

Propriedades de

Linguagens Regulares

Lema do Bombeamento

Operações Fechadas sobre LR's

Aplicações

(H&U, 1969), (H&U, 1979),

(H;M;U, 2001) e (Menezes, 2002)

a^n



Lema do Bombeamento para LR

- Como decidir que uma linguagem é ou não regular?
 - Não bastaria não conseguir exibir um AF ou uma ER
- Toda linguagem regular satisfaz o Lema do Bombeamento (LB).
- Se alguém apresenta a você uma falsa LR, use o LB para mostrar a contradição, pois ela não vai satisfazê-lo.

- O Lema do Bombeamento (Pumping Lemma) nos diz que
 - “ qualquer cadeia suficientemente longa w de uma LR pode ser decomposta em 3 partes:
 $w = xyz$, de maneira que podemos construir outras cadeias da linguagem pela repetição da parte central y . ”
- Isto é, todas as cadeias da forma xy^kz estão também na linguagem. Ou seja, podemos acionar a *bomba* quantas vezes quisermos, para criar quantas sentenças novas da linguagem desejarmos: $xz, xyz, xyyz, xyyyz, \dots$

Mostrando que uma Linguagem não é Regular

- Para mostrar que uma linguagem **não** é regular, mostramos que
 - não há como decompor uma cadeia (qualquer, arbitrariamente longa) da linguagem de forma que seja possível *bombear* e o resultado continuar na linguagem.

Lema do Bombeamento

Se L é uma linguagem regular, então:
existe uma constante natural n (que depende de L) tal que qualquer cadeia w de L , com $|w| \geq n$, pode ser decomposta em três subcadeias x, y, z ($w=xyz$) de forma que:

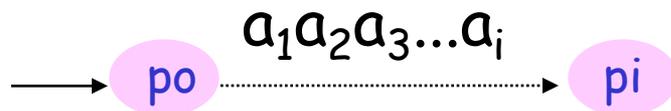
- $|xy| \leq n$
- $y \neq \lambda$ (isto é, $|y| \geq 1$) e
- para qualquer $k \geq 0$, $xy^kz \in L$.

- **Demonstração (simplificada):** Baseia-se no fato de que, para as **cadeias longas w** , é necessário usar pelo menos um loop de estados num AFD que aceite a linguagem.
- Assim, os símbolos de **x** são usados para chegarmos a um estado **q** do loop;
- os símbolos de **y** são usados para dar a volta no loop, de volta ao estado **q** ;
- os símbolos de **z** são usados para ir de **q** até um estado final.
- Portanto, podemos dar quantas voltas no loop quisermos, e repetir **y** um número qualquer **k** de vezes: **xy^kz** .

As **cadeias curtas (comprimento $< n$)** não são consideradas porque podem ser aceitas sem passar por qualquer loop.

Prova formal do Lema do Bombeamento

Suponha que L seja regular. Então existe um AFD A , com n estados, que a reconhece. Considere qualquer string w de comprimento n ou maior: $w = a_1 a_2 a_3 \dots a_m$, onde $m \geq n$, e cada a_i é um símbolo de entrada. Para $i = 0, 1, \dots, n$, defina o estado p_i como $\delta(q_0, a_1 a_2 a_3 \dots a_i)$. Isto é, p_i é o estado em que A se encontra depois de ler os primeiros i símbolos de w . Note que $p_0 = q_0$.

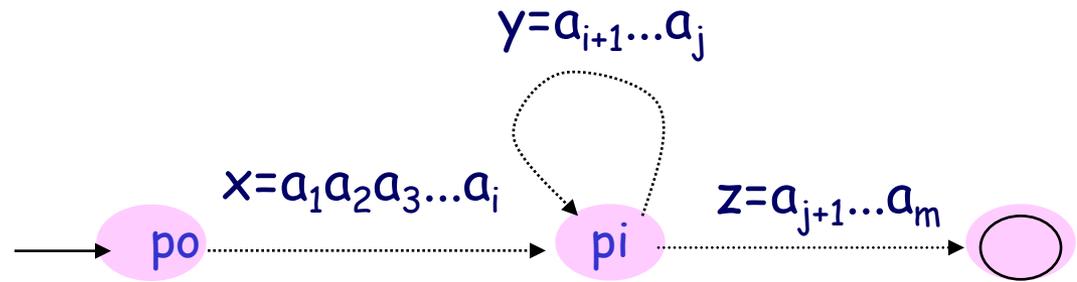


Como só existem n estados, não é possível que os $n+1$ diferentes p_i , para $i=0, 1 \dots n$ sejam distintos. Desse modo, podemos encontrar dois inteiros diferentes, i e j , com $0 \leq i < j \leq n$, tais que $p_i = p_j$. Agora, podemos dividir $w = xyz$ como a seguir:

