

Langages rationnels

G. Guillo

es langages.

Théorème de

Expressions égulières sous

_angages no

Chapitre IV: Langages rationnels

Goulven GUILLOU

UBO - Département Informatique



1/24

UI36

Quels sont les langages reconnaissables?

Langages rationnels

o. oum

troduction

es langages ationnels

Théorème de Kleene

> Expressions régulières sou JNIX

Langages n

On sait que l'ensemble des langages reconnaissables est fermé par union (+), concaténation (.) et fermeture itérative (*) et contient les langages finis.

On note Rec(A) ou $Rec(A^*)$ l'ensemble des langages reconnaissables sur l'alphabet A.

Donc les langages construits à partir d'un alphabet et des opérateurs d'union, de concaténation et de fermeture itérative sont reconnaissables.

Y en a-t-il d'autres?

2 / 24

OEiU

Les langages rationnels

Langages rationnels

G. Guillou

Les langages rationnels

Théorème d

Expressions régulières sous

Langages no rationnels

Soit A un alphabet, la classe des langages rationnels ou réguliers sur l'alphabet A, notée $Rat(A^*)$ ou Rat est le plus petit ensemble satisfaisant les conditions suivantes :

- \blacksquare $\emptyset \in Rat.$
- $\mathbf{x} \in Rat, \ \forall x \in A.$
- Soient $L \in Rat$ et $L' \in Rat$ alors $L + L' \in Rat$, $L.L' \in Rat$ et $L^* \in Rat$.

OEIU

Expressions régulières

Langages rationnels

G. Guille

Les langages

Théorème de Kleene

Expressions régulières sou UNIX

Langages no

L'expression d'un langage selon une décomposition en unions, produits et étoiles s'appelle une expression régulière du langage. Une expression régulière est construite à partir de caractères, des opérateurs binaires de concaténation et d'union, de l'opérateur unaire * et de parenthèses.

Par exemple : (a+b)*abb(a+b)*

Un langage est *rationnel* ou *régulier* ssi il est dénoté par une expression régulière.

Voir le chapitre 3 pour les priorités permettant de supprimer des parenthèses.

3/24 4/24

OEIU

Exemples de langages réguliers

Langages rationnels

Les langages

héorème de

Expressions

Langages no

- sur un alphabet A, le langage L constitué des mots de longueur paire est régulier car il est décrit par l'expression $L = (AA)^*$.
- sur $A = \{0, 1\}$, le langage L des mots commençant par 1 est décrit par l'expression régulière $L = 1(1+0)^*$. De même $L' = 0^*10^*10^*10^*$ est régulier et constitué des mots contenant exactement trois fois la lettre 1.
- sur $A = \{a, b, c, ..., z\}$, le langage des mots contenant comme facteur facteur ou factrice est dénoté par A^* fact(eur + rice) A^* et est donc régulier.
- lacksquare sur $A=\{a,b\}$, $L=\{a^nb^n|n\in\mathbb{N}\}$ n'est pas régulier.

UI30

Théorème de Kleene

Langages rationnels

G. Guille

Les langage

Théorème o

Expressions régulières so UNIX

Langages n

Un langage est régulier si et seulement si il est accepté par un automate fini. Autrement dit :

$$Rat(A^*) = Rec(A^*)$$

5 / 24

7 / 24

6 / 24

8 / 24

UBO

Expression régulière vers automate (Thompson)

Langages rationnels

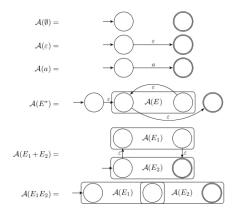
Introduction

Les langage

Théorème

Expressions régulières sou UNIX

Langages no rationnels Les automates générés ont un unique état initial sans transition entrante et un unique état accepteur sans transition sortante. On procède par induction sur l'expression.



