Automates, Langages et Logique

2. Automates finis déterministes

L2, Université d'Orléans — S1 2025/2026

Nicolas Ollinger

Rappels

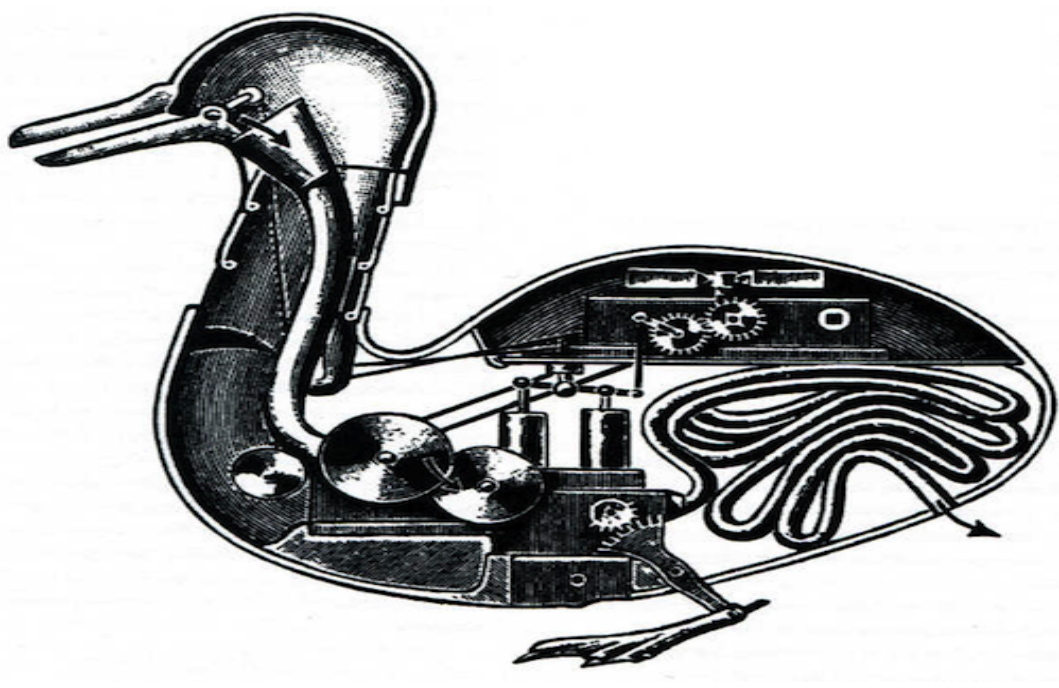
Les **mots** sont des suites notées en juxtaposant leurs **lettres**.

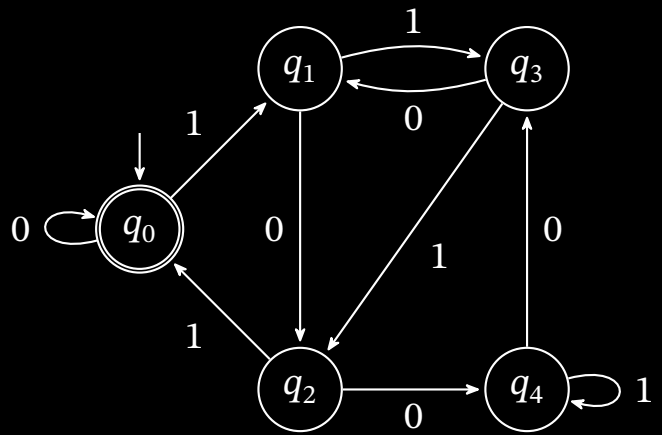
ε	le mot vide	$\lvert\varepsilon\rvert = 0$
$abaab$	le mot (a, b, a, a, b)	$\lvert abaab\rvert = 5$

Les **expressions rationnelles** décrivent des **langages** à l'aide des opérations **produit**, **somme** et **étoile**. La notation est simplifiée pour la rendre plus lisible :

$$E, F \coloneqq \emptyset \mid x \mid EF \mid E + F \mid E^* \mid (E)$$

Exemple $(a + b)^* aba^*$




q_0
q_1
q_2
q_3
q_4
0
1
0
1
0
1
0
1
0
1

Reconnaisseurs de langages

Un **reconnaisseur de langage** est un dispositif abstrait sur un alphabet A.

