Classe P

Esdras Lins Bispo Jr.
bispojr@ufg.br

Teoria da Computação
Bacharelado em Ciência da Computação

08 de fevereiro de 2018

Plano de Aula

Sumário

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

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*.

JFG
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.

UFG
Regional Jataí

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 $f : \mathbb{N} \to \mathbb{N}$, 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

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.

UFG
Regional Jataí

Complexidade de Tempo

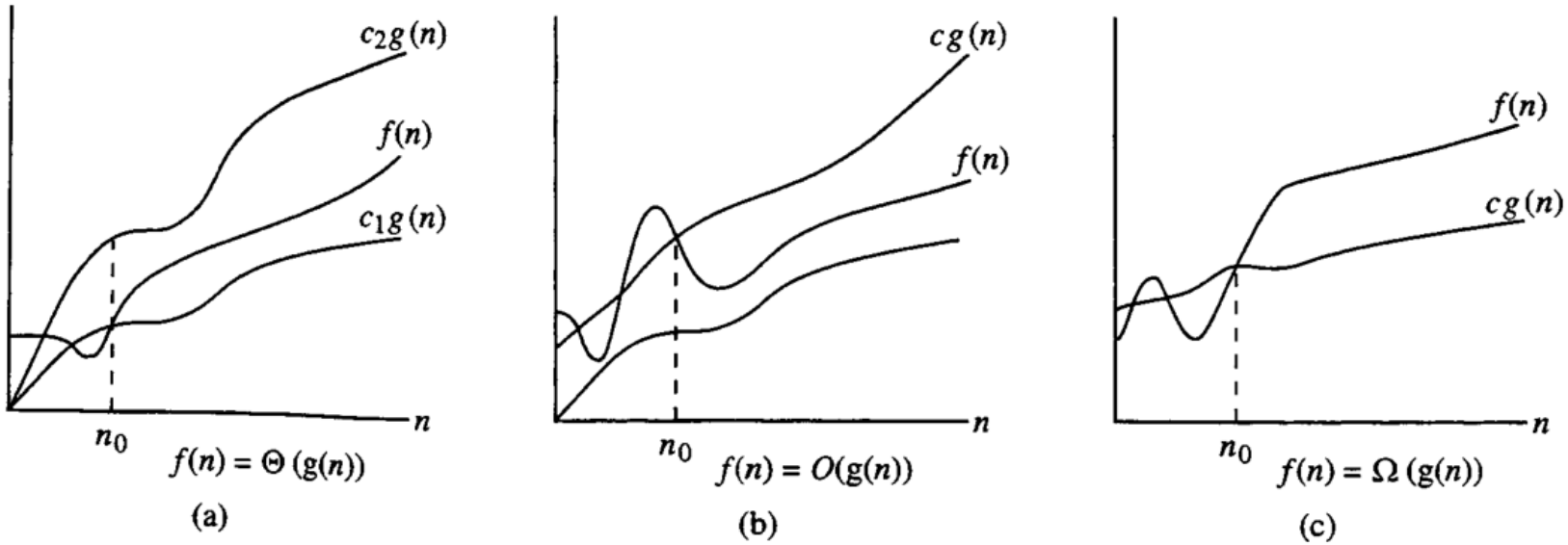


Figura: Comportamento das notações Θ, O e Ω.

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

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

É verdade porque...

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

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

para todo $n \geq 10$.

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

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

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...

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

UFG Regional Jataí

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

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

Porque...

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

UFG
Regional Jataí

Sumário

Definição 7.7

Seja $t : \mathbb{N} \to \mathbb{R}^+$ uma função. Defina a **classe de complexidade de tempo**, **TIME($t(n)$)**, como sendo a coleção de todas as linguagens que são decidíveis por uma máquina de Turing de tempo $O(t(n))$.

Definição 7.7

Seja $t : \mathbb{N} \to \mathbb{R}^+$ uma função. Defina a **classe de complexidade de tempo**, $\mathbf{TIME}(t(n))$, como sendo a coleção de todas as linguagens que são decidíveis por uma máquina de Turing de tempo $O(t(n))$.

