

Langages Formels & Automates

J. Bond

bureau 433, bond@essi.fr, ☎ 0492965141

Les enseignants

Groupe	Enseignant	e-mail
1	M. Marc Gaetano	gaetano@essi.fr
2	J. Bond	bond@essi.fr
3	J. Bond	bond@essi.fr
4	M. Michel Cosnard	Michel.Cosnard@sophia.inria.fr
5	M. Michel Cosnard	Michel.Cosnard@sophia.inria.fr

Bibliographie

- Danièle BEAUQUIER, Jean BERSTEL et Philippe CHRETIENNE: *Éléments d'algorithmique*, Masson 1992

(ce livre est épuisé, mais téléchargeable sur le Web à l'adresse
<http://www-igm.univ-mlv.fr/~berstel/Elements/Elements.html>)

Il comporte plusieurs chapitres qui peuvent (doivent ?) vous intéresser à différents titres (cours d'algorithmique, maths discrètes, ...) et le chapitre 9 qui concerne les automates.

Bibliographie

- John HOPCROFT, Jeffrey ULLMAN : *Introduction to Automata Theory and Computation*, Addison Wesley, 1979.

Nouvelle édition, revue et corrigée :

- John HOPCROFT, Rajeev MOTWANI, Jeffrey ULLMAN : *Introduction to Automata Theory, Languages and Computation*, Addison Wesley, 2001.

Bibliographie

- Michael SIPSER : *Introduction to the Theory of Computation*, PWS publishing comp. 1997.
- Jacques STERN : *Fondements mathématiques de l'informatique*, McGraw Hill, 1990.
- Pierre WOLPER : *Introduction à la calculabilité*, Inter Éditions 1991 (deuxième édition : Dunod, 2001).

Plan du cours

- Concepts de base (langages rationnels)
- Automates finis
 - Déterministes
 - Non-déterministes
- Théorème de Kleene (rationnel $\Leftrightarrow$ reconnu)
- Grammaires
- Automates à pile

Construire/ Engendrer

- Comment engendrer une suite de symboles?
- Système qui engendre tous les mots d'un langage (et seulement ceux-ci)

Cours de mathématiques discrètes:
définitions inductives. P.e. sur $\Sigma=\{0,1\}$,
MBP : $0 \in L$
 $m \in L \Rightarrow m0 \in L$



Difficile pour certains langages p.e. langue naturelle

Reconnaître

- Mécaniquement
- Par quels outils?
- Automate

Pourquoi ?

Les automates sont partout:

- Dans les machines à café pour compter les pièces
- Dans les téléphones cellulaires (pour le décodage des messages)
- Dans les ordinateurs (les compilateurs ne sont que des automates améliorés)
- Dans votre machine à laver (pour le choix du programme)
- Pour interpréter vos pages html (c'est ce que sont en train de faire les ESSI2 en compilation)

Reconnaître

- Mécaniquement
- Par quels outils?
- Automate
- Quelles propriétés pour reconnaître facilement?

Utilité des langages formels

- Spécification de langages de programmation
- Compilation
- Recherche des motifs dans
 - Un texte
 - Une bases de données
 - Sur le Web
- Preuves de programmes
- Codage pour la transmission
- Décodage du génome
- Compression de textes...

Utilisation dans la théorie

- Pour coder des problèmes
 - Un **mot** = une instance d'un problème
 - Un **langage** = codage d'un problème
- **Calculabilité** :
où se situent les limites de l'informatique
- **Complexité** :
où est la frontière entre ce qui est efficace et ce qui ne l'est pas

Symbole et alphabet

- Entité de base : le symbole
pas plus de définition formelle qu'un point en géométrie
- Alphabet = ensemble FINI de symboles
 - $\Sigma = \{0,1\}$
 - $\Sigma = \{0,1,2\}$
 - $\Sigma = \{a,b,\dots,z\}$
 - $\Sigma = \{0,\dots,9\}$
 - $\Sigma = \{\text{public, class, static, void, main, if, for}, \dots\}$

