# Théorie des langages rationnels : THLR

## CM 5

Uli Fahrenberg

EPITA Rennes

S3 2024

# Aperçu

# Programme du cours

1. Mots, langages
2. Langages rationnels, expressions rationnelles
3. Automates finis
4. Langages non-rationnels
5. Langages reconnaissables, minimisation

# Propriétés de clôture

# Propriétés de clôture

**Théorème**

*Les langages rationnels sont clos par*

- *union, concaténation, étoile,*
- *préfixe, suffixe, facteur,*
- *intersection et complémentation.*

# Propriétés de clôture

**Théorème**

*Les langages rationnels sont clos par*

- *union, concaténation, étoile,* ✓
- *préfixe, suffixe, facteur,* ? ✓
- *intersection et complémentation.* ? ✓

# Clôture par préfixe etc.

## Lemme

*Soit $L \subseteq \Sigma^*$ un langage rationnel, alors* Pref$(L)$, Suff$(L)$ *et* Fact$(L)$ *sont rationnels aussi.*

## Démonstration.

1. Soit $A$ un automate fini tel que $L = L(A)$.
2. Notons $A = (\Sigma, Q, Q_0, F, \delta)$.

# Clôture par préfixe etc.

**Lemme**

*Soit $L \subseteq \Sigma^*$ un langage rationnel, alors* Pref$(L)$, Suff$(L)$ *et* Fact$(L)$ *sont rationnels aussi.*

**Démonstration.**

1. Soit $A$ un automate fini tel que $L = L(A)$.
2. Notons $A = (\Sigma, Q, Q_0, F, \delta)$.
3. Soient
   - $\text{pref}(A) = (\Sigma, Q, Q_0, Q, \delta)$,
   - $\text{suff}(A) = (\Sigma, Q, Q, F, \delta)$,
   - $\text{fact}(A) = (\Sigma, Q, Q, Q, \delta)$.
4. Alors $L(\text{pref}(A)) = \text{Pref}(L(A))$, $L(\text{suff}(A)) = \text{Suff}(L(A))$ et $L(\text{fact}(A)) = \text{Fact}(L(A))$.

# Clôture par complémentation

**Lemme**

*Soit $L \subseteq \Sigma^*$ un langage rationnel, alors $\overline{L} = \Sigma^* \setminus L$ est rationnel aussi.*

**Démonstration.**

1. Soit $A$ un automate fini tel que $L = L(A)$.

# Clôture par complémentation

**Lemme**

*Soit $L \subseteq \Sigma^*$ un langage rationnel, alors $\overline{L} = \Sigma^* \setminus L$ est rationnel aussi.*

**Démonstration.**

1. Soit $A$ un automate fini déterministe complet tel que $L = L(A)$.

# Clôture par complémentation

**Lemme**

*Soit $L \subseteq \Sigma^*$ un langage rationnel, alors $\overline{L} = \Sigma^* \setminus L$ est rationnel aussi.*

**Démonstration.**

1. Soit $A$ un automate fini déterministe complet tel que $L = L(A)$.
2. Notons $A = (\Sigma, Q, q_0, F, \delta)$.

# Clôture par complémentation

**Lemme**

*Soit $L \subseteq \Sigma^*$ un langage rationnel, alors $\overline{L} = \Sigma^* \setminus L$ est rationnel aussi.*

**Démonstration.**

1. Soit $A$ un automate fini déterministe complet tel que $L = L(A)$.
2. Notons $A = (\Sigma, Q, q_0, F, \delta)$.
3. Soit $A' = (\Sigma, Q, q_0, Q \setminus F, \delta)$.
4. Alors $L(A') = \overline{L(A)} = \overline{L}$.

# Clôture par intersection

## Corollaire

*Soient $L_1$ et $L_2$ des langages rationnels, alors $L_1 \cap L_2$ l'est aussi.*

## Démonstration.

Par la loi de de Morgan,

# Clôture par intersection

**Corollaire**

*Soient $L_1$ et $L_2$ des langages rationnels, alors $L_1 \cap L_2$ l'est aussi.*

**Démonstration.**

Par la loi de de Morgan, $L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$.

