Computabilidade, Decibilidade e Máquinas de Turing

ANA CLÁUDIA GOMES SOUZA

aninha.g.souza@hotmail.com

Curso de Ciência da Computação - UEZO – Rio de Janeiro - RJ

RESUMO

Em 1936 Alan Turing desenvolveu uma das maiores invenções da computação, a máquina de Turing que permitiu que outros ciêntistas do ramo da matemática pudessem dar contitunidade a tais estudos como o matemático Alonzo Church que com sua tese nos permite entender o conceito de decibilidade. Com a máquina de Turing foi possível perceber problemas que antes não tínhamos conhecimento de sua existência como, por exemplo, o problema da parada, problema do azulejo (tiling problem) entre outros problemas. Além disso, podemos agora definir o que é computável e oque não pode ser computável e também podemos afirmar que para todo algoritmo existe uma máquina de Turing equivalente.

Palavras-chave: maquina de Turing, decibilidade, problema da parada, problema do azulejo (tilling problem), computável, não computável

1 INTRODUÇÃO

A busca por uma aplicação ou software que resolva problemas com eficiência e agilidade é algo que estamos sempre buscando aprimorar e tentando solucionar o maior número possível de problemas por meio do computador. Porém, existem problemas pelos quais não conseguimos resolver através do uso do computador e para isso temos uma algumas demonstrações matemáticas que podem nos explicar porque não conseguimos solucionar determinado problema com o auxílio do computador.

2 COMPUTABILDADE

O termo "computável" foi proposto por Alan Turing para representar os números reais que possuem uma expansão decimal que permite ser calculado em tempo finito, através de recursos finitos. Alan Turing argumenta que este conjunto corresponde ao conjunto de funções ou predicados computáveis. A proposta de Turing tem como base a identificação dos possíveis processos envolvidos no ato de "computar um número". Desta forma, Turing definiu um artefato teorico, que chamou de "máquina de computar", de maneira que, todo número que possui expansão decimal pudesse ser obtido a partir de operações da máquina seria chamado de "número computado por máquina".

Ao identificar os métodos necessários para computar um número, Turing estaria indiretamente identificando o conjunto de números (funções, predicados, etc) computáveis e, consequentemente, respondendo a seguinte pergunta: Quais tarefas podemos executar na máquina de computar?

Através da computabilidade podemos chegar a dois importantes resultados negativos em relação ao computador. O primeiro resultado negativo é baseado na teoria de Gödel que demonstra que nem tudo é possível resolver por meio do computador. O segundo resultado negativo foi abordado por Turing que por meio de uma demonstração que indica a impossibilidade de caracterizar a classe de problemas computáveis. Podemos perceber que existem dois tipos de problemas: Aqueles que possuem um procedimento que sempre termina (obtémos um resultado) chamamos de algoritmos e aqueles que possuem um procedimento que não é possível comprovar que sempre termina chamamos de procedimentos. Como exemplo de computabilidade temos as funções de Turing-Computáveis. Para exemplificarmos tais funções vamos criar um máquina de Turing que calcule a função soma = x + y

Soma = x + y

onde:

x =valor inteiro

y = valor inteiro

Soma = variavel que possui o valor da soma

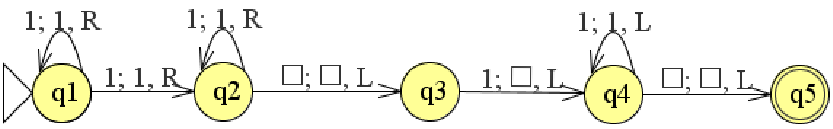

Fig.1: Máquina de Turing referente a função soma = x + y

Observação:

Esta máquina de Turing é um dos exemplos apresentados em Jflap

Podemos definir um termo não-computável como sendo os problemas que não podem ser solucionados por meio do computador. Seguem um exemplo de um problema não-computável:

Problema do azulejo (tiling problem)

Um azulejo é um quadrado com orientação fixa no plano de interesse. O desafio é descobrir uma área quadrada finita com azulejos respeitando a restrição de que os azulejos devem se encostar no lado de mesma cor. A entrada para este problema é um conjunto T composto por azulejos.

