

Resumo FTC - Prova I

Sophia
Carrazza



Materia que vai cair:

- 2 - Introdução: preliminares matemáticas
- 3 - Exercícios
- 4 - Linguagens: conjuntos e expressões regulares
- 5 - Exercícios
- 6 - Autômatos
- 7 - Exercícios
- 8 - Autômatos e Linguagens: pumping lemma
- 9 - Exercícios
- 10 - Gramáticas

Preliminares Matemáticas

> Relação Binária

↳ uma forma matemática de expressar qualquer tipo de correspondência entre conjuntos:

$$R \subseteq A \times B \quad \xrightarrow{\text{(subconjunto do produto cartesiano } A \times B)}$$

> Reflexiva: $\forall x \in A [x R x] \quad (\forall a \in A : (a, a) \in R)$

> Simétrica: $\forall x, y \in A [x R y \rightarrow y R x]$

$$\hookrightarrow (\forall a, b \in A : (a, b) \in R \Rightarrow (b, a) \in R)$$

> Transitiva: $\forall x, y, z \in A [(x R y \cap y R z) \rightarrow x R z]$

$$\hookrightarrow (\forall a, b, c \in A : (a, b) \in R \cap (b, c) \in R \Rightarrow (a, c) \in R)$$

> Funções

* funções:

parcial $\quad f \text{ de } A \text{ para } B$

- parcial: $f: A \rightarrow B$ é uma relação binária $f \subseteq A \times B$ tal que se $(x, y) \in f$ e $(x, z) \in f$, então $y = z$.
↳ uma função parcial só está definida pr. alguns valores do seu domínio inicial ($x = \frac{1}{x}$, não existe $\frac{1}{0}$, então a função não está definida em zero).

↳ nem todo valor de entrada tem saída garantida.

total

- total: $f: A \rightarrow B$ é uma função parcial definida pr. todo en-
quanto $x \in A$.

↳ um função total está definida pr. todos os valores de argumento inicial ($x: a$ soma sobre os reais, $+R^2 \rightarrow R$. Sua definição não é total, por causa de 0).

↳ qualquer valor de entrada de domínio tem uma saída.

- Injetora: se para $x, y \in A$ quaisquer, $x \neq y \rightarrow f(x) \neq f(y)$

↳ a função nunca envia dois valores diferentes pr. mesmo resultado ($x: f(x) = x+1$).

↳ entradas diferentes dão saídas diferentes.

- Sobreyetora: se B é a imagem de f .

↳ se todo elemento do conjunto de chegada é atingido por al-
guna saída ($x: f(x) = x^3$)

↳ "correspondência um para um"

- Bijetora: se é injetora e sobreyetora. Cada elemento do domínio corresponde a um único elemento da Imagem ($x: f(x) = x+2$)

Sintaxe e Semântica

- ↳ disposição dos polones
na frase e a relação lógica
dos pronomes entre si.
 - recordando antes de numântica
 - tratamento + simples

→ estuda a significade
dos palavras na fala
e dos frases no discurso
- ancladação que a intérprete
- tratamento + elaborado
e subjetivo

> linguagem formal

- alfabeto
 - codia de caracteres / pletos

> alfabeto

- alfabeto
 - conjunto finito de símbolos ou caracteres
 - notação: Σ
 - São alfabetos = $\{a, b, c\}$, \emptyset (conjunto vazio)
 - não são alfabetos = N (conjunto dos naturais), $\{a, bb, cc\}$
 - ↳ é infinito
 - indica que não é finito

> polaron

- sequência finita de símbolos juntapostos ou caractere
 - cadeia de caracteres ; sentença
 - notação : ω

ex em $\{a, b\} \Rightarrow ab, bb, aa, \dots$ (codicis usq.)
 $\hookrightarrow X \Rightarrow ab\dots, aa\dots, abc$

