

# Mathematical Foundation

CSE 680

**Suggested Reading:** Sections 1.1, 1.2, 2.1, 2.2, 3.1, 3.2.

## 1 Running Time

- Analysis of algorithm: to estimate the running time of an algorithm.
- Elementary operations:
  - arithmetic and boolean operations:  $+$ ,  $-$ ,  $\times$ ,  $/$ , mod, div, and, or
  - comparison: **if**  $a < b$ , **if**  $a = b$ , etc.
  - branching: **go to**
  - assignment:  $a \leftarrow b$
  - and so on
- The *running time* of an algorithm is the number of elementary operations required to execute the algorithm.
- It depends on the input (instance):
  - size of the input
  - content of the input
- The *worst-case running time* of an algorithm:

$$T(n) = \max\{\text{running time over all instances of size } n\}$$

- Express  $T(n)$  in the  $O$ ,  $\Omega$ , or  $\Theta$  notation.
- These are called asymptotic notations. They describe the behavior of a function  $f(n)$  when  $n$  is sufficiently large.
- A function  $f(n)$  is *asymptotically positive* if  $f(n)$  is positive for sufficiently large  $n$ .
- A function  $f(n)$  is *asymptotically increasing* if  $f(n)$  is increasing for sufficiently large  $n$ .

## 2 $O$ -Notation

**Note:** Unless otherwise indicated, all functions considered in this class are assumed to be nonnegative.

- **Conventional Definition:** We say  $f(n) = O(g(n))$  iff there exist positive constants  $c$  and  $n_0$  such that  $f(n) \leq cg(n)$  for all  $n \geq n_0$ .

- **More Abstract Definition:**

$$O(g(n)) = \{f(n) : \text{there exist positive constants } c \text{ and } n_0 \\ \text{such that } f(n) \leq cg(n) \text{ for all } n \geq n_0\}$$

That is,

$$O(g(n)) = \{f(n) : f(n) = O(g(n) \text{ in the conventional meaning})\}$$

- Thus, the following all have the same meaning:
  - $f(n) \in O(g(n))$ .
  - $f(n) = O(g(n))$ .
- Question: Does  $f(n) = O(g(n))$  means  $f(n)$  and  $g(n)$  are in the same order of magnitude?

• **Theorem 1** If  $f_1(n) = O(g_1(n))$  and  $f_2(n) = O(g_2(n))$  then

1.  $f_1(n) + f_2(n) = O(\max(g_1(n), g_2(n)))$

2.  $f_1(n) \cdot f_2(n) = O(g_1(n) \cdot g_2(n))$

**Proof.**  $f_1(n) \leq c_1 g_1(n)$  and  $f_2(n) \leq c_2 g_2(n)$  for large  $n$ . Thus,

$$\begin{aligned} f_1(n) + f_2(n) &\leq c_1 g_1(n) + c_2 g_2(n) \\ &\leq c_1 \max(g_1(n), g_2(n)) + c_2 \max(g_1(n), g_2(n)) \\ &\leq (c_1 + c_2) \max(g_1(n), g_2(n)). \end{aligned}$$

By definition, equation 1 holds. Equation 2 can be similarly proved.

**Q.E.D.**

### 3 $\Omega, \Theta, o, \omega$ Notation

- **Conventional Definition of  $\Omega$**  : We say  $f(n) = \Omega(g(n))$  iff there exist positive constants  $c, n_0$  such that  $f(n) \geq cg(n)$  for all  $n \geq n_0$ .

- **More Abstract Definition of  $\Omega$** :

$$\Omega(g(n)) = \{f(n) : \text{there exist positive constants } c \text{ and } n_0 \\ \text{such that } f(n) \geq cg(n) \text{ for all } n \geq n_0\}$$

That is,

$$\Omega(g(n)) = \{f(n) : f(n) = \Omega(g(n)) \text{ in the conventional meaning}\}$$

- **Theorem 2** If  $f_1(n) = \Omega(g_1(n))$  and  $f_2(n) = \Omega(g_2(n))$  then

1.  $f_1(n) + f_2(n) = \Omega(\max(g_1(n), g_2(n)))$
2.  $f_1(n) \cdot f_2(n) = \Omega(g_1(n) \cdot g_2(n))$

- **Definition of  $\Theta$** :

$$f(n) = \Theta(g(n)) \text{ if and only if } f(n) = O(g(n)) \text{ and } f(n) = \Omega(g(n)).$$

In terms of set,  $\Theta(g(n)) = O(g(n)) \cap \Omega(g(n))$ .

- **Definition of  $o$** :  $f(n) = o(g(n))$  if  $\lim_{n \rightarrow \infty} f(n)/g(n) = 0$ .
- **Definition of  $\omega$** :  $f(n) = \omega(g(n))$  if  $\lim_{n \rightarrow \infty} f(n)/g(n) = \infty$ .

## 4 Properties of Asymptotic Notation

- $f(n) = O(f(n))$ .
- If  $f(n) = O(g(n))$  and  $g(n) = O(h(n))$  then  $f(n) = O(h(n))$ .
- $f(n) = O(g(n))$  iff  $g(n) = \Omega(f(n))$ .
- $f(n) = \Theta(g(n))$  iff  $g(n) = \Theta(f(n))$ .
- $\log_a n = \Theta(\log_b n)$  if  $a, b > 1$ .
- $a^n \neq \Theta(b^n)$  if  $a \neq b$ .

## 5 Some Notations, Functions, Formulas

- $\lfloor x \rfloor$  = the floor of  $x$ .
- $\lceil x \rceil$  = the ceiling of  $x$ .
- $\log n = \log_2 n$ .
- $1 + 2 + \dots + n = n(n+1)/2 = \Theta(n^2)$ .
- For a constant  $k > 0$ ,  $1 + 2^k + 3^k + \dots + n^k = \Theta(n^{k+1})$ .
- If  $a \neq 1$ , then
$$1 + a + a^2 + \dots + a^n = \frac{1 - a^{n+1}}{1 - a} = \frac{a^{n+1} - 1}{a - 1}.$$
- If  $a > 1$ , then  $1 + a + a^2 + \dots + a^n = \Theta(a^n)$ .
- If  $0 \leq a < 1$ , then  $1 + a + a^2 + \dots + a^n = \Theta(1)$ .