Revisão de autômatos finitos determinísticos e de expressões regulares

- Definições
  - Alfabeto → conjunto finito de símbolos.
  - String → sequência de símbolos de um determinado alfabeto.
    (inclui a string vazia ε)
  - Linguagem → conjunto de strings

- DFA - Deterministic Finite Automata
  - o formalismo matemático para definir linguagens.

Consiste em 5 coisas:

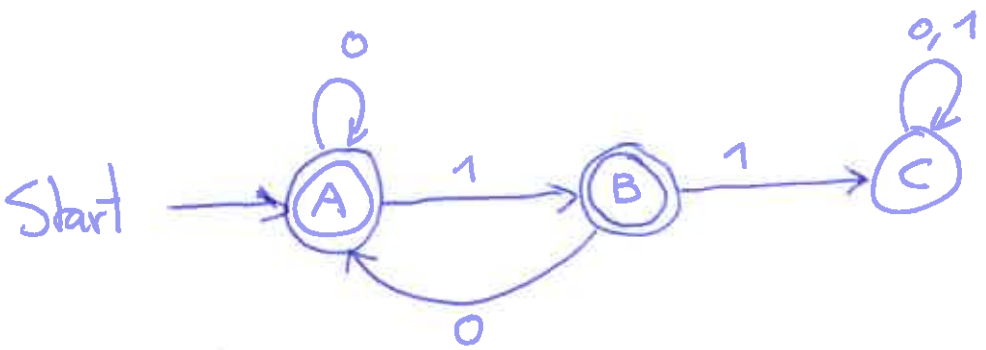
1) Conjunto finito de estados (S)
2) Alfabeto de input (Σ)
3) Função de transição (δ)
4) Estado inicial ($s_0 \in S$)
5) Conjunto de estados finais ($F \subseteq S$)

- Função de transição $\delta(s, a) = s'$

  - s: estado $\in S$
  - a: símbolo $\in \Sigma$
  - s': estado $\in S$

- Um DFA pode ser representado como um grafo orientado.
  - estados → nós
  - transições → arcos
  - seta com label "Start" para o estado inicial.
  - estados finais anotados com duplo círculo.

- Exemplo:



↳ Aceita todas as strings binárias que não tenham 2 1s consecutivos.

$$\Sigma = \{0, 1\}$$

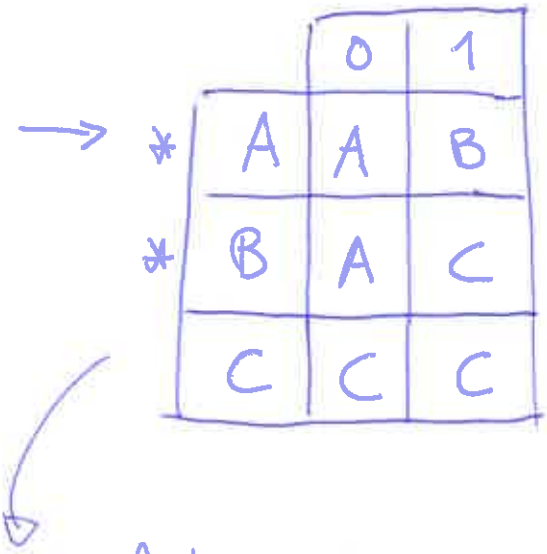
$$S = \{A, B, C\}$$

$$s_0 = A$$

$$F = \{A, B\}$$

$$\begin{cases} \delta(A,0) = A \\ \delta(A,1) = B \\ \delta(B,0) = A \\ \delta(B,1) = C \\ \delta(C,0) = C \\ \delta(C,1) = C \end{cases}$$

DFA também pode ser representado na forma de tabela



		0	1
→ *	A	A	B
*	B	A	C
	C	C	C

← símbolos de input nas colunas

uma linha para cada estado.

- Estados finais anotados com *
- Estado inicial anotado com '→'

(quando não há seta assume-se o estado que está na 1ª linha como sendo o estado inicial)

Função de transição extendida

$\delta(s, z)$

- s: estado
- z: string de símbolos do alfabeto

Se $z = a_1 a_2 \dots a_n$

$$\begin{aligned}\delta(s, z) &= \delta(s, a_1 a_2 \dots a_n) \\ &= \delta(\delta(s, a_1), a_2 \dots a_n) \\ &= \delta(\delta(\delta(s, a_1), a_2), a_3 \dots a_n) \\ &= \dots\end{aligned}$$

Formalmente por indução:

Base: $\delta(s, \varepsilon) = s$

Indução: $\delta(s, wa) = \delta(\delta(s, w), a)$

- w: string
- a: símbolo

Linguagem de um DFA

- Se A é um autómato, $L(A)$ é a linguagem definida pelo autómato.

- $L(A)$ é o conjunto de todas as strings que andam caminhos do estado inicial até um estado final. Ou seja, é o conjunto de todas as strings que são aceites pelo autómato.

$$L(A) = \{ w \mid \delta(s_0, w) \in F \}$$

s_0: estado inicial

F: estados finais

Linguagens Regulares

- Uma linguagem L é regular se todas as strings $\in L$ forem aceites por um DFA e esse DFA não aceitar nenhuma outra string.
- Nem todas as linguagens são regulares.

Expressões Regulares

- são usadas para descrever linguagens regulares
- expressões regulares definem linguagens regulares.
- Se E é uma e.r., $L(E)$ é a linguagem definida pela e.r.

- A descrição de E.Rs. e suas linguagens pode ser feita de forma recursiva.

Base

1. ε é uma E.R., e $L(\varepsilon) = \{\varepsilon\}$
2. Se $\underline{a}$ é um símbolo, então $\underline{a}$ é uma E.R., e $L(a) = \{a\}$

Indução

1. Se E_1 e E_2 são E.Rs, então $E_1|E_2$ é uma E.R. e $L(E_1|E_2) = L(E_1) \cup L(E_2)$

↓

ou

Conjunto de strings x tais que $x \in L(E_1)$ ou $x \in L(E_2)$

2. Se E_1 e E_2 são E.Rs., então

E_1E_2 é uma E.R., e

$L(E_1E_2) = L(E_1)\,L(E_2)$

↓

Concatenação

conjunto de strings xy tais que

$x \in L(E_1)$ e $y \in L(E_2)$

3. Se E é uma E.R., então E^* é uma E.R.,

e $L(E^*) = (L(E))^*$

↓

fecho ou iteração

(em inglês, closure ou Kleene closure)

conjunto de strings $w_1 w_2 \ldots w_n$ para um
$n \geq 0$ tais que cada $w_i \in L(E)$. Ou seja,

E^* corresponde a zero ou mais concatenações de E.

Precedência de operadores

maior precedência

1. * ⇝ closure
2. concatenação
3. | ⇝ ou

menor precedência

Usamos parentesis para agrupar, tal como em expressões aritméticas.

Exemplos:

$L(01) = \{01\}$

$L(01|0) = \{01, 0\}$

$L(1(0|1)) = \{10, 11\}$

$L(0^*) = \{\varepsilon, 0, 00, 000, \ldots\}$