* concatenação \Rightarrow operação binária sobre um conjunto de palavras
- associa duas palavras → é associativa, então podemos sempre trocar os parenteses
- é a juxtaposição da 1^a com a 2^a. de parênteses.

$$v(wt) = (v w)t$$

- Elemento neutro dos palavras: $E \omega = \omega E$

↳ concatenação sucessiva

$$- b^4 = bbbbh \quad - a^0 = E \quad - a^3 b^5 = aaa bb bbb$$

↳ se Σ é um alfabeto:

- Σ^* é o conjunto de todos os palavras periódicas sobre Σ
- $\Sigma^+ = \Sigma^* - E$

> linguagem formal

- é um subconjunto de Σ^*

- ou seja, é um subconjunto de todos "palavras" possíveis dentro de um alfabeto.

- notação: L

↳ ex: \emptyset e $\{E\}$, $\Sigma^* \Sigma^+$ são linguagens sobre qualquer alfabeto

• Conjun

> gramática

↳ é uma quadrupla $G = (V, \Sigma, P, S)$

V = conjunto de símbolos não-terminais ou variáveis (ex: S, A, B)

Σ = alfabeto \rightarrow conjunto de símbolos terminais (ex: a, b, c)

P = conjunto de produções ou regras $P \subseteq (V \cup \Sigma)^* \times (V \cup \Sigma)^*$

S = símbolo inicial ou variável da partida (SEV)

> expressões regulares

↳ é uma notação que representa um conjunto regular (linguagem regular)

↳ sua definição recursiva é:

- BASE

- P.R (padrão recursivo)

- FECHAMENTO

$$\text{ex: } (a \cup b)^* = \{\epsilon, a, b, aa, ab, acc \dots\}$$

> linguagem regular

- ↳ é uma linguagem L regular se e somente se:
- existe um autômato finito que a reconhece (AFD, AFN ou AFN- λ)
 - deve possuir uma expressão regular que a descreve
 - ex regular: $\{ w \mid w \text{ termina com } 0L \}$
 - não regular: $\{ 0^n L^n \mid n \geq 0 \}$ (número m de 0s e ls)
- ↳ provamos com bombeamento

DEFINIÇÕES RECURSIVAS

↳ Todos os números naturais são construídos a partir de 0 e da função de sucessor S.

- definição do conjunto dos naturais \mathbb{N}

$$\mathbb{N}: \begin{cases} \text{base: } 0 \in \mathbb{N} \\ \text{P.R.: se } n \in \mathbb{N}: S(n) \in \mathbb{N} \end{cases} \xrightarrow{\text{sucessor}}$$

- definição de sucessor

$$S(n): \begin{cases} \text{base: } S: \mathbb{N} \rightarrow \mathbb{N}, \\ \text{P.R.: } S(n) = n + 1 \end{cases}$$

Sem usar o operador +:

$$\begin{aligned} \text{soma}(a, 0) &= a \\ \text{soma}(a, S(b)) &= S(\text{soma}(a, b)) \end{aligned}$$

- definição de soma

$$+: \begin{cases} \text{base: } a + 0 = a \\ \text{P.R.: } a + c = a + S(b) = S(a + b) \end{cases}$$

- definição de multiplicação

$$\cdot: \begin{cases} \text{base: } a \cdot 0 = 0 \\ \text{P.R.: } a \cdot c = a \cdot S(b) = a + a \cdot b \\ \quad \underbrace{3 \cdot 4}_{12} = 3 \cdot S(3) = 3 + \underbrace{3 \cdot 3}_{(4)} = \underbrace{3 + 9}_{12} \end{cases}$$

- definição de exponenciação (potência)

base: $a^0 = S(0) \rightarrow$ (ou seja, L)

P.R.: $a^{S(b)} = (a^b) \cdot a$

$$2^3 = 2^{S(2)} = 2^2 \cdot 2$$

1. Dê a definição recursiva do conjunto de strings sobre o alfabeto { a, b } que contenha um número par de b's.

O b's é um tamanto par

{ base: $\lambda \in L, a \in L$

{ P.R.: - se $w \in L$: $\begin{cases} wa \in L \\ aw \in L \end{cases}$ → se $w \in L$, já tem qtd par de b's, então odd. a's não altera

- se $w \notin L$: $\begin{cases} wb \in L \\ bw \in L \end{cases}$ → se $w \notin L$, qtd de b's é impar, então adicionando + uma dica par.

