909 LANGAGES RATIONNELS - EXEMPLES ET APPLICATIONS -

On supposera connues les définitions de : - automate fini déterministe (complet)
- automate fini non déterministe
- automate fini non déterministe avec ε-transitions

et l'équivalence entre tous ces automates.
Par la suite, Σ désignera un alphabet fini.

I Langages rationnels et leurs caractérisations

La classe des langages rationnels est le premier niveau de la hiérarchie de Chomsky et peut être définie de beaucoup de manières différentes. En voici qqes unes :

1) Une première définition [Car] p33

Définition [Langages rationnels] La classe $Rat(\Sigma^*)$ des langages rationnels (sur Σ) est la plus petite famille de langages telle que :
* $\emptyset \in Rat(\Sigma^*)$
* $\forall a \in \Sigma \quad \{a\} \in Rat(\Sigma^*)$
* $Rat(\Sigma^*)$ est close par union (finie), par produit (i.e concaténation) et par l'étoile.

Exemples : • $\Sigma \in Rat(\Sigma^*)$ car il s'écrit $\Sigma = \bigcup_{a \in \Sigma} \{a\}$

• Les mots de longueur paire forment un langage rationnel car égal à $(\Sigma\Sigma)^*$.

• Le langage des mots contenant le facteur abba est rationnel car égal à $\Sigma^*\{abba\}\Sigma^*$

2) Expressions rationnelles. [Sak] p133

Définition [Expressions rationnelles] : Une expression rationnelle sur Σ est une formule obtenue inductivement de la manière suivante (on suppose que Σ ne contient pas les lettres '$\emptyset$', 'ε', '+', '.', '*', '(', ')')

• $\emptyset$, ε et a, pour tout a dans Σ sont des expressions rationnelles.

• Si E et F sont deux expressions rationnelles alors $(E+F)$, $(E.F)$ et $(E)^*$ sont des expressions rationnelles.

(Pour une meilleure lisibilité on enlèvera les parenthèses qui n'impliquent pas d'ambiguïté au niveau des expressions en imposant des priorités entre les opérateurs)

Ex : - $(a+b)^* abba (a+b)^*$

Définition À chaque expression rationnelle E on fait correspondre un langage noté $L(E)$ tel que :
* $L(\emptyset) = \emptyset$ $\quad L(\varepsilon) = \{\text{mot vide}\}$ $\quad L(a) = \{a\}$ pour $a \in \Sigma$
* $L(E+F) = L(E) \cup L(F)$ $\quad L(E.F) = L(E).L(F)$ et $L(E^*) = (L(E))^*$.

Proposition Un langage est rationnel ssi il est dénoté par une expression rationnelle.

Il faut distinguer expression = objet syntaxique et le langage qu'elle dénote = sémantique

Ex : $(a+b)^*$ et $(a^*b)^*a^*$ sont des expressions différentes mais les langages qu'elles dénotent sont les mêmes.

Définition Deux expressions sont dites équivalentes si elles dénotent le même langage.

Problème : Comment déterminer si deux expressions données sont équivalentes ?

3) Théorème de Kleene

Théorème de Kleene : Un langage L est rationnel ssi il existe un automate (fini) $\mathcal{A}$ tel que $L = L(\mathcal{A})$.

Des expressions aux automates : [Sak] p155

La transition peut se faire de manière standard comme suit :

Union : $(L(\mathcal{A}) \cup L(\mathcal{B}))$

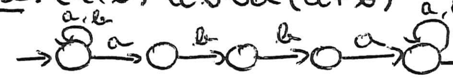
Produit : $(L(\mathcal{A}).L(\mathcal{B}))$

ε-transitions des états terminaux de $\mathcal{A}$ vers l'état initial de $\mathcal{B}$

Étoile : $(L(\mathcal{A})^*)$

ε-transitions des états terminaux vers l'état initial + création d'un nouvel état initial

Ex : $(a+b)^* abba (a+b)^*$



(après élimination des ε-transitions)

