Desenvolvimento de um software Simulador de Máquina de Turing usando a plataforma Java Standard Edition

Leandro Aparecido de Almeida

Resumo: Este artigo apresenta um software simulador de Máquina de Turing implementado em linguagem de programação Java usando a plataforma Java Standard Edition (Java SE). O software permite a definição dos parâmetros para a construção da Máquina de Turing via interface gráfica de usuário e também via programação, por meio de uma linguagem desenvolvida especificamente para a descrição de uma Máquina de Turing para o simulador.

Palavras-chave: Máquinas universais; Computabilidade; Máquina de Turing.

Abstract: This article presents a Turing Machine simulator software implemented in Java programming language using the Java Standard Edition (Java SE) platform. The software allows the definition of parameters for the construction of the Turing Machine via a graphical user interface and also via programming, through a language developed specifically for the description of a Turing Machine for the simulator.

Keywords: Universal machines; Computability; Turing machine.

1. Introdução

Em 1936 o matemático e lógico britânico Alan Mathison Turing (1912 – 1954) definiu matematicamente o conceito moderno de computador e estabeleceu e simplificou os limites da computabilidade, publicando a sua ideia nas edições de 30 de novembro e 23 de dezembro do *Proceedings of the London Mathematical Society* com o título *On Computable Numbers, with an Application to the Entscheidungsproblem (Sobre Números Computáveis, com uma Aplicação ao Entscheidungsproblem)*. O artigo é uma descrição matemática do que

ele denominou de Máquina Universal – uma abstração que poderia, em princípio, resolver qualquer problema matemático que fosse apresentado a ela de forma simbólica. Hoje o dispositivo é conhecido como Máquina de Turing, termo cunhado pelo matemático norte-americano Alonzo Church (1903 – 1995) em 1936, professor de Turing durante seu doutorado na Universidade de Princeton (1936 – 1938).

A Máquina de Turing é um dispositivo imaginário, projetado para replicar os "estados mentais" matemáticos e as habilidades de manipulação de símbolos de um computador humano. Na época de Turing, o trabalho de calcular ainda era realizado manualmente por funcionários matemáticos humanos, como ilustrado na **Fig. 1**. Esses trabalhadores eram chamados de computadores – daí o termo computador humano.

Fig. 01: *Computadores de Harvard. Um grupo de mulheres contratadas pelo astrônomo Edward Charles Pickering. diretor do Harvard Observatory, para processar dados astronômicos. Fotografia de 1890.*
Fonte: *Harvard College Observatory / Wikimedia Commons.*

Turing concebeu seu dispositivo para responder a última das três questões sobre matemática colocadas pelo matemático alemão David Hilbert (1862 – 1943) no Congresso Internacional de Matemáticos realizado em Bolonha, na Itália, no ano de 1928. As questões eram as seguintes:

1. A matemática é completa, ou seja, a todo enunciado verdadeiro expresso no próprio sistema cabe uma prova formalizada no sistema?

2. A matemática é consistente, isto é, impossibilita a derivação do absurdo?

3. A matemática é decidível, ou seja, é possível encontrar um mecanismo genérico e finitário que, ao considerar um enunciado qualquer do sistema formal, no caso, a Lógica de Primeira Ordem, fosse capaz de verificar sua validade ou não?

A terceira questão, também conhecida como *Entscheidungsproblem (problema de decisão),* propunha encontrar um método efetivo (diz-se mecânico ou algorítmico) com o qual, dada uma fórmula da linguagem do cálculo de predicados, se determina se essa fórmula é, ou não, um teorema da lógica, ou seja, deduzível apenas a partir dos axiomas do cálculo de predicados. Um método ou procedimento é efetivo se:

1. Puder ser descrito através de um número finito de instruções exatas;

2. Produzir o resultado desejável ao fim de um número finito de passos (desde que se sigam as instruções sem erro);

3. Puder, em princípio, ser executado por um ser humano apenas com a ajuda de papel e lápis;

4. Não exigir nem criatividade nem perspicácia por parte do ser humano.

Os algoritmos que as crianças aprendem para efetuar as operações básicas da aritmética são exemplos de procedimentos efetivos.

