Minimisation

Problème

- **Donnée** : A un AFD complet dont chaque état est accessible depuis l'état initial

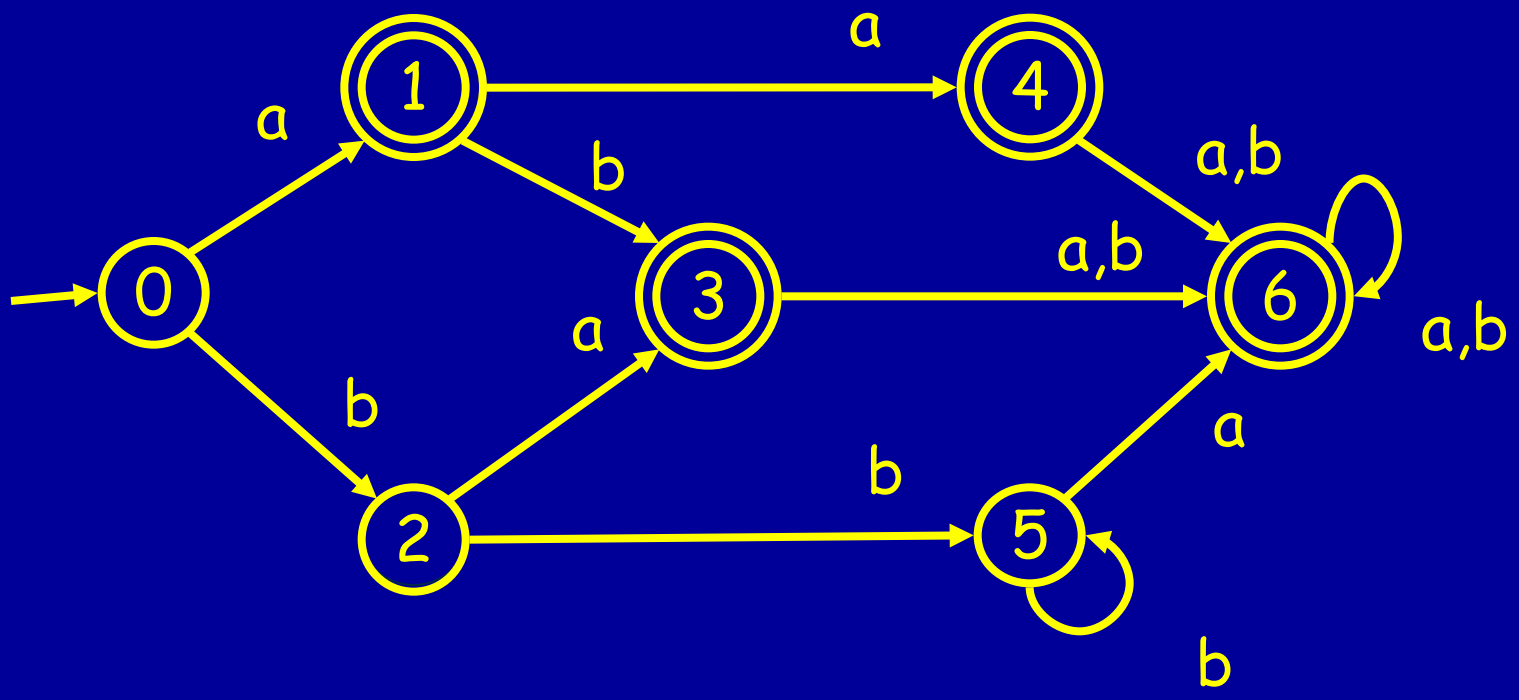
- **Problème** : construire un AFD minimal qui reconnaisse le même langage que A.

- **Idée** : fusionner les états équivalents.

En pratique, l'algorithme est fondé sur un principe de séparation des états ...

Exemple

$$\forall w \in \Sigma^* \begin{cases} \delta(p,w) \in F \text{ et } \delta(q,w) \in F \\ \text{ou} \\ \delta(p,w) \notin F \text{ et } \delta(q,w) \notin F \end{cases}$$

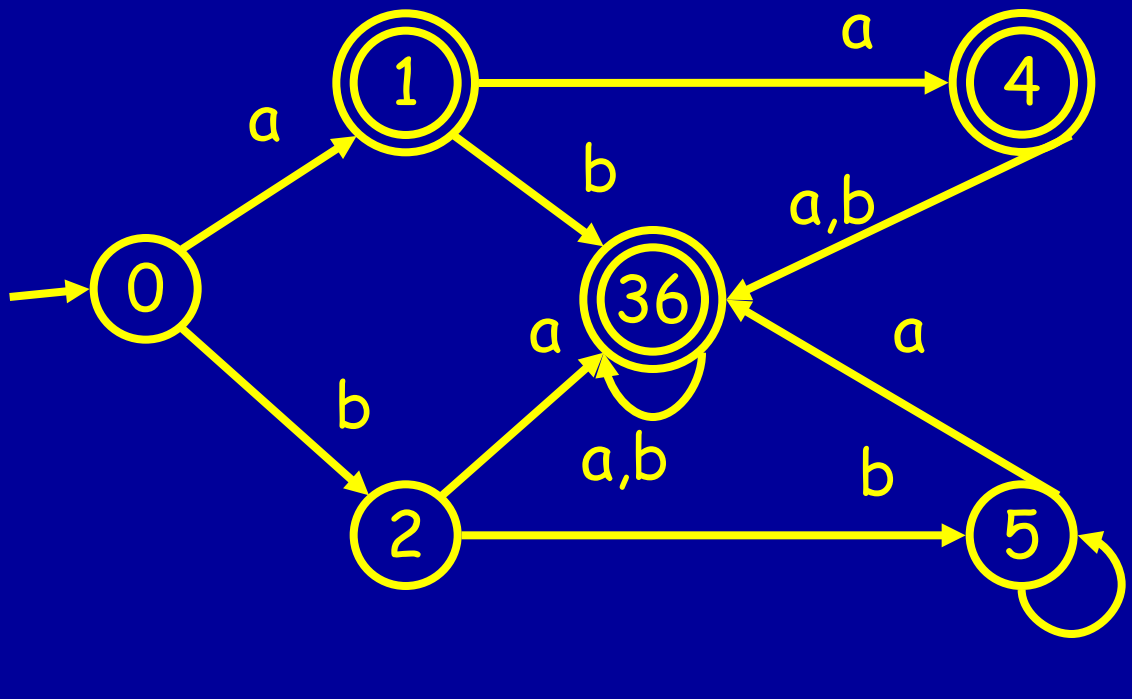


$0 \approx 1$? Non car $0 \notin F$ et $1 \in F$

$3 \approx 6$? Oui car $3,6 \in F$, $\delta(3,a)=\delta(6,a)$ et $\delta(3,b)=\delta(6,b)$

