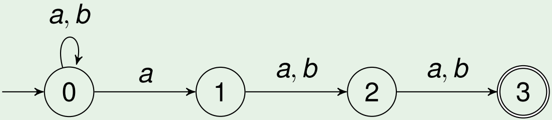


Remarque: La définition donne un automate à $2^4 = 16$ états. On a construit l'automate restreint aux états accessibles.

Déterminiser un automate - Exemple

Exemple

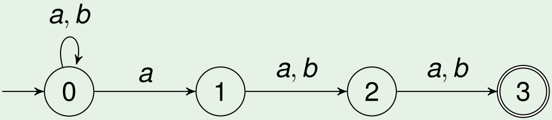
Automate sur l'alphabet $A = \{a, b\}$ reconnaissant les mots dont la troisième lettre avant la fin est un a ?

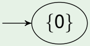


Déterminiser un automate - Exemple

Exemple

Automate sur l'alphabet $A = \{a, b\}$ reconnaissant les mots dont la troisième lettre avant la fin est un a ?

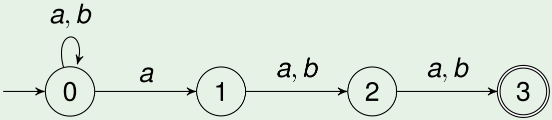


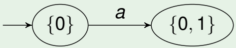


Déterminiser un automate - Exemple

Exemple

Automate sur l'alphabet $A = \{a, b\}$ reconnaissant les mots dont la troisième lettre avant la fin est un a ?

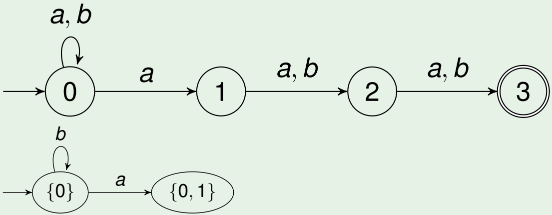




Déterminiser un automate - Exemple

Exemple

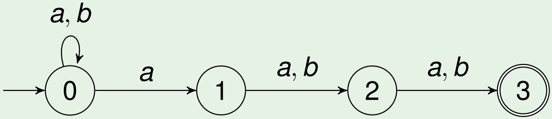
Automate sur l'alphabet $A = \{a, b\}$ reconnaissant les mots dont la troisième lettre avant la fin est un a ?

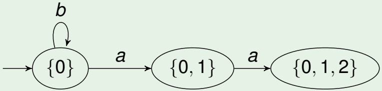


Déterminiser un automate - Exemple

Exemple

Automate sur l'alphabet $A = \{a, b\}$ reconnaissant les mots dont la troisième lettre avant la fin est un a ?

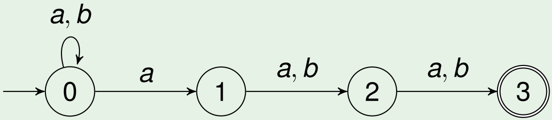


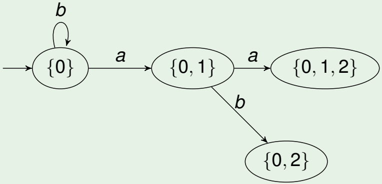


Déterminiser un automate - Exemple

Exemple

Automate sur l'alphabet $A = \{a, b\}$ reconnaissant les mots dont la troisième lettre avant la fin est un a ?