O matemático tcheco Kurt Gödel (1906 – 1978) já havia demonstrado que a aritmética (e, por extensão, a matemática) era ao mesmo tempo inconsistente e incompleta, respondendo as duas primeiras questões levantadas por Hilbert. Turing demonstrou com sua máquina universal que a matemática também era indecidível, resolvendo a terceira questão ou *Entscheidungsproblem*. Ao fazer isto, ele desenvolveu uma formalização de algoritmo, um conceito fundamental para a computação.

Trabalhando independentemente de Turing, Alonzo Church também já tinha demonstrado a indecidibilidade da matemática usando seu método chamado de Lambda-Cálculo (λ-cálculo), publicando o resultado em seu artigo *A Note on the Entscheidungsproblem (Uma nota sobre Entscheidungsproblem)* alguns meses antes de Turing. Também o matemático e lógico polonês-americano Emil Leon Post (1897 – 1954), desenvolveu e publicou em outubro de 1936 um modelo matemático de computação que era essencialmente equivalente à máquina de Turing. Este modelo é chamado de Máquina de Post ou máquina de Post-Turing.

Neste artigo descrevo um simulador de Máquina de Turing, desenvolvido por mim como um complemento ao estudo da disciplina de Linguagens Formais e Autômatos. O simulador desenvolvido implementa dois modelos específicos. O primeiro deles é um modelo padrão, de apenas uma fita e uma única cabeça de leitura, e o segundo é um modelo com múltiplas fitas e múltiplas cabeças de leitura, uma para cada fita. Vou demonstrar também a equivalência entre os dois modelos, executando um programa gerado para uma máquina com múltiplas fitas em uma máquina com uma única fita e vice-versa, provando que um modelo mais elaborado pode ser simulado pelo modelo padrão, ou seja, que um modelo mais elaborado não é computacionalmente mais poderoso que um modelo padrão, de fita única.

2. Modelo de Máquina de Turing

Como visto na seção anterior, a Máquina de Turing visa replicar uma pessoa calculando manualmente algum tipo de problema. O dispositivo deve simular a situação na

qual esta pessoa, munida de um equipamento de escrita e um apagador, realiza cálculos em uma folha de papel, organizada em quadrados.

Suponha que inicialmente a folha de papel contenha somente os dados iniciais do problema. O trabalho realizado pela pessoa sobre esta folha de papel pode ser resumido em sequências de operações muito simples como:

- ✔ Ler um símbolo de um quadrado;
- ✔ Alterar um símbolo de um quadrado;
- ✔ Mover os olhos para outro quadrado.

Quando é encontrada alguma condição satisfatória para a resposta desejada, a pessoa termina seus cálculos.

O modelo proposto por Turing parte então de algumas premissas:

- ✔ Pode ser assumido que o papel consiste de uma única fita infinita, organizada em quadrados;
- ✔ O conjunto de símbolos pode ser finito, pois se pode utilizar sequências de símbolos;
- ✔ O conjunto de estados da mente da pessoa durante o processo de cálculo é finito. Entre estes estados, existem dois em particular: o estado inicial, correspondendo ao início dos cálculos, e o estado final, definindo o fim dos cálculos.
- ✔ O comportamento da pessoa a cada momento é determinado somente pelo seu estado atual e pelo símbolo para o qual sua atenção está voltada;

✔ A pessoa é capaz de observar e alterar o símbolo de apenas um quadrado de cada vez, bem como de transferir sua atenção somente para um dos quadrados adjacentes.

Este processo resulta numa máquina constituída de três partes:

1. **Fita.** Usada simultaneamente como dispositivo de entrada, de saída e de memória de trabalho;

2. **Unidade de Controle.** Reflete o estado corrente da máquina. Possui uma unidade de leitura e gravação (Cabeça de Leitura/Escrita), a qual acessa uma célula da fita de cada vez e movimenta-se para a esquerda ou para a direita;

3. **Programa ou Função de Transição.** Função que define o estado da máquina e comanda as leituras, as gravações e o sentido de movimento da Cabeça de Leitura/Escrita.

