Automate Fini Non-déterministe
Théorème de Kleene

Intelligence Artificielle et Systèmes Formels
Master 1 I2L

Sébastien Verel
verel@lisic.univ-littoral.fr
http://www-lisic.univ-littoral.fr/~verel

Université du Littoral Côte d'Opale
Laboratoire LISIC
Equipe CAMOME

Objectifs de la séance 03

- Connaitre la définition d'un automate fini non-déternimiste
- Savoir déterminiser un automate
- Savoir construire un automate à état fini reconnaissant un language rationnel simple
- Connaître le théorème de Kleene

Objectifs de la séance 03

- Connaitre la définition d'un automate fini non-déternimiste
- Savoir déterminiser un automate
- Savoir construire un automate à état fini reconnaissant un language rationnel simple
- Connaître le théorème de Kleene

Questions principales du jour :

Comment définir des automates reconnaissant un langage de manière simple ?

Références

Rappels de références concernant les langages et les automates :

- `www-igm.univ-mlv.fr/~Eberstel/Elements/Elements.html`
- S. Julia, `deptinfo.unice.fr/~julia/IT/`
- `www.polytech.unice.fr/~claudine`

Plan

1. Introduction
2. Automate Fini Non-déterministe
3. Déterminisation
4. Théorème de Kleene

Equivalence expression régulière et langage rationnel

Théorème (admis)

Un langage est rationnel (ou régulier)
si et seulement si
il est décrit par une expression régulière.

Equivalence expression régulière et langage rationnel

Théorème (admis)

Un langage est rationnel (ou régulier)
si et seulement si
il est décrit par une expression régulière.

Cardinalité

L'ensemble des langages rationnels est dénombrable.

Remarque : il existe beaucoup de langage non rationnel...

Automate Fini Déterministe (AFD)

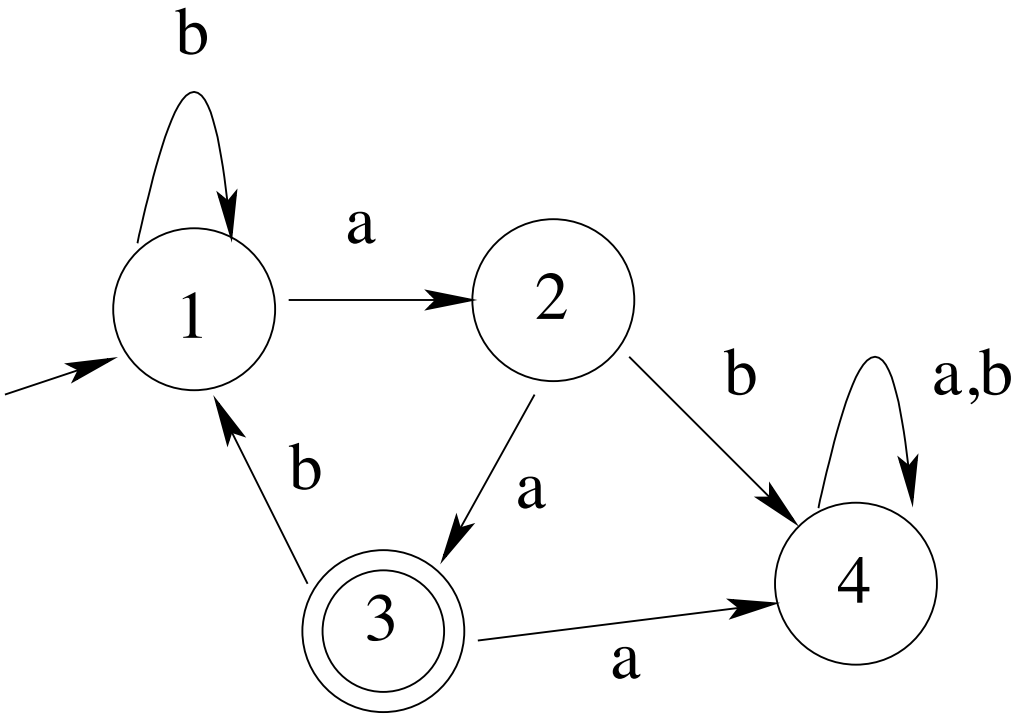
Automate Fini Déterministe (AFD)

Automate Fini Déterministe (AFD)

Un **Automate Fini Déterministe** est un quintuplet (Q, Σ, T, q_0, A) avec :

- Σ est l'*alphabet* de l'automate,
- Q un ensemble fini appelé *ensemble des états* de l'automate,
- T est une application de $Q \times \Sigma$ dans Q, appelée la *fonction de transition*
- q_0 est un élément de Q, appelé l'*état initial*
- A est un sous-ensemble de Q, appelé l'*ensemble des états acceptants*.

Exemple



Langage reconnu par un automate

fonction de transition itérée

La **fonction de transition itérée** est l'application $T^* : Q \times \Sigma^* \to Q$ définie par :

- *base :* si $w = \epsilon$ alors $T^*(q, w) = q$
- *induction :* si $w = w_0 x$ avec $x \in \Sigma$ alors $T^*(q, w) = T(T^*(q, w_0), x)$

Langage décidé

Soient M est un automate d'alphabet Σ et L un langage sur Σ

M **décide** L ssi L est l'ensemble des mots acceptés par M.

Reconnaissance d'un langage de cardinal 1

Soit Σ un alphabet et $L = \{u\}$ un langage sur Σ de cardinal 1.

u s'écrit alors comme $u = a_1 a_2 a_3 \dots a_n$ avec $\forall i \ a_i \in \Sigma$.

Reconnaissance d'un langage de cardinal 1

Soit Σ un alphabet et $L = \{u\}$ un langage sur Σ de cardinal 1.

u s'écrit alors comme $u = a_1 a_2 a_3 \dots a_n$ avec $\forall i \ a_i \in \Sigma$.