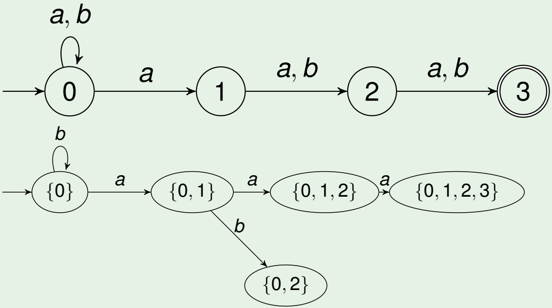




Déterminiser un automate - Exemple

Exemple

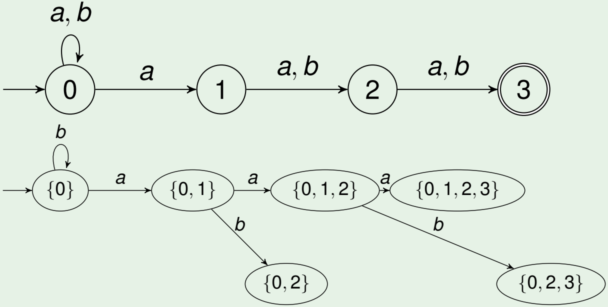
Automate sur l'alphabet $A = \{a, b\}$ reconnaissant les mots dont la troisième lettre avant la fin est un a ?



Déterminiser un automate - Exemple

Exemple

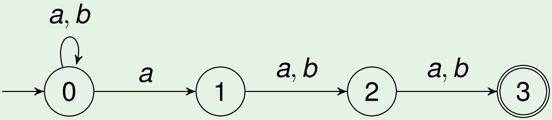
Automate sur l'alphabet $A = \{a, b\}$ reconnaissant les mots dont la troisième lettre avant la fin est un a ?

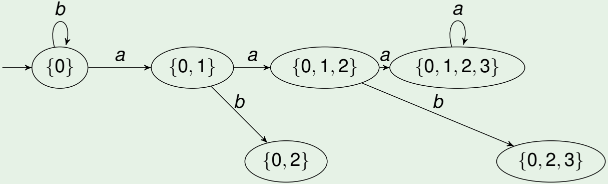


Déterminiser un automate - Exemple

Exemple

Automate sur l'alphabet $A = \{a, b\}$ reconnaissant les mots dont la troisième lettre avant la fin est un a?

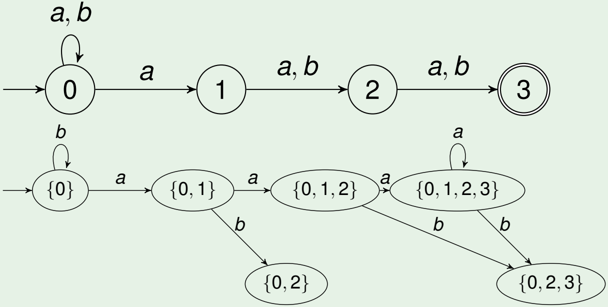




Déterminiser un automate - Exemple

Exemple

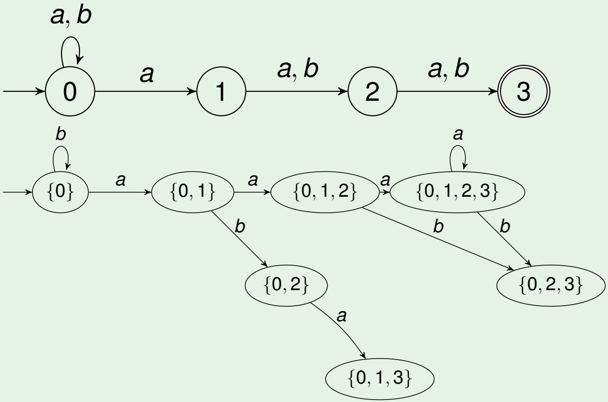
Automate sur l'alphabet $A = \{a, b\}$ reconnaissant les mots dont la troisième lettre avant la fin est un a?



Déterminiser un automate - Exemple

Exemple

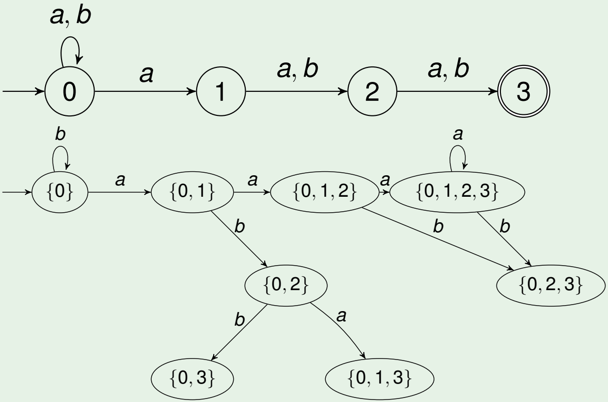
Automate sur l'alphabet $A = \{a, b\}$ reconnaissant les mots dont la troisième lettre avant la fin est un a?