- se $w_1, w_2 \in L$: $w_1 b w_2 \in L$ + dais b's mantém a paridade par.

> Autômatos:

> sistemas de estados finitos

- modelo matemático de sistema
- comporta por entradas e estados
- não tem memória
- tem um n° finito e predefinido de estados
- a máquina só pode estar em um estado por vez

> tipos:

determinístico

- a partir de um determinado estado e do símbolo lido, pode assumir um único estado.

não-determinístico

- a partir de um determinado estado e do símbolo lido pode assumir um conjunto de estados.

com mov. vagas

- a partir de um determinado estado e sem ler um símbolo pode assumir um conjunto de estados.

* Autômato Finito Determinístico (AFD)

↳ possui um único estado de transição para cada par de estado e símbolo de entrada, resultando em um caminho único e fixo.

$$M = (\Sigma, Q, \delta, q_0, F)$$

alfabeto conjuntos de função de estado inicial estados finais
estados possíveis transições

$\delta: Q \times \Sigma \rightarrow Q$

→ estado inicial → estados finais → função de transição por tabela → ex: $\delta(p, a) = q$

q_0

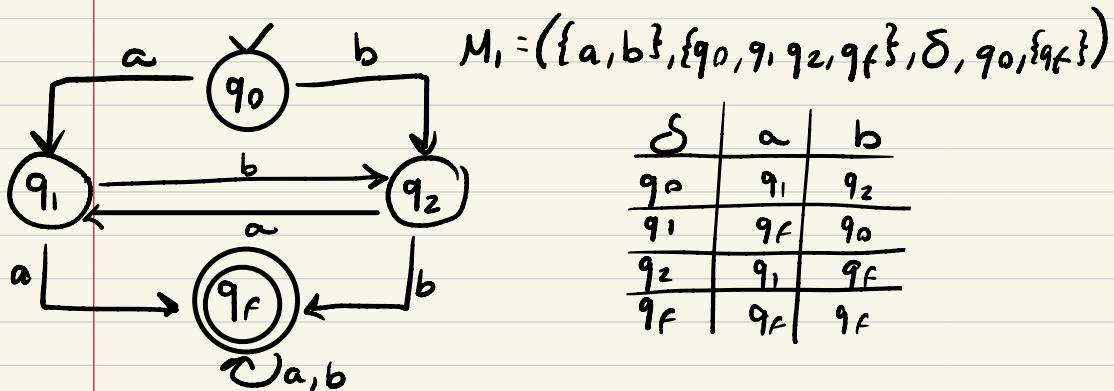
q_f

δ	a	...
p	q	...
q

→ exemplo:

$\Sigma = \{a, b\}$ que pedia como subpalavra aa ou bb

↳ $L = \{w \mid w \text{ possui } \underline{aa} \text{ ou } \underline{bb} \text{ como subpalavra}\}$



δ	a	b
q_0	q_1	q_2
q_1	q_f	q_0
q_2	q_1	q_f
q_f	q_f	q_f

Determinante Incompleto = quando nem todos os transições não especificados (a função de δ é parcial)

* Para toda linguagem finita existe um AFD!!!
 → ele pode ser não mínimo!

* Autômato Finito Não Determinístico (AFN)

↳ permitem + de uma transição partindo de um estado com o mesmo símbolo do alfabeto.

