

TEORIA DA COMPUTAÇÃO

AULA 4: AUTÔMATOS E LINGUAGENS (CAP 1)

AS OPERAÇÕES REGULARES

PROFESSOR: LUCAS CAMBUIM

Autômatos Finitos: Operações Regulares

- Em aritmética, os objetos básicos são números e as ferramentas são operações para manipulá-los, tais como $+$ e $\times$.
- Na teoria da computação os objetos são linguagens e as ferramentas incluem operações especificamente projetadas para manipulá-las.
- Definimos três operações sobre linguagens, chamadas *operações regulares*, e as usamos para estudar propriedades de linguagens regulares.
- Isso ajuda a projetar autômatos para reconhecer linguagens específicas.
- Isso ajuda a provar que certas linguagens são não-regulares (isto é, além da capacidade de autômatos finitos).

Autômatos Finitos: Operações Regulares

Sejam A e B linguagens. Definimos as operações regulares *união*, *concatenação*, e *estrela* da seguinte forma.

União: $A \cup B = \{x \mid x \in A \text{ ou } x \in B\}$.

Concatenação: $A \circ B = \{xy \mid x \in A \text{ e } y \in B\}$.

Estrela: $A^* = \{x_1 x_2 \dots x_k \mid k \geq 0 \text{ e cada } x_i \in A\}$.

Operações regulares: Exemplo 1.24

Suponha que o alfabeto Σ seja o alfabeto padrão de 26 letras $\{a, b, \dots, z\}$. Se $A = \{\text{legal}, \text{ruim}\}$ e $B = \{\text{garoto}, \text{garota}\}$, então

- $A \cup B = \{\text{legal}, \text{ruim}, \text{garoto}, \text{garota}\},$
- $A \circ B = \{\text{legalgaroto}, \text{legalgarota}, \text{ruimgaroto}, \text{ruimgarota}\},$ e
- $A^* = \{\varepsilon, \text{legal}, \text{ruim}, \text{legallegal}, \text{legalruim}, \text{ruimlegal}, \text{ruimruim}, \text{legallegallegal}, \text{legallegalruim}, \text{legalruimlegal}, \text{legalruimruim}, \dots\}.$

Operações Regulares

Teorema: A classe de linguagens regulares é fechada sob a operação de união.

Em outras palavras, se A_1 e A_2 são linguagens regulares, o mesmo acontece com $A_1 \cup A_2$.

Idéia da Prova: Se A_1 e A_2 são linguagens regulares, então existem AFs M_1 e M_2 que as reconhecem, respectivamente.

Vamos fazer uma prova **construtiva**, ou seja, vamos construir um AF M , que reconheça $A_1 \cup A_2$, a partir de M_1 e M_2 .