- aussi, construction directe par produit d'automates finis déterministes complets

# Minimisation

# Minimisation

Soit $L$ un langage rationnel. On s'intéresse aux questions d'existence et unicité d'un automate fini minimal qui reconnait $L$.

- très compliqué pour des automates non-déterministes
- p.ex. [Brzozowski, Tamm : Theory of átomata. Theor. Comput. Sci. 539 : 13-27 (2014)]
- mais pour des automates finis déterministes :

**Théorème**

*Pour tout langage rationnel $L$ il existe un unique automate fini déterministe complet $A$ avec nombre d'états minimal t.q. $L = L(A)$.*

# Indistinguabilité

Soit $A = (\Sigma, Q, Q_0, F, \delta)$ un automate fini.

- on note $L_q$, pour tout $q \in Q$, le langage reconnu par $A$ depuis état initial $q$

**Définition**

Deux états $q_1, q_2 \in Q$ sont indistinguables si $L_{q_1} = L_{q_2}$.

- si deux états sont indistinguables, on peut les identifier
- écrivons $q_1 \sim q_2$ si $q_1$ et $q_2$ sont indistinguables : une relation d'équivalence dans $Q$

**Théorème**

*Si $A$ est déterministe complet, alors l'automate quotient $A_{/\sim}$ est l'automate fini déterministe complet minimal pour $L(A)$.*

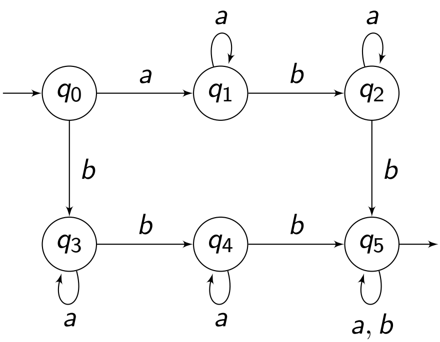
# L'automate quotient

## Définition

Soit $A = (\Sigma, Q, Q_0, F, \delta)$ un automate fini et $R \subseteq Q \times Q$ une relation d'équivalence. L'automate quotient de $A$ sur $R$ est $A_{/R} = (\Sigma, Q', Q'_0, F', \delta')$ defini comme suite :

- $Q' = Q_{/R} = \{[q]_R \mid q \in Q\}$, l'ensemble de classes d'équivalence de $R$
- $Q'_0 = \{[q_0]_R \mid q_0 \in Q_0\}$
- $F' = \{[q_f]_R \mid q_f \in F\}$
- $\delta' = \{([p]_R, a, [q]_R) \mid (p, a, q) \in \delta\}$

# Exemple ( sur tableau )



# Démonstration

- rappel : $q_1 \sim q_2$ ssi $L_{q_1} = L_{q_2}$

**Théorème ( rappel )**

*Soit $A$ un automate fini déterministe complet, alors $A_{/\sim}$ est l'unique automate fini déterministe complet minimal pour $L(A)$.*

**Démonstration.**

1. $A_{/\sim}$ est déterministe complet et $L(A_{/\sim}) = L(A)$. ( *Pourquoi ?* )
2. On finit la démonstration par le lemme suivant.

**Lemme**

*Si $A$ et $A'$ sont deux automates finis déterministes complets avec $L(A) = L(A')$, alors $A_{/\sim}$ et $A'_{/\sim}$ sont isomorphes.*

- *Qu'est-ce que c'est « isomorphe » ?*
- *Pourquoi le lemme démontre-t-il le théorème ?*

# Démonstration, suite

- rappel : $q_1 \sim q_2$ ssi $L_{q_1} = L_{q_2}$

## Lemme ( rappel )

*Si $A$ et $A'$ sont deux automates finis déterministes complets avec $L(A) = L(A')$, alors $A_{/\sim}$ et $A'_{/\sim}$ sont isomorphes.*

