Linguagens Formais e Autômatos

Conversão de Expressões Regulares (ER) para Autômatos Finitos Determinísticos (AFD)

Cristiano Lehrer, M.Sc.

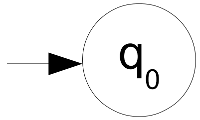
Introdução

- A construção sistemática de um Autômato Finito para reconhecer *strings* de uma linguagem regular é realizada em três etapas:
  - Construção de um autômato finito que representa diretamente os elementos de uma expressão regular.
    - Pela característica dessa construção, esse primeiro autômato é não determinístico.
    - Algoritmo de Thompson.
  - Conversão do autômato finito não determinístico para um autômato finito determinístico equivalente, ou seja, que reconheça *strings* da mesma linguagem.
    - Método da Construção de subconjuntos.
  - Reduzir, se possível, o número de estados do autômato.
    - Para tanto, o procedimento nessa etapa procura identificar estados que sejam redundantes e, se os encontra, os substitui por um único estado.
    - Minimização de estados do autômato.

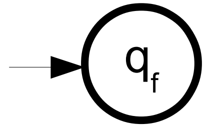
Algoritmo de Thompson (1/4)

- O **algoritmo de Thompson** define uma sequência de passos para, a partir de uma expressão regular, obter um autômato finito com movimentos vazios que reconheça sentenças da correspondente linguagem regular.

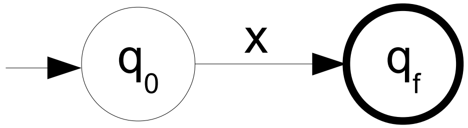
- ER = $\emptyset$



- ER = ε

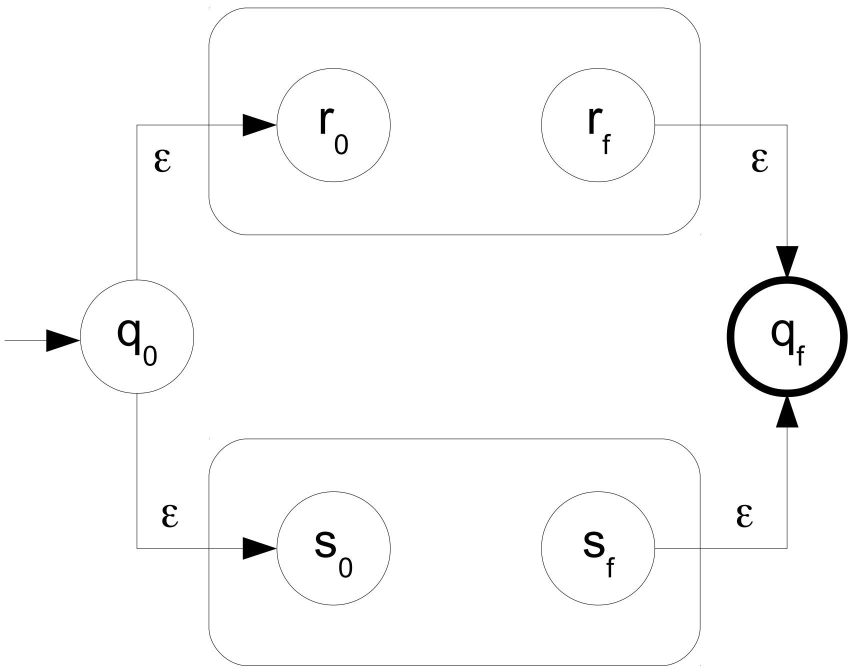


- ER = x



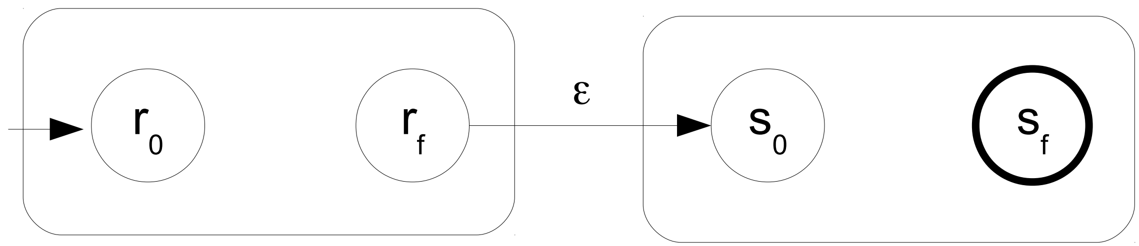
Algoritmo de Thompson (2/4)