Exemple

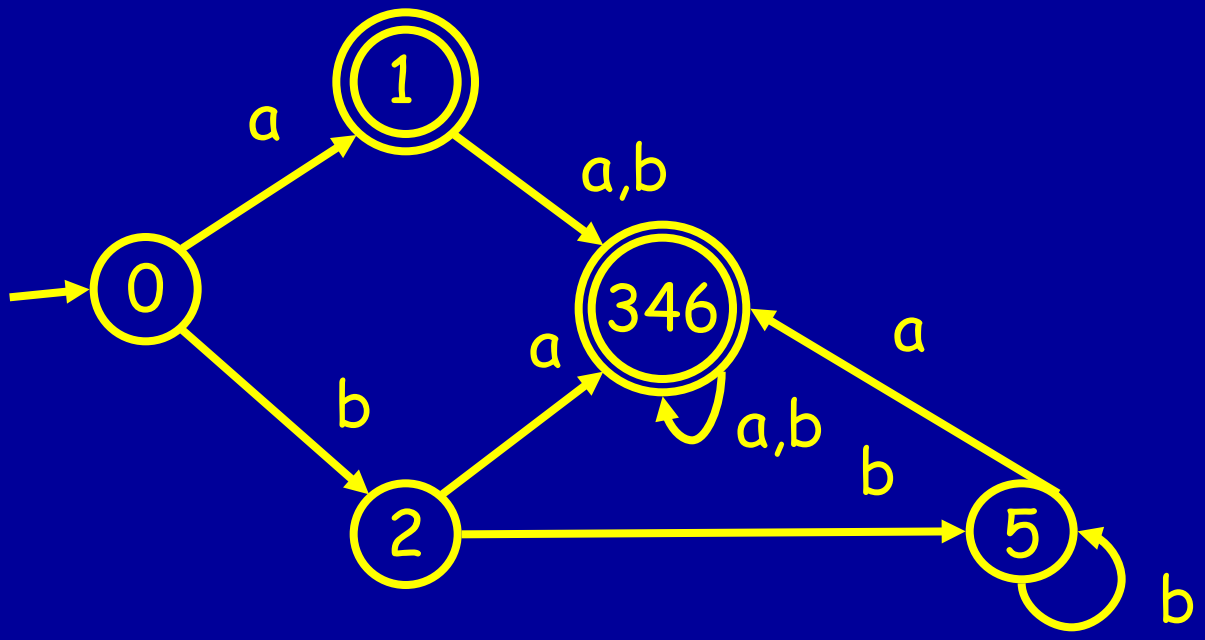
$$\forall w \in \Sigma^* \begin{cases} \delta(p,w) \in F \text{ et } \delta(q,w) \in F \\ \text{ou} \\ \delta(p,w) \notin F \text{ et } \delta(q,w) \notin F \end{cases}$$



4≈6? Oui car 4,6∈F, $\delta(4,a)=\delta(6,a)$ et $\delta(4,b)=\delta(6,b)$

Exemple

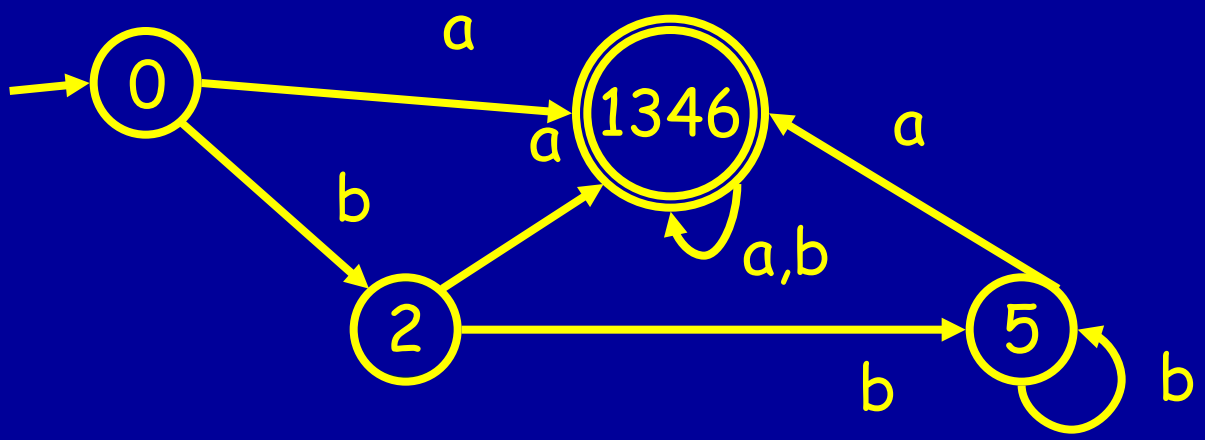
$$\forall w \in \Sigma^* \begin{cases} \delta(p,w) \in F \text{ et } \delta(q,w) \in F \\ \text{ou} \\ \delta(p,w) \notin F \text{ et } \delta(q,w) \notin F \end{cases}$$



1≈6? Oui car 1,6∈F, $\delta(1,a) = \delta(6,a)$ et $\delta(1,b) = \delta(6,b)$

Exemple

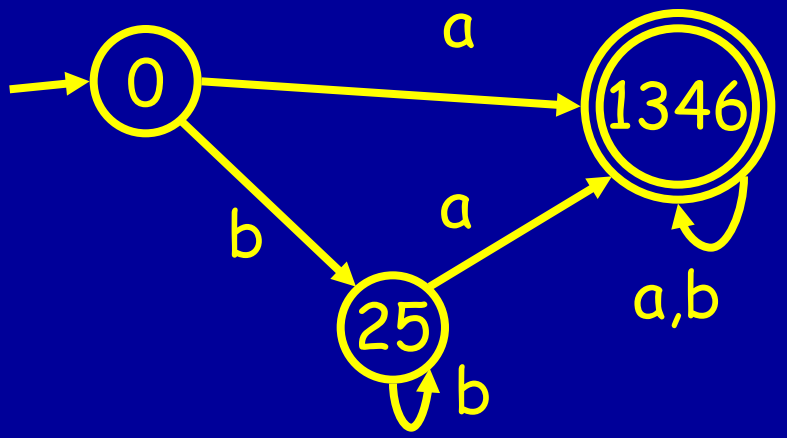
$$\forall w \in \Sigma^* \begin{cases} \delta(p,w) \in F \text{ et } \delta(q,w) \in F \\ \text{ou} \\ \delta(p,w) \notin F \text{ et } \delta(q,w) \notin F \end{cases}$$



2≈5? Oui car 2,5∉F, $\delta(2,a)=\delta(5,a)$ et $\delta(2,b)=\delta(5,b)$

Exemple

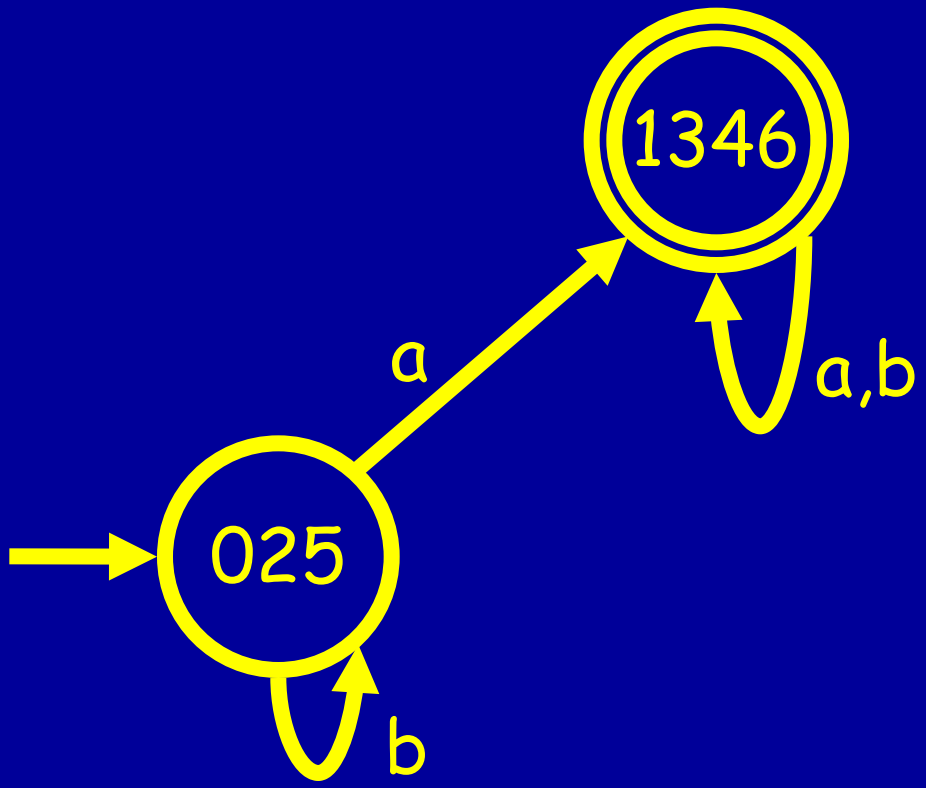
$$\forall w \in \Sigma^* \begin{cases} \delta(p,w) \in F \text{ et } \delta(q,w) \in F \\ \text{ou} \\ \delta(p,w) \notin F \text{ et } \delta(q,w) \notin F \end{cases}$$