Reconnaisseurs de langages

Un **reconnaisseur de langage** est un dispositif abstrait sur un alphabet A.

Il **accepte** les mots d'un **langage** L et **rejette** les mots de $A^* \setminus L$.

Reconnaisseurs de langages

Un **reconnaisseur de langage** est un dispositif abstrait sur un alphabet A.

Il **accepte** les mots d'un **langage** L et **rejette** les mots de $A^* \setminus L$.

Pour l'utiliser on commence par l'**initialiser**. Puis on lui transmet les lettres du mot u à reconnaître les unes à la suite des autres $(u_0, u_1, \cdots, u_{n-1})$. Enfin, on observe si le reconnaisseur **accepte** ou **rejette** le mot.

Reconnaisseurs de langages

Un **reconnaisseur de langage** est un dispositif abstrait sur un alphabet A.

Il **accepte** les mots d'un **langage** L et **rejette** les mots de $A^* \setminus L$.

Pour l'utiliser on commence par l'**initialiser**. Puis on lui transmet les lettres du mot u à reconnaître les unes à la suite des autres $(u_0, u_1, \cdots, u_{n-1})$. Enfin, on observe si le reconnaisseur **accepte** ou **rejette** le mot.

Si le dispositif est **déterministe**, il est toujours dans la même **configuration** après avoir lu la même séquence de lettres. Son **état** est entièrement déterminé par la suite des opérations effectuées depuis l'initialisation.

Reconnaisseurs de langages

Un **reconnaisseur de langage** est un dispositif abstrait sur un alphabet A.

Il **accepte** les mots d'un **langage** L et **rejette** les mots de $A^* \setminus L$.

Pour l'utiliser on commence par l'**initialiser**. Puis on lui transmet les lettres du mot u à reconnaître les unes à la suite des autres $(u_0, u_1, \cdots, u_{n-1})$. Enfin, on observe si le reconnaisseur **accepte** ou **rejette** le mot.

Si le dispositif est **déterministe**, il est toujours dans la même **configuration** après avoir lu la même séquence de lettres. Son **état** est entièrement déterminé par la suite des opérations effectuées depuis l'initialisation.

Le **langage reconnu** par le dispositif est l'ensemble des **mots acceptés**.

Automate fini déterministe (AFD)

Définition Un **automate fini déterministe** (noté **AFD** dans la suite) $\mathcal{A}$ est un quintuplet (Q, A, δ, q_0, F) avec

- Q l'ensemble fini des **états** ;
- A l'**alphabet** d'entrée ;
- $\delta : Q \times A \to Q \cup \{\bot\}$ la **fonction partielle de transition** ;
- $q_0 \in Q$ l'**état initial** ;
- $F \subseteq Q$ l'ensemble des **états acceptants** de l'automate.

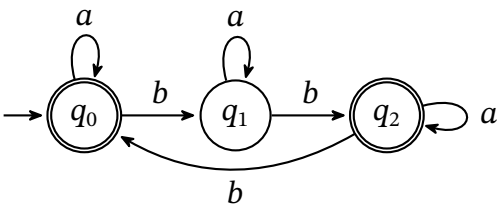
Automate fini déterministe (AFD)

Définition Un **automate fini déterministe** (noté **AFD** dans la suite) $\mathcal{A}$ est un quintuplet (Q, A, δ, q_0, F) avec

- Q l'ensemble fini des **états** ;
- A l'**alphabet** d'entrée ;
- $\delta : Q \times A \to Q \cup \{\bot\}$ la **fonction partielle de transition** ;
- $q_0 \in Q$ l'**état initial** ;
- $F \subseteq Q$ l'ensemble des **états acceptants** de l'automate.

