Teoria da Computação
Tese de Church Turing
Exercícios

Leonardo Takuno
{leonardo.takuno@gmail.com}

Centro Universitário Senac

Sumário

Sumário

Definição Formal

Uma **máquina de Turing** é uma 7-upla $(Q, \Sigma, \Gamma, \delta, q_0, q_{aceita}, q_{rejeita})$, onde Q, Σ, Γ, são todos conjuntos finitos e

1. Q é o conjunto de estados
2. Σ é o alfabeto de entrada não contém o simbolo em branco $\sqcup$
3. Γ é o alfabeto da fita, onde $\sqcup \in \Gamma$, e $\Sigma \subseteq \Gamma$
4. $\delta : Q \times \Gamma \longrightarrow Q \times \Gamma \times \{L, R\}$ é a função de transição
5. $q_0 \in Q$ é o estado inicial.
6. $q_{aceita} \in Q$ é o estado de aceitação, e
7. $q_{rejeita} \in Q$ é o estado de rejeição, onde $q_{rejeita} \neq q_{aceita}$

Sumário

Exercício 01

Uma máquina de Turing pode alguma vez escrever o símbolo branco $\sqcup$ em sua fita?

Exercício 01

Uma máquina de Turing pode alguma vez escrever o símbolo branco $\sqcup$ em sua fita?
Sim. Γ é o alfabeto da fita. Uma MT pode escrever qualquer símbolo em Γ na fita, e $\sqcup \in \Gamma$ de acordo com a definição.

Exercício 01

O alfabeto de fita Γ pode ser o mesmo que o alfabeto de entrada Σ?

Exercício 01

O alfabeto de fita Γ pode ser o mesmo que o alfabeto de entrada Σ?

Não. Σ nunca contém $\sqcup$ mas Γ sempre contém $\sqcup$. Portanto, eles não podem ser iguais.

Exercício 01

A cabeça de uma máquina de Turing pode alguma vez estar na mesma localização em dois passos sucessivos?

Exercício 01

A cabeça de uma máquina de Turing pode alguma vez estar na mesma localização em dois passos sucessivos?
Sim. Se a MT tenta mover sua cabeça para fora da extremidade esquerda da fita, ele permanece na mesma célula da fita.

Exercício 01

Uma máquina de Turing pode conter apenas um único estado?

Exercício 01

Uma máquina de Turing pode conter apenas um único estado? Não. Qualquer MT deve conter dois estados distintos q_{aceita} e $q_{rejeita}$. Então, a MT deve conter pelo menos dois estados.

Exercício 02

Explique por que a descrição a seguir não é uma descrição de uma máquina de Turing legítima.
M_{ruim} = "A entrada é um polinômio p sobre as variáveis $x_1, \cdots, x_k$.

1. Tente todos os valores inteiros de x.
2. Calcule o valor de p sobre todos esses valores.
3. Se algum torna o valor de p igual a 0, *aceite*; caso contrário, *rejeite*."

Exercício 02

M_{ruim} = "A entrada é um polinômio p sobre as variáveis $x_1, \cdots, x_k$.

1. Tente todos os valores inteiros de x.
2. Calcule o valor de p sobre todos esses valores.
3. Se algum torna o valor de p igual a 0, *aceite*; caso contrário, *rejeite*."

- Passo 1 não é simples, queremos manter os passos simples.
- Passo 3 o termo caso contrário, indica que nenhum valor inteiro de x torna p igual a 0. Ou seja, é preciso testar todos os valores inteiros, o que é impossível.

Exercício 03

Dê descrições no nível de implementação de máquina de Turing que decidam as seguintes linguagens sobre o alfabeto $\{0, 1\}$.

$$A = \{w | w \text{ possui o mesmo número de 0s e 1s}\}$$

ex:

- $01100011 \in A$
- $0111 \notin A$

Exercício 03

$$A = \{w | w \text{ possui o mesmo número de 0s e 1s}\}$$

M_1 = "Sobre a entrada w:

1. Varrer a fita e marque um 0 que não tenha sido marcado. Se não há nenhum 0 marcado, vá para o passo 4. Senão, mova a cabeça para o início da fita.
2. Faça uma varredura na fita e marque o primeiro 1 que não tiver sido marcado. Se nenhum 1 não marcado for encontrado, rejeite.
3. Mova a cabeça para o início da fita e vá para o passo 1.
4. Mova a cabeça de volta para o início da fita. Faça uma varredura na fita para ver se resta algum 1 não marcado. Se nenhum for encontrado, aceite; caso contrário, rejeite"

Exercício 03

$$A = \{w | w \text{ possui o mesmo número de 0s e 1s}\}$$

Reescreva o algoritmo acima o algoritmo em alto-nível para a linguagem A.