Déterminiser un automate - Exemple

Exemple

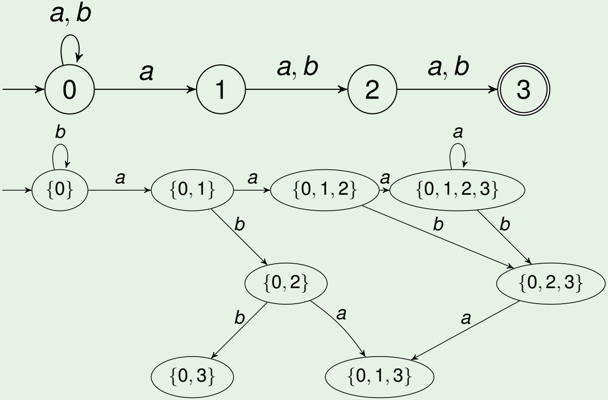
Automate sur l'alphabet $A = \{a, b\}$ reconnaissant les mots dont la troisième lettre avant la fin est un a ?



Déterminiser un automate - Exemple

Exemple

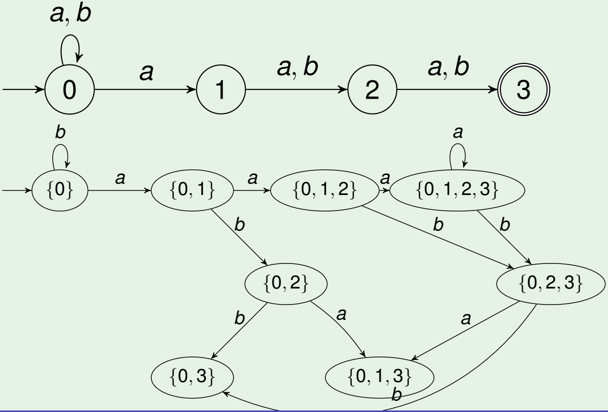
Automate sur l'alphabet $A = \{a, b\}$ reconnaissant les mots dont la troisième lettre avant la fin est un a ?



Déterminiser un automate - Exemple

Exemple

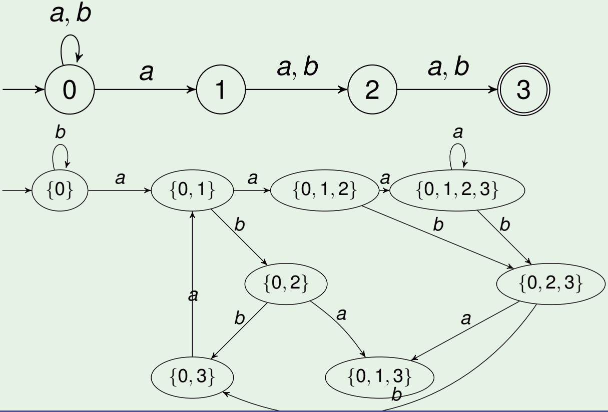
Automate sur l'alphabet $A = \{a, b\}$ reconnaissant les mots dont la troisième lettre avant la fin est un a?



Déterminiser un automate - Exemple

Exemple

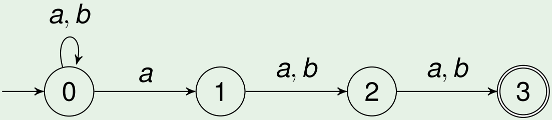
Automate sur l'alphabet $A = \{a, b\}$ reconnaissant les mots dont la troisième lettre avant la fin est un a ?