Exemplo

- $A = \{0^k 1^k \mid k \geq 0\}$
- $A \in \mathbf{TIME}(n^2)$, pois

UFG
Regional Jataí

Definição 7.7

Seja $t : \mathbb{N} \to \mathbb{R}^+$ uma função. Defina a **classe de complexidade de tempo**, $\mathbf{TIME}(t(n))$, como sendo a coleção de todas as linguagens que são decidíveis por uma máquina de Turing de tempo $O(t(n))$.

Exemplo

- $A = \{0^k 1^k \mid k \geq 0\}$
- $A \in \mathbf{TIME}(n^2)$, pois
- M_1 decide A em tempo $O(n^2)$

UFG
Regional Jataí

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*.

JFG
Regional Jataí

Problema

Existe uma máquina que decide assintoticamente a linguagem A mais rapidamente?

Problema

Existe uma máquina que decide assintoticamente a linguagem A mais rapidamente?

Com outras palavras...

$A \in \mathbf{TIME}(t(n))$, para algum $t(n) = o(n^2)$?

Descrição de uma outra MT simples

M_2 = "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 enquanto alguns 0s e alguns 1s permanecem sobre a fita:
   1. Faça uma varredura na fita, verificando se o número total de 0s e 1s remanescentes é par ou ímpar. Se for ímpar, *rejeite*.
   2. Faça uma varredura novamente na fita,
      cortando alternadamente um 0 não e outro sim
      (começando com o primeiro 0)
      e então cortando alternadamente um 1 não e outro sim
      (começando com o primeiro 1).
3. Se nenhum 0 e nenhum 1 permanecer sobre a fita, *aceite*. Caso contrário, *rejeite*.

UFG Regional Jataí

Problema

Podemos decidir a linguagem A em tempo $O(n)$ (também chamado **tempo linear**)?

Problema

Podemos decidir a linguagem A em tempo $O(n)$ (também chamado **tempo linear**)?

Sim... é possível!

Se utilizarmos uma máquina de Turing com duas fitas!

Descrição de uma outra MT simples

M_3 = "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. Faça uma varredura nos 0s sobre a fita 1 até o primeiro 1. Ao mesmo tempo, copie os 0s para a fita 2.
3. Faça uma varredura nos 1s sobre a fita 1 até o final da entrada. Para cada 1 lido sobre a fita 1, corte um 0 sobre a fita 2. Se todos os 0s estiverem cortados antes que todos os 1s sejam lidos, *rejeite*.
4. Se todos os 0s tiverem agora sido cortados, *aceite*. Se algum 0 permanecer, *rejeite*.

UFG
Regional Jataí

Teorema 7.8

Seja $t(n)$ uma função, em que $t(n) \geq n$. Então toda máquina de Turing multifita de tempo $t(n)$ tem uma máquina de Turing de um única fita equivalente de tempo $O(t^2(n))$.

Teorema 7.8

Seja $t(n)$ uma função, em que $t(n) \geq n$. Então toda máquina de Turing multifita de tempo $t(n)$ tem uma máquina de Turing de um única fita equivalente de tempo $O(t^2(n))$.

Teorema 7.11

Seja $t(n)$ uma função, em que $t(n) \geq n$. Então toda máquina de Turing não-determinística de uma única fita de tempo $t(n)$ tem uma máquina de Turing de um única fita equivalente de tempo $2^{O(t(n))}$.

UFG
Regional Jataí

Diferenças de complexidade de tempo

- MT simples *x* MT multi-fita:
  potência quadrática (ou *polinomial*)
- MT simples *x* MT não-determinística:
  no máximo *exponencial*.

UFG
Regional Jataí

Diferenças entre as taxas de crescimento

Exemplo: n^3 e 2^n

Diferenças entre as taxas de crescimento

Exemplo: n^3 e 2^n

- Admita $n = 1000$;

Diferenças entre as taxas de crescimento

Exemplo: n^3 e 2^n

- Admita $n = 1000$;
- Logo, $n^3 = 1$ bilhão;

Diferenças entre as taxas de crescimento

Exemplo: n^3 e 2^n

- Admita $n = 1000$;
- Logo, $n^3 = 1$ bilhão;
- Mas, 2^n é maior que o número de átomos do universo.

UFG
Regional Jataí

Definição 7.12

P é a classe de linguagens que são decidíveis em tempo polinomial sobre uma máquina de Turing determinística de uma única fita.
Em outras palavras

$$\mathbf{P} = \bigcup_k \mathbf{TIME}\ (n^k).$$

Definição 7.12

P é a classe de linguagens que são decidíveis em tempo polinomial sobre uma máquina de Turing determinística de uma única fita.
Em outras palavras

$$\mathbf{P} = \bigcup_k \mathbf{TIME}\ (n^k).$$

P é importante porque...

