INF 302 : Langages & Automates

Chapitre 1b : Automates à États Finis Déterministes
— opérations sur les automates et fermeture des langages à états

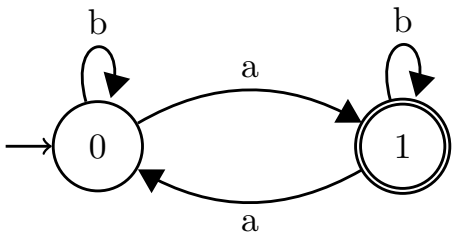
Yliès Falcone

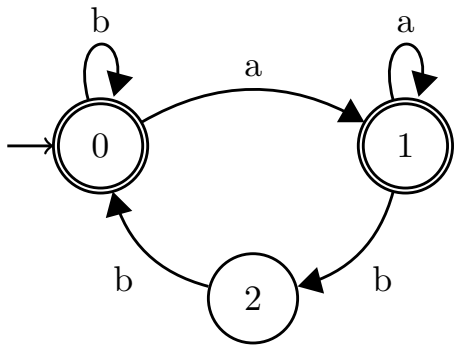
ylies.falcone@univ-grenoble-alpes.fr — www.ylies.fr

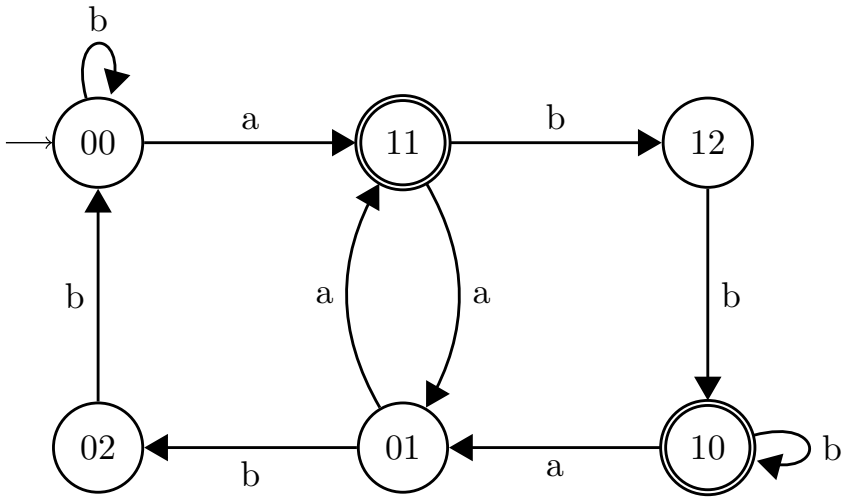
Univ. Grenoble-Alpes, Inria
-
Laboratoire d'Informatique de Grenoble - www.liglab.fr
Équipe de recherche LIG-Inria, Corse - team.inria.fr/corse/