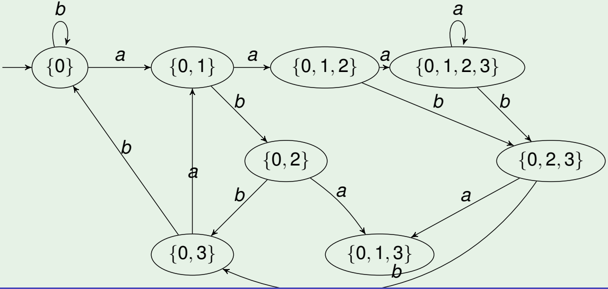


Déterminiser un automate - Exemple

Exemple

Automate sur l'alphabet $A = \{a, b\}$ reconnaissant les mots dont la troisième lettre avant la fin est un a ?

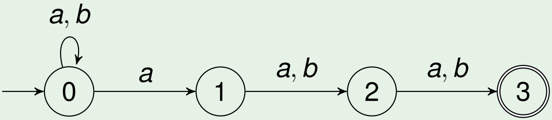


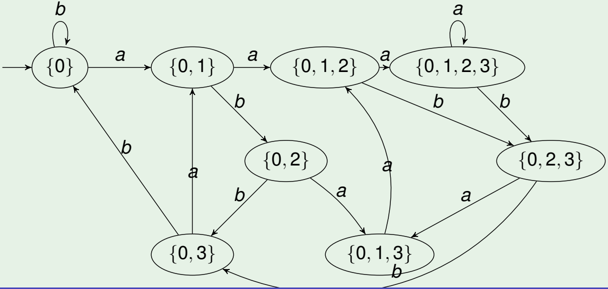


Déterminiser un automate - Exemple

Exemple

Automate sur l'alphabet $A = \{a, b\}$ reconnaissant les mots dont la troisième lettre avant la fin est un a ?

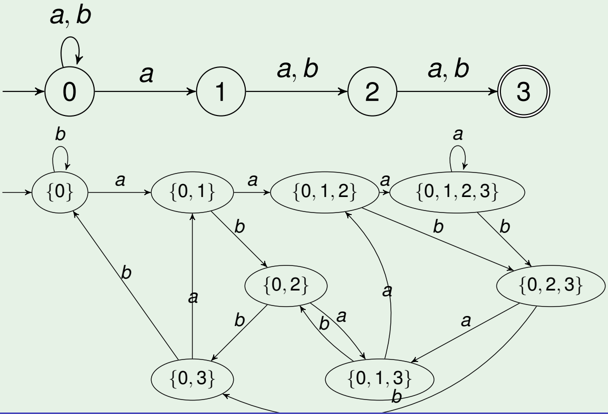




Déterminiser un automate - Exemple

Exemple

Automate sur l'alphabet $A = \{a, b\}$ reconnaissant les mots dont la troisième lettre avant la fin est un a ?



Propriété de clôtures des langages reconnaissables - Complémentaire

Théorème

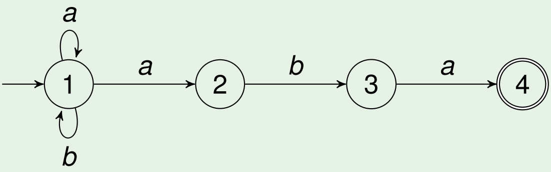
Si $L \subseteq A^*$ est un langage reconnaissable, alors $\overline{L}$ est reconnaissable.

Contruction: Soit $\mathcal{A}$ tel que $L(\mathcal{A}) = L$. On suppose que $\mathcal{A}$ est *déterministe* et *complet* (sinon on le déterminise et/ou complète). Si $\mathcal{A} = (S, T, i, F)$ alors on construit $\mathcal{B} = (S, T, i, F')$ avec $F' = S \setminus F$.

- $L(\mathcal{B}) = \overline{L}$.
- Cela ne fonctionne que si $\mathcal{A}$ est bien déterministe et complet !