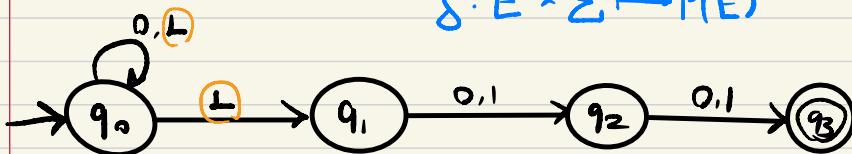
$$\hookrightarrow M = (E, \Sigma, S, i, F)$$

cogunto de estados livres

↳ alfabeto Σ → estados de entrada
cogunto de estados → estado inicial

$$S: E \times \Sigma \rightarrow P(E)$$

↳ função de transição



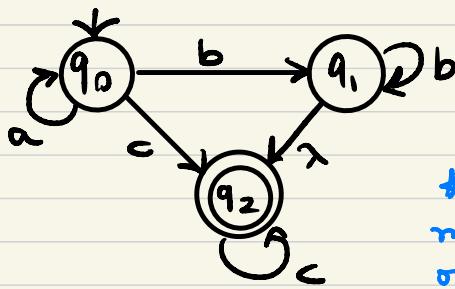
Uma palavra pode gerar diferentes computações!

AFN
não

* Autômato Finito com Movimentos Vazios (AFN- λ)

↳ é não determinístico, e possui transições lambda (λ) ou epsilon (ϵ) vazias, que permitem que o autômato mude de estado sem consumir nenhum símbolo.

- generaliza os movimentos não-determinísticos
- transição na leitura, vazio (incapsulada).



$$M = (\Sigma, Q, S, q_0, F)$$

* em uma transição vazia, ambos mudam simultaneamente o estado origem e destino!

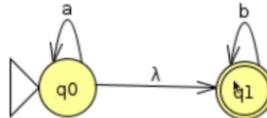
- $L = \{ w \mid w \text{ possui } a's \text{ que antecedem } b's \}$

- AFN_ε

- Matriz de transição

δ	a	b	ϵ
q_0	{ q_0 }	-	{ q_1 }
q_1	-	{ q_1 }	-

Exemplo de
AFN- λ



TRANSFORMAÇÕES

↳ AFN- λ → AFN

(remoção de transições vazias)

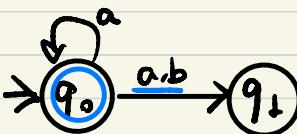
p/ todo AFN- λ , existe um AFN.

* para a passo:

- anularmos tanto o nó origem quanto o destino, p/ cada símbolo do alfabeto.
- substituirmos o vazio por todos os símbolos do alfabeto (pq pode vir qualquer um delas)

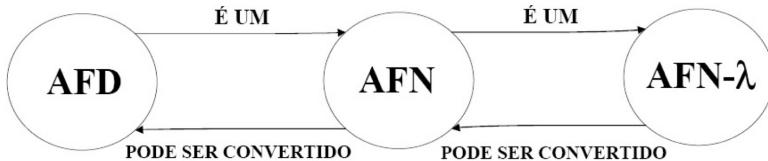
neste mesmo exemplo:

a	b	λ
$q_0 \{q_0\}$	$\{\}$	$\{q_1\}$
$q_1 \{\}$	$\{q_1\}$	$\{\}$



se não substituir os vazios, a palavra nula não é aceita.

↳ devemos anular cada caso específico



↳ AFN → AFD

* passo a passo:

1º tabela de transição AFN

2º tabela de transição AFD

↳ estados = todos os possíveis combinações do AFN

↳ total de estados = 2^n

↳ quando um estado for um conjunto, o novo será a união das transições

3º marcar os novos estados finais

↳ todos os conjuntos que contêm algum estado final do AFN

4º eliminar os estados:

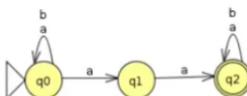
↳ inalcancíveis

↳ que não permitem saídas e não são finais

5º verificar no grafo se há algum ciclo inalcancível.