$0 \approx 2$? Oui car $0,2 \notin F$, $\delta(0,a)=\delta(2,a)$ et $\delta(0,b)=\delta(2,b)$

Exemple

$$\forall w \in \Sigma^* \begin{cases} \delta(p,w) \in F \text{ et } \delta(q,w) \in F \\ \text{ou} \\ \delta(p,w) \notin F \text{ et } \delta(q,w) \notin F \end{cases}$$



$0 \approx 1$? Non car $0 \notin F$ et $1 \in F$

Classe d'équivalence et automate associé

La relation ≈ est une relation d'équivalence (elle est réflexive, symétrique, transitive).

Si q est un état, on note $[q]$ l'ensemble des états qui lui sont équivalents et on définit l'automate des classes d'équivalence :

Classe d'équivalence et automate associé

Étant donné un AFD $A = (\Sigma, Q, \delta, q_0, F)$, l'automate minimal associé à A est :

$$\mu A = (\Sigma, Q', \delta', [q_0], F')$$

- $Q' = \{[q], q \in Q\}$
- $F' = \{[f], f \in F\}$
- $\delta' = \{ ([p],\sigma,[q])$ tels que $\exists p' \in [p], \exists q' \in [q]$ $(p',\sigma,q') \in \delta \}$

Justification

- 3 étapes:
  - L'automate μA des classes d'équivalence de A est bien défini, réduit et $L(\mu A) = L(A)$.
  - Pour tout AFD B tel que $L(B) = L(A)$,

    $$\#\text{états}(B) \geq \#\text{états}(\mu A)$$

  - Tout automate minimal B tel que $L(B) = L(A)$, est **isomorphe*** à A

---

* **Isomorphe =** Il existe une bijection ϕ entre les états de A et ceux de B qui préserve

- les états spéciaux (initial et d'acceptation)
- les transitions : $\forall p,q \in Q_A,\ \delta_A(p,a)=q \Leftrightarrow \delta_B(\phi(p),a)=\phi(q)$

Sur les quotients gauches

- L'automate Q(L) des quotients gauches défini comme $\{L_q(A):q \in Q\} = Q(L)$ est-il bien minimal?
- Supposons, par l'absurde qu'il ne le soit pas. Alors il existe au moins p et q, deux états de l'automate tels que $L_p(A)=L_q(A)$, par définition de Q(L).
  - Si tel est le cas, par définition de l'équivalence, $p \approx q$.
  - Il s'ensuit que p et q peuvent être fusionnés, contredisant la minimalité de l'automate des quotients gauches.

Complexité du regroupement d'états

- Pour chaque paire d'états, il faut considérer l'ensemble des mots de longueur n sur Σ.

  $O(n^2)$ paires d'états

  $|\Sigma|^n$ mots de longueur n

  (n = nombre d'états de l'AFD)

- Algorithme en $O(n^2 |\Sigma|^n)$... catastrophique

- Trouver un meilleur algorithme !

Principe

- Au lieu de fusionner les états équivalents,
  - on groupe tous les états;
  - on sépare **inductivement** les états non équivalents;
  - quand on ne peut plus séparer on a terminé.

- La séparation inductive se fait en construisant inductivement ≈

Construction inductive de ≈

Base :

$$p \approx_0 q \Leftrightarrow (p \in F \wedge q \in F) \vee (p \notin F \wedge q \notin F)$$

- Règle :

$$p \approx_i q \Leftrightarrow (p \approx_{i-1} q) \wedge (\forall a \in \Sigma,\ \delta(p,a) \approx_{i-1} \delta(q,a))$$

La base permet de partitionner Q

La règle affine la partition de Q

Remarque : $p \approx_i q$ si on ne peut pas séparer p de q par un mot de longueur au plus i.

Cas d'arrêt

- dès que 2 équivalences successives coïncident

$$\approx_i = \approx_{i+1} \Rightarrow \forall k,\ \approx_i = \approx_{i+k}$$

- Par hypothèse, $\approx_i = \approx_{i+1}$. Alors

$$p \approx_i q \text{ et } \forall a \in \Sigma,\ \delta(p,a) \approx_i \delta(q,a) \Leftrightarrow p \approx_{i+1} q$$

$$p \approx_{i+1} q \text{ et } \forall a \in \Sigma,\ \delta(p,a) \approx_{i+1} \delta(q,a) \Leftrightarrow p \approx_{i+2} q$$

- Conséquence : dès qu'il y a coïncidence de 2 équivalences successives, on a obtenu l'automate minimal

Cas d'un AFD déjà minimal

- Aucune paire d'états n'est équivalente.

$$\approx = \approx_{n-2} \text{ pour } n = |Q|$$

- $\approx_0$ partitionne Q en deux classes;

puisque $\forall i \quad \approx_i \neq \approx_{i+1}$

- $\approx_{i+1}$ partitionne Q avec au moins une classe de plus que $\approx_i$
- on ne peut avoir plus de n classes (n =|Q|), donc $\approx = \approx_{n-2}$

Minimisation de $A=<Q,\Sigma,\delta,i,T>$

La séparation des états est définie par :

- Pour chaque classe G de Π faire
  - p et q sont dans des classes d'équivalence différentes SSI
    $\exists a \in \Sigma$: $\delta(p,a)$ et $\delta(q,a)$ sont dans des classes différentes
  - Remplacer G par les sous-groupes ainsi formés.

Terminer

- Choisir un état [p] représentant chaque classe de Π

- Pour chaque transition $\delta(p,a)=q$ de A, ajouter une transition de [p] vers [q] étiquetée par a.

