Aula 002 – Linguagens Regulares

Autômatos Finitos Determinísticos

Prof. Rogério Aparecido Gonçalves[1]
rogerioag@utfpr.edu.br

[1]Universidade Tecnológica Federal do Paraná (UTFPR)
Departamento de Computação (DACOM)
Campo Mourão - Paraná - Brasil

Bacharelado em Ciência da Computação
Ciência da Computação
BCC34B - Linguagens Formais, Autômatos e Computabilidade

Linguagens Regulares

Linguagens Regulares i

- O estudo de **Linguagens Regulares** ou tipo 3 é abordado usando os seguintes formalismos, pelos quais elas podem ser geradas ou reconhecidas.
  - **Autômatos Finitos:** Formalismo operacional ou reconhecedor, sendo um sistema de estafos finitos. Autâmatos Finitos Determinísticos e Autômatos Finitos Não Determinísticos.
  - **Conjuntos Regulares (Expressões Regulares):** Formalismo denotacional, também considerado gerador, pois se pode inferir como construir todas as palavras da correspondente linguagem. Definido a partir de conjuntos (linguagens) básicos e de operações de concatenação e união.
  - **Gramáticas Regulares:** Trata-se de um formalismo axiomático ou gerador o qual, como o nome indica, é uma gramática, mas com restrições da forma das regras de produção.

Linguagens Regulares ii

- De acordo com a **Hierarquia de Chomsky**, as linguagens regulares constituem a classe de linguagens mais simples, sendo posssível desenvolver algoritmos de reconhecimento, de geração e conversão entre formalismos de pouca complexidade, de grande eficiência e de fácil implementação.
- Linguagens Regulares possuem fortes limitações de expressividade. Por exemplo, uma linguagem que possua duplo balanceamento não é uma linguagem regular. [Parênteses balanceados, cada um aberto tenha o seu correspondente, aninhamento de IFs e ELSEs]
- Portanto o tratamento de expressões aritméticas comuns não pode ser realizado usando formalismo regular. A maioria das linguagens de programação de propósito gerais como Pascal, C, Java, etc... são não regulares.

Linguagens Regulares iii

- Um exemplo típico e simples de utilização de linguagens regulares é a **Análise Léxica**. Outros exemplos: Sistemas de animação, hipertextos e hipermídias.

Definição

Uma linguagem é chamada de uma **linguagem regular** se algum autômato finito a reconhece.

Autômatos Finitos

- *Autômatos Finitos* ou *Máquinas de Estados Finitos* são uma forma matemática de descrever tipos particulares de algoritmos (ou "máquinas").
- Podem ser utilizados para descrever o processo de reconhecimento de padrões em cadeias de entrada, e assim podem ser utilizados para construir **sistemas de varredura**.
- Há uma forte relação entre **autômatos finitos** e **expressões regulares**
  - Autômatos possibilitam a implementação de **reconhecedores** de linguagens geradas por gramáticas ou especificadas por Expressões Regulares.

Definição

Um **autômato finito determinístico** é uma 5-upla $M = (Q, \Sigma, \delta, q_0, F)$, onde:

1. Q é um conjunto finito de **estados**,
2. Σ é um conjunto finito chamado o **alfabeto** de entrada,
3. δ é a **função de transição**, tal que $\delta : Q \times \Sigma \rightarrow Q$,
4. q_0 é o **estado inicial**, tal que $q_0 \in Q$ e
5. $F \subseteq Q$ é **o conjunto de estados de aceitação**.

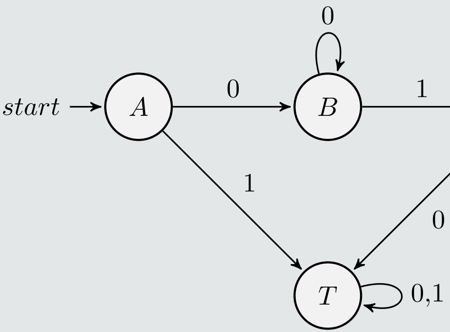
Exemplo

$\{0^n1^m | n \geq 1 \land m \geq 1\}$

Autômatos Finitos iv

Exemplo

$\{0^n 1^m | n \geq 1 \land m \geq 1\}$



Autômatos Finitos v

Tabela de Transições

Estado	0	1
A	B	T
B	B	C
C*	T	C
T	T	T

Exemplo: Processamento de uma palavra

Estado atual	Símbolo atual	Estado Resultante	Palvra restante
A	0	$\delta(A, 0) = B$	0111
B	0	$\delta(B, 0) = B$	111
B	1	$\delta(B, 1) = C$	11
C	1	$\delta(C, 1) = C$	1
C	1	$\delta(C, 1) = C$	ϵ