## Démonstration.

1. On note $A_{/\sim} = (\Sigma, Q, q_0, F, \delta)$ et $A'_{/\sim} = (\Sigma, Q', q'_0, F', \delta')$.
2. Soit $R \subseteq Q \times Q'$ la relation défini par $q \, R \, q'$ ssi $L_q = L_{q'}$.
3. $L_{q_0} = L(A_{/\sim}) = L(A) = L(A') = L(A'_{/\sim}) = L_{q'_0}$, alors $q_0 \, R \, q'_0$.
4. $q_1 \, R \, q'$ et $q_2 \, R \, q' \Rightarrow L_{q_1} = L_{q'} = L_{q_2} \Rightarrow q_1 \sim q_2 \Rightarrow q_1 = q_2$ ;
5. $q \, R \, q'_1$ et $q \, R \, q'_2 \Rightarrow L_{q'_1} = L_q = L_{q'_2} \Rightarrow q'_1 \sim q'_2 \Rightarrow q'_1 = q'_2$ ;
6. alors $R$ est une bijection.
7. *Est-ce qu'on a fini ?*

# Myhill-Nerode

- même chose qu'avant, sans passer par un automate :

**Définition**

Soit $L \subseteq \Sigma^*$ et $u, v \in \Sigma^*$, alors $u$ et $v$ sont indistinguables dans $L$ si pour tout $w \in \Sigma^*$, $uw \in L \iff vw \in L$.

- écrivons $u \equiv_L v$ si $u$ et $v$ sont indistinguables dans $L$ : une relation d'équivalence dans $\Sigma^*$

**Théorème (Myhill-Nerode)**

*Un langage $L \subseteq \Sigma^*$ est rationnel ssi le nombre $n$ de classes d'équivalence de $\equiv_L$ est fini. Dans ce cas, $n$ est aussi le nombre d'états de l'automate fini déterministe complet minimal reconnaissant $L$.*

- voir le poly pour une démonstration
- l'automate a comme états les classes d'équivalence de $\equiv_L$

# Algorithme de minimisation

Soit $A = (\Sigma, Q, q_0, F, \delta)$ un automate fini déterministe complet.

- rappel : $q_1 \sim q_2$ ssi $L_{q_1} = L_{q_2}$

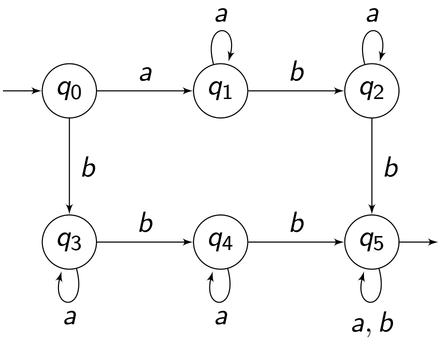
**Théorème ( rappel )**

*$A_{/\sim}$ est l'unique automate fini déterministe complet minimal pour $L(A)$.*

**Algorithme**

1. Initialiser avec deux classes d'équivalence : $F$ et $Q \setminus F$
2. Itérer jusqu'à stabilisation :
   - pour tout $p, q \in Q$ dans une même classe d'équivalence $C$ :
   - s'il existe $p \xrightarrow{a} p'$ et $q \xrightarrow{a} q'$ tel que $p'$ et $q'$ ne sont pas dans la même classe :
   - séparer $C$ en classes $C_1 \ni p$ et $C_2 \ni q$

# Exemple ( sur tableau )



# Égalité est décidable

## Corollaire

*Il existe un algorithme qui, pour automates finis $A_1$ et $A_2$, décide si $L(A_1) = L(A_2)$.*

## Démonstration.

1. Convertir $A_1$ et $A_2$ en automates finis déterministes complets minimaux.
2. Décider si $A_1$ et $A_2$ sont isomorphes.

# Langages reconnaissables

# Théorème de Kleene

**Théorème (Kleene)**

*Un langage $L \subseteq \Sigma^*$ est rationnel ssi il est reconnaissable.*

**Démonstration.**

$\Longrightarrow$ algorithme de Thompson : convertir une expression rationelle dans un automate fini à transitions spontanées

$\Longleftarrow$ algorithme de Brzozowski & McCluskey : convertir un automate fini dans une expression rationelle $\longleftarrow$ maintenant

- outil : automates finis généralisés, avec transitions étiquetées en expressions rationnelles

# Automates finis généralisés

## Définition

Un automate fini généralisé est une structure $(\Sigma, Q, Q_0, F, \delta)$ où

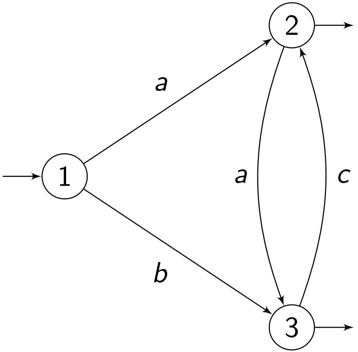
- $\Sigma$ est un ensemble fini de symboles,
- $Q$ est un ensemble fini d'états,
- $Q_0 \subseteq Q$ est l'ensemble des états initiaux,
- $F \subseteq Q$ est l'ensemble des états finaux, et
- $\delta \subseteq Q \times RE(\Sigma) \times Q$ est la relation de transition.

