Problema da Parada

Juliana Kaizer Vizzotto

Universidade Federal de Santa Maria

Disciplina de Teoria da Computação

Roteiro

.

- Introdução
- Problema da Parada
- Método da Diagonalização
- Prova que o Problema da Parada é Indecidível

Introdução

- Quais problemas que são algoritmicamente insolúveis?
- Vamos ver que os computadores são limitados de uma maneira fundamental.
- Problema da Parada
- Outro problema insolúvel: problema geral da verificação de software! (Verificar que o programa funciona como especificado)
-

Problema da Parada

- Problema de se determinar se uma Máquina de Turing aceita uma dada cadeia de entrada:

$$A_{MT} = \{\langle M, w\rangle | M \text{eh uma } MT \text{ e } M \text{ aceita } w\}.$$

- **Teorema**: A_{MT} é indecidível.

- Significa que não conseguimos construir uma máquina de Turing que sempre pára aceitando ou rejeitando a entrada.

Problema da Parada

- Entretanto, A_{MT} é Turing-reconhecível.
- Reconhecedores são mais poderosos que decisores.
- Requerer que uma MT para sobre todas as entradas restringe os tipos de linguagens que ela pode reconhecer.

Problema da Parada

- A máquina de Turing U a seguir reconhece A_{MT}.
- $U =$ "Sobre a entrada $\langle M, w \rangle$, onde M é uma MT e w é uma cadeia:
  1. Simule M sobre a entrada w.
  2. Se M em algum momento entra no seu estado de aceitação, *aceite*; se M em algum momento entra em seu estado de rejeição, *rejeite*."
- Note que essa máquina entra em *loop* sobre a entrada $\langle M, w \rangle$ se M entra em *loop* sobre w, e é por isso que essa máquina não decide A_{MT}.

Problema da Parada

- Se o algoritmo tivesse alguma forma de determinar que M não iria parar sobre w, ele poderia dizer *rejeite*.
- Esse problema é denominado **problema da parada**.
- Vamos demonstrar que o algoritmo não tem como fazer essa determinação!!
- Essa máquina de Turing U é um exemplo de *máquina de Turing Universal*.
- Capaz de simular qualquer outra máquina de Turing a partir da descrição da mesma.
- A máquina de Turing universal desempenhou um papel importante no estímulo ao desenvolvimento de computadores de programa armazenado.

Introdução

- A prova da *indecidibilidade* do problema da parada usa uma técnica chamada *diagonalização*, descoberta pelo matemático George Cantor em 1873.
- Problema: medir conjuntos infinitos.
- Para dois conjuntos infinitos como podemos dizer se um é maior que o outro ou se eles têm o mesmo tamanho?
- Para conjuntos finitos: contamos os elementos!
- Contar elementos de conjuntos infinitos? Nunca terminamos!
- Portanto, não podemos usar o método de contagem para determinar os tamanhos relativos de conjuntos infinitos.

Introdução

- Por exemplo, tome o conjunto dos inteiros pares e o conjunto de todas as cadeias sobre $\{0, 1\}$.
- Ambos conjuntos são infinitos e, portanto, maiores que qualquer conjunto finito, mas um dos dois é maior que o outro? Como podemos comparar seu tamanho relativo?
- Cantor propôs uma solução elegante para esse problema.
- Ele observou que dois conjuntos finitos têm o mesmo tamanho se os elementos de um deles puderem ser emparelhados com os elementos do outro.
- Esse método compara os tamanhos sem recorrer a contagem.
- Cantor estendeu esse método para conjuntos infinitos.

Correspondência

- Suponha que tenhamos os conjuntos A e B e uma função f de A para B. Digamos que f é **um-para-um** se ela nunca mapeia dois elementos diferentes para o mesmo lugar - ou seja, se $f(a) \neq f(b)$ sempre que $a \neq b$. Digamos que f é **sobrejetora** se ela atinge todo o elemento de B - ou seja, se para todo $b \in B$ existe um $a \in A$ tal que $f(a) = b$. Digamos que A e B são de **mesmo tamanho** se existe uma função um-para-um e sobrejetora $f :\to B$. Uma função que é tanto um-para-um quanto sobrejetora é denominada **correspondência**.

Correspondência

- Em uma correspondência, todo o elemento de A mapeia para um único elemento de B e cada elemento de B tem um único elemento de A mapeando para ele.

- Uma correspondência é simplesmente uma maneira de emparelhar os elementos de A com os elementos de B.

- Exemplo: Seja $\mathbf{N} = \{1, 2, 3, ...\}$ e $\xi = \{2, 4, 6, ...\}$, tal que $f(n) = 2n$.