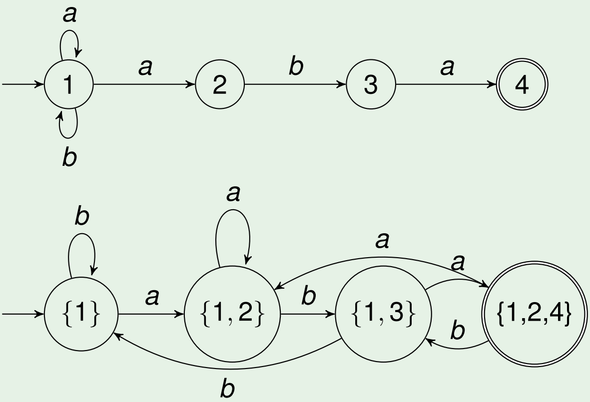
Propriété de clôtures des langages reconnaissables - Complémentaire

Exemple



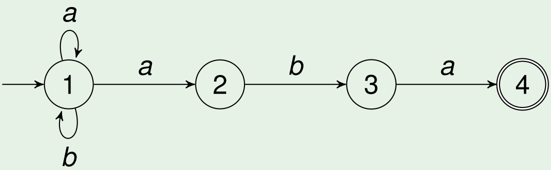
Propriété de clôtures des langages reconnaissables - Complémentaire

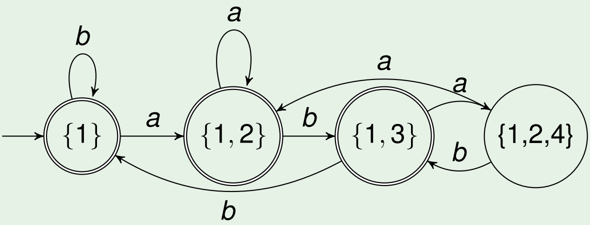
Exemple



Propriété de clôtures des langages reconnaissables - Complémentaire

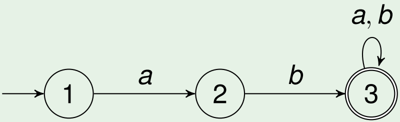
Exemple





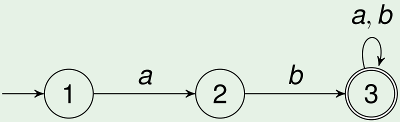
Propriété de clôtures des langages reconnaissables - Complémentaire

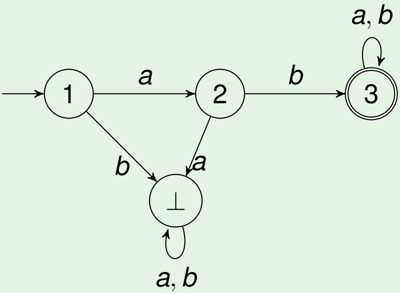
Exemple



Propriété de clôtures des langages reconnaissables - Complémentaire

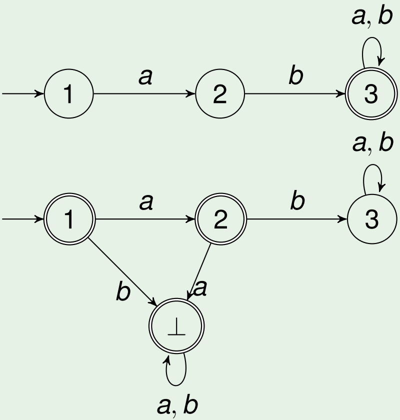
Exemple





Propriété de clôtures des langages reconnaissables - Complémentaire

Exemple



Propriété de clôtures des langages reconnaissables - Intersection et union

Théorème

Si $L_1, L_2 \subseteq A^*$ sont deux langages reconnaissables, alors les langages $L_1 \cap L_2$ et $L_1 \cup L_2$ sont reconnaissables.

Propriété de clôtures des langages reconnaissables - Intersection et union

Théorème

Si $L_1, L_2 \subseteq A^*$ sont deux langages reconnaissables, alors les langages $L_1 \cap L_2$ et $L_1 \cup L_2$ sont reconnaissables.

Contruction: Soient $\mathcal{A}_1 = (S_1, T_1, I_1, F_1)$ et $\mathcal{A}_2 = (S_2, T_2, I_2, F_2)$ deux automates reconnaissant respectivement L_1 et L_2. On définit l'automate produit $\mathcal{A} = (S_1 \times S_2, T, I_1 \times I_2, F)$ avec
$T = \{((s_1, s_2), a, (s_1', s_2')) \mid (s_1, a, s_1') \in T_1 \text{ et } (s_2, a, s_2') \in T_2\}$

- **Intersection** Si $F = F_1 \times F_2$, $L(\mathcal{A}) = L_1 \cap L_2$.
- **Union** *Si $\mathcal{A}_1$ et $\mathcal{A}_2$ sont complets*, et $F = (F_1 \times S_2) \cup (S_1 \times F_2)$, alors $L(\mathcal{A}) = L_1 \cup L_2$.