1. $x = a_1 a_2 a_3 \dots a_i$

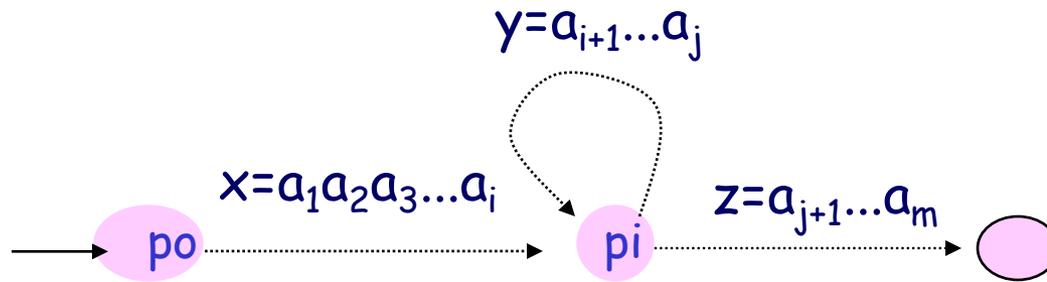
2. $y = a_{i+1} a_{i+2} \dots a_j$

3. $z = a_{j+1} a_{j+2} \dots a_m$



Observe que x pode ser vazio ($i=0$) e z pode ser vazio, se $j=n=m$. Mas y não pode ser vazio, pois i é estritamente menor que j . Também é verdade que $|xy| \leq n$. Falta verificar a última condição:

Vejam os o que acontece com A para entradas xy^kz para qualquer $k \geq 0$:



- Se $k=0$ ($w=xz$): A vai de p_0 para p_i na entrada x . Tendo em vista que $p_i=p_j$, A deve ir de p_i para o estado final, para a entrada z . Desse modo, A aceita xz .
- Se $k > 0$ ($w=xy^kz$): A vai de p_0 a p_i sobre a entrada x , circula de p_i para p_i k vezes para a entrada y^k , e depois vai para o estado de aceitação para a entrada z . Dessa forma, para qualquer $k \geq 0$, xy^kz também é aceito por A; ou seja, xy^kz está em L . $\square \square \quad \square$

Exemplo provando que L não é regular

$$L = \{a^i b^i \mid i \geq 0\}$$

Considere a cadeia $w = a^i b^i$. Qualquer decomposição $w = xyz$ deve ter em y o mesmo número de a 's e de b 's, para que a propriedade de que o número de a 's é igual ao de b 's se mantenha nas cadeias $x y^k z$.

- Se isso não acontecer, quando acrescentarmos mais um y (aumentando k de 1), obteremos uma cadeia fora da linguagem. Portanto, y deve ser da forma $a^j b^j$, com $j > 0$, já que y não pode ser vazia.
- Mas nesse caso, $x y^2 z$ conterá a cadeia $a^j b^j a^j b^j$, com pelo menos um a e depois um b , o que não pode acontecer na linguagem.
- Ou seja, nenhuma decomposição é possível, contrariando o Lema, e podemos concluir que L não é regular.

Exemplo: provando que uma LR obedece o LB

JFLAP : <untitled1>

File Input Test Convert Help

Editor Simulate: abbbba

```
graph LR; start(( )) --> q0((q0)); q0 -- a --> q1((q1)); q1 -- b --> q2((q2)); q2 -- b --> q1; q2 -- a --> q3(((q3)));
```

q0

abbbba

$L = ??$

Step Reset Freeze Thaw Trace Remove

Iniciar aut_er_1.ppt Bombeamento.ppt SCE_521_185 JFLAP : <untitled1> PT 20:02

- Seja $L = \{ab^n a \mid n \geq 1 \text{ e ímpar}\}$ ou $\{ab^{2n+1} a \mid n \geq 0\}$
- Pelo Lema do Bombeamento, temos que:
- seja $n = 4$ (número de estados do AF)
- Para $w = abbba$ $|w| = 5 \geq 4$

Sejam:

- $x = ab$
- $y = bb \quad |xy| \leq 4; |y| \geq 1$
- $z = a$

Para todo $k \geq 0$, $ab(bb)^k a \in L$

Exercícios

Mostre que as linguagens abaixo não são regulares:

1) $L = \{ xx^r \mid x \in \{0,1\}^* \}$

2) $L = \{ 0^n 1 0^n \mid n \geq 1 \}$

3) $L = \{ 0^n 1^m 2^n \mid n \text{ e } m \text{ são inteiros quaisquer} \}$

4) $L = \{ 0^n 1^m \mid n \leq m \}$

5) $L = \{ 0^n 1^{2^n} \mid n \geq 1 \}$

Operações que preservam a propriedade de ser uma LR

- Existem muitas operações que, quando aplicadas a LR, resultam em uma LR (dizemos que são **fechadas**).
- Entre elas temos: união, complemento e intersecção (operações booleanas), concatenação, fechamento, diferença e reverso ($a_1a_2\dots a_n \rightarrow a_n\dots a_2a_1$).
- Elas são úteis também como ferramentas de construção de autômatos.