Mots finis et langage

- Mot = suite finie de symboles
011010
j'aime les langages

```
class HelloWorldApp{
    public static void main(String[] args){
        System.out.println(« Bonjour »);
    }
}
```
- Langage = ensemble de mots (infini?)
langage des entiers binaires
langage des programmes java

Concepts de base

- Longueur d'un mot:
 - Nombre de symboles de la suite
 - Notation $|w|$

$$|011010| = 6$$

$$|\text{langage}| = 7$$
 - Mot de longueur nulle: le mot vide noté ε
- $i^{\text{ème}}$ symbole d'un mot:
 - Fonction rendant le $i^{\text{ème}}$ symbole d'un mot m = langage

$$m(1) = 1, m(2) = a = m(5)$$

$$m: \{1, \dots, |m|\} \rightarrow \Sigma$$

Concepts de base

- Occurrence : apparition de $l \in \Sigma$ dans m s'il existe

$$i \text{ t.q. } m(i) = l.$$
- Nombre d'occurrences : fonction qui renvoie le nombre de fois ou une lettre l apparaît dans le mot m .

$$|\text{langage}|_a = 2$$

Concaténation

- Σ^* = collection de tous les mots finis sur Σ
= ensemble de tous les mots finis
- Opération interne associée : concaténation "."

$$\Sigma^* \times \Sigma^* \rightarrow \Sigma^*$$

$$(u,v) \rightarrow u.v$$

$$u = \text{ES}, v = \text{SI}, u.v = \text{ESSI}$$
- Élément neutre: mot vide $m.\varepsilon = \varepsilon.m = m$
- concaténation = opération associative :

$$(u.v).w = u.(v.w)$$
- $(\Sigma^*, .)$ est un monoïde

Monoïde

de Wikipédia :

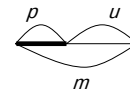
- Un **monoïde** est une structure algébrique consistant en un ensemble muni d'une loi de composition interne associative et d'un élément neutre.
- En d'autres termes, $(E, *)$ est un **monoïde** si :
 - $\forall x, y \in E, x*y \in E$ (composition interne)
 - $\forall x, y, z \in E, x*(y*z) = (x*y)*z$ (associativité)
 - $\exists e \in E \text{ t.q. } \forall x \in E, x*e = e*x = x$

Autre vision des langages

- Langage = ensemble de mots (infini?)
- Langage = sous-ensemble de Σ^*
ensemble des nombres ordinaires
ensemble des programmes Java (syntaxiquement corrects)
- Langage vide $L = \{ \} = \emptyset \neq \{ \epsilon \}$
- $L = \{ \epsilon \}$, langage du mot vide
- Langage fini de mots finis
 $L = \{ ab, ba, aca \}$
- Langage infini dénombrable de mots finis
 $L = \{ \text{mots binaires pairs} \}$

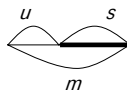
Préfixe

- Opérations sur les mots
- Un mot p de longueur l est un préfixe d'un mot m de longueur n si
 $\forall i \leq l, p(i) = m(i)$
 - si $l < n$, le préfixe est propre
 - p préfixe de m s'il existe un mot u t.q. $pu = m$



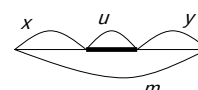
Suffixe

- Opérations sur les mots
- Un mot s de longueur l est un suffixe de m de longueur n si
 $\forall i \leq l, s(n-l+i) = m(n-l+i)$
 - si $l < n$, le suffixe est propre
 - s est un suffixe de m s'il existe u t.q. $us = m$



Facteur

- Opérations sur les mots
- Un mot u est un facteur d'un mot v si
 - u est un suffixe d'un préfixe de v .
 - u est un préfixe d'un suffixe de v .
 - u est un facteur de v ssi $\exists x, y \in \Sigma^*$ t.q. $xuv = v$