- ER = r + s



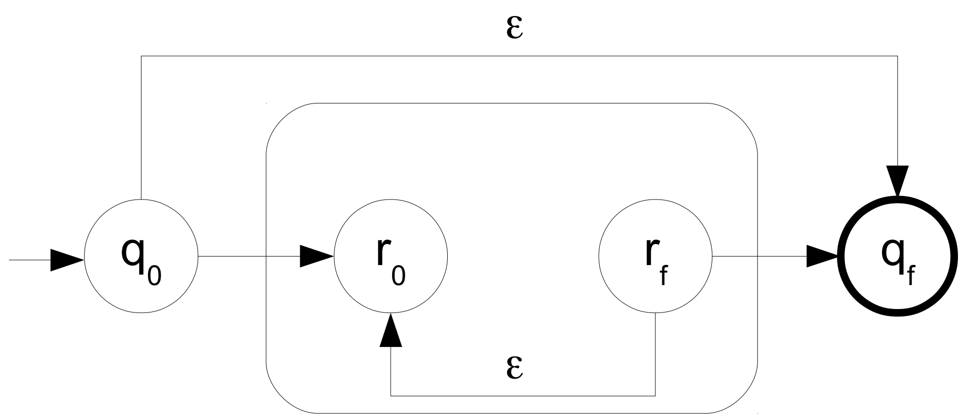
Algoritmo de Thompson (3/4)

- ER = rs



Algoritmo de Thompson (4/4)

- ER = r*



Método da Construção de Subconjuntos (1/4)

- Autômatos finitos com movimentos vazios (AFε) e autômatos finitos não determinísticos (AFN) não são muito práticos para realizar o reconhecimento de *strings*.
- Para realizar o reconhecimento automático da *string* de forma mais direta, é mais interessante utilizar um autômato finito determinístico (AFD).
- Existe um procedimento sistemático para transformar um AFε num AFD que aceita a mesma linguagem:
  - O procedimento apresentado frequentemente é denominado **método da construção de subconjuntos**.

Método da Construção de Subconjuntos (2/4)

- O princípio associado a esse método é criar novos estados, no AFD, que estejam associados a todas as possibilidades de estados originais em um dado momento da análise da sentença no processo de reconhecimento.
- Para cada estado original, essas possibilidades incluem o estado do AFε e todos os estados que podem ser atingidos a partir dele com transições pela *string* vazia.
- Esse conjunto de estados do AFε é definido pela operação ε* (lê-se épsilon-clausura):
  - A aplicação da ε* a um conjunto de estados resulta no conjunto que inclui, além dos próprios estados, cada um dos demais estados do autômato que podem ser alcançados a partir desses com transições pela *string* vazia.

Método da Construção de Subconjuntos (3/4)