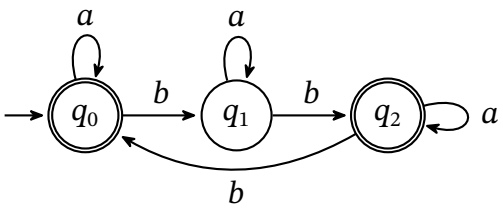
Un AFD est **complet** si sa fonction de transition est **totale**, c'est-à-dire définie pour toute paire état/lettre.

3 modes de représentations



représentation graphique

3 modes de représentations

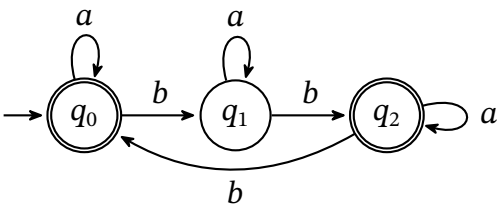


représentation graphique

	a	b
$\rightarrow q_0$ *	q_0	q_1
q_1	q_1	q_2
q_2 *	q_2	q_0

représentation tabulaire

3 modes de représentations



représentation graphique

	a	b
$\rightarrow q_0 *$	q_0	q_1
q_1	q_1	q_2
$q_2 *$	q_2	q_0

tabulaire — états en lignes

	$\downarrow q_0 *$	q_1	$q_2 *$
a	q_0	q_1	q_2
b	q_1	q_2	q_0

tabulaire — états en colonnes

Fonction de transition étendue

La **fonction de transition** est étendue des **lettres** aux **mots** par récurrence.

La **fonction de transition étendue** applique successivement la fonction de transition aux différentes lettres qui composent le mot.

Fonction de transition étendue

La **fonction de transition** est étendue des **lettres** aux **mots** par récurrence.

La **fonction de transition étendue** applique successivement la fonction de transition aux différentes lettres qui composent le mot.

Pour tout **état** $q \in Q$, **lettre** $x \in A$ et **mot** $u \in A^*$, on pose :

$$\delta^*(q, \varepsilon) = q$$

$$\delta^*(q, u \cdot x) = \begin{cases} \delta(\delta^*(q, u), x) & \text{si } \delta^*(q, u) \neq \bot \\ \bot & \text{sinon} \end{cases}$$

Fonction de transition étendue

La **fonction de transition** est étendue des **lettres** aux **mots** par récurrence.

La **fonction de transition étendue** applique successivement la fonction de transition aux différentes lettres qui composent le mot.

Pour tout **état** $q \in Q$, **lettre** $x \in A$ et **mot** $u \in A^*$, on pose :

$$\delta^*(q, \varepsilon) = q$$

$$\delta^*(q, u \cdot x) = \begin{cases} \delta(\delta^*(q, u), x) & \text{si } \delta^*(q, u) \neq \bot \\ \bot & \text{sinon} \end{cases}$$

Intuitivement, $\delta^*(q, u)$ est l'**état atteint** en partant de l'état q et en lisant le mot u dans l'automate.

Chemin dans un automate

La **fonction de transition** δ décrit les transitions $q \xrightarrow{a} \delta(q, a)$ possibles dans l'automate.

Chemin dans un automate

La **fonction de transition** δ décrit les transitions $q \xrightarrow{a} \delta(q, a)$ possibles dans l'automate.

Définition Un **chemin** dans un automate (Q, A, δ, q_0, F) est une suite finie de transitions successives $p_0 \xrightarrow{x_0} p_1 \xrightarrow{x_1} \cdots \xrightarrow{x_{n-1}} p_n$ où p_0 est l'**état de départ**, p_n est l'**état d'arrivée** et $\delta(p_i, x_i) = p_{i+1}$ pour tout $i < n$. Le mot $x_0 \cdots x_{n-1}$ est l'**étiquette** du chemin.

Chemin dans un automate

La **fonction de transition** δ décrit les transitions $q \xrightarrow{a} \delta(q,a)$ possibles dans l'automate.