Propriedade de Fechamento das Linguagens da Hierarquia de Chomsky

operação (op)	LR op LR		
união	LR		
concatenação	LR		
fechamento	LR		
complemento	LR		
intersecção	LR		
diferença	LR		
reverso	LR		

Lema 3.1 (H&U, 69) - União

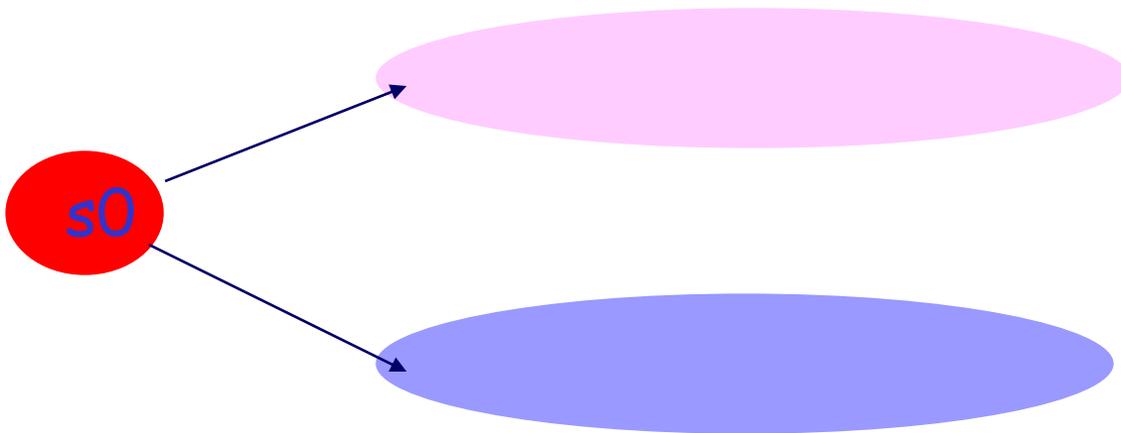
- A classe das LR é fechada sobre a união. Se $L1$ é LR e $L2$ é LR então $L1 \cup L2$ também é.
- (Prova via ER: direta)
- **PROVA:** Sejam $L1$ e $L2$ reconhecidas por AF $M1 = (Q1, \Sigma1, \delta1, qo, F1)$ e $M2 = (Q2, \Sigma2, \delta2, ro, F2)$. Suponha $Q1 \cap Q2 = \emptyset$ e $so \notin Q1; so \notin Q2$.

$M3 = (Q1 \cup Q2 \cup \{so\}, \Sigma1 \cup \Sigma2, \delta3, so, F)$ é um AFND onde:

- Se $\lambda \in L1$ ou $L2$ então $F = F1 \cup F2 \cup \{s0\}$ cc.
 $F = F1 \cup F2$

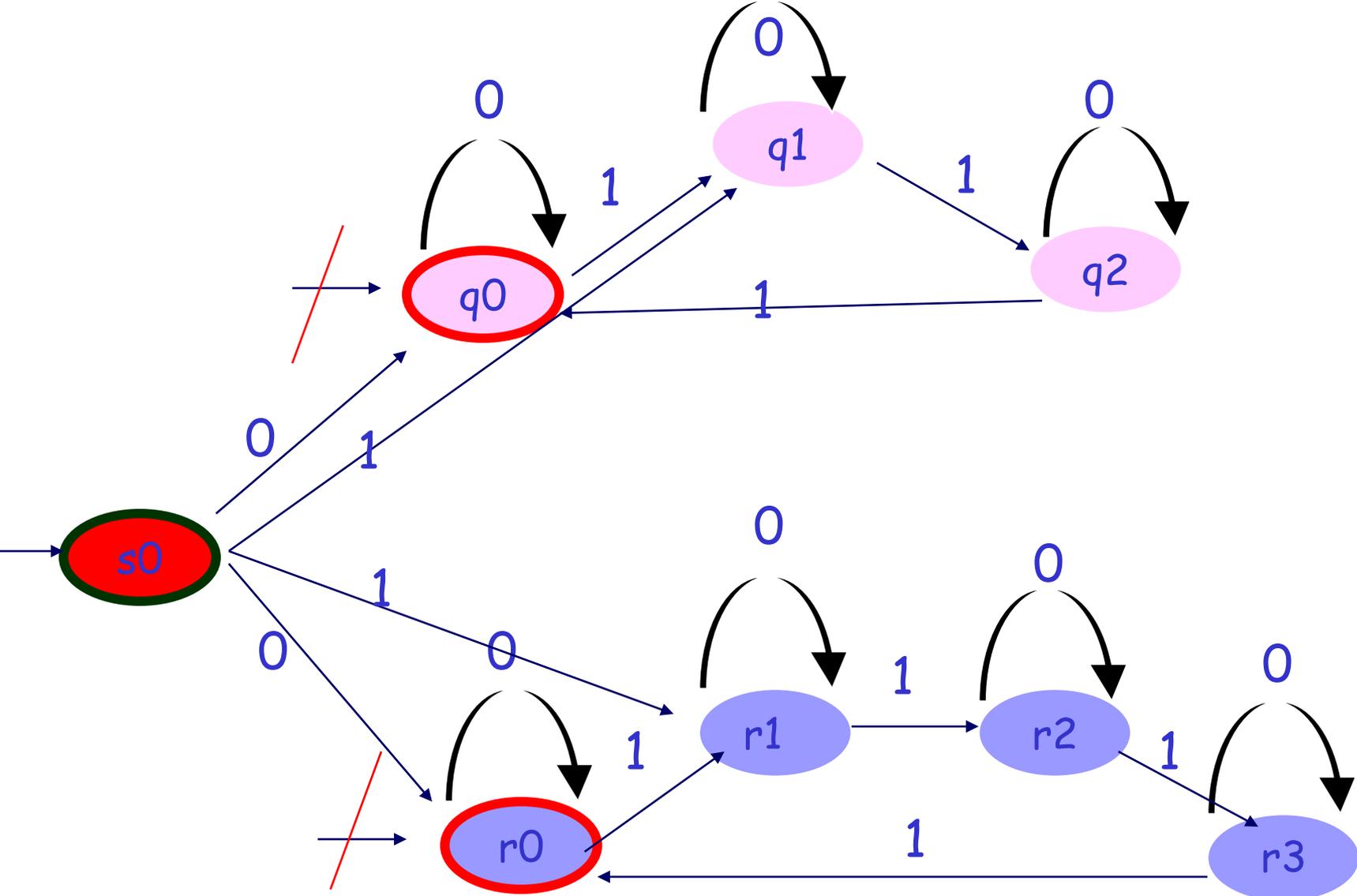
- $\delta3$ {
 - 1) $\delta3(s0, a) = \{\delta1(q0, a), \delta2(r0, a)\}$ para todo $a \in \Sigma1 \cup \Sigma2$
 - 2) $\delta3(q, a) = \delta1(q, a)$ para todo $q \in Q1$ e $a \in \Sigma1$
 - 3) $\delta3(q, a) = \delta2(q, a)$ para todo $q \in Q2$ e $a \in \Sigma2$
 } Continuam os mesmos

