909 - LANGAGES RATIONNELS ET AUTOMATES FINIS. EXEMPLES ET APPLICATIONS

Prérequis : Les notions d'alphabet, mot et langage

Dans la suite, Σ désigne un alphabet (fini, non vide)

I / Langages rationnels, langages reconnaissables

1) Définition d'un langage rationnel

def 1.1 Opérations sur les langages.

Soit L et L' deux langages.

L'union de L et L', notée $L+L'$, est le langage $\{u \in \Sigma^* \mid u \in L \text{ ou } u \in L'\}$

La concaténation de L et L' est $\{uv \in \Sigma^* \mid u \in L \text{ et } v \in L'\}$

L'étoile du langage L est définie par récurrence :

$L^0 = \{\varepsilon\}$, $\forall i \geq 0$, $L^{i+1} = LL^i$ et $L^* = \bigcup_{i \geq 0} L^i$

def 1.2 Langages rationnels. L'ensemble des langages rationnels sur Σ, noté $Rat(\Sigma)$, est la plus petite famille de langages telle que $\emptyset \in Rat(\Sigma)$, $\forall a \in \Sigma$, $\{a\} \in Rat(\Sigma)$ et close par union, concaténation et étoile.

ex 1.3 $\{\varepsilon\} = \emptyset^* \in Rat(\Sigma)$; $\Sigma = \bigcup_{a \in \Sigma} \{a\} \in Rat(\Sigma)$

def 1.3 Expressions rationnelles. On définit l'ensemble $\mathcal{E}$ des expressions rationnelles par induction : $\emptyset \in \mathcal{E}$; $\forall a \in \Sigma$, $a \in \mathcal{E}$ et

$\forall E, E' \in \mathcal{E}$, $(E+E')$, $(E.E')$ et $E^* \in \mathcal{E}$.

rq 1.4 En choisissant des priorités sur les opérations, on peut se passer de certaines parenthèses. Par exemple, on écrira plutôt $a(b+a^*)a$ que $((a.(b+a^*)).a)$

prop 1.5 Cette définition est non ambiguë

def 1.6 À toute expression rationnelle E on associe le langage $L(E)$ par induction :

$L(\emptyset) = \emptyset$; $\forall a \in \Sigma$, $L(a) = \{a\}$ et $\forall E, E' \in \mathcal{E}$, $L((E+E')) = L(E) + L(E')$,

$L((E.E')) = L(E)L(E')$ et $L(E^*) = L(E)^*$

prop 1.7 Un langage L est rationnel ssi il est dénoté par une expression rationnelle c'est à dire ssi il existe $E \in \mathcal{E}$ telle que $L(E) = L$

2) Automates finis et langages reconnaissables

def 1.8 Un automate fini est un quintuplet $(\Sigma, Q, \delta, I, F)$ où Σ est un alphabet, Q est un ensemble fini d'états, $I \subset Q$ sont les états initiaux ; $F \subset Q$ sont les états finaux et $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \to \mathcal{P}(Q)$ est la fonction de transition.

rq 1.9 . Un tel automate est une machine de Turing dont la tête ne fait que lire et ne peut que aller à droite

- on note $p \xrightarrow{a} q$ ($p, q \in Q$ et $a \in \Sigma \cup \{\varepsilon\}$) si $q \in \delta(p,a)$
- Un automate tel que $\delta : Q \times \Sigma \to \mathcal{P}(Q)$ est un automate sans ε-transitions

def 1.10 Un mot $a_1 \dots a_m$ est reconnu par l'automate $A = (\Sigma, Q, \delta, I, F)$ si il existe $q_0, q_1, \dots, q_m, q_{i_1}, \dots q_{i_m}, q_F$ et une suite de transitions $q_0 \overset{\varepsilon}{\rightsquigarrow} q_{i_1} \xrightarrow{a_1} q_1 \overset{\varepsilon}{\rightsquigarrow} q_{i_2} \xrightarrow{a_2} q_2 \dots \rightsquigarrow q_{i_m} \xrightarrow{a_m} q_m \overset{\varepsilon}{\rightsquigarrow} q_F$ où $q_0 \in I$ et $q_F \in F$

on note $L(A)$ le langage des mots reconnus par A

def 1.11 on note $Rec(\Sigma)$ l'ensemble des langages reconnus par un automate.

ex 1.12 Automate reconnaissant $(a+b)^*ab$ cf Annexe A

def 1.13 Un automate $A = (\Sigma, Q, \delta, I, F)$ sans ε-transition est dit complet si $\forall (p,a) \in Q \times \Sigma$, $card(\delta(p,a)) \geq 1$ et est dit déterministe si $|I| = 1$ et $\forall (p,a) \in Q \times \Sigma$, $card(\delta(p,a)) \leq 1$.

prop 1.14 Soit A un automate fini.

Alors il existe un automate $\tilde{A}$ sans ε-transition (resp complet) (resp déterministe) tel que $L(A) = L(\tilde{A})$.

3) Théorème de Kleene

prop 1.15 Soit A et A' deux automates reconnaissant respectivement L et L'. Alors on peut construire 3 automates reconnaissant respectivement $L+L'$, LL' et L^*. cf annexe B.

lemme 1.16 Lemme d'Arden. Soit K et L deux langages et E l'équation $X = KX+L$

Si $\varepsilon \notin K$, l'équation E admet une unique solution : K^*L

Th 1.17 Théorème de Kleene : $Rat(\Sigma) = Rec(\Sigma)$

II/ Propriétés et caractérisation des langages rationnels

1) Stabilités

Rappel 2.1 Par construction, $Rat(\Sigma)$ est stable par union, concaténation, étoile

prop 2.2 Si $L \in Rat(\Sigma)$, alors $\Sigma^* \setminus L \in Rat(\Sigma)$

prop 2.3 Si $L, L' \in Rat(\Sigma)$ alors $L \cap L' \in Rat(\Sigma)$

rq 2.4 Il existe cependant des langages non rationnels

c-ex 2.5 $Rat(\Sigma)$ n'est pas clos par sous-langage :
$\{a^n b^n \mid n \in \mathbb{N}\}$ n'est pas rationnel, bien qu'inclus dans $\Sigma^* \in Rat(\Sigma)$

2) Résiduels d'un langage

Def 2.6 Soit L un langage, $u \in L$ un mot
Le résiduel de L par u est le langage $u^{-1}L = \{ v \mid uv \in L\}$

Prop 2.7 Soit $a \in \Sigma$. Soient K, L deux langages

- $a^{-1}(K+L) = a^{-1}K + a^{-1}L$
- $a^{-1}(KL) = (a^{-1}K)L + \varepsilon(K)a^{-1}L$ où $\varepsilon(K) = \begin{cases} \varepsilon \text{ si } \varepsilon \in K \\ \emptyset \text{ sinon} \end{cases}$
- $a^{-1}(L^*) = (a^{-1}L)L^*$

ex 2.8 $a^{-1}(ab^*a+b)^* = b^*a\,(ab^*a+b)^*$

Prop 2.9 Un langage est rationnel ssi il admet un nombre fini de résiduels

def 2.10 Soit L un langage (ayant un nombre fini de résiduels).
L'automate des résiduels (ou automate minimal) de L est l'automate $\mathcal{A}_L = (\Sigma, Q, \delta, I, F)$ où

$Q = \{u^{-1}L \mid u \in \Sigma^*\}$ $\quad I = \{L\}$ $\quad F = \{u^{-1}L \mid u \in L\}$

$\delta = \{ u^{-1}L \xrightarrow{a} (ua)^{-1}L \mid u \in \Sigma^*, a \in \Sigma\}$

prop 2.11 Cet automate est déterministe, complet, reconnaît L et a le nombre minimal d'états possibles parmi les automates déterministes complets reconnaissant L.

ex 2.12 Pour $L = (ab)^*$, 3 résiduels : cf annexe C

def 2.13 Congruence de Nérode
C'est la relation d'équivalence $\sim$ sur Q^2 pour $\mathcal{A} = (\Sigma, Q, \delta, I, F)$ automate déterministe complet telle que pour $q, q' \in Q$,

$$q \sim q' \Leftrightarrow [\forall w \in \Sigma^*, (qw \in F \Leftrightarrow q'w \in F)]$$

prop 2.14 Soit $\mathcal{A}$ un automate déterministe complet, acceptant un langage L. Alors l'automate des résiduels $\mathcal{A}_L$ est égal à $\mathcal{A}/\sim$.

application 2.15 Algorithme de Hopcroft

entrée : $\mathcal{A} = (\Sigma, Q, \delta, I, F)$
$P \leftarrow (F, Q \setminus F)$
$S \leftarrow \{(\min(F, Q\setminus F), a) \mid a \in \Sigma\}$
Tant que $S \neq \emptyset$ faire.
  $(C,a) \leftarrow$ élément de S au hasard
  $S \leftarrow S \setminus \{(C,a)\}$
  Pour chaque B coupé par (C,a) en B_1, B_2 faire
    remplacer B par B_1, B_2 dans P
    Pour tout $b \in \Sigma$ faire
      Si $(B,b) \in S$ alors
        remplacer (B,b) par (B_1,b) et (B_2,b) dans S
      Sinon
        ajouter $(\min(B_1,B_2), b)$ à S

* Cet algorithme termine et calcule la congruence de Nérode de $\mathcal{A}$ $\boxed{\text{DVPT 1}}$

3) Lemmes de l'étoile

thm 2.16 lemmes de l'étoile. Soit L un langage rationnel, alors

(1) $\exists N \in \mathbb{N}$, $\forall u \in L$ tq $|u| \geq N$, $\exists\, vtw = u$, $t \neq \varepsilon$ tq $\forall n \in \mathbb{N}$, $vt^n w \in L$

(2) $\exists N \in \mathbb{N}$, $\forall u \in L$ tq $u = v_1 u_1 w_1$ tq $|u_1| \geq N$, $\exists\, vtw = u_1$, $t \neq \varepsilon$ tq $\forall n \in \mathbb{N}$, $v_1 v t^* w w_1 \in L$

(3) $\exists N \in \mathbb{N}$, $\forall u \in L$, $u = v u_1 \cdots u_N w$ tq $\forall i, u_i \neq \varepsilon$, $\exists\, 0 \leq j < k \leq N$ tq $v u_1 \cdots u_j (u_{j+1} \cdots u_k)^* u_{k+1} \cdots u_N w \in L$

rq 2.17 Ces lemmes permettent de montrer que certains langages ne sont pas rationnels.

ex 2.18
- $L = \{a^n b^n \mid n \in \mathbb{N}\}$ ne vérifie pas (1)
- $L = \{a^n b^n \mid n \in \mathbb{N}\} \cup \Sigma^* ba \Sigma^*$ vérifie (1) mais pas (2)

Rq 2.19 [ADT IS] une réciproque au lemme de l'étoile. Soit L un langage
L et $\Sigma^* \setminus L$ sont rationnels ssi ils vérifient tous deux le lemme de l'étoile par blocs (3). [Ehrenfeucht, Parikh et Rozenberg]

III / Applications des langages rationnels

1) Problèmes de décision.

prop 3.1 Soit $L, L' \in Rat(\Sigma)$ et $u \in \Sigma^*$. Les problèmes suivants sont décidables :

- $L = \emptyset$
- $u \in L$
- L est infini
- $L = L'$

rq 3.2 Ce dernier problème une fois reformulé montre que le problème "deux expressions rationnelles dénotent-elles le même langage" est aussi décidable

def 3.3 Le problème de séparation par automate (PSA) consiste à chercher l'existence, pour des langages S et T et un entier k donné, d'un automate fini déterministe à k états A tel que $S \subset L(A)$ et $T \cap L(A) = \emptyset$

Th 3.4 PSA est NP-complet DEV 2

2) Recherche de motifs

On cherche un motif u dans un texte t. On cherche donc à reconnaître le langage $\Sigma^* u \Sigma^*$ (qui est rationnel)

def 3.5 Automate de recherche de u.

On note $Pref(u)$ l'ensemble des préfixes de u.

On note R_u l'automate $(\Sigma, Pref(u), \delta, \{\varepsilon\}, \{u\})$

tel que $\delta(p, a)$ = le plus long suffixe de pa dans $Pref(u)$.

prop 3.6 R_u reconnaît $\Sigma^* u$ et est l'automate minimal pour ce langage

ex 3.7 Recherche de $abaab$. Cf annexe D

3) Analyse lexicale

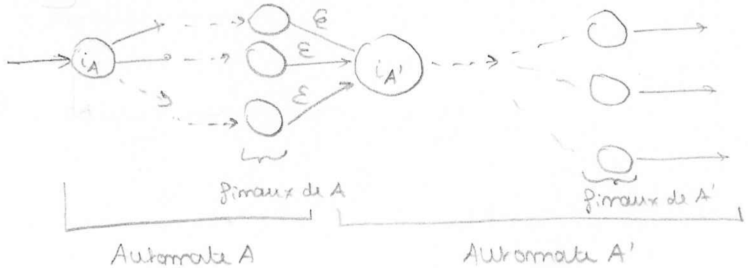
Quand on l'aura fait !

ANNEXE A

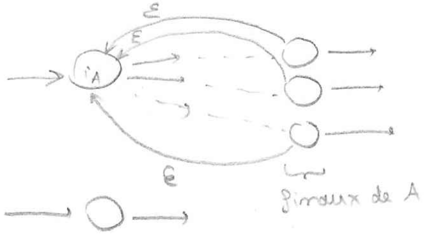
reconnaît $(a+b)^*ab$. Les états initiaux sont figurés par une flèche entrante et les finaux par une flèche sortante

ANNEXE B

A reconnaît L et A' reconnaît L'. On peut supposer que A et A' n'ont qu'un seul état initial.

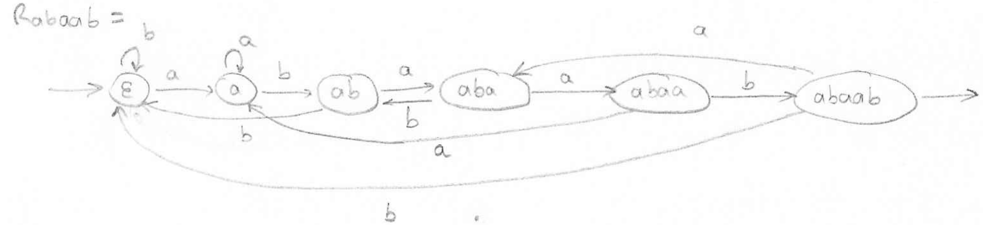
- Juxtaposer A et A' donne 1 automate qui reconnaît L+L'
- 

reconnaît LL'

- 

reconnaît L^*

ANNEXE D

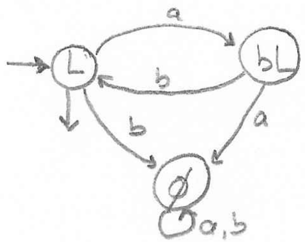


ANNEXE C

$L = (ab)^*$

Résiduels : L ; $a^{-1}L = b(ab)^* = bL$; $b^{-1}L = \emptyset$

D'où



reconnaît $(ab)^*$