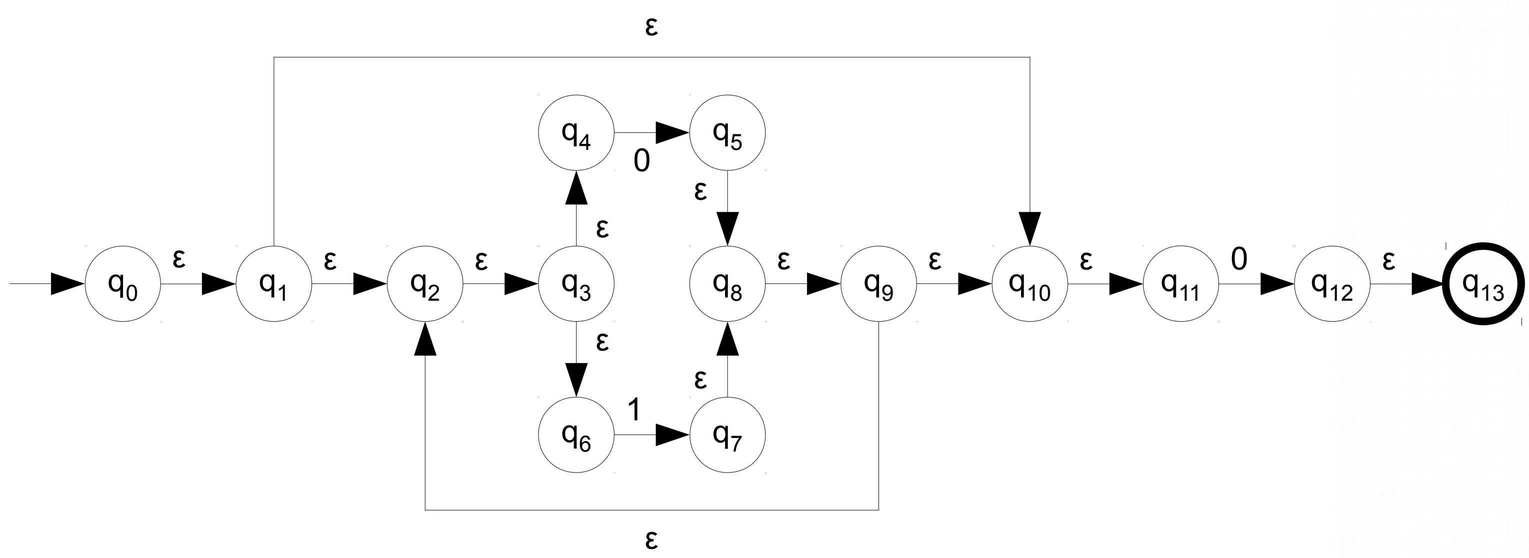
- A aplicação do método da construção de subconjuntos começa pela computação da ε^* do conjunto que contém apenas o estado inicial do AFε.
- O conjunto de estados resultante representa um único estado no novo AFD.
- Como esse estado inclui o estado inicial do autômato original, será também o estado inicial do novo autômato.
- Do mesmo modo, se o estado final do autômato original for elemento desse conjunto, o novo estado será também um estado final no novo autômato.
- Cada novo estado que é gerado dessa maneira é incluído numa lista de estados a analisar.

Método da Construção de Subconjuntos (4/4)

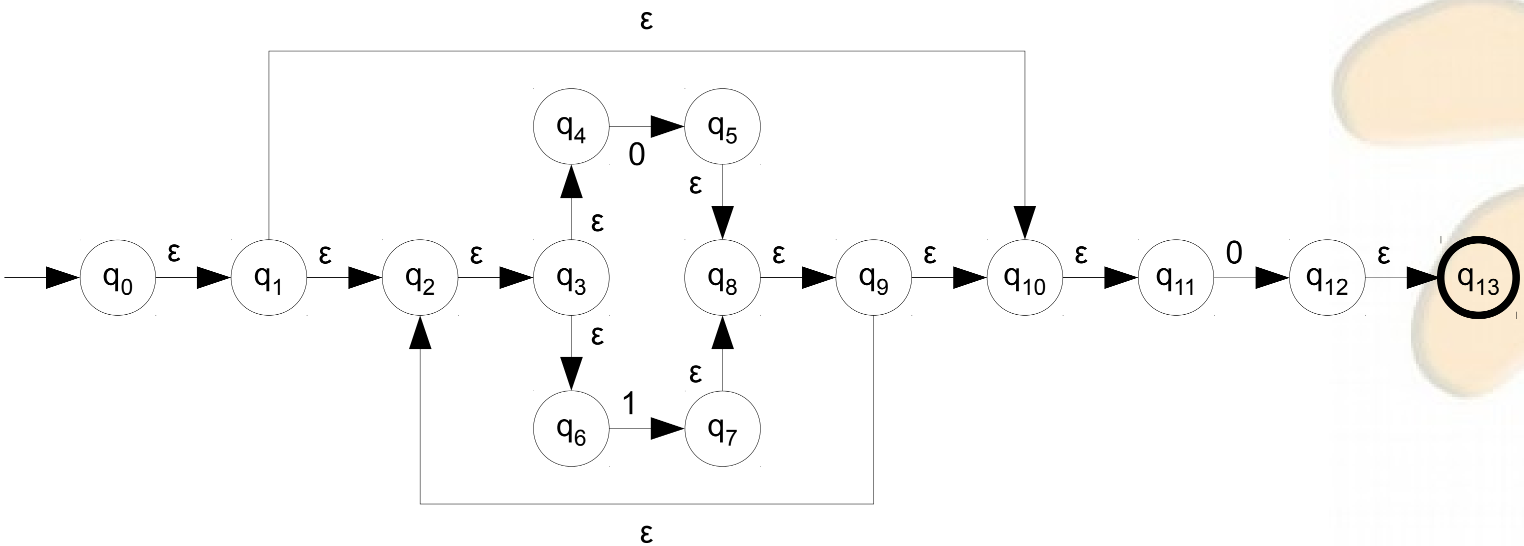
- O procedimento prossegue com a análise de estados ainda não analisados:
  - O objetivo dessa análise é verificar, para cada símbolo do alfabeto, se há transição possível a partir do estado sob análise e, se houver transição, para qual estado ele leva.
  - A transição por um símbolo α do alfabeto será possível se houver, no conjunto de estados originais associado ao estado analisado, pelo menos um elemento que tenha a transição pelo mesmo símbolo no autômato original.
  - Se houver, o novo estado resultante da transição será a ε^* do conjunto de estados que resulta da transição por esse símbolo no AFε.