$$L(M3) = L(M1) \cup L(M2)$$



Exemplo

- Use o Lema 3.1 para fazer um AFND que reconheça $L(M) = \{ x \in \{0,1\}^* \mid \text{o nro de 1's em } x \text{ é múltiplo de 3 OU de 4} \}$

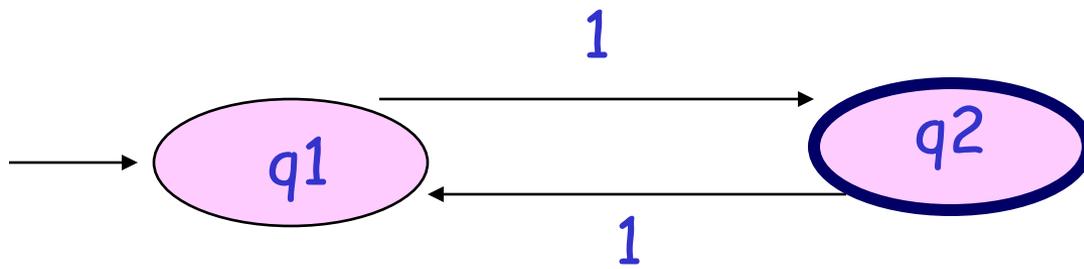


M3 = ?

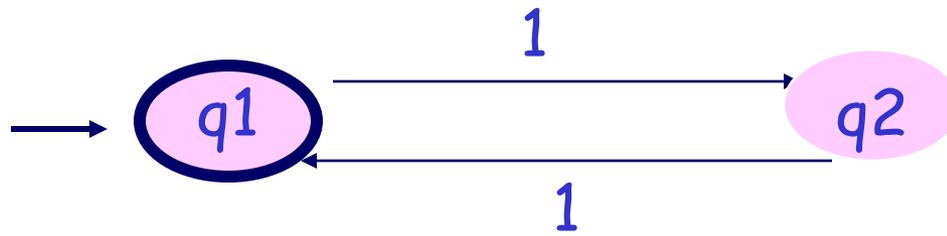
Lema 3.2 (H&U, 69) - Complemento

- A classe dos conjuntos aceitos por um AF é fechada sobre o complemento (Se L é uma LR então \bar{L} também é).
- **Prova:** $M1 = (Q, \Sigma1, \delta1, q0, F)$ é um AF que aceita um conjunto $L1$. Seja $\Sigma2$ um alfabeto contendo $\Sigma1$ (pode ser o mesmo) e d um estado $\notin Q$. $M2$ aceita $\Sigma2^* - L1$.
- $M2 = (Q \cup \{d\}, \Sigma2, \delta2, q0, (Q - F) \cup \{d\})$

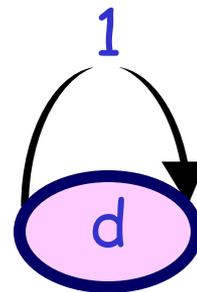
- $\delta_2(q,a) = \delta_1(q,a)$ para cada $q \in Q$ e $a \in \Sigma_1$
(continua o anterior)
- $\delta_2(q,a) = d$ para cada $q \in Q$ e $a \in (\Sigma_2 - \Sigma_1)$
do particular estado
- $\delta_2(d,a) = d$ para cada $a \in \Sigma_2$
- **Exemplo:** Use o Lema 3.2 para reconhecer o complemento de $L(M_1) = \{x \in \{1\}^* \mid x \text{ possui um nro ímpar de } 1\text{'s}\}$.



M1: reconhece
ímpar



M2: Reconhece
par



D é um estado
inacessível que
pode ser
eliminado.
Portanto M pode
ser reduzido

Teo 3.6 (H&U, 69) - Intersecção

- A classe das LR é fechada sobre a intersecção. Se $L1$ é LR e $L2$ é LR então $L1 \cap L2$ também é.
- Prova: dos Lemas 3.1 e 3.2 e da Lei de De Morgan, desde que a própria **intersecção** usa operações de união e complemento.

$$L1 \cap L2 = \overline{\overline{L1} \cup \overline{L2}}$$

- A classe dos conjuntos aceitos por um AF forma uma Álgebra de Boole. Isto é, é uma coleção de conjuntos fechados sobre a união, complemento e intersecção.

