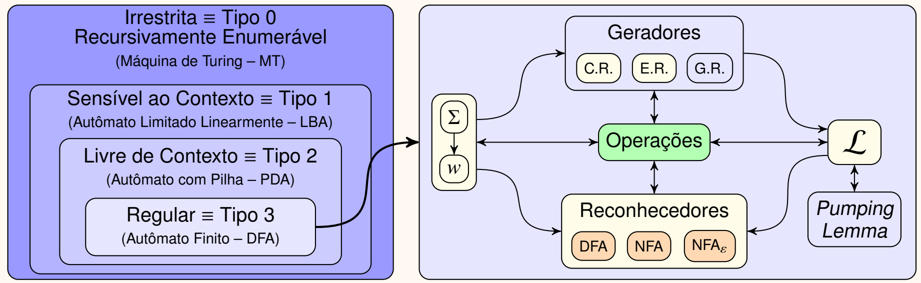
Linguagens Formais e Autômatos

Humberto Longo

Instituto de Informática
Universidade Federal de Goiás

Bacharelado em Ciência da Computação, 2021/1

Roteiro



Equivalência entre DFA's e NFA's

- Existe uma linguagem aceita por um NFA e que não é aceita por nenhum DFA?
  - Nao! Todo DFA é um NFA.
  - NFA é uma generalização do conceito de DFA.
- Dado um NFA qualquer, pode-se construir um DFA equivalente:
  - DFA que aceita a mesma linguagem que o NFA.

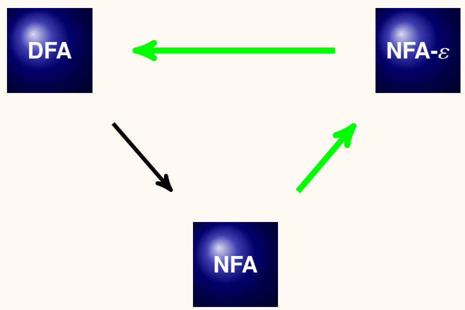
NFA-ε – autômato com transição ε
(Nondeterministic Finite Automata with ε-moves)

NFA – autômato não determinístico
(Nondeterministic Finite Automata)

DFA – autômato finito
(Deterministic Finite Automata)

Relação entre classes de autômatos finitos

- Dado um NFA qualquer, pode-se construir um DFA equivalente:



Fecho ε

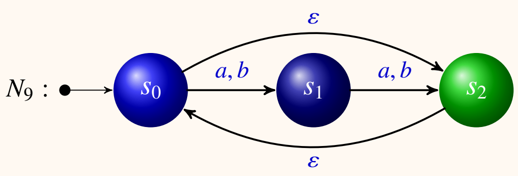
Definição 1.59

- O **Fecho** ε de um estado s_i, $\mathcal{F}_\varepsilon(s_i)$, é definido recursivamente como:

  Base: $s_i \in \mathcal{F}_\varepsilon(s_i)$.

  Recursão: Seja $s_j \in \mathcal{F}_\varepsilon(s_i)$. Se $s_k \in \delta(s_j, \varepsilon)$, então $s_k \in \mathcal{F}_\varepsilon(s_i)$.

  Fecho: $s_j \in \mathcal{F}_\varepsilon(s_i)$ somente se pode ser obtido a partir de s_i com a aplicação da recursão um número finito de vezes.