A fita é finita à esquerda e infinita à direita, sendo que cada uma de suas células armazena um único símbolo. Os símbolos podem pertencer ao alfabeto de entrada, ao alfabeto auxiliar ou ser algum dos símbolos "especiais", como o branco (β) e o marcador de início da fita (⊛), representando uma célula em branco, ou vazia e a célula mais à esquerda da fita, respectivamente. Em um primeiro momento, a palavra a ser processada ocupa as células mais à esquerda, logo adiante do marcador de início da fita. As demais células receberão um marcador branco.

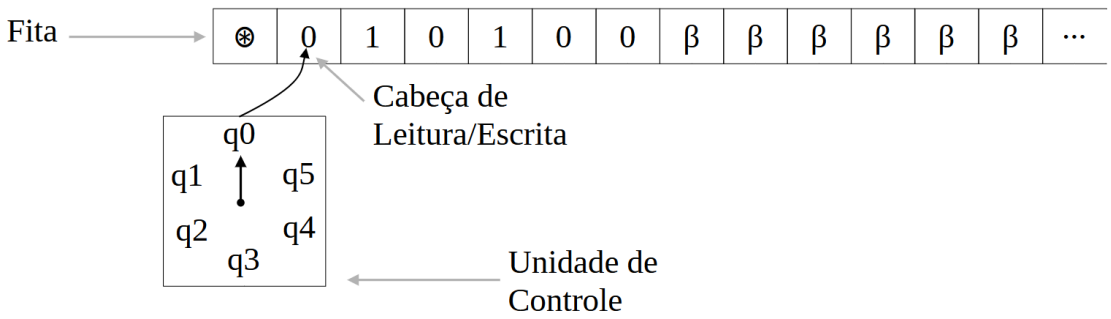

Fig. 2: *Representação de uma máquina de Turing. A fita é finita à esquerda e infinita à direita. A Unidade de Controle controla as transições de estado, a escrita de um novo símbolo e o movimento da Cabeça de Leitura/Escrita. A Cabeça pode andar apenas uma célula por vez, indo para a direita ou para a esquerda, de acordo com o definido na função de transição. A primeira célula da fita recebe um símbolo de marcador de início, logo adiante vem os símbolos da cadeia de entrada (os 0's e 1's), quando o processamento vai ser iniciado. As demais casas da fita recebem um marcador de branco.*

A Unidade de Controle possui um número finito e pré-definido de estados. Ela controla a leitura/gravação da fita pela Cabeça de Leitura/Escrita. A cabeça lê o símbolo de uma célula de cada vez e grava um novo símbolo no lugar. Após a leitura/gravação, ela move uma célula para a direita ou para a esquerda. O símbolo gravado na célula e o sentido do movimento da Cabeça de Leitura/Escrita são definidos pela função de transição, ou programa, que fará a transição de acordo com o estado corrente da máquina e o símbolo lido da fita.

2.1. Modelo formal

Uma máquina de Turing *M* é uma 8-upla:

$$M = (\Sigma, Q, \delta, q_0, F, V, \beta, \circledast)$$

onde:

Σ Alfabeto de símbolos de entrada;

Q Conjunto de estados possíveis da máquina;

δ Função de transição ou programa, tal que:

$$\delta\colon\ Q \times (\Sigma \cup V \cup \{\beta,\ \circledast\}) \rightarrow Q \times (\Sigma \cup V \cup \{\beta,\ \circledast\}) \times \{E,\ D\}$$

q_0 Estado inicial da máquina $(q_0 \in Q)$;

F Conjunto dos estados finais $(F \subset Q)$;

V Alfabeto auxiliar;

β Símbolo especial branco;

⊛ Símbolo especial marcador de início da fita.