OSIU

Langages

G. Guillo

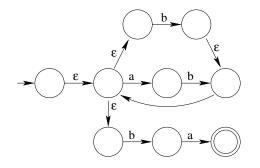
itroduction es langages

Théorème (Kleene

régulières sous UNIX

_angages non rationnels

Exemple de construction de Thompson : (ab+b)*ba

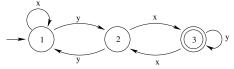


Automate vers expression régulière (McNaughton & Yamada)

Langages rationnels

Kleene

 $W_{p,E,q}$ = ensemble des mots permettant de passer de p à q en n'utilisant que les états intermédiaires de $E \subseteq Q$.



$$\begin{array}{lll} W_{1,\{1,2,3\},3} & = & W_{1,\{2,3\},3} \cup W_{1,\{2,3\},1}.W_{1,\{2,3\},1}^*.W_{1,\{2,3\},3}\\ & = & W_{1,\{2,3\},1}^*.W_{1,\{2,3\},3}\\ & = & W_{1,\{2,3\},1}^*.W_{1,\{2\},3}.W_{3,\{2\},3}^*\\ W_{1,\{2\},3} & = & \{yx\}\\ W_{3,\{2\},3} & = & W_{3,\emptyset,3} \cup W_{3,\emptyset,2}W_{2,\emptyset,2}^*.W_{2,\emptyset,3} & = \{\varepsilon,y,xx\}\\ W_{1,\{2,3\},1} & = & W_{1,\{2\},1} \cup W_{1,\{2\},3}.W_{3,\{2\},3}^*.W_{3,\{2\},1}\\ W_{1,\{2\},1} & = & \{\varepsilon,x,yy\}\\ W_{3,\{2\},1} & = & \{xy\}\\ \mathbf{d'où} \ L & = & (x+yy+yx(y+xx)^*xy)^*yx(y+xx)^* \end{array}$$

9 / 24

Méthode pratique : lemme d'Arden

Proposition 1 : Soient U et V deux langages rationnels tels que $\varepsilon \notin U$ alors l'unique langage solution de L = LU + V est VU^* .

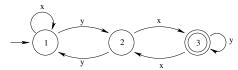
On a la proposition symétrique :

Proposition 2 : Soient U et V deux langages rationnels tels que $\varepsilon \notin U$ alors l'unique langage solution de L = UL + V est U^*V .

10 / 24

Méthode 1

On note L_i le langage de l'automate ayant le même état initial que l'automate de départ avec i pour unique état accepteur.



$$\left\{ \begin{array}{l} L_1 = L_1x + L_2y + \varepsilon \text{ (car 1 \'etat initial)} \\ L_2 = L_1y + L_3x \\ L_3 = L_2x + L_3y \text{ (langage que l'on cherche)} \end{array} \right.$$

Attention dans le cas où l'automate a plusieurs états accepteurs, faire l'union des L_i correspondants!

Méthode 1 : résolution

Langages rationnels

Elimination à partir de L_3 jusqu'à L_1 .

$$\begin{cases} L_{1} = L_{1}x + L_{2}y + \varepsilon \\ L_{2} = L_{1}y + L_{3}x \\ L_{3} = L_{2}x + L_{3}y \end{cases} \begin{cases} L_{1} = L_{1}x + L_{2}y + \varepsilon \\ L_{2} = L_{1}y + L_{2}xy^{*}x \\ L_{3} = L_{2}xy^{*} \end{cases}$$
$$\begin{cases} L_{1} = L_{1}x + L_{1}y(xy^{*}x)^{*}y + \varepsilon \\ L_{2} = L_{1}y(xy^{*}x)^{*} \end{cases} \begin{cases} L_{1} = (x + y(xy^{*}x)^{*}y)^{*} \\ L_{2} = L_{1}y(xy^{*}x)^{*} \\ L_{3} = L_{2}xy^{*} \end{cases}$$

D'où
$$L_2 = (x + y(xy^*x)^*y)^*y(xy^*x)^*$$
 et

$$L_3 = (x + y(xy^*x)^*y)^*y(xy^*x)^*xy^*$$

11 / 24

Méthode 1 : autre résolution

Langages rationnels

Elimination de L_1 vers L_3 (ce que l'on cherche).