- un calcul dans $A$ : $\sigma = q_1 \xrightarrow{e_1} q_2 \xrightarrow{e_2} \cdots \xrightarrow{e_{n-1}} q_n$
- l'étiquette d'un calcul : $\lambda(\sigma) = e_1 e_2 \dots e_{n-1} \in RE(\Sigma)$
- un calcul réussi : $q_1 \in Q_0$ et $q_n \in F$
- Le langage reconnu par $A$ :
$$L(A) = \bigcup \{ L(\lambda(\sigma)) \mid \sigma \text{ calcul réussi dans } A \}$$

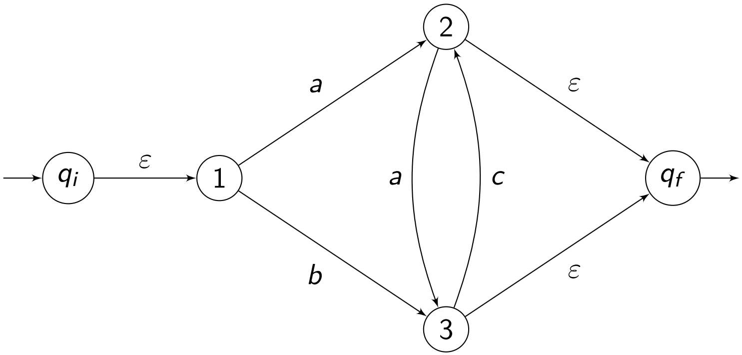
# Algorithme de Brzozowski & McCluskey

1. Soit $A$ un automate fini
2. « Convertir » $A$ en automate fini généralisé
3. Convertir $A$ en automate fini généralisé pure :
   - une unique transition entre chaque pair d'états
   - un état initial unique $q_0$ sans transitions entrantes
   - un état final unique $q_f$ sans transition sortante
4. `while` $Q \neq \{q_0, q_f\}$ :
   - supprimer un état $q \notin \{q_0, q_f\}$
   - corriger étiquettes
5. `return` l'étiquette de la transition unique

