Autômatos e computabilidade

Expressões regulares

Pedro A D Rezende
UnB – IE – CIC

A & C: Expressões Regulares

Expressões Regulares

A notação algébrica que denota

- Concatenação como “produto”
- União de conjuntos como “soma” e
- Fecho de Kleene como “série de potências”

serve para descrever linguagens formais.

Uma fórmula nesta notação é dita, por motivos que ficarão óbvios, *expressão regular*

A & C: Expressões Regulares

Definição recursiva de Expressão Regular

Dado Σ, são expressões regulares (ER) sobre Σ:

- $\emptyset$ (o conjunto vazio)

A & C: Expressões Regulares

Definição recursiva de Expressão Regular

Dado Σ, são expressões regulares (ER) sobre Σ:

- $\varnothing$ (o conjunto vazio)
- $\boldsymbol{\lambda}$ (ER que denota $\{\lambda\}$)

A & C: Expressões Regulares

Definição recursiva de Expressão Regular

Dado Σ, são expressões regulares (ER) sobre Σ:

- $\varnothing$ (o conjunto vazio)
- λ (ER que denota $\{\lambda\}$)
- a (ER que denota $\{a\}$, $\forall a \in \Sigma$)

A & C: Expressões Regulares

Definição recursiva de Expressão Regular

Dado Σ, são expressões regulares (ER) sobre Σ:

- $\varnothing$ (o conjunto vazio)
- λ (ER que denota $\{\lambda\}$)
- a (ER que denota $\{a\}$, $\forall a \in \Sigma$)
- Se x, y são ERs que denotam as linguagens X e Y respectivamente, também são ERs as expressões (x+y), (xy) e (x*) denotando, respectivamente, $X \cup Y$, $X \circ Y$ e X^*

OBS: x^+ abrevia xx^* ; x^n abrevia xx...x n vezes

A & C: Expressões Regulares

Notação simplificada

Usando a precedência das operações homônimas, omite-se “(“ “)” onde possível

A & C: Expressões Regulares

Notação simplificada

Usando a precedência das operações homônimas, omite-se “(“ “)” onde possível

Exemplos (fixando $\Sigma=\{0, 1\}$):

`01*+0` simplifica `((0(1*))+0)`

A & C: Expressões Regulares

Notação simplificada

Usando a precedência das operações homônimas, omite-se “(“ “)” onde possível

Exemplos (fixando $\Sigma=\{0, 1\}$):

`01`*`+0` simplifica `((0(1`*`))+0)`

`(0+1)`*`0` denota `L =` $\{\alpha 0 \mid \alpha \in \Sigma^*\}$

A & C: Expressões Regulares

Notação simplificada

Usando a precedência das operações homônimas, omite-se "(" ")" onde possível

Exemplos (fixando $\Sigma=\{0, 1\}$):

`01`*`+0` simplifica `((0(1`*`))+0)`

`(0+1)`*`0` denota L = $\{\alpha 0 \mid \alpha \in \Sigma^*\}$

`(10+1)`* denota L =

A & C: Expressões Regulares

Notação simplificada

Usando a precedência das operações homônimas, omite-se “(“ “)” onde possível

Exemplos (fixando $\Sigma=\{0, 1\}$):

`01*+0` simplifica `((0(1*))+0)`

`(0+1)*0` denota L = $\{\alpha 0 \mid \alpha \in \Sigma^*\}$

`(10+1)*` denota L =

$\{\alpha \in \Sigma^* \mid \alpha$ começa com 1 e não repete 0s$\}$

A & C: Expressões Regulares

Notação simplificada

Usando a precedência das operações homônimas, omite-se “(“ “)” onde possível

Exemplos (fixando $\Sigma=\{0, 1\}$):

`01`*`+0` simplifica `((0(1`*`))+0)`

`(0+1)`*`0` denota L = $\{\alpha 0 \mid \alpha \in \Sigma^*\}$

`(10+1)`* denota L =

$\{\alpha \in \Sigma^* \mid \alpha$ começa com 1 e não repete 0s$\}$

`(0+1)`* `00(0+1)`* denota L =

A & C: Expressões Regulares

Notação simplificada

Usando a precedência das operações homônimas, omite-se "(" ")" onde possível

Exemplos (fixando $\Sigma=\{0, 1\}$):

