# Definition

For a given function $g : \mathbb{N} \to \mathbb{R}$,

$$\begin{aligned}
O(g(n)) = \{f(n) \ &: \exists \text{ positive constants } c \text{ and } n_0 \text{ such that} \\
&\quad 0 \le f(n) \le c\, g(n),\ \forall n \ge n_0\}, \\
\Omega(g(n)) = \{f(n) \ &: \exists \text{ positive constants } c \text{ and } n_0 \text{ such that} \\
&\quad 0 \le c\, g(n) \le f(n),\ \forall n \ge n_0\}, \\
\Theta(g(n)) = \{f(n) \ &: \exists \text{ positive constants } c_1, c_2 \text{ and } n_0 \text{ such that} \\
&\quad 0 \le c_1\, g(n) \le f(n) \le c_2\, g(n),\ \forall n \ge n_0\}, \\
o(g(n)) = \{f(n) \ &: \forall \text{ positive constant } c, \exists \text{ constant } n_0 \text{ such that} \\
&\quad 0 \le f(n) < c\, g(n),\ \forall n \ge n_0\}, \\
\omega(g(n)) = \{f(n) \ &: \forall \text{ positive constant } c, \exists \text{ constant } n_0 \text{ such that} \\
&\quad 0 \le c\, g(n) < f(n),\ \forall n \ge n_0\}.
\end{aligned}$$

# Theorem

1. For any two functions $f : \mathbb{N} \to \mathbb{R}$ and $g : \mathbb{N} \to \mathbb{R}$,

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

**Proof.**

$\boxed{\Longrightarrow}$

Assume that $f(n) = \Theta(g(n))$, that is, we have positive constants $\hat{c_1}, \hat{c_2}$ and $\hat{n_0}$ such that for all $n \ge \hat{n_0}$,

$$0 \le \hat{c_1}\, g(n) \le f(n) \le \hat{c_2}\, g(n).$$

Then we find $c = \hat{c_2}$ and $n_0 = \hat{n_0}$ such that

$$0 \le f(n) \le c\, g(n), \quad \forall n \ge n_0,$$

thus, $f(n) = O(g(n))$.

Similarly, we can find $c = \hat{c_1}$ and $n_0 = \hat{n_0}$ such that

$$0 \le c\, g(n) \le f(n), \quad \forall n \ge n_0,$$

thus, $f(n) = \Omega(g(n))$.

$\boxed{\Longleftarrow}$

Assume that $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$, that is, we have positive constants $c_1, c_2$ and $n_1, n_2$ such that

$$0 \le c_1\, g(n) \le f(n), \quad \forall n \ge n_1,$$

$$0 \le f(n) \le c_2\, g(n), \quad \forall n \ge n_2.$$

Then we can choose $n_0 = \max(n_1, n_2)$ such that

$$0 \le c_1\, g(n) \le f(n) \le c_2\, g(n), \quad \forall n \ge n_0,$$

thus, $f(n) = \Theta(g(n))$.

2. For any functions $f : \mathbb{N} \to \mathbb{R}^+$ and $g : \mathbb{N} \to \mathbb{R}^+$,

$$f(n) = o(g(n)) \Longleftrightarrow \lim_{n\to\infty} \frac{f(n)}{g(n)} = 0.$$

**Proof.**

$\boxed{\Longrightarrow}$

Assume that $f(n) = o(g(n))$, then we prove by the definition of limits.

Let $\epsilon > 0$ be arbitrary.

Because $f(n) = o(g(n))$, there exists positive constant $N$ such that for all $n \ge N$ we have $0 \le f(n) < \epsilon\, g(n)$, which yields

$$\left|\frac{f(n)}{g(n)}\right| < \epsilon, \quad \text{whenever } n \ge N$$

Hence, by the $\epsilon$-$N$ definition of a limit, $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$.

$\boxed{\Longleftarrow}$

Assume that $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$, then for any arbitrary constant $c > 0$, we can find a $n_0 > 0$ such that

$$n \ge n_0 \Longrightarrow \left|\frac{f(n)}{g(n)}\right| < c,$$

which yields

$$0 \le f(n) < c\, g(n), \quad \forall n \ge n_0.$$

Hence, we have $f(n) = o(g(n))$.

3. **Big-O Transitivity:** For any functions $f : \mathbb{N} \to \mathbb{R}$, $g : \mathbb{N} \to \mathbb{R}$, and $h : \mathbb{N} \to \mathbb{R}$,

$$f(n) = O(g(n)) \text{ and } g(n) = O(h(n)) \Longrightarrow f(n) = O(h(n)).$$

**Proof.**

Assume that $f(n) = O(g(n))$ and $g(n) = O(h(n))$, then there exist positive constants $c_1, n_1$ and $c_2, n_2$ such that

$$0 \le f(n) \le c_1\, g(n), \quad \forall n \ge n_1,$$

$$0 \le g(n) \le c_2\, h(n), \quad \forall n \ge n_2.$$

Combine those, we have

$$0 \le f(n) \le c_1\, g(n) \le c_1 c_2\, h(n), \quad \forall n \ge \max(n_1, n_2).$$

Thus, we can find positive constants $c_0 = c_1 c_2$ and $n_0 = \max(n_1, n_2)$ such that

$$0 \le f(n) \le c_0\, h(n), \quad \forall n \ge n_0.$$

Therefore, $f(n) = O(h(n))$.