À noter : On peut directement obtenir un automate déterministe grâce à la méthode de Brzozowski (automate des dérivées)

Des automates aux expressions [Car] p38

Algorithme de McNaughton-Yamada : On étiquette Q, ensemble des états d'un automate $\mathcal{A}$, par $1, 2, \dots, n$.
Pour $(q,q') \in Q^2$ on définit alors
$$L^{(k)}_{q,q'} = \{a_1 \dots a_n \mid q \xrightarrow{a_1} p_1 \xrightarrow{a_2} \dots \xrightarrow{a_n} q' \text{ et } p_1, \dots, p_{n-1} \in \{1, \dots, k\}\}$$
On a les formules :
$$L^{0}_{q,q'} = \{a \mid q \xrightarrow{a} q'\} \cup \{\varepsilon \mid \text{si } q = q'\}$$
$$L^{(k+1)}_{q,q'} = L^{(k)}_{q,q'} + L^{(k)}_{q,k+1}\left(L^{(k)}_{k+1,k+1}\right)^* L^{(k)}_{k+1,q'}$$
Et d'autre part : $\boxed{L(\mathcal{A}) = \bigcup_{k \in F} L^{(n)}_{i,k}}$ (i : état initial ; F : états finals)

Élimination de Gauss :

Lemme d'Arden : Soit l'équation $X = AX + B$ où l'inconnue X est un langage.

1. Si $\varepsilon \notin A$, $\quad X = A^*B$.

2, Si $\varepsilon \in A$ X est de la forme $X = A^*(B+P)$ où $P \subset \Sigma^*$.

Principe de la méthode: On appelle pour $p \in Q$ $L_p = \{w \mid \exists f \in F \; p \xrightarrow{w} f\}$

On a donc $L(A) = L_i$, où i est l'état initial.

On résout alors le système

$$\left\{ L_p = \sum_{p \xrightarrow{a} q} a L_q \; \underset{\text{si } p \text{ est un état final}}{+ \varepsilon} \right.\quad p \in Q$$

en réduisant par substitutions successives les équations grâce au lemme d'Arden.

4) Autres caractérisations

a) Automate boustrophédon [Sak p185]

Un automate boustrophédon est une machine de Turing dont la tête qui se déplace sur la bande est seulement une tête de lecture (et pas d'écriture).

Théorème: Un langage est rationnel ssi il est reconnu par un automate boustrophédon.

(La preuve requiert la finitude des quotients à gauche pour un langage régulier)

b) Grammaires linéaires [Wiki p ???]

Une grammaire est dite *linéaire* à droite (resp à gauche) ssi ces productions sont de la forme

$A \to wB$ ou $A \to w$ (resp $A \to Bw$ ou $A \to w$) avec A, B non terminaux, w une chaîne de terminaux

Proposition Un langage est rationnel ssi il est engendré par une grammaire linéaire (à gauche ou à droite)

Ex: $(a+b)^*abba(a+b)^*$ est engendré par

$S \to aS \mid bS \mid abbaT$

$T \to aT \mid bT \mid \varepsilon$.

(l'axiome est S)

II Conséquences du théorème de Kleene

1) Propriétés de clôture [Reg-Sal p83]

Propriété: La classe des langages rationnels est stable par:
- union (finie) — def?
- intersection (finie)
- complémentation
- concaténation (finie)
- étoile
- image miroir

Théorème: L'image d'un langage rationnel par un transducteur fini* reste un langage rationnel.

Corollaire: Pour L un langage rationnel, les ensembles suivants sont rationnels:
- préfixes$(L) = \{x \mid \exists y \in \Sigma^* \; xy \in L\}$
- suffixes$(L) = \{y \mid \exists x \in \Sigma^* \; xy \in L\}$
- infixes$(L) = \{y \mid \exists x, z \in \Sigma^* \; xyz \in L\}$

Corollaire 2: L'image d'un langage rationnel par un morphisme (i.e une application h telle que $\forall x, y \in \Sigma^* \; h(x.y) = h(x).h(y)$) est un langage rationnel.