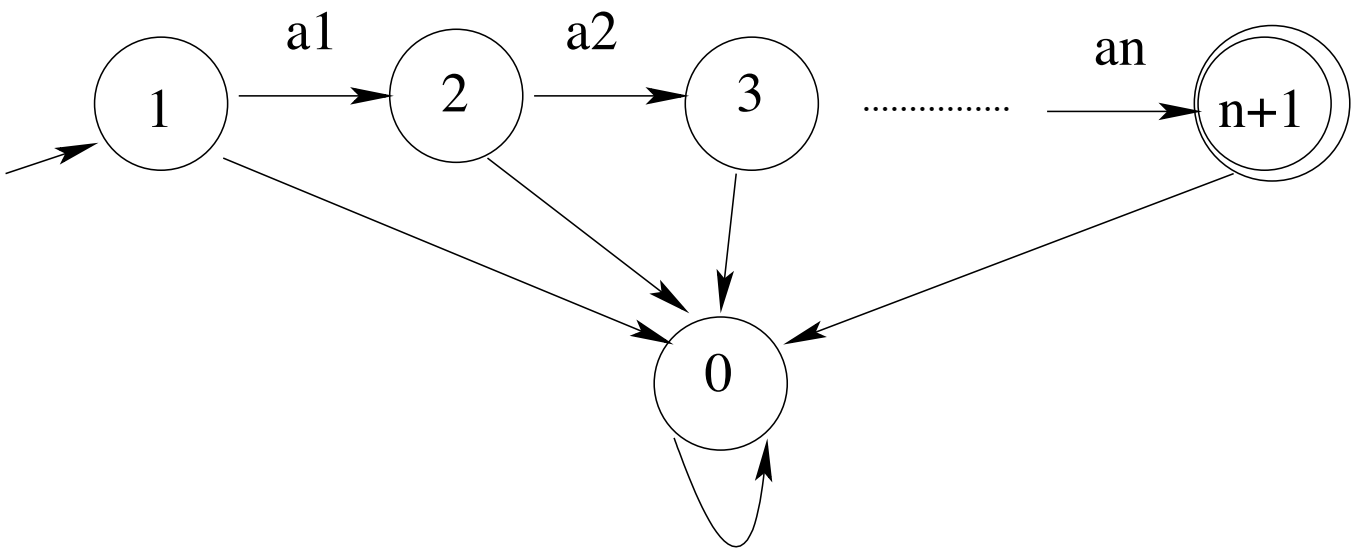
Automate reconnaissant le langage L :

Reconnaissance d'un langage de cardinal 1

Soit Σ un alphabet et $L = \{u\}$ un langage sur Σ de cardinal 1.

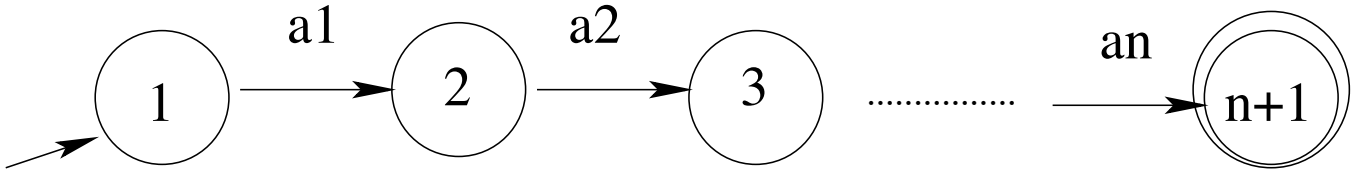
u s'écrit alors comme $u = a_1 a_2 a_3 \dots a_n$ avec $\forall i \ a_i \in \Sigma$.

Automate reconnaissant le langage L :



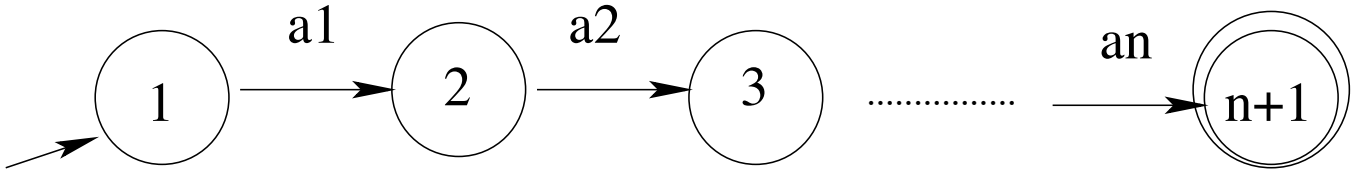
Quelques difficultés pratiques

- Il n'est pas "pratique" de devoir définir toutes les transitions, on aimerait :



Quelques difficultés pratiques

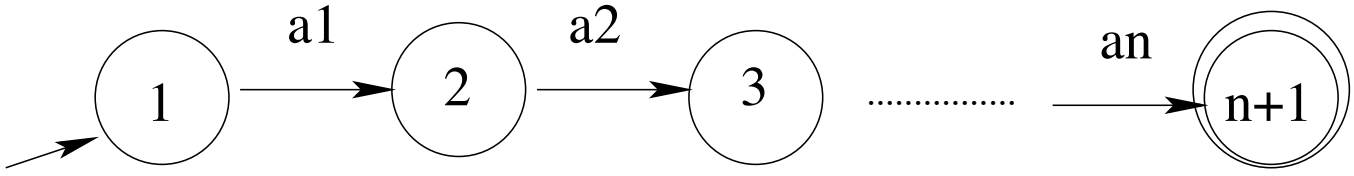
- Il n'est pas "pratique" de devoir définir toutes les transitions, on aimerait :