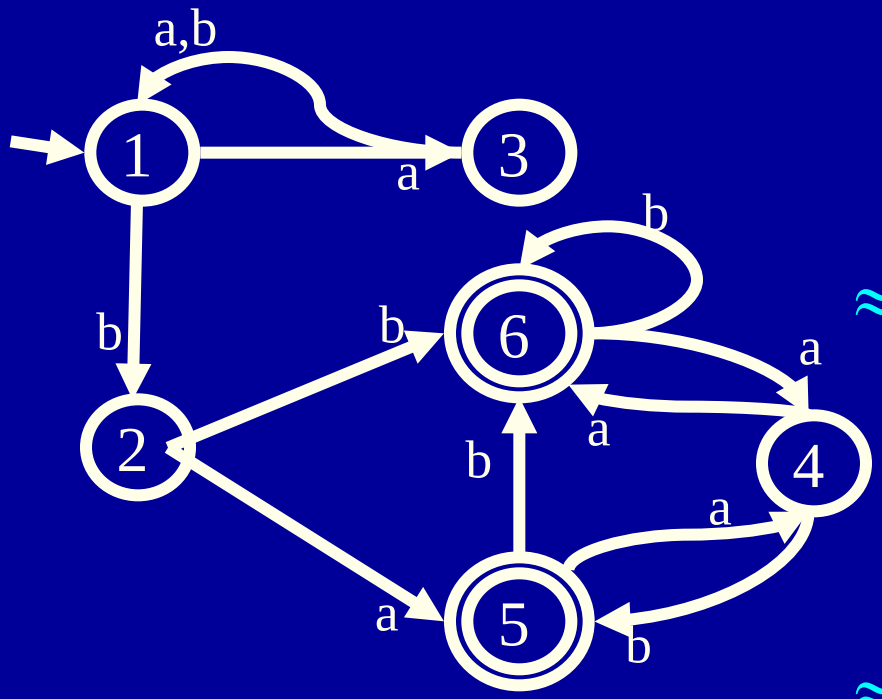
- État initial : l'état représentant la classe de i
- États terminaux : les état représentant les classes contenant des terminaux de A.

Complexité

- La définition inductive fournit un algorithme en $O(n^2|\Sigma|)$ pour $n=|Q|$, qui détermine les classes d'équivalence et construit donc l'AFD minimal.

- Avec quelques améliorations, on peut construire l'AFD minimal en $O(n \log n\ |\Sigma|)$

Exemple



δ	1	2	3	4	5	6
a	3	5	1	6	4	4
b	2	6	1	5	6	6

$\approx_0$

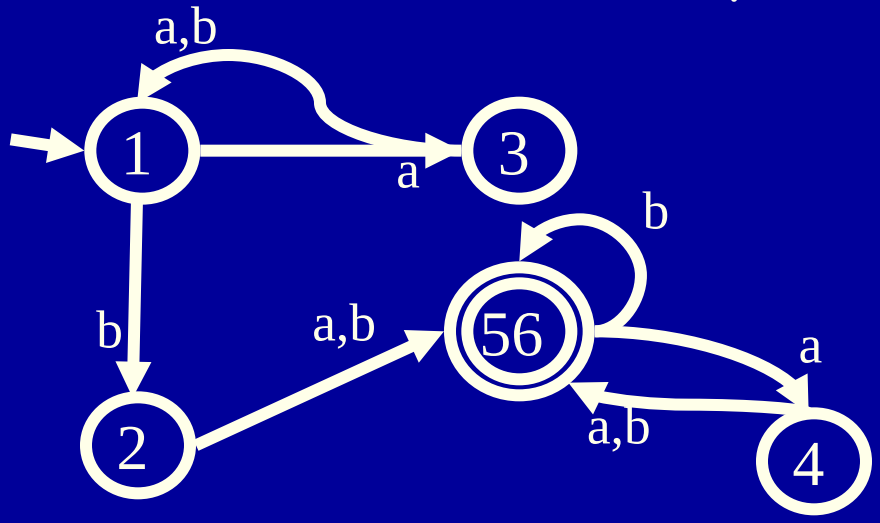
$\approx_1$

$\approx_2$

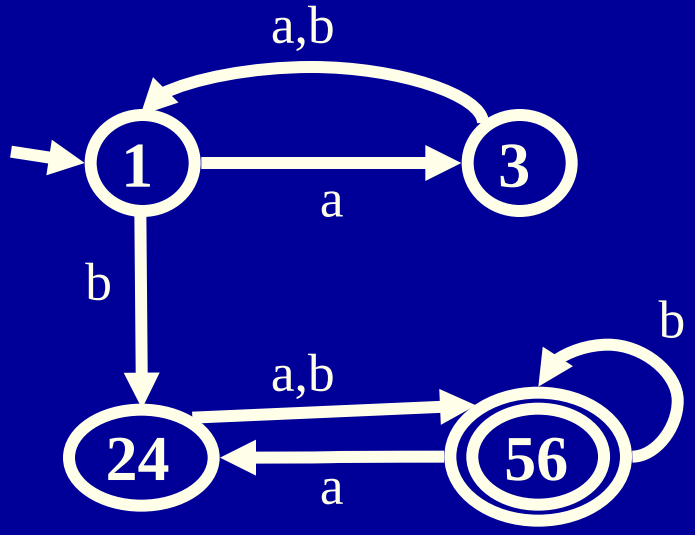
$\approx_3 = \approx_2$ **Plus rien ne peut se séparer**

Exemple

5 et 6 sont équivalents



2 et 4 sont équivalents



i	k	[j]		[f]		
1	3	2	4	5	6	
k	i	f	f	j	j	a
j	i	f	f	f	f	b

Propriétés de clôture

But

- Savoir quelles sont les opérations qui conservent la rationalité d'un langage.

- On connaît déjà plusieurs manières de considérer les langages rationnels
  - Par les expressions rationnelles
  - Par les automates

Clôture par complémentation

- La classe des langages rationnels est close par complémentation : $L \in Rat(\Sigma) \Rightarrow \Sigma^* \backslash L \in Rat(\Sigma)$,
- Preuve par automates :
  - $L \in Rat(\Sigma) \Rightarrow$ il existe A un AFD **complet**, $A=\langle Q,\Sigma,\delta,i,F\rangle$ t.q. $L(A) = L$
  - on définit A' à partir de A pour reconnaître $\Sigma^* \backslash L$:
    - $A'=\langle Q,\Sigma,\delta,i,Q \backslash F\rangle$
  - Tous les états non terminaux deviennent terminaux et vice versa

Clôture par l'intersection

- Si L et M sont deux langages rationnels alors $L \cap M$ est également rationnel.

$$L \cap M = \overline{\overline{L} \cup \overline{M}}$$

- Preuve directe:

  Comme l'ensemble des langages rationnels est clos pour la complémentation et l'union, il est clos pour l'intersection

Preuve par automates

- Soit $A=\langle Q,\Sigma,\delta,i,T\rangle$ tel que $L(A)=L$
- Soit $B=\langle Q',\Sigma,\delta',j,T'\rangle$ tel que $L(B)=M$