Propriété: Pour L langage rationnel et K un langage quelconque, le quotient à gauche de L par K i.e $K^{-1}L = \{y \mid \exists x \in K \; xy \in L\}$ et le quotient à droite de L par K i.e $LK^{-1} = \{x \mid \exists y \in K \; xy \in L\}$ sont des langages rationnels.

2) Lemme(s) de l'étoile et variantes [Sak p...]

Définition [Facteur itérant] Un mot v est dit *facteur itérant* d'un langage L lorsqu'il existe u et w dans Σ^* tels que $u v^* w \subset L$

Lemme de l'étoile 1: Dans un langage rationnel, tout mot suffisamment long contient un facteur itérant non vide.

Application: $\{a^n b^n \mid n \in \mathbb{N}\}$ n'est pas rationnel.

Lemme de l'étoile 2: Dans un langage rationnel L, tout facteur suffisamment long d'un mot de L contient un facteur itérant non vide i.e il existe un entier N tel que pour tout f dans L et toute factorisation $f = g_1 h g_2$ telle que $|h| \geq N$, il existe une factorisation de $h = uvw$, avec $v \neq \varepsilon$ telle que $g_1 u v^* w g_2 \subset L$.

Application: $\{a^n b^n \mid n \in \mathbb{N}\} \cup \Sigma^* ba \Sigma^*$ n'est pas rationnel.

Lemme de l'étoile par bloc: Si L est rationnel, il existe $N \geq 0$ tel que pour tout $f \in L$ et tout factorisation de la forme $f = u v_1 v_2 \dots v_N w$, où $v_i \neq \varepsilon$ il existe (j, k), $0 \leq j < k \leq N$, tel que: $u v_1 v_2 \dots v_j (v_{j+1} \dots v_k)^* v_{k+1} \dots v_N w \subset L$

Application: $\{(aab)^n (abb)^n \mid n \in \mathbb{N}\} \cup \Sigma^* \{aaa, bbb\} \Sigma^*$ n'est pas rationnel.

Théorème: [Reg-Sal p86] Un langage L est régulier ssi il existe $N \geq 0$ tel que pour tout $w \in \Sigma^*$ avec $|w| \geq N$ il existe une factorisation $w = xyz$, $y \neq \varepsilon$ tel que:

$\forall i \in \mathbb{N} \; \forall v \in \Sigma^* \; (wv \in L \iff x y^i z v \in L)$

Théorème: L'ensemble des facteurs itérants d'un langage rationnel forme un langage rationnel. **DVP**

3) Quotients [Bq p312]

Définition [Quotient gauche, quotient droit] Les quotients gauche et droit d'un langage L par un mot w sont respectivement: $w^{-1}L = \{v \mid wv \in L\}$ et $Lw^{-1} = \{u \mid uw \in L\}$

Notons que $\varepsilon^{-1}L = L$ et $(uv)^{-1}L = v^{-1}(u^{-1}L)$

Proposition Soit L un langage reconnu par un automate $\mathcal{A}$, on note pour $q \in Q(\mathcal{A})$ $L_q = \{w \in \Sigma^* \mid q \xrightarrow{w} \text{état final de } \mathcal{A}\}$

Alors les L_q décrivent l'ensemble des quotients gauches de L.

*: notion supposée connue

Théorème : Un langage est rationnel ssi il admet un nombre fini de quotients gauches.

Ex : $L = (a+b)^* ab (a+b)^*$: $a^{-1}L = L + b(a+b)^*$; $b^{-1}L = L$; $(ab)^{-1}L = (a+b)^*$; $(aa)^{-1}L = L$; $(aba)^{-1}L = (abb)^{-1}L = (a+b)^*$

III Minimisation des automates

correcte ??