Passo a Passo (1/3)

- Expressão Regular (ER):
  - `(0 + 1)* 0`
- Autômato Finito com Movimentos Vazios (AFε):
  - $(\{0, 1\}, \{q_0, q_1, ..., q_{12}, q_{13}\}, \delta, q_0, \{q_{13}\})$



Passo a Passo (2/3)



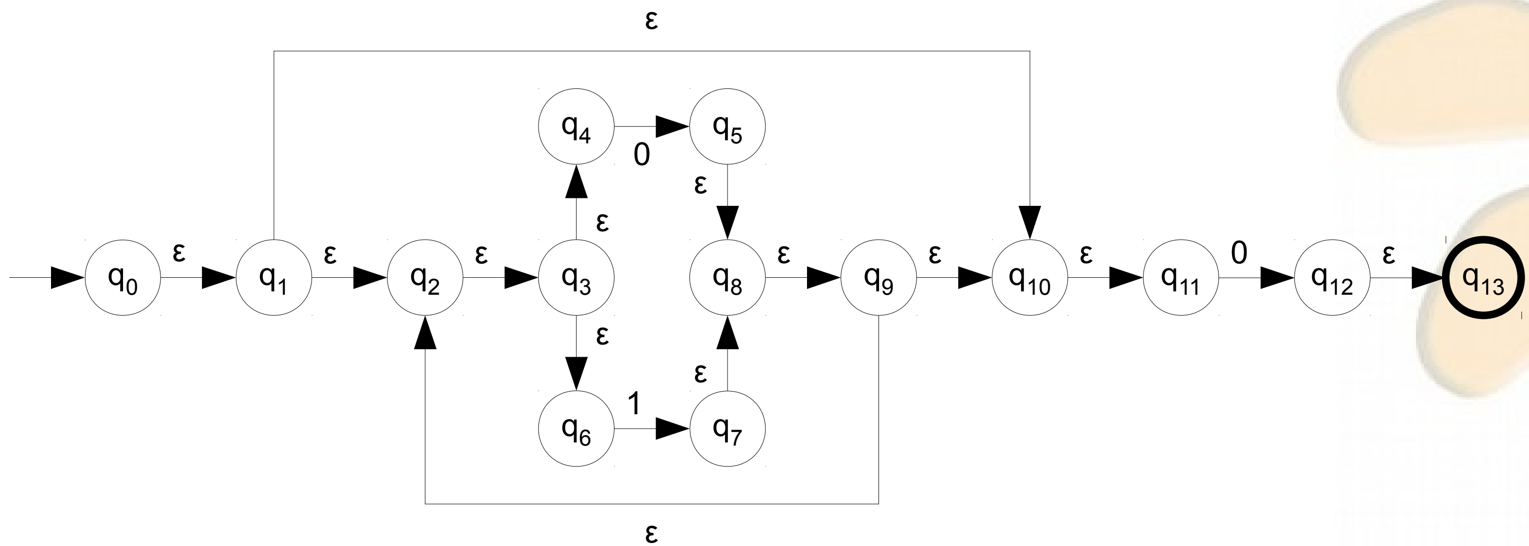
$\varepsilon^* \{q_0\} = \{q_0, q_1, q_2, q_3, q_4, q_6, q_{10}, q_{11}\} \rightarrow (s_0)$

s_0	q_0	q_1	q_2	q_3	q_4	q_6	q_{10}	q_{11}
0	-	-	-	-	q_5	-	-	q_{12}
1	-	-	-	-	-	q_7	-	-

$\varepsilon^* \{q_5, q_{12}\} = (s_1)$

$\varepsilon^* \{q_7\} = (s_2)$

Passo a Passo (2/3)