Correspondência

- **Definição**: Um conjunto é **contável** se é finito ou tem o mesmo tamanho que $\mathbf{N}$.
- **Exemplo**: Seja $\mathbf{Q} = \{\frac{m}{n} | m, n \in \mathbf{N}\}$ o conjunto de números racionais positivos.
- $\mathbf{Q}$ parece ser muito maior que $\mathbf{N}$.
- Ainda assim, esses dois conjuntos são do mesmo tamanho conforme a definição de correspondência.
- Vamos mostrar uma correspondência de $\mathbf{N}$ com $\mathbf{Q}$,i.e., vamos mostrar que $\mathbf{Q}$ é contável.
- Uma maneira fácil de fazer isso é listar todos os elementos de Q.
- Então, emparelhamos o primeiro elemento da lista com o número 1 de $\mathbf{N}$, o segundo da lista com o número 2 de $\mathbf{N}$ e assim por diante. Cuidado com as repetições!

Correspondência

- Depois de ver a correspondência de $\mathbf{N}$ com $\mathbf{Q}$ você pode achar que quaisquer dois conjuntos infinitos podem ser mostrados como tendo o mesmo tamanho.
- Afinal, precisamos somente exibir uma correspondência!
- Esse exemplo revela que realmente existem algumas correspondências surpreendentes!
- Entretanto, para alguns conjuntos infinitos, nenhuma correspondência com $\mathbf{N}$ existe!
- Esses conjuntos são simplesmente grandes demais.
- Tais conjuntos são chamados **incontáveis**.

Método da Diagonalização

- O conjunto dos números reais é um exemplo de conjunto incontável!
- Seja **R** o conjunto dos números reais, Cantor provou que **R** é incontável.
- Ao construir essa prova, Cantor introduziu o método da **diagonalização**.
- **Teorema**: **R** é incontável.
- **Prova**: Para provar que **R** é incontável, mostramos que não existe correspondência entre **N** e **R**.
- A prova é por contradição.

Método da Diagonalização

- **Prova**: Suponha que existe uma correspondência f entre **N** e **R**. Nossa tarefa é mostrar que f não funciona como deveria.
- Para ela ser uma correspondência temos que emparelhar todos os membros de **N** com todos os membros de **R**.
- Mas encontraremos um x em **R** que não é emparelhado com nada em **N**. Isto será a contradição.
- A maneira pela qual encontramos esse x é verdadeiramente construindo-o.
- Escolhemos dada dígito de x para tornar x diferente de um dos números reais que está emparelhado com um dos elementos de **N**.
- No final asseguramos que x seja diferente de qualquer número real que esteja emparelhado.

Método da Diagonalização

- **Corolário**: Algumas linguagens não são Turing-reconhecíveis.
- **Prova**: Mostrar que o conjunto de todas as MTs é contável e que o conjunto das linguagens é incontável.
- Para mostrar que o conjunto de todas as MTs é contável, primeiro observamos que o conjunto de todoas as cadeias Σ^* é contável para qualquer alfabeto Σ.
- Com apenas uma quantidade finita de cadeias de cada comprimento, podemos formar uma lista de Σ^* listando todas as cadeias de comprimento 0, comprimento 1, comprimento 2, e assim por diante...
- O conjunto de todas as MTs é contável pq cada MT M tem uma codificação em uma cadeia $\langle M \rangle$.
- Se simplesmente omitirmos aquelas cadeias que não são codificações legítimas de MTs, podemos obter uma lista de todas as máquinas de Turing.

Método da Diagonalização

- **Corolário**: Algumas linguagens não são Turing-reconhecíveis.
- **Prova**: Para mostrar que o conjunto de todas as linguagens é incontável, primeiro observamos que o conjunto de todas as sequências binárias infinitas é incontável.
- Uma sequência binária infinita é uma sequência interminável de 0s e 1s.
- Seja **B** o conjunto de todas as sequências binárias infinitas.
- Podemos mostrar que **B** é incontável usando uma prova por diagonalização. Mostre!

Método da Diagonalização

- **Corolário**: Algumas linguagens não são Turing-reconhecíveis.
- **Prova**:Primeiro mostramos que o conjunto de todas as Máquinas de Turing é contável.
- Observamos que o conjunto de todas as cadeias Σ^* é contável para qualquer alfabeto Σ.
- Com apenas uma quantidade finita de cadeias de comprimento, podemos formar uma lista de Σ^* listando todas as cadeias de comprimento 0, comprimento 1, comprimento 2, e assim por diante.
- **O conjunto de todas as máquinas de Turing é contável porque cada máquina de Turing M tem uma codificação em uma cadeia $\langle M \rangle$.**
- Se simplesmente omitirmos aquelas cadeias que não são codificações legítimas de MTs, podemos obter uma lista de todas as Mts.

Método da Diagonalização

- **Corolário**: Algumas linguagens não são Turing-reconhecíveis.
- **Prova**: Para mostrar que o conjunto de todas as linguagens é incontável, primeiro observamos que o conjunto de todas as sequências binárias infinitas é incontável.
- *Uma sequência binária infinita* é uma sequência interminável de 0s e 1s.
- Seja **B** o conjunto de todas as sequências binárias infinitas.
- Podemos mostrar que **B** é incontável usando uma prova por *diagonalização*.

