Complexidade de Algoritmos

Paulino Ng

2020-03-20

Plano da aula

Esta aula apresenta as funções típicas de complexidade de algoritmos. Para tanto começamos com uma rápida revisão matemática.

1. Comparação de tempos de execução
2. Revisão Matemática
3. Comportamento assintótico de curvas

Comparação de Tempos de Execução

Para comparar tempos de execução, vamos resolver o problema 1-1 do [CLRS]:

> *Para cada função $f(n)$ e o tempo t na tabela abaixo, determine o maior tamanho n de um problema que pode ser resolvido em tempo t. Assume-se que o algoritmo para resolver o problema leva $f(n)$ microsegundos.*

	1 *segundo*	1 *minuto*	1 *hora*	1 *dia*	1 *mes*	1 *ano*	1 *século*
$lg\ n$							
$\sqrt{n}$							
n							
$n\ lg\ n$							
n^2							
n^3							
2^n							
$n!$							

Solução para t = 1s = $10^6 \mu$s = 1.000.000 μs

- $lg\ n = 10^6 \rightarrow n = 2^{10^6} \approx 10^{301030}$
- $\sqrt{n} = 10^6 \rightarrow n = 10^{12}$
- $n = 10^6$
- $n^2 = 10^6 \rightarrow n = 10^3$
- $n^3 = 10^6 \rightarrow n = 10^2$
- $2^n = 10^6 \rightarrow n = 6.\log_2 10 \approx 20$
- $n! = 10^6 \rightarrow n \approx 9$

Calcule o restante da tabela. Observe como o aumento do tempo não resulta num aumento muito significativo do n para as duas últimas funções.

$1s = 10^6 \mu s; \quad 1\mathrm{min} = 60\ .\ 10^6 \mu s = 6\ .\ 10^7 \mu s;$
$1h = 3,6\ .\ 10^9 \mu s; \quad 1\mathrm{dia} = 8,64\ .\ 10^{10} \mu s;$
$1\mathrm{mes} = 2,592\ .\ 10^{12} \mu s; \quad 1\mathrm{ano} = 3,1104\ .\ 10^{13} \mu s$

Revisão matemática: Séries

- Somatória de séries finitas. Seja a sequência: $a_0, a_1, \ldots, a_{n-1}$:
  - Série aritmética:
    $a_i = a_0 + i.d \longrightarrow \sum_{i=0}^{n-1} a_i = \frac{1}{2}n(a_0 + a_{n-1}) = \frac{1}{2}(2a_0 + (n-1)d)$
    - $1 + 1 + \ldots + 1 = n$
    - $1 + 2 + \ldots + n = \frac{1}{2}n(n+1)$
    - $1 + 3 + \ldots + (2n-1) = n^2$
  - série geométrica: $a_i = a_0 r^i \longrightarrow \sum_{i=0}^{n-1} a_i = \frac{a_0(1-r^n)}{1-r} = \frac{a_0 - r a_{n-1}}{1-r}$
    - $a_0 + a_0.r + a_0.r^2 + \ldots = \frac{a}{1-r} \quad \text{para} -1 \leq r \leq 1$
    - $1 + \frac{1}{2} + \frac{1}{4} + \ldots + \frac{1}{2^7} = \frac{1(1-\frac{1}{2^8})}{1-\frac{1}{2}} = 1,9921875$
    - $1 + \frac{1}{2} + \frac{1}{4} + \ldots = 2$

Séries importantes

- $1 - \frac{1}{2} + \frac{1}{3} - \frac{1}{5} + \ldots = \ln 2$
- $1^2 + 2^2 + 3^2 + \ldots + n^2 = \frac{n(n+1)(2n+1)}{6}$
- $1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \ldots = \frac{\pi}{4}$
- $e^n = 1 + n + \frac{n^2}{2!} + \frac{n^3}{3!} + \ldots$
- $a^n = e^{n \ln a} = 1 + n \ln a + \frac{(n \ln a)^2}{2!} + \frac{(n \ln a)^3}{3!} + \ldots$

Assíntotas

- [ZIVIANI] afirma que a *escolha do algoritmo* não é um problema crítico para problemas de tamanho pequeno. Logo, a análise de algoritmos é realizada para valores grandes de n.
- Estuda-se o *comportamento assintótico* das **funções de custo**.
  Definição: *Uma função $f(n)$ domina assintoticamente outra função $g(n)$ se existem duas constantes positivas c e m tais que, para $n \geq m$, temos $|g(n)| \leq c|f(n)|$.*

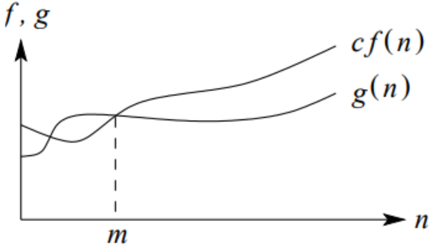

Figure 1: A função f(n) domina assintoticamente a função g(n) [ZIVIANI].

Exemplos de dominação assintótica

- $g(n) = n$ é assintoticamente dominada por $f(n) = -n^2$ para todo n natural.
- $g(n) = (n+1)^2$ e $f(n) = n^2$ dominam assintoticamente uma à outra.
- As funções dadas na tabela exercício do início da aula foram escolhidas de maneira que as funções das linhas inferiores dominam assintoticamente as funções das linhas superiores.

Notação *big-O*

- Segundo [ZIVIANI], Knuth sugeriu a notação usada atualmente para indicar que uma função $f(n)$ domina assintoticamente a outra, $g(n)$: $\mathcal{O}(f(n)) = g(n)$
- A wikipedia diz que a notação é parte da notação de Bachmann–Landau.
- Uma definição mais formal é:
  Seja g uma função complexa, ou real, e f uma função real, ambas definidas num subconjunto não limitado dos números reais positivos, tais que $f(x)$ é estritamente positiva para todos os x suficientemente grandes. Escreve-se: $g(x) = \mathcal{O}(f(x))$ com $x \to \infty$ se, e apenas se, para todo valor suficientemente grande de x, o valor absoluto de $g(x)$ é no máximo igual a uma constante positiva vezes o $f(x)$.

Propriedades e Exemplos

[ZIVIANI] Uma função $g(n)$ é $\mathcal{O}(f(n))$ se existem duas constantes positivas c e m tais e $g(n) \leq c\ f(n)$, $\forall\ n \geq m$.

- Exemplo: Seja $g(n) = (n+1)^2$. Logo, $g(n)$ é $\mathcal{O}(n^2)$, quando $m = 1$ e $c = 4$. Isso porque $(n+1)^2 \leq 4\ n^2$ para $n \geq 1$.
- Exemplo: A função $g(n) = 3\ n^3 + 2\ n^2 + n$ é $\mathcal{O}(n^3)$.
- Exemplo: A função $g(n) = \log_5\ n$ é $\mathcal{O}(\log\ n)$.

Propriedades

$$
\boxed{
\begin{aligned}
f(n) &= \mathcal{O}(f(n)) \\
c\ \times \mathcal{O}(f(n)) &= \mathcal{O}(f(n)) \quad c = constante \\
\mathcal{O}(f(n)) + \mathcal{O}(f(n)) &= \mathcal{O}(f(n)) \\
\mathcal{O}(\mathcal{O}(f(n))) &= \mathcal{O}(f(n)) \\
\mathcal{O}(f(n)) + \mathcal{O}(g(n)) &= \mathcal{O}(max(f(n), g(n))) \\
\mathcal{O}(f(n))\ \mathcal{O}(g(n)) &= \mathcal{O}(f(n)\ g(n)) \\
f(n)\ \mathcal{O}(g(n)) &= \mathcal{O}(f(n)\ g(n))
\end{aligned}
}
$$

Exercícios

1. Suponha que um programa tenha 3 trechos com tempos de execução $\mathcal{O}(n)$, $\mathcal{O}(n^2)$ e $\mathcal{O}(n \log n)$. Qual o tempo de execução do programa como um todo?
2. Indique quais afirmações abaixo são verdadeiras ou falsas:
   a. $2^{n+1} = \mathcal{O}(2^n)$
   b. $2^{2n} = \mathcal{O}(2^n)$
   c. $f(n) = \mathcal{O}(u(n)) \text{ e } g(n) = \mathcal{O}(v(n)) \Rightarrow f(n) + g(n) = \mathcal{O}(u(n) + v(n))$
   d. $f(n) = \mathcal{O}(u(n)) \text{ e } g(n) = \mathcal{O}(v(n)) \Rightarrow f(n) - g(n) = \mathcal{O}(u(n) - v(n))$
3. [desafio] Prove que $f(n) = 1^2 + 2^2 + \ldots + n^2$ é igual a $\frac{n^3}{3} + \mathcal{O}(n^2)$

Outras definições

- A notação big-O diz que $f(n)$ é um limite superior para a taxa de crescimento da função $g(n)$. Outras definições permitem outras aproximações assintóticas.
- Definição da notação Ω: Uma função $g(n)$ é $\Omega(f(n))$ se existirem duas constantes c e m tais que $g(n) \geq c\ f(n)$, para todo $n \geq m$.
  - $g(n)$ é $\Omega(f(n))$ quer dizer que $f(n)$ é um limite inferior para a taxa de crescimento de $g(n)$.
  - Exemplo: $g(n) = 3\ n^3 + 2\ n^2$ é $\Omega(n^3)$
- Definição notação Θ: Uma função $g(n)$ é $\Theta(f(n))$ se existirem constantes positivas c_1, c_2 e m tais que $0 \leq c_1\ f(n) \leq g(n) \leq c_2\ f(n)$, para todo $n \geq m$.
  - $c_1 f(n)$ está abaixo de $g(n)$, $c_2 f(n)$ está acima de $g(n)$, dizemos que $f(n)$ é um **limite assintótico firme** de $g(n)$.
- Definição notação o: Uma função $g(n)$ é $o(f(n))$ se, para qualquer constante $c > 0$, então $0 \leq g(n) < c\ f(n)$, $\forall n \geq m$.
  - Exemplo: $2n = o(n^2)$, mas $2n^2 \neq o(n^2)$

Mais uma definição e Exercícios

- A diferença entre $\mathcal{O}$ e o é que na big-O existe uma constante c e na o a relação vale para todo c positivo.
- Definição da notação ω: Uma função $g(n)$ é $\omega(f(n))$ se, para qualquer constante $c > 0$, então $0 \leq c\ f(n) \leq g(n),\ \forall n \geq m$.
  - Exemplo: $\frac{n^2}{2} = \omega(n)$, mas $\frac{n^2}{2} \neq \omega(n^2)$