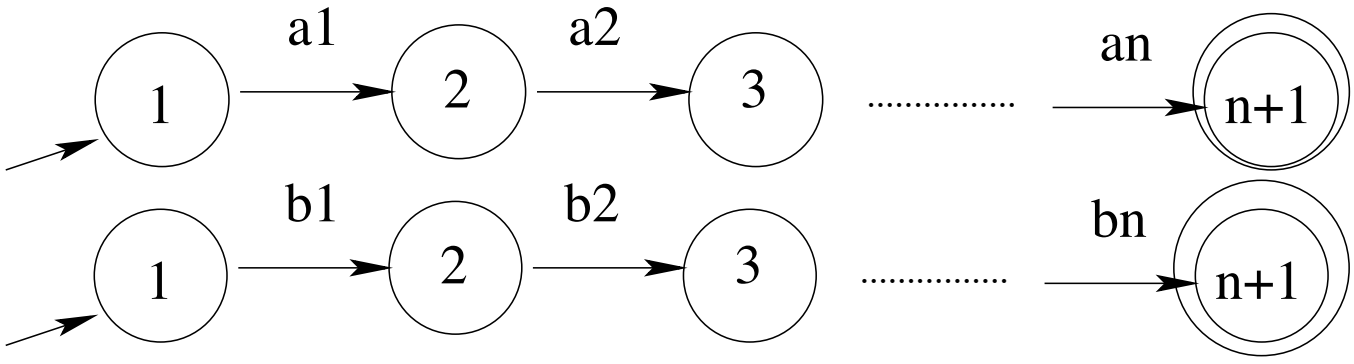
- Comment construire un automate qui reconnait deux mots ?

Quelques difficultés pratiques

- Il n'est pas "pratique" de devoir définir toutes les transitions, on aimerait :



- Comment construire un automate qui reconnait deux mots ?



Plus généralement

Pour pouvoir définir un automate qui reconnait un langage rationnel,

Il faudrait définir un automate qui puisse reconnaitre :

Plus généralement

Pour pouvoir définir un automate qui reconnait un langage rationnel,

Il faudrait définir un automate qui puisse reconnaitre :

- la réunion de langages,

Plus généralement

Pour pouvoir définir un automate qui reconnait un langage rationnel,

Il faudrait définir un automate qui puisse reconnaitre :

- la réunion de langages,
- la concaténation de langages,

Plus généralement

Pour pouvoir définir un automate qui reconnait un langage rationnel,

Il faudrait définir un automate qui puisse reconnaitre :

- la réunion de langages,
- la concaténation de langages,
- l'étoile d'un langage (fermeture de Kleene).

Introduction de non-déterministes

Non-déterministe

En informatique, **non-déterministe** est souvent associé à "plusieurs choix possibles" par opposition **déterministe** où l'opération ou l'action à effectuer est unique, *i.e.* complètement déterminé par l'état actuel du système (sans ambiguité).

Introduction de non-déterministes

Non-déterministe

En informatique, **non-déterministe** est souvent associé à "plusieurs choix possibles" par opposition **déterministe** où l'opération ou l'action à effectuer est unique, *i.e.* complètement déterminé par l'état actuel du système (sans ambiguité).

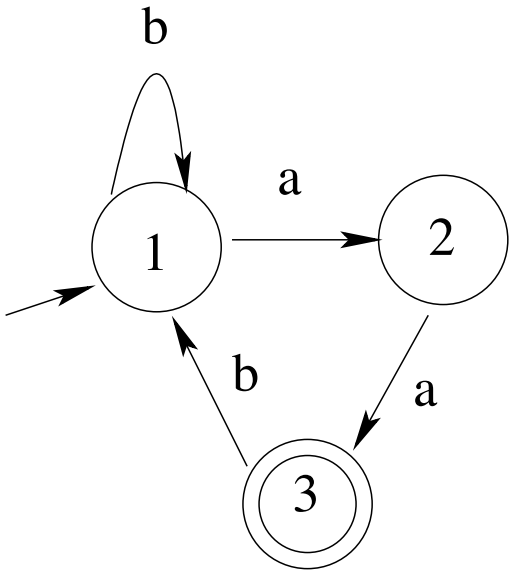
Non-déterministe dans les automates

Plusieurs sources de non-déterministe dans les automates :

- Absence de transition,
- Plusieurs transitions pour une même lettre.
- Plusieurs états initiaux
- des transitions sur des mots vides : ϵ-transitions

Absence de transition

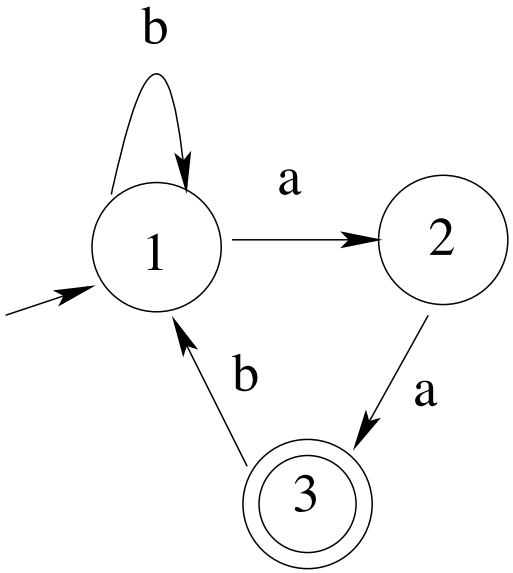
lecture de *bbba*



S'il n'y a plus de transition possible
et que le mot est encore en cours de lecture
Alors le mot est refusé