Définition Un **chemin** dans un automate (Q, A, δ, q_0, F) est une suite finie de transitions successives $p_0 \xrightarrow{x_0} p_1 \xrightarrow{x_1} \cdots \xrightarrow{x_{n-1}} p_n$ où p_0 est l'**état de départ**, p_n est l'**état d'arrivée** et $\delta(p_i, x_i) = p_{i+1}$ pour tout $i < n$. Le mot $x_0 \cdots x_{n-1}$ est l'**étiquette** du chemin.

En utilisant la **fonction de transition généralisée** on s'autorise une notation plus concise $p_0 \xrightarrow{x_0 x_1 \cdots x_{n-1}} p_n = \delta^*(p_0, x_0 x_1 \cdots x_{n-1})$.

Exercice Montrer que si $\delta^*(q,u) \neq \bot$ alors

$$\delta^*(q, u \cdot v) = \delta^*(\delta^*(q,u), v) \quad .$$

Exercice En déduire que $q \xrightarrow{\varepsilon} q$ pour tout $q \in Q$ et que si $q \xrightarrow{u} q'$ et $q' \xrightarrow{v} q''$ alors $q \xrightarrow{u \cdot v} q''$.

Langage reconnu

Définition Un chemin est **acceptant** pour un AFD lorsque l'état de départ est l'**état initial** et l'état d'arrivée un **état acceptant**.

$$q_0 \xrightarrow{u} q = \delta^*(q_0, u) \in F$$

Langage reconnu

Définition Un chemin est **acceptant** pour un AFD lorsque l'état de départ est l'**état initial** et l'état d'arrivée un **état acceptant**.

$$q_0 \xrightarrow{u} q = \delta^*(q_0, u) \in F$$

Définition Un mot u est **accepté** par un automate $\mathcal{A}$ s'il est l'étiquette d'un **chemin acceptant** de $\mathcal{A}$.

Langage reconnu

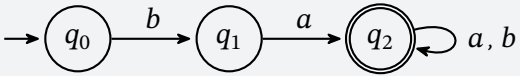
Définition Un chemin est **acceptant** pour un AFD lorsque l'état de départ est l'**état initial** et l'état d'arrivée un **état acceptant**.

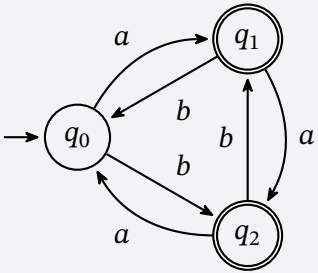
$$q_0 \xrightarrow{u} q = \delta^*(q_0, u) \in F$$

Définition Un mot u est **accepté** par un automate $\mathcal{A}$ s'il est l'étiquette d'un **chemin acceptant** de $\mathcal{A}$.

Définition Le **langage reconnu** par un automate $\mathcal{A}$ est l'ensemble $L(\mathcal{A})$ des **mots acceptés** par $\mathcal{A}$.

Exercice Décrire les langages reconnus par les automates finis déterministes suivants.





Complétion

Définition Deux automates $\mathcal{A}$ et $\mathcal{B}$ sont **équivalents** s'ils reconnaissent le même langage, *i.e.* si $L(\mathcal{A}) = L(\mathcal{B})$.

Complétion

Définition Deux automates $\mathcal{A}$ et $\mathcal{B}$ sont **équivalents** s'ils reconnaissent le même langage, *i.e.* si $L(\mathcal{A}) = L(\mathcal{B})$.

Un **état puits** est un état dont on ne sort jamais, *i.e* un état $q \in Q$ tel que $\delta(q, a) \in \{q, \perp\}$ pour tout $a \in A$.

Complétion

Définition Deux automates $\mathcal{A}$ et $\mathcal{B}$ sont **équivalents** s'ils reconnaissent le même langage, *i.e.* si $L(\mathcal{A}) = L(\mathcal{B})$.

Un **état puits** est un état dont on ne sort jamais, *i.e* un état $q \in Q$ tel que $\delta(q, a) \in \{q, \perp\}$ pour tout $a \in A$.

Définition Un AFD est **complet** si sa fonction de transition est **totale**.

