CSC4251_4252 : Automate Fini

Pascal Hennequin, J. Paul Gibson, Denis Conan

Novembre 2024

Sommaire

1 Abrégé de la théorie des langages

- **Définitions**
  - Alphabet Σ : ensemble fini de lettres
  - Mot w sur Σ : suite finie de lettres de Σ
  - Σ^* : ensemble de tous les mots = monoïde libre
  - Langage L : ensemble de mots inclus dans Σ^*
  - Mot vide ϵ, Langage vide $\emptyset \neq \{\epsilon\}$
- **Opérations**
  - Alternative (union), Concaténation (produit),
  - Fermeture de Kleene (étoile) : $L^* = \{\epsilon\} \cup L \cup LL \cup LLL$
  - Aussi Intersection, Complémentaire, Quotient, etc.
- **Question**
  - $w \in L$?, $L = \emptyset$?, L fini ?, $L1 = L2$?, etc.
    décidable, calculable, facilement calculable ?

1.1 Décidable, calculable, facilement calculable ?

Langage	Rationnel	Algébrique déterministe	Algébrique	Contextuel	Récursif	Récursivement énumérable
Machine	Automate fini	Automate à pile		Automate borné	Mach. de Turing qui s'arrête	Machine de Turing
Spécification	Expression régulière	Grammaire *context-free*		Grammaire *context-sensitive*	Grammaire générale ou *phrase-struture*	
Exemples	a^p et p paire	$a^n b^n$ $a^n b^n c^p d^p$ Dyck ([] ())	Palindrome $a^n b^n c^p \cup$ $a^p b^n c^n$ a^p et p carré	Papou $a^n b^n c^n$, $a^n b^p c^n d^p$ a^p et p premier	$\mathrm{diag(CSL)}$ Ackermann	$\overline{\mathrm{diag(TM)}}$ pas $\mathrm{diag(TM)}$
Déterministe $\equiv$ non déterm.	OUI	NON, mais décidable ; Ambiguïté indécidable		Problème ouvert	OUI	OUI
$w \in L$	$O(n)$	$O(n)$	$O(n^3)$	PSPACE	EXPSPACE	Indécidable
L rationnel	OUI	Décidable	Indécidable	Indécidable	Indécidable	Indécidable
$L = \emptyset$	Décidable	PTIME	PTIME	Indécidable	Indécidable	Indécidable
$L = \Sigma^*$	Décidable	Décidable	Décidable	Indécidable	Indécidable	Indécidable
$L_1 = L_2$, $L_1 \subset L_2$	Décidable	Décidable	Indécidable	Indécidable	Indécidable	Indécidable
Union, concat. étoile	Fermé	Non clos	Fermé	Fermé	Fermé	Fermé
Intersection	Fermé	Non clos	Non clos	Fermé	Fermé	Fermé
Complémentaire	Fermé	Fermé	Non clos	Fermé	Fermé	Fermé

2 Automate fini

- **Principe**

```
int état
tant que pas_fini {
    lire un caractère c
    selon état et c
        changer état
}
```

- **Automate Fini** (*Finite State Automaton*)

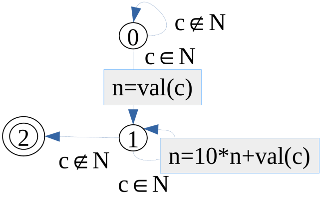
$A = (\Sigma, Q, q_0, Q^T, \delta)$ avec

  - $\Sigma = \{c_0, ..., c_{m-1}\}$, alphabet d'entrée (ou **événements**)
  - $Q = \{q_0, ..., q_{n-1}\}$, ensemble fini d'**états**
  - q_0 **état initial**
  - $Q^T \subset Q$, ensemble des états **terminaux** (ou **accepteurs**)
  - $\delta : Q \times \Sigma \to Q$, **fonction de transition**
    - $q = \delta(p, c)$: transition de p à q sur lecture de c

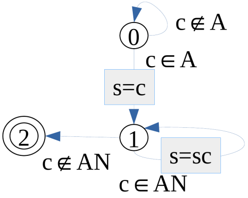
2.1 Exemples d'automates finis

Étant donné `N = [0-9]`, `A = [a-zA-Z]`, et `AN = A ∪ N`

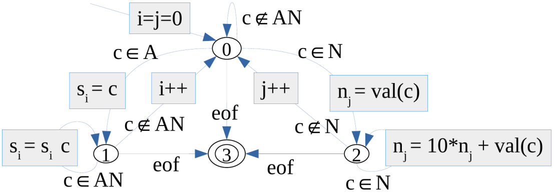
Reconnaître un entier



Reconnaître un identificateur



Reconnaître tous les identificateurs et tous les entiers



2.2 Représentations d'un automate

- Diagramme d'états

État initial	$\longrightarrow$ 0
État final	*n*
Transitions	$\xrightarrow{\text{événement} \quad \text{action}}$