$$\begin{cases}
L_1 = L_1 x + L_2 y + \varepsilon \\
L_2 = L_1 y + L_3 x \\
L_3 = L_2 x + L_3 y
\end{cases}
\begin{cases}
L_1 = (L_2 y + \varepsilon) x^* \\
L_2 = (L_2 y + \varepsilon) x^* y + L_3 x \\
L_3 = L_2 x y^*
\end{cases}$$

$$\begin{cases} L_1 = (L_2y + \varepsilon)x^* \\ L_2 = (x^*y + L_3x)(yx^*y)^* \\ L_3 = (x^*y + L_3x)(yx^*y)^*xy^* \end{cases}$$

$$L_3 = x^* y (yx^* y)^* xy^* (x(yx^* y)^* xy^*)^*$$

13 / 24

$$\begin{cases} L_{1} = L_{1}x + L_{2}y + \varepsilon \\ L_{2} = L_{1}y + L_{3}x \\ L_{3} = L_{2}x + L_{3}y \end{cases} \qquad \begin{cases} L_{1} = (L_{2}y + \varepsilon)x^{*} \\ L_{2} = (L_{2}y + \varepsilon)x^{*}y + L_{3}x \\ L_{3} = L_{2}xy^{*} \end{cases}$$

$$\begin{cases} L_1 = (L_2y + \varepsilon)x^* \\ L_2 = (x^*y + L_3x)(yx^*y)^* \\ L_3 = (x^*y + L_3x)(yx^*y)^*xy^* \end{cases}$$

D'où

Propriétés de clôture de $Rat(X^*)$

 $Rat(X^*)$ est fermé par union, intersection, complémentaire et fermeture itérative.

Méthode 2

initial.

Langages rationnels

 $\left\{ \begin{array}{l} L_1=xL_1+yL_2 \text{ (langage que l'on cherche)} \\ L_2=yL_1+xL_3 \\ L_3=xL_2+yL_3+\varepsilon \text{ (car 3 \'etat accepteur)} \end{array} \right.$

$$L_2 = yL_1 + xL_3$$

$$L_3 = xL_2 + yL_3 + \varepsilon$$
 (car 3 état accepteur

On résout par élimination de L_3 jusqu'à L_1 :

$$\begin{cases}
L_1 = xL_1 + yL_2 \\
L_2 = yL_1 + xy^*xL_2 + xy^* \\
L_3 = y^*(xL_2 + \varepsilon)
\end{cases}
\begin{cases}
L_1 = (x + y(xy^*x)^*y)L_1 \\
+ y(xy^*x)^*xy^* \\
L_2 = (xy^*x)^*(yL_1 + xy^*) \\
L_3 = y^*(xL_2 + \varepsilon)
\end{cases}$$

On note L_i le langage de l'automate de départ ayant i pour état

D'où $L_1 = (x + y(xy^*x)^*y)^*y(xy^*x)^*xy^*$

Le même résultat que la première résolution de la méthode 1!

14 / 24

Extensions UNIX pour les expressions régulières

Sous UNIX on utilise des expressions régulières dans au moins 3 types de commandes :

- les éditeurs UNIX comme vi ou emacs.
- le programme d'équivalence de motifs grep et ses cousins.
- l'analyse lexicale avec la commande UNIX flex.

Les classes de caractères

Langages rationnels

rationne

Introduction

es langages

Théorème de

Expressions régulières sous UNIX

_angages no

[aotu] désigne l'ensemble des lettres apparaissant dans auto.

[aotu] * désigne l'ensemble des mots composés des lettres a, o, t, u uniquement.

[A-Za-z] indique l'ensemble des lettres de l'alphabet en minuscules ou majuscules.

ATTENTION [-+*/] est différent de [+-*/]!

[^aotu] désigne n'importe quel caractère excepté a, o, t ou u.