Année Académique 2018 - 2019

Intuition et objectifs



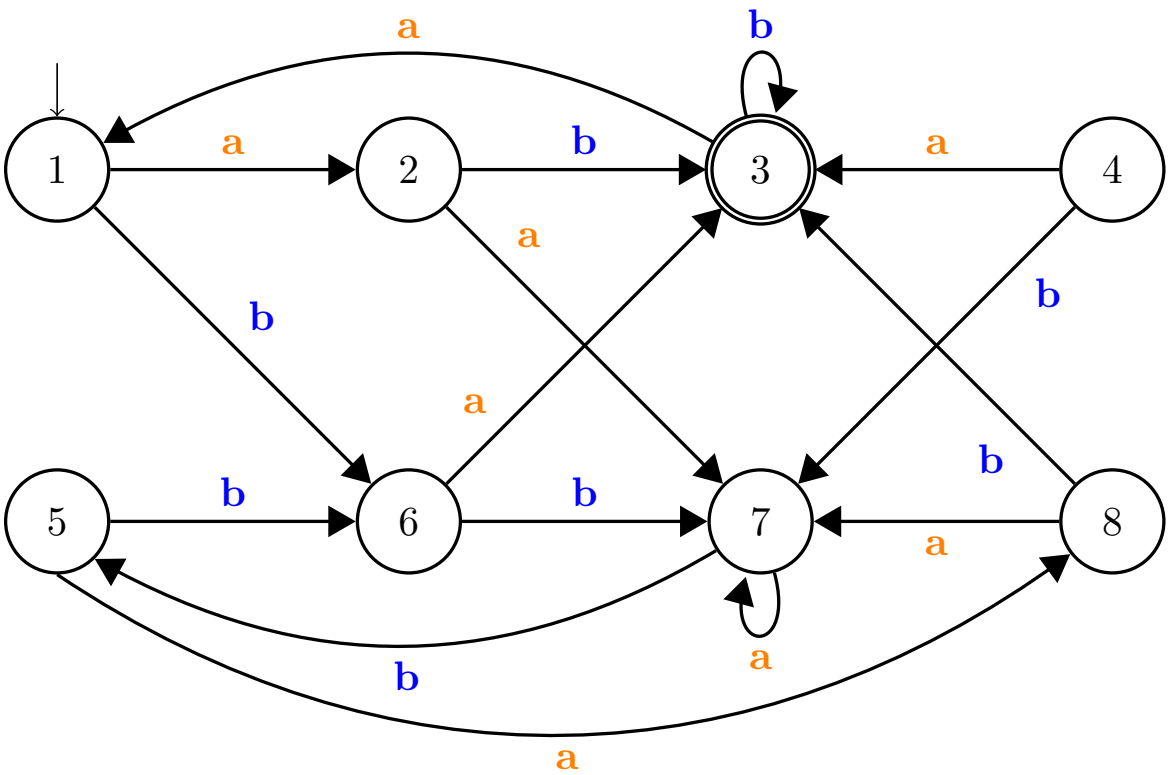




- Ingrédients de base : états (accepteurs), symboles, transitions — *syntaxe*.
- Exécution, mot accepté, langage accepté — *sémantique*.
- Problèmes de décision : langage vide, langage infini.
- Opérations sur automate/opérations sur langage : négation/complémentation, produit/intersection.

Plan Chap. 1b - Automates à États Finis Déterministes

Équivalence et Minimisation : motivations par un exemple



Questions

- Quels états peuvent être "distingués" ?
- Quels états sont "équivalents" ?

De manière plus générale :

- Peut-on définir une équivalence entre états ?
- Peut-on dire si des automates sont équivalents ?
- Peut-on avoir une représentation « canonique » (on dira minimale) d'un automate ?

Équivalence/distinguabilité sont reliées à la notion d'**acceptation**.

Plan Chap. 1b - Automates à États Finis Déterministes

Distinguabilité entre états

Soit $A = (Q, \Sigma, \delta, q_{\text{init}}, F)$ un AEFD dont tous les états sont accessibles.

"Deux états sont distinguables s'il existe un mot qui, à partir de l'un des états, mène à un état accepteur, et à partir de l'autre état, mène à un état non accepteur."

Définition (Relation de distinguabilité entre états)

La relation de distinguabilité $\neq$ entre états sur Q est définie par :

$$\forall p, q \in Q : \quad \Big(p \neq q \quad \text{ssi} \quad \exists u \in \Sigma^* : \big(\delta^*(p, u) \in F \iff \delta^*(q, u) \in F \big) \Big)$$

Théorème : à propos de la relation de distinguabilité

La relation de distinguabilité $\neq$ entre états de Q est :

- anti-réflexive : $\forall q \in Q : \neg(q \neq q)$,
- symétrique : $\forall p, q \in Q : p \neq q \implies q \neq p$.

Distinguabilité entre états **à k pas**

Question

Comment calculer $\neq$?

- Limiter la relation de distinguabilité à k symboles.
- Calculer $\neq$ de manière itérative.

Définition (Distinguabilité à k pas) Par recurrence

Pour chaque $k \in \mathbb{N}$, on introduit la relation $\neq_k$ sur Q :

1. $q \neq_0 q'$ ssi $q \in F \not\Leftrightarrow q' \in F$.
2. Pour $k \in \mathbb{N}$, $p \neq_{k+1} q$ ssi

$$(p \neq_k q) \vee (\exists a \in \Sigma : \delta(p, a) \neq_k \delta(q, a)) .$$

+ simple

Construction de $\neq$ à partir de $\neq_k$

C'est + clair !