Remarque: La construction préserve le déterminisme : si $\mathcal{A}_1$ et $\mathcal{A}_2$ sont déterministes, $\mathcal{A}$ est déterministe aussi.

Propriété de clôtures des langages reconnaissables - Intersection et union

Exemple

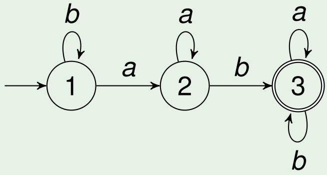
Sur $A = \{a, b\}$, $L_1 = A^*\{ab\}A^*$ et $L_2 = \{b\}A^*\{a\}$

Propriété de clôtures des langages reconnaissables - Intersection et union

Exemple

Sur $A = \{a, b\}$, $L_1 = A^*\{ab\}A^*$ et $L_2 = \{b\}A^*\{a\}$

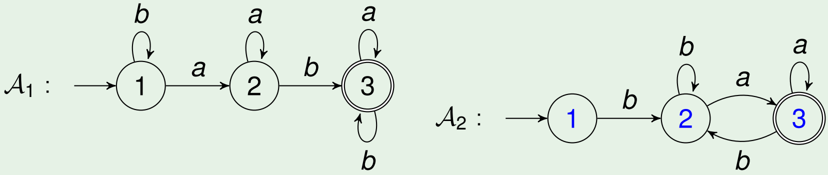
$\mathcal{A}_1$:



Propriété de clôtures des langages reconnaissables - Intersection et union

Exemple

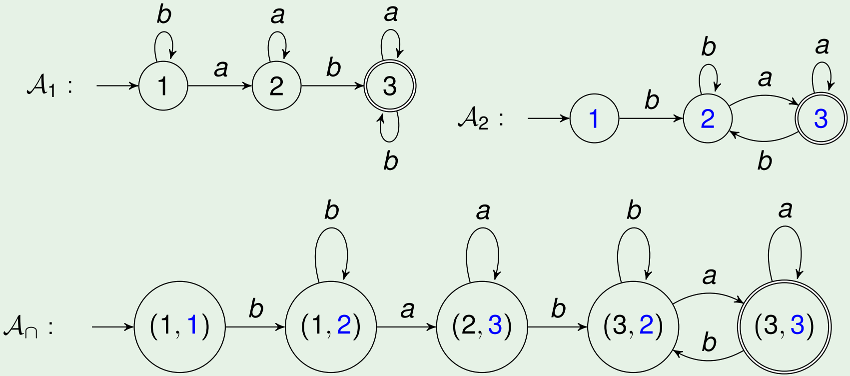
Sur $A = \{a, b\}$, $L_1 = A^*\{ab\}A^*$ et $L_2 = \{b\}A^*\{a\}$



Propriété de clôtures des langages reconnaissables - Intersection et union

Exemple

Sur $A = \{a, b\}$, $L_1 = A^*\{ab\}A^*$ et $L_2 = \{b\}A^*\{a\}$

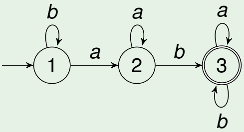


Propriété de clôtures des langages reconnaissables - Intersection et union

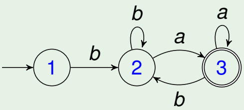
Exemple

Sur $A = \{a, b\}$, $L_1 = A^*\{ab\}A^*$ et $L_2 = \{b\}A^*\{a\}$

$\mathcal{A}_1$:



$\mathcal{A}_2$:

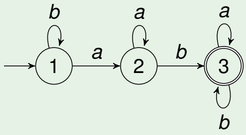


Propriété de clôtures des langages reconnaissables - Intersection et union

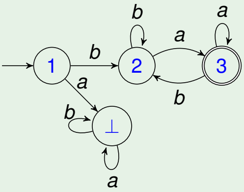
Exemple

Sur $A = \{a, b\}$, $L_1 = A^*\{ab\}A^*$ et $L_2 = \{b\}A^*\{a\}$

$\mathcal{A}_1$:



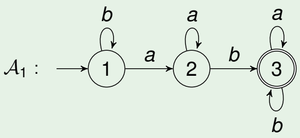
$\mathcal{A}_2$:

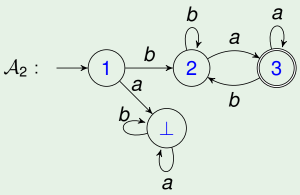


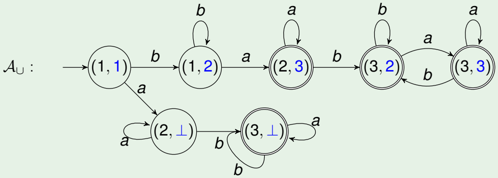
Propriété de clôtures des langages reconnaissables - Intersection et union

Exemple

Sur $A = \{a, b\}$, $L_1 = A^*\{ab\}A^*$ et $L_2 = \{b\}A^*\{a\}$







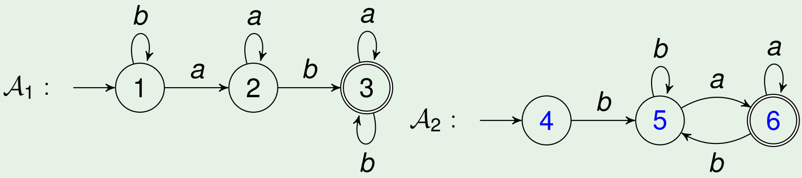
Propriété de clôtures des langages reconnaissables - Union, autre construction

Contruction: Soient $\mathcal{A}_1 = (S_1, T_1, I_1, F_1)$ et $\mathcal{A}_2 = (S_2, T_2, I_2, F_2)$ deux automates reconnaissant respectivement L_1 et L_2. Si $S_1 \cap S_2 = \emptyset$, ce qui est toujours possible par renommage, on construit
$\mathcal{A} = (S_1 \cup S_2, T_1 \cup T_2, I_1 \cup I_2, F_1 \cup F_2)$.
$L(\mathcal{A}) = L_1 \cup L_2$.
Remarque: Il s'agit de considérer les deux automates comme un seul automate non déterministe.

Propriété de clôtures des langages reconnaissables - Union, autre construction

Exemple

Sur $A = \{a, b\}$, $L_1 = A^*\{ab\}A^*$ et $L_2 = \{b\}A^*\{a\}$

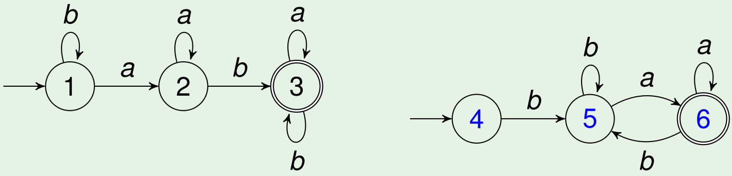


Propriété de clôtures des langages reconnaissables - Union, autre construction

Exemple

Sur $A = \{a, b\}$, $L_1 = A^*\{ab\}A^*$ et $L_2 = \{b\}A^*\{a\}$

$\mathcal{A}$:



Propriétés de clôture des langages reconnaissables - Concaténation

Théorème

Si $L_1, L_2 \subseteq A^*$ sont deux langages reconnaissables, alors $L_1.L_2$ est reconnaissable.

Contruction: Soient $\mathcal{A}_1 = (S_1, T_1, I_1, F_1)$ et $\mathcal{A}_2 = (S_2, T_2, I_2, F_2)$ deux automates reconnaissant respectivement L_1 et L_2. On construit
$\mathcal{A} = (S, T, I, F_2)$ avec $S = S_1 \cup S_2$,
$T = T_1 \cup T_2 \cup \{(s, a, i) \mid i \in I_2 \text{ et il existe } f \in F_1 \text{ tel que } (s, a, f) \in T_1\}$ et

$$I = \begin{cases} I_1 & \text{si } I_1 \cap F_1 = \emptyset \\ I_1 \cup I_2 & \text{sinon.} \end{cases}$$

$L(\mathcal{A}) = L_1.L_2$

Remarque: Cette construction ne préserve pas le déterminisme.