Teo 3.8 (H&U, 69) - Concatenação

- A classe dos conjuntos aceitos por um AF é fechada sobre a concatenação. Se L_1 é LR e L_2 é LR, então $L_1.L_2$ é LR.
- (Prova via ER: direta)
- Prova: Seja $M_1 = (Q_1, \Sigma_1, \delta_1, q_1, F_1)$ aceitando L_1 e $M_2 = (Q_2, \Sigma_2, \delta_2, q_2, F_2)$ aceitando L_2 .
- Assumimos Q_1 e Q_2 disjuntos e $\Sigma = \Sigma_1 = \Sigma_2$.
- $M_3 = (Q_1 \cup Q_2, \Sigma, \delta_3, q_1, F)$ é um AFND onde:

- $\delta_3(q,a) = \{\delta_1(q,a)\}$ para cada $q \in (Q_1 - F_1)$ e $a \in \Sigma$.

M_3 age como M_1 para o começo da cadeia possivelmente vazia

- $\delta_3(q,a) = \{\delta_1(q,a), \delta_2(q_2,a)\}$ para cada $q \in F_1$ e $a \in \Sigma$.

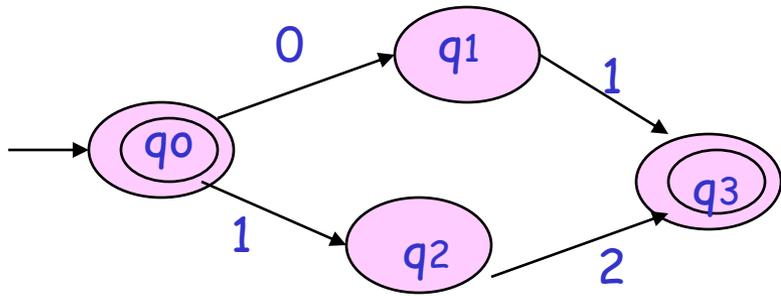
M_3 continua em M_1 ou vai para M_2

- $\delta_3(q,a) = \{\delta_2(q,a)\}$ para cada $q \in Q_2$. M_3 age como M_2 depois que a cadeia de entrada pertence a L_2 .

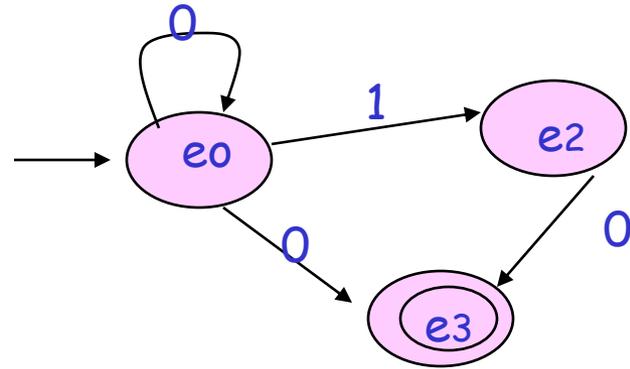
- Se $\lambda \notin L_2$ então $F = F_2$ (vai aceitar em M_2)

- Se $\lambda \in L_2$ então $F = F_1 \cup F_2$ (aceita também cadeias de L_1).

