



UNIVERSIDADE ESTADUAL DO PARANÁ - CAMPUS APUCARANA

Lissa Guirau Kawasaki

RELATÓRIO TÉCNICO - MÁQUINA DE TURING

Linguagens Formais, Autômatos e Computabilidade



APUCARANA – PR

2024

Lissa Guirau Kawasaki

RELATÓRIO TÉCNICO - MÁQUINA DE TURING

Linguagens Formais, Autômatos e Computabilidade

Trabalho apresentado à disciplina de
Linguagens Formais, Autômatos e
Computabilidade do curso de Bacharelado
em Ciência da Computação.

Professor: Guilherme Henrique de Souza
Nakahata.

APUCARANA – PR

2024

SUMÁRIO

INTRODUÇÃO	04
CAPÍTULO 1: OBJETIVOS	05
CAPÍTULO 2: MOTIVAÇÃO E RECURSOS UTILIZADOS	06
2.1 Motivação	06
2.3 Linguagem de programação e demais informações	06
2.2 Estrutura de Dados	06
CAPÍTULO 3: RESULTADOS	11
CONCLUSÃO	14
REFERÊNCIAS	11

INTRODUÇÃO

A área relacionada a Linguagens Formais, Autômatos e Computabilidade se destaca como fundamental para a compreensão dos princípios de processamento e manipulação da informação. Essa área engloba conceitos abstratos que permitem a análise de linguagem, de comportamento de máquinas e os limites da computação. As Linguagens Formais servem como a base para a construção de sistemas computacionais. Dentro desta área, encontramos a Hierarquia de Chomsky, a classificação, numérica de 0 a 4, das linguagens formais pelas suas restrições, complexidades e liberdade em suas regras. Conhecida como tipo-0 na hierarquia, a linguagem recursivamente enumerável é uma linguagem formal para a qual existe uma máquina de Turing que irá parar e aceitar quando a entrada de um usuário pertencer à linguagem apresentada e que pode parar e rejeitar ou ficar em um loop infinito quando a entrada não pertencer à linguagem.

Uma máquina de Turing (MT) é constituída por conjuntos finitos de estados, uma fita dividida em células, e uma cabeça que pode realizar a leitura e escrita da entrada de um usuário, que atua sobre a célula da fita, analisando uma posição de cada vez. Foi desenvolvida secretamente para o Governo Inglês, durante a Segunda Guerra Mundial, por Alan Turing, que conseguiu decifrar a máquina criptográfica sofisticada criada pelos Alemães, para manterem as suas comunicações secretas. Para alcançar este feito, Turing concebeu teoricamente a “máquina de Turing”, que lhe permitiu desenvolver uma máquina computacional que decifrava a famosa “Enigma”, máquina de criptografia alemã, isto após os aliados terem conseguido capturar um desses equipamentos.

OBJETIVOS

Alan Turing apresentou ao mundo um modelo teórico: a máquina de Turing, esta criação estabeleceu as bases da computação moderna e definiu os princípios básicos da programação, podendo simular qualquer outra máquina computacional, da mais simples à mais complexa.

Quando programamos uma máquina de Turing, na prática, estamos formando a lógica de programação que permite à máquina realizar uma tarefa específica, realizada através de um conjunto de instruções detalhadas que determinam como movimentar a cabeça da fita, quais símbolos devem ser lidos, e escreva e em que condições deve mudar de estado. Dessa forma, podemos visualizar passo a passo como os algoritmos funcionam, entender os mecanismos internos e abrir caminho para estudos em áreas influenciadas por máquinas, como matemática, criptografia e inteligência artificial. O principal objetivo do código-fonte da Máquina de Turing (MT) é interpretar as características de um conjunto de estados que definem a máquina, o que lhe permite avaliar e aceitar palavras de acordo com as regras linguísticas especificadas nas transições. O código-fonte da MT funciona em duas frentes principais: análise da linguagem, leitura de transições, validação de símbolos e palavras, análise e manipulação da fita, leitura e escrita de símbolos de acordo com as instruções de transição, modificação da fita de acordo com as regras da linguagem, adicionar, remover e substituir símbolos durante o processo de análise. Ao interpretar as transições e controlar com precisão as operações, o código-fonte permite que a MT execute suas tarefas com eficiência e precisão.