Absence de transition

lecture de *bbba*

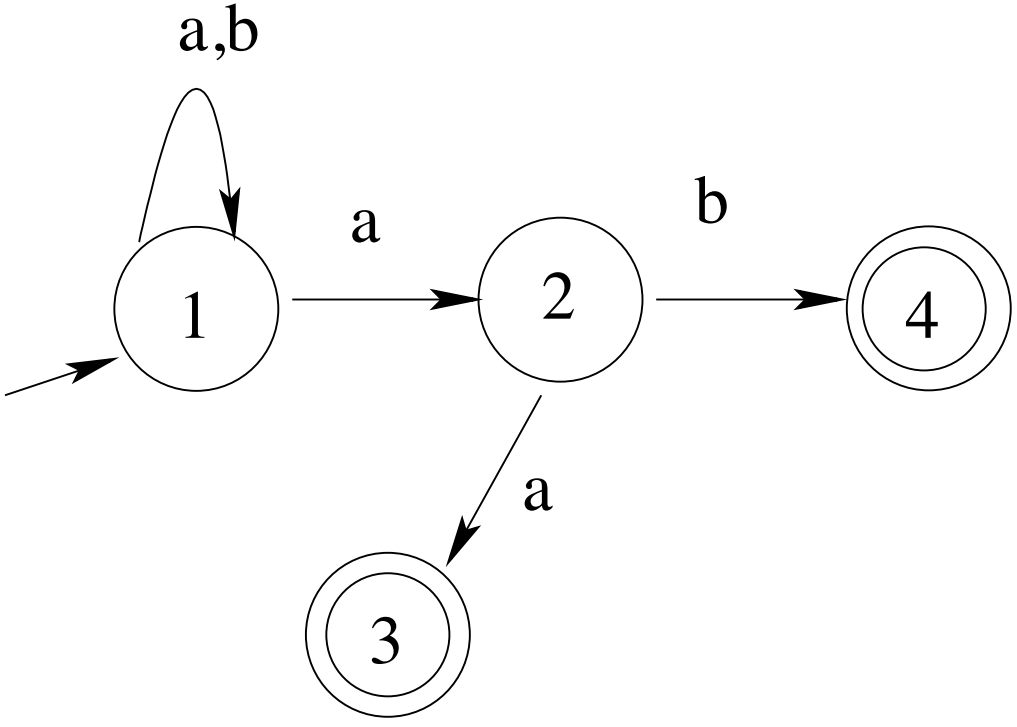


S'il n'y a plus de transition possible
et que le mot est encore en cours de lecture
Alors le mot est refusé

Remarque : remplace la technique de l'état "puit"

Plusieurs transitions

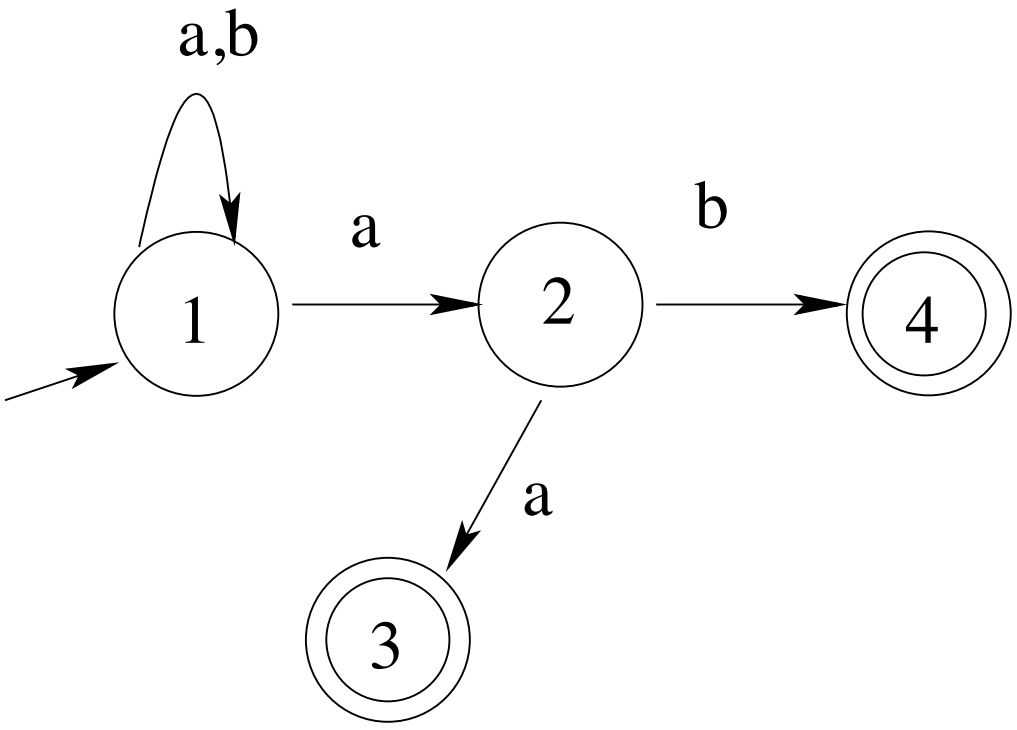
Lecture de *aabaabab* et de *aaaaaba*



Le mot est accepté lorsqu'il existe au moins une lecture menant à un état acceptant.

Plusieurs transitions

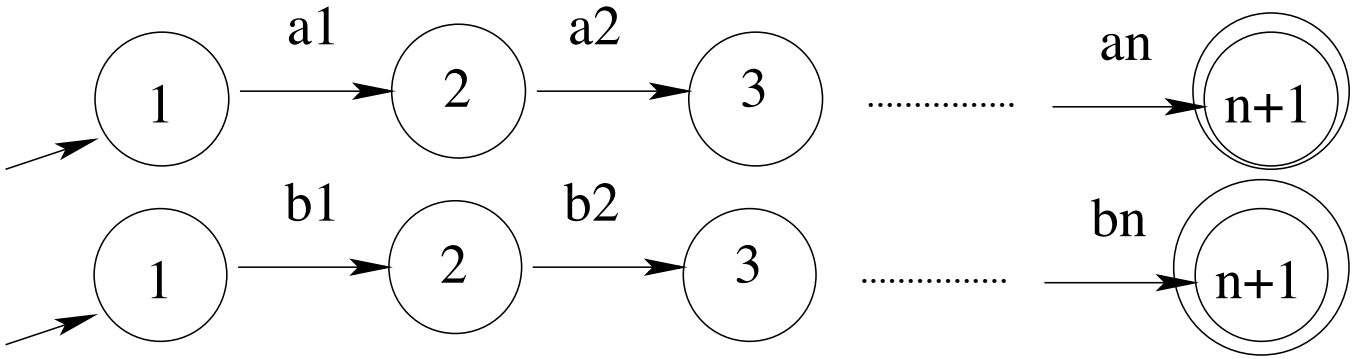
Lecture de *aabaabab* et de *aaaaaba*



Le mot est accepté lorsqu'il existe au moins une lecture menant à un état acceptant.