$\mathcal{F}_\varepsilon(s_0) = \{s_0, s_2\}$

$\mathcal{F}_\varepsilon(s_1) = \{s_1\}$

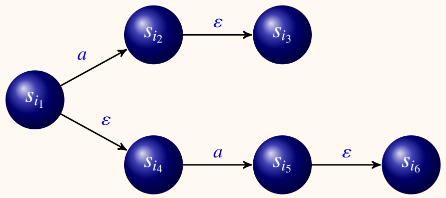
$\mathcal{F}_\varepsilon(s_2) = \{s_0, s_2\}$

Função de transição na cadeia

- A função δ de transição de estados em DFA's e NFA's "processa" os símbolos da cadeia de entrada.
- A função τ, em um NFA-ε, relaciona as transições ao processamento da cadeia de entrada.
  - $\tau(s_i, a)$: conjunto de estados alcançáveis, a partir de s_i, pelo processamento do símbolo a.
- Construção de $\tau(s_i, a)$ envolve três subconjuntos:
  1. estados alcançáveis, a partir de s_i, sem processar a;
  2. estados alcançáveis, a partir dos estados construídos no passo 1, ao processar a;
  3. estados alcançáveis, a partir dos estados construídos no passo 2, com transições vazias.

Função de transição na cadeia

Caminho	Símbolo
s_{i_1}, s_{i_2}	a
$s_{i_1}, s_{i_2}, s_{i_3}$	a
s_{i_1}, s_{i_4}	ε
$s_{i_1}, s_{i_4}, s_{i_5}$	a
$s_{i_1}, s_{i_4}, s_{i_5}, s_{i_6}$	a



- $\tau(s_{i_1}, a) = \{s_{i_2}, s_{i_3}, s_{i_5}, s_{i_6}\}$.
  - O estado s_{i_4} não faz parte do conjunto porque a transição a partir de s_{i_1} não processa o símbolo a.

Função de transição na cadeia

Definição 1.60

A Função τ de Transição na Cadeia de um NFA-ε $N = \langle \Sigma, S, s_0, \delta, F \rangle$ é uma função de $S \times \Sigma$ em $\mathcal{P}(S)$, definida por:

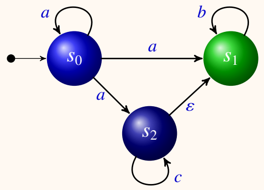
$$\tau(s_i, a) = \bigcup_{s_j \in \mathcal{F}_\varepsilon(s_i)} \mathcal{F}_\varepsilon(\delta(s_j, a)).$$

- A função τ tem a mesma forma da função δ.
- A função τ é idêntica à função δ de um NFA sem transições ε.

Função de transição na cadeia

Exemplo 1.61

- NFA-ε N, tal que $\mathcal{L}(N) = a^+c^*b^*$:



δ	a	b	c	ε
s_0	$\{s_0, s_1, s_2\}$	$\emptyset$	$\emptyset$	$\emptyset$
s_1	$\emptyset$	$\{s_1\}$	$\emptyset$	$\emptyset$
s_2	$\emptyset$	$\emptyset$	$\{s_2\}$	$\{s_1\}$

τ	a	b	c
s_0	$\{s_0, s_1, s_2\}$	$\emptyset$	$\emptyset$
s_1	$\emptyset$	$\{s_1\}$	$\emptyset$
s_2	$\emptyset$	$\{s_1\}$	$\{s_1, s_2\}$

Remoção de não determinismo

- A aceitação de uma cadeia por uma máquina não determinística depende da existência de um processamento que termina em um estado final.
- Em um NFA-ε podem existir vários caminhos que representam o processamento de uma cadeia.
  - Em um DFA este caminho é único.
- Para remover o não determinismo, o DFA resultante deve simular a exploração de todos os possíveis caminhos em um NFA-ε.

Remoção de não determinismo

Algoritmo 5: Constrói DFA equivalente a um NFA-ε

Entrada: NFA-ε $N = \langle \Sigma, S, s_0, \delta, F \rangle$ e função τ.
Saída: DFA $M = \langle \Sigma, S', s'_0, \delta', F' \rangle$.