17 / 24

OEIU

Début et fin de ligne

Langage: rationnel

0. 00.

troduction

Les langage

Théorème o Kleene

Expressions régulières sous UNIX

Langages n

Le symbole ^ indique le début d'une ligne, et \$ indique la fin d'une ligne.

Utilisation : grep '^[aotu]*\\$' monfichier affichera tous les mots de monfichier qui tiennent sur une ligne et composés uniquement des lettres appartenant au mot auto.

18 / 24

UB6

Le symbole joker

Langages

Cuille

Introduction

Théorème d

Expressions régulières sous HNIX

Langages nor rationnels

Le caractère . remplace un caractère quelconque hormis le caractère de passage à la ligne.

L'expression régulière : .*a.*e.*i.*o.*u.

décrit toutes les chaînes contenant les voyelles dans l'ordre.

Avec grep il suffit de faire : grep a.*e.*i.*o.*u

OEIU

Opérateurs ? et + et références arrières

Langages rationnels

G. Guillou

. . .

ationnels

Kleene

régulières so UNIX

> _angages non ationnels

R? signifie "au plus une fois R".

R+ équivaut à RR*

[-]?[0-9]+ désigne un entier signé.

On peut faire référence à toute partie d'une expression régulière encadrée par $\ (et\)$ par $n \ étant l'ordre d'apparition.$

grep '^\(.\).*\1\$' monfichier.txt va rechercher toutes les lignes qui commencent et finissent par le même caractère.

19/24 20/24

OSIU

Cardinalité et dénombrabilité

Langages rationnels

G. Guillo

es langages

Théorème de

expressions égulières sous

Langages no rationnels Deux ensembles ont même *cardinal* ou *cardinalité* s'il existe une bijection entre les deux.

"Avoir la même cardinalité" est une relation d'équivalence. On appelle n la cardinalité de $\{0, 1, ..., n-1\}$.

 $\begin{tabular}{ll} \textbf{D\'efinition:} Un ensemble est d\'enombrable s'il est fini ou s'il existe une bijection entre lui et N. \\ \end{tabular}$

21 / 24

23 / 24

UBG

Diagonale de Cantor

Langages rationnels

0. 00....

Les langages

Théorème c

Expressions

Langages r

Théorème: L'ensemble des sous-ensembles d'un ensemble infini dénombrable n'est pas dénombrable.

preuve:

	a 0	a_1	a ₂	a 3	a 4	
<i>S</i> ₀	×	×		×		
s_1	×	Δ	×	×		
<i>s</i> ₂		×	×		×	
s 3	×		×	Δ		
<i>S</i> ₄		×		×	Δ	

22 / 24

24 / 24

Peu de langages sont réguliers

Langages

G. Guillo

ntroductic

rationnels

Théorème de Kleene

Langages no

Les expressions régulières sont dénombrables.

Les langages réguliers sont donc dénombrables.

Les langages ne sont pas dénombrables car c'est l'ensemble des parties d'un ensemble dénombrable.

Conclusion : il y a plus de langages que de langages réguliers.

OEiU

Pumping théorème

Langage rationne

G. Guillot

Introduction

Les langage

héorème de (leene

régulières soi UNIX

angages non

Soit L un langage régulier infini. Alors il existe un entier m>0 tel que tout mot $w\in L$ de longueur supérieure ou égale à m se décompose en 3 parties : w=xyz avec $|y|\geq 1$ et $xy^kz\in L$ pour tout $k\in\mathbb{N}$.

Exemple : $\{a^nb^n, n \in \mathbb{N}\}$ n'est pas régulier.

Supposons $\{a^nb^n, n \in \mathbb{N}\}$ régulier. Alors il existe un entier m > 0 tel que tout mot de longueur au moins m soit décomposable. En particulier $w = a^mb^m$ est décomposable.

On a trois cas possibles :

- $y = a^{s}, s > 0$
- $-y = a^{s}b^{t}, s > 0 \text{ et } t > 0$
- $-y = b^t, t > 0$

Dans aucun de ces cas xy^2z n'appartient à $\{a^nb^n, n \in \mathbb{N}\}.$