Exercício 03

$$A = \{w | w \text{ possui o mesmo número de 0s e 1s}\}$$

Reescreva o algoritmo acima o algoritmo em alto-nível para a linguagem A.

M_1 = "Sobre a entrada w:

1. Enquanto houver 0s e 1s que não foram marcados faça:
   Marque um 0 e encontre 1 equivalente
2. Se todos os 0s e todos 1s foram marcados, aceite; caso contrário existe algum 0 ou 1 não marcado, então rejeite"

Exercício 04

Mostre que a coleção de linguagens decidíveis é fechada sob a operação de união.

Exercício 04

Mostre que a coleção de linguagens decidíveis é fechada sob a operação de união.

Resposta: Para quaisquer duas linguagens decidíveis L_1 e L_2, sejam M_1 e M_2 MTs que as decidem. Construímos uma MT M' que decide a união de L_1 e L_2:

Exercício 04

Mostre que a coleção de linguagens decidíveis é fechada sob a operação de união.

Resposta: Para quaisquer duas linguagens decidíveis L_1 e L_2, sejam M_1 e M_2 MTs que as decidem. Construímos uma MT M' que decide a união de L_1 e L_2:

M' = "Sobre a entrada w:

1. Rode M_1 sobre w. Se ela aceita, *aceite*.
2. Rode M_2 sobre w. Se ela aceita, *aceite*. Caso contrário, *rejeite*."

M' aceita w se M_1 ou M_2 a aceita. Se ambas rejeitam, M' rejeita.

Exercício 05

Mostre que a coleção de linguagens decidíveis é fechada sob a operação de concatenação.

Exercício 05

Mostre que a coleção de linguagens decidíveis é fechada sob a operação de concatenação.

Resposta: Para quaisquer duas linguagens decidíveis L_1 e L_2, sejam M_1 e M_2 MTs que as decidem. Construímos uma MT Não Determinística M' que decide a concatenação de L_1 e L_2:

Exercício 05

Mostre que a coleção de linguagens decidíveis é fechada sob a operação de concatenação.

Resposta: Para quaisquer duas linguagens decidíveis L_1 e L_2, sejam M_1 e M_2 MTs que as decidem. Construímos uma MT Não Determinística M' que decide a concatenação de L_1 e L_2:

M' = "Sobre a entrada w:

1. Para cada maneira de cortar w em duas partes $w = w_1 w_2$:
2. Rode M_1 sobre w_1.
3. Rode M_2 sobre w_2.
4. Se ambas aceitarem, *aceite*. Caso contrário, continue com o próximo $w_1 w_2$
5. Se todos os cortes foram testados sem sucesso, então *rejeite*."

Exercício 05

Tentamos todos os possíveis cortes de w. Se em algum momento encontrar um corte tal que a primeira parte é aceita por M_1 e a segunda parte é aceita por M_2, w está na concatenação de L_1 e L_2. Portanto M' aceita w. Caso contrário, w não pertence à concatenação de linguagens e é rejeitada.

Exercício 06

Mostre que a coleção de linguagens decidíveis é fechada sob a operação de estrela.

Exercício 06

Mostre que a coleção de linguagens decidíveis é fechada sob a operação de estrela.

Resposta: Para qualquer linguagem decidível L, seja M uma MT que a decide. Construímos uma MT Não Determinística M' que decide a estrela de L:

Exercício 06

Mostre que a coleção de linguagens decidíveis é fechada sob a operação de estrela.

Resposta: Para qualquer linguagem decidível L, seja M uma MT que a decide. Construímos uma MT Não Determinística M' que decide a estrela de L:

M' = "Sobre a entrada w:

1. Para cada maneira de cortar w em partes $w = w_1 w_2 \cdots w_n$:
2. Rode M sobre w_i para $i = 1, 2, \cdots, n$. Se M aceita cada um dessas strings w_i, *aceite*.
3. Se todos os cortes foram testados sem sucesso, então *rejeite*."

Exercício 06

Se há uma maneira de cortar w em diferentes substrings tal que toda substring seja aceita por M, w pertence a estrela de L e assim M' aceita w. Caso contrário, w é rejeitado. Como há um número finito de possíveis cortes de w, M' pára após um número finito de passos.

Exercício 07

Mostre que a coleção de linguagens decidíveis é fechada sob a operação de complemento.