MOTIVAÇÃO E RECURSOS UTILIZADOS

2.1 Motivação.

No mundo da informática, a máquina de Turing se tornou uma ferramenta essencial, e programar uma máquina de Turing pode servir como uma base sólida para o aprendizado de linguagens de programação reais. Através da experiência prática com conceitos como estados, transições e símbolos, é possível dominar sintaxe e estruturas, descobrir habilidades de manipulação de símbolos para criptografar e descriptografar mensagens e desenvolver compiladores e modelos de computador desde autômatos finitos até redes neurais artificiais.

Dominar o código-fonte da Máquina de Turing significa uma compreensão profunda da complexidade da análise de linguagem e da manipulação de fitas, o que nos permite explorar as áreas da computação, aprimorar nossas habilidades, compreender e dominar uma ferramenta fundamental para o mundo moderno.

2.2 Linguagem de Programação.

Para o código, foi utilizado a linguagem de programação Java, que se destaca como uma linguagem de programação robusta, multiplataforma e orientada a objetos, oferecendo uma base sólida para o desenvolvimento de softwares de alta qualidade e escaláveis. Sua versatilidade, robustez e facilidade de manipulação a tornam uma linguagem pertinente acerca do projeto apresentado em questão.

2.3 Estrutura de Dados.

- Classe Fita:

A classe Fita é responsável por representar e gerenciar a fita da Máquina de Turing. Seus métodos permitem que a máquina leia, escreva e navegue na fita, facilitando a implementação e o uso. Ela possui os seguintes elementos:

Variáveis de Instância:

- Fita: Um array de caracteres que armazena os símbolos na fita.
- PosicaoCabeca: Uma variável inteira que indica a posição atual da cabeça de leitura/escrita na fita.

Construtores:

- Fita (Entrada do Usuário) Inicializa a fita com a palavra de entrada fornecida, e adiciona espaços em branco para evitar que a cabeça saia dos limites da fita. A posição da cabeça é inicializada no índice 0.

Métodos:

- lerSimbolo(): Retorna o símbolo na posição atual da cabeça.
- escreverSimbolo(char símbolo): Escreve o símbolo fornecido na posição atual da cabeça.
- moverCabeca(char direção): Move a cabeça de leitura/escrita para a direita ('R') ou para a esquerda ('L').
- posicaoValida(): Verifica se a posição da cabeça está dentro dos limites da fita.
- toString(): Retorna a fita em formato de string, destacando a posição da cabeça com colchetes.

- **Classe Transição.**

A classe Transição representa uma transição em uma Máquina de Turing. Ela encapsula as regras necessárias para definir como a máquina deve se comportar em um determinado estado, quando lê um símbolo específico na fita. A classe possui os seguintes elementos:

Variáveis de Instância:

- Próximo Estado: Uma string que armazena o nome do próximo estado para o qual a máquina deve ler.
- Símbolo Fita: Um caractere que representa o símbolo que a máquina deve substituir na fita para que a transição seja aplicável.
- Direção: Um caractere que indica para onde a cabeça de leitura/escrita da máquina deve se mover após a transição ser aplicada.

Construtor:

- Transição(String proximoEstado, char simboloFita, char direcao): Inicializa uma transição com os valores fornecidos para o próximo Estado, o símbolo da fita e a direção.

Métodos Getters:

As variáveis de instância da classe são privadas, o que significa que só podem ser acessadas através dos métodos getters. Dessa forma, os métodos getters permitem que o código externo acesse os valores das variáveis de instância de uma transição.

- getProximoEstado(): Retorna o nome do próximo estado para o qual a máquina deve mudar.
- getsimboloFita(): Retorna o símbolo que a máquina deve substituir na fita para que a transição seja aplicável.
- getDireção(): Retorna o caractere que indica para onde a cabeça de leitura/escrita da máquina deve se mover após a transição ser aplicada.

Ao armazenar informações sobre o próximo estado, o símbolo a ser lido e a direção do movimento da cabeça, a classe Transição permite que a máquina navegue na fita e modifique seus estados de acordo com as regras definidas.