Na literatura, diversos autores representam modelos formais utilizando o conceito de *Alfabeto da Fita*, simbolizado por Γ, que representa o conjunto de todos os símbolos possíveis que podem ser gravados na fita. Traçando um paralelo com o modelo aqui adotado, baseado no livro **Linguagens Formais e Autômatos – 4ª edição**, do autor Paulo Blauth de Menezes, ele representaria a união do conjunto do alfabeto de símbolos de entrada, o alfabeto auxiliar e os símbolos especiais de branco e de início da fita.

$$\Gamma = \Sigma \cup V \cup \{\beta, \circledast\}$$

Como é comumente encontrada e tem um formalismo bem definido, em alguns pontos do texto será utilizada esta notação, para simplificação das equações. Por exemplo, a representação da função de transição δ, acima denotada por:

$$\delta\colon\ Q \times (\Sigma \cup V \cup \{\beta,\ \circledast\}) \rightarrow Q \times (\Sigma \cup V \cup \{\beta,\ \circledast\}) \times \{E,\ D\}$$

ficaria:

$$\delta\colon\ Q \times \Gamma \rightarrow Q \times \Gamma \times \{E,\ D\}$$

A adoção de tal formalismo é apenas um artifício utilizado para a simplificação das equações. Ele não aparece no livro utilizado como referência no projeto, no qual este artigo e o simulador desenvolvido foram baseados. Mas como é facilmente deduzível a partir do modelo adotado, será utilizado ao longo do restante do texto.

O símbolo de início da fita aparece apenas uma vez, na célula mais à esquerda da fita. Inicialmente a cadeia de símbolos de entrada – ou palavra, representada por *w*, ocupa as casas mais à esquerda, logo adiante do símbolo de início da fita. A palavra de entrada pode conter apenas os símbolos do alfabeto de entrada e o símbolo de branco, não podendo, entretanto, terminar com branco. O alfabeto auxiliar eventualmente pode ser vazio.

A função de transição, ou função programa considera:

- ✔ Estado corrente;
- ✔ Símbolo lido da fita.

Para determinar

- ✔ Novo estado;
- ✔ Símbolo a ser gravado;
- ✔ Sentido de movimento da Cabeça de Leitura/Escrita, sendo *E* movimento para a esquerda e *D* movimento para a direita.

Dessa forma, tem-se que:

δ(estado corrente, símbolo lido) = (novo estado, símbolo gravado, sentido do movimento)

Por exemplo, suponha que $p, q \in Q$, $a_u, a_v \in \Gamma$ e $m \in \{E, D\}$, a função programa é definida como:

$$\delta(p,\ a_u) = (q,\ a_v,\ m)$$

Esta função pode ser interpretada como um grafo finito direto, como ilustrado no diagrama da **Fig. 3**.

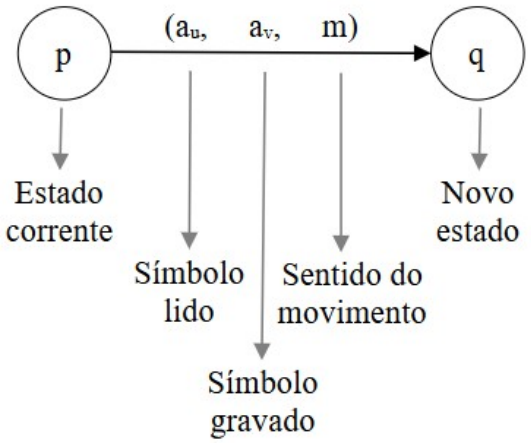

Fig 3. *Representação da função programa como um grafo.*

A função programa pode ser representada também na forma de uma tabela, como ilustrado na **Fig. 4**.

δ	**⊛**	**…**	**a_u**	**…**	**a_v**	**…**	**β**
p			(q, a_v, m)				
q							

Fig. 4: *Representação da função programa como uma tabela para o caso* $\delta(p,\ a_u) = (q,\ a_v,\ m)$.

O processamento de uma Máquina de Turing *M* para uma palavra de entrada *w* consiste na sucessiva aplicação da função programa a partir do estado inicial q_0, e da Cabeça de Leitura/Escrita posicionada na célula mais à esquerda da fita, até ocorrer uma condição de parada. A parada pode ser de duas maneiras: aceitando ou rejeitando a palavra de entrada.