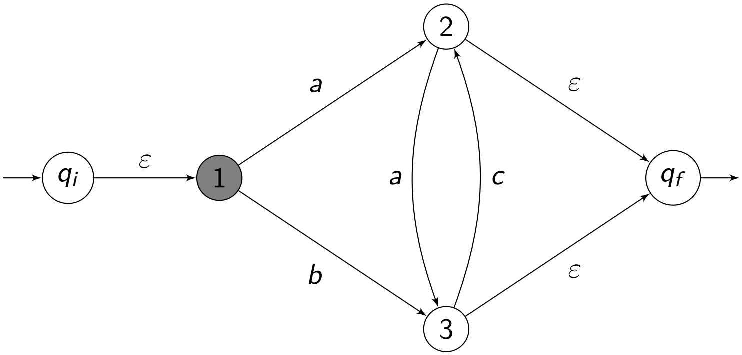
# Exemple



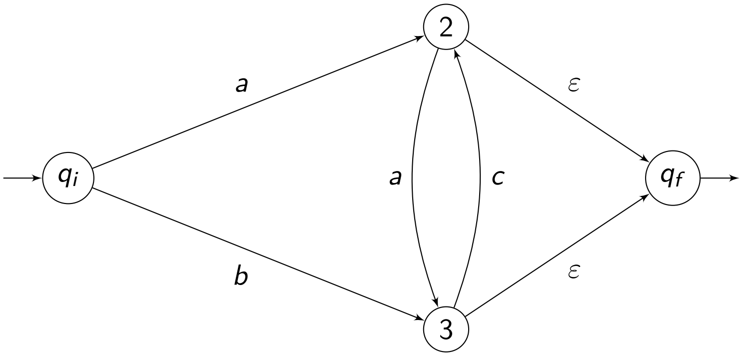
# Exemple



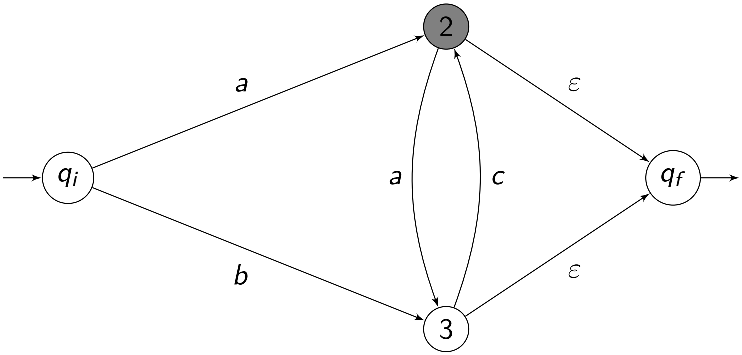
# Exemple



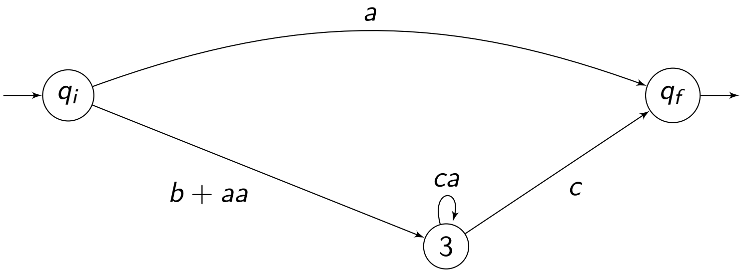
# Exemple



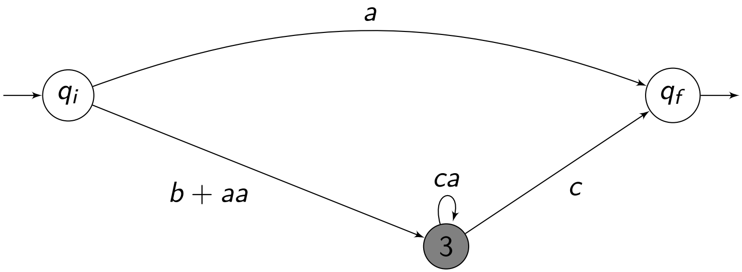
# Exemple



# Exemple



# Exemple



# Exemple

$$\longrightarrow q_i \xrightarrow{\quad a + (b + aa)(ca)^* c \quad} q_f \longrightarrow$$

# Algorithme de Brzozowski & McCluskey, détail

1. Soit $(\Sigma, Q, Q_0, F, \delta)$ un automate fini
2. Convertir $A$ en automate fini généralisé pure $(\Sigma, Q', q_0, f, \Delta)$ :
   - une unique transition entre chaque pair d'états
   - un état initial unique $q_0$ sans transitions entrantes
   - un état final unique $q_f$ sans transition sortante

# Algorithme de Brzozowski & McCluskey, détail

1. Soit $(\Sigma, Q, Q_0, F, \delta)$ un automate fini
2. Convertir $A$ en automate fini généralisé pure $(\Sigma, Q', q_0, f, \Delta)$ :
    - une unique transition entre chaque pair d'états
    - un état initial unique $q_0$ sans transitions entrantes
    - un état final unique $q_f$ sans transition sortante

- $Q' = Q \cup \{q_0, q_f\}$ pour $q_0, q_f \notin Q$
- $\Delta : Q' \times Q' \to RE(\Sigma)$
- $\Delta(q_1, q_2) = \sum \{a \mid (q_1, a, q_2) \in \delta\}$ pour $q_1, q_2 \in Q$
    - c.à.d. $\Delta(q_1, q_2) = \varnothing$ si $\{a \mid (q_1, a, q_2) \in \delta\} = \emptyset$
- $\Delta(q_i, q_2) = \begin{cases} \varepsilon & \text{si } q_2 \in Q_0 \\ \varnothing & \text{sinon} \end{cases}$ $\qquad \Delta(q_1, q_f) = \begin{cases} \varepsilon & \text{si } q_1 \in F \\ \varnothing & \text{sinon} \end{cases}$