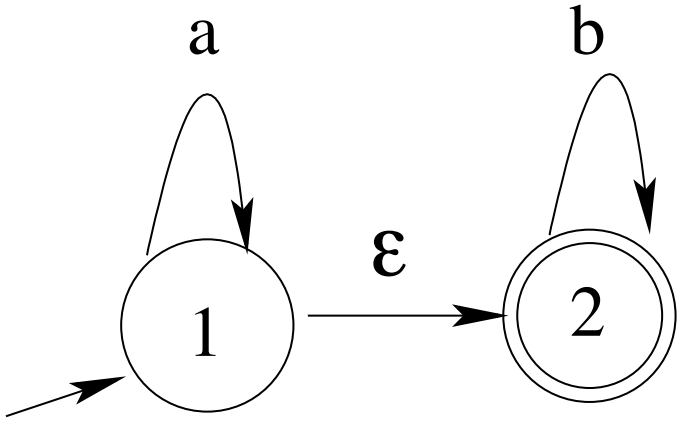
Conséquence : Il faut essayer tous les lectures possibles pour est sûr que le mot est refusé

Plusieurs états initiaux



Le mot est accepté lorsqu'il existe une lecture à partir de l'un des états initiaux menant à un état acceptant.

ϵ-transistions



Une ϵ-transistion est une transition par lecture du mot vide.

Pendant la lecture d'un mot, il est possible de choisir d'effectuer la transition ϵ sans lire aucune lettre.

Définition AFN

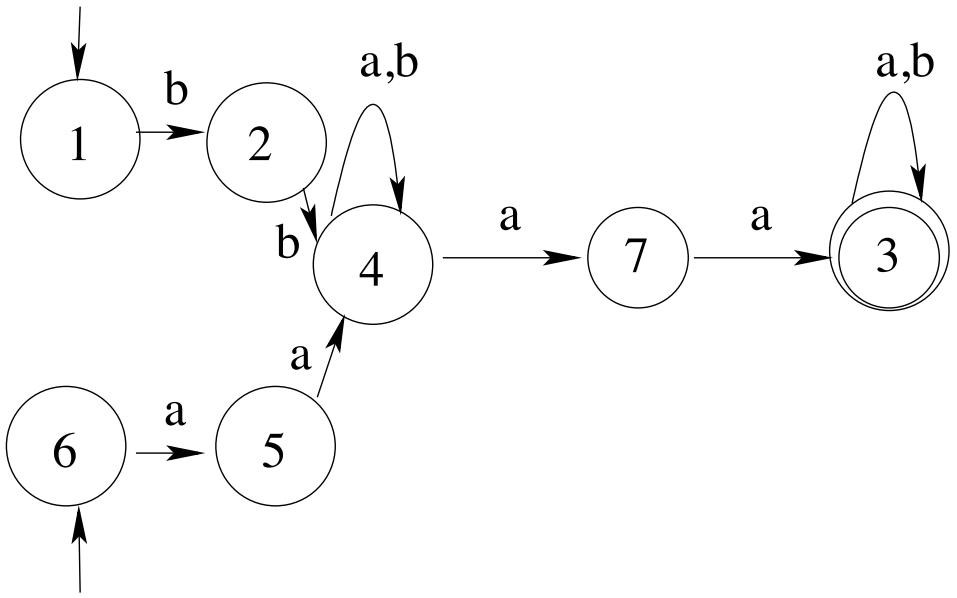
Automate Fini Non-déterministe (AFN)

Un **Automate Fini Non-déterministe** est un quintuplet (Q, Σ, T, I, A) où :

- Σ est l'*alphabet* de l'automate,
- Q un ensemble fini appelé *ensemble des états* de l'automate,
- T est une application de $Q \times \Sigma$ dans $\mathcal{P}(Q)$, appelée la *fonction de transition*
- I est un sous-ensemble de Q, appelé l'**ensemble** *des états initiaux*
- A est un sous-ensemble de Q, appelé l'*ensemble des états acceptants*.

Exemple

		a	b
→	1	-	2
	2	-	4
	3	3	3
	4	4,7	4
	5	4	-
→	6	5	-
	7	3	-



Lecture / reconnaissance

Lecture

Soient $M = (Q, \Sigma, T, I, A)$ un AFN et $u = x_1 x_2 \dots x_l$ un mot sur Σ.

Une lecture de u par M est une suite d'états $(q_0, q_1, \dots, q_l)$ vérifiant :

i $q_0 \in I$, et

ii $q_i \in T(q_{i-1}, x_i)$ pour $1 \leq i \leq l$.

Lecture / reconnaissance

Lecture

Soient $M = (Q, \Sigma, T, I, A)$ un AFN et $u = x_1 x_2 \dots x_l$ un mot sur Σ.

Une lecture de u par M est une suite d'états $(q_0, q_1, \dots, q_l)$ vérifiant :

i $q_0 \in I$, et

ii $q_i \in T(q_{i-1}, x_i)$ pour $1 \leq i \leq l$.

Acceptation

Le mot u est accepté par M s'il existe au moins une lecture de u par M qui se termine par un état acceptant.

Equivalence déterministe / non-déterministe

Définition équivalence

Soient M et $M^{'}$ deux automates. On dit que M et $M^{'}$ sont équivalents s'ils acceptent et refusent exactement les mêmes mots.

Equivalence déterministe / non-déterministe

Définition équivalence

Soient M et $M^{'}$ deux automates. On dit que M et $M^{'}$ sont équivalents s'ils acceptent et refusent exactement les mêmes mots.