`01*+0` simplifica `((0(1*))+0)`

`(0+1)*0` denota L = $\{\alpha 0 \mid \alpha \in \Sigma^*\}$

`(10+1)*` denota L =

$\{\alpha \in \Sigma^* \mid \alpha$ começa com 1 e não repete 0s$\}$

`(0+1)*00(0+1)*` denota L =

$\{\alpha \in \Sigma^* \mid \alpha$ repete 0 pelo menos uma vez$\}$

A & C: Expressões Regulares

Expressões Regulares geram

Linguagens Regulares

Dado r ER, existe M λFA tal que $L(r) = L(M)$

A & C: Expressões Regulares

Expressões Regulares geram Linguagens Regulares

Dado r ER, existe M λFA tal que $L(r) = L(M)$

Demonstração: Por indução sobre $op(r) = |\text{operações em } r|$, usando grafos para λFAs

A & C: Expressões Regulares

Expressões Regulares geram

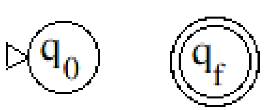
Linguagens Regulares

Dado r ER, existe M λFA tal que $L(r) = L(M)$

Demonstração: Por indução sobre $op(r) =$

$|$operações em $r|$, usando grafos para λFAs

P(0): (zero operações em r)



$r = \varnothing$

A & C: Expressões Regulares

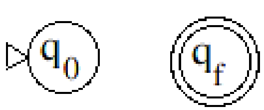
Expressões Regulares geram
Linguagens Regulares

Dado r ER, existe M λFA tal que L(r) = L(M)

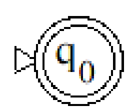
Demonstração: Por indução sobre op(r) =

|operações em r|, usando grafos para λFAs

P(0): (zero operações em r)



$r = \emptyset$



$r = \lambda$

A & C: Expressões Regulares

Expressões Regulares geram

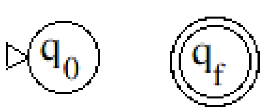
Linguagens Regulares

Dado r ER, existe M λFA tal que L(r) = L(M)

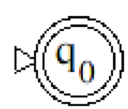
Demonstração: Por indução sobre op(r) =

|operações em r|, usando grafos para λFAs

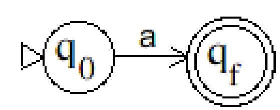
P(0): (zero operações em r)



r = $\varnothing$



r = λ

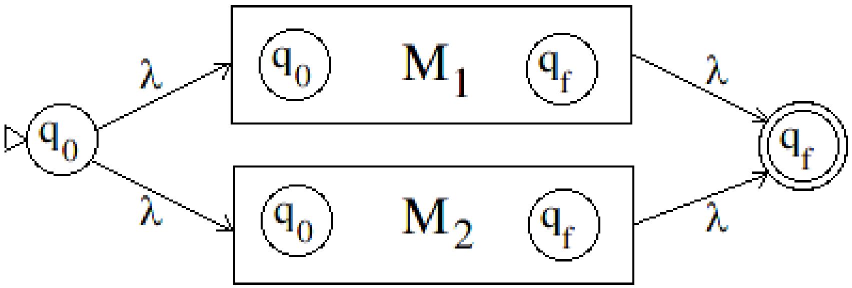


r = a

A & C: Expressões Regulares

ER geram Linguagens Regulares (cont)

$\forall n \in \mathbb{N}$ [P(n)$\Rightarrow$P(n+1)]: supondo $L(r) \in \mathcal{L}_{Reg}$ se op(r)=n



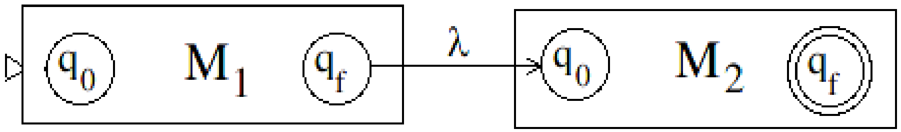
Renomeando estados, λFA M que reconhece

$$r = r_1 + r_2$$

A & C: Expressões Regulares

ER geram Linguagens Regulares (cont)

$\forall n \in \mathbb{N}\ [P(n) \Rightarrow P(n+1)]$: supondo $L(r) \in \mathcal{L}_{Reg}$ se $op(r)=n$



