Complexidade de Tempo

Esdras Lins Bispo Jr.
bispojr@ufg.br

Teoria da Computação
Bacharelado em Ciência da Computação

27 de junho de 2016

UFG
Regional Jataí

Plano de Aula

1. Pensamento
2. Revisão
   - O Método da Diagonalização
3. Complexidade de Tempo

UFG
Regional Jataí

Sumário

Pensamento

Pensamento

Frase

Para todo problema complexo existe sempre uma solução simples, elegante e completamente errada.

Quem?

Henry Mencken (1880-1956)
Jornalista e crítico social americano.

Sumário

Conjuntos Incontáveis

Teorema 4.17

$\mathbb{R}$ é incontável.

Ideia da Prova

- De forma a mostrar que $\mathbb{R}$ é incontável, mostramos que nenhuma correspondência existe entre $\mathbb{N}$ e $\mathbb{R}$.
  - Supomos, a princípio, que a correspondência f existe.
  - Logo após, apresentamos um valor $x \in \mathbb{R}$ que não está emparelhado com valor algum em $\mathbb{N}$
    (o que indica um absurdo).

Conjuntos Incontáveis

n	$f(n)$
1	$3,14159\ldots$
2	$55,55555\ldots$
3	$0,12345\ldots$
4	$0,50000\ldots$
$\vdots$	$\vdots$

Figura: Suposta correspondência f entre $\mathbb{N}$ e $\mathbb{R}$.

Conjuntos Incontáveis

n	$f(n)$
1	$3{,}\underline{1}4159\ldots$
2	$55{,}5\underline{5}555\ldots$
3	$0{,}12\underline{3}45\ldots$
4	$0{,}500\underline{0}0\ldots$
$\vdots$	$\vdots$

$$\boxed{x = 0{,}4641\ldots}$$

Figura: Construção de x a partir da correspondência f.

UFG
Regional Jataí

Conjuntos Incontáveis

Considerações

Apenas deve-se ter o cuidado de escolher dígitos para x diferentes de 0 e 9, devido ao fato de

$$3,999\ldots = 4,000\ldots$$

Conjuntos Incontáveis

Corolário do Teorema 4.17

Algumas linguagens não são Turing-reconhecíveis.

Ideia da Prova

1. Observar que o conjunto de todas as máquinas de Turing é contável;
2. Observar que o conjunto de todas as linguagens é incontável.
3. Como há mais linguagens do que máquinas de Turing, então algumas linguagens não podem ser Turing-reconhecíveis.

UFG
Regional Jataí

Conjuntos Incontáveis

O conjunto de todas as máquinas de Turing é contável

- Σ^* é contável;
- Cada máquina de Turing pode ser codificada em uma cadeia $\langle M \rangle$;
- O conjunto C de todas as máquinas de Turing pode ser representado por um conjunto de cadeias $\langle M \rangle$;
- É possível enumerar C;
- Logo C é contável.

Conjuntos Incontáveis

O conjunto de todas as linguagens é incontável

- O conjunto B de todas as sequências binárias infinitas é incontável;
- Qualquer linguagem pode ser descrita como uma sequência característica;
- O conjunto L de todas as linguagens podem ser representado por um conjunto de sequências característica;
- A função $f : L \rightarrow B$
  (em que $f(A)$ é igual à sequência característica de A)
  é uma correspondência;
- Logo, como B é incontável, L é incontável.

Conjuntos Incontáveis

$$
\begin{array}{rcccccccccccl}
\Sigma^* = \{ & \varepsilon\ , & 0\ , & 1\ , & 00\ , & 01\ , & 10\ , & 11\ , & 000\ , & 001\ , & \cdots & \} & ; \\
A = \{ & & 0\ , & & 00\ , & 01\ , & & & 000\ , & 001\ , & \cdots & \} & ; \\
\chi_A = & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & \cdots & & .
\end{array}
$$

Figura: Construção de $\mathcal{X}A$ a partir da correspondência Σ^*.

Conjuntos Incontáveis

Exercício

Mostrar que o problema da parada é indecidível.

Sumário

UFG
Regional Jataí

Complexidade

Por que estudar complexidade?

Um problema pode ser até decidível, mas pode levar uma quantidade de tempo ou memória bastante elevada.

Complexidade

Por que estudar complexidade?

Um problema pode ser até decidível, mas pode levar uma quantidade de tempo ou memória bastante elevada.