- Table états/transitions

état \ car.lu	c $\in$ A	c $\in$ N	c $\notin$ AN	c $=$ EOF [1]
0	1	2	0	3
1	1	2	0	3
2	0	2	0	3

1. EOF = *End Of File*

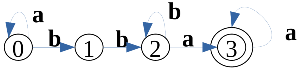
3 Reconnaissance par un automate

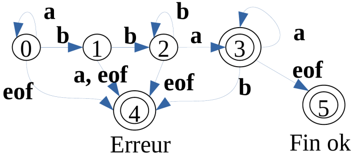
Langage reconnu par un automate

- Les mots dont la séquence de caractères mène l'automate de l'état initial q_0 à un état terminal $q \in Q^T$

Exemple

- Sur l'alphabet $\Sigma = \{\mathrm{a}, \mathrm{b}\}$, reconnaître une chaîne
  - Contenant une seule chaîne d'au moins deux « b » consécutifs
  - Éventuellement précédée d'un ou plusieurs « a »
  - Et obligatoirement suivie d'au moins un « a »
- $L = \{\mathrm{a}^n\mathrm{b}^m\mathrm{a}^p, n \geq 0, m > 1, p > 0\}$
- Est-ce une expression régulière ?





4 Reconnaissance par des outils

- **Reconnaissance multiple de chaînes valides dans un texte**
  - Glouton versus Récalcitrant
    - Retourner la forme la plus étendue (glouton, *greedy*) ou la plus courte (récalcitrant, *reluctant*, *lazy*)
    - `[0-9]+` sur « `1234` » ?
    - `< .* >` sur « `<b>bold</b>` » ?
  - Avec ou sans recouvrement (aussi : avec ou sans trou)
    - `a*bb+a+` sur « `abbaabba` » ?
      - « `abbaa` » et « `aabba` » ou sans recouvrement : « `abbaa` » et « `bba` »
  - Plusieurs automates simultanément
    - `[0-9]+` et `[a-zA-Z]+` en même temps
- **Il existe des algorithmes pour outiller l'automate de base afin d'avoir les différents comportements**

5 Déterministe *versus* Non Déterministe

- **Machine non déterministe**

Déterministe	Non déterministe
$\forall$ couple (p,c), $\exists!$ transition p $\xrightarrow{c}$ q	– Plusieurs transitions possibles p $\xrightarrow{c}$ q1, p $\xrightarrow{c}$ q2 – ou des ϵ-transitions (sans caractère lu) p $\longrightarrow$ q
Exécution linéaire	Exécution par exploration de toutes les possibilités. Un mot est reconnu s'il existe un chemin vers un état terminal.

- **Théorème**
  - Pour tout automate fini non déterministe (NFA), il existe un automate fini déterministe (DFA) reconnaissant le même langage

6 Langage Rationnel et Automate Fini I

- **Théorème de Kleene (1956) : Expressions Régulières $\Longleftrightarrow$ Automates Finis**
- Preuve : cyclique et constructive

1. Regexp $\rightsquigarrow$ NFA avec ϵ-transition

   Traduction concaténation/union/étoile sur les automates
2. NFA avec ϵ-transition $\rightsquigarrow$ NFA sans ϵ-transition

   Algorithme de fermeture transitive
3. NFA sans ϵ-transition $\rightsquigarrow$ DFA

   « Construction par sous-ensemble »
   États du DFA $=$ ensemble des parties des états du NFA
4. DFA $\rightsquigarrow$ Regexp

   Récurrence à 3 indices (McNaughton & Yamada)

6 Langage Rationnel et Automate Fini II

- **Théorème de minimisation**
  - Il existe des algorithmes (Nérode, Hopcroft...) pour construire des automates déterministes de taille minimale

    $\Longrightarrow$ 3bis) DFA $\rightsquigarrow$ DFA minimal

- **Corolaire pratique**
  - Outils de reconnaissance par expressions régulières = 1 + 2 + 3 + 3bis

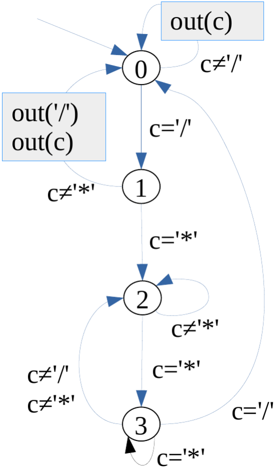
- **Théorèmes faciles**

1. Le complémentaire d'un langage rationnel est rationnel
2. L'intersection de deuxlangages rationnels est rationnel

Preuve ?