$FIM \leftarrow F$;
$S' \leftarrow \mathcal{F}_\varepsilon(s_0)$;
repita
  se $\exists\, X \in S'$ *e* $a \in \Sigma$ *e* $\nexists$ *arco rotulado* a *saindo de* X **então**
    $Y \leftarrow \bigcup_{s_i \in X} \tau(s_i, a)$;
    se $Y \notin S'$ **então** $S' \leftarrow S' \cup \{Y\}$;
    Adicione um arco de X a Y rotulado de a;
  senão
    $FIM \leftarrow V$;
até $(FIM = V)$;
$s'_0 \leftarrow \mathcal{F}_\varepsilon(s_0)$;
$F' \leftarrow \{X \in S' \mid X$ contém um elemento $s_i \in F\}$;
retorna (M);

Remoção de não determinismo

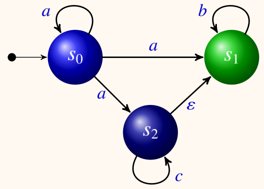
- Vértices do DFA são conjuntos de vértices do NFA-ϵ.
  - S' é o conjunto de vértices do DFA.
  - Vértice inicial do DFA é o Fecho ε do vértice inicial do NFA-ε.
- Y é o conjunto de todos os estados alcançáveis pelo processamento de um símbolo a partir de qualquer estado no conjunto X.

Remoção de não determinismo

- Algoritmo adiciona arcos ao DFA repetidas vezes.
  - À medida que os arcos são inseridos, novos vértices podem ser criados e inseridos no diagrama de estados do DFA.
  - Procedimento termina quando todos os vértices são determinísticos.
- No máximo $|\mathcal{P}(S)|$ vértices são construídos, já que cada vértice é um subconjunto de S.
  - O algoritmo sempre termina, um vez que $|\mathcal{P}(S)||\Sigma|$ é um limite superior para o número de iterações.

Remoção de não determinismo

Exemplo 1.62



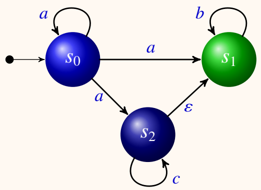
τ	a	b	c
s_0	$\{s_0, s_1, s_2\}$	$\emptyset$	$\emptyset$
s_1	$\emptyset$	$\{s_1\}$	$\emptyset$
s_2	$\emptyset$	$\{s_1\}$	$\{s_1, s_2\}$

$\mathcal{F}_\varepsilon(s_o) = \{s_0\}$

- Elementos do DFA:
  1. Arco rotulado de a do vértice $\{s_0\}$ para o $\{s_0, s_1, s_2\}$.
     - Transição a partir de s_0 lendo a termina em s_0, s_1 ou s_2.
  2. Vértice $\{s_0\}$ tem de ter arcos rotulados b e c saindo do mesmo.
  3. Um arco saindo do vértice $\{s_0, s_1, s_2\}$ termina no vértice com todos os estados alcançáveis (pelo processamento de um símbolo a partir dos estados s_0, s_1 ou s_2) no NFA-ε.

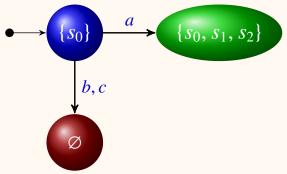
Remoção de não determinismo

Exemplo 1.62



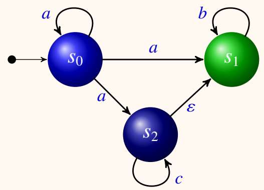
τ	a	b	c
s_0	$\{s_0, s_1, s_2\}$	$\emptyset$	$\emptyset$
s_1	$\emptyset$	$\{s_1\}$	$\emptyset$
s_2	$\emptyset$	$\{s_1\}$	$\{s_1, s_2\}$