Exercício 07

Mostre que a coleção de linguagens decidíveis é fechada sob a operação de complemento.
Resposta: Para qualquer linguagem decidível L, seja M uma MT que a decide. Construímos uma MT M' que decide o complemento de L:

Exercício 07

Mostre que a coleção de linguagens decidíveis é fechada sob a operação de complemento.

Resposta: Para qualquer linguagem decidível L, seja M uma MT que a decide. Construímos uma MT M' que decide o complemento de L: $M' =$ "Sobre a entrada w:

1. Execute M sobre w. Se M aceita, rejeite; Se M rejeita, *aceite*."

Como M' faz o oposto que qualquer M faz, então ele decide o complemento de L.

Exercício 08

Mostre que a coleção de linguagens decidíveis é fechada sob a operação de intersecção.

Exercício 08

Mostre que a coleção de linguagens decidíveis é fechada sob a operação de intersecção.
Resposta: Para quaisquer duas linguagens decidíveis L_1 e L_2, sejam M_1 e M_2 MTs que as decidem. Construímos uma MT Não Determinística M' que decide a intersecção de L_1 e L_2:

Exercício 08

Mostre que a coleção de linguagens decidíveis é fechada sob a operação de intersecção.

Resposta: Para quaisquer duas linguagens decidíveis L_1 e L_2, sejam M_1 e M_2 MTs que as decidem. Construímos uma MT Não Determinística M' que decide a intersecção de L_1 e L_2:

$M' =$ "Sobre a entrada w:

1. Execute M_1 sobre w. Se M_1 rejeita, rejeite;
2. Execute M_2 sobre w. Se M_2 aceita, aceite; Caso contrário, rejeite."

M' aceita w se ambos M_1 e M_2 a aceitam. Se algum deles rejeita, M' rejeita w, também.

Exercício 09

Dê uma definição formal de um enumerador. Considere-o como um tipo de máquina de Turing de duas fitas que suas sua segunda fita como a impressora.

Exercício 09

Um enumerador é uma 7-tupla $(Q, \Sigma, \Gamma, \delta, q_0, q_{print}, q_{aceita})$, onde Q, Σ, Γ são todos conjuntos finitos e

i) Q é o conjunto de estados,
ii) Γ é o alfabeto da fita de trabalho,
iii) Σ é o alfabeto da fita de saída,
iv) $\delta : Q \times \Gamma \longrightarrow Q \times \Gamma \times \{E, D\} \times \Sigma_\varepsilon$ é a função de transição,
v) $q_0 \in Q$ é o estado inicial,
vi) $q_{print} \in Q$ é o estado de impressão, e
vii) $q_{halt} \in Q$ é o estado de parada, onde $q_{print} \neq q_{halt}$.

Exercício 09

A computação de enumerador E é definido como em uma MT comum, exceto pelos seguintes pontos.

- Ela tem duas fitas, uma fita de trabalho e uma fita de impressão, ambos inicialmente brancos.
- A cada passo, a máquina pode escrever um símbolo de Σ na fita de saída, ou nada, como determinado por δ.
- Se $\delta(q, a) = (r, b, E, c)$, significa que em um estado q, lendo a, o enumerador E entra no estado r, escreve um b na fita de trabalho, move a cabeça da fita de trabalho para a esquerda (ou direita, se E fosse D), escreve c na fita de saída, e move a cabeça da fita de saída para a direita se $c \neq \varepsilon$.

Exercício 09

Quando o estado q_{print} é alcançado, a fita de saída é reiniciado para branco e a cabeça de leitura retorna para a extremidade esquerda. A máquina pára quando q_{aceita} é alcançada.

$$L(E) = \{w \in \Sigma^* | w \text{ aparece na fita de trabalho se } q_{print} \text{ é alcançado } \}.$$

Exercício 10

Outra maneira de escreve a linguagem do enumerador é

$$L(E) = \{w \in \Sigma^* | E \text{ eventualmente, em tempo finito, imprime } w\}.$$

Pergunta, $L(E)$ pode ser infinita?
a) Não, strings devem impressas em tempo finito.
b) Não, strings devem todos ser de tamanho finito.
c) Sim, pode acontecer se E não pára.
d) Sim, todas as linguagens $L(E)$ são infinitas.
e) Nenhuma das alternativas.

