Funcionamento do Simulador

O simulador de autômatos consiste em três classes como pode ser observado na Figura 1.

Figura 1: Classes do simulador de autômatos.

A entrada para o simulador é uma arquivo texto. O arquivo deve conter a especificação do autômato. Essa especificação se assemelha a definição formal de um autômato não determinístico.

O arquivo de entrada deve conter na primeira linha o conjunto de estados, cada estado deve ser separado por espaço. Na segunda linha o alfabeto, cada símbolo deve ser separado por espaço. Na próxima linha deverá ser especificado o estado inicial. A quarta linha deverá conter o conjunto de estados aceitáveis, separados por espaço. Por fim após a especificação do conjunto de estados aceitáveis, deverá ser especificado o conjunto de transições, também separadas por espaço.

O conjunto de transições deve ser especificado de acordo com a matriz M(i,j), onde a componente *i* representa o estado e a componente *j* representa o símbolo do alfabeto. A ordem das componentes é determinado conforme a especificação dos estados (primeira linha do arquivo) pelo conjunto de símbolos do alfabeto na ordem que foram especificados na segunda linha do arquivo. Por ser um autômato não determinístico, um estado pode ter transições *epsilon* (representada com o carácter "e" no alfabeto de entrada) e uma transição pode derivar para mais de um estado com o mesmo símbolo do alfabeto (representamos o conjunto de estados possíveis separados por vírgula. Ex: E1,E2,E3). Transições vazias são representadas com o carácter *"-"*.

A Figura 2 ilustra a representação do autômato no arquivo de entrada, conforme descrito. Podemos perceber que a especificação é praticamente a definição formal de autômatos não determinísticos.

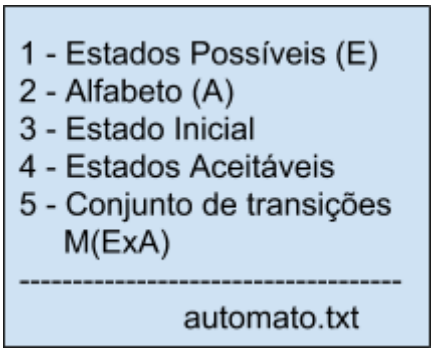

Figura 2: Representação do autômato no arquivo de entrada.

O algoritmo interpreta esse arquivo de entrada no método construtor da classe *Autômato*. Nesse método são instanciados os estados e também é definido algumas variáveis da classe *Estado*, como se o estado é inicial ou se pertence ao conjunto de estados aceitáveis. Nesse método também são instanciadas as transições para cada estado, ou seja cada estado possui uma lista de transições possíveis. Se o autômato possuir transições com epsilon ("e"), o símbolo "e" deverá ser especificado no conjunto de símbolos do alfabeto.
Realizado a construção do autômato é então instanciado o método para computar a strings de entrada. Para especificar uma string de entrada, é necessário incluí-la em um arquivo texto, no simulador o arquivo de entrada é o *"inputString.txt"*. Esse arquivo pode conter várias strings, onde cada linha representa uma *string* de entrada.

Para computar uma string de entrada o simulador inicia pelo estado inicial do autômato e verifica o conjunto de transições desse estado. Uma verificação entre o conjunto de transições do estado é realizada para identificar os próximos estados alcançáveis consumindo o primeiro símbolo da *string* (*string* de entrada especifica). Assim para cada estado alcançável o algoritmo verifica o conjunto de transições e consome um caracter da string de entrada. Quando a *string* de entrada for consumida (é vazia) é feita uma verificação para identificar se o estado presente é estado final, se for então o simulador retorna que a *string* é aceita pelo autômato, caso contrário rejeita.
Cabe ressaltar que o algoritmo que realiza a computação da *string* de entrada é recursivo, devido a característica dos autômatos não-determinísticos, um estado pode ter várias transições válidas para o mesmo símbolo do alfabeto ou ainda conter transições *epsilon.*

Para melhor explicar o funcionamento do simulador vamos considere o exemplo da linguagem L = {w | w termina em 00}. A Figura 3 ilustra o autômato para a linguagem L, considerando o alfabeto A = {0,1}.