$\mathcal{F}_\varepsilon(s_o) = \{s_0\}$



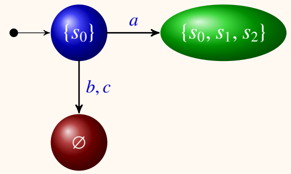
Remoção de não determinismo

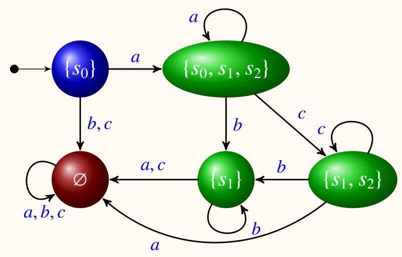
Exemplo 1.62



τ	a	b	c
s_0	$\{s_0, s_1, s_2\}$	$\emptyset$	$\emptyset$
s_1	$\emptyset$	$\{s_1\}$	$\emptyset$
s_2	$\emptyset$	$\{s_1\}$	$\{s_1, s_2\}$

$\mathcal{F}_\varepsilon(s_o) = \{s_0\}$





Equivalência entre DFA's e NFA's

Teorema 1.63

- *Seja $N = \langle \Sigma, S, s_0, \delta, F \rangle$ um NFA-ε e $M = \langle \Sigma, S', s'_0, \delta', F' \rangle$ o DFA obtido a partir de N com o algoritmo 5. Seja, ainda, $w \in \Sigma^*$ e $S_w = \{s_{w_1}, s_{w_2}, \ldots, s_{w_j}\}$ o conjunto de vértices alcançados, no NFA-ε N, ao término do processamento de w. Portanto, o processamento de w no DFA M termina no estado S_w.*

Equivalência entre DFA's e NFA's

Demonstração.

- Indução no comprimento da cadeia w:

Base: Se $|w| = 0$, o processamento em N termina em um vértice em $\mathcal{F}_\varepsilon(s_0)$. Este é o vértice inicial em M.

□

Equivalência entre DFA's e NFA's

Demonstração.

- Indução no comprimento da cadeia w:

Hipótese: Suponha que o resultado é válido para todas as cadeias de comprimento n.

□

Equivalência entre DFA's e NFA's

Demonstração.

- Indução no comprimento da cadeia w:

Passo:
- Seja $w = ua$, tal que $|w| = n + 1$.
- Seja $S_u = \{s_{u_1}, s_{u_2}, \ldots, s_{u_k}\}$ o conjunto de vértices finais obtido pelo processamento da cadeia u.
- Por hipótese de indução, o processamento de u em M termina no vértice S_u.

□

Equivalência entre DFA's e NFA's

Demonstração.

- Indução no comprimento da cadeia w:

Passo:

- Processamento de ua em M termina em estados (conjunto S_w) que podem ser alcançados, a partir de um estado em S_u, pelo processamento de a.
- S_w é definido por $S_w = \bigcup_{i=1}^{k} \tau(s_{u_i}, a)$.
- Como S_w é o estado alcançado a partir de S_u, pelo processamento de a, no DFA M, a prova fica completa.

$\square$

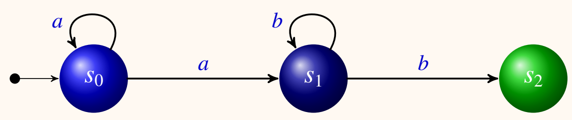
Equivalência entre DFA's e NFA's

Corolário 1.64

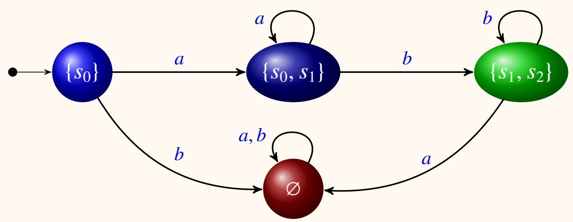
- *Se $M = \langle \Sigma, S', s'_0, \delta', F' \rangle$ é o DFA obtido a partir do NFA-ε $N = \langle \Sigma, S, s_0, \delta, F \rangle$ com o Algoritmo 5, então M e N são equivalentes.*

