ICMC-USP

Trabalho em Grupo 2

SCC-0505

1º. Semestre de 2022
Professor: João Luís G.Rosa - e-mail: joaoluis@icmc.usp.br
Estagiário PAE: Fernando Aguiar Neto - e-mail: fernando.soares.aguiar@usp.br
01/07/2022

1 Objetivo

Desenvolver o entendimento de Linguagens Formais e seu potencial de representação através da implementação de simuladores de máquinas de Turing.

2 Descrição

O trabalho deve ser preferencialmente realizado em grupos de no máximo cinco alunos. Cada grupo deve projetar e desenvolver um *Simulador Universal de Máquinas de Turing*, empregando qualquer linguagem de programação. Um documento com instruções de execução deve ser anexado ao programa fonte, além do executável:

- *Simulador Universal de Máquinas de Turing*: O programa deve aceitar a especificação de uma máquina de Turing determinística e a partir daí para uma dada lista de cadeias, dizer quais as que pertencem (saída: **aceita**) e quais as que não pertencem (saída: **rejeita**) à linguagem reconhecida pela máquina.

3 Produto

O programa a ser implementado neste projeto deve seguir rigorosamente os formatos de entrada e saída, que será dada por teclado e monitor (stdin e stdout, ver seção "Arquivos Texto de Entrada e de Saída" abaixo), e enviado ao Escaninho de um membro do grupo na plataforma Tidia-Ae 4.0 (ae4.tidia-ae.usp.br), na página da disciplina SCC-0505, até às 23h59 do dia **1 de julho de 2022**. **O prazo final é improrrogável**. Além do programa fonte e um executável em Windows, um relatório com a descrição do trabalho deverá ser entregue (ver seção "Critérios" abaixo).

4 Critérios

Os critérios de correção dos trabalhos são:

1. (80%) **Implementação**: O programa funciona corretamente para todos os casos de teste;
2. (20%) **Documentação**: Relatório simples que explica as técnicas utilizadas para implementar a máquina escolhida. Discutir a qualidade da solução implementada, a

 Continua na próxima página...

estruturação do código e a eficiência da solução em termos de espaço e tempo. A documentação deverá ser submetida ao Tidia-Ae, juntamente com o código fonte e um arquivo executável em Windows. **IMPORTANTE**: Incluir explicações claras sobre como executar o programa em um arquivo texto de nome **manualT2.txt**. Portanto, quatro arquivos devem ser anexados: o relatório (arquivo txt ou PDF), o arquivo fonte (arquivo txt), executável e manual.

Atenção: **O plágio (cópia) de programas não será tolerado. Quaisquer programas similares terão nota zero independente de qual for o original e qual for a cópia.**

5 Arquivos Texto de Entrada e de Saída

Arquivo Texto de Entrada:

- 1^a. Linha: número de estados: para o conjunto de estados Q, assume-se os nomes dos estados de q_0 a q_{n-1}, onde n é o número de estados (Obs.: q_0 é o estado inicial). Portanto, basta entrar com o número de estados. Assuma $1 \leq n \leq 10$;
- 2^a. Linha: o conjunto de símbolos terminais (Σ): entrar com a quantidade de símbolos terminais seguida dos elementos separados por espaço simples. Assume-se tamanho máximo igual a 10;
- 3^a. Linha: a quantidade de símbolos de Σ' (alfabeto estendido de fita) não presentes em Σ, seguido pelos símbolos, separados por espaço simples;
- 4^a. Linha: o estado de aceitação: entrar com o estado de aceitação (q_a). Lembre-se de entrar apenas com os números de 0 a 9;
- 5^a. Linha: o número de transições (δ) da máquina (máximo de 50).
- a partir da 6^a Linha: as transições: entra-se com um δ em cada linha, com os elementos separados por espaço: $q\ x\ q'\ y\ D$, onde $q, q' \in Q$, $x, y \in \Sigma'$ e $D \in \{R, L, S\}$. Assuma os limites da fita como o símbolo branco (B). Represente a cadeia vazia (λ) como "-".
- Linha depois das transições: entrar com o número de cadeias de entrada (máximo de 10).
- Próximas Linhas: cadeias de entrada: entrar com uma em cada linha. Comprimento máximo de cada cadeia = 20 símbolos.

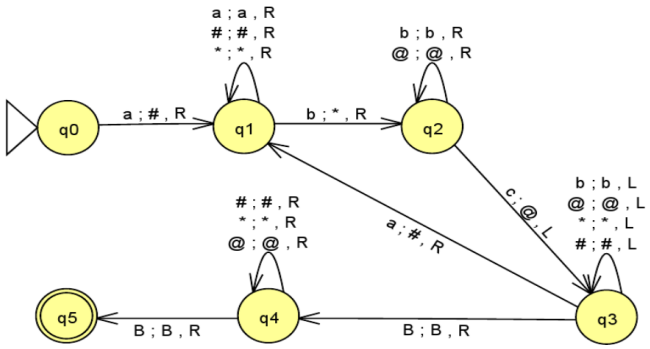
Arquivo Texto de Saída:

 Continua na próxima página...

- a partir da 1^a. Linha: a informação sobre a aceitação ou não da respectiva cadeia de entrada, **na ordem** do arquivo de entrada. Se a cadeia de entrada pertencer à linguagem reconhecida pelo autômato, a cadeia de saída será "aceita". Caso a cadeia de entrada não pertença à linguagem reconhecida pelo autômato, a cadeia de saída será "rejeita".

6 Exemplo

- Exemplo: Máquina de Turing determinística que processa a linguagem $a^n b^n c^n$, com $n > 0$.



Arquivo Texto de Entrada[1]:

1. 6
2. 3 a b c
3. 4 # * @ B
4. 5
5. 18
6. 0 a 1 # R
7. 1 # 1 # R
8. 1 a 1 a R
9. 1 * 1 * R
10. 1 b 2 * R
11. 2 b 2 b R
12. 2 @ 2 @ R

[1]Os números das linhas **não** devem aparecer na entrada. Estão colocados aqui apenas para facilitar o entendimento.

 Continua na próxima página...

13. 2 c 3 @ L
14. 3 # 3 # L
15. 3 * 3 * L
16. 3 @ 3 @ L
17. 3 b 3 b L
18. 3 a 1 # R
19. 3 B 4 B R
20. 4 # 4 # R
21. 4 * 4 * R
22. 4 @ 4 @ R
23. 4 B 5 B R
24. 10
25. abbcca
26. aabbcc
27. bac
28. aaabbbcccc
29. -
30. abcabc
31. abc
32. abcc
33. c
34. aaabbbbccc

 Continua na próxima página...

Arquivo Texto de Saída:

1. rejeita
2. aceita
3. rejeita
4. rejeita
5. rejeita
6. rejeita
7. aceita
8. rejeita
9. rejeita
10. rejeita

7 Notas

1. Apenas máquinas determinísticas serão testadas;
2. Apenas linguagens recursivas serão testadas, ou seja, linguagens para as quais a máquina de Turing sempre para independentemente da aceitação;
3. O processamento da cadeia começa no estado q_0 e no símbolo mais à esquerda de w. Exemplo: para a fita $abba$, a descrição instantânea inicial é q_0abba;
4. Arquivos de exemplo estão disponíveis. Para testar, utilizem o arquivo de entrada como input para o programa via stdin (usando $<$), redirecionem a saída para um outro arquivo (usando $>$) e comparem com o arquivo de exemplo de saída. Exemplo no windows:
   main.exe < exemploIn.txt > out.txt
   fc out.txt exemploOut.txt