Questões do estudo de complexidade

- Quanto tempo[espaço] leva[ocupa] um determinado algoritmo?
- O que faz um algoritmo gastar[ocupar] mais tempo[espaço] do que um outro?
- É possível classificar os algoritmos em termos de complexidade?

UFG
Regional Jataí

Complexidade de Tempo

Problema

Seja a linguagem $A = \{0^k 1^k \mid k \geq 0\}$. Quanto tempo uma máquina de Turing simples precisa para decidir A?

Complexidade de Tempo

Problema

Seja a linguagem $A = \{0^k 1^k \mid k \geq 0\}$. Quanto tempo uma máquina de Turing simples precisa para decidir A?

Descrição de uma possível MT simples

M_1 = "Sobre a cadeia de entrada ω:

1. Faça uma varredura na fita e *rejeite* se um 0 for encontrado à direita de um 1.
2. Repita se ambos 0s e 1s permanecem sobre a fita:
    1. Faça uma varredura na fita, cortando um único 0 e um único 1.
3. Se 0s ainda permanecerem após todos os 1s tiverem sido cortados, ou se 1s ainda permanecerem após todos os 0s tiverem sido cortados, *rejeite*. Caso contrário, se nem 0s nem 1s permanecerem sobre a fita, *aceite*.

Complexidade de Tempo

Analisando a entrada

- Grafo: número de nós, número de arestas;
- Estrutura de dados: tamanho do vetor, altura da árvore;
- Cadeia: tamanho da cadeia de entrada.

UFG Regional Jataí

Complexidade de Tempo

Analisando a entrada

- Grafo: número de nós, número de arestas;
- Estrutura de dados: tamanho do vetor, altura da árvore;
- Cadeia: tamanho da cadeia de entrada.

Tipos de Análise

- Análise do pior caso;
- Análise do caso médio;
- Análise do melhor caso.

UFG
Regional Jataí

Complexidade de Tempo

Analisando a entrada

- Grafo: número de nós, número de arestas;
- Estrutura de dados: tamanho do vetor, altura da árvore;
- Cadeia: tamanho da cadeia de entrada.

Tipos de Análise

- Análise do pior caso;
- Análise do caso médio;
- Análise do melhor caso.

Utilizaremos aqui...

O tamanho da cadeia de entrada e a análise de pior caso.

Complexidade de Tempo

Definição 7.1

Seja M uma máquina de Turing determinística que pára sobre todas as entradas. O tempo de execução ou **complexidade de tempo** de M é a função $f : \mathbb{N} \rightarrow \mathbb{N}$, em que $f(n)$ é o número máximo de passos que M usa sobre qualquer entrada de comprimento n.

Se $f(n)$ for o tempo de execução de M, dizemos que M *roda* em tempo $f(n)$ e que M é uma máquina de Turing *de tempo* $f(n)$. Costumeiramente usamos n para representar o comprimento da entrada.

UFG
Regional Jataí

Complexidade de Tempo

Notação O-Grande

Sejam f e g funções $f, g : \mathbb{N} \to \mathbb{R}^+$.
Vamos dizer que $f(n) = O(g(n))$ se inteiros positivos c e n_0 existem tais que para todo inteiro $n \geq n_0$ em que

$$f(n) \leq c.g(n)$$

Quando $f(n) = O(g(n))$, dizemos que $g(n)$ é um **limitante superior** para $f(n)$, ou mais precisamente, que $g(n)$ é um **limitante superior assintótico** para $f(n)$, para enfatizar que estamos suprimindo fatores constantes.

Complexidade de Tempo

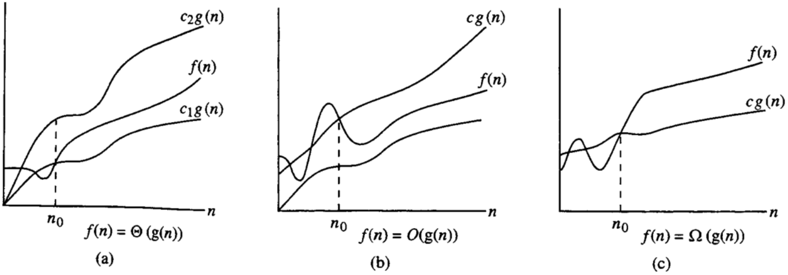

Figura: Comportamento das notações Θ, O e Ω.