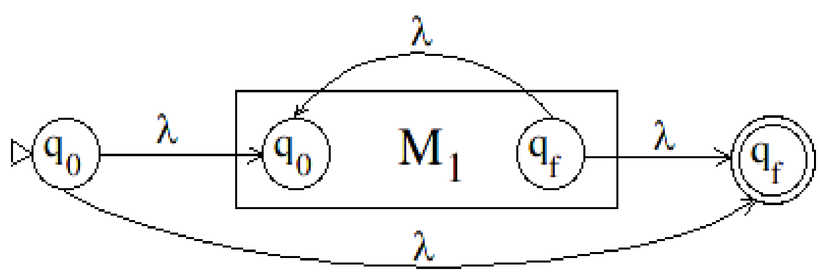
Renomeando estados, λFA M que reconhece

$r = r_1 r_2$

A & C: Expressões Regulares

ER geram Linguagens Regulares (cont)

$\forall n \in \mathbb{N}$ [P(n)$\Rightarrow$P(n+1)]: supondo L(r)$\in \mathcal{L}_{Reg}$ se op(r)=n



Renomeando estados, λFA M que reconhece

$$r = r_1^*$$

A & C: Expressões Regulares

Linguagens Regulares são descritas por ERs

Dado M DFA, existe ER r tal que $L(M) = L(r)$

A & C: Expressões Regulares

Linguagens Regulares são descritas por ERs

Dado M DFA, existe ER r tal que L(M) = L(r)

Demonstração: Por indução sobre (k) em $R_{ij(k)}$,

com Q = {q_1(start),..., q_n} ordenado, $R_{ij(k)}$ dado por:

$$R_{ij(k)} = R_{ik(k-1)}(R_{kk(k-1)})^* R_{ij(k-1)} \cup R_{kj(k-1)}$$

$$R_{ij(0)} = \{a \mid \delta(q_i,a)=q_j\} \; [\cup\{\lambda\} \text{ se } i = j\,]$$

($R_{ij(k)}$ são as cadeias correspondentes aos caminhos de q_i a q_j que não passam por nenhum q_s $s > k$; $L(M) = \bigcup_s R_{1s(n)} \mid q_s \in F$)

A & C: Expressões Regulares

Linguagens Regulares são descritas por ERs

Dado M DFA, existe ER r tal que L(M) = L(r)

Demonstração (cont):

P(0): (base da indução)

$R_{ij(0)} = \{a_{s_1}, \ldots, a_{s_n}\}[\cup\lambda]$ é descrita por $a_{s_1}+\ldots+a_{s_n}[+\lambda]$

A & C: Expressões Regulares

Linguagens Regulares são descritas por ERs

Dado M DFA, existe ER r tal que L(M) = L(r)

Demonstração (cont):

P(0): (base da indução)

$R_{ij(0)} = \{a_{s_1}, \ldots, a_{s_n}\}[\cup\lambda]$ é descrita por $a_{s_1} + \ldots + a_{s_n} [+\lambda]$

$\forall n \in \mathbb{N}$ [P(n) $\Rightarrow$ P(n+1)]: (passo da indução)

$R_{ij(k)}$ é descrita por $r_{ik(k-1)}(r_{kk(k-1)})^* r_{kj(k-1)} + r_{ij(k-1)}$

A & C: Expressões Regulares

Linguagens Regulares são descritas por ERs

Dado M DFA, existe ER r tal que $L(M) = L(r)$

Demonstração (cont):

P(0): (base da indução)

$R_{ij(0)} = \{a_{s_1}, \ldots, a_{s_n}\}[\cup\lambda]$ é descrita por $a_{s_1} + \ldots + a_{s_n} [+\lambda]$

$\forall n \in \mathbb{N}$ [$P(n) \Rightarrow P(n+1)$]: (passo da indução)

$R_{ij(k)}$ é descrita por $r_{ik(k-1)}(r_{kk(k-1)})^* r_{ij(k-1)} + r_{ij(k-1)}$

($R_{ij(k)}$ são as cadeias correspondentes aos caminhos de q_i a q_j que não passam por nenhum q_s $s > k$; $L(M) = \bigcup_s R_{1s(n)} \mid q_s \in F$)