Definição 7.12

P é a classe de linguagens que são decidíveis em tempo polinomial sobre uma máquina de Turing determinística de uma única fita. Em outras palavras

$$\mathbf{P} = \bigcup_k \mathbf{TIME}\ (n^k).$$

P é importante porque...

- **P** é invariante para todos os modelos de computação polinomialmente equivalentes à máquina de Turing determinística de uma única fita;

Definição 7.12

P é a classe de linguagens que são decidíveis em tempo polinomial sobre uma máquina de Turing determinística de uma única fita. Em outras palavras

$$\mathbf{P} = \bigcup_k \mathbf{TIME}\ (n^k).$$

P é importante porque...

- **P** é invariante para todos os modelos de computação polinomialmente equivalentes à máquina de Turing determinística de uma única fita;
- **P** corresponde aproximadamente à classe de problemas que são realisticamente solúveis em um computador.

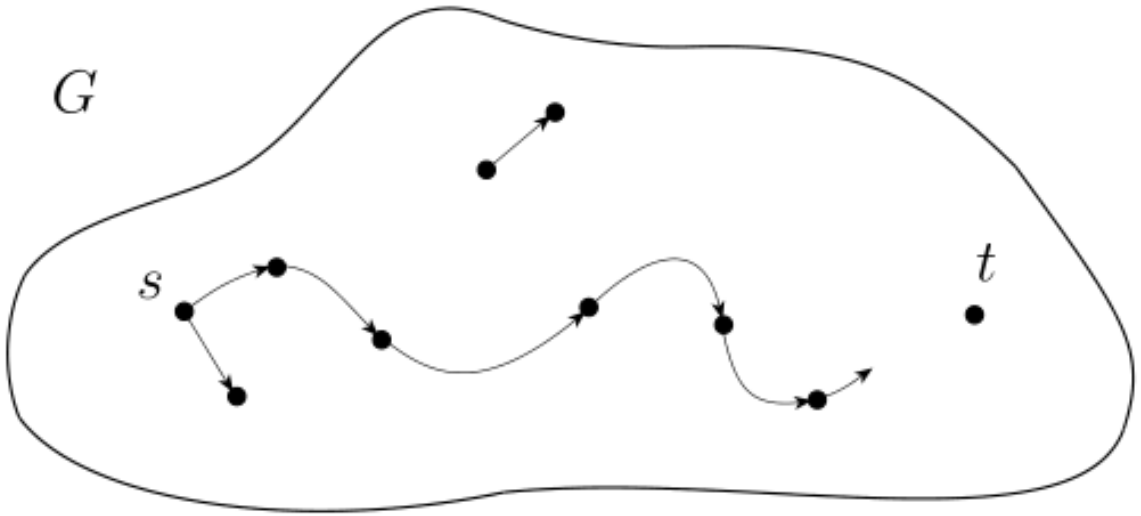
JFG
Regional Jataí

Problema do caminho em um grafo

$CAM = \{\langle G, s, t\rangle \mid G$ é um grafo direcionado que tem um caminho direcionado de s para $t\}$.

Problema do caminho em um grafo

$CAM = \{\langle G, s, t\rangle \mid G$ é um grafo direcionado que tem um caminho direcionado de s para $t\}$.



UFG
Regional Jataí

Teorema 7.14

$CAM \in \mathbf{P}$

Teorema 7.14

$CAM \in \mathbf{P}$

Prova

$M =$ "Sobre a cadeia de entrada $\langle G, s, t \rangle$ em que G é um grafo direcionado com nós s e t:

1. Ponha uma marca sobre o nó s.
2. Repita o seguinte até que nenhum nó adicional seja marcado:
   1. Faça uma varredura em todas as arestas de G. Se uma aresta (a, b) for encontrada indo de um nó marcado a para um nó não marcado b, marque o nó b.
3. Se t estiver marcado, *aceite*. Caso contrário, *rejeite*.

UFG Regional Jataí

Classe P

Esdras Lins Bispo Jr.
bispojr@ufg.br

Teoria da Computação
Bacharelado em Ciência da Computação

08 de fevereiro de 2018