Complétion

Définition Deux automates $\mathcal{A}$ et $\mathcal{B}$ sont **équivalents** s'ils reconnaissent le même langage, *i.e.* si $L(\mathcal{A}) = L(\mathcal{B})$.

Un **état puits** est un état dont on ne sort jamais, *i.e* un état $q \in Q$ tel que $\delta(q, a) \in \{q, \perp\}$ pour tout $a \in A$.

Définition Un AFD est **complet** si sa fonction de transition est **totale**.

Proposition Tout AFD est **équivalent** à un AFD **complet** avec au plus deux **état puits**, l'un acceptant et l'autre non.

Langage reconnaissable

On sait maintenant associer un **langage** à un **automate**…

Langage reconnaissable

On sait maintenant associer un **langage** à un **automate**…

Et dans l'autre sens ?

Langage reconnaissable

On sait maintenant associer un **langage** à un **automate**…

Et dans l'autre sens ?

Définition Un langage L sur un alphabet A est **reconnaissable** s'il existe un **automate fini déterministe** $\mathcal{A}$ qui le reconnaît, *i.e.* tel que $L(\mathcal{A}) = L$.

La famille des **langages reconnaissables** sur A est notée $\mathrm{Rec}\, A^*$.

Compétences attendues

(1) Savoir identifier un **langage reconnu** par un automate donné ;

(2) Savoir montrer qu'un langage est **reconnaissable**.

Exercice Montrer que les langages ci-dessous sont **reconnaissables** :

(1) $L_1 = \{u \in \{a,b\}^* \,|\, |u|_a \geqslant 3\}$;

(2) le langage L_2 des mots binaires (en base 2) qui codent des entiers divisibles par 4.

Langages finis

Un **langage** L est **fini** s'il contient un nombre fini de mots, *i.e.* si $|L| \in \mathbb{N}$.

Proposition Tout **langage fini** est **reconnaissable**.

Idée Construire l'automate des préfixes.

Propriétés de clôture

Une famille de langages $\mathcal{F}$ est **close** par une opération si en appliquant cette opération à des éléments de $\mathcal{F}$, on obtient un élément de $\mathcal{F}$.

Proposition La famille des **langages reconnaissables** est close par **union**, **intersection** et **passage au complémentaire**.

Idée Supposer qu'on dispose d'**AFD complets** sur un même alphabet et construire un **automate produit** dont les états sont des paires d'états des deux AFD considérés.

De l'utilité des états

Définition Un état d'un automate est **utile** s'il apparaît dans un **chemin acceptant**, *i.e.* s'il existe deux mots u et v et un état q' tels que

$$q_0 \xrightarrow{u} q \xrightarrow{v} q' \in F \quad .$$

De l'utilité des états

Définition Un état d'un automate est **utile** s'il apparaît dans un **chemin acceptant**, *i.e.* s'il existe deux mots u et v et un état q' tels que

$$q_0 \xrightarrow{u} q \xrightarrow{v} q' \in F \quad .$$

Définition Un état q d'un automate (Q, A, δ, q_0, F) est :

- **accessible** s'il existe un chemin de q_0 à q ;
- **co-accessible** s'il existe un chemin de q à un état de F.

De l'utilité des états

Définition Un état d'un automate est **utile** s'il apparaît dans un **chemin acceptant**, *i.e.* s'il existe deux mots u et v et un état q' tels que

$$q_0 \xrightarrow{u} q \xrightarrow{v} q' \in F \quad .$$

Définition Un état q d'un automate (Q, A, δ, q_0, F) est :

- **accessible** s'il existe un chemin de q_0 à q ;
- **co-accessible** s'il existe un chemin de q à un état de F.

Proposition Soit $\mathcal{A}$ un AFD à n états et m transitions. On peut calculer l'ensemble des **états accessibles/co-accessibles** en temps $O(n + m)$.

Émonder les automates

Définition Un automate est :

- **accessible** si tous ses états sont **accessibles** ;
- **co-accessible** si tous ses états sont **co-accessibles** ;
- **émondé** si tous ses états sont **utiles**.

