Crescimento de Funções

Profa. Juliana Kaizer Vizzotto

Projeto e Análise de Algoritmos

Roteiro

- Revisão
- Notação Assintótica

Revisão

- A ordem de crescimento do tempo de execução de um algoritmo, da uma caracterização simples da sua eficiência, possibilitando assim comparar a performance relativa de algoritmos alternativos.
- Quando consideramos tamanhos de entrada grandes o suficiente tal que somente a ordem de crescimento do tempo é relevante, estamos estudando a eficiência **assintótica** de algoritmos.
- Isto é, estamos interessados em quanto o tempo de execução de um algoritmo aumenta com o tamanho da entrada.
- Em geral, um algoritmo que é assintóticamente mais eficiente será a melhor escolha.
- Vamos estudar métodos para simplificar a análise assintótica de algoritmos.

Notação Assintótica

- As notações que usamos para descrever o tempo de execução assintótico de um algoritmos são definidas em termos de funções cujo os domínios são o conjunto dos números naturais $\mathbf{N} = \{0, 1, 2, ...\}$.
- Tais notações são convenientes para descrever o tempo de execução $T(n)$, o qual é usualmente definido somente em entradas inteiras.
- Entretanto, algumas vezes podemos abusar da notação assintótica. Por exemplo, a notação pode ser facilmente estendida para o domínio dos números reais.
- Precisamos, contudo, entender o significado preciso da notação.

Notação Assintótica: Θ

- Para uma dada função $g(n)$, denotamos por $\Theta(g(n))$ o *conjunto de funções*:

$$\Theta(g(n)) = \{f(n) : \text{ existem constantes positivas } c_1, c_2 \text{ e } n_0, \text{ tal que } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \text{ para todo } n \geq n_0\}$$

- Um função $f(n)$ pertence ao conjunto $\Theta(g(n))$ se existem constantes positivas c_1 e c_2 tal que ela possa ser "imprensada" entre $c_1 g(n)$ e $c_2 g(n)$, para um n suficientemente grande.

Notação Assintótica: Θ

- Já vimos que, utilizando a notação assintótica, podemos cortar os termos de baixa ordem e ignorar os coeficientes. Vamos justificar essa intuição utilizando a definição formal para mostrar que $1/2n^2 - 3n = \Theta(n^2)$.
- Para fazer isso, precisamos determinar constantes, c_1, c_2 e n_0 tal que:

$$c1 * n^2 \leq 1/2n^2 - 3n \leq c_2 * n^2$$

para todo $n \geq n_0$. Dividindo por n^2 temos:

$$c_1 \leq 1/2 - 3/n \leq c_2.$$

Notação Assintótica: Θ

- A inequalidade do lado direito pode valer, para qualquer valor de $n \geq 1$, $c_2 \geq 1/2$.
- Da mesma maneira pode valer, para qualquer valor de $n \geq 7$, $c_1 \leq 1/14$.
- Então, escolhendo $c_1 = 1/14$, $c_2 = 1/2$, e $n_0 = 7$, podemos verificar que $1/2n^2 - 3n = \Theta(n^2)$.
- Certamente, podemos fazer outras escolhas para os valores das constantes, mas o importante é que alguma escolha exista.
- Note que essas constantes dependem da função $1/2n^2 - 3n$; uma função diferente, que também pertence a $\Theta(n^2)$ requer, em geral, constantes diferentes.

Notação Assintótica: Θ

- Podemos também, usar essa definição formal para verificar que $6n^3 \neq \Theta(n^2)$.
- Suponha para provar por contradição que c_2 e n_0 existem, tal que $6n^3 \leq c_2 n^2$, para todo $n \geq n_0$.
- Mas então, $n \leq c_2/6$, o qual pode possivelmente não valer para um n arbitrariamente grande, já que c_2 é constante.

Notação Assintótica: Θ

- Intuitivamente, os termos de baixa ordem de uma função assintóticamente positiva, podem ser ignorados quando vamos determinar limites assintóticos, pois eles são insignificantes para n grande.
- *Uma fração bem pequena dos termos de alta ordem é suficiente para dominar os termos de baixa ordem!*
- Assim, atribuindo a c_1 um valor que é um pouquinho menor que o coeficiente do termo de maior ordem e a c_2 um valor que é um pouquinho maior, permite que as inequalidades sejam satisfeitas.
- O coeficiente do termo de maior ordem pode também ser ignorado, pois ele somente troca c_1 e c_2 por um fator constante igual ao coeficiente.

Notação Assintótica: Θ

- Como um exemplo, considere qualquer função quadrática $f(n) = an^2 + bn + c$, onde a, b e c são constantes e $a > 0$.
- Desprezando os termos de mais baixa ordem e ignorando as constantes, obtemos: $f(n) = \Theta(n^2)$.
- Para mostrar formalmente, a mesma coisa, pegamos constantes $c_1 = a/4$, $c_2 = 7a/4$, e $n_0 = 2.max((|b|/a), \sqrt{(|c|/a)})$.
- Podemos verificar que $0 \leq c_1 n^2 \leq an^2 + bn + c \leq c_2 n^2$ para todo $n \geq n_0$.
- Em geral, para qualquer polinomial $p(n) = \sum_{i=0}^{d} a_i n_i$, onde a_i são constantes e $a_d > 0$, temos que $p(n) = \Theta(n^d)$.

Notação Assintótica: O

- A notação Θ coloca limites assintóticos inferiores e superiores na função.
- Quando temos somente um limite assintótico superior, usamos a notação O.
- Para uma dada função $g(n)$, denotamos por $O(g(n))$ (pronuncia-se "Ó grande", ou "Ó de g") o *conjunto de funções*:

$$\Theta(g(n)) = \{f(n) : \text{ existem constantes positivas } c \text{ e } n_0, \\ \text{tal que } 0 \leq f(n) \leq cg(n) \text{ para todo } n \geq n_0\}$$

- Note que $f(n) = \Theta(g(n))$ implica que $f(n) = O(g(n))$, pois Θ é uma notação mais forte que O.
- Teoricamente, $\Theta(g(n)) \subseteq O(g(n))$.
- Então, uma prova de que qualquer função quadrática $an^2 + bn + c$, onde $a > 0$ está em $\Theta(n^2)$, também mostra que qualquer função quadrática é $O(n^2)$.

Notação Assintótica: O

- Surpreendentemente, qualquer função linear $an + b$ está em $O(n^2)$, o que pode ser verificado facilmente, atribuindo $c = a + |b|$ e $n_0 = 1$.
- Como a notação O descreve um limite superior, quando usamos ela para o pior caso de um algoritmo, temos um limite no tempo do algoritmo em qualquer entrada.
- Então, $O(n^2)$ como limite no pior caso do insertion-sort também se aplica a seu tempo em qualquer entrada.
- Já $\Theta(n^2)$ como limite do pior caso do insertion sort, não se aplica para qualquer entrada.
- Por exemplo, quando a entrada do insertion sort já está ordenada, seu tempo é $\Theta(n)$.

Notação Assintótica: O

- Tecnicamente, é um abuso dizer que o tempo do insertion sort é $O(n^2)$, já que para um dado n, o tempo varia, dependendo da entrada particular de tamanho n.
- Quando dizemos que "o tempo é $O(n^2)$", significa que existe uma função $f(n)$ que é $O(n^2)$, tal que para qualquer valor de n, não importando qual entrada particular de tamanho n, o tempo nessa entrada é ligado por $f(n)$.
- Equivalentemente, isso significa que no pior caso é $O(n^2)$.

Exercícios: Notação Θ e O

1. Prove formalmente que (usando a definição de Θ) que,$1/2 * n^2 - 3 * n = \Theta(n^2)$.
2. $n^2 = O(n^3)$?
3. Explique porque a declaração "O tempo de execução no algoritmo A é no mínimo $O(n^2)$"é isenta de significado.
4. É verdade que $2^{n+1} = O(2^n)$? É verdade que $2^{2n} = O(2^n)$?