Exercício 10

Outra maneira de escreve a linguagem do enumerador é

$$L(E) = \{w \in \Sigma^* | E \text{ eventualmente, em tempo finito, imprime } w\}.$$

Pergunta, $L(E)$ pode ser infinita?
a) Não, strings devem impressas em tempo finito.
b) Não, strings devem todos ser de tamanho finito.
c) Sim, pode acontecer se E não pára. $\leftarrow$
d) Sim, todas as linguagens $L(E)$ são infinitas.
e) Nenhuma das alternativas.

Exercício 11

Para cada Σ, existe um enumerador cuja linguagem é o conjunto de todas strings sobre Σ.

a) Verdadeiro

b) Falso

c) Depende de Σ

d) nenhuma das alternativas

Exercício 11

Para cada Σ, existe um enumerador cuja linguagem é o conjunto de todas strings sobre Σ.

a) Verdadeiro $\leftarrow$
b) Falso
c) Depende de Σ
d) nenhuma das alternativas

Décimo problema de Hilbert

$D = \{\langle p \rangle | p$ é um polinômio de uma variável com coeficientes inteiros que tem uma raiz inteira$\}$
M = "Sobre a entrada $\langle p \rangle$ sobre a variável x:

1. Para $i = 0, -1, +1, -2, +2, \cdots$
2. Calcule $p(i)$. Se ele avalia para 0, aceite a entrada. Caso contrário vá para o próximo valor de i.

Exercício 11

A MT M do slide anterior é um decisor?
a) sim
b) não
c) não é MT
d) eu não sei.

Exercício 11

A MT M do slide anterior é um decisor?

a) sim

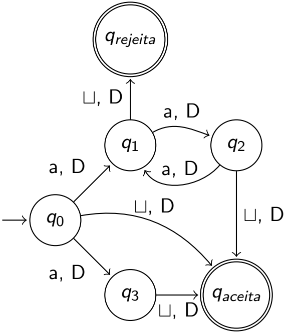
b) não $\leftarrow$

c) não é MT

d) eu não sei.

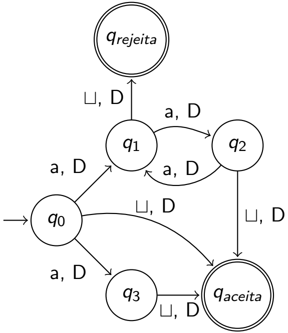
Exercício 12

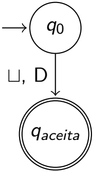
Seja $L = \{a^k | k \text{ é par ou } k = 1\}$ e a Máquina de Turing Não Determinística N.



Exercício 12

Entrada : ε

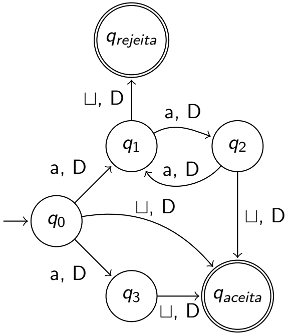


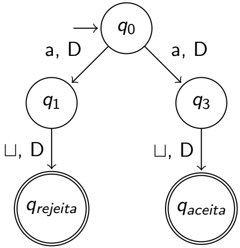


árvore de computação

Exercício 12

Entrada : a

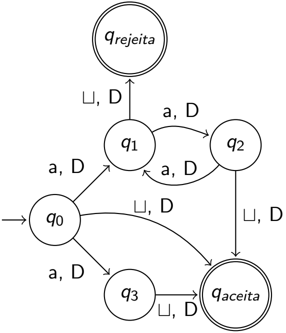


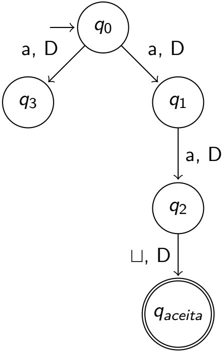


árvore de computação

Exercício 12

Entrada : *aa*





árvore de computação

Exercício 12

Teorema 3.16: Toda máquina de **Turing Não-Determinística** tem uma máquina de Turing de uma única fita que lhe é equivalente.

Prova:
Dado uma máquina de Turing não-determinística (N) mostrar como construir uma máquina determinística (D).
- Se N aceita em qualquer ramo, a MT D vai aceitar
- Se N pára em qualquer ramo sem aceitar, D vai parar e rejeitar.

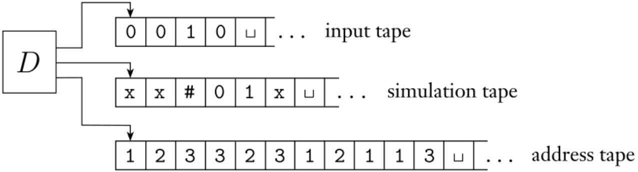
Exercício 12

Teorema 3.16: Toda máquina de **Turing Não-Determinística** tem uma máquina de Turing de uma única fita que lhe é equivalente.