Lemme

Pour tout $k \in \mathbb{N}$, $q \neq_k q'$ ssi

$$\exists u \in \Sigma^* : |u| \leq k \wedge \left(\delta^*(q,u) \in F \iff \delta^*\left(q',u\right) \in F\right).$$

Corollaire

$$\bigcup_{k \in \mathbb{N}} \neq_k \quad = \quad \neq$$

Nous pouvons en déduire un algorithme de calcul des états distinguables.

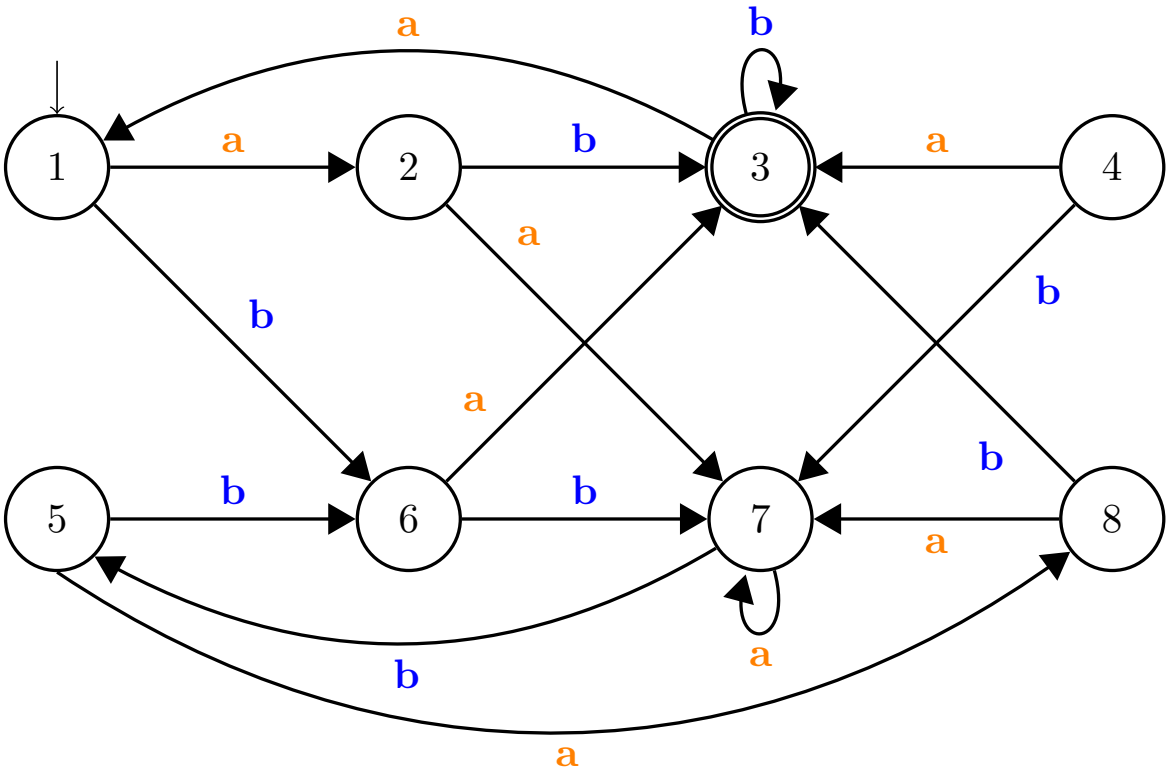
Distinguabilité entre états

Algorithme 1 Calcul des états distinguables

Entrée : $A = (Q, \Sigma, \delta, q_{\text{init}}, F)$ un AEFD dont tous les états sont accessibles
Sortie : $D \subseteq Q \times Q$ relation de distinguabilité entre états de Q