Seja T o conjunto dos azulejos:

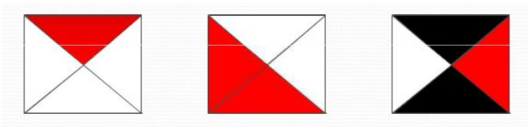

Fig.2: Conjunto de azulejos

Desejamos preencher uma área de 3X3 azulejos, porém observamos por meio da figura acima que isto não é possível.

Seja T o conjunto dos azulejos:

Fig.3: Conjunto de azulejos

Desejamos preencher uma área 5X5. Obtemos o resultado da figura abaixo.

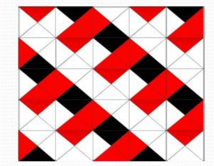

Fig.4: Area preenchida do conjunto de azulejos da fig.3

Podemos chegar a conclusão que não existe um algoritmo que resolva de forma eficiente o preenchimento de qualquer área do azulejo, desta forma podemos definir que este é um problema não-computável.

3 DECIDIBILIDADE

Um problema é considerado decidível se sua solução é encontrada num tempo finito, ou seja, existe uma máquina de Turing que retorna uma resposta. Caso contrário ele é considerado indecidível. O conceito de decidibilidade não trata a quantidade de tempo gasto e sim se ele é finito (o conceito que trabalha com a quantidade de tempo gasto é o da complexidade). Desta forma, podemos ver a diferença entre decidível e não decidível com realação a linguagem. Se o problema pode ser representado pr uma linguagem recursiva (onde a máquina de Turing sempre para) então o problema é decidível. Para exemplificarmos o conceito de decidibilidade primeiro é necessário conhecermos a Tese de Church-Turing.

Tese de Church-Turing

A tese de Church-Turing diz que qualquer programa de computador pode ser traduzido em uma máquina de Turing, e qualquer máquina de Turing pode ser traduzida para uma linguagem de programação de propósito geral. Desta forma, a tese é equivalente a dizer que qualquer linguagem de programação de propósito geral é suficiente para expressar qualquer algoritmo. Como exemplo da tese de Church-Turing temos a demonstração da soma de dois numeros inteiros por meio do algoritmo e de sua respectiva máquina de Turing.

Soma = x + y

onde:

x =valor inteiro

y = valor inteiro

Soma = variavel que possui o valor da soma

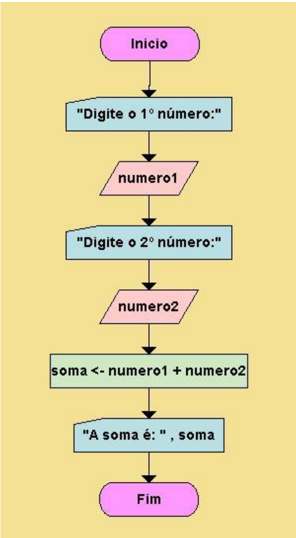

Fig.6: Fluxograma do algoritmo soma

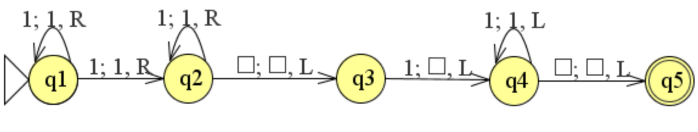

Fig.7: Maquina de Turing referente a soma de dois valores

Observações:

Esta máquina de Turing é um dos exemplos apresentados em Jflap

A máquina de Turing soma segue a seguinte linguagem: x = |z(x)| com z(x) € {1}*, ou seja, o número será representado pela quantidade de 1, por exemplo: 5 + 3 equivale a w = 111110111

Um exemplo de um problema indecidível é o problema da parada que pode ser delarado de maneira informal da seguinte forma:

Dado um programa e uma entrada finita, decida se o programa termina de rodar ou se rodará indefinidamente.