- **Classe MáquinaDeTuring**

A classe `MaquinaDeTuring` é a principal classe para a execução da Máquina de Turing, encapsulando todos os elementos necessários para simular a execução desejada. Ela possui os seguintes atributos:

Variáveis de Instância:

- Estados: Uma lista que armazena os estados inseridos pelo usuário.
- Símbolos: Uma lista que armazena os símbolos permitidos na fita..
- Símbolos Fita: Uma lista que armazena os símbolos adicionais utilizados na fita..
- Estado Inicial: Uma string que indica o estado inicial..
- Estados Finais: Uma lista que armazena os estados finais.
- Função Transição: Uma matriz bidimensional que armazena as transições da Máquina de Turing. Cada elemento da matriz representa uma transição e contém informações sobre o próximo estado, o símbolo a ser escrito e a direção do movimento da cabeça.
- Marcador Início: Um caractere que representa o marcador de início e fim da fita da Máquina de Turing.

Construtor:

- `MaquinaDeTuring(ArrayList<String> estados, ArrayList<Character> simbolos, ArrayList<Character> simbolosFita, String estadoInicial, ArrayList<String> estadosFinais, Transicao[][] funcaoTransicao, Character marcadorInicio)`: Inicializa a Máquina de Turing com os valores fornecidos para todos os seus atributos.

Método `executarMT()`:

- Este método é o principal responsável por executar a Máquina de Turing. Ele realiza as seguintes etapas:
 1. Lê uma palavra do usuário através da entrada padrão.
 2. Inicializa a fita com a palavra de entrada e o marcador de início.
 3. Define o estado atual como o estado inicial.

4. Entra em um loop principal que executa as seguintes ações a cada iteração:

- Lê o símbolo atual na fita.
- Obtém os índices do estado atual e do símbolo atual nas listas de estados e símbolos da fita.
- Recupera a transição correspondente na matriz `funcaoTransicao`.
- Verifica se os índices obtidos e a transição são válidos.
- Se todas as condições forem válidas, imprime-se as informações sobre o estado atual, o símbolo atual, a fita, e a próxima transição, escreve o símbolo da transição na fita, move a cabeça de leitura/escrita na direção indicada pela transição, atualiza o estado atual para o próximo estado da transição e verifica se o estado atual é um estado final. Ao atingir o estado final, concluímos o funcionamento da Máquina de Turing, imprimindo uma mensagem informando que a palavra foi aceita e mostrando a fita final. Se algum dos passos anteriores falhar, imprime-se uma mensagem informando que a palavra não foi aceita.

5. O loop principal termina quando a palavra é aceita, a cabeça de leitura/escrita sai dos limites da fita, ou o estado atual não é um estado final e nenhuma transição válida é encontrada.

- **Adições tardias.**

Devido ao erro `java.lang.ArrayIndexOutOfBoundsException: Index -1 out of bounds for length`, que ocorre quando a cabeça se move para o índice -1, que está fora dos limites válidos do array. Para resolver esse problema, é necessário garantir que a cabeça de leitura nunca se mova para uma posição negativa, modificando o método `moverCabeça` para evitar que a cabeça se mova para fora dos limites e adicionando um método para expandir a fita, dessa forma, caso ocorra um movimento fora dos limites da fita, é possível acomodar o movimento.

RESULTADO

Com os procedimentos realizados anteriormente, o pleno funcionamento completo do código foi alcançado conforme o esperado - Para melhor visualização dos resultados, é utilizado a seguinte Máquina de Turing:

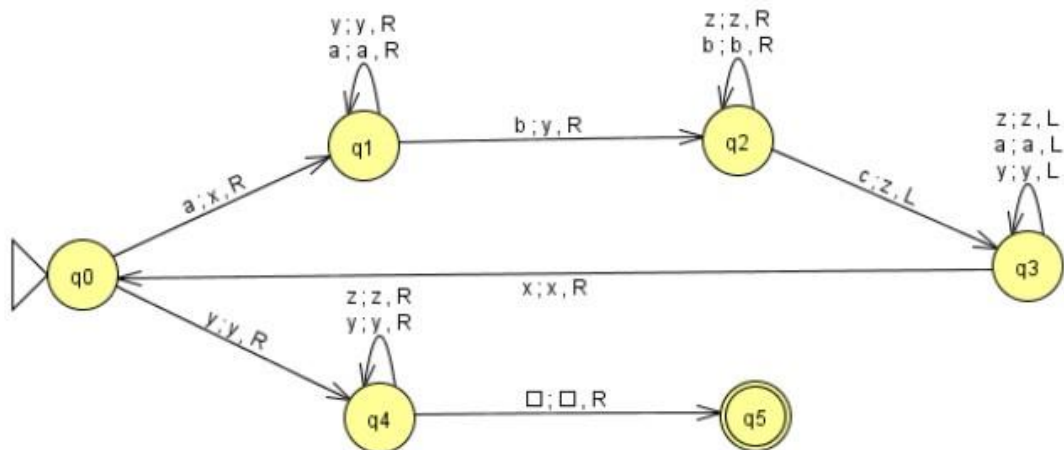


Figura 1, Máquina de Turing que aceita $L = \{a^n b^n c^n \mid n \geq 1\}$.

Primeiramente, é realizada a entrada do usuário, utilizando da definição formal da Máquina de Turing.

```

Informe o estado inicial: q0
Informe os estados finais (separados por espaço): q5
Informe o conjunto de estados (separados por espaço): q0 q1 q2 q3 q4 q5
Informe os símbolos de entrada (separados por espaço): a b c
Informe os símbolos da fita, incluindo o símbolo branco. (separados por espaço): a b c x y z _
Informe o marcador de início: <
  
```

Figura 2.

A definição formal é definida por:

- Q é um conjunto finito de estados
- Σ é um alfabeto finito de símbolos
- $\Gamma^i, i = 1, \dots, k$ é o alfabeto da fita i (conjunto finito de símbolos)
- k é o número de fitas
- $s \in Q$ é o estado inicial
- $b \in \Gamma$ é o símbolo branco
- $F \subseteq Q$ é o conjunto dos estados finais
- $\delta : Q \times \Gamma \longrightarrow Q \times \Gamma \times \{\leftarrow, \rightarrow\}$ é uma função parcial chamada função de transição, onde \leftarrow é o movimento para a esquerda e \rightarrow é o movimento para a direita.

Em seguida, é preenchida as transições, também pelo usuário.

```
Preencha as transições:
? (q0,a): q1,x,R
? (q0,b): X
? (q0,c): X
? (q0,x): X
? (q0,y): q4,y,R
? (q0,z): X
? (q0,_): X
? (q1,a): q1,a,R
? (q1,b): q2,y,R
? (q1,c): X
? (q1,x): X
? (q1,y): q1,y,R
? (q1,z): X
? (q1,_): X
? (q2,a): X
? (q2,b): q2,b,R
? (q2,c): q3,z,L
? (q2,x): X
? (q2,y): X
? (q2,z): q2,z,R
? (q2,_): X
? (q3,a): q3,a,L
? (q3,b): q3,b,L
? (q3,c): X
? (q3,x): q0,x,R
? (q3,y): q3,y,L
? (q3,z): q3,z,L
? (q3,_): X
? (q4,a): X
? (q4,b): X
? (q4,c): X
? (q4,x): X
? (q4,y): q4,y,R
? (q4,z): q4,z,R
? (q4,_): q5_,R
? (q5,a): X
? (q5,b): X
? (q5,c): X
? (q5,x): X
? (q5,y): X
? (q5,z): X
? (q5,_): X
```

Figura 3.