Abordagem:
- Simule N
- Simule todos os ramos de computação
- Procure um caminho que N aceita e aceite (Busca em largura)

Exercício 12

Simulando MT não-determinística em uma MT determinística multifita.



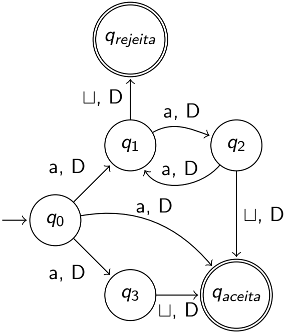
Exercício 12

Algoritmo

- Inicialmente: Fita 1 contém entrada e Fita 2 e Fita 3 estão vazias.
- Copie Fita 1 para a Fita 2
- Use a Fita 2 como fita de trabalho.
- Consulte a Fita 3 para percorrer a árvore de computação.
- Execute a simulação seguindo o caminho indicado pela Fita 3 o mais profundo possível (ou até parar).
- Reinicie a Fita 2 e atualize a Fita 3 com o próximo endereço em ordem lexicográfica.
- Se o estado de aceitação é alcançado, aceite.

Exercício 12

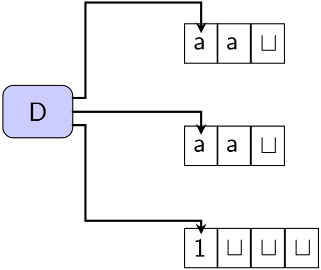
Seja $L = \{a^k | k \text{ é par ou } k = 1\}$ e a Máquina de Turing Não Determinística N.

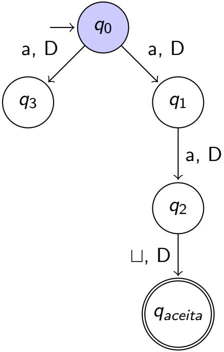


Simule a máquina não determinística N em uma máquina multi-fita.

Exercício 12

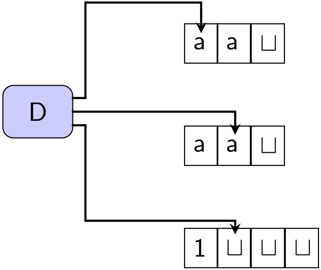
Entrada : *aa*

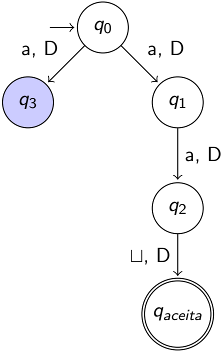




Exercício 12

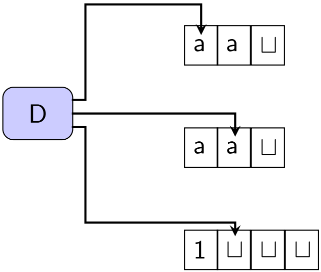
Entrada : *aa*

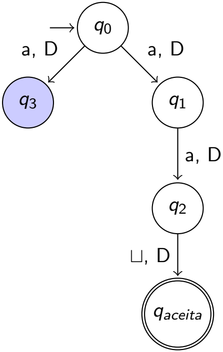




Exercício 12

Entrada : *aa*

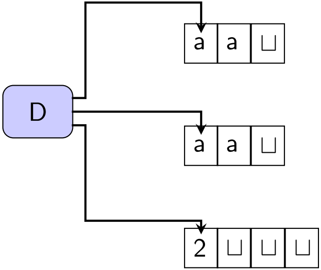


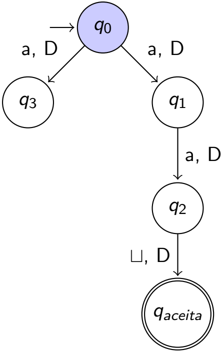


O estado q_3 não é estado final, e não terminou de ler a fita 2. Reseta fita 2 e fita 3 com um novo caminho em ordem lexicográfica.