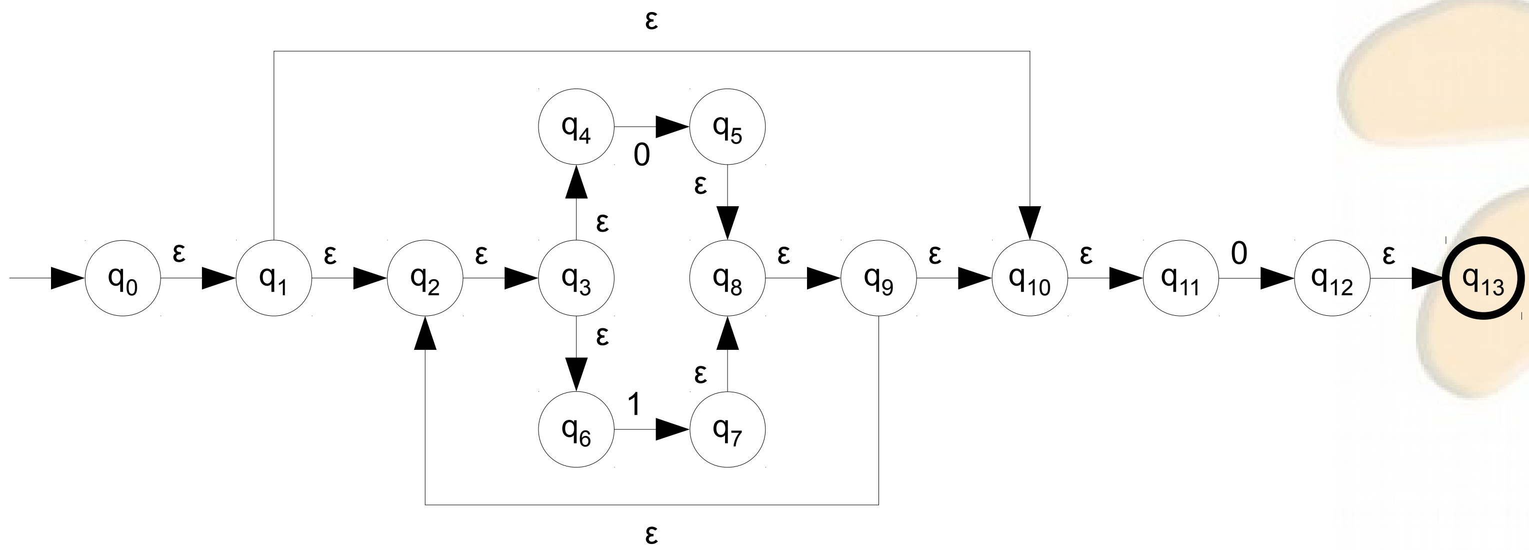
$\varepsilon^* \{q_5, q_{12}\} = \{q_2, q_3, q_4, q_5, q_6, q_8, q_9, q_{10}, q_{11}, q_{12}, q_{13}\} \rightarrow (s_1)$

s_1	q_2	q_3	q_4	q_5	q_6	q_8	q_9	q_{10}	q_{11}	q_{12}	q_{13}
0	-	-	q_5	-	-	-	-	-	q_{12}	-	-
1	-	-	-	-	q_7	-	-	-	-	-	-

$\varepsilon^* \{q_5, q_{12}\} = (s_1)$

$\varepsilon^* \{q_7\} = (s_2)$

Passo a Passo (2/3)



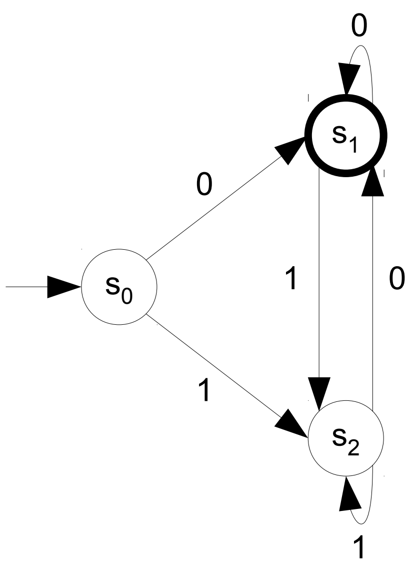
$\varepsilon^* \{q_7\} = \{q_2, q_3, q_4, q_6, q_7, q_8, q_9, q_{10}, q_{11}\} \rightarrow (s_2)$

s_2	q_2	q_3	q_4	q_6	q_7	q_8	q_9	q_{10}	q_{11}	
0	-	-	q_5	-	-	-	-	-	q_{12}	$\varepsilon^* \{q_5, q_{12}\} = (s_1)$
1	-	-	-	q_7	-	-	-	-	-	$\varepsilon^* \{q_7\} = (s_2)$