Método da Diagonalização

- **Corolário**: Algumas linguagens não são Turing-reconhecíveis.
- **Prova**: seja **L** o conjunto de todas as linguagens sobre o alfabeto Σ.
- Mostramos que **L** é incontável dando uma correspondência com **B**, consequentemente mostrando que os dois conjuntos são do mesmo tamanho.
- Seja $\Sigma^* = \{s_1, s_2, s_3, \ldots\}$. Cada $A \in \mathbf{L}$ tem uma sequência única em **B**.
- O i-ésimo bit dessa sequência é 1 se $s_i \in A$ e é 0 se $s_i \notin A$, o que é chamado de **sequência característica** de A.

Método da Diagonalização

- Por exemplo, se A fosse a linguagem de todas as cadeias começando com 0 sobre o alfabeto $\{0, 1\}$, sua sequência característica ξ_A seria:

$$
\begin{array}{lcl}
\Sigma^* & = & \{\sigma, 0, 1, 00, 01, 10, 11, 000, 001, ...\} \\
A & = & \{0, 00, 01, 000, 001, ..\} \\
\xi_a & = & 010110011...
\end{array}
$$

- A função $f : \mathbf{L} \to \mathbf{B}$, onde $f(A)$ é igual a sequência característica de A, é um-para-um e sobrejetora e, portanto, uma correspondência.
- Consequentemente como $\mathbf{B}$ é incontável, $\mathbf{L}$ também é incontável.

Método da Diagonalização

- Assim, mostramos que o conjunto de todas as linguagens não pode ser posto em correspondência com o conjunto de todas as máquinas de Turing.

- Concluímos que algumas linguagens não são reconhecidas por nenhuma máquina de Turing.

O problema da Parada é Indecidível

- Considere a linguagem:

$$A_{MT} = \{\langle M, w\rangle | M \text{ eh uma MT e } M \text{ aceita } w\}$$

- **Prova**: Supomos que A_{MT} seja decidível e obtemos uma contradição.
- Suponha que H seja um decisor para $A_M T$. Sobre a entrada $\langle M, w\rangle$, na qual M é uma MT, e w é uma cadeia, H pára e aceita se M aceita w. Além disso, H pára e rejeita se M falha em aceitar w.
- Assim, H é uma MT tal que:

$$H(\langle M, w\rangle) = \begin{cases} \text{aceite se M aceita w} \\ \text{rejeite se M não aceita w} \end{cases}$$

O problema da Parada é Indecidível

- Agora construímos uma nova MT D com H como subrotima.
- Essa nova MT chama H para determinar o que M faz quando a entrada para M é sua própria descrição $\langle M \rangle$.
- D faz o oposto então, ou seja, ela rejeita se M aceita e aceita se M não aceita.
- $D =$ "Sobre a entrada $\langle M \rangle$, onde M é uma MT:
  1. Rode H sobre a entrada $\langle M, \langle M \rangle \rangle$.
  2. Dê como saída o oposto de que H dá como saída; ou seja, se H aceita, *rejeite* e se H rejeita, *aceite*.

O problema da Parada é Indecidível

- Não se atrapalhe com a ideia de rodar uma máquina sobre a própria descrição!
- Trata-se se rodar um programa consigo próprio como entrada, algo que de fato ocorre ocasionalmente na prática, por exemplo, um compilador. Um compilador para Haskell, pode ele próprio se escrito em Haskell. Portanto, rodar esse programa sobre si próprio faria sentido.
- Assim:

$$D(\langle M \rangle) = \begin{cases} \text{aceite se M não aceita } \langle M \rangle \\ \text{rejeite se M aceita } \langle M \rangle \end{cases}$$

O problema da Parada é Indecidível

- O que acontece quando rodamos D com sua própria descrição?

$$D(\langle D \rangle) = \begin{cases} \text{aceite se D não aceita } \langle D \rangle \\ \text{rejeite se D aceita } \langle D \rangle \end{cases}$$

- Uma contradição!!!

O problema da Parada é Indecidível

- Suponha que uma MT H decida A_{MT}.
- Então use H para construir uma MT D que, quando recebe uma dada entrada $\langle M \rangle$, aceita exatamente quando M não aceita a entrada $\langle M \rangle$.
- Finalmente rode D sobre si própria.
- As máquinas tomam as seguintes ações, com a última linha sendo a contradição:
  - H aceita $\langle M, w \rangle$ exatamente quando M aceita w.
  - D rejeita $\langle M \rangle$ exatamente quando M aceita $\langle M \rangle$.
  - D rejeita $\langle D \rangle$ exatamente quando D aceita $\langle D \rangle$.

O problema da Parada é Indecidível

- Vamos entender onde entra o método da diagonalização!

- Vamos examinar as tabelas de comportamento para as MTs H e D.

Consequências da Insolubilidade do Problema da Parada

- Não temos como fazer um programa que teste se “os nossos programas entram em loop”
- Ora, se esse algoritmo existisse seria possível implementá-lo usando uma MT. Vimos que tal MT não existe!
- Podemos usar o Problema da Parada para provar a insolubilidade de outros problemas.
- **O Problema da Parada é Insolúvel**: não existe uma solução algoritmica.

Exercícios

- Seja $T = \{(i, j, k) | i, j, k \in \mathbf{N}\}$. Mostre que T é contável.