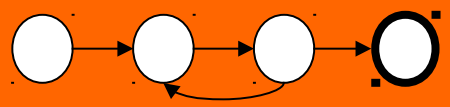
Automata e Linguagens Formais

CTC 34



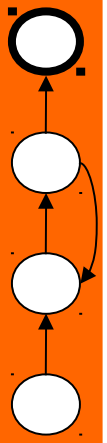
4

Prof. Carlos H. C. Ribeiro

carlos@ita.br

Ramal: 5895 Sala: 106

Algoritmos de Decisão para AFDs
Minimização de AFDs

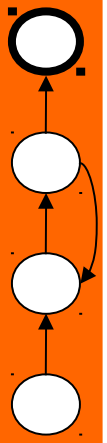


Algoritmos de Decisão para AFDs

Seria ótimo se tivéssemos algoritmos para:

- Determinar se uma dada linguagem é vazia (nenhuma cadeia);
- Determinar se uma dada linguagem é finita (no. finito de cadeia);
- Determinar se uma dada linguagem é infinita (no. infinito de cadeia);
- Determinar se dois autômatos aceitam a mesma linguagem.

Estes algoritmos **existem**! Vamos então estudá-los...



Teorema: Seja M um AFD com k estados.

Então, L(M) é não-vazio $\Leftrightarrow$ M aceita uma cadeia z com $|z| < k$.

Prova:

M aceita uma cadeia z com $|z| < k \Rightarrow$ L(M) não-vazio

Trivial!

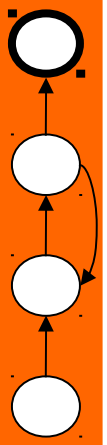
L(M) não-vazio $\Rightarrow$ M aceita uma cadeia z com $|z| < k$

Seja x_{min} a menor cadeia de L(M). Assuma (**por absurdo**) que $|x_{min}| > k-1$.

Pelo *Pumping Lemma*: $x_{min} = uvw$, com $uv^i w \in L$.

Em particular, $uv^0w=uw \in L$. Mas $|uw| < |x_{min}|$, ou seja, a menor cadeia de L(M) seria maior do que uma outra cadeia uw de L(M) (absurdo). Logo, a menor cadeia de L(M) tem comprimento < k.

OBS (notação): |x| = length(x)



Teorema: Seja M um AFD com k estados. Então,
L(M) tem número infinito de cadeias $\Leftrightarrow$ M aceita alguma cadeia x com $k \leq |x| < 2k$.

Prova:

M aceita alguma cadeia x com $k \leq |x| < 2k \Rightarrow$ L(M) tem número infinito de cadeias
Pelo *Pumping Lemma*: x=uvw, com u v^i w $\in$ L, qualquer que seja i: número infinito de cadeias (diferindo só no número de "ciclos" realizados).

L(M) tem número infinito de cadeias $\Rightarrow$ M aceita alguma cadeia x com $k \leq |x| < 2k$
O número de cadeias com |.| < k é limitado pelo no. de símbolos do alfabeto e pelo no. de estados. Logo, teremos cadeias x tal que $|x| \geq k$. Assuma porém (**absurdo**) que não exista x tal que |x| < 2k. Seja então x_{min} a **menor cadeia** tal que $|x_{min}| \geq 2k$.

Pumping Lemma: x_{min} = uvw, $|uv| \leq k$ e uv^0w = uw $\in$ L(M). Temos duas opções para uw:

a) Se $|uw| \geq 2k$,
então x_{min} não seria a menor cadeia com comprimento $\geq 2k$ (pois $|uw| < |uvw = x_{min}|$).
b) Se |uw| < 2k,
então $|uw| \geq k$, pois senão teríamos: $|x_{min}| = |uw| + |v| < 2k$ (contradição)