# Algorithme de Brzozowski & McCluskey, détail

1. Soit $(\Sigma, Q, Q_0, F, \delta)$ un automate fini
2. Convertir $A$ en automate fini généralisé pure $(\Sigma, Q', q_0, f, \Delta)$ :
3. `while` $Q \neq \{q_0, q_f\}$ :
   - supprimer un état $q \notin \{q_0, q_f\}$
   - corriger étiquettes

# Algorithme de Brzozowski & McCluskey, détail

1. Soit $(\Sigma, Q, Q_0, F, \delta)$ un automate fini
2. Convertir $A$ en automate fini généralisé pure $(\Sigma, Q', q_0, f, \Delta)$ :
3. `while` $Q \neq \{q_0, q_f\}$ :
   - supprimer un état $q \notin \{q_0, q_f\}$
   - corriger étiquettes

- $Q' \leftarrow Q' \setminus \{q\}$
- pour tout $p, r \in Q'$ ( donc aussi pour $p = q$ ! ) :
- $\Delta(p, r) \leftarrow \Delta(p, r) + \Delta(p, q)\Delta(q, q)^*\Delta(q, r)$

# Algorithme de Brzozowski & McCluskey, détail

1. Soit $(\Sigma, Q, Q_0, F, \delta)$ un automate fini
2. Convertir $A$ en automate fini généralisé pure $(\Sigma, Q', q_0, f, \Delta)$ :
3. `while` $Q \neq \{q_0, q_f\}$ :
   - supprimer un état $q \notin \{q_0, q_f\}$
   - corriger étiquettes

- $Q' \leftarrow Q' \setminus \{q\}$
- pour tout $p, r \in Q'$ ( donc aussi pour $p = q$ ! ) :
- $\Delta(p, r) \leftarrow \Delta(p, r) + \Delta(p, q)\Delta(q, q)^*\Delta(q, r)$

4. `return` l'étiquette de la transition unique

- donc $\Delta(q_i, q_f)$

# Exercice

Utiliser

1. l'algorithme de Thompson pour convertir l'expression rationnelle $a(b^*a + b)$ en automate fini à transitions spontanées $A$ ;
2. l'algorithme de Brzozowski et McCluskey pour reconvertir $A$ en expression rationnelle.

# Conclusion

# Récapitulatif

1. Mots, langages
2. Langages rationnels, expressions rationnelles
3. Automates finis
4. Langages non-rationnels
5. Langages reconnaissables, minimisation

- poly chapitres 1-4
- moins 2.3.2, 2.3.5, 2.4.4 et 4.1.3

# Théorème de Kleene

**Théorème (Kleene)**

*Un langage $L \subseteq \Sigma^*$ est rationnel ssi il est reconnaissable.*

$$
\begin{array}{ccc}
\underline{\text{syntaxe}} & & \underline{\text{sémantique}} \\
\text{aut. finis dét. complets} & & \text{langages reconnaissables} \\
\subsetneq & & \| \\
\text{aut. finis déterministes} & & \text{langages reconnaissables} \\
\subsetneq & L(\cdot) & \| \\
\text{automates finis} & \longrightarrow & \text{langages reconnaissables} \\
\subsetneq & & \| \\
\text{aut. finis à trans. spontanées} & & \text{langages reconnaissables} \\
\hline
 & & \| \\
\text{expressions rationnelles} & & \text{langages rationnelles}
\end{array}
$$

# Applications

- automate fini $\hat{=}$ algorithme en mémoire constante
- lien vers les algorithmes online / streaming
- parsage, analyse lexicale, `grep` etc. : expression rationnelle $\rightsquigarrow$ automate fini déterministe / non-déterministe ( !)
- apprentissage par automates : automate fini $\hat{=}$ représentation compacte
- traduction automatique : automates probabilistes
- vérification : modélisation par automates probabilistes / pondérés / temporisés / hybrides / etc.
- et après ? langages algébriques, automates à pile, analyse syntaxique, compilation $\rightsquigarrow$ THL