Equivalence : Déterministe $\Rightarrow$ Non-déterministe

L'automate déterministe $M = (Q, \Sigma, T, q_0, A)$ est équivalent à l'automate non-déterministe $M = (Q, \Sigma, T^{'}, \{q_0\}, A)$ avec $T^{'}(q, x) = \{T(q, x)\}$.

Equivalence déterministe / non-déterministe

Equivalence : Non-déterministe ⇒ Déterministe (admis)

Soient $M = (Q, \Sigma, T, I, A)$ un AFN. Alors M est équivalent l'AFD M' définit par $M' = (\mathcal{P}(Q), \Sigma, T', I, A')$ avec :

- $T'(X, x) = \cup_{q \in X} T(q, x)$
- $A' = \{X \in \mathcal{P}(Q) \mid X \cap A \neq \emptyset\}$

Remarques :

- Un état dans l'automate déterministe est un ensemble.
- Un état pour M' est acceptant lorsqu'il contient un état acceptant pour M.

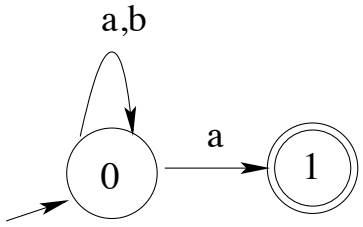
Algorithme de déterminisation

Les mots se terminant par a :

Algorithme de déterminisation

Les mots se terminant par a :

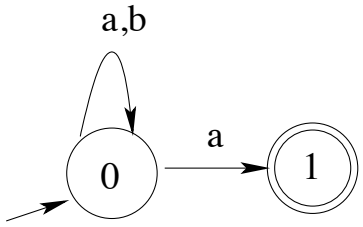
		a	b
$\rightarrow$	0	0,1	0
	$\boxed{1}$	-	-



Algorithme de déterminisation

Les mots se terminant par a :

		a	b
$\rightarrow$	0	0,1	0
	1	-	-



AFD équivalent :

		a	b
$\rightarrow$	0	0,1	0
	0,1	0,1	0

On part de l'état initial et pour chaque état suivant, on réunit l'ensemble des états atteignables depuis cet état.

Algorithme de déterminisation

		a	b
$\rightarrow$	0	0	2
$\rightarrow$	1	2	1
	$\boxed{2}$	-	-

On regroupe les états initiaux dans un même ensemble :

Algorithme de déterminisation

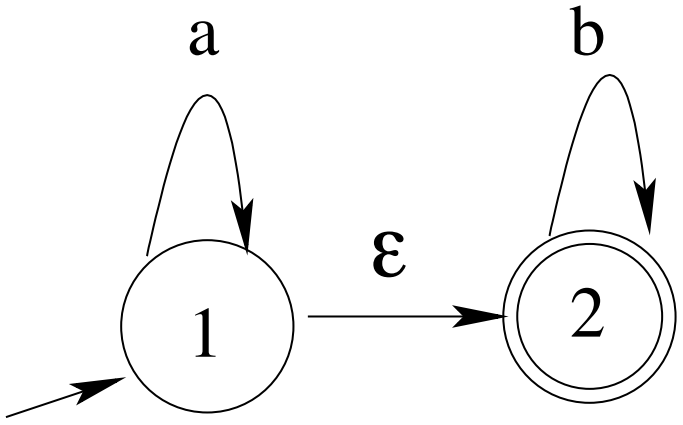
		a	b
$\rightarrow$	0	0	2
$\rightarrow$	1	2	1
	$\boxed{2}$	-	-

On regroupe les états initiaux dans un même ensemble :

		a	b
$\rightarrow$	0,1	0,2	1,2
	$\boxed{0,2}$	0	2
	$\boxed{1,2}$	2	1
	0	0	2
	$\boxed{2}$	$\square$	$\square$
	1	2	1
	$\square$	$\square$	$\square$

$\square$ est un état "puit"

Et les ϵ-transistions ?



Une ϵ-transistion est une transition sur un mot vide.

Pendant la lecture d'un mot, il est possible de choisir d'effectuer la transition ϵ sans lire aucune lettre.

Définition $\Sigma_@$

Définir les ϵ-transitions consiste à définir un alphabet $\Sigma_@$ où il existe une lettre supplémentaire correspondant à ϵ.

Définition de $\Sigma_@$

$$\Sigma_@ = \Sigma \cup \{@\}$$

Notons $\pi_@ : \Sigma_@^* \to \Sigma^*$ la fonction (projection) qui remplace :

- chaque lettre de Σ par elle-même
- @ par le mot vide.

Exemple

Si $\Sigma = \{a, b\}$ alors $\Sigma_@ = \{a, b, @\}$
et $\pi_@(@aa@b@@b) = aabb$

Remarque : @ représente le mot vide ϵ, $\pi_@(u)$ sous-mot de u

Définition AFN_ϵ

AFN_ϵ

Un **Automate Fini Non-déterministe avec ϵ-transitions** est un quintuplet (Q, Σ, T, I, A) où :

- Σ est l'*alphabet* de l'automate,
- Q un ensemble fini appelé *ensemble des états* de l'automate,
- T est une application de $Q \times \Sigma_@$ dans $\mathcal{P}(Q)$, appelée la *fonction de transition*
- I est un sous-ensemble de Q, appelé l'*ensemble des états initiaux*
- A est un sous-ensemble de Q, appelé l'*ensemble des états acceptants*.

Acceptation

Acceptation

Un mot u sur Σ est accepté par l'AFN$_\epsilon$ (Q, Σ, T, I, A) s'il existe au moins un mot $u_@$ sur $\Sigma_@$ qui est accepté par l'AFN $(Q, \Sigma_@, T, I, A)$ et tel que $u = \pi_@(u_@)$.