$<k$ $\leq k$

Algoritmo para Determinar Cardinalidade de Linguagem Regular

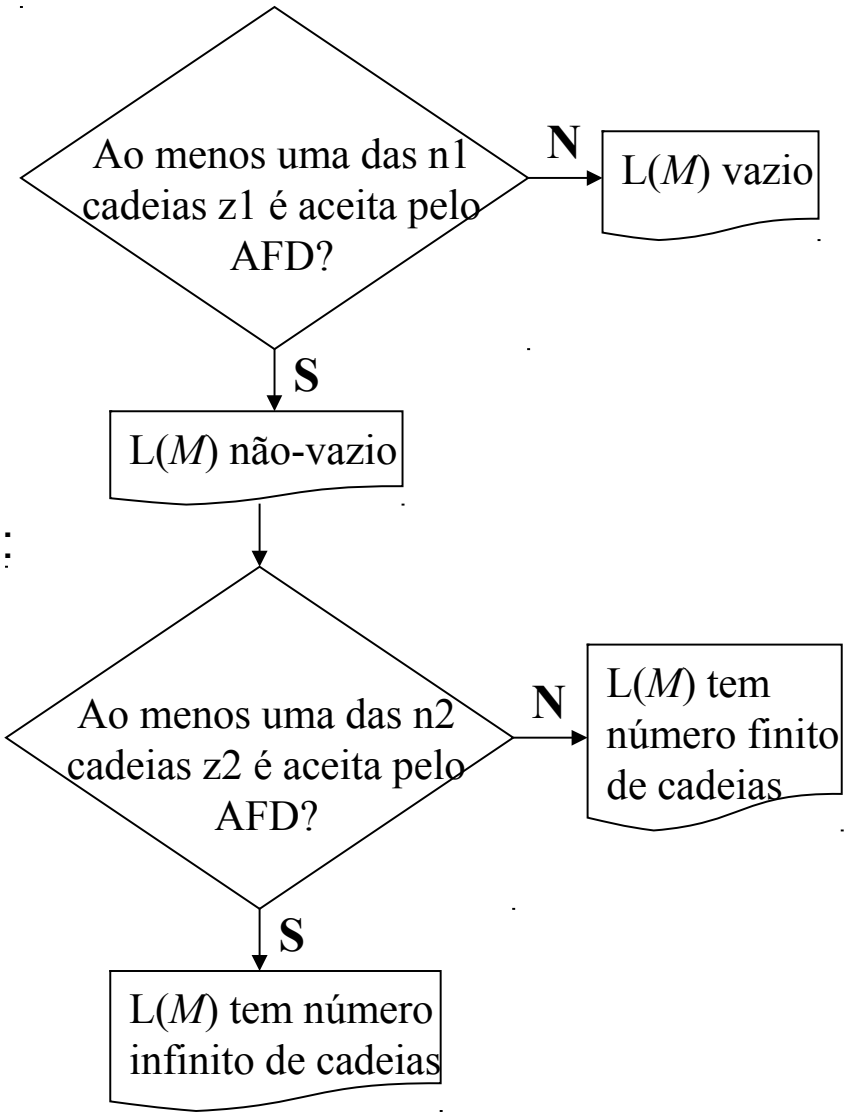
Sejam **k** o número de estados e **j** o tamanho do alfabeto de uma AFD *M*.

No. **n1** de cadeias z1, length(z1) < k :

$1 + j + j^2 + \ldots + j^{k-1} = 1.(j^k - 1) / (j-1)$

No. **n2** de cadeias z2, k ≤ length(z2) < 2k :