Exercício 12

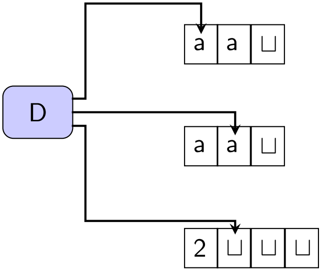
Entrada : *aa*

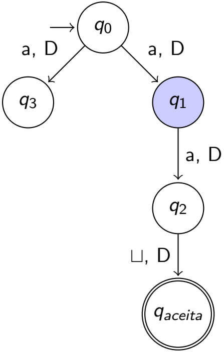




Exercício 12

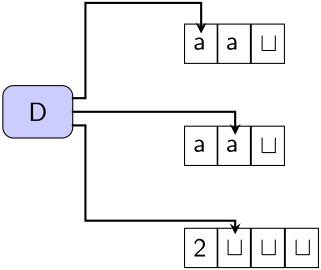
Entrada : *aa*

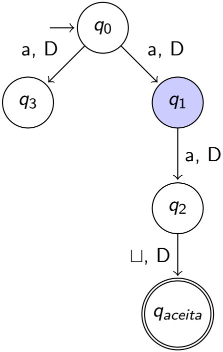




Exercício 12

Entrada : *aa*

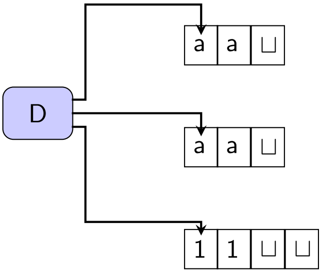


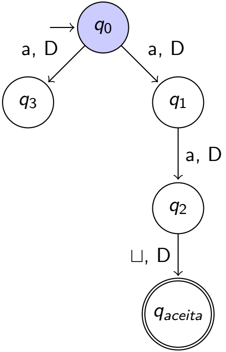


O estado q_1 não é estado final, e na fita 2 ainda restam símbolos a serem processados. Reseta fita 2 e fita 3 com um novo caminho em ordem lexicográfica.

Exercício 12

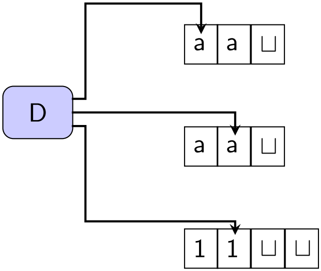
Entrada : *aa*

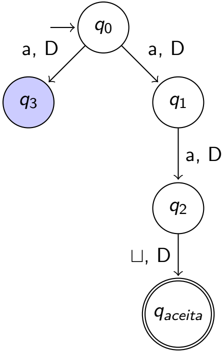




Exercício 12

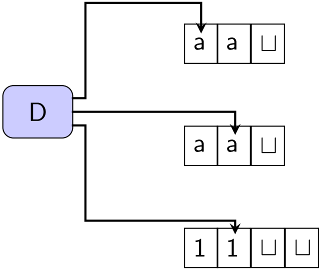
Entrada : *aa*

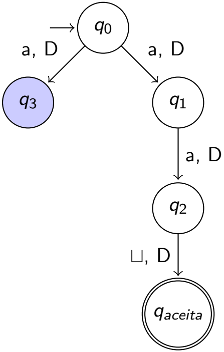




Exercício 12

Entrada : *aa*

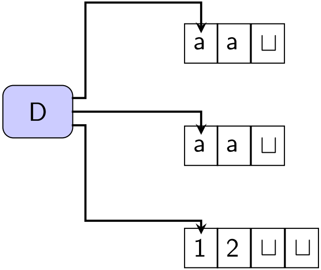


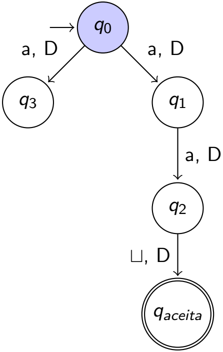


A partir de q_3, lendo símbolo *a*, leva para estado de rejeição. Reseta fita 2 e fita 3 com um novo caminho em ordem lexicográfica.

