O Problema da Parada

Danilo Souza, Hugo Santos, Iago Medeiros, Welton Araújo
Matrículas - 10080000801, 10080000701, 10080001901, 10080000501

3 de Julho de 2012

Abstract

During the 30s, many years before the invention of the computers, important definitions of the computer logic were being discussed by mathematicians of the world. Concepts like "algorithm", introduced by David Hilbert, were invented even without having a concrete application.
The most relevant advance in this period was the establishment of the Turing Machine. It was proved as the most loyal abstract representation of a computer. However, its biggest challenge is worldwide known as The Halting Problem. After a short preview about reducibility, computability and decidability, this paper will show this famous math puzzle.

Turing Machine, Halting Problem, reducibility, decidability

1 Resumo

Durante a década de 30, muitos anos antes da criação do computador, importantes definições da lógica computacional já eram discutidos por matemáticos do mundo todo. Conceitos como "algoritmo", introduzido por David Hilbert, foram inventados mesmo sem ter uma aplicação concreta.
O avanço mais relevante dessa época foi o estabelecimento da máquina de Turing. Ela foi apresentada como a representação abstrata mais fiel de um computador. Porém, seu maior desafio é conhecido mundialmente como O Problema da Parada (*The Halting Problem*). Após uma breve ideia sobre redutibilidade, computabilidade e decidibilidade, será apresentado este famoso quebra-cabeças da matemática.

2 Histórico

Muito antes dos computadores modernos terem sido inventados, a teoria da computação já havia sido iniciada e estava em desenvolvimento. Matemáticos de todo o mundo buscavam resolver problemas através de processos com um número finito de passos que pudessem sempre retornar resultados corretos para uma mesma questão.
O conceito de algoritmo sempre foi utilizado e aperfeiçoado de forma implícita pela comunidade matemática. Em 1900, durante o II Congresso Internacional de Matemática realizado em Paris, David Hilbert apresentou 23 questões não

resolvidas pertinentes à toda a sociedade matemática. Nesta lista o décimo problema dizia respeito aos algoritmos, ainda usados com conceitos abstratos, Hilbert requisitou explicitamente um algoritmo para a identificação de existência de raízes inteiras para um polinômio.
Mesmo com o problema não resolvido (ou provado que não pode ser resolvido), um grande avanço permitiu que ele fosse melhor analisado: a publicação da chamada Tese de Church-Turing, em 1936. O artigo definia precisamente a noção de algoritmo que até então não existia e estava inviabilizando a continuidade do estudo na área.
A Tese de Church-Turing afirma que, possuindo tempo e capacidade de espaço suficiente, qualquer função poderia ser computada naturalmente através de um algoritmo em um computador. Através disso é possível perceber que até o mais simples computador, possui teoricamente um poder equivalente a um outro qualquer.

3 Introdução

A definição concreta da palavra algoritmo foi um processo que levou vários anos e foi sendo desenvolvido gradativamente pela comunidade matemática com o passar dos tempo. Em 1900, o matemático alemão David Hilbert apresentou explicitamente o conceito de algoritmo e seu significado, como conhecemos hoje. Essa definição é muito importante para a computação. A ideia de definir um conjunto finito de passos que retornem sempre um resultado correto para a mesma situação é a base ca computação, pois com ela é possível projetar e implementar estruturas físicas capazes de resolver problemas comuns do dia-a-dia.

4 Decidibilidade

Para entendermos melhor o conceito de decidibilidade, é necessário explorarmos brevemente o conceito de computabilidade. Um programa é dito computável se for possível calcular seu valor independente do elemento de seu domínio. Logo, se o programa for um loop infinito, é dito não computável. Um programa pode ainda ser parcialmente computável, onde é possível calcular sua saída para um conjunto finito de elementos de seu domínio. Por exemplo, um programa que possui uma variável x em um laço incremental, a partir de um determinado valor de x o programa não para.
Decidibilidade é um caso particular da computabilidade quando o programa (ou função) só admite dois valores, usada para definir se um problema é solúvel ou não. A parcialidade também se aplica neste caso, onde um programa é parcialmente decidível se for decidível para um subconjunto contido no seu conjunto de argumentos admissíveis.

5 Redutibilidade

A técnica da redução é o principal método utilizado para provar que alguns problemas relacionados à máquinas de Turing são indecidíveis. A técnica utiliza

um algoritmo que converte instâncias de um problema X para instâncias de um problema Y.
A partir dessa conversão, assumindo que o problema Y seja decidível, o problema X também deve ser. Por conseguinte, se a uma dada situação X possuir resposta verdadeira, Y obrigatoriamente deve possuir a mesma resposta. Então a solução encontrada com Y também é uma solução para X. Este método é uma poderosa forma de provar que um problema é indecidível porque um problema de decisão não solucionado pode ser provado insolúvel através da conversão dele em um problema insolúvel. Ou seja, prova que este é indecidível.

6 O Problema da Parada

O problema da parada consiste em projetar um computador P (máquina de Turing) que seja capaz de analizar uma máquina de Turing M e uma entrada w, e dizer se a máquina M para quando alimentada por w ou se a máquina M não para quando alimentada por w.

$$P = \{(M, n) \mid M \textit{ é uma } MT \textit{ e } M \textit{ aceita } w\} \tag{1}$$

Supomos agora um procedimento H decisor para a máquina P, onde H sempre vai parar, independente de M aceitar ou não a palavra w. Se rodarmos H sobre o procedimento P, H irá aceitar se M aceitar w, ou H irá rejeitar P, se M rejeitar w.

Consideremos agora uma nova Máquina de Turing G que possui H como subrotina.

- Definimos a MT P
- Definimos o procedimento decisor H
- Rodar H sobre P
  - H para e aceita se M aceita w
  - H para e rejeita se M rejeita w
- Definimos a MT G (com H como subrotina)
- G chamará H quando a entrada de M for <M > (descrição de M)
  - G aceita se M não aceita <M >
  - G rejeita se M aceita <M >
- Rodar G sobre sua própria descrição (<G >)
  - G aceita se <G > não aceita
  - G rejeita se <G > aceita

  Com isso conclui-se que, independente do funcionamento da máquina G, ela fará sempre o contrário, resultando sempre em uma contradição. Tendo em vista essa contradição, pode-se afirmar que nem a máquina G e nem o procedimento H podem existir. A figura mostra um fluxograma para entermos melhor o funcionamento do algoritmo.

7 Conclusão

Teoria da computação embute no nome que sua criação surgiu a partir da utilização de computadores para resolver problemas matemáticos, entretanto ela mesma surgiu muito antes de os computadores terem uma posição de destaque nesse mundo decidindo guerras e ajudando no desenvolvimento de tecnologias. Máquina de Turing foi uma contribuição muito importante.
O problema da parada surgiu como uma tentativa de detectar se uma MT para devido a uma entrada qualquer. Várias técnicas foram utilizadas para tentar gerar uma prova se um algoritmo podia ou não ser gerado para qualquer MT como, por exemplo, técnica de redutibilidade e decidibilidade que provam, através de uma contradição, ser impossível tal algoritmo.
O problema da parada está longe de ser o único sem uma solução algoritmica como Entscheidungsproblem (problema de decisão em alemão) ou Post Correspondence Problem (PCP). No entanto ele mesmo serve para provar que outros problemas são indecidíveis através da redutibilidade.

8 Bibliografia