Função de transição estendida

Dado o AFD $A =< Q, \Sigma, q_0, \delta, F >$, a Função de Transição Estendida δ é definida como $\delta : Q \times \Sigma^* \to Q$, tal que:

- $\forall s \in Q, \delta(s, \epsilon) = s$
- $(\forall s \in Q) \wedge (\forall a \in \Sigma) \wedge (\forall x \in \Sigma^* \mid |x| = 1 \wedge x_1 = a), \delta(s, x) = \delta(s, a)$
- $(\forall s \in Q) \wedge (\forall a \in \Sigma) \wedge (\forall x \in \Sigma^*), \delta(s, ax) = \delta(\delta(s, a), x)$

Função de transição estendida

Dado o AFD $A =< Q, \Sigma, q_0, \delta, F >$, a Função de Transição Estendida δ é definida como $\delta : Q \times \Sigma^* \rightarrow Q$, tal que:

- $\forall s \in Q, \delta(s, \epsilon) = s$
  - Para todo estado s do autômato finito, a função de transição estendida a partir desse estado s e para a palavra vazia (ϵ) terá como resultado o próprio estado s.
- $(\forall s \in Q) \wedge (\forall a \in \Sigma) \wedge (\forall x \in \Sigma^* \mid |x| = 1 \wedge x_1 = a)$, $\delta(s, x) = \delta(s, a)$
- $(\forall s \in Q) \wedge (\forall a \in \Sigma) \wedge (\forall x \in \Sigma^*)$, $\delta(s, ax) = \delta(\delta(s, a), x)$

Autômatos Finitos ix

Função de transição estendida

Dado o AFD $A =< Q, \Sigma, q_0, \delta, F >$, a Função de Transição Estendida δ é definida como $\delta : S \times \Sigma^* \rightarrow S$, tal que:

- $\forall s \in Q, \delta(s, \epsilon) = s$
- $(\forall s \in Q) \wedge (\forall a \in \Sigma) \wedge (\forall x \in \Sigma^* \mid |x| = 1 \wedge x_1 = a), \delta(s, x) = \delta(s, a)$
    - Para todo estado s do autômato finito, para todo símbolo a do alfabeto de entrada e para toda palavra x de tamanho 1 e que tenho o símbolo a, a função de transição estendida a partir do estado s e para a palavra x terá o mesmo resultado da função de transição (normal) aplicada para o estado s e o símbolo a do alfabeto de entrada.
- $(\forall s \in Q) \wedge (\forall a \in \Sigma) \wedge (\forall x \in \Sigma^*), \delta(s, ax) = \delta(\delta(s, a), x)$

Função de transição estendida

Dado o AFD $A =< Q, \Sigma, q_0, \delta, F >$, a Função de Transição Estendida δ é definida como $\delta : S \times \Sigma^* \rightarrow S$, tal que:

- $\forall s \in Q, \delta(s, \epsilon) = s$
- $(\forall s \in Q) \wedge (\forall a \in \Sigma) \wedge (\forall x \in \Sigma^* \mid |x| = 1 \wedge x_1 = a), \delta(s, x) = \delta(s, a)$
- $(\forall s \in Q) \wedge (\forall a \in \Sigma) \wedge (\forall w \in \Sigma^*) \wedge (w = ax), \delta(s, ax) = \delta(\delta(s, a), x)$
    - Para todo estado s do autômato finito, para todo símbolo do alfabeto de entrada e para toda palavra w considerando o alfabeto de entrada (sendo que a palavra w possui o símbolo a como prefixo de uma palavra x), temos que, para a função de transição estendida a partir do estado s e para a palavra w, o resultado será igual ao da função de transição (normal) a partir do estado s e com o primeiro símbolo a que prefixa a palavra w em questão.

AFD: Decisão sobre a pertinência de uma palavra i

Pertinência

Dado o AFD $A =< Q, \Sigma, q_0, \delta, F >$ e uma palavra $x \in \Sigma^*$, temos que:

- A aceita x se e somente se (sse) $\delta(S_0, x) \in F$.
- A rejeita x se e somente se (sse) $\delta(S_0, x) \notin F$.