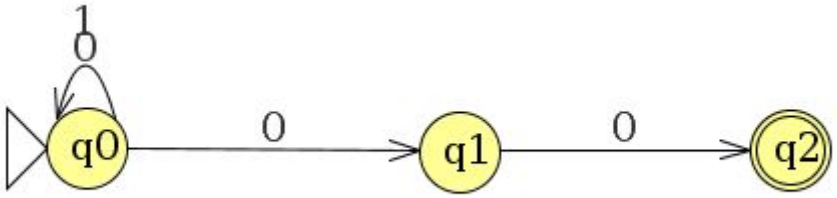

Figura 3: Autômato para linguagem L.

A descrição do arquivo de entrada para esse autômato é representado pela Figura 4.

```
q0 q1 q2
0 1
q0
q2
q0,q1 q0
q2 -
- -
```

Figura 4: Arquivo de entrada "*automato.txt*".

Define-se um conjunto de strings para teste no arquivo "*inputString.txt",* como ilustra a Figura 5.

```
0
01
00101000
1100
00
1000
-
100
001
09
```

Figura 5: Arquivo "*inputString.txt".*

Ao executar o simulador tem-se como resultado se as strings foram aceitas ou não. A Figura 6 representa a saída do simulador para o exemplo da linguagem L.

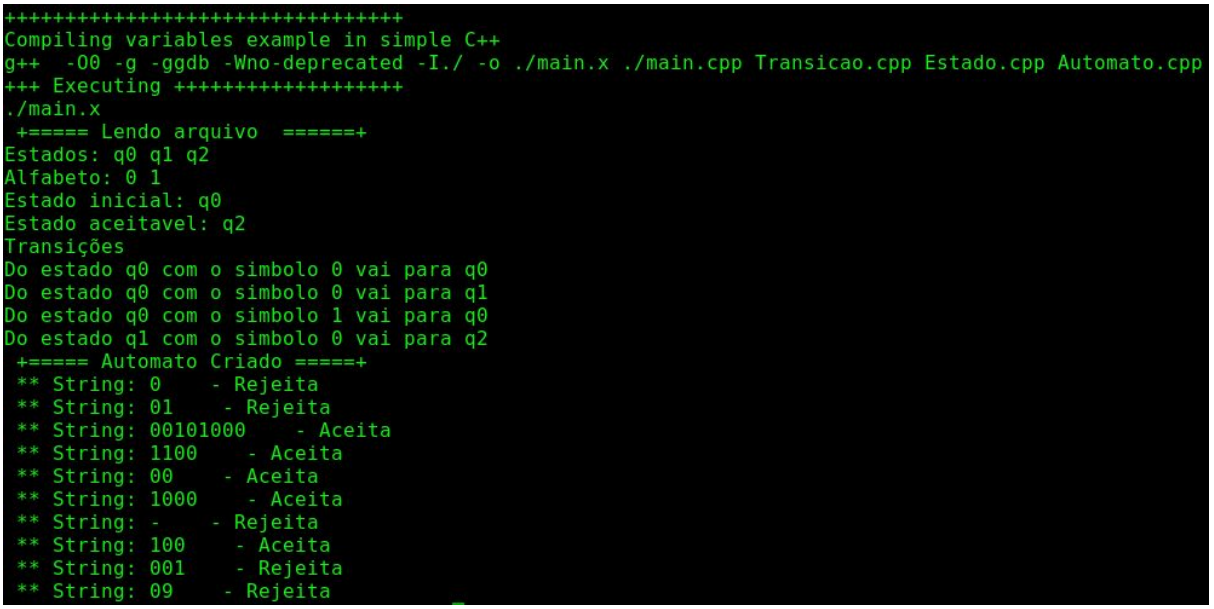

Figura 6: Resultado do simulador.