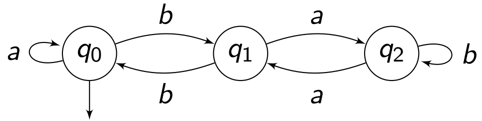
# Pour aller plus loin

- ( le poly ! )
- O. Carton, *Langages formels*. Vuibert 2014
- J. Sakarovitch, *Eléments de théorie des automates*. Vuibert 2003
- T.A. Sudkamp, *Languages and Machines*. Addison-Wesley 2005
- D.C. Kozen, *Automata and Computability*. Springer 2012

- algèbre de Kleene pour la vérification des programmes
- Tony Hoare et.al : Concurrent Kleene algebra pour la vérification des systèmes distribués

# Pour aller plus loin

$A$ :




- soit $L_i$ le langage reconnu par $A$ avec état initial $q_i$
- alors
$$\begin{aligned} L_0 &= \{a\}L_0 \cup \{b\}L_1 \cup \{\varepsilon\} \\ L_1 &= \{a\}L_2 \cup \{b\}L_0 \\ L_2 &= \{a\}L_1 \cup \{b\}L_2 \end{aligned}$$
- comme équation matrice-vecteur : $\begin{bmatrix} L_0 \\ L_1 \\ L_2 \end{bmatrix} = \begin{bmatrix} a & b & \emptyset \\ b & \emptyset & a \\ \emptyset & a & b \end{bmatrix} \begin{bmatrix} L_0 \\ L_1 \\ L_2 \end{bmatrix} \cup \begin{bmatrix} \varepsilon \\ \emptyset \\ \emptyset \end{bmatrix}$

### Slogan

Un automate fini sur $\Sigma$ est une transformation affine dans un espace vectoriel sur le demi-anneau $\mathcal{P}(\Sigma^*)$.

# Pour aller plus loin

Un demi-anneau est une structure algébrique $(S, \oplus, \otimes, \mathbb{0}, \mathbb{1})$ telle que

- $(S, \oplus, \mathbb{0})$ forme un monoïde commutatif,
- $(S, \otimes, \mathbb{1})$ forme un monoïde,
- $x(y \oplus z) = xy \oplus xz$, $(x \oplus y)z = xz \oplus yz$ et $x\mathbb{0} = \mathbb{0}x = \mathbb{0}$

$S$ est idempotent si $x \oplus x = x$.

## Théorème

*L'ensemble de langages finis forme le demi-anneau idempotent libre.*

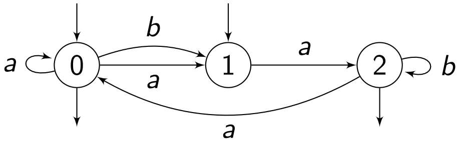
Une algèbre de Kleene est un demi-anneau idempotent $S$ équipé avec toutes les sommes géométriques $\bigoplus_{n\geq 0} x^n$, pour tout $x \in S$, et telle que $x\left(\bigoplus_{n\geq 0} y^n\right)z = \bigoplus_{n\geq 0}(xy^n z)$ pour tout $x, y, z \in S$.

## Théorème

*L'ensemble de langages rationnels forme l'algèbre de Kleene libre.*

# Pour aller plus loin

$A$ :




## Définition (variante)

Un automate fini ( pondéré ) avec $n$ états sur un demi-anneau $S$ est composé d'une matrice $\Delta \in S^{n \times n}$ et deux vecteurs $i, f \in S^n$.

- ici : $S = \mathcal{P}(\Sigma^*)$, $\Delta = \begin{bmatrix} \{a\} & \{a,b\} & \emptyset \\ \emptyset & \emptyset & \{a\} \\ \{a\} & \emptyset & \{b\} \end{bmatrix}$, $i = \begin{bmatrix} \{\varepsilon\} \\ \{\varepsilon\} \\ \emptyset \end{bmatrix}$, $f = \begin{bmatrix} \{\varepsilon\} \\ \emptyset \\ \{\varepsilon\} \end{bmatrix}$

## Théoréme / définition

Si $S$ est une algèbre de Kleene, alors $S^{n \times n}$ l'est aussi, et le langage reconnu par $A$ est $L(A) = i\Delta^* f$.


That's all Folks!