Propriétés de clôture des langages reconnaissables - Concaténation

Exemple

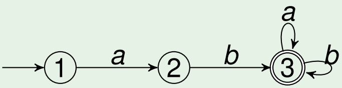
Sur $A = \{a, b\}$, $L_1 = \{ab\}A^*$, $L_2 = A^*\{ba\}$, $L_1.L_2 =?$

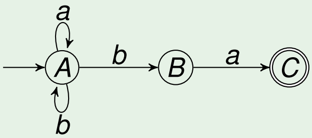
Remarque: Différence entre $L_1.L_2$ et $L_1 \cap L_2$?

Propriétés de clôture des langages reconnaissables - Concaténation

Exemple

Sur $A = \{a, b\}$, $L_1 = \{ab\}A^*$, $L_2 = A^*\{ba\}$, $L_1.L_2 = ?$



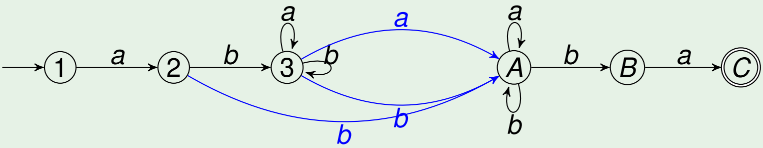


Remarque: Différence entre $L_1.L_2$ et $L_1 \cap L_2$?

Propriétés de clôture des langages reconnaissables - Concaténation

Exemple

Sur $A = \{a, b\}$, $L_1 = \{ab\}A^*$, $L_2 = A^*\{ba\}$, $L_1.L_2 = ?$



Remarque: Différence entre $L_1.L_2$ et $L_1 \cap L_2$?

Propriétés de clôture des langages reconnaissables - Étoile

Théorème

Soit $L \subseteq A^*$ un langage reconnaissable, alors L^+ et L^* sont reconnaissables.

Contruction: Soit $\mathcal{A} = (S, T, I, F)$ un automate reconnaissant L. On construit d'abord $\mathcal{A}_+ = (S, T_+, I, F)$ un automate reconnaissant L^+ :
$T_+ = T \uplus \{(s, a, i) \mid i \in I \text{ et il existe } f \in F, (s, a, f) \in T\}$. On construit alors $\mathcal{A}_* = (S \uplus \{j\}, T_+, I \uplus \{j\}, F \uplus \{j\})$.
$L(\mathcal{A}_+) = L^+$ et $L(\mathcal{A}_*) = L^*$.

Propriétés de clôture des langages reconnaissables - Étoile

Exemple

Sur $A = \{a, b\}$, $L = \{\text{mots contenant } 2a\}$. $L^+ =$

Propriétés de clôture des langages reconnaissables - Étoile

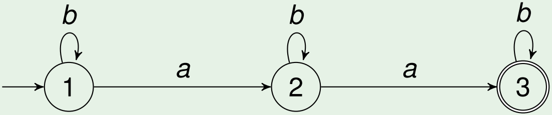
Exemple

Sur $A = \{a, b\}$, $L = \{\text{mots contenant } 2a\}$. $L^+ = \{\text{mots contenant } 2ka \mid k \geq 1\}$

Propriétés de clôture des langages reconnaissables - Étoile

Exemple

Sur $A = \{a, b\}$, $L = \{\text{mots contenant } 2a\}$. $L^+ = \{\text{mots contenant } 2ka \mid k \geq 1\}$



Propriétés de clôture des langages reconnaissables - Étoile

Exemple

Sur $A = \{a, b\}$, $L = \{\text{mots contenant } 2a\}$. $L^+ = \{\text{mots contenant } 2ka \mid k \geq 1\}$