AFN to AFD

- Exemplo



δ	a	b
q0	{q0,q1}	{q0}
q1	{q2}	-
q2	{q2}	{q2}

Tabela de transição do AFN

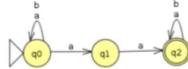
- Estado $\{q0, q1\}$

- São dois estados, logo será a união deles
- AFN, q0 ao processar "b" vai para {q0}
- AFN, q1 ao processar "b" vai para -
 $\{q0\} \cup - = \{q0\}$

φ	a	b
{q0}	{q0,q1}	{q0}
{q1}	{q2}	-
{q2}	{q2}	{q2}
{q0,q1}	{q0,q1,q2}	{q0}
{q0,q2}		
{q1,q2}		
{q0,q1,q2}		
\emptyset		

Tabela de transição do AFD

• Exemplo



δ	a	b
q0	{q0,q1}	{q0}
q1	{q2}	-
q2	{q2}	{q2}

Tabela de transição do AFN

	φ	a	b
S0	{q0}	S3	S0
S3	{q0,q1}	S6	S0
*S4	{q0,q2}	S6	S3
*S6	{q0,q1,q2}	S6	S4

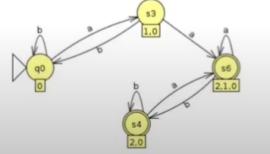


Tabela de transição do AFD

	φ	a	b
S0	{q0}	S3	S0
S1	{q1}	S2	-
*S2	{q2}	S2	S2
S3	{q0,q1}	S6	S0
*S4	{q0,q2}	S6	S4
*S5	{q1,q2}	S2	S2
*S6	{q0,q1,q2}	S6	S4
S7	-	-	-

R

- Qual(is) era(m) final(is) no AFN?
- R: q2
- Agora basta marcar no AFD todos que tem esses estados, eles serão os novos estados finais

- Eliminar estados inalcançáveis
 - Na prática é eliminar os estados que não aparecem nas colunas "a" e "b" de outro estado
 - Ou seja, ninguém chega nele
- eliminar os estados que não permitem saídas e não são finais (loops)

Expressão Regular a Partir de um Autômato:

$$\text{I} - \text{ } O \xrightarrow{\alpha} O \xrightarrow{\beta} O \Rightarrow O \xrightarrow{\alpha\beta} O$$

$$\text{II} - O \xrightarrow{\alpha} O \xrightarrow{\beta} \overset{\beta}{\circlearrowleft} \xrightarrow{\gamma} O \Rightarrow O \xrightarrow{\alpha\beta^*\gamma} O$$

$$\text{III} - O \xrightarrow{\alpha} O \xrightarrow{\beta} O \Rightarrow O \xrightarrow{\alpha+\beta} O$$

Lema do Bombeamento

Em um autômato finito, um trecho de cadeia pode ser repetido em um loop várias vezes ("bombeado") sem nunca levar o autômato a um estado não aceitante - sem tirar a palavra da linguagem.

Então, quando suspeitarmos que uma linguagem L não pertence aos regulares (pode ter loops infinitos), utilizaremos esse lema como prova por contradição.

→ (ou abundo)?

* O LEMA:

Seja L uma linguagem regular. Então existe um inteiro $K \geq L$ (comprimento de bombeamento) tal que, p/ todo palavra $w \in L$ com $|w| \geq K$, podemos escrever:

$$w = pqv$$

ex: $w = 0011$

$$w = \lambda | 00 | 11$$

$P | P | V$

Satisfazendo:

I) $w = pqv$

II) $|pq| \leq K$

III) $|q| \neq \lambda$ ($|q| \neq 0$)

IV) $w_i = pq^i v \in L \quad \forall i \geq 0 \rightarrow$ tem que continuar fazendo parte da linguagem

* $|w| \geq K$ pq a palavra deve

passar por um loop para fazermos o lema!