Exercício 12

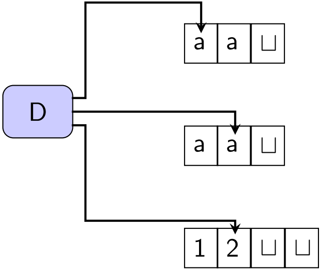
Entrada : *aa*

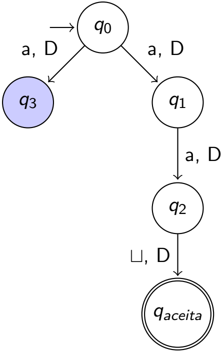




Exercício 12

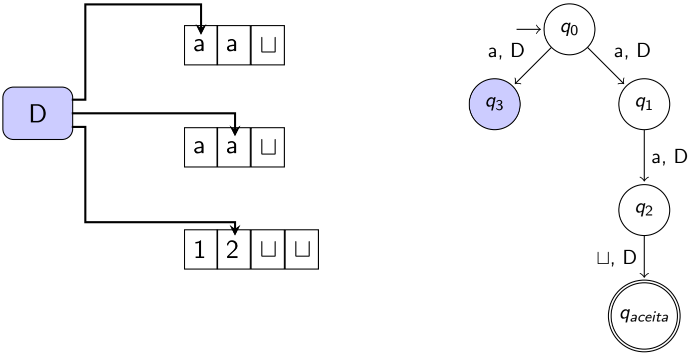
Entrada : *aa*





Exercício 12

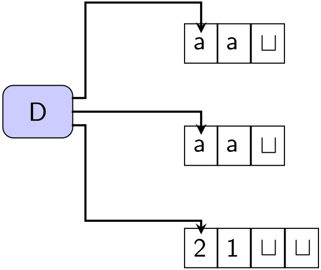
Entrada : *aa*

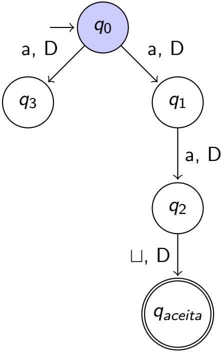


A partir do estado q_3, lendo o símbolo *a*, leva a estado de rejeição. Reseta fita 2 e fita 3 com um novo caminho em ordem lexicográfica.

Exercício 12

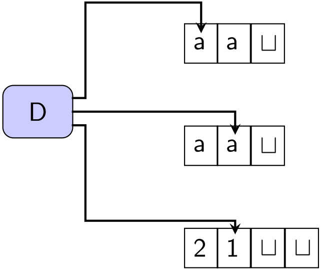
Entrada : *aa*

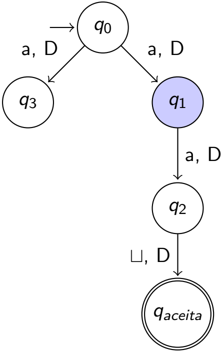




Exercício 12

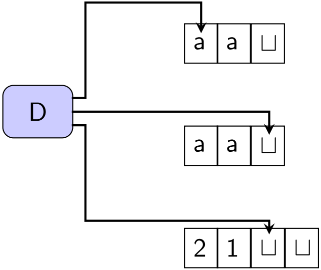
Entrada : *aa*

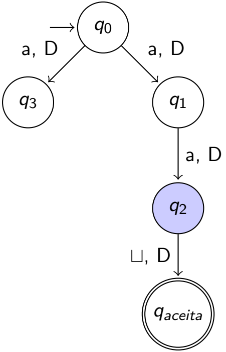




Exercício 12

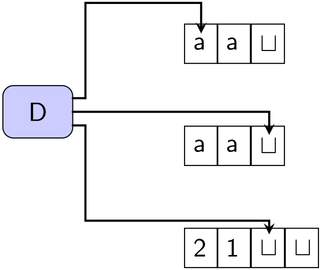
Entrada : *aa*

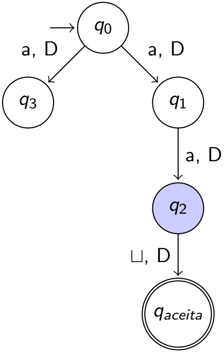




Exercício 12

Entrada : *aa*

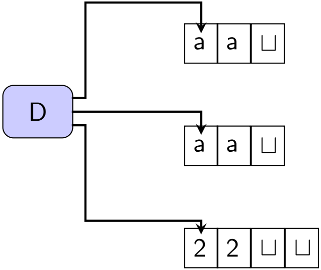


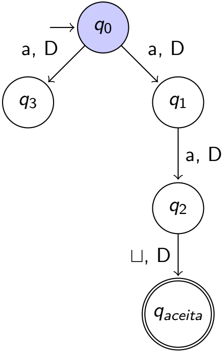


O estado q_2 não é estado final, a máquina de Turing ainda não terminou de processar a cadeia. Reseta fita 2 e fita 3 com um novo caminho em ordem lexicográfica.