Concluindo, o programa pede ao usuário que insira uma palavra a ser verificada, dessa forma, retornando a validade da palavra inserida baseada nas informações previamente preenchidas. Verificaremos a palavra aabbcc.

```

Insira a palavra a ser verificada: aabbcc -----
Estado atual: q0                               Estado atual: q1
Símbolo atual: a                               Símbolo atual: b
Fita: [a]abbcc                                Fita: xxy[b]zc
Próxima transição: (q0,a) -> (q1,x,R)        Próxima transição: (q1,b) -> (q2,y,R)
-----
Estado atual: q1                               Estado atual: q2
Símbolo atual: a                               Símbolo atual: z
Fita: x[a]bbcc                                Fita: xxyy[z]c
Próxima transição: (q1,a) -> (q1,a,R)        Próxima transição: (q2,z) -> (q2,z,R)
-----
Estado atual: q1                               Estado atual: q2
Símbolo atual: b                               Símbolo atual: c
Fita: xa[b]bcc                                Fita: xxyyz[c]
Próxima transição: (q1,b) -> (q2,y,R)        Próxima transição: (q2,c) -> (q3,z,L)
-----
Estado atual: q2                               Estado atual: q3
Símbolo atual: b                               Símbolo atual: z
Fita: xay[b]cc                                Fita: xxyy[z]z
Próxima transição: (q2,b) -> (q2,b,R)        Próxima transição: (q3,z) -> (q3,z,L)
-----
Estado atual: q2                               Estado atual: q3
Símbolo atual: c                               Símbolo atual: y
Fita: xayb[c]c                                Fita: xxy[y]zz
Próxima transição: (q2,c) -> (q3,z,L)        Próxima transição: (q3,y) -> (q3,y,L)
-----
Estado atual: q3                               Estado atual: q3
Símbolo atual: b                               Símbolo atual: y
Fita: xay[b]zc                                Fita: xx[y]yzz
Próxima transição: (q3,b) -> (q3,b,L)        Próxima transição: (q3,y) -> (q3,y,L)
-----
Estado atual: q3                               Estado atual: q3
Símbolo atual: y                               Símbolo atual: x
Fita: xa[y]bzc                                Fita: x[x]yyzz
Próxima transição: (q3,y) -> (q3,y,L)        Próxima transição: (q3,x) -> (q0,x,R)
-----
Estado atual: q3                               Estado atual: q0
Símbolo atual: a                               Símbolo atual: y
Fita: x[a]ybzc                                Fita: xx[y]yzz
Próxima transição: (q3,a) -> (q3,a,L)        Próxima transição: (q0,y) -> (q4,y,R)
-----
Estado atual: q3                               Estado atual: q4
Símbolo atual: x                               Símbolo atual: y
Fita: [x]aybzc                                Fita: xxy[y]zz
Próxima transição: (q3,x) -> (q0,x,R)        Próxima transição: (q4,y) -> (q4,y,R)
-----
Estado atual: q0                               Estado atual: q4
Símbolo atual: a                               Símbolo atual: z
Fita: x[a]ybzc                                Fita: xxyy[z]z
Próxima transição: (q0,a) -> (q1,x,R)        Próxima transição: (q4,z) -> (q4,z,R)
-----
Estado atual: q1                               Estado atual: q4
Símbolo atual: y                               Símbolo atual: z
Fita: xx[y]bzc                                Fita: xxyyz[z]
Próxima transição: (q1,y) -> (q1,y,R)        Próxima transição: (q4,z) -> (q4,z,R)

```

Figura 4.

```

-----
Estado atual: q4
Símbolo atual: _
Fita: xxyyzz[_]_____
Próxima transição: (q4,_) -> (q5,_,R)
-----
Palavra aceita!

Fita início:
<aabbcc<
Fita final:
<xxyyzz[_]_____

```

Figura 5.

CONCLUSÃO

A partir dos testes realizados acima, tomamos as seguintes conclusões:

O programa foi capaz de concluir sua proposta de forma satisfatória. Ao utilizarmos Java em sua aplicação, foi possível visualizar de forma desconstruída a lógica de programação por trás da implementação de uma Máquina de Turing, tendo sucesso em exemplificar o funcionamento e reconhecer linguagens especificadas pelo usuário. O programa não apenas funcionou como esperado, mas também serviu como um exemplo prático para ilustrar o funcionamento dessas máquinas e sua capacidade de reconhecer linguagens definidas pelo usuário.

Ao longo do trabalho, foi possível explorar de forma prática e lógica as Máquinas de Turing, ultrapassando a barreira de conceitos abstratos. Assim como dito anteriormente, as Máquinas de Turing e a teoria das linguagens formais desempenham um papel crucial na ciência da computação, especialmente na modelagem e análise de sistemas computacionais. As Máquinas de Turing são utilizadas para representar e verificar o comportamento de programas de software e hardware, garantindo que eles funcionem conforme o esperado. Através do projeto, foi possível confirmar sua essencialidade na área e seus estudos.

REFERÊNCIA

POZZA, O.; PENEDO, S. A MÁQUINA DE TURING. [s.l: s.n.]. Disponível em:
<<https://www.inf.ufsc.br/~j.barreto/trabaluno/MaqT01.pdf>>.

BRANDÃO, P. Alan Turing: da necessidade do cálculo, a máquina de Turing até à computação. **Revista de Ciências da Computação**, p. 73–88, 2017.