1: **ensemble de** couples d'états D, D_{pre} ; (* D contient les couples d'états distinguables *)
(* D_{pre} est la valeur de D à l'itération précédente *)
2: **ensemble de** couples d'états X ;
3: $D := \Big(F \times (Q \setminus F)\Big) \cup (Q \setminus F) \times F\Big)$; (* D initialisé avec états accepteurs/non accepteurs *)
4: $D_{pre} := \emptyset$; (* maj de D_{pre} *)
5: **tant que** $D_{pre} \neq D$ **faire**
6: $D_{pre} := D$;
7: $X := \Big\{(p,q),(q,p) \in Q \times Q \mid \exists a \in \Sigma : \Big(\delta(p,a), \delta(q,a)\Big) \in D)\Big\}$;
(* calcul des nouveaux états distinguables *)
8: $D := D \cup X$; (* ajouter les états distinguables à D *)
9: **fin tant que**
10: **retourner** D ;

Distinguabilité entre états : exemple



	1	2	3	4	5	6	7
2	x						
3	x	x					
4	x	x	x				
5		x	x	x			
6	x	x	x		x		
7	x	x	x	x	x	x	
8	x		x	x	x	x	x

Distinction entre états : correction de l'algorithme

Soit $A = (Q, \Sigma, \delta, q_{\text{init}}, F)$ un AEFD dont tous les états sont accessibles.

Théorème : Correction de l'algorithme de distinction entre états

- L'algorithme distingue *uniquement* des états distinguables.
- L'algorithme distingue *tous* les états distinguables.

- Pour le premier point, il suffit de montrer que l'algorithme calcule la limite de la suite $(D_i)_{i \in \mathbb{N}}$ définie comme suit :
  - $D_0 = F \times (Q \setminus F) \cup (Q \setminus F) \times F$;
  - $X_{n+1} = \{(p, q), (q, p) \mid \exists a \in \Sigma : (\delta(p, a), \delta(q, a)) \in D_n\}$;
  - $D_{n+1} = D_n \cup X_{n+1}$.
- Pour le second point, nous faisons une preuve par l'absurde. La démonstration utilise δ^*, la fonction de transition δ étendue aux mots.

Distinction entre états : correction de l'algorithme – preuve

Démonstration.

Supposons que le théorème soit faux (cad, il y a un automate contre-exemple). Alors, il existe au moins une "mauvaise paire" d'états $\{p, q\}$ t.q. :

- p et q sont distinguables : il existe $w \in \Sigma^*$ tel que soit $\delta^*(p, w) \in F$ soit $\delta^*(q, w) \in F$ (ou exclusif),
- l'algorithme ne distingue pas ces états.

Soit $w = a_1 \cdot a_2 \cdots a_n$ le plus court mot distinguant une mauvaise paire $\{p, q\}$

- $w \neq \epsilon$ d'après l'initialisation de l'algorithme (ligne 5)
- soient $p' = \delta(p, a_1)$ et $q' = \delta(q, a_1)$
  - p' et q' sont distingués par $a_2 \cdots a_n$ car $\delta^*(p', a_2 \cdots a_n) = \delta^*(p, w)$ et $\delta^*(q', a_2 \cdots a_n) = \delta^*(q, w)$
  - $a_2 \cdots a_n$ est plus court que n'importe quel mot distinguant une mauvaise paire
  - $\{p', q'\}$ ne peut pas être une mauvaise paire
- l'algorithme déclarera donc $\{p', q'\}$ comme distinguables
- d'après le corps de la boucle de l'algorithme, dans le pire des cas, à l'itération suivante, $\{p, q\}$ sera marquée.

□

Plan Chap. 1b - Automates à États Finis Déterministes

Tester l'équivalence entre deux automates

Considérons deux AEFDs *complets* :

- $A = (Q^A, \Sigma, q^A_{\text{init}}, \delta^A, F^A)$,
- $B = (Q^B, \Sigma, q^B_{\text{init}}, \delta^B, F^B)$.

Question

Comment savoir si A et B acceptent le même langage ?

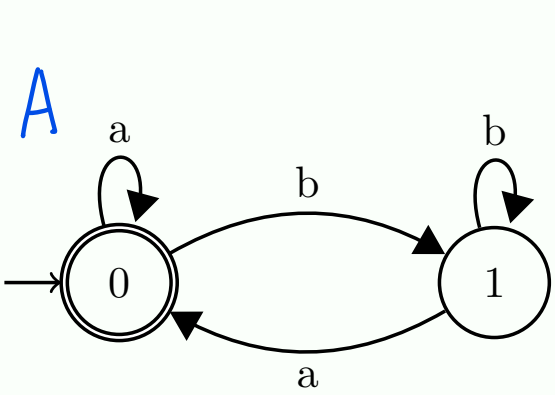
Procédure pour tester l'équivalence entre deux automates

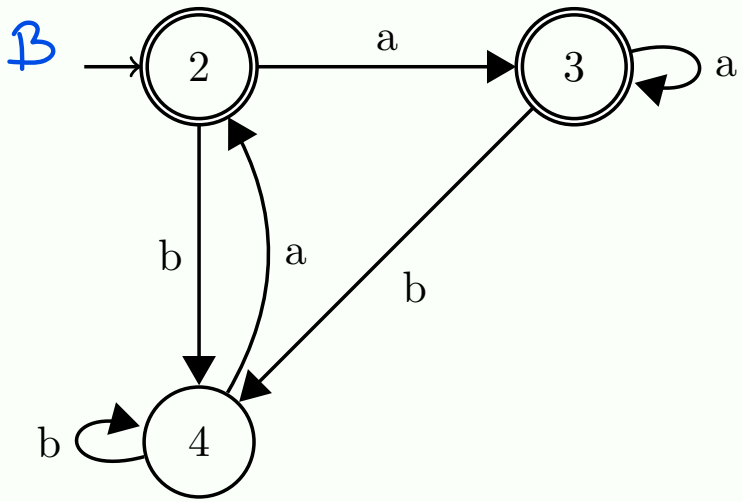
1. Construire l'automate $E = (Q^A \cup Q^B, \Sigma, q^A_{\text{init}}, \delta^A \cup \delta^B, F^A \cup F^B)$.
2. Tester si q^A_{init} et q^B_{init} sont distinguables dans E.

Tester l'équivalence entre deux automates

Exemple

Exemple (Deux automates équivalents)





	0	1	2	3
1	x			
2		x		
3		x		
4	x		x	x

Les états initiaux ne sont pas distinguable

$\Rightarrow A \equiv B$

«équivalent»

Plan Chap. 1b - Automates à États Finis Déterministes

Équivalence entre états

Soit $A = (Q, \Sigma, \delta, q_{\text{init}}, F)$ un AEFD dont tous les états sont accessibles

Définition (Relation d'équivalence entre états)

La relation d'équivalence $\approx$ entre états sur Q est définie par :

$$\forall p, q \in Q : \qquad p \approx q \quad \text{ssi} \quad \forall u \in \Sigma^* : \left(\delta^*(p, u) \in F \iff \delta^*(q, u) \in F\right)$$

$\approx$ est effectivement une relation d'équivalence. C'est une relation :

- réflexive,
- symétrique,
- transitive.

Notations :

- $[q]$: la classe d'équivalence de l'état q
- $Q_{/\approx}$: l'ensemble des classes d'équivalence (dans un automate avec ensemble d'états Q).

Équivalence et distinguabilité sont duales

Deux états sont équivalents si et seulement s'ils ne sont pas distinguables.

Minimisation : automate minimisé et équivalence

Soit $A = (Q, \Sigma, \delta, q_{\text{init}}, F)$ un AEFD complet dont tous les états sont accessibles.

Définition (Minimisé d'un automate)

Le minimisé de A est l'automate $A_{/\approx} = (Q_{/\approx}, \Sigma, [q_{\text{init}}], \delta_{/\approx}, F_{/\approx})$ où :

- $\delta_{/\approx}$ est l'application de transition t.q. :

$$\begin{array}{rcl} \delta_{/\approx} & : & Q_{/\approx} \times \Sigma \to Q_{/\approx} \\ \delta_{/\approx}([q], a) & \stackrel{\text{def}}{=} & [\delta(q, a)] \end{array}$$