Passo a Passo (3/3)

- Autômato Finito Determinístico (AFD):
  - $(\{0, 1\}, \{s_0, s_1, s_2\}, \delta, s_0, \{s_1\})$

δ	s_0	s_1	s_2
0	s_1	s_1	s_1
1	s_2	s_2	s_2



Minimização de Estados do Autômato (1/4)

- O procedimento é baseado na construção iterativa de partições do conjunto `K` de estados do autômato.
- A cada iteração, o objetivo é identificar se há um comportamento diferenciado entre os estados que fazem parte de uma partição, ou seja, é preciso analisar o que ocorre nas transições associadas a estados dessa partição.
- Se há essa diferença de comportamento, então os estados não são redundantes e novas partições são criadas a partir dessa.
- Caso contrário, se todos os estados numa partição têm exatamente o mesmo comportamento, então os estados são redundantes e podem ser combinados, num autômato minimizado, em um único estado.
- Se uma partição tem um único estado, então aquele estado não era redundante no autômato original.

Minimização de Estados do Autômato (2/4)

- O primeiro particionamento do conjunto de estados do autômato reflete a diferença entre estados que são finais e estados que não são finais.
- Se F representa o conjunto de estados finais do autômato, então essa primeira iteração cria uma partição $P1 = \{C_1, C_2\}$ com dois subconjuntos, um com os estados finais, $C_1 = F$, e outro com os demais estados, $C_2 = K - F$.
- Se uma partição P_i contém entre seus elementos um subconjunto não-unitário, então esse subconjunto deve ser analisado para descobrir se seus estados são ou não redundantes.

Minimização de Estados do Autômato (3/4)

- Seja esse subconjunto C_i:
  - Para cada estado de C_i, é preciso verificar para qual subconjunto da partição P_i as transições pelos símbolos do alfabeto levam.
  - Se dois ou mais estados têm transições levando, a partir de cada símbolo, a subconjuntos distintos, então seus comportamentos são distintos e esses estados deverão, na próxima partição P_{i+1}, estar em subconjuntos distintos.
  - Caso contrário, se não houver nenhuma diferença de comportamento entre os estados do subconjunto C_i, então seus elementos são redundantes e podem ser representados por um único estado no autômato minimizado.

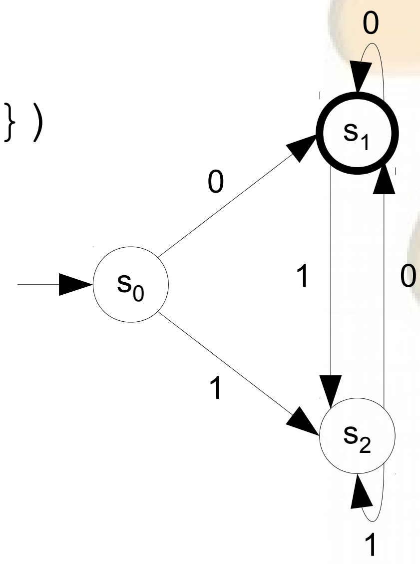
Minimização de Estados do Autômato (4/4)

- O refinamento da partição do conjunto de estados deve continuar até que não haja mais possibilidades de particionar nenhum subconjunto da partição, seja porque o subconjunto é redundante, seja porque é unitário.
- O autômato que resulta desse procedimento de minimização terá um estado associado a cada subconjunto da última partição que foi obtida.

Passo a Passo (1/2)

- Autômato Finito Determinístico (AFD):
  - $(\{0, 1\}, \{s_0, s_1, s_2\}, \delta, s_0, \{s_1\})$

δ	s_0	s_1	s_2
0	s_1	s_1	s_1
1	s_2	s_2	s_2



- $P_1 = \{C_0, C_1\}$, com $C_0 = \{s_1\}$ e $C_1 = \{s_0, s_2\}$

C_0	s_1
0	C_0
1	C_1

C_1	s_0	s_2
0	C_0	C_0
1	C_1	C_1

Passo a Passo (2/2)

- Autômato Finito Determinístico (AFD) Minimizado:
  - $(\{0, 1\}, \{C_0, C_1\}, \delta, C_1, \{C_0\})$

C_1	s_0	s_2
0	C_0	C_0
1	C_1	C_1