Exercício 12

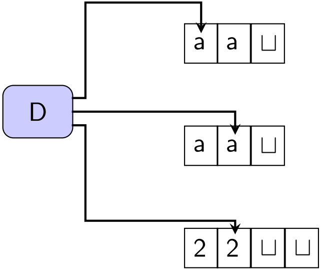
Entrada : aa

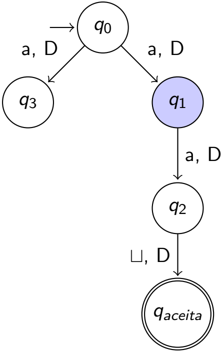




Exercício 12

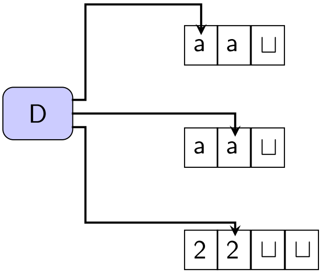
Entrada : *aa*

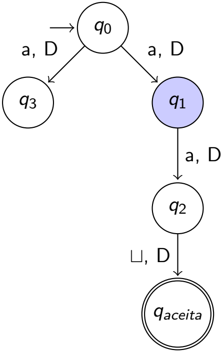




Exercício 12

Entrada : *aa*

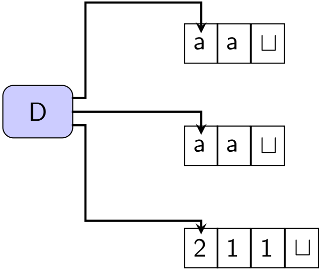


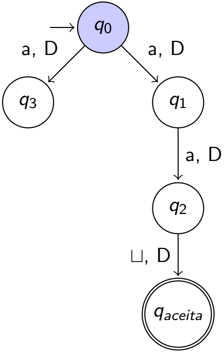


A partir de q_1, lendo o símbolo a, leva a apenas q_2. Portanto, o caminho determinado pela fita 3 está inválido. Reseta a fita 2 e fita 3 com um novo caminho em ordem lexicográfica.

Exercício 12

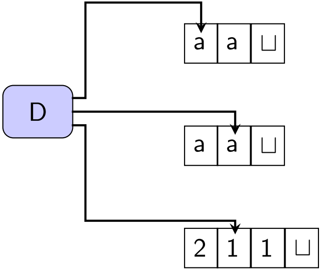
Entrada : aa

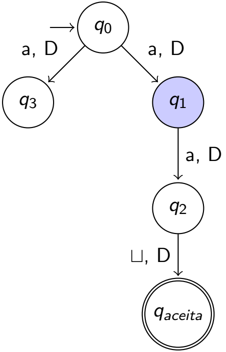




Exercício 12

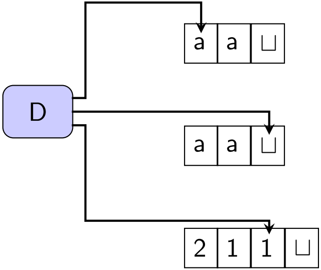
Entrada : *aa*

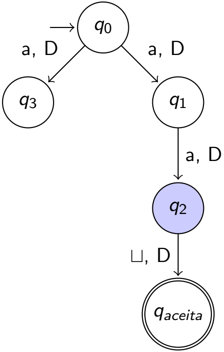




Exercício 12

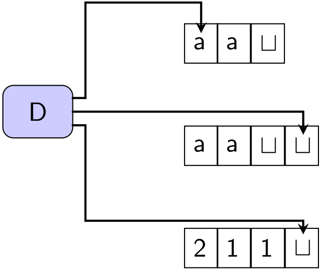
Entrada : *aa*

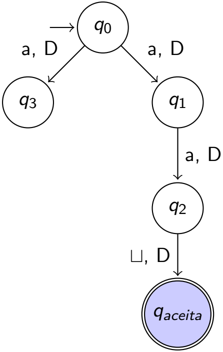




Exercício 12

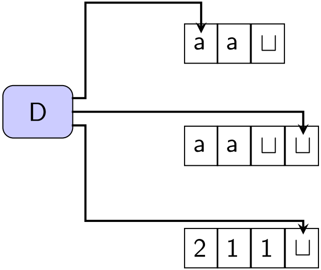
Entrada : aa

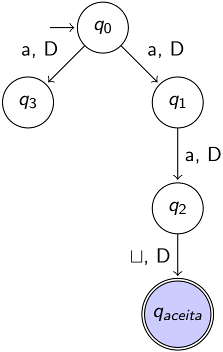




Exercício 12

Entrada : *aa*





A simulação na árvore de computação alcançou o estado de aceitação. Logo a cadeia *aa* é aceita pela máquina de Turing Não Determinística.

Curiosidade

Turing machines explained visually
https://www.youtube.com/watch?v=-ZS_zFg4w5k