Motivation(s) de cette partie :

* Obtenir un procédé efficace (principalement au niveau de l'occupation mémoire) pour déterminer si un mot donné appartient à un langage rationnel fixé ou pas. (aspect pratique)
* Pouvoir tester l'équivalence de deux expressions rationnelles (aspect théorique)

1) Automate minimal et équivalence de Nerode [Bq p315]

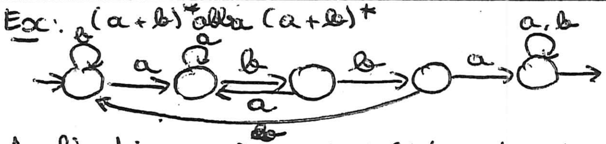
Définition [Équivalence de Nerode]
Pour un automate $A = (Q, \Sigma, \delta, i, F)$ on définit une relation d'équivalence sur Q appelée équivalence de Nerode définie comme

$p \sim q$ ssi $L_p = L_q$
ssi $\forall u \in \Sigma^*$ $\delta^*(p,u) \in F \Leftrightarrow \delta^*(q,u) \in F$

On notera $A/\sim \; = (Q/\sim, \Sigma, \bar\delta, \bar i, \bar F)$ l'automate A quotienté par $\sim$
($Q/\sim$ est l'ensemble des classes de la relation $\sim$ et $\bar\delta : (\bar q, a) \mapsto \overline{\delta(q,a)}$ est bien définie car $\sim$ est régulière à droite)

Théorème : L'automate minimal (de L) est l'automate ayant le moins d'états parmi les automates déterministes complets qui reconnaissent L. Il est unique à isomorphisme près. Il est égal à $A/\sim$, mais aussi à l'automate des quotients.

Ex : $(a+b)^* abba (a+b)^*$



Application : On peut décider si oui ou non deux expressions données sont équivalentes.

2) Calcul de l'automate minimal [Car p85]

Méthode de Moore : (On fixe $A = (Q, \Sigma, \delta, i, F)$ dét.)

Proposition : L'équivalence de Nerode est la relation d'équiv. sur Q régulière à droite (i.e $p \sim q \Rightarrow \delta(p,a) \sim \delta(q,a)$) la plus grossière qui est plus fine que $\{F, \bar F\}$.

On définit alors une suite $(\sim_i)$ telle que
$q \sim_0 q' \Leftrightarrow (q \in F \Leftrightarrow q' \in F)$
$q \sim_{i+1} q' \Leftrightarrow (q \sim_i q'$ et $\forall a \in \Sigma$ $\delta(q,a) \sim_i \delta(q',a))$
Si $\sim_k = \sim_{k+1}$, alors $\sim_k$ est l'équivalence de Nerode (l'algorithme est en $O(mn^2)$) $m = card(\Sigma)$, $n = card(Q)$

Algorithme de Hopcroft :
Définition [Stabilité] Soient $P \subset Q$, $R \subset Q$ et $a \in \Sigma$
P est dit stable pour (R,a) ssi $P \subset a^{-1}R$ ou $P \subset \overline{a^{-1}R}$.
Sinon P est dit brisé pour (R,a) i.e $P \cap a^{-1}R \neq \emptyset$ et $P \cap \overline{a^{-1}R} \neq \emptyset$
Une partition $\mathcal{P}$ de Q est stable pour (R,a) ssi $\forall P \in \mathcal{P}$ P est stable pour (R,a)
$\mathcal{P}$ est dite stable (tout court) ssi $\forall R \in \mathcal{P}$ $\mathcal{P}$ est stable pour (R,a).

Proposition Une relation d'équiv sur Q est régulière à droite ssi la partition associée est stable.

Théorème : L'algorithme dit de Hopcroft minimise l'automate A en une complexité $O(m n \log n)$ où $m = card(\Sigma)$ et $n = card(Q)$

DVP (l'invariant sera explicité mais non prouvé)

IV Applications

1) En biologie, en mathématiques [Roz-Sal p41]

• À l'origine introduit pour modéliser les neurones
• Peut être utilisé pour résoudre des problèmes de combinatoire... Exemples à l'oral

2) Reconnaissance de motifs [Sak p63] [AR ...]