Alan Turing provou em 1936 que um algoritmo genérico para resolver o problema da parada para todos os pares de programa-entrada possíveis não pode exisitir. Desta forma, dizemos que o problema da parada é indecidível nas máquinas de Turing.

4 MÁQUINA DE TURING

Em 1936, Alan Turing introduziu um modelo matemático do processo de computação conhecido como máquina de Turing. Sua estrutura é composta basicamente pelas seguintes partes: Uma fita que é responsável pela entrada, saída e memória sendo divida em células que armazenam um símbolo de cada vez. A unidade de controle que indica o estado de controle da máquina. Possui uma unidade de leitura e gravação que pode se deslocar da esquerda (L) e da direita (R) da fita, podendo ler / gravar um único símbolo em cada movimento. A função de transição é resposável pela leitura e gravação, o sentido de movimento da cabeça define o estado da máquina. Podemos definir uma máquina de Turing como um sétupla M = (Q, ∑ ,Γ, δ, q0, F) onde:

Q – Conjunto de estados internos

∑ - Conjunto do alfabeto de entrada

Γ – Conjunto finito de símbolos, chamado de alfabeto da fita

δ – Função de transição, definida por S: Q x Γ ⟶ Q x Γ x {L,R}

qo – Estado inicial (q0 ϵ Q)

F – Conjunto de estados finais (F ϵ Q)

Observação:

O símbolo □ é um símbolo especial chamado de branco (□ ϵ Γ)

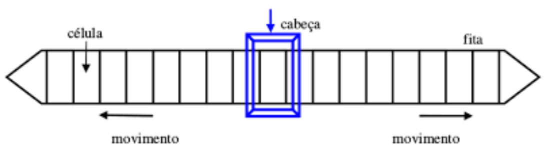

Fig. 8: Demonstração do funcionamento da máquina de Turing

Ainda em 1936, Alonzo Church apresentou sua tese que afirma que qualquer função computável pode ser processada por uma máquina de Turing. Existem vários modelos de máquinas de Turing. Asseguir veremos alguns desses modelos, mas antes faremos um panorama das linguagens e seus respectivos formalismos.

Tab. 1: Tabela de Linguagens e seus Formalismos

Linguagens	**Formalismos**
Tipo 03 – Regulares	Operacional ou reconhecedor – Automato finito (deterministico, não-deterministico e mínimo) Axiomático ou gerador – Gramática regular Denatacional – Expressão regular
Tipo 02 - Livres de Contexto	Automato com pilha (deterministico e não-deterministico) Axiomatico ou gerador – Gramática livre de contexto
Tipo 01 – Sensíveis de Contexto	Operacional ou reconhecedor - Máquina de Turing com fita limitada Axiomatico ou gerador – Gramática sensível ao contexto
Tipo 0 – Enumeráveis Recursivamente	Operacional ou reconhecedot – Máquina de Turing Axiomático ou gerador – Gramática irrestrita

Máquinas de Turing como Reconhecedores

Dada uma linguagem que seja aceita ou reconhecida por uma máquina de Turing é dada pela definição abaixo:

Seja uma máquina de Turing M = (Q, ∑, Γ, δ, q0, F), então a linguagem reconhecida por M é: L (M) = { W ϵ $\sum^{+}$: qoW ⊦ * x1qfx2 para algum qf ϵ F e x1, x2 ϵ Γ*}

Exemplo: Para M = {0,1}, a máquina de Turing que aceita a expressão regular ER = 0* pode ser definida como:

$M = (\{q0, q1\}\ \{0\}, \{0, \square\}, \delta, q0, \{q1,\})$ com

$\delta\ (q0,0) = (q0,0,R)$ e

$\delta\ (q0,\square) = (q1, \square, R)$

Máquinas de Turing como Transdutores

Utilizadas para resolver calculos de uma função é dada pela seguinte definição:

Uma funação f, como domínio D, é dita ser Turing-Computável ou simplemente computável se existe alguma máquinade Turing $M = (Q, \sum, \Gamma, \delta, q0,)$ tal que $qoW \vdash * Mqf \int(W)$, $qf \in F$ para toda cadeia $W \in D$.