Exemplo

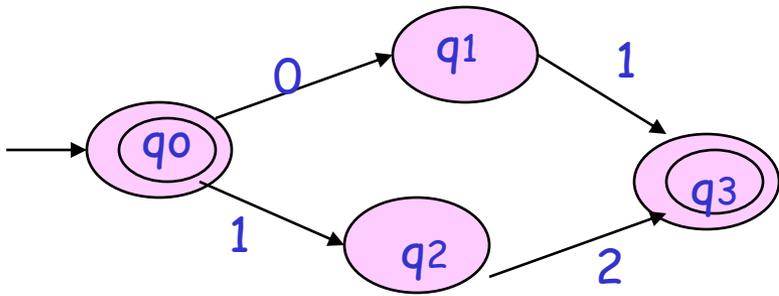


$L1 = \{01, 12, \lambda\}$

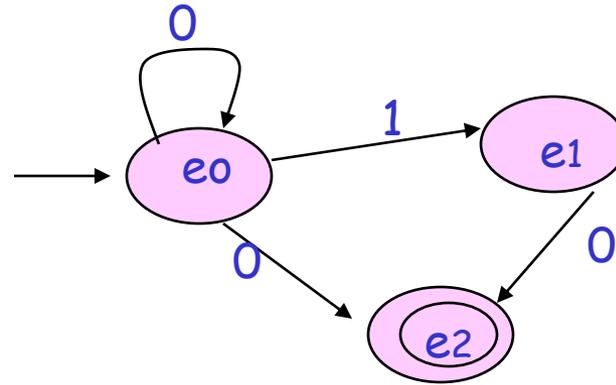


$L2 = 0^*(10+0)$

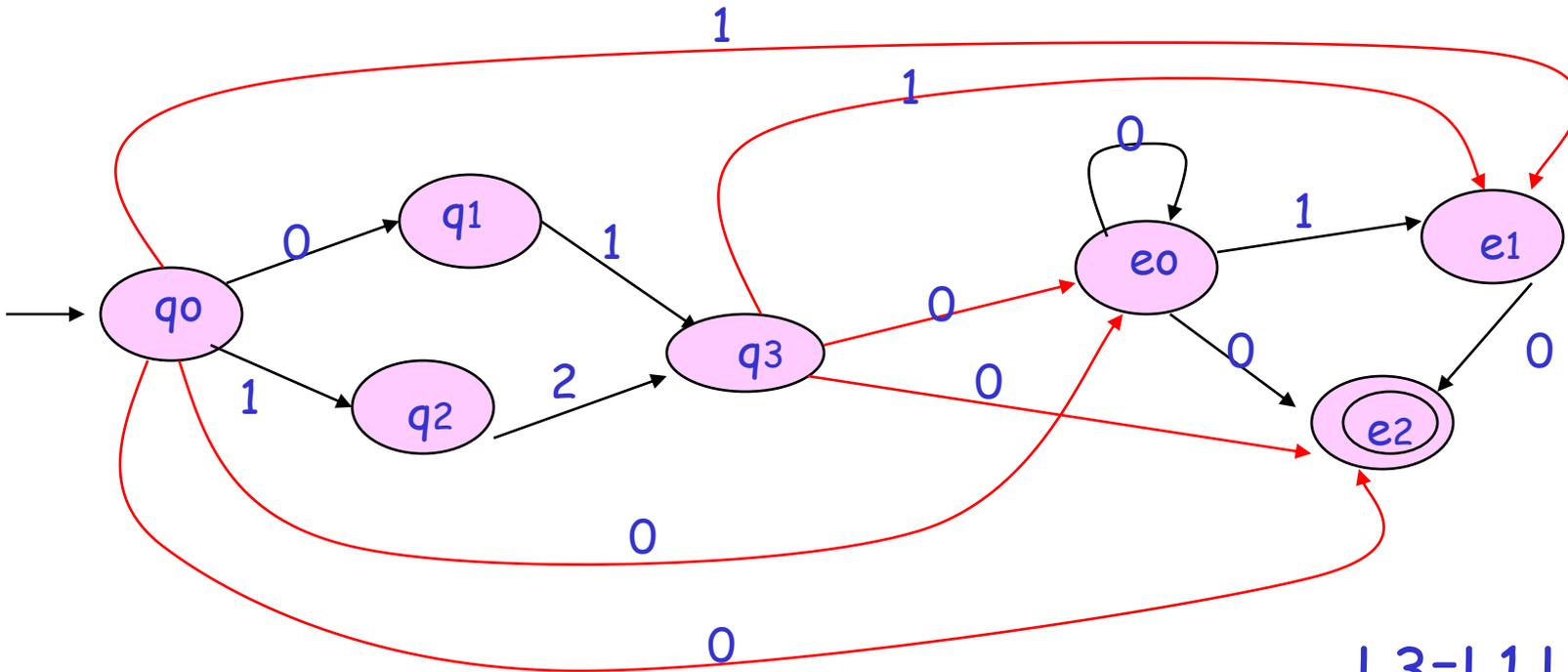
Exemplo



$L1 = \{01, 12, \lambda\}$



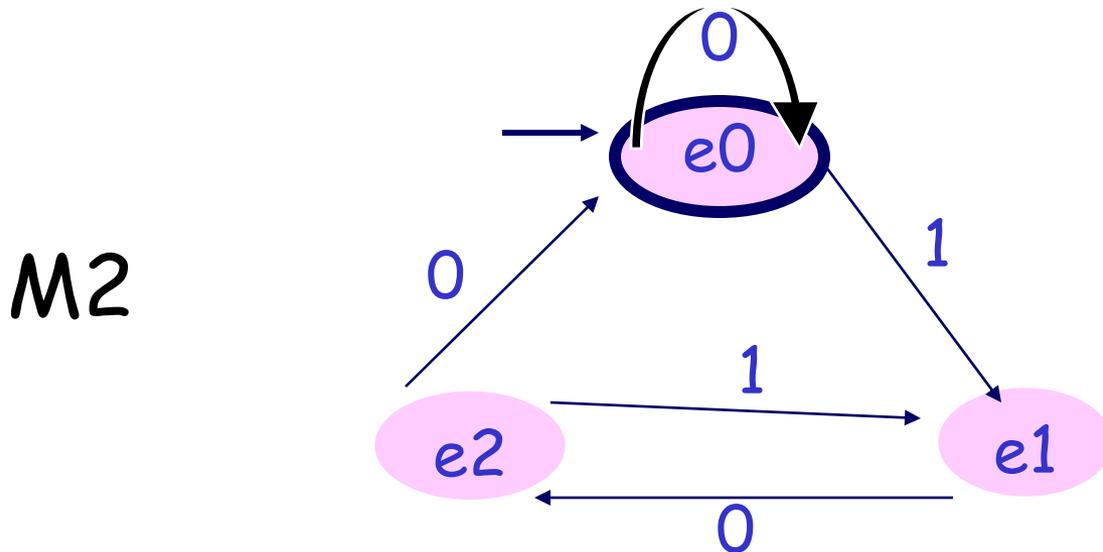
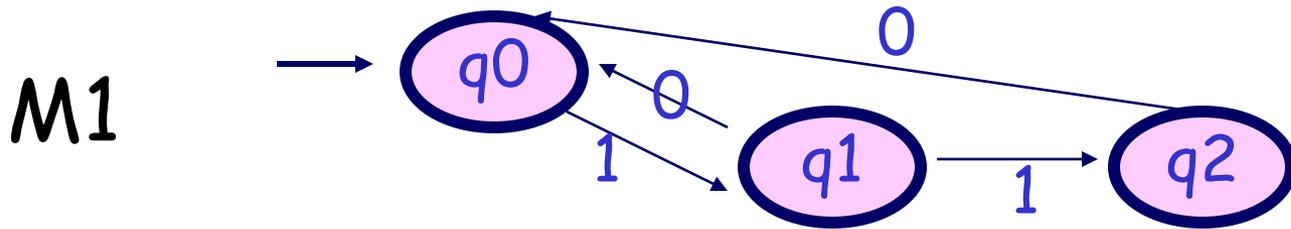
$L2 = 0^*(10+0)$

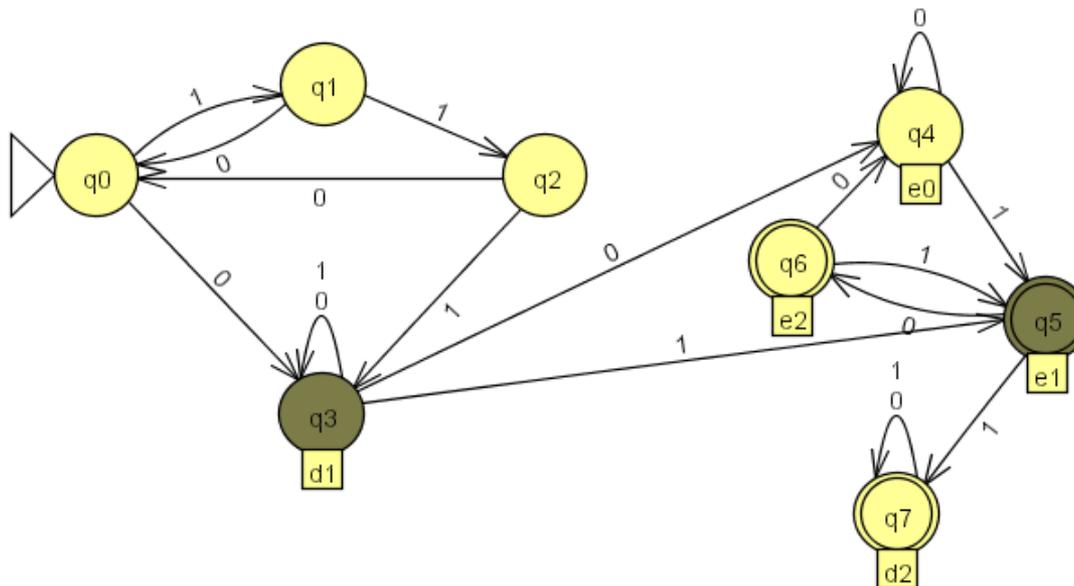


$L3 = L1.L2$

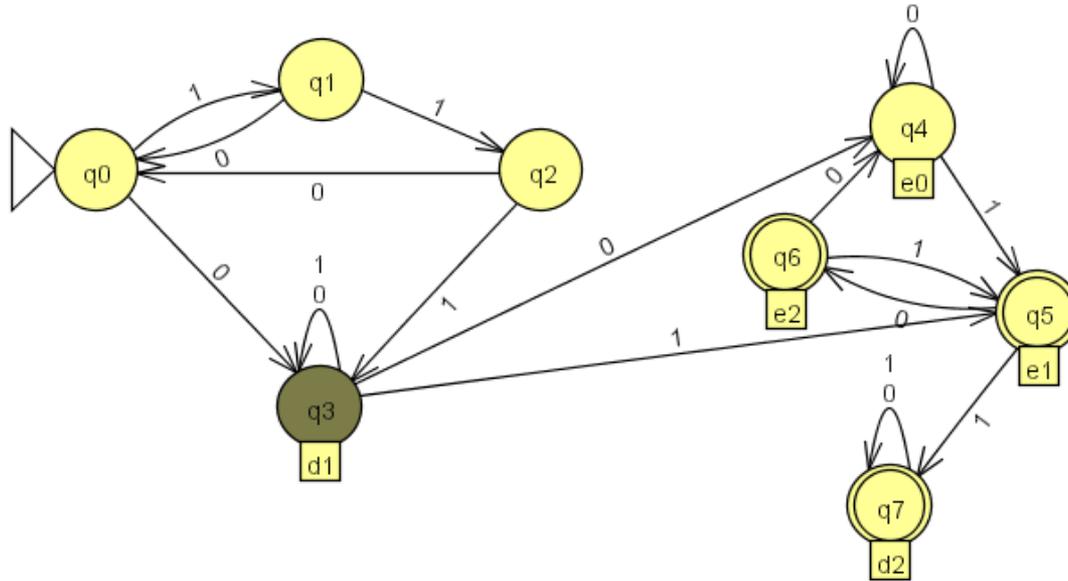
Exemplo

- Use o Lema 3.2 (complemento) e o Teo 3.8 (concatenação) para construir um AF M que a partir de $M1$ e $M2$ reconheça $L = \overline{L1} . \overline{L2}$





<p>q3</p> <p>11101</p>	<p>q5</p> <p>11101</p>	<p>q5</p> <p>11101</p>	
------------------------	------------------------	------------------------	--



q3

11101

Step Reset Freeze Thaw Trace Remove

Moves existing valid configurations to the next configurations.