As condições de parada são as seguintes:

- ✔ **Estado Final:** A máquina assume um estado final. A máquina para e a palavra de entrada é aceita;

- ✔ **Função indefinida:** A função programa é indefinida para o argumento (símbolo lido e estado corrente). A máquina para e a palavra de entrada é rejeitada;

- ✔ **Movimento inválido:** O argumento corrente da função define um movimento à esquerda e a Cabeça de Leitura/Escrita já se encontra na célula mais à esquerda. A máquina para e a palavra é rejeitada.

O processamento de M para a palavra de entrada w pode entrar em loop infinito e nunca parar, assim como pode acontecer com qualquer programa escrito em alguma linguagem de programação.

As notações adotadas para definir formalmente o comportamento de uma Máquina de Turing são as seguintes:

- ✔ **ACEITA (M) ou L(M):** Conjunto de todas as palavras de Σ* aceitas por M;

- ✔ **REJEITA (M):** Conjunto de todas as palavras de Σ* rejeitadas por M;

- ✔ **LOOP (M):** Conjunto de todas as palavras de Σ* para as quais M fica em loop infinito.

De tal forma que:

- ✔ $ACEITA(M) \cup REJEITA(M) \cap LOOP(M) = \emptyset$

- ✔ $ACEITA(M) \cup REJEITA(M) \cup LOOP(M) = \Sigma^*$

- ✔ O complemento de:

$$ACEITA(M) \text{ é } REJEITA(M) \cup LOOP(M)$$
$$REJEITA(M) \text{ é } ACEITA(M) \cup LOOP(M)$$
$$LOOP(M) \text{ é } ACEITA(M) \cup REJEITA(M)$$

Diversas variações do modelo adotado nesta seção podem ser encontradas. As mais significativas estão nas características da fita e no movimento da Cabeça de Leitura/Escrita. Alguns exemplos são:

- ✔ **Inexistência do Marcador de Início da Fita:** A fita pode não ter um marcador de início da fita. A célula mais à esquerda da fita contém o primeiro símbolo da entrada (ou branco, se a cadeia for vazia);

- ✔ **Cabeça de Leitura/Escrita não se move em uma leitura/gravação:** Na função programa é possível especificar, adicionalmente ao movimento para a esquerda ou direita, que a Cabeça de Leitura/Escrita permaneça parada na mesma célula de leitura/gravação.

- ✔ **Estado Final de Rejeição:** Pode ser definido um estado final de rejeição para explicitar a condição de parada com rejeição da entrada. Tal modificação se torna interessante, pois facilita a compreensão da lógica da função programa.

Tais variações não alteram o poder computacional do formalismo, ou seja, todas são equivalentes no que diz respeito à capacidade de processamento de símbolos.

3. Modelos Equivalentes à Máquina de Turing

O modelo de Máquina de Turing descrito na seção anterior é o modelo mais básico, contendo uma única fita e uma única Cabeça de Leitura/Escrita que percorre esta fita a partir da célula mais à esquerda. Apesar da aparente simplicidade do dispositivo, ele pode executar qualquer algoritmo escrito para ser executado por um computador real, embora de maneira lenta e desajeitada.

Existem diversos modelos equivalentes à Máquina de Turing. As variações mais importantes são:

a) **Autômato com Múltiplas Pilhas.** Um Autômato com Duas Pilhas (A2P) tem poder computacional equivalente ao da Máquina de Turing;

b) **Máquina de Turing Não-Determinística.** Numa Máquina de Turing Não-Determinística a função de transição poderá gerar subconjuntos, ou seja, várias possibilidades a partir de um estado e um símbolo lido;

c) **Máquina de Turing com Fita Infinita à Direita e à Esquerda.** No modelo com fita infinita à esquerda e à direita não há o símbolo de início da fita;

d) **Máquina de Turing com Múltiplas Fitas.** A Máquina de Turing com múltiplas fitas possui k fitas infinitas à esquerda e à direita. Neste modelo, a palavra de entrada é armazenada na primeira fita, ficando as demais com valor branco;

e) **Máquina de Turing multidimensional.** Neste modelo a fita é substituída por uma estrutura do tipo arranjo k-dimensional, infinita em todas as 2k dimensões.

f) **Máquina de Turing com Múltiplas Cabeças.** Neste modelo

No modelo desenvolvido, foi implementado um modelo de Máquina de Turing de Múltiplias Fitas.