Facteur

- Opérations sur les mots
- Un mot u est un facteur d'un mot m si
 - u est un suffixe d'un préfixe de m .
 - u est un préfixe d'un suffixe de m .
 - u est un facteur de v ssi $\exists x, y \in \Sigma^*$ t.q.
 $xuy = m$

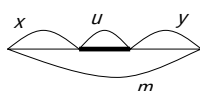


Image miroir

- Opérations sur les mots
- Appelé aussi inverse de m , notée m^{-1}
 $m = \text{sophia}$
 $m^{-1} = \text{aihpos}$

Opérations ensemblistes	
Opérations sur les langages	▪ Langage = ensemble (de mots) ▪ toutes les opérations ensemblistes ▪ <i>Exemple</i> : $L_1 = \{aa,aaa,aaaa\} = \{a^2,a^3,a^4\}$ et $L_2 = \{aa,aaaaa\} = \{a^2,a^5\}$ deux langages • $L_1 \cup L_2 = \{a^2,a^3,a^4,a^5\}$ • $L_1 \cap L_2 = \{aa\}$ • $\overline{L_2} = \{\varepsilon,a,a^3,a^4,a^6,a^7,\dots\}$

Concaténation	
Opérations sur les langages	▪ L_1 et L_2 deux langages $L_1.L_2 = \{u.v : u \in L_1 \text{ et } v \in L_2\}$ ▪ $L_1 = \{a,aa,aaa\}$ $L_2 = \{0,00\}$ $L_1.L_2 = \{a0,a00,aa0,aa00,aaa0,aaa00\}$ ▪ $L_1 = \Sigma^*$ pour $\Sigma = \{0,1\}$, $L_2 = \{0\}$ $L_1.L_2 = \Sigma^*. \{0\}$: mots binaires pairs

Opération * de Kleene	
Opérations sur les langages	▪ L un langage, L^* = concaténation de mots de L $L^0 = \{\varepsilon\}$, $L^1 = L$, $L^{i+1} = L^i.L \quad \forall i \geq 0$ $L^* = \bigcup_{i \geq 0} L^i$, $L^+ = \bigcup_{i \geq 1} L^i$ ▪ $L = \{a,0\}$ $L^2 = \{aa,a0,0a,00\}$ $L^3 = \{aaa,aa0,a0a,a00,0aa,0a0,00a,000\}$ ▪ L^* = plus petit langage de Σ^* clos pour la concaténation contenant ε et L. C'est un sous-monoïde de Σ^* ▪ Opération idempotente: $(L^*)^* = L^*$

Langages rationnels	
▪ Intérêt particulier pour la suite ▪ sous-ensemble de l'ensemble des langages ▪ définition inductive ▪ Notation simplifiée par expressions rationnelles (recherche sur le Web, etc...)	

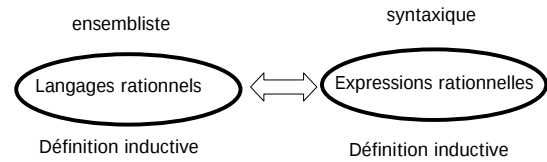
Définition inductive	
▪ Base : • $\emptyset$ est un langage rationnel • $\{\varepsilon\}$ est un langage rationnel • $\forall a \in \Sigma$, $\{a\}$ est un langage rationnel ▪ Induction : • Si R et S sont deux langages rationnels, $R \cup S$, $R.S$ et R^* sont aussi rationnels	

Expressions rationnelles (ER)	
▪ Base : • $\emptyset$ est une expression rationnelle (ER) • ε est une ER qui représente $\{\varepsilon\}$ • $\forall a \in \Sigma$, a est une ER qui représente $\{a\}$ (le mot a) ▪ Induction : Si r et s sont des ER, • $(r + s)$ est une ER qui représente $R \cup S$ • (rs) est une ER qui représente $R.S$ • (r^*) est une ER qui représente R^*	