Teo 3.9 (H&U, 69) - Fechamento

- A classe dos conjuntos aceitos por um AF é fechada sobre o fechamento. Se L é LR então L^* é LR.
- **Prova:** Seja $M = (K, \Sigma, \delta, q_0, F)$ um AF que aceita L .
 $M' = (K \cup \{q_0'\}, \Sigma, \delta', q_0', F \cup \{q_0'\})$ aceita L^* .

 q_0' é também final pois $\lambda \in a_0$ fecho

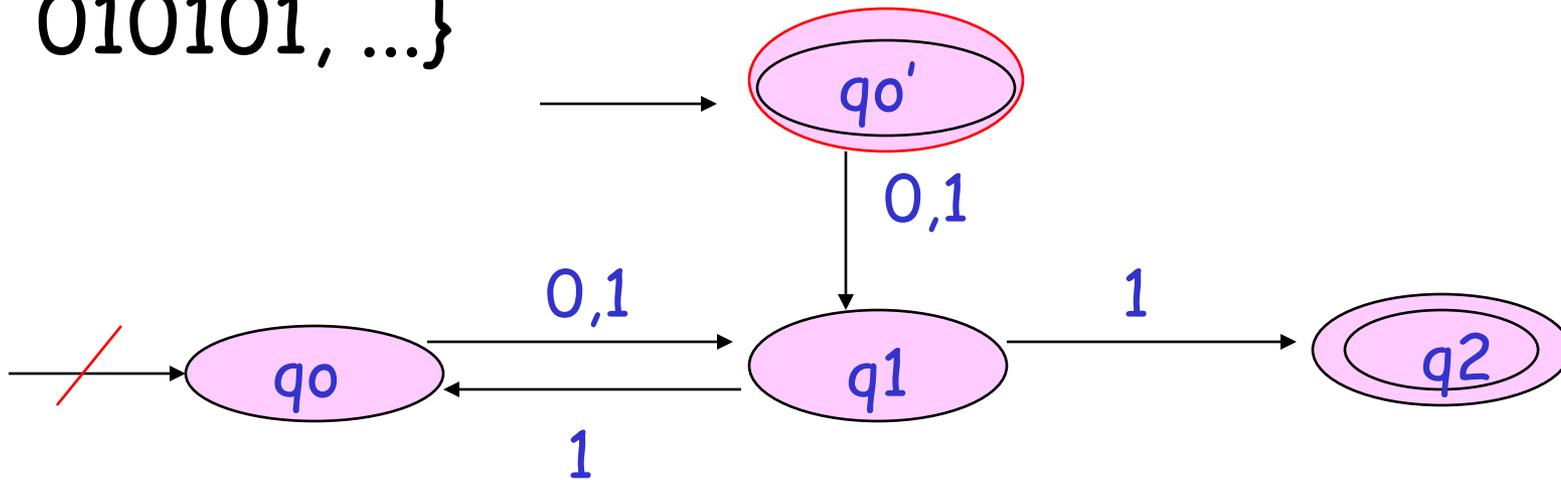
$$\delta'(q, a) = \begin{cases} \{\delta(q, a), q_0\} & \text{se } \delta(q, a) \in F \\ \{\delta(q, a)\} & \text{c.c. para } \forall q \in K \end{cases}$$

(todas as transições que estavam chegando no final, ganham uma cópia para voltar ao início)

$$\delta'(q_0', a) = \begin{cases} \{\delta(q_0, a), q_0\} & \text{se } \delta(q_0, a) \in F \\ \{\delta(q_0, a)\} & \text{cc} \end{cases}$$

Exemplo

- Use o Teo 3.9 para construir um AF que reconheça o fecho de $L = \{01,11\}$
- $L^* = \{\lambda, 01,11,0101,0111,1101,1111,010101, \dots\}$



Teo 4.10 (H,M,U,2001) Diferença

- Se L e M são linguagens regulares então $L - M$ também é.
- Prova: $L - M = L \cap \overline{M}$. Como o complemento e a intersecção de linguagens regulares também são regulares, $L - M$ também é.

Como testar a pertinência em uma LR

Dadas uma cadeia w e uma linguagem regular L , $w \in L$?

- Se L é representada por um AFD:
 - simule o autômato e verifique se um estado final é alcançado. Se $|w|=n$ e o AFD é representado por um estrutura de dados adequada (matriz de transição), então cada transição exigirá um tempo constante e a verificação levará tempo $O(n)$. (veja que o tamanho da entrada é dado pelo comprimento da cadeia)

Como testar a equivalência de LR's, ou se descritores distintos definem a mesma linguagem

- Há uma forma para minimizar um AFD. Ou seja, sempre é possível encontrar um AFD equivalente que tenha o número mínimo de estados.
- Esse autômato mínimo é **único**: dados dois AFDs quaisquer, com número mínimo de estados, sempre podemos renomear os estados de modo que os dois AFDs se tornem iguais.