On veut cherche un mot u dans un texte.
Concrètement on veut reconnaître le langage $\Sigma^* u \Sigma^*$ (ou $\Sigma^* u$)
(exemple repris durant toute cette leçon !) avec $u = abba$

On note $Pref(u)$ = ensemble des préfixes de u

$B_u = (Pref(u), \Sigma, \delta, \{\varepsilon\}, \{u\})$
avec $\delta(p,a)$ = le plus long préfixe de pa dans $Pref(u)$

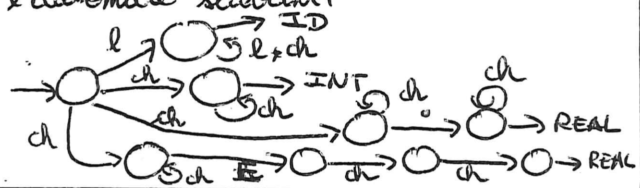
Théorème : B_u reconnaît $\Sigma^* u$.
De plus, il s'agit de l'automate minimal.

3) Analyse lexicale [WM p208]

C'est la première étape d'un compilateur. Elle précède l'analyse syntaxique.
Elle est chargée de lire un programme source (sous forme d'une suite de caractères) afin de le décomposer en une succession d'unités lexicales, appelées lexèmes.

Ex de lexèmes : Entier, Identificateur, mots réservés au langage (comme begin, if, int)

Par ex on veut reconnaître le langage $\underbrace{l(ch+l)^*}_{ID} + \underbrace{ch\,ch^*}_{INT} + \underbrace{ch\,ch^* . ch^*}_{REAL} + \underbrace{ch\,ch^* E\,ch\,ch}_{REAL}$
où $ch = \{1, 2, \ldots, 9\}$ et $l = \{a, b, \ldots, z\}$
On peut alors déterminiser & minimiser l'automate suivant.



ANNEXES

À propos des références

[Car] Carton : Un des plus clairs sur le sujet à mon avis

[Sak] Sakarovitch : Complet, bien expliqué, j'aime beaucoup, il a juste un humour assez bizarre

[Roz-Sal] Rozenberg - Salomaa (Handbook of Formal Languages) Livre super théorique, en anglais, mais pas mal expliqué.

[Bq] Beauquier (Éléments d'Algorithmique) Si, si, on parle des automates dans le Beauquier, j'aime bcp la partie sur la minimisation, c'est un des ouvrages le plus clair sur le sujet

Mais aussi :

[Wolp] Wolper : Apparemment intéressant, (d'après Loulou) mais je n'en ai pas eu besoin.

[WM] Wilhelm Maurer (Les compilateurs) No comment.

J'avais emprunté le Autebert mais je n'avais pas du tout aimé

À propos du plan

: la partie apparemment contestable serait celle sur la minimisation des automates. Moi je suis convaincu que ça a sa place ici, à vous de faire votre choix (et votre avis).

- Je déconseillerais de parler de dérivations des expressions rationnelles / hauteurs d'étoile / langages sans étoile. Bcp trop théorique avec peu d'application pratique, pas sûr que ça plaise au jury. (quoi que la dérivation reste à discuter...)
- Bien sûr tout n'est pas indispensable dans ce que j'ai mis. J'ai été même parfois (un peu) redondant. Il aurait également fallu que je précise un peu mieux les notations utilisées

À propos des développements

les développements que je propose, à savoir la régularité des facteurs itérants et l'algorithme de Hopcroft sont plutôt longs et pas du genre faciles je pense. Le second a le gros avantage de pouvoir se recaler dans 3 leçons (dont celle sur Diviser pour régner) Un investissement rentable ? (Car il y a du [illegible]).

J'avais pensé aussi à :

- Th de Kleene
- Localisation / Automates des préfixes (dvp d'Aurélia quoi)
- Équivalence avec les grammaires linéaires
- Méthode de Brzozowski - complexité en terme d'espace de l'équivalence de 2 expr rationnelles ?