7 Petit commentaire

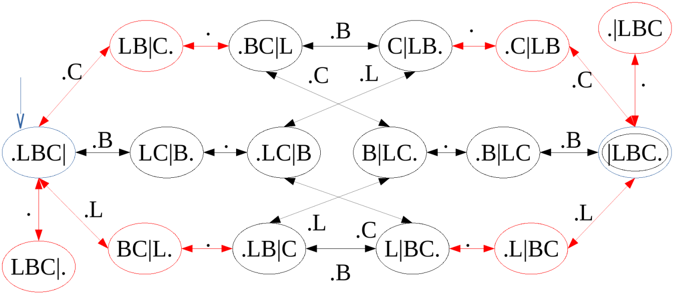
- **Supprimer les commentaires** `/* ... */`
  - Expression régulière
    - — `"/*" .* "*/"` !!!
    - — `"/*"` puis tout sauf `"*/"` et ensuite `"*/"` ???
  - Plus facile : un automate



- Des outils comme lex, flex, JFlex offrent aussi une vision automate : *Start-Condition*, Super-état

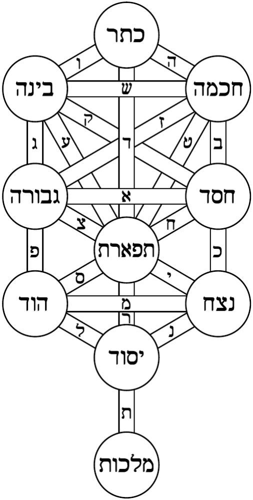
8 Illustrations I

- **LuBuChu**
  - une rivière, une barque, un fermier, un loup, un bouc, un chou



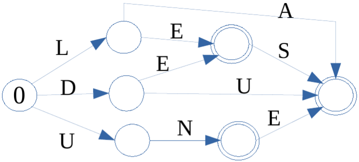
8 Illustrations II

- L'arbre de vie
  - 10 Sephiroth et 22 sentiers
  - Azriel de gérone XIIIème siècle

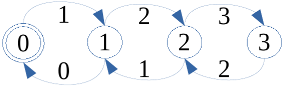


8 Illustrations III

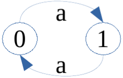
- **Langage fini**
  Structure de données pour dictionnaire (de scrabble)

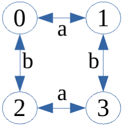


- **Ascenseur**



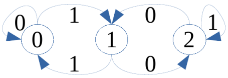
- **Parité**



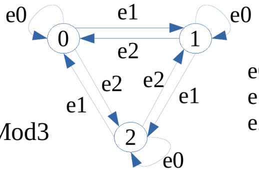


8 Illustrations IV

Numération et Modulo



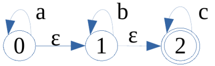
Bin-Mod3



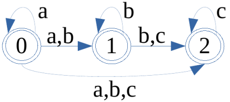
Dec-Mod3

e0= 0|3|6|9
e1= 1|4|7
e2= 2|5|8

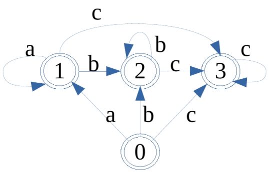
Déterminisme



Non déterministe avec ε



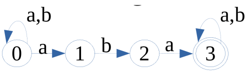
Non déterministe sans ε



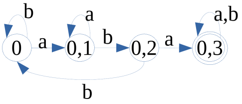
Déterministe

8 Illustrations V

- **Négation de** « .*aba.* »
  Exercice TP ABBA-3 : preuve du corrigé



Non déterminisme



Déterminisme complet

- Négation :

  États finaux = 0, {0,1}, {0,2}

  Re(0→0) = b*(aa*bbb*)* = b*(a+bb+)*

  Re(0→{0,1}) = Re(0→0)aa*

  Re(0→{0,2}) = Re(0→0)aa*b

  Re(0→final) = b*(a+bb+)*(ϵ | aa* | aa*b)

$\Longrightarrow$ Not (.*aba.*) = b*(a+bb+)*(a* | a+b)

  aussi = b*(a+bb+)*a*b?