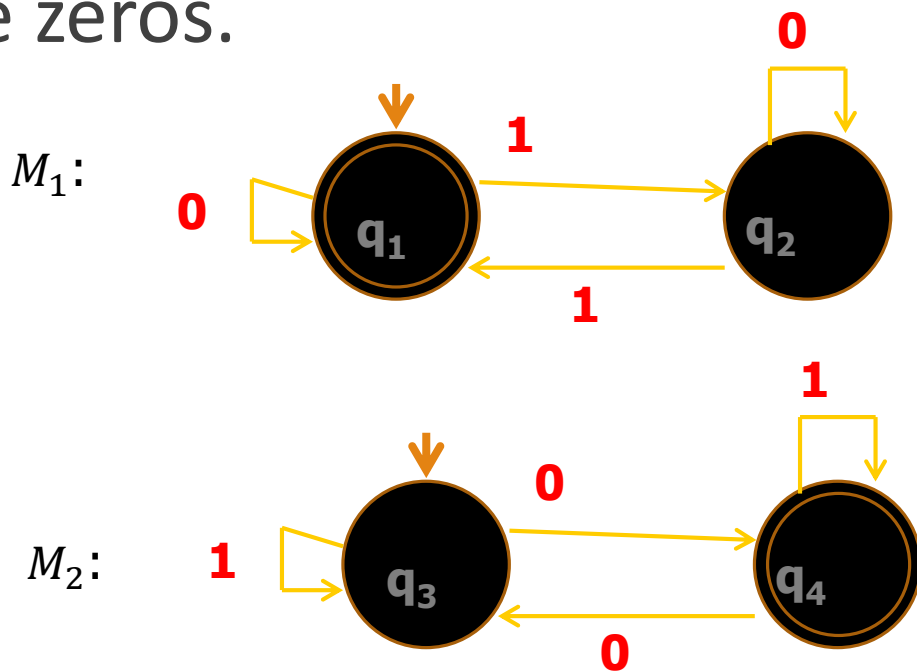
Operações Regulares

Como vamos construir um AF M , que reconheça $A_1 \cup A_2$, a partir de M_1 e M_2 ?

- Simulando M_1 e M_2 simultaneamente.
- Para controlar ambas as simulações é preciso **guardar** o estado em que cada máquina estaria se ela tivesse lido até um ponto na entrada.
- Consequentemente, você precisa guardar um par de estados.
- Quantos pares de estados existem?

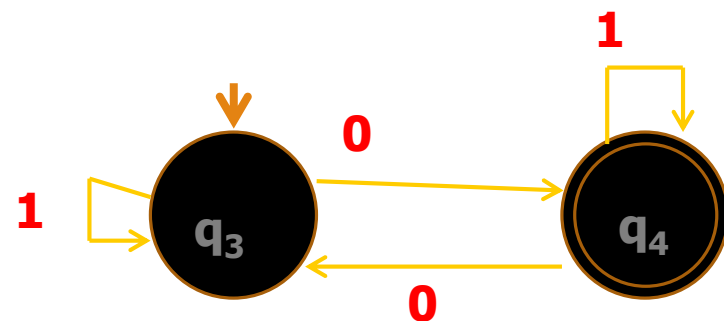
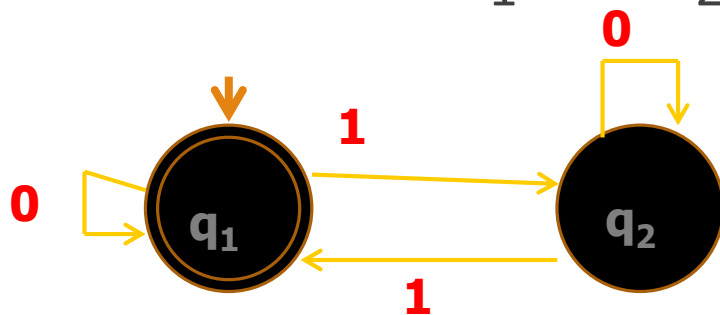
Operações Regulares

Vamos ver um exemplo. Seja M_1 um AF que reconheça as cadeias de bits com um número par de 1s e M_2 reconhece aquelas com um número ímpar de zeros.



Operações Regulares

Construindo $M_1 \cup M_2$

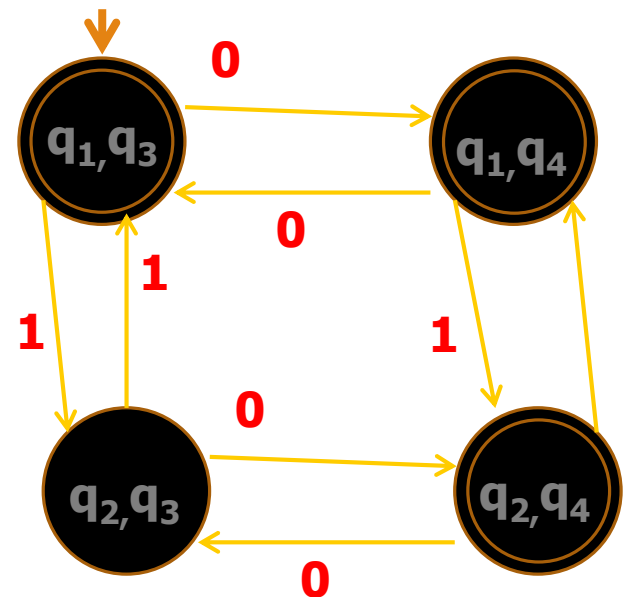


$$M_1 = (Q_1, \Sigma_1, \delta_1, q_1, F_1), \quad M_2 = (Q_2, \Sigma_2, \delta_2, q_3, F_2)$$