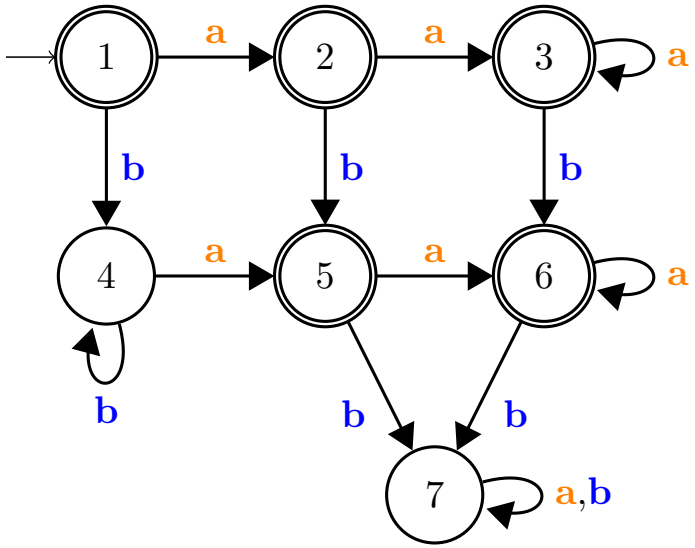
- $F_{/\approx} = \{[q] \mid q \in F\}$.

Théorème

Étant donnés A et son minimisé $A_{/\approx}$:

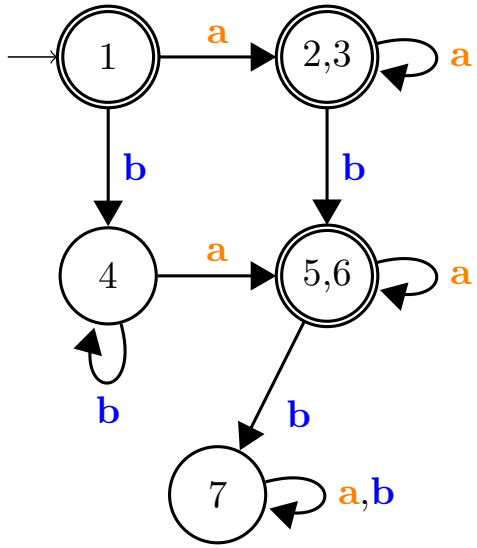
1. $L(A_{/\approx}) = L(A)$;
2. $A_{/\approx}$ est minimal pour $L(A)$: il n'existe pas d'AEFD complet qui reconnaît $L(A)$ et contient moins d'états que $A_{/\approx}$

Minimisation d'automate : exemple



À écrire sur copie (pas au brouillon)

$\equiv_0$	$\equiv_1$	$\equiv_2$	$\equiv_3$
1	2	2	2
2	3	3	3
3	1	1	1
5	5	5	5
6	6	6	6
4	4	4	4
7	7	7	7



Pourquoi l'algorithme de minimisation est optimal

Soit $A = (Q, \Sigma, \delta, q_{\text{init}}, F)$ un AEFD complet dont tous les états sont accessibles.

Soit M l'automate minimisé par l'algorithme de minimisation.

Supposons qu'il existe un automate minimisé N qui accepte le même langage que A mais avec moins d'états que M.

- Appliquer la procédure pour tester l'équivalence entre automates sur M et N.
- Les états initiaux de M et N sont indistinguables car $L(M) = L(N)$.
- Remarquer que si p et q sont indistinguables alors tous les successeurs sur n'importe quel symbole sont indistinguables (sinon p et q seraient distinguables).
- Tous les états de M sont indistinguables d'au moins un état de N.
  - prenons p de M, il existe $w \in \Sigma^*$ depuis l'état initial de M vers p
  - par w nous pouvons atteindre un état de N depuis sont état initial
  - par induction, p et l'état atteint dans N par w sont indistinguables
- Comme N a moins d'états que M, il y a deux états de M qui sont indistinguables du même état dans N.
- Par transitivité de la relation d'indistinguabilité, ces deux états sont indistinguables l'un de l'autre.
- Contradiction : M a été conçu tel que tous ses états sont distinguables.

Plan Chap. 1b - Automates à États Finis Déterministes

Résumé du Chapitre 5 : *Équivalence et Minimisation* d'Automate à États Fini Déterministes

Équivalence et Minimisation d'Automate à États Fini Déterministes

- équivalence entre états,
- équivalence entre automates,
- minimisation d'automate.

Pour le prochain cours

- Déterminer pourquoi les algos de calculs des relations d'états distinguables et équivalents terminent.
- Définir des procédures permettant de calculer des automate reconnaissant l'union et le xor des langages d'automates passés en paramètres.