$j^k + j^{k+1} + \ldots + j^{2k-1} = j^k(j^k - 1) / (j-1)$



Algoritmo para Determinar Equivalência de AFDs

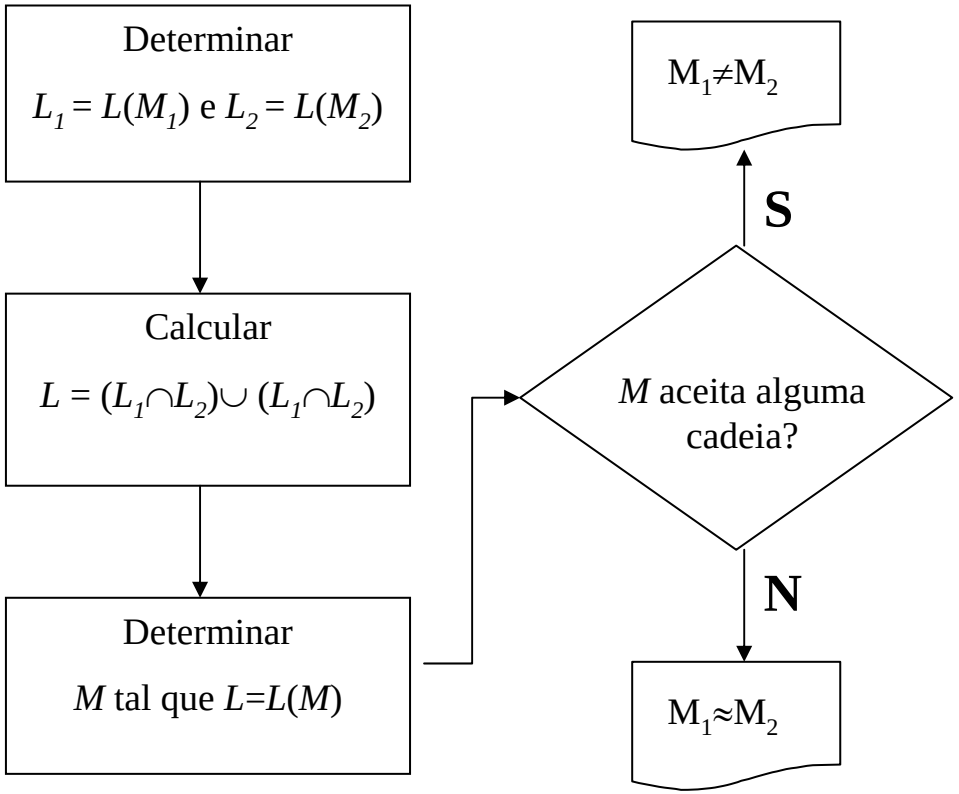
Def.: Dois automata finitos determinísticos M_1 e M_2 são equivalentes se aceitam a mesma linguagem (ou seja, se $L(M_1) = L(M_2)$).

Teorema: Sejam M_1 e M_2 AFDs. Existe um procedimento para determinar se M_1 e M_2 são equivalentes.

Prova: Sejam L_1 e L_2 as linguagens aceitas por M_1 e M_2. Considere então a linguagem

$$L = \left(L_1 \cap \overline{L_2}\right) \cup \left(\overline{L_1} \cap L_2\right)$$

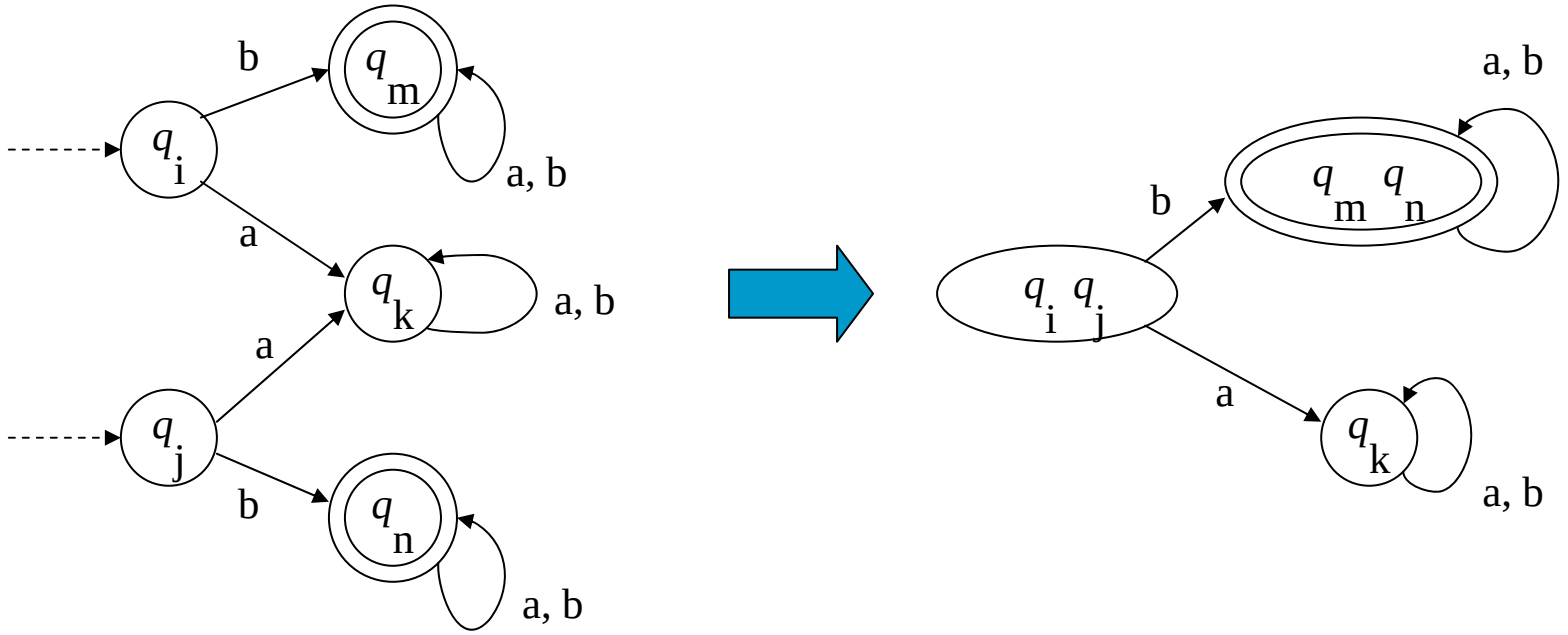
- *L é regular*
- $L = \varnothing$ sss L_1 e L_2 forem idênticos