Complexidade de Tempo

$f_1(n) = 5n^3 + 2n^2 + 22n + 6$

$$
\begin{aligned}
O(f_1(n)) &= O(5n^3 + 2n^2 + 22n + 6) \qquad (1)\\
&= O(5n^3) \qquad (2)\\
&= O(n^3) \qquad (3)
\end{aligned}
$$

Complexidade de Tempo

$f_1(n) = 5n^3 + 2n^2 + 22n + 6$

$$
\begin{aligned}
O(f_1(n)) &= O(5n^3 + 2n^2 + 22n + 6) && (1)\\
&= O(5n^3) && (2)\\
&= O(n^3) && (3)
\end{aligned}
$$

É verdade porque...

Basta admitir $c = 6$, e $n_0 = 10$. Logo

$$5n^3 + 2n^2 + 22n + 6 \leq 6n^3$$

para todo $n \geq 10$.

Complexidade de Tempo

$f_1(n) = 5n^3 + 2n^2 + 22n + 6$

$$
\begin{aligned}
O(f_1(n)) &= O(5n^3 + 2n^2 + 22n + 6) && (4)\\
&= O(5n^3) && (5)\\
&= O(n^3) && (6)
\end{aligned}
$$

Complexidade de Tempo

$f_1(n) = 5n^3 + 2n^2 + 22n + 6$

$$
\begin{aligned}
O(f_1(n)) &= O(5n^3 + 2n^2 + 22n + 6) && (4)\\
&= O(5n^3) && (5)\\
&= O(n^3) && (6)
\end{aligned}
$$

Também é verdade dizer que...

$f_1(n) = O(n^4)$, pois n^4 é maior que n^3 e portanto é ainda um limitante assintótico superior sobre f_1.

UFG
Regional Jataí

Complexidade de Tempo

$f_1(n) = 5n^3 + 2n^2 + 22n + 6$

$$
\begin{aligned}
O(f_1(n)) &= O(5n^3 + 2n^2 + 22n + 6) && (4)\\
&= O(5n^3) && (5)\\
&= O(n^3) && (6)
\end{aligned}
$$

Também é verdade dizer que...

$f_1(n) = O(n^4)$, pois n^4 é maior que n^3 e portanto é ainda um limitante assintótico superior sobre f_1.

Mas...

$f_1(n) \neq O(n^2)$.

Complexidade de Tempo

$f_2(n) = \log_{13} n + 5$

Complexidade de Tempo

$f_2(n) = \log_{13} n + 5$

$$
\begin{aligned}
O(f_2(n)) &= O(\log_{13} n + 5) && (7)\\
&= O(\log_{13} n) && (8)\\
&= O(\log n) && (9)
\end{aligned}
$$

Complexidade de Tempo

$f_2(n) = \log_{13} n + 5$

$$
\begin{aligned}
O(f_2(n)) &= O(\log_{13} n + 5) && (7)\\
&= O(\log_{13} n) && (8)\\
&= O(\log n) && (9)
\end{aligned}
$$

Porque...

$$\log n = \log_{10} n = \frac{\log_{13} n}{\log_{13} 10}$$

UFG
Regional Jataí

Complexidade de Tempo

$f_3(n) = 3n\log_2 n + 5n\log_2\log_2 n + 2$

Complexidade de Tempo

$f_3(n) = 3n\log_2 n + 5n\log_2\log_2 n + 2$

$$
\begin{aligned}
O(f_3(n)) &= O(3n\log_2 n + 5n\log_2\log_2 n + 2) \quad (10)\\
&= O(3n\log_2 n) \quad (11)\\
&= O(n\log n) \quad (12)
\end{aligned}
$$

Complexidade de Tempo

$f_3(n) = 3n\log_2 n + 5n\log_2\log_2 n + 2$

$$
\begin{aligned}
O(f_3(n)) &= O(3n\log_2 n + 5n\log_2\log_2 n + 2) && (10)\\
&= O(3n\log_2 n) && (11)\\
&= O(n\log n) && (12)
\end{aligned}
$$

Porque...

$n\log n$ domina sobre $\log\log n$.

Complexidade de Tempo

Esdras Lins Bispo Jr.
bispojr@ufg.br

Teoria da Computação
Bacharelado em Ciência da Computação

27 de junho de 2016