$$M = (Q_1 \times Q_2, \Sigma, \delta, (q_1, q_3), F)$$

$$\delta((r_1, r_2), a) = (\delta_1(r_1, a), \delta_2(r_2, a))$$

$$F_1 = (F_1 \times Q_2) \cup (Q_1 \times F_2)$$



Operações Regulares

Construção formal de M para reconhecer $A_1 \cup A_2$, onde $M = (\mathbf{Q}, \Sigma, \mathbf{q}_0, \mathbf{F})$.

1. $\mathbf{Q} = \{(r_1, r_2) | r_1 \in Q_1 \text{ e } r_2 \in Q_2\}$
 - a) Escrito como produto cartesiano $Q_1 \times Q_2$
2. Σ é o mesmo em M_1 e M_2 . Se o alfabeto forem diferentes então $\Sigma = \Sigma_1 \cup \Sigma_2$
3. $\delta = \delta(r_1, r_2), a) = (\delta_1(r_1, a), \delta_2(r_2, a))$
4. $\mathbf{q}_0$ é o par (q_1, q_2)
 - a) Onde q_1 é o estado inicial de A_1 e q_2 é o estado inicial de A_2
5. $\mathbf{F} = \{(r_1, r_2) | r_1 \in F_1 \text{ ou } r_2 \in F_2\}$
 - a) Equivalente à: $F = (F_1 \times Q_2 \cup Q_1 \times F_2)$
 - b) Obs: Não é equivalente à $F = Q_1 \times Q_2$. **Por quê?**

Operações Regulares

?

Construção formal de M para reconhecer $A_1 \cap A_2$, onde $M = (Q, \Sigma, q_0, F)$.

1. $Q = \{(r_1, r_2) | r_1 \in Q_1 \text{ e } r_2 \in Q_2\}$
 - a) Escrito como produto cartesiano $Q_1 \times Q_2$
2. Σ é o mesmo em M_1 e M_2 . Se o alfabeto forem diferentes então $\Sigma = \Sigma_1 \cup \Sigma_2$
3. $\delta = \delta(r_1, r_2), a) = (\delta_1(r_1, a), \delta_2(r_2, a))$
4. q_0 é o par (q_1, q_2)
 - a) Onde q_1 é o estado inicial de A_1 e q_2 é o estado inicial de A_2
5. $F = \{(r_1, r_2) | r_1 \in F_1 \text{ ou } r_2 \in F_2\}$
 - a) Equivalente à: $F = Q_1 \times Q_2$

Operações Regulares

Teorema: A classe de linguagens regulares é fechada sob a operação de concatenação.

Em outras palavras, se A_1 e A_2 são linguagens regulares, o mesmo acontece com $A_1 \circ A_2$.

De modo análogo à prova do teorema anterior, vamos construir um autômato M para reconhecer $A_1 \circ A_2$ a partir de M_1 e M_2 .

M aceita sua entrada se ela puder ser quebrada em duas partes, onde M_1 aceita a primeira parte e M_2 aceita a segunda parte.

O problema é que M não sabe onde quebrar sua entrada. Para resolver esse problema introduzimos uma nova técnica chamada **não-determinismo**.

Exercício

Cada uma das linguagens a seguir é a interseção de duas linguagens mais simples. Em cada caso, construa AFDs para as linguagens mais simples, e depois combine-os usando a construção discutida no slide 10 para obter o diagrama de estados de um AFD para a linguagem dada.

- a) $\{w \mid w \text{ tem pelo menos três } \mathbf{a}\text{s e pelo menos dois } \mathbf{b}\text{s}\}$
- b) $\{w \mid w \text{ tem um número par de } \mathbf{a}\text{s e um ou dois } \mathbf{b}\text{s}\}$
- c) $\{w \mid w \text{ tem um número ímpar de } \mathbf{a}\text{s e termina com um } \mathbf{b}\}$
- d) $\{w \mid w \text{ tem comprimento par e um número ímpar de } \mathbf{a}\text{s}\}$