PCS3616 - Laboratório 3 - Máquinas de Turing

Executar os comandos abaixo em um terminal. Em caso de dúvida ou problemas, peça ajuda.

Para colar um comando no terminal: Ctrl+Shift+V

Instalação do Graphviz e Ruby (vamos usar para gerar as máquinas de estado):

```
sudo apt-get update
sudo apt-get install -y graphviz
which dot # Saída esperada: /usr/bin/dot
sudo apt-get install ruby-dev
sudo gem install byebug --no-document
sudo gem install paint --no-documen
```

Instalação do simulador e corretor:

```
mkdir -p ~/Documents/pcs3616
cd ~/Documents/pcs3616
mkdir aula3
wget https://github.com/MiguelSarraf/pcs3616/archive/aula3.zip
unzip aula3.zip
rm aula3.zip
mv pcs3616-aula3/* aula3/
rmdir pcs3616-aula3
cd aula3/
mkdir mts
mkdir svgs
mkdir dots
```

Gerando diagrama da Máquina de Turing:

```
ruby tm_to_dot.rb mts/sua_maquina.txt > dots/sua_maquina.dot && dot
-Tsvg dots/sua_maquina.dot -o svgs/sua_maquina.svg
```

Testar um caso:

```
#Dentro de um terminal Python
#Importa a biblioteca
from turingmachine import *
#Carrega a sua maquina
load("mts/sua_maquina.txt")
#Executa um teste
run("string com fita de entrada")
```

Roda o testador padrão:

```
#Dentro de um terminal Python
#Importa a biblioteca
from turingmachine import *
#Carrega a sua maquina
load("mts/sua_maquina.txt")
#Testa os casos padrao
test("inputs/arquivo_de_teste.in")
```

ATENÇÃO: "sua_maquina.txt" não é um arquivo dentre os que você baixou, você tem que substituir esse arquivo pelo que você escrever.

OBS.: você pode criar scripts Python para realizar os passos acima, isso economiza tempo e evita typos.

Resolução dos exercícios

1. Projetar e implementar as seguintes máquinas de Turing (o arquivo da MT deve ter um dos seguintes nomes, de acordo com o exercício):

1) `mt_soma.txt`

Implementar uma MT que calcule a soma `x + y`. Veja o formato da resposta no arquivo de exemplo (`inputs/ex1-soma.in`).

2) `mt_subtracao.txt`

Implementar uma MT que calcule a soma `x - y`, com `x > y` e **tratamento de erros** (veja as observações abaixo). Veja o formato da resposta no arquivo de exemplo (`inputs/ex2-subtracao.in`).

3) `mt_soma_binaria.txt`

(DESAFIO! (Mas vale nota)) Implementar uma MT que calcule a soma `x + y` em binário, com tamanho máximo de dígitos limitado a 8. Veja o formato da resposta no arquivo de exemplo (`inputs/ex4-soma-binaria.in`).

Observações:

- Nos exercícios 1 e 2, a representação de todos os valores é em unário: `0 = 1, 1 = 11, 2 = 111, ...`
- No exercício 3, a representação de todos os valores é em binário: `0 = 0, 1 = 1, 2 = 10, 3 = 11, …`
- Todos os exercícios possuem exemplos de entrada e saída, nos arquivos do diretório **`inputs/XXXXX.in`** (por exemplo, os exemplos de execução para o exercício 1 estão em **`inputs/ex1-soma.in`**.
- Tratamento de erros: nesta aula, isso significa que a sua MT, se receber entradas inválidas para processar, não pode entrar em um loop infinito, e também não deve terminar em um estado de aceitação (estado final). A máquina deve parar em qualquer estado que não seja final (você pode, inclusive, criar um estado só para casos de erro), e o conteúdo da fita pode ser qualquer coisa (para nós, o conteúdo não é relevante se a máquina deu erro).

5. Enviar para o Sharif Judge **se estiver correto**:

5.1 Criar um arquivo zip com o arquivo da MT e o SVG do diagrama de transição, dar o nome correspondente ao exercício (e.g., "mt_soma.zip") e enviar.

6. Repetir para os demais exercícios.

Padrão do arquivo da MT:

```
ATM
PCS 2302/2024  X+Y, soma 2 números unários (formato 1.1)
1 # $      // alfabeto de entrada: $ é o início da fita, # é o separador
1 B # $  // alfabeto da fita: B é branco (default)
1    // número de fitas
1    // número de trilhas na fita 0
2    // fita 0 é infinita nas duas direções
q0  // estado inicial
q6  // estado final
q0  $  q1  $  R      // q0  - início da fita, move para a direita
q1  1  q1      1  R // q1  - se X tiver um dígito unário válido, move para a direita
q1  #  q2  1  R      // q1  - final de X, escreve 1 e move para a direita
q2  1  q2  1  R      // q2  - se Y tiver um dígito unário válido, move para a direita
q2  #  q3  B  L      // q2  - final de Y, escreve B e move para a esquerda
q3  1  q4  B  L      // q3  - último dígito de Y, escreve B e move para a esquerda
q4   1   q5   #   L      // q4   - penúltimo dígito de Y, escreve # e move para a
    esquerda
q5  1  q5  1  L      // q5  - move para esquerda até o início da fita
q5  $  q6  $  R      // q5  - início da fita, move para a direita e pára
end                      // final da máquina
```

Prólogo

Ações