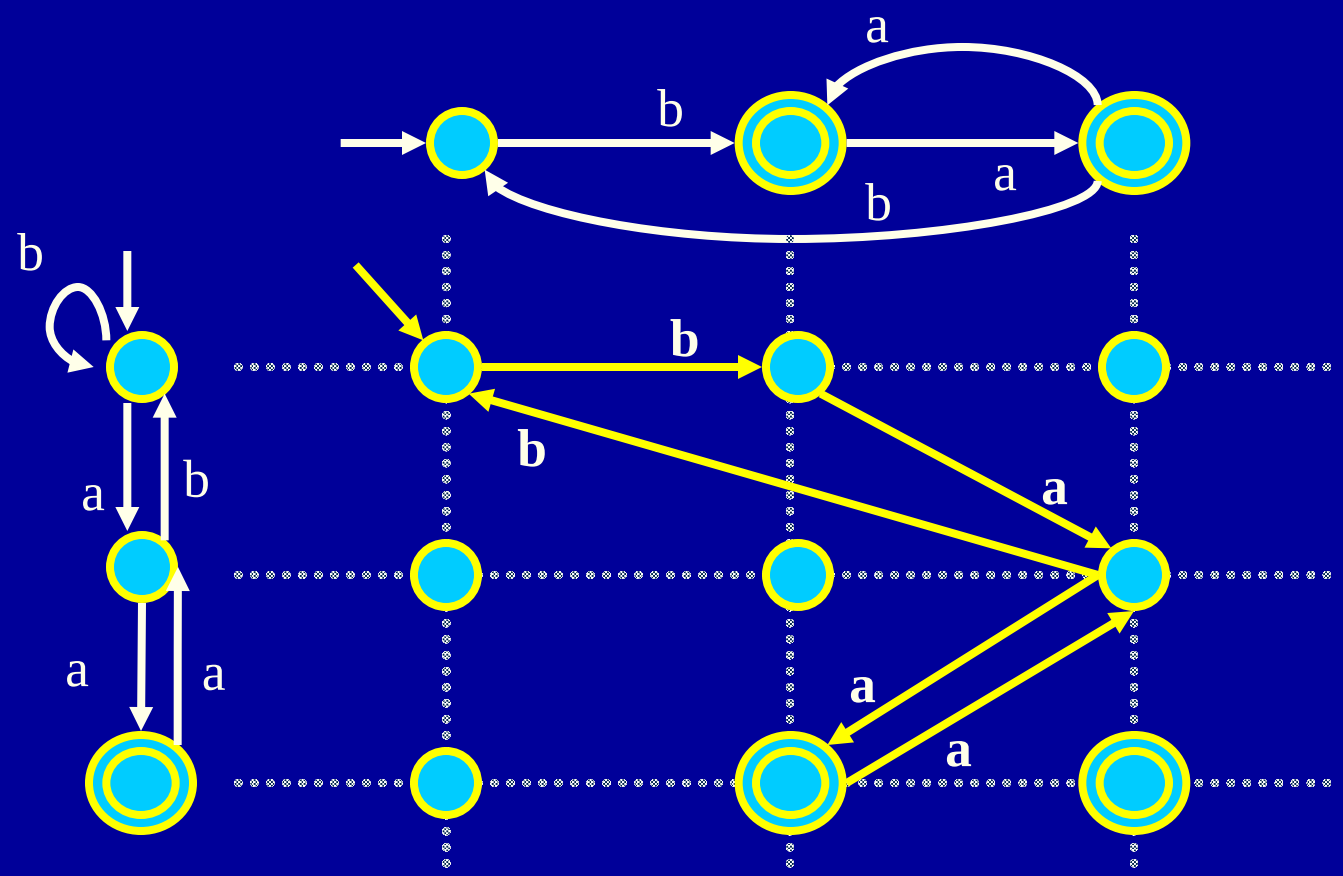
Alors, $C=\langle Q\times Q',\Sigma,\delta_C,[i,j],T\times T'\rangle$ pour

$$\delta_C([p,q]),a)=[\delta(p,a),\ \delta'(q,a)]$$

Pour tout $p\in Q$, $q\in Q'$ et $a\in\Sigma$

Reconnaît $L\cap M$

Exemple



Clôture par l'union

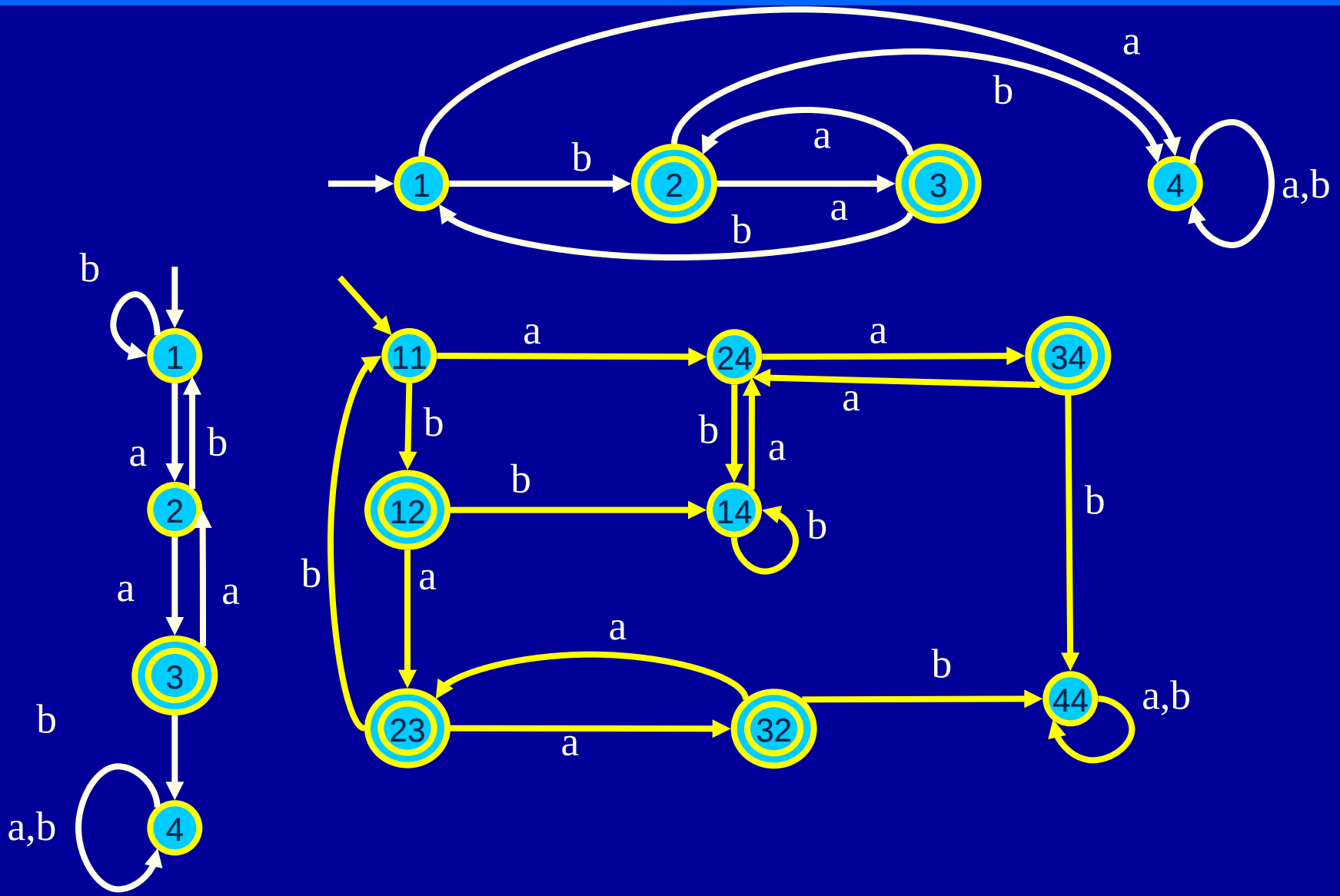
- La construction précédente permet également de prouver la clôture par l'union
- Si L et M sont deux langages rationnels alors L∪M est également rationnel.
- Preuve par automates :
  - Soit $A=\langle Q,\Sigma,\delta,i,F\rangle$ **complet** tel que $L(A)=L$
  - Soit $B=\langle Q',\Sigma,\delta',j,F'\rangle$ **complet** tel que $L(B)=M$

  Alors, $D=\langle Q\times Q',\Sigma,\delta_D,[i,j],\{[f,f'] \mid f\in F \text{ ou } f'\in F'\}\rangle$ pour

  $$\delta_D([p,q]),a)=[\delta(p,a), \delta'(q,a)]$$

  pour tout $p\in Q$, $q\in Q'$ et $a\in\Sigma$ reconnaît $L \cup M$

Exemple


a
b
b
a
1
2
3
4
a,b
b
b
1
a
b
2
a
a
3
b
a,b
4
11
a
24
a
34
a
b
b
a
12
b
14
b
b
b
a
a
b
23
a
32
44
a,b

Clôture par substitution

- À chaque lettre de l'alphabet d'une expression rationnelle on associe un langage rationnel: on substitue un langage à une lettre.

$$f(\varepsilon)= \varepsilon \text{ et } f(ma)=f(m)f(a)$$

m un mot et a une lettre

- Pour les langages:

$$f(L) = \cup_{m \in L} f(m)$$

Exemple

- f(0)=a et f(1)=b*
- f(010)=ab*a
- Pour L=0*(0+1)1*, f(L)=a*(a+b*)(b*)*=a*b*

Clôture par substitution

- Soit $L \in Rat(\Sigma)$ et $\forall a \in \Sigma$, $R_a \in Rat(\Delta)$.
  Soit la substitution $f: \Sigma \rightarrow \Delta^*$, $f(a)=R_a$
  f remplace toute occurrence de a dans L par R_a.

- $f(L \cup M)=f(L) \cup f(M)$, $f(L.M)=f(L).f(M)$, $f(M^*)=f(M)^*$

- On montre par récurrence sur la structure de L que l'expression rationnelle obtenue représente bien f(L).

Récapitulatif

Clôture image miroir

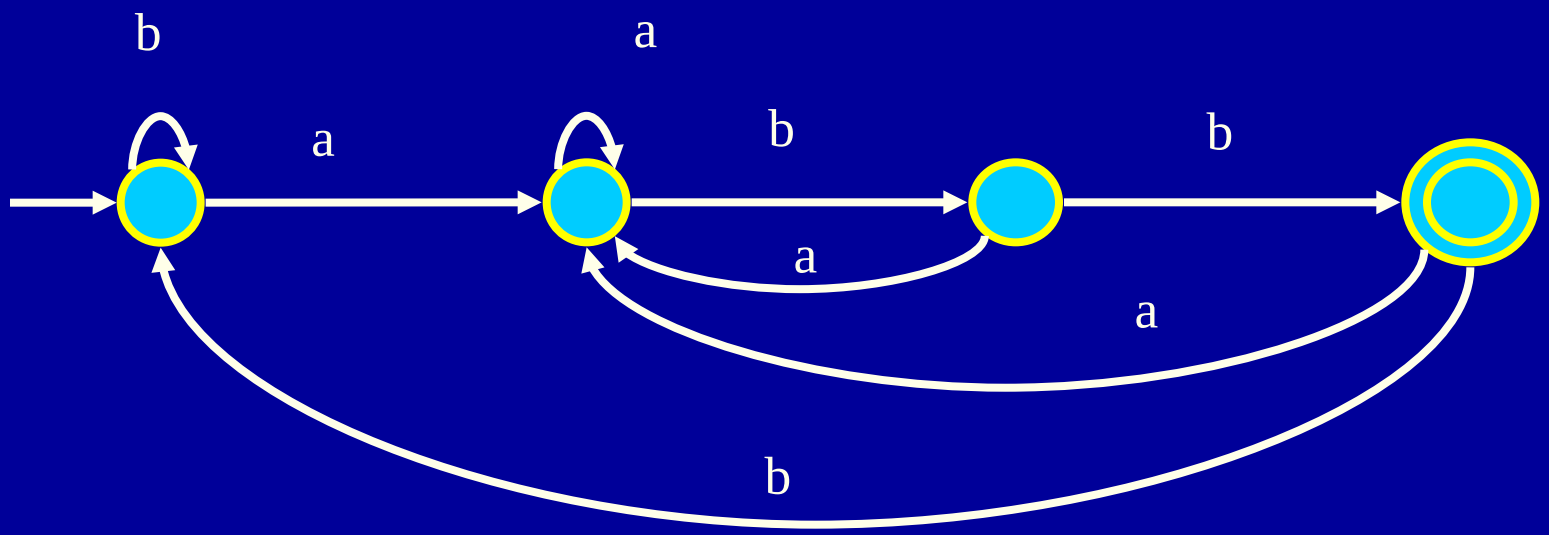
- Si L est un langage rationnel alors le langage r(L), composé des images miroirs des mots de L est également rationnel.
- La preuve est facile. En effet, si l'automate A reconnaît L, alors l'automate inverse r(A), reconnaît r(L).

Exemple

- Le langage des mots ayant abb comme suffixe

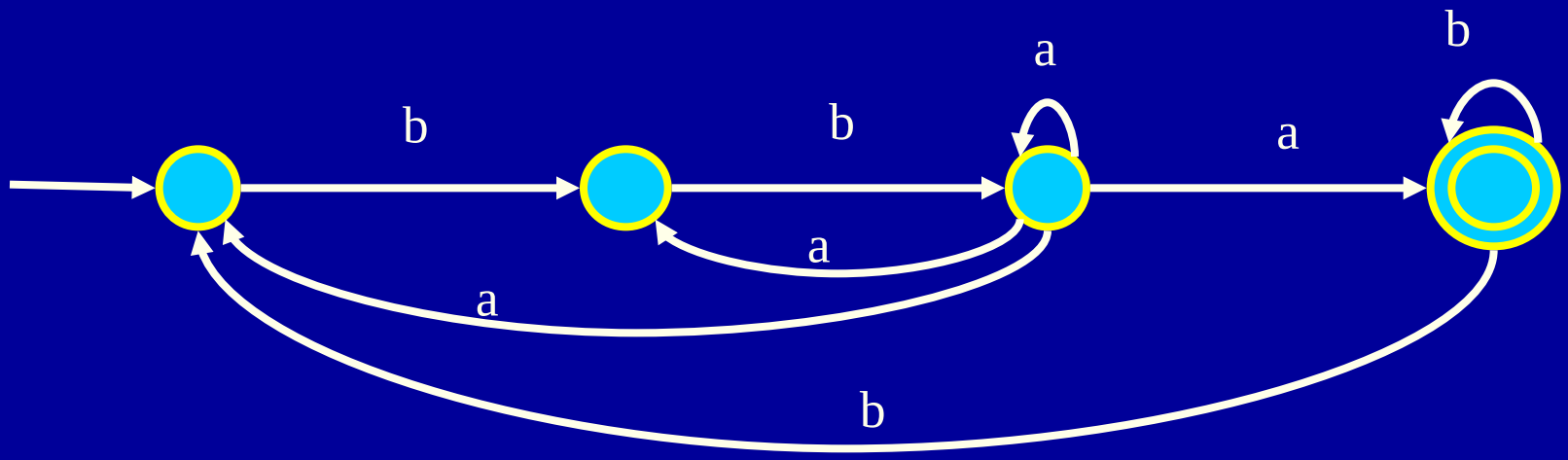
(a+b)*abb

est reconnu par l'automate A



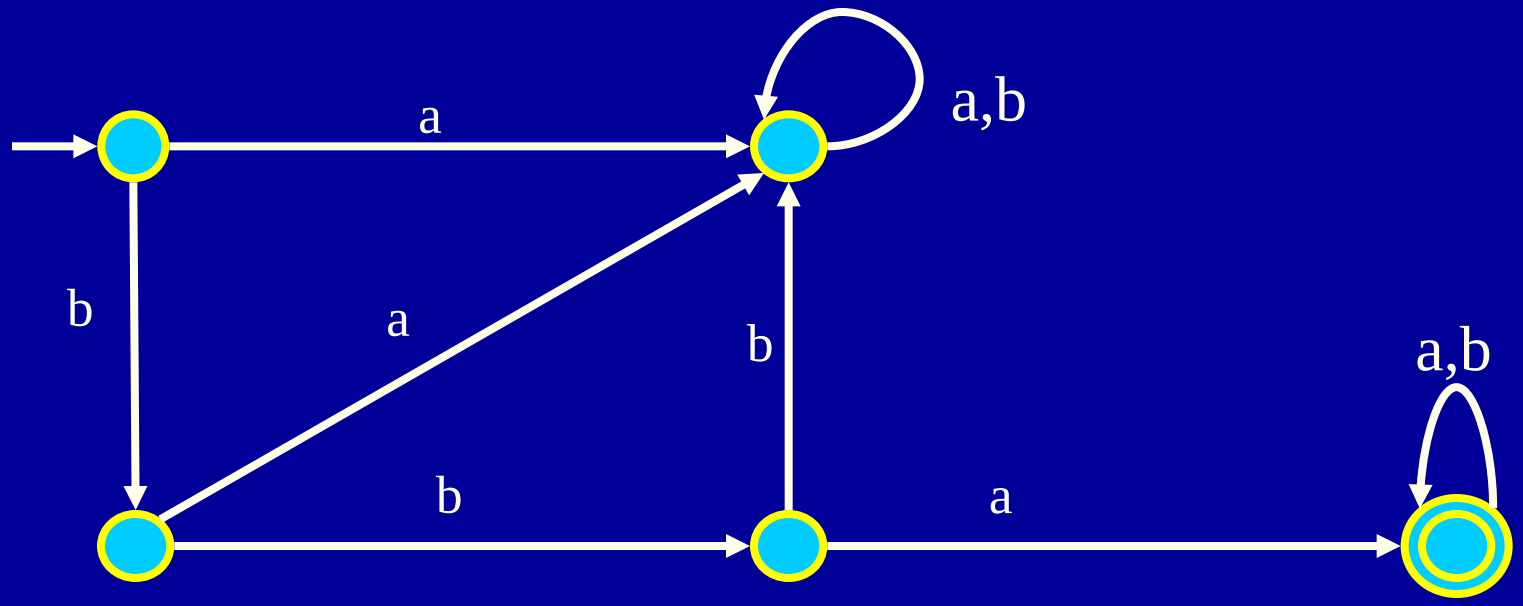
Exemple

Pour reconnaître le langage miroir, il suffit d'inverser l'orientation des arcs - r(*A*).



Exemple (suite)

- Mais, nous avons obtenu un automate *A'* non déterministe !
- Il faut donc déterminiser, pour obtenir d(*A'*)



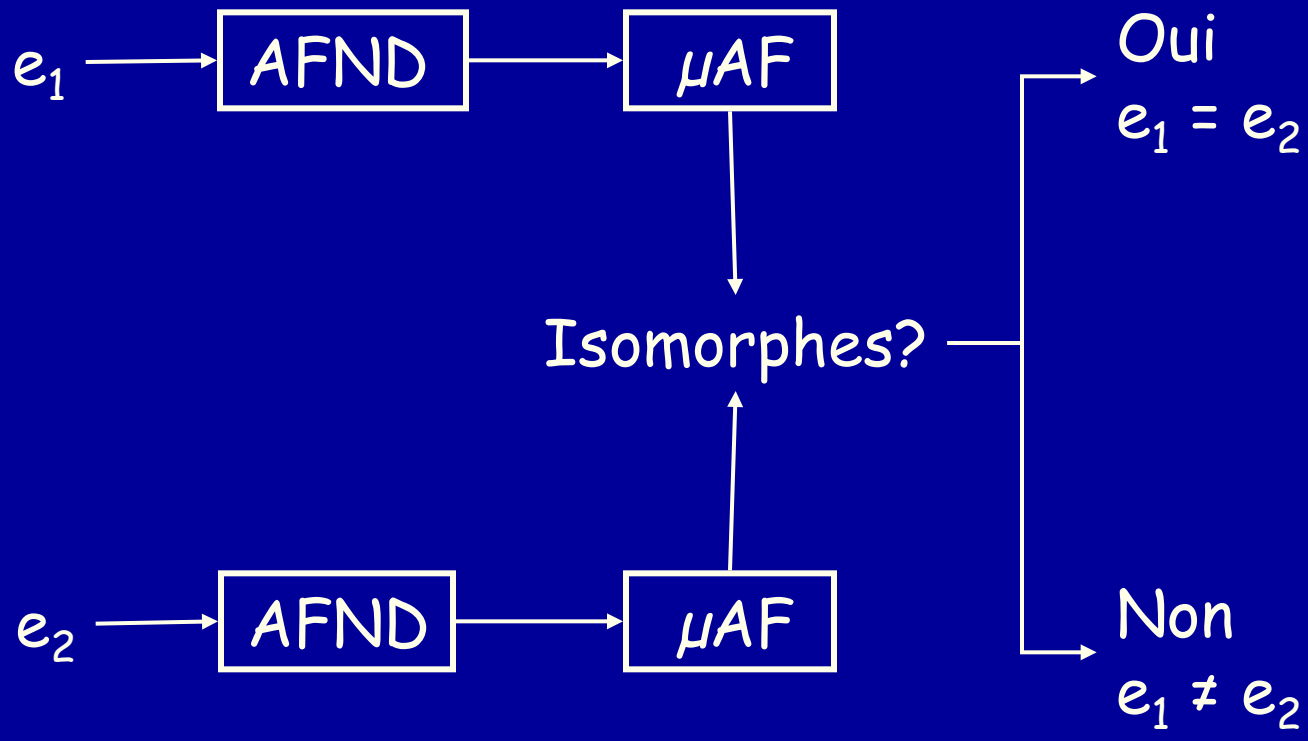
Problème de l'égalité d'expressions rationnelles

Égalité d'expressions rationnelles

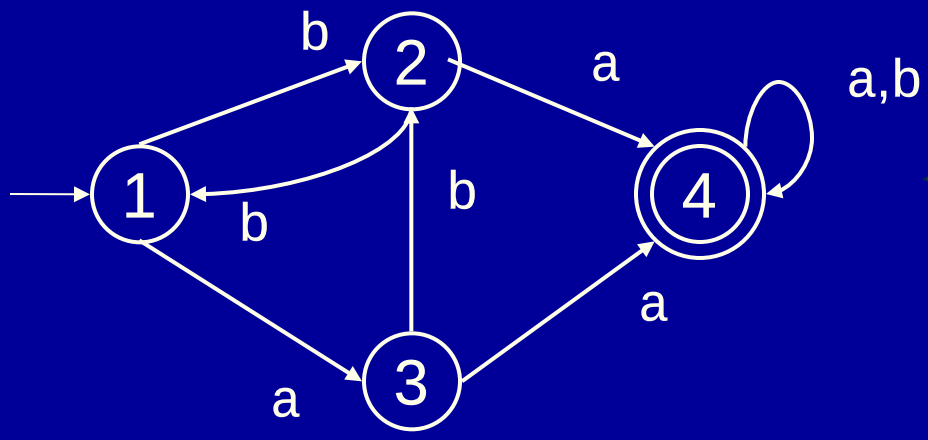
- Problème
  - Données : e_1 et e_2 deux expressions rationnelles
  - Question : $e_1 = e_2$?

- Exemple
  - (a*b*)*(b*a*)*bb((b*a*)*+(b+a)*)=(a+b)*bb(a+b)*

Fonctionnement



Exemple



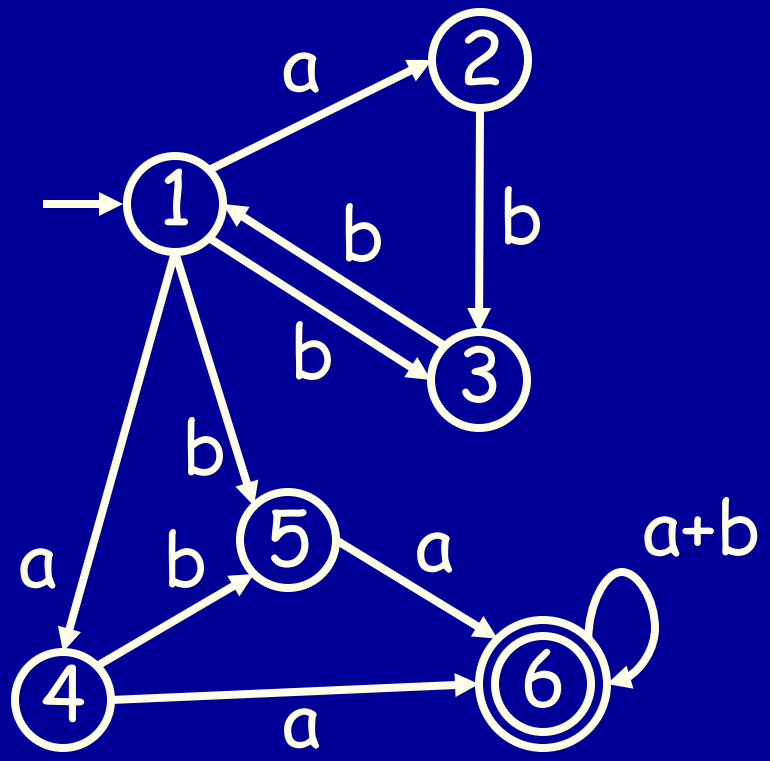
minimal

L=[(bb)*a((bb) +a)*(b(bb)*a+a)+(bb)*ba](a+b)*

L'= (abb+bb)*(aa+aba+ba) (a+b)*

Exemple

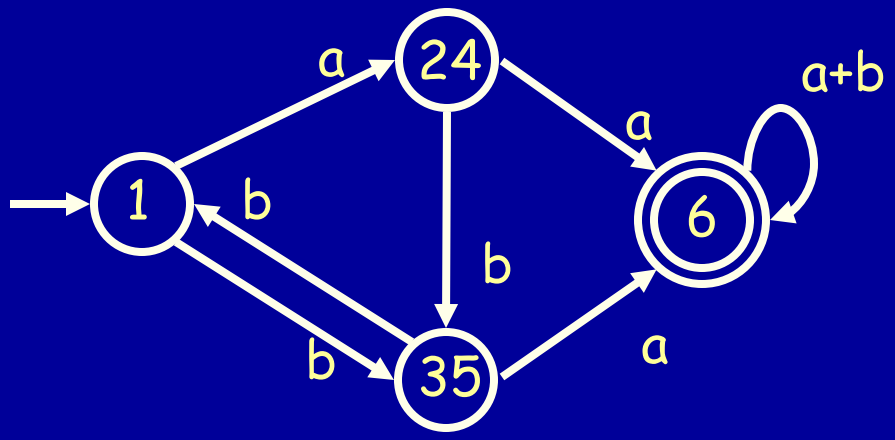
$L' = (abb+bb)^*(a+ab+b)a\ (a+b)^*$



	a	b
→1	24	35
2	-	3
3	-	1
4	6	5
5	6	-
←6	6	6

Exemple

L'= (abb+bb)*(a+ab+b)a (a+b)*



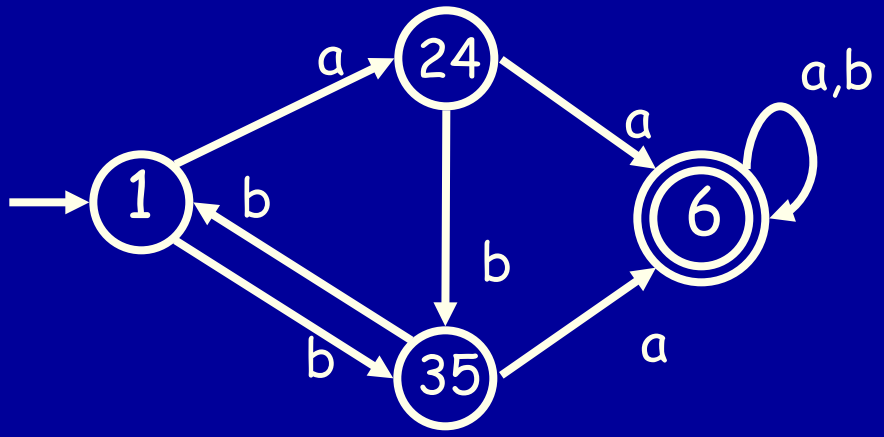
Minimal, à vue de nez...

	a	b
→1	24	35
2	-	3
3	-	1
4	6	5
5	6	-
←6	6	6

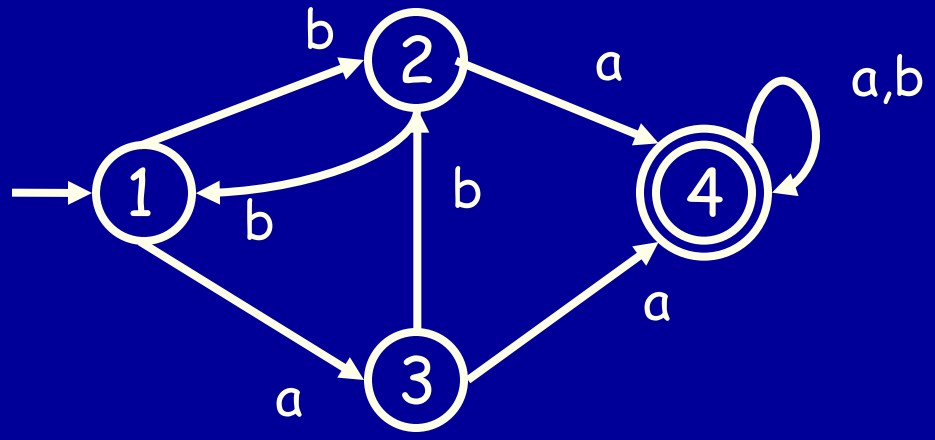
→1	24	35
24	6	35
35	6	1
←6	6	6

Exemple

L'= (abb+bb)*(a+ab+b)a (a+b)*



isomorphes



L=[(bb)*a((bb) +a)*(b(bb)*a+a)+(bb)*ba](a+b)*