Exemples

- $(a+b)^*$ tous les mots avec des a et des b
- $(a+b)^*ab(a+b)^* = (b^*a^*)^*ab(a+b)^*$
- $(b+ba)^*$ mots sans facteur aa et qui ne commencent pas par un a
- $(a+\varepsilon)(b+ba)^*$ mots sans facteur aa

Résumé



Relations (déjà vu en MD)

Relations sur les mots

- On s'intéresse aux couples de mots
 $R = \{(a,0), (aa,00), (aaa,000), \dots\}$
égalité des longueurs
- Propriétés d'une relation R
 - réflexivité pour tout a, aRa
 - irreflexivité il n'existe pas de a, aRa
 - transitivité aRb et bRc implique aRc
 - symétrie aRb implique bRa
 - antisymétrie aRb implique non bRa

Relations particulières

- Équivalence
 - Réflexive
 - Transitive
 - Symétrique
- Ordre (préfixe propre)
 - réflexive
 - transitive
 - antisymétrique
} pré ordre
- Ordre strict
 - Transitive
 - irreflexive
- Deux types d'ordres
 - Partiel : tous les éléments ne sont pas comparables
 - Total ou linéaire : tous les éléments sont comparables

Ordre préfixe

- $m \in \Sigma^*$ préfixe de $w \in \Sigma^*$
s'il existe $u \in \Sigma^*$ t.q. $w = m.u$
- Définition inductive:
 - Base : $\varepsilon <_p \varepsilon$
 - Induction : si $m <_p w$, alors $\forall x \in \Sigma$,
 - $xm <_p xw$ (ajout lettre à gauche)
 - $m <_p w <_p wx$ (ajout lettre à droite du plus grand)

Ordre préfixe

Ce n'est pas un ordre total dès que $|\Sigma| \geq 2$

- 10 et 11 sont incomparables
 - 10 non préfixe de 11
 - 11 non préfixe de 10

L'ordre préfixe est un ordre partiel

Ordre lexicographique

- L'ordre du dictionnaire
- Définition inductive : $(\Sigma, <)$
 - Base :
 - (1) $\forall x \in \Sigma \cup \{\varepsilon\}, \varepsilon <_{\text{lex}} x$
 - (2) $\forall x, y \in \Sigma, x < y, x <_{\text{lex}} y$
 - Induction : $\forall x \in \Sigma, \forall v, w \in \Sigma^*$,
 - (3) $v <_{\text{lex}} w \Rightarrow xv <_{\text{lex}} xw \text{ et } vx <_{\text{lex}} wx$
 - (4) $v <_{\text{lex}} w \text{ et } |w| < |v| \Rightarrow vx <_{\text{lex}} w$

Ordre lexicographique

- Ordre total (tous éléments comparables)

• Base :

- (1) $\forall x \in \Sigma \cup \{\varepsilon\}, \varepsilon <_{\text{lex}} x$
- (2) $\forall x, y \in \Sigma, x < y, x <_{\text{lex}} y$

• Induction : $\forall x \in \Sigma, \forall v, w \in \Sigma^*$,

- (3) $v <_{\text{lex}} w \Rightarrow xv <_{\text{lex}} xw \text{ et } vx <_{\text{lex}} wx$
- (4) $v <_{\text{lex}} w \text{ et } |w| < |v| \Rightarrow vx <_{\text{lex}} w$

Ordre lexicographique

- Par récurrence, on choisit (u, v) t.q. $|u| + |v|$ est minimale
 - Base:
 - $|u| + |v| = 0$: ils sont comparables par règle 1
 - Si $|u| = 0, |v| > 0$, par règle 1 et 3b ils sont comparables

Ordre lexicographique

- Étape inductive $|u| > 0 \text{ et } |v| > 0$
 - $u(1) < v(1)$, par règle 2 et 4, $u <_{\text{lex}} v(1)$ et par règle 3b, $u <_{\text{lex}} v$
 - $u(1) = v(1)$, revient à comparer u' et v' , $u = u(1)u'$ et $v = v(1)v'$ comparables par HR, u et v comparables règle 3a