Émonder les automates

Définition Un automate est :

- **accessible** si tous ses états sont **accessibles** ;
- **co-accessible** si tous ses états sont **co-accessibles** ;
- **émondé** si tous ses états sont **utiles**.

Proposition Tout automate $\mathcal{A}$ est équivalent à un **automate émondé** $\mathcal{A}_m$ effectivement calculable.

Exercice Soit L un langage reconnaissable, montrer que les langages des **préfixes**, **suffixes** et **facteurs** de L sont eux aussi **reconnaissables**.

Idée Supposer que L est reconnu par un AFD émondé.

Équivalence entre automates

Il existe des automates **équivalents**. Comment tester si deux AFD le sont ?

Équivalence entre automates

Il existe des automates **équivalents**. Comment tester si deux AFD le sont ?

Proposition Soit $\mathcal{A}$ un AFD à k états acceptants et m transitions. On peut **décider** si le langage reconnu par $\mathcal{A}$ est **vide** en temps $O(k+m)$.

Équivalence entre automates

Il existe des automates **équivalents**. Comment tester si deux AFD le sont ?

Proposition Soit $\mathcal{A}$ un AFD à k états acceptants et m transitions. On peut **décider** si le langage reconnu par $\mathcal{A}$ est **vide** en temps $O(k+m)$.

Proposition On peut **décider** effectivement si deux AFD **reconnaissent le même langage**.

Exercice Proposer un algorithme pour tester si le langage reconnu par un AFD est **infini**.

Idée Un langage **infini** contient des mots **arbitrairement longs**. À quoi ressemble un chemin acceptant pour de très long mots ?

Langages non reconnaissables

La famille des **langages reconnaissables** semble très riche. Existe-t-il des langages non reconnaissables ?

Langages non reconnaissables

La famille des **langages reconnaissables** semble très riche. Existe-t-il des langages non reconnaissables ? Évidemment ! Il suffit de les compter !

Langages non reconnaissables

La famille des **langages reconnaissables** semble très riche. Existe-t-il des langages non reconnaissables ? Évidemment ! Il suffit de les compter !

Proposition $\mathrm{Rec}\, A^*$ est **dénombrable** alors que $\mathcal{P}(A^*)$ ne l'est pas.

Langages non reconnaissables

La famille des **langages reconnaissables** semble très riche. Existe-t-il des langages non reconnaissables ? Évidemment ! Il suffit de les compter !

Proposition $\mathrm{Rec}\, A^*$ est **dénombrable** alors que $\mathcal{P}(A^*)$ ne l'est pas.

On peut construire des **exemples explicites**.

Langages non reconnaissables

La famille des **langages reconnaissables** semble très riche. Existe-t-il des langages non reconnaissables ? Évidemment ! Il suffit de les compter !

Proposition $\mathrm{Rec}\, A^*$ est **dénombrable** alors que $\mathcal{P}(A^*)$ ne l'est pas.

On peut construire des **exemples explicites**.

Proposition Le langage $\{a^n b^n | n \in \mathbb{N}\}$ n'est **pas reconnaissable**.

Langages non reconnaissables

La famille des **langages reconnaissables** semble très riche. Existe-t-il des langages non reconnaissables ? Évidemment ! Il suffit de les compter !

Proposition $\mathrm{Rec}\, A^*$ est **dénombrable** alors que $\mathcal{P}(A^*)$ ne l'est pas.

On peut construire des **exemples explicites**.

Proposition Le langage $\{a^n b^n | n \in \mathbb{N}\}$ n'est **pas reconnaissable**.

Exercice Il existe dans la littérature de nombreuses variantes du lemme suivant. Démontrez la version proposée ici.

Lemme de l'étoile Si $L \in \mathrm{Rec}\, A^*$ alors il existe un entier N tel que pour tout mot $uvw \in L$ avec $|v| \geqslant N$, il existe une factorisation $v = u'v'w'$ avec $|v'| > 0$ telle que $uu'v'^*w'w \subseteq L$.

Idée Raisonner sur les chemins acceptants d'un AFD reconnaissant L.