Intuitivement, un mot est accepté s'il existe un parcours de l'automate avec ϵ-transitions "spontanées"

Equivalence AFN_ϵ / AFN

Equivalence (admis)

Soient $M = (Q, \Sigma, T, I, A)$ un AFN_ϵ. Alors M est équivalent l'AFN M' définit par $M' = (Q, \Sigma, T', I', A')$ avec :

- $T'(q, x) = \cup_{q' \in cl(q)} T(q', x)$
- $I' = \cup_{q \in I} cl(q)$
- $A' = \{q \mid cl(q) \cap A \neq \emptyset\}$

$cl(q)$ est la cloture (union des itérés) de q par ϵ-transitions, c'est-à-dire l'ensemble des états atteignables par ϵ-transitions itérées (cf. suite).

Equivalence AFN_ϵ / AFN

Cloture de q

$cl(q)$ est la cloture de q par ϵ-transitions, c'est-à-dire l'ensemble des états atteignables par ϵ-transitions itérées.

Cloture : Définition ascendante

$$cl(q) = \cup_{i \in \mathbb{N}} X_i$$

avec :

- $X_0 = \{q\}$
- $X_{i+1} = X_i \cup \{q' : q' = T(q_i, @) \text{ avec } i \in X_i\}$

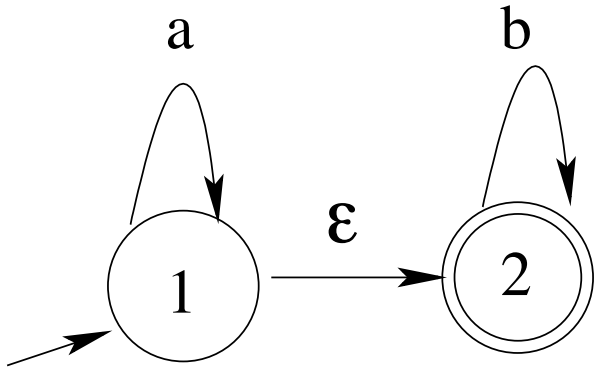
Cloture : Définition descendante

$$cl(q) = \cap\{X \ : \ q \in X \text{ et } X \text{ stable par } \epsilon\text{-transistion}\}$$

On dit que $X \subset \mathcal{P}(Q)$ est stable par ϵ-transition si $T(X, @) \subset X$.

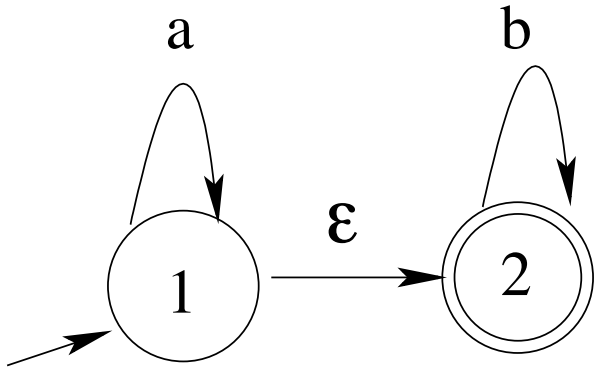
Algorithme de déterminisation

		a	b	ϵ
$\rightarrow$	1	1	-	2
	2	-	2	-



Algorithme de déterminisation

		a	b	ϵ
$\rightarrow$	1	1	-	2
	2	-	2	-



AFN équivalent :

$cl(1) = \{1, 2\}$

		a	b
$\rightarrow$	**1**	1	2
$\rightarrow$	**2**	-	2

Question

Quel rapport entre les langages reconnus par un Automate Fini et les langages décrits par une expression régulière (langage rationnel) ?

Théorème de Kleene

Théorème de Kleene (admis...)

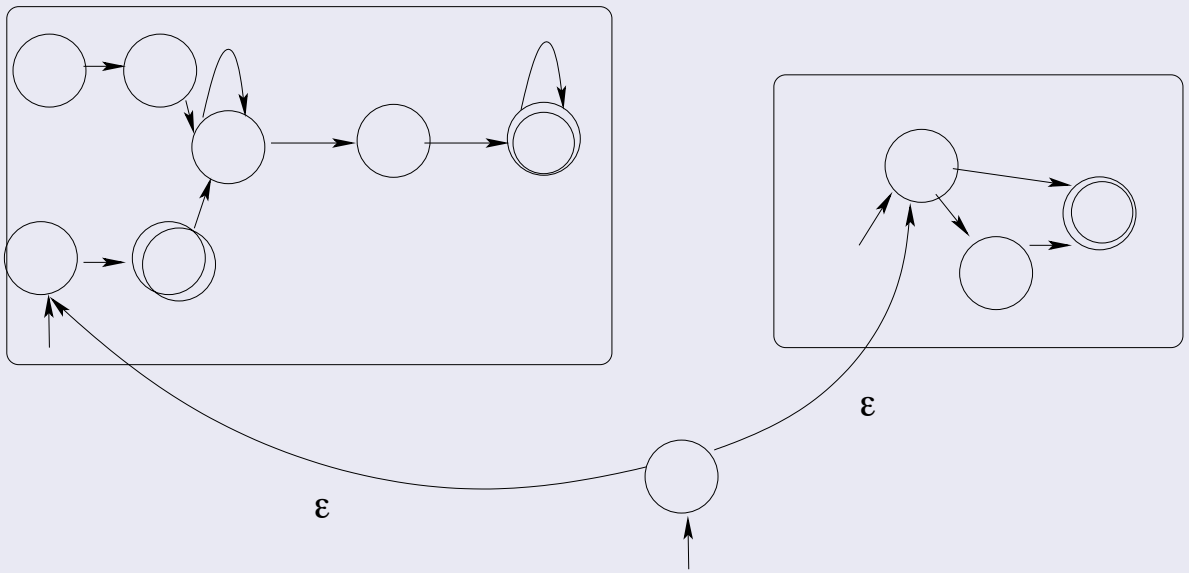
Un langage sur un alphabet Σ est rationnel si et seulement si il est reconnu par un automate fini.

Idée de la démonstration :
On peut construire de manière inductive l'ensemble des langages rationnels et les automates reconnaissant ces langages.

