AE2ADS: Algorithms Data Structures and Efficiency

Lecturer: Heshan Du
Email: Heshan.Du@nottingham.edu.cn

University of Nottingham Ningbo China

Big-Oh

Let $f(n)$ and $g(n)$ be functions mapping positive integers to positive real numbers.

We say that $f(n)$ is $O(g(n))$, if there exist a real constant $c > 0$ and an integer constant $n_0 \geq 1$ such that for every $n \geq n_0$, $f(n) \leq cg(n)$.

Exercise 1

- 5 is $O(1)$
  - $c = 5, n_0 = 1$
- 6 is $O(n)$
  - $c = 6, n_0 = 1$
- $2n + 3$ is $O(n)$
  - $c = 5, n_0 = 1$
- $3\log n$ is $O(n)$
  - $c = 3, n_0 = 2$
- $\frac{1}{n}$ is $O(1)$
  - $c = 1, n_0 = 1$

Exercise 2

- $100n + 1000$ is $O(n)$
  - $c = 1100, n_0 = 1$
- $n^2 + 8n - 6$ is $O(n^2)$
  - $c = 9, n_0 = 1$
- $n \log n$ is $O(n^2)$
  - $c = 1, n_0 = 2$
- $(\log n)^2$ is $O(n \log n)$
  - $c = 1, n_0 = 2$
- n^3 is $O(2^n)$
  - $c = 1, n_0 = 10$

Multiplication Rule for Big-Oh

Let $d(n), f(n), e(n), g(n)$ be functions mapping positive integers to positive real numbers.

Show that if $d(n)$ is $O(f(n))$ and $e(n)$ is $O(g(n))$, then $d(n)e(n)$ is $O(f(n)g(n))$.

Multiplication Rule for Big-Oh

Let $d(n), f(n), e(n), g(n)$ be functions mapping positive integers to positive real numbers. Show that if $d(n)$ is $O(f(n))$ and $e(n)$ is $O(g(n))$, then $d(n)e(n)$ is $O(f(n)g(n))$.

Proof: Suppose $d(n)$ is $O(f(n))$ and $e(n)$ is $O(g(n))$. Then by the definition of Big-Oh, there exist real positive constant c_1 and real positive natural number n_1 such that for every $n \geq n_1$, $d(n) \leq c_1 f(n)$. Similarly, there exist real positive constant c_2 and real positive natural number n_2 such that for every $n \geq n_2$, $e(n) \leq c_2 g(n)$.

Let $n_3 = \max(\mathrm{n}_1, \mathrm{n}_2)$, $c_3 = c_1 c_2$. Then it is easy to see that for every $n \geq n_3$, $d(n)e(n) \leq c_3\big(f(n)g(n)\big)$. By the definition of Big-Oh, *d(n)e(n)* is *O(f(n)g(n))*.

Big-Oh Rules: Drop smaller terms

Let $f(n), h(n)$ be functions mapping positive integers to positive real numbers. Show that if $f(n) = (1 + h(n))$ with $h(n) \to 0$ as $n \to \infty$, then $f(n)$ is $O(1)$.

Big-Oh Rules: Drop smaller terms

Show that if $f(n) = (1 + h(n))$ with $h(n) \rightarrow 0$ as $n \rightarrow \infty$, then $f(n)$ is $O(1)$.

Proof (sketch):

- $h(n) \rightarrow 0$ as $n \rightarrow \infty$ means that for large enough n then $h(n)$ will become arbitrarily close to 0.
- Hence, there exists n_0 such that

$$h(n) \leq 1 \text{ for all } n \geq n_0$$

- Hence, $f(n) \leq 2$ for all $n \geq n_0$.
- Therefore, $f(n)$ is $O(1)$.

Exercise 3

What is big-Oh of each of the following functions? Apply the big-Oh rules to justify your answer.

1. $f(n) = n^2 + n$
2. $f(n) = n^2 + 2^n$
3. $f(n) = (n \log n) + n^2$

Exercise 3

What is big-Oh of each of the following functions? Apply the big-Oh rules to justify your answer.

1. $f(n) = n^2 + n = n^2\left(1 + \frac{1}{n}\right) \in O(n^2)$
2. $f(n) = n^2 + 2^n = 2^n\left(1 + \frac{n^2}{2^n}\right) \in O(2^n)$
3. $f(n) = (n \log n) + n^2 = n^2\left(1 + \frac{n \log n}{n^2}\right) \in O(n^2)$

Exercise 4

What is big-Oh of each of the following functions? Apply the big-Oh rules to justify your answer.

1. $f(n) = 5n^2 + 1000n + 10000$
2. $f(n) = 6n^2 + 2^n/1000$
3. $f(n) = (10000n \log n) + n^2$

Exercise 4

What is big-Oh of each of the following functions? Apply the big-Oh rules to justify your answer.

1. $f(n) = 5n^2 + 1000n + 10000 \in O(n^2)$
2. $f(n) = 6n^2 + 2^n/1000 \in O(2^n)$
3. $f(n) = (10000n \log n) + n^2 \in O(n^2)$

Exercise 5

Order the following functions by asymptotic growth rate.

$$4n(\log n) + 2n, 2^{20}, 2^{\log n}$$

$$3n + 100 \log n \ , 4n, 2^n$$

$$n^2 + 10n, n^3, n(\log n)$$

Exercise 5

Order the following functions by asymptotic growth rate.

$$4n(\log n) + 2n, 2^{20}, 2^{\log n}$$

$$3n + 100 \log n \ , 4n, 2^n$$

$$n^2 + 10n, n^3, n(\log n)$$

$2^{20} \in O(1)$; $2^{\log n} = n \in O(n)$

The rank from the lowest to the highest growth rates is:

$2^{20}, 2^{\log n}, 3n + 100 \log n \, , 4n, 4n \ (\log n) + 2n, n \ (\log n), n^2 + 10n, n^3, 2^n$

The order of $2^{\log n}, 3n + 100 \log n \, , 4n$ does not matter, as they are all in $O(n)$.
The order of $4n \ (\log n) + 2n, n \ (\log n)$ does not matter, as they are both in $O(n \log n)$

More Exercises

M. T. Goodrich, R. Tamassia and M. H. Goldwasser, *Data Structures and Algorithms in Java*, 6th Edition, 2014.

- Chapter 4. Analysis Tools