A Máquina de Turing está no cerne do conceito moderno de algoritmo de computador. Apesar da simplicidade aparente do dispositivo, que descreverei nas próximas seções, há uma máquina de Turing para simular qualquer computador que possamos conceber, desde um microcontrolador simples baseado em silício até um computador quântico. Ao longo deste artigo será descrito cada detalhe do software desenvolvido para o simulador. Durante a elaboração do projeto, logo se percebeu a necessidade de criar uma linguagem de programação que instruísse a montagem da máquina, permitindo ao utilizador salvar seus programas e recarregá-los posteriormente, bem como digitar o código do programa no editor incorporado. Desta forma, o software, funciona como uma IDE (Integrated Development Environment – Ambiente de Desenvolvimento Integrado) básica.

Referências

Diverio, Tiarajú Asmuz; Menezes, Paulo Blauth. **Teoria da Computação: Máquinas Universais e Computabilidade**. 2. ed. Porto Alegre: Editora Sagra Luzzatto, 2004. cap. 3, p. 83-134. ISBN: 9798524105936.

Menezes, Paulo Blauth. **Linguagens Formais e Autômatos**. 4. ed. Porto Alegre: Editora Sagra Luzzatto, 2000. cap. 4, p. 133-152. ISBN: 85-241-0554-2.

Lewis, Harry R.; Papadimitriou, Christos H. **Elementos de Teoria da Computação**. 2. ed. Porto Alegre: Bookman, 2000. p. 176-265. ISBN: 85-7307-534-1.

https://www.historyofinformation.com/detail.php?id=619 Acesso em 06/06/2024, 10:36.

https://pt.wikipedia.org/wiki/Entscheidungsproblem. Acesso em 06/06/2024, 11:04

https://medium.com/@jan-kulawa/hilbert-g%C3%B6del-church-e-turing-fb019383c8f. Acesso em 06/06/2024, 11:52

https://www.britannica.com/biography/Alan-Turing#ref710077. Acesso em 06/06/2024, 12:00.

https://webpages.ciencias.ulisboa.pt/~fjferreira/Entscheidungsproblem.pdf. Acesso em 06/06/2024, 12:04.

http://profs.ic.uff.br/~isabel/pdf/Computabilidade,%20Hist%C3%B3ria%20e%20Matem%C3%A1tica.pdf. Acesso em 06/06/2024, 03:58

https://commons.wikimedia.org/wiki/File:Astronomer_Edward_Charles_Pickering%27s_Harvard_computers.jpg. 07/06/2024, 05:01

https://edisciplinas.usp.br/mod/publication/view.php?id=4925373&download=128947896. 07/06/2024, 05:39.

https://www.cs.virginia.edu/~robins/Turing_Paper_1936.pdf. 07/06/2024, 05:52.

https://en.wikipedia.org/wiki/Turing_machine_equivalents. 08/06/2024, 07:06

https://en.wikipedia.org/wiki/Entscheidungsproblem. 08/06/2024, 07:32.

https://pt.wikipedia.org/wiki/Computadores_de_Harvard. 08/06/2024, 09:02.

https://www.manhattanrarebooks.com/pages/books/1324/alan-turing/on-computable-numbers-with-an-application-to-the-entscheidungsproblem-with-on-computable?soldItem=true. 10/06/2024, 09:13.

https://www.gcsu.edu/sites/files/page-assets/node-808/attachments/brodkorb.pdf. 11/06/2023, 07:06.

https://www.cs.virginia.edu/~robins/Turing_Paper_1936.pdf. 11/06/2024, 07:12.