UNIVERSIDADE FEDERAL DO CEARÁ - UFC
CAMPUS DE CRATEÚS
CURSO: CIÊNCIA DA COMPUTAÇÃO
DISCIPLINA: TEORIA DA COMPUTAÇÃO
PROFESSOR: RAFAEL MARTINS BARROS
ALUNO: GUSTAVO CAMPELO DE SOUSA

Tutorial de como executar a máquina de Turing a partir de um JSON

Este arquivo é um tutorial de execução do trabalho de uma máquina de Turing, que contém a seguinte configuração $M = (K, \Gamma, \delta, s, H)$, onde:

- K é o conjunto de estados da máquina, {q0, q1, q2, ..., qn};
- Γ é o alfabeto da Máquina de Turing, indicando o que ela aceitará na sua entrada.
- δ é a função de transição, que indica o estado atual e o que há na fita na posição atual do cabeçote, e o estado em que a máquina deve se mover e o que será escrito na posição atual do cabeçote;
- s é o estado inicial da Máquina de Turing;
- H é o conjunto de estados de parada da Máquina de Turing {h}

Para executar corretamente, são necessários alguns passos para a configuração. Há um arquivo JSON, que é o local onde toda a configuração da máquina é realizada, contendo todos os parâmetros citados acima. A seguir, segue como executar:

1. Criar um arquivo JSON com a definição da máquina:
   É definido um arquivo com a extensão JSON contendo o que a máquina deve conter, de acordo com sua definição formal:

```
{
  "description": "Máquina de Turing para a^n b^m (n,m >= 1)",
  "K": ["q0", "q1", "q2", "q_accept", "q_reject"],
  "Gamma": ["a", "b", "␣", "▷"],
  "branco": "␣",
  "s": "q0",
  "H": ["q_accept", "q_reject"],
  "delta": {
    "(q0,▷)": ["q0", "▷", "R"],
    "(q0,a)": ["q1", "a", "R"],
    "(q0,b)": ["q_reject", "b", "R"],
    "(q0,␣)": ["q_reject", "␣", "R"],

    "(q1,a)": ["q1", "a", "R"],
    "(q1,b)": ["q2", "b", "R"],
    "(q1,␣)": ["q_reject", "␣", "R"],
```

```
    "(q2,b)": ["q2", "b", "R"],
    "(q2,a)": ["q_reject", "a", "R"],
    "(q2,␣)": ["q_accept", "␣", "R"]
  }
}
```

Como temos uma máquina de Turing determinística, ela sempre irá parar, aceitando ou rejeitando a entrada, caso ela pertença a linguagem, por isso temos os estados de q_accept e q_reject. Gamma é o alfabeto aceito pela máquina, com seus símbolos correspondentes. Branco é o símbolo branco "␣", que representa a ausência de símbolo na fita. H é o conjunto de estados de parada da máquina. E delta é a função de transição, que representa as transições para cada entrada. É essencial que a máquina esteja configurada corretamente, senão a Máquina não executará os passos corretamente. Ela sempre parará decidindo, mas se não estiver configurada corretamente, pode aceitar ou rejeitar palavras que não deveria ocorrer tal parada.
Delta se trata da leitura (estado, caractere lido): (estado de destino, escrever na fita na posição atual do cabeçote, mover cabeçote). Lembre-se de que caso seja uma linguagem que, dependendo de como sua execução foi pensada, e se houver marcadores como X,Y,Z e eles forem necessários para marcar caracteres já visitados, estes devem estar definidos no alfabeto da fita; claro, para esse trabalho acabou sendo a forma que conseguiu-se fazer funcionar.

2. Execute o programa

O código é um só arquivo main.py, onde contém a definição da Máquina de Turing especificada de acordo com o que contém no arquivo JSON. E definimos mais algumas coisas nessa classe, como as transições que serão feitas e como os símbolos serão selecionados para que tudo seja realizado corretamente. O Python facilita muito essa questão, onde ao vir do JSON, é capaz de utilizarmos artifícios para remoção de chave e do valor que queremos. Com isso definimos a configuração inicial da fita, iniciando a fita como uma lista, a posição do cabeçote e os passos da execução em cada ação. Após isso definimos as funções necessárias: a de entrada, o passo na fita (esquerda, direita...), a execução da máquina a partir de uma quantidade de passos e a de histórico de execução no terminal da IDE, ou em qualquer outro executando o código. Há a main, que é onde fica a entrada para resgate do arquivo contendo a configuração da MT, o limite de passos a cada intervalo definido anteriormente, mas que também pode ser personalizado. Há também a informação de quando ela para aceitando e rejeitando. A certeza maior que temos, é na informação dada ao terminal de quando a MT para nos estados q_accept ou q_reject, onde podemos saber que a entrada pertence à linguagem ou não. E ao final conseguimos saber pelo seu histórico como foi esse processamento todo.
Para executar, digite no terminal
python main.py
Digite o nome do arquivo de configuração a ser utilizado para processar
Ex: *mtExponencial.json*

Digite a palavra de entrada ao ser testada (certifique de que pertence a Gamma, senão não funcionará nem para processar um passo)
Ex: *00000000 (2^3)*
Digite o número de transições (o padrão é 100, mas pode escolher mais)
Ex: *500*
Caso a máquina pare, tendo extrapolado a quantidade de passos, você pode inserir a letra "s", que ela repetirá a quantidade de passos definida no passo anterior.
Ao final do processamento, chegando em q_accept ou q_reject, você poderá visualizar todas as transições da máquina impressas no terminal.
Por exemplo, para processar 2^6, foram necessários 3137 passos. Então para algumas máquinas configuradas, é bom ter em mente definir quantos passos quererá que a máquina faça por vez.

3. Observe se a palavra é aceita ou rejeitada
   Caso a palavra seja aceita, mesmo não pertencendo à linguagem, é bem provável que haja um erro na definição da configuração da MT.
   Caso a palavra seja rejeitada, mesmo devendo pertencer à linguagem, é bem provável que haja um erro na definição da configuração da MT
   Por se basear na definição via uma espécie de tabela de transições, tende a ficar cansativo defini-la, por isso foi pensado em deixar prontos 3 arquivos com diferentes linguagens, para caso se queira testar variações, e ter a certeza de que a máquina no arquivo main.py foi definida corretamente.