Linguagem de AFD

Dado o AFD $A =< Q, \Sigma, q_0, \delta, F >$, define-se a linguagem $L(A)$, ou seja, a linguagem reconhecida pelo autômato A, como:

- $L(A) = \{x \in \Sigma^* \mid \delta(S_0, x) \in F\}$

Configuração de AFD

Representa a configuração atual do autômato. Permite:

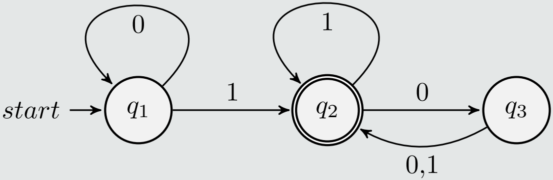
- continuar a execução em outro momento,
- estabelecer as configurações possíveis do autômato.

Para um AFD, é suficiente conhecer: * estado atual * sequência de símbolos que faltam ser processados

Não é necessário saber quais estados que foram alcançados anteriormente. * Isto não interfere na aplicação da função de transição e, por conseguinte, no funcionamento do autômato.

Exercícios

Descreva o Autômato Finito



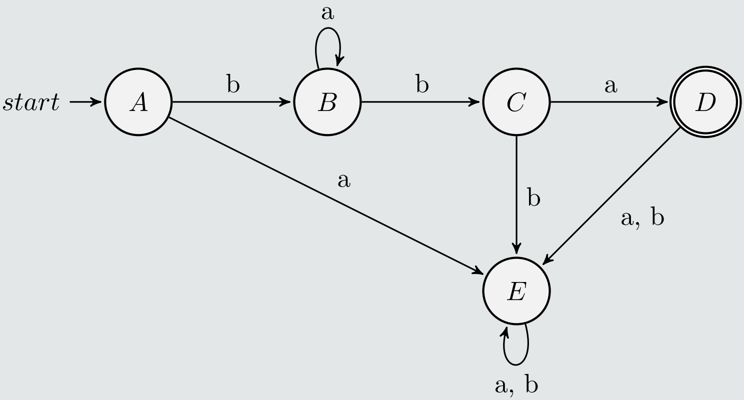
Podemos descrever M_1 escrevendo $M_1 = (Q, \Sigma, \delta, q_1, F)$:

1. $Q = ?$
2. $\Sigma = ?$
3. $\delta = ?$
4. Estado inicial = ?
5. $F = ?$

1. AFD $A = \{ba^n ba \mid n \geq 0\}$

1. AFD $A = \{ba^nba \mid n \geq 0\}$

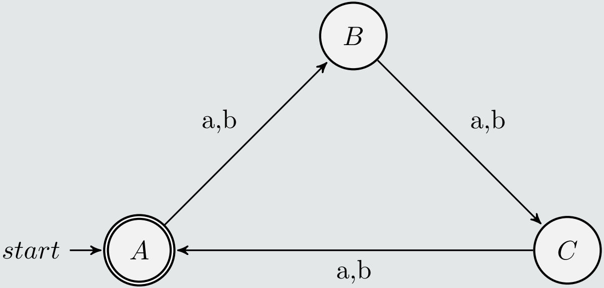
Solução



2. AFD $A = \{x \in \{a, b\}* \mid |x| \text{ mod } 3 = 0\}$

2. AFD $A = \{x \in \{a, b\}* \mid |x| \text{ mod } 3 = 0\}$

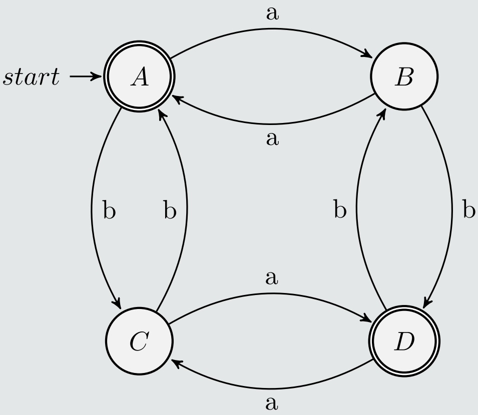
Solução



3. $A = \{w \in \{a, b\}^* \mid (|w|_a + |w|_b) \text{ mod } 2 = 0\}$

3. $A = \{w \in \{a,b\}^* \mid (|w|_a + |w|_b) \bmod 2 = 0\}$

Solução

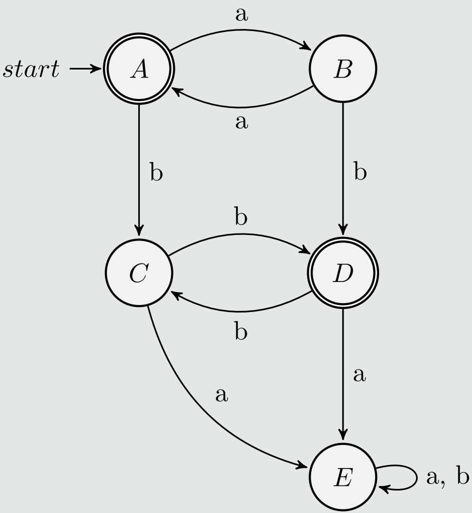


4. AFD $A = \{a^m b^n \mid m, n \geq 0 \wedge (m + n) \text{ mod } 2 = 0\}$

Exercícios – 4 ii

4. AFD $A = \{a^m b^n \mid m, n \geq 0 \wedge (m+n) \bmod 2 = 0\}$

Solução



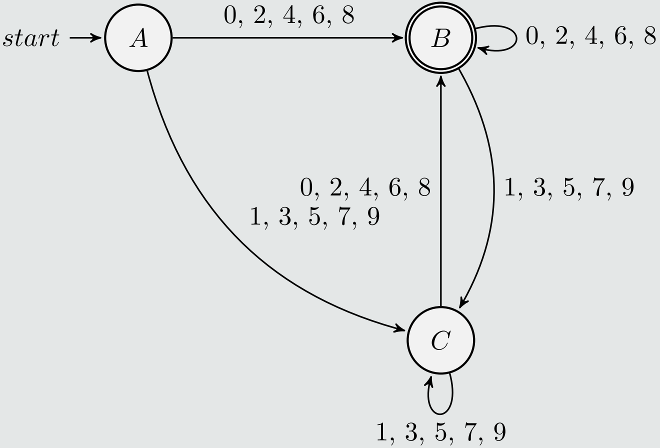
Considere $\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ e defina autômatos finitos determinísticos para as seguintes linguagens:

5. AFD $A = \{x \in \Sigma^* \mid x \text{ mod } 2 = 0\}$
6. AFD $A = \{x \in \Sigma^* \mid x \text{ mod } 5 = 0\}$

Considere $\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ e defina autômatos finitos determinísticos para as seguintes linguagens:

5. AFD $A = \{x \in \Sigma^* \mid x \text{ mod } 2 = 0\}$

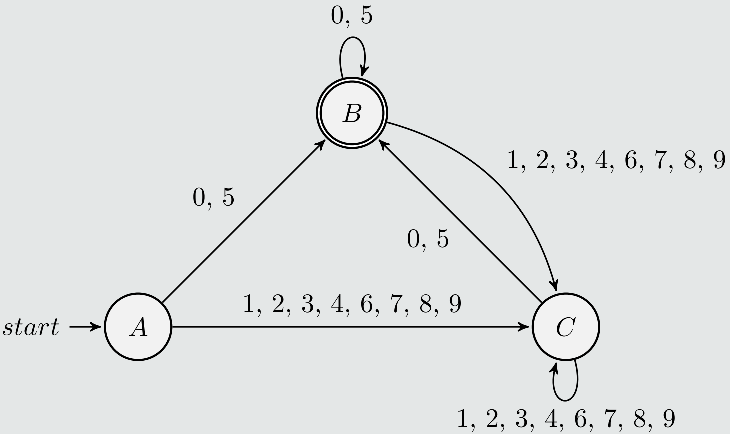
Solução



Considere $\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ e defina autômatos finitos determinísticos para as seguintes linguagens:

6. AFD $A = \{x \in \Sigma^* \mid x \text{ mod } 5 = 0\}$

Solução



Próximas Aulas

Próximas Aulas: Linguagens Regulares

- Linguagens Regulares

Livros Texto

.

MENEZES, P. B. Linguagens formais e autômatos. Porto Alegre: Bookman, 2011. ISBN 9788577807994.

SIPSER, M. Introdução à teoria da computação. [s. l.]: Thomson Learning, 2006. ISBN 9788522104994.

Word Cloud

Referências i

Menezes, Paulo Blauth. 2011. *Linguagens Formais e Autômatos.* Bookman. https://search.ebscohost.com/login.aspx?direct=true&db=edsmib&AN=edsmib.000000444&lang=pt-br&site=eds-live&scope=site.

Sipser, Michael. 2007. *Introdução à Teoria Da Computação.* Cengage Learning. https://search.ebscohost.com/login.aspx?direct=true&db=edsmib&AN=edsmib.000008725&lang=pt-br&site=eds-live&scope=site.