Union de langages rationnels

Soient deux automates finis déterministes M_1 et M_2 reconnaissant respectivement les langages L_1 et L_2

$L_1 \cup L_2$ est reconnu par :

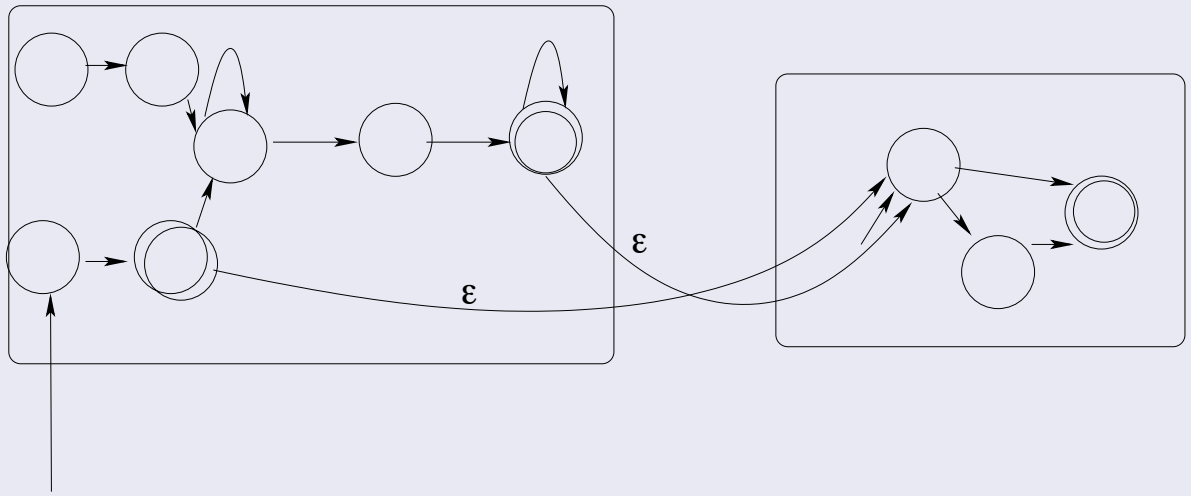


On ajoute des ϵ-transitions entre un nouvel état initial et les états initiaux de M_1 et de M_2

Produit de concaténation de langages rationnels

Soient deux automates finis déterministes M_1 et M_2 reconnaissant respectivement les langages L_1 et L_2

$L_1.L_2$ est reconnu par :

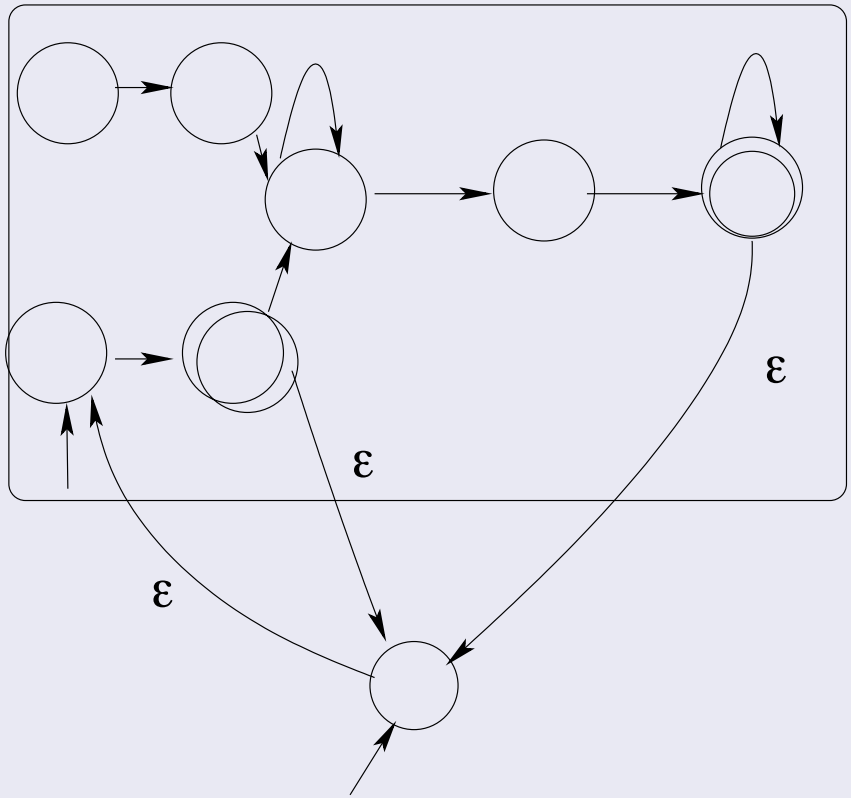


On ajoute des ϵ-transitions entre les états acceptants de M_1 et l'état intial de M_2

Etoile (cloture de Kleene) de langages rationnels

Soit un automate fini déterministe M reconnaissant le langage L

L^* est reconnu par :



On ajoute des ϵ-transitions entre les états finaux et le nouvel état initial

Conclusion (1)

- A chaque **langage rationnel** est associé
  un **automate fini**, et réciproquement.
- Les automates sont des machines abstraites capables de réaliser des calculs sur des mots :
  entrée : mot (donnée du problème)
  sortie : oui/non (une décision)
- Lien très fort entre langage et machine :
  - Langage : définit un ensemble de mots
  - Machine : calcul un ensemble de mots

Conclusion (2)

- Il est possible de définir d'autres machines abstraites qui permettent de définir d'autres classes de langages.

  L'expressivité du langage et la capacité de calcul de la machine sont alors différentes.
- Les questions que l'on se pose sont alors les mêmes :
  - mode de lecture,
  - description algébrique langage (souvent à l'aide d'une définition inductive),
  - équivalence avec d'autres classes de langages,
  - complexité de calcul d'une machine reconnaissant le langage.

Par exemple, on peut remplacer automate par machine de Turing...