Remoção de não determinismo

Exemplo 1.65



δ	a	b
s_0	$\{s_0, s_1\}$	$\varnothing$
s_1	$\varnothing$	$\{s_1, s_2\}$
s_2	$\varnothing$	$\varnothing$



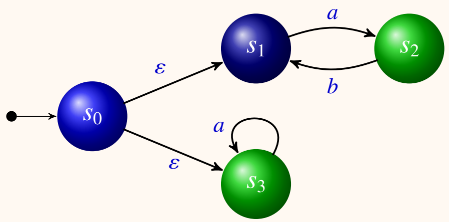
$\mathcal{F}_\varepsilon(s_0) = \{s_0\}$

τ	a	b
s_0	$\{s_0, s_1\}$	$\varnothing$
s_1	$\varnothing$	$\{s_1, s_2\}$
s_2	$\varnothing$	$\varnothing$

Remoção de não determinismo

Exemplo 1.66 ($\mathcal{L}(N) = a^* \cup a(ba)^*$)

- NFA-ε original:



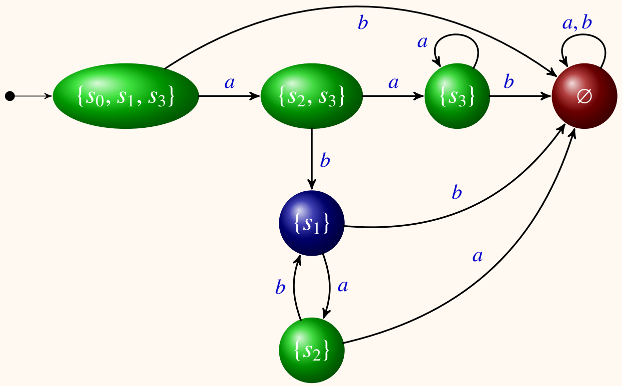
$\mathcal{F}_\varepsilon(s_0) = \{s_0, s_1, s_3\}$

τ	a	b
s_0	$\{s_2, s_3\}$	$\emptyset$
s_1	$\{s_2\}$	$\emptyset$
s_2	$\emptyset$	$\{s_1\}$
s_3	$\{s_3\}$	$\emptyset$

Remoção de não determinismo

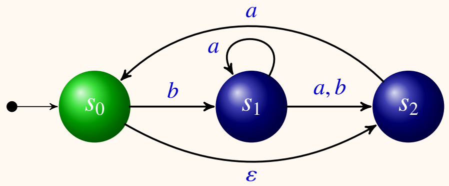
Exemplo 1.66 ($\mathcal{L}(N) = a^* \cup a(ba)^*$)

- DFA equivalente:



Remoção de não determinismo

Exemplo 1.67

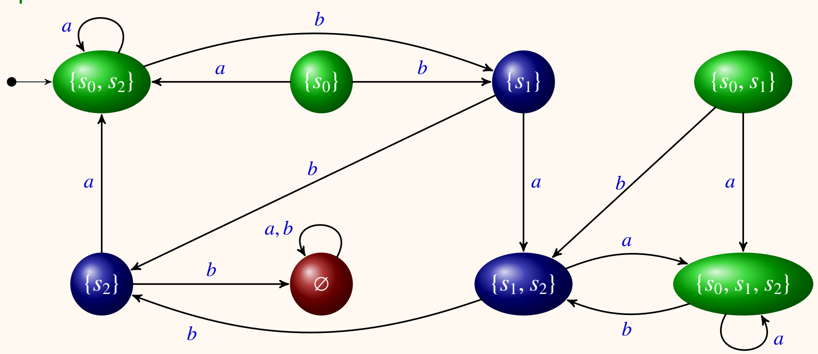


$\mathcal{F}_\varepsilon(s_0) = \{s_0, s_2\}$

τ	a	b
s_0	$\{s_0, s_2\}$	$\{s_1\}$
s_1	$\{s_1, s_2\}$	$\{s_2\}$
s_2	$\{s_0, s_2\}$	$\varnothing$