Equivalência de Estados de um AFD

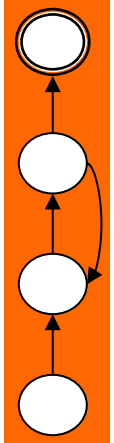
Def.: Seja M=(Q,Σ,δ,q_0,F) um AFD. Dois estados q_i e q_j são **equivalentes** se

$$\hat{\delta}(q_i, u) \in \mathrm{F} \Leftrightarrow \hat{\delta}(q_j, u) \in \mathrm{F}, \text{ para todo } u \in \Sigma^*$$



Estados equivalentes são também ditos **não-distinguíveis**

Estados não-equivalentes são também ditos **distinguíveis**

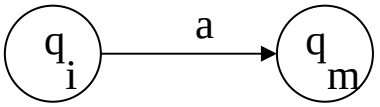


Algoritmo para Identificar Equivalências

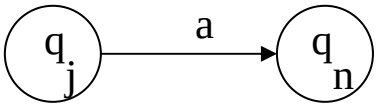
Associados a cada par q_i , q_j (i>j), dois vetores D (de 0´s e 1´s) e S (de conjuntos de pares de inteiros):

- D[i, j]: valor 1 quando q_i e q_j são distinguíveis, caso contrário valor 0.
- S[i, j]: conjunto de pares (m,n) tais que a não-equivalência de q_m e q_n pode ser determinada a partir daquela de q_i e q_j.

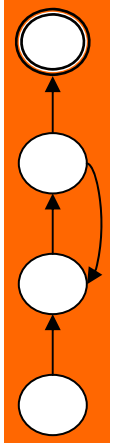
Idéia do algoritmo:



D[n,m]=1: q_m e q_n distinguíveis. Ao examinar q_i e q_j observo que S[m,n] contém o par [i,j], logo são distinguíveis também (e faço D[i,j]=1).



D[n,m] não definido: não sei se q_m e q_n são distinguíveis. Ao examinar q_i e q_j observo que S[m,n] contém o par [i,j], logo serão distinguíveis se, em algum momento, descobrir que D[n,m]=1 (e aí faço D[i,j]=1).



Algoritmo para Identificar Equivalências

Entrada: AFD $M=(Q,\Sigma,\delta,q_0,F)$

- **Inicialização**

  Para cada par q_i, q_j com i<j faça D[i,j]=0, S[i,j]=$\varnothing$

- **Para cada** par i, j com i<j

  Se q_i (q_j) é um estado final e q_j (q_i) não é um estado final **então** faça D[i, j]=1 (estados distinguíveis para a cadeia vazia).

- **Para cada** par i, j com i<j e D[i, ,j]=0

  Se para algum $a\in\Sigma$, $\delta(q_i, a)=q_m$ e $\delta(q_j, a)=q_n$ e D[n,m]=1 (ou D[m,n]=1), então DIST(i, j)

  Senão para cada $a\in\Sigma$ faça

  $\delta(q_i, a)=q_x$ e $\delta(q_j, a)=q_y$

  Se x<y e [i, j] $\neq$ [x, y] **então** adicione [i, j] a S[x, y]

  Senão se x>y e [i, j] $\neq$ [x, y] **então** adicione [i, j] a S[y, x]

DIST(i, j):

D[i, j]=1

para todo [m, n] $\in$ S[i, j] DIST(m, n)

Exemplo

Determine o AFD mínimo equivalente a: