Linguagens Formais e Autômatos (LFA)

Aula de 23/09/2013

Parte 1
Exercícios - Lema do Bombeamento

Exemplos de Solução dos Exercícios da Aula Anterior

II. $L = a^n.b^k$ para n ímpar OU k par

Tabela Verdade para n ímpar OU k par

Caso	N	K	Condição
a)	Ímpar (aa)*a	Ímpar (bb)*b	V
		Par $(bb)^+$	
b)	Par $(aa)^+$	Par $(bb)^+$	V
~~c)~~	~~Par~~	~~Ímpar~~	~~F~~

Cadeia mínima para dar conta do caso (a) é
min=(aa)*a(bb)*b, |min|=2; neste caso p=|min|+1 = 3
Cadeia mínima do caso (b) é min=$(aa)^+(bb)^+$, |min|=4, p=|min|+1 =

O maior p é "5". Tomemos este valor. ($p = 5 \Rightarrow |w| \geq 5$)

Caso	Condições N,K	Padrões a testar	Bombeamento
(a) $\|w\| \geq 5$	N ímpar K par ou ímpar	~~w = ab~~ a) w = ab^+ - (w=abbbb) b) w = (aa)*ab - (w= aaaaab) c) w = (aa)*ab+ - (w=aaabb)	$\|w\| \geq 5$; $\|xy\| \leq p$; $\|y\|>0$; ------------------------------ a) x=a; y=b; z=bbb(b)* xy*z ∈ L OK b) x=a; y=aa; z=aa(aa)*b xy*z ∈ L OK c) x=ε; y=aa; z=(aa)*abb xy*z ∈ L OK
(b) $\|w\| \geq 5$	N par K par	a) w = (aa)*aabb(bb)* - (w=aaaabb) b) w = (aa)*aabb(bb)* - (w=aabbbb)	$\|w\| \geq 5$; $\|xy\| \leq p$; $\|y\|>0$; ------------------------------ a) x=aa; y=aa; z=(aa)*bb b) x=aa; y=bb; z=bb(bb)* xy*z ∈ L OK

Passo 1: O LEMA DO BOMBEAMENTO SE APLICA EM L

I. L= $(ab)^{2n}$ para n>0

1. Existe p inteiro tal que para toda cadeia w ∈ L e $|w| \geq p$
2. w = xyz, $|xy| \leq p$, $|y|>0$
3. xy*z ∈ L

O valor de p é associado à quantidade de estados do autômato finito AF que supostamente aceita todas e somente as cadeias de L.

Na cadeia mínima $(ab)^{2n}$ para n>0 **n=1**; cadeia mínima min=$(ab)^2$; |min|=4.
Número mínimo de estados necessários para aceitar min = |min|+1 = 5
Se estipulamos p = 5 então o bombeamento da cadeia mínima tem de pertencer a L (y = min; y* = min*). Ora, ε ∉ L. Logo, p tem de ser maior do que 5.

Qual a menor cadeia w ∈ L tal que $|w| \geq 6$?

- $|(ab)^{2n}| \geq 6$
- $|(abab)^n| \geq 6$
- n=1 < 6 -- w=abab
- n=2 > 6 -- w=abababab

Teste das condições do Lema para w = abababab:

p = 6
$|w| \geq 6$

x = abab	|x| = 4
y = abab	|y| = 4
$\|xy\| \leq p$	-- FALSO
x = ε	|x| = 0
y = abab	|y| = 4
$\|xy\| \leq p$	-- VERDADEIRO
z = abab	-- cadeia mínima de L
xy*z ∈ L	ε.(abab)*abab ∈ L

Novo Exercício – (Turma INF1626 de 2011-2)

Profs Edward Hermann Haeusler e Bruno Lopes

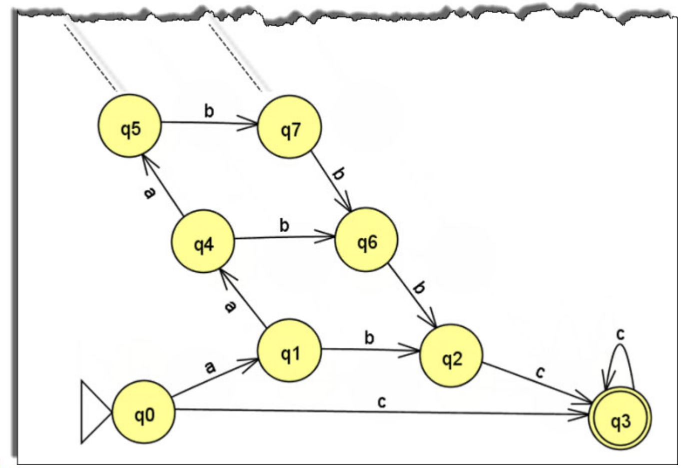
12 de Outubro de 2011

Justifique as suas respostas sempre.
Questões:

1 Lema do Bombeamento

1. A linguagem $L = \{a^k b^k c^n / 0 \leq k \leq 1000 \text{ e } n > 0\}$ é regular. No entanto, Deivid Pé Torto, usando o lema do bombeamento, obteve a seguinte prova de que L não é regular. Aponte o erro na prova de Deivid Pé Torto e argumente que a linguagem é de fato regular.

   Suponho que L é regular, então vale o lema do bombeamento. Isto é, existe um número positivo m tal que para qualquer palavra ω de L com tamanho maior que m, tem-se u,v e z, tal que , $\omega = uvz$, $|uv| \leq m$, $|v| > 0$ e para todo $i \in Nat$, $uv^iz \in L$. Ora, $a^{900}b^{900}c^m$ tem tamanho maior que m, então, uv só tem a's e portanto uv^0z que é o mesmo que uz é da forma $a^{900-r}b^{900}c^m$ com $r > 0$. Como $a^{900-r}b^{900}c^m \notin L$, temos a conclusão desejada.



Dica visual

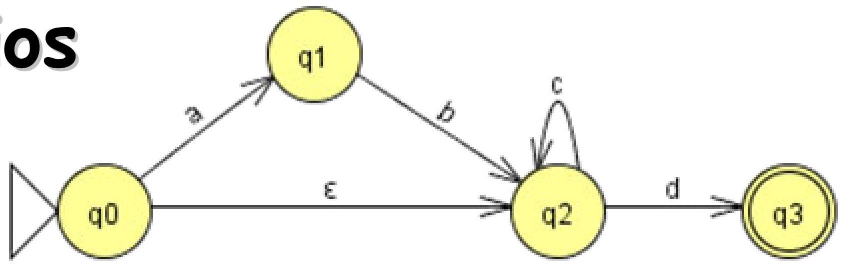
Linguagens Formais e Autômatos (LFA)

Aula de 23/09/2013

Parte 2

Eliminação de Arcos Vazios
Transformação AFND em AFD

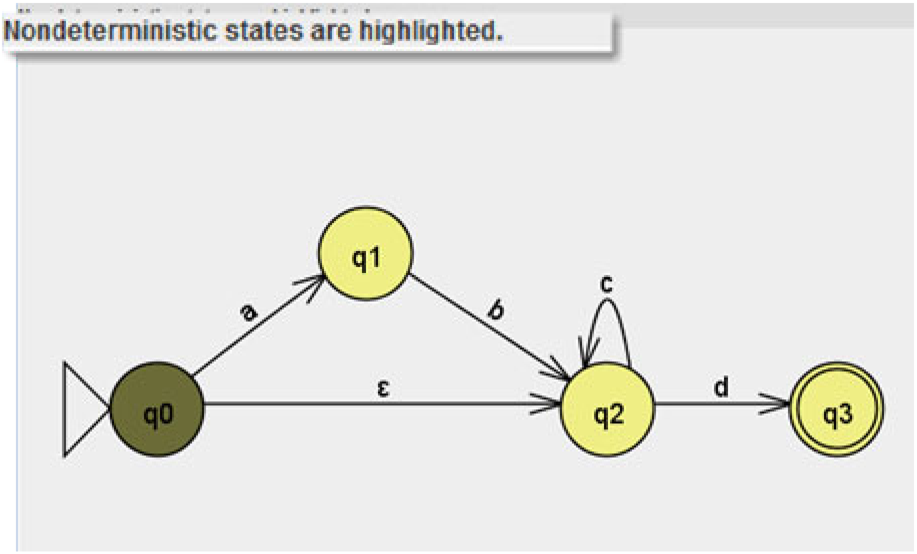
Eliminação de Arcos Vazios



O que são arcos vazios?

Arcos que permitem uma transição de estado SEM avanço da posição do cursor na fita de entrada.

Criam um **não determinismo** (pois o avanço é sempre uma possibilidade).



Para eliminar transições em vazio

Se houver uma transição em vazio de qi para qj (ou seja: qi,ε,qj)

- copiar para as transições de qi todas as transições que **partem de qj** (possibilitadas pela transição em vazio)

Exemplo:

q0,a,q1

q0,ε,q2

q1,b,q2

q2,c,q2

q2,d,q3

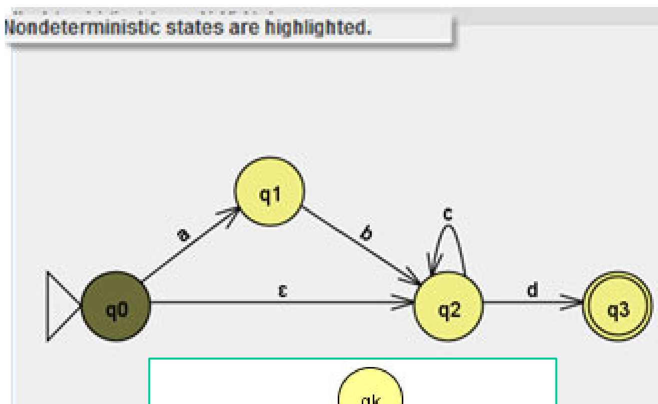
qi,ε,qj → q0,ε,q2

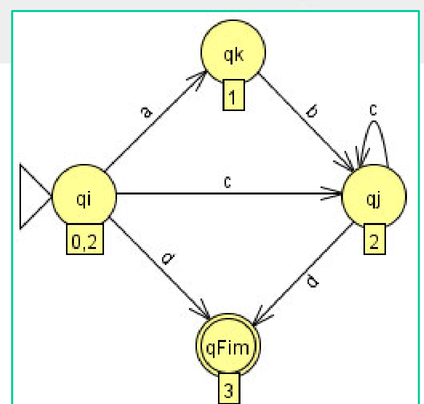
----

q0,c,q2

q0,d,q3

O processo de eliminação é iterativo. Se houver outras transições em vazio, devem ser iterativamente eliminadas.





Transformação de AFND em AFD

1. A eliminação das transições em vazio é um dos casos
2. O outro caso é o de qi,σ,{qj1,qj2,...,qjn}.

O procedimento para eliminar o não determinismo quando as transições não são vazias tem similaridades com a eliminação das transições em vazio:

- Envolve fusão de estados
- Envolve cópias (+ substituições)
- É iterativo

Algoritmo de Conversão Ramos, 2009 p.161

Sejam:

- Q1 o conjunto de estados do AFND
- Q2 o conjunto de estados do AFD
- q01 o estado inicial do AFND
- q02 o estado inicial do AFD
- F1 o conjunto de estados finais de AFND
- F2 o conjunto de estados finais de AFD

Algoritmo 3.4 (Eliminação de não-determinismos) *"Obtenção de um autômato finito determinístico M_2 a partir de um autômato finito não-determinístico M_1."*

- *Entrada: um autômato não-determinístico* $M_1 = (Q_1, \Sigma, \delta_1, q_{01}, F_1)$, *com* $\delta_1 : Q_1 \times \Sigma \rightarrow 2^{Q_1}$;
- *Saída: um autômato determinístico* $M_2 = (Q_2, \Sigma, \delta_2, q_{02}, F_2)$, *com* $\delta_2 : Q_2 \times \Sigma \rightarrow Q_2$, *tal que* $L(M_2) = L(M_1)$;
- *Método:*

  1. $Q_2 \leftarrow \emptyset$;
  2. $\forall i \geqslant 0$, *se* $q_{1i} \in Q_1$ *então* $Q_2 \leftarrow Q_2 \cup \{q_{2i}\}$;
  3. $F_2 \leftarrow \emptyset$;
  4. $\forall i \geqslant 0$, *se* $q_{1i} \in F_1$ *então* $F_2 \leftarrow F_2 \cup \{q_{2i}\}$;
  5. $\delta' \leftarrow \emptyset$;
  6. $\forall q_{1i} \in Q_1, \sigma \in \Sigma$, *se* $\delta_1(q_{1i}, \sigma) = \{q_{11}, ..., q_{1n}\}, n \geqslant 1$ *então* $\delta_2(q_{2i}, \sigma) = \{q_{21}, ..., q_{2n}\}$;
  7. *Substituir todos os elementos* $\{q_{2i}\}$ *de* δ_2 *por* q_{2i};
  8. *Enquanto houver transições não-determinísticas em* δ_2, *faça:*

     a) *Selecione uma transição não-determinística qualquer* $\delta_2(q, \sigma)$, *e considere* $\delta_2(q, \sigma) = \{q_{21}, ..., q_{2i}, ..., q_{2n}\}, n \geqslant 2$;

     b) *Acrescente um novo estado* $q_{21}...q_{2i}...q_{2n}$ *à tabela de transição de estados (nesta notação, os estados do conjunto são concatenados formando uma cadeia, em que os índices dos estados estão organizados em ordem crescente, ou em qualquer outra ordem conveniente); se* $q_{2i} = q_{2i1}...q_{2im}$, *considerar a ordenação de todos os estados obtidos pela substituição de* q_{2i} *por* $q_{2i1}...q_{2im}$ *em* $q_{21}...q_{2i}...q_{2n}$;

     c) *Substitua, na tabela, todas as referências a* $\{q_{21}, ..., q_{2n}\}$ *por* $q_{21}...q_{2n}$;

     d) *Para cada* $\sigma \in \Sigma$, *faça:*

        i. $\delta_2(q_{21}...q_{2n}, \sigma) \leftarrow \emptyset$;

        ii. *Para cada estado* $q_{2j} \in \{q_{21}, ..., q_{2n}\}$, *faça:*

           A. $\delta_2(q_{21}...q_{2n}, \sigma) \leftarrow \delta_2(q_{21}...q_{2n}, \sigma) \cup \delta_2(q_{2j}, \sigma)$;

           B. *Se* $q_{2j} \in F_2$, *então* $F_2 \leftarrow F_2 \cup \{q_{21}...q_{2n}\}$.

Algoritmo de Conversão AFND→ADF (1/n)

Seja um AFND = {Q1,Σ,δ1,F1} (δ1 são transições: $Q1X\Sigma \rightarrow 2^{Q1}$)

O AFD equivalente = {Q2,Σ,δ2,F2} (δ2 são transições: $Q2X\Sigma \rightarrow 2^{Q2}$) constrói-se pelo seguinte método:

1. $Q2 \leftarrow \varnothing$;
2. $\forall i \geq 0$, se $q1i \in Q1$ então $Q2 \leftarrow Q2 \cup \{q2i\}$
3. $F2 \leftarrow \varnothing$;
4. $\forall i \geq 0$, se $q1i \in F1$ então $F2 \leftarrow F2 \cup \{q2i\}$
5. $\delta 2 \leftarrow \varnothing$;
6. $\forall$ $qi \in Q1, \sigma \in \Sigma$ se $\delta 1(q1i,\sigma)=\{q11 \ldots q1n\}$ $n \geq 1$ então $\delta 2 \leftarrow \delta 2(q2i,\sigma)=\{q21 \ldots q2n\}$

Inicializa Estados e copia Q1 para Q2

Inicializa Finais e copia F1 para F2

Inicializa Transições e copia δ1 para δ2

Continua...

Algoritmo de Conversão AFND→ADF (1/n)

7. Substituir todos os elementos {q2i} de $\delta 2$ por q2i

Para todas as transições únicas, trocar conjunto unitário por estado (direto)

8. Enquanto houver transições ND em $\delta 2$, faça:

   a) Selecione uma transição ND $\delta 2(q,\sigma) = \{q21,\ldots,q2i,\ldots q2n\}$ para n>1

   b) Acrescente um novo estado Qnovo a Q2 para representar a reunião de {q21,…,q2i,…q2n}

   c) Substitua todas as ocorrências de {q21,…,q2i,…q2n} por Qnovo

   d) Para cada $\sigma \in \Sigma$

Inicializa transições de Qnovo para cada símbolo do alfabeto

      i. $\delta 2(Qnovo,\sigma) \leftarrow \varnothing$

      ii. Para cada estado $q2j \in \{q21,\ldots,q2i,\ldots q2n\}$, faça:

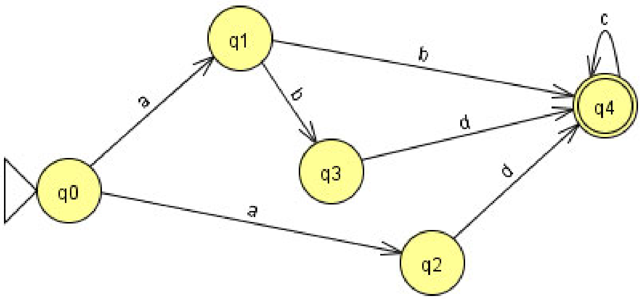
      $\delta 2(Qnovo,\sigma) \leftarrow \delta 2(Qnovo,\sigma) \cup (q2j, \sigma)$

      Se $q2j \in F2$ então $F2 \leftarrow F2 \cup \{q21,\ldots,q2i,\ldots q2n\}$

      Elimina estados inúteis/inacessíveis

Constrói transições para Qnovo correspondentes às de cada estado reunido em Qnovo.
Se algum dos estados de Qnovo for final, então todos os estados reunidos em Qnovo tornam-se finais.

Exemplo




```
AFND

Q1 ={q0,q1,q2,q3,q4}
Inicial: q0
F1={}q4
-----------------
Transições delta1
-----------------
q0,a,{q1,q2}
q1,b,{q3,q4}
q2,d,q4
q3,d,q4
q4,c,q4
```

```
Q2 = {q0,q1,q2,q3,q4}
F2: {q4}
Alfabeto: {a,b,c,d}
-----------
Delta2
q0,a,{q1,q2}        -- Qnovo1
q1,b,{q3,q4}        -- Qnovo2
q2,d,q4
q3,d,q4
q4,c,q4

Passo 8
a) {q1,q2} = Qnovo1
b) Q2 = {q0,q1,q2,q3,q4,Qnovo1}
c) substituição de  {q1,q2} por Qnovo1
     q0,a,Qnovo1
     q1,b,{q3,q4}
     q2,d,q4
     q3,d,q4
     q4,c,q4

d) Criação das transições para Qnovo1
correspondentes a q1 e q2
   Qnovo1,b,{q3,q4}
   Qnovo1,d,q4
   -- q1 e q2 não são finais; F2 inalterado
```

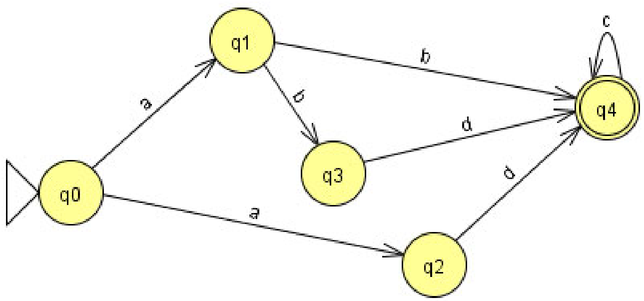
Primeira iteração (Qnovo1)

```
Estado Corrente de Q2
     q0,a,Qnovo1
     q1,b,{q3,q4}
     q2,d,q4
     q3,d,q4
     q4,c,q4
     Qnovo1,b,{q3,q4}
     Qnovo1,d,q4
```

Seja $M = (Q_1, \Sigma, \delta_1, q_{10}, F_1)$ um autômato finito qualquer. Formalmente, um estado $q_{1i} \in Q_1$ é dito "inacessível" quando não existir caminho, formado por transições válidas, que conduza o autômato do seu estado inicial q_{10} até o estado q_{1i}. Em outras palavras, não existe $\alpha \in \Sigma^*$ tal que $\delta(q_{10}, \alpha) = q_{1i}$. Caso contrário, o estado q_{1i} é dito "acessível".

Estados inacessíveis não contribuem para a definição da linguagem aceita por M, podendo ser sistematicamente identificados e eliminados do conjunto de estados, sem prejuízo para a linguagem aceita pelo autômato.

Exemplo




```
AFND

Q1 ={q0,q1,q2,q3,q4}
Inicial: q0
F1={}q4
-----------------
Transições delta1
-----------------
q0,a,{q1,q2}
q1,b,{q3,q4}
q2,d,q4
q3,d,q4
q4,c,q4

```

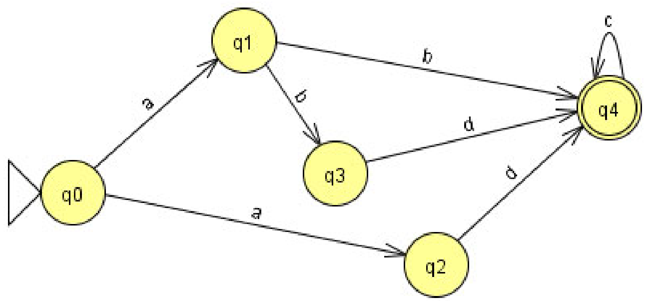
```
Q2 = {q0,q1,q2,q3,q4}
F2: {q4}
Alfabeto: {a,b,c,d}
-----------
Delta2
q0,a,{q1,q2}        -- Qnovo1
q1,b,{q3,q4}        -- Qnovo2
q2,d,q4
q3,d,q4
q4,c,q4

Passo 8
a) {q1,q2} = Qnovo1
b) Q2 = {q0,q1,q2,q3,q4,Qnovo1}
c) substituição de  {q1,q2} por Qnovo1
    q0,a,Qnovo1
    q1,b,{q3,q4}
    q2,d,q4
    q3,d,q4
    q4,c,q4

d) Criação das transições para Qnovo1
correspondentes a q1 e q2
   Qnovo1,b,{q3,q4}
   Qnovo1,d,q4
   -- q1 e q2 não são finais; F2 inalterado
```

Primeira iteração (Qnovo1)

Estado Corrente de Q2
q0,a,Qnovo1
~~q1,b,{q3,q4}~~
~~q2,d,q4~~
q3,d,q4
q4,c,q4
Qnovo1,b,{q3,q4}
Qnovo1,d,q4

q1 e q2 não são destino de nenhuma trnasição; são inúteis

Exemplo




```
AFND

Q1 ={q0,q1,q2,q3,q4}
Inicial: q0
F1={}q4
-----------------
Transições delta1
-----------------
q0,a,{q1,q2}
q1,b,{q3,q4}
q2,d,q4
q3,d,q4
q4,c,q4

```

```
Passo 8
a) {q3,q4} = Qnovo2
b) Q2 = {q0,q3,q4,Qnovo1,Qnovo2}
c) substituição de  {q3,q4} por Qnovo2
     q0,a,Qnovo1
     q3,d,q4
     q4,c,q4
     Qnovo1,b,Qnovo2
     Qnovo1,d,q4

d) Criação das transições para Qnovo2
correspondentes a q3 e q4
    Qnovo2,d,q4
    Qnovo2,c,q4
    -- 4 é final; F2 <-- Qnovo2

Estado Corrente de Q2
     q0,a,Qnovo1
     q3,d,q4
     q4,c,q4
     Qnovo1,b,Qnovo2
     Qnovo1,d,q4
     Qnovo2,d,q4
     Qnovo2,c,q4
```

Segunda iteração (Qnovo2)

Estado Corrente de Q2

q0,a,Qnovo1

~~q3,d,q4~~

q4,c,q4

Qnovo1,b,Qnovo2

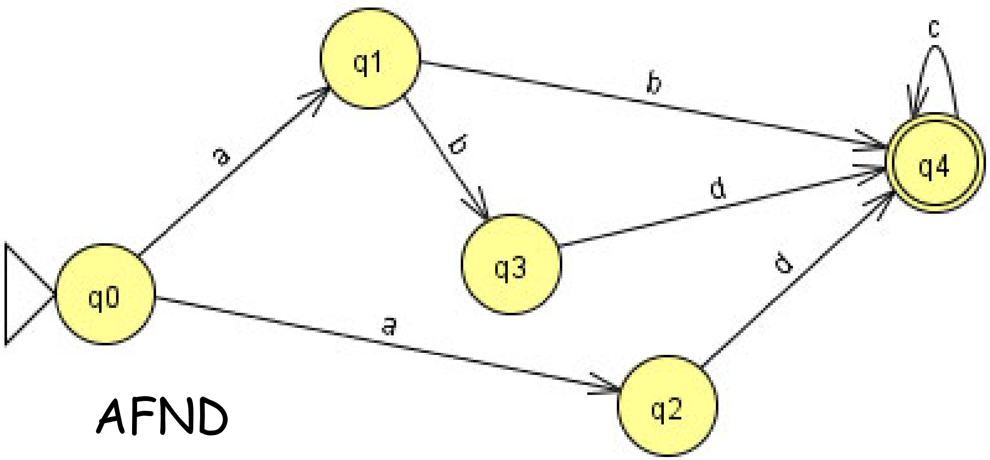
Qnovo1,d,q4

Qnovo2,d,q4

Qnovo2,c,q4

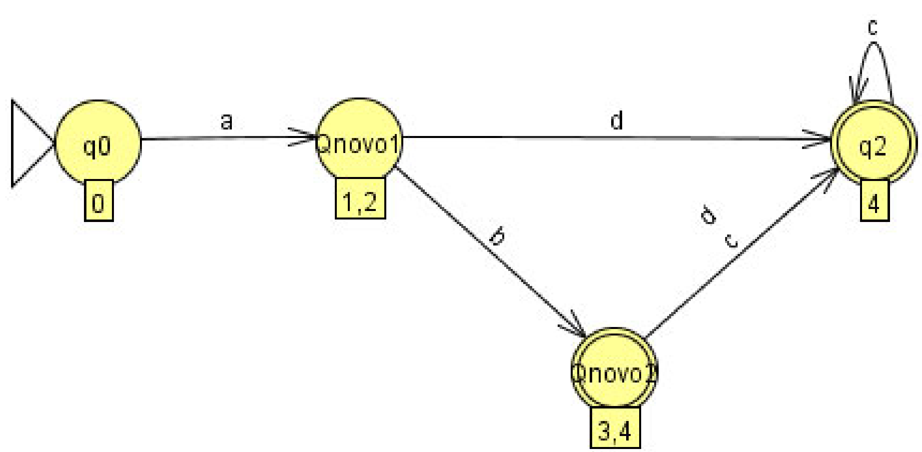
q4 é estado destino de várias transições; não pode ser eliminado. q3 pode.

Autômato Resultante da Conversão



AFND

AFD



Exercício: Conversão AFND em AFD

Exemplo do Exercício sobre L = $a^n b^k$ para n ímpar ou k par

```
            -- qa1:     inicial
            -- qb2,q5:  finais
qa1,a,qa3
qa1,a,qa2
qa2,b,qb2   // final
qa2,a,qa1
qa3,a,qa4
qa4,b,qb1
qa4,a,qa3
qb2,b,qb2   // final
qb1,b,q5    // final
q5,b,qb1
```