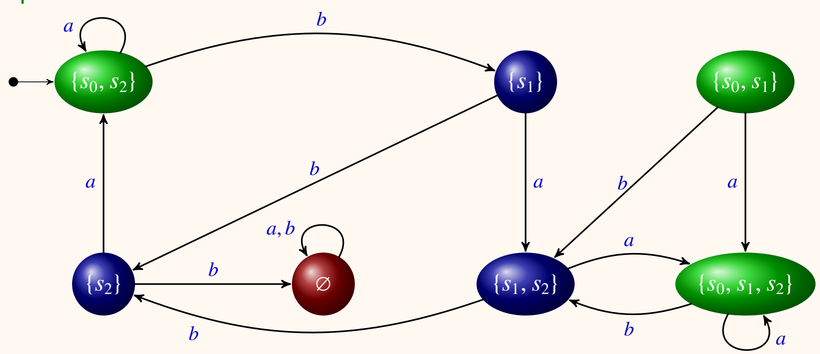
Remoção de não determinismo

Exemplo 1.67



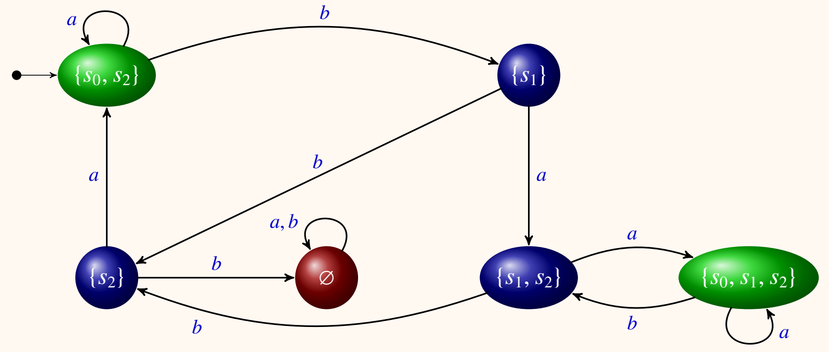
Remoção de não determinismo

Exemplo 1.67



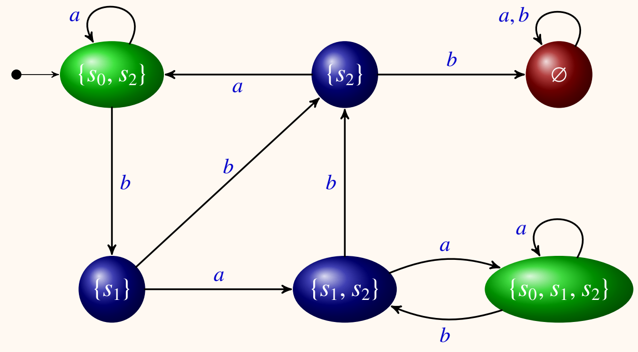
Remoção de não determinismo

Exemplo 1.67



Remoção de não determinismo

Exemplo 1.67



Livros texto

R. P. Grimaldi
Discrete and Combinatorial Mathematics – An Applied Introduction.
Addison Wesley, 1994.

D. J. Velleman
How To Prove It – A Structured Approach.
Cambridge University Press, 1996.

J. E. Hopcroft; J. Ullman.
Introdução Ã Teoria de Autômatos, Linguagens e Computação.
Ed. Campus.

T. A. Sudkamp.
Languages and Machines – An Introduction to the Theory of Computer Science.
Addison Wesley Longman, Inc. 1998.

J. Carroll; D. Long.
Theory of Finite Automata – With an Introduction to Formal Languages.
Prentice-Hall, 1989.

M. Sipser.
Introduction to the Theory of Computation.
PWS Publishing Company, 1997.

H. R. Lewis; C. H. Papadimitriou
Elementos de Teoria da Computação.
Bookman, 2000.