# Deux manières

Expression rationnelle $\xrightarrow{\text{construction}}$ AFND

Quotients gauches

déterminisation

Pour $L \subseteq \Sigma^*$, on définit :

- $A(L) = \langle Q(L), \Sigma, \delta, \{L\}, F(L) \rangle$

  $Q(L) = \{u^{-1}L \mid u \in \Sigma^*\}$

  $F(L) = \{u^{-1}L \mid \varepsilon \in u^{-1}L\}$

  $\delta(Y,a) = a^{-1}Y$ pour $Y \in Q(L)$, $a \in \Sigma$

AFD minimal $\longleftarrow$ AFD



# Deux manières

Expression rationnelle $\xrightarrow{\text{construction}}$ AFND

Quotients gauches

déterminisation

Pour $A = \langle \Sigma, Q, i, F, \delta \rangle$ un AFD.

$\forall q \in Q$, $[q]$ est la classe d'équivalence de $q$ sur $Q/\approx$

On définit $\mu A = \langle \Sigma, Q', [i], F', \delta' \rangle$

$Q' = \{[q] \mid q \in Q\}$

$F' = \{[f] \mid f \in F\}$

$\delta' = \{([p],a,[q]) \mid (p',a,q') \in \delta$ pour $p' \in [p], q' \in [q]\}$

$$p \approx q \text{ si } L_p(A) = L_q(A) \Leftrightarrow \forall w \in \Sigma^* \begin{cases} \delta(p,w) \in F \text{ et } \delta(q,w) \in F \\ \text{ou} \\ \delta(p,w) \notin F \text{ et } \delta(q,w) \notin F \end{cases}$$

AFD minimal $\xleftarrow{\text{équivalence d'états}}$ AFD

Comment le construire ?



# Regroupement d'états

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

  $O(n^2)$ 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 $\approx$



# Construction inductive de $\approx$

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

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

  $p \approx_{i+1} q$ 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.

  $\approx \; = \; \approx_{n-2}$ 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

  $\approx \; = \; \approx_{n-2}$ .

7

## Minimisation de $A = \langle Q, \Sigma, \delta, i, T \rangle$

- Construire partition ∏ des états en deux groupes
  - un des états terminaux
  - un des autres (non terminaux)

```
Répéter
   construire un nouvelle partition ∏'
    en séparant les états
   Si ∏≠∏' alors ∏←∏' fsi
Jusqu'à ∏=∏'
```

- Terminer

8

## Minimisation de $A = \langle Q, \Sigma, \delta, i, T \rangle$

La séparation des états est définie par :

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

9

## Terminer

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

- Pour chaque transition δ(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*.

10

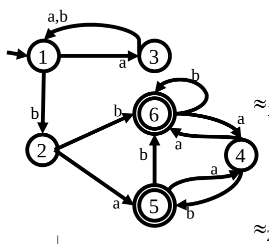
## Complexité

- 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 $O(n \log n \, |\Sigma|)$

11

## Exemple




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

$\approx_0$

| [i] | | | | [f] | | |
|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | |
| 3 | 5 | 1 | 6 | 4 | 4 | a |
| 2 | 6 | 1 | 5 | 6 | 6 | b |

| [i] | | | | [f] | | |
|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 | |
| i | f | i | f | i | i | a |
| i | f | i | f | f | f | b |

$\approx_1$

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

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

$\approx_2$

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

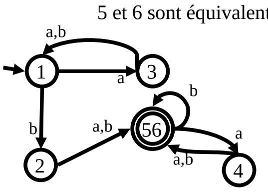
| i | k | [j] | | [f] | | |
|---|---|---|---|---|---|---|
| 1 | 3 | 2 | 4 | 5 | 6 | |
| k | i | f | f | j | j | a |
| j | i | f | f | f | f | b |

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

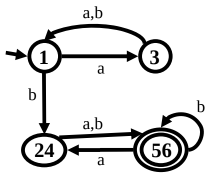
12

## 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

- *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**, $A=<Q,\Sigma,\delta,i,F>$ t.q. $L(A) = L$
  - on définit A' à partir de A pour reconnaître $\Sigma^* \backslash L$ :
    - $A'=<Q,\Sigma,\delta,i,Q\backslash F>$
  - 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.

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

- <u>Preuve directe:</u>

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



## Preuve par automates

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

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

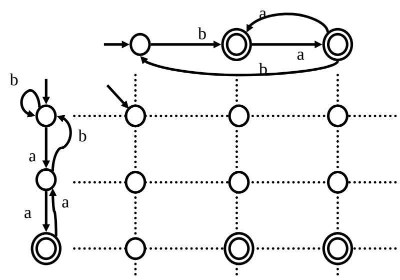
$\delta_C([p,q]),a)=[\delta(p,a), \delta'(q,a)]$

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

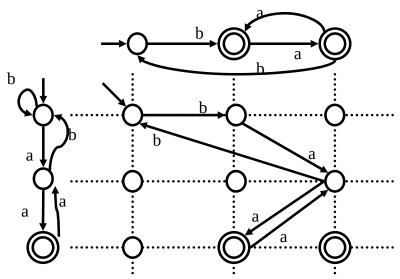
Reconnaît $L \cap M$



## Exemple



# 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 $L \cup M$ 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 $L \cup M$



# 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 $f: \Sigma \to \Delta^*$, $f(a)=R_a$
  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

| | |
|---|---|
| Union | Oui |
| Intersection | Oui |
| Etoile | Oui |
| Concaténation | Oui |
| Substitution | Oui |