Exemplo: Seja $x = |z(x)|$ com $z(x) \in \{1\}*$ a máquina de Turing deve calcular: $q0\ W = q0\ z(x)\ 0\ z(y) \vdash *\ qf\ z\ (x + y)\ 0$

Seja $M = (\{q0, q1, q2, q3, q4\}, \{1\}, \{1, 0, \square\}, \delta, q0\ \{q4\})$

$\delta\ (q0, 1) = (q0, 1, R)$

$\delta\ (q0, 0) = (q1, 1, R)$

$\delta\ (q1, 1) = (q1, 1, R)$

$\delta\ (q1, \square) = (q2, \square, L)$

$\delta\ (q2, 1) = (q3, o, L)$

$\delta\ (q3, 1) = (q3, 1, L)$

$\delta\ (q3, \square) = (q4, \square, R)$

Máquina de Turing com Fita Semi-Finita

Uma máquina de Turing pode ter uma fita ilimitada a esquerda e / ou a direita. Se a fita é ilimita a esquerda e a diteira então temos uma máquina de Tuirng padrão. Caso haja um limite na fita a direita ou a esquerda ela é chamada de máquina de Turing com fita semi-finita. Dessa forma, em uma direção da fita os movimentos são restritos. Não pode mover para fora da fita.

Se nas duas direções há restrições do movimento então estaremos trabalhando com uma máquina de Turing com fita limitada. Essa também é chamda de Automato Limitado Linearmente ou Automato de Fita Limitada.

Máquina de Turing com Múltiplas Fitas

Essa máquina de Turing possui mais que uma fita e para cada uma dessas fitas existe uma cabeça de leitua / escrita.

$\delta : Q \times \Gamma^n \longrightarrow Q \times \Gamma^n \times \{L,R\}^n$

Onde n é a quantidade de fitas na máquina de Turing

Máquina de Turing com Multiplas Cabeças

Essa máquina de Turing possui uma única fita e K (K > 1) cabeças de leitura / gravação sobre a mesma fita. O processamento dependerá do estado corrente e do símbolo lido em cada uma das cabeças.

Máquina de Turing Múltidimensional

Nessa máquina de Turing a fita é substituída por uma estrutura m-dimencional, infita em todas as suas direções. Por exemplo, numa máquina de Turing bidimensional a cabeça de leitura / gravação da fita pode ir para esquerda, direita, acima e abaixo. A função de transição pode ser definida como:

$\delta : Q \times \Gamma \longrightarrow Q \times \Gamma \times \{L,R,U,D\}$

Onde U (up) é para cima e D (down) é para baixo

5 CONCLUSÕES

Podemos concluir que apesar da máquina de Turing ter 82 anos levando em consideração seu ano de criação 1936, seus conceitos ainda são muito importantes nos dias atuais para que possamos entender a definição de computável e não-computável, decidível e não decidível. Pecebemos que existem problemas pelos quias o computador consegue resolver, porém existem outros onde isso não é possível. E também podemos concluir que uma máquina de Turing pode representar um algoritmo e vice-versa.

REFERÊNCIAS BIBLIOGRÁFICAS

https://pt.wikipedia.org/wiki/Problema_da_parada

https://pt.wikipedia.org/wiki/Computabilidade

https://pt.wikipedia.org/wiki/Tese_de_Church-Turing

http://docs.fct.unesp.br/docentes/dmec/olivete/tc/arquivos/Aula6.pdf