4. **Little-o Transitivity:** For any functions $f : \mathbb{N} \to \mathbb{R}$, $g : \mathbb{N} \to \mathbb{R}$, and $h : \mathbb{N} \to \mathbb{R}$,

$$f(n) = o(g(n)) \text{ and } g(n) = o(h(n)) \Longrightarrow f(n) = o(h(n)).$$

**Proof.**

Assume that $f(n) = o(g(n))$ and $g(n) = o(h(n))$, then for every $\hat{c} > 0$, there exist positive constants $n_1$ and $n_2$ such that

$$0 \le f(n) < \hat{c}\, g(n), \quad \forall n \ge n_1,$$

$$0 \le g(n) < \hat{c}\, h(n), \quad \forall n \ge n_2.$$

Combine those, we have

$$0 \le f(n) < \hat{c}\, g(n) < \hat{c}^2\, h(n), \quad \forall n \ge \max(n_1, n_2).$$

Thus, for every $c > 0$, we have $\hat{c} = \sqrt{c}$ and choose $n_0 = \max(n_1, n_2)$ such that

$$0 \le f(n) < c\, h(n), \quad \forall n \ge n_0.$$

Therefore, $f(n) = o(h(n))$.

5. **Big-Theta Reflexivity:** For any function $f : \mathbb{N} \to \mathbb{R}^+$,

$$f(n) = \Theta(f(n)).$$

**Proof.**

We can simply choose any $c_1 \in (0, 1]$, $c_2 \in [1, \infty)$ and $n_0 \in [1, \infty)$ to satisfy

$$0 \leq c_1 f(n) \leq f(n) \leq c_2 f(n), \quad \forall n \geq n_0.$$

6. **Symmetry:** For any functions $f : \mathbb{N} \to \mathbb{R}$ and $g : \mathbb{N} \to \mathbb{R}$,

$$f(n) = \Theta(g(n)) \Longrightarrow g(n) = \Theta(f(n)).$$

**Proof.**

Assume that $f(n) = \Theta(g(n))$, then we have positive constants $c_1, c_2$ and $n_0$ such that

$$0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n), \quad \forall n \geq n_0,$$

which yields

$$0 \leq \frac{1}{c_2} f(n) \leq g(n) \leq \frac{1}{c_1} f(n), \quad \forall n \geq n_0.$$

Thus, $g(n) = \Theta(f(n))$.

# Exercises

1. Let $f(n)$ and $g(n)$ be asymptotically nonnegative functions. Using the basic definition of $\Theta$-notation, prove that $\max(f(n), g(n)) = \Theta(f(n) + g(n))$.

**Proof.**

Given that they are asymptotically nonnegative, we have $n_0$ such that

$$0 \leq f(n) \text{ and } 0 \leq g(n), \quad \forall n \geq n_0.$$

Then we can choose $c_1 = 1/2$ and $c_2 = 1$ such that

$$0 \leq c_1(f(n) + g(n)) \leq \max(f(n), g(n)) \leq c_2(f(n) + g(n)), \quad \forall n \geq n_0.$$

Thus, $\max(f(n), g(n)) = \Theta(f(n) + g(n))$.

2. Show that for any real constants $a$ and $b$, where $b > 0$,

$$(n+a)^b = \Theta(n^b).$$

**Proof.**

For the case where $a = 0$, we have $n^b = \Theta(n^b)$ by the reflexity of Big-Theta.

Now consider $a \neq 0$, our target is to find the positive constants $c_1, c_2$ and $n_0$ such that

$$0 \leq c_1 n^b \leq (n+a)^b \leq c_2 n^b, \quad \forall n \geq n_0.$$

Divide $n^b$ on both side, we get

$$c_1 \leq (1 + \frac{a}{n})^b \leq c_2.$$

Choose $n_0 = 2|a|$, then we have

$$(1+\frac{a}{n})^b = (1+\frac{|a|}{n})^b, \quad 1 \leq (1+\frac{|a|}{n})^b \leq (1+\frac{a}{n_0})^b = (\frac{3}{2})^b, \quad \text{if } a > 0,$$

$$(1+\frac{a}{n})^b = (1-\frac{|a|}{n})^b, \quad 1 \geq (1-\frac{|a|}{n})^b \geq (1-\frac{|a|}{n_0})^b = (\frac{1}{2})^b, \quad \text{if } a < 0.$$

Combine those, we have

$$(\frac{1}{2})^b \leq (1+\frac{a}{n})^b \leq (\frac{3}{2})^b, \quad \forall n \geq n_0.$$

Thus, we can always choose $c_1 = (\frac{1}{2})^b, c_2 = (\frac{3}{2})^b$, and $n_0 = 2|a|$ when $a \neq 0$.

Therefore, $(n+a)^b = \Theta(n^b)$.

3. Explain why the statement, "The running time of algorithm $A$ is at least $O(n^2)$," is meaningless.

**Answer.**

The Big-O notation speaks about an upper bound, while the term "at least" represents an lower bound. So the statement is saying that the growth of the running time $T(n)$ is "at least" "no more than" $n^2$, which is nonsense.

4. Is $2^{n+1} = O(2^n)$? Is $2^{2n} = O(2^n)$?

**Answer.**

The first part is true. Since $2^{n+1} = 2 \cdot 2^n$, we can simply choose $c = 3$ and $n_0 = 1$.

The second part is false. $2^{2n} = 4^n$ has more significant growth rate than $2^n$. Let's have a more rigorous proof.

Suppose to the contrary that we had positive constants $c$ and $n_0$ such that

$$0 \le 4^n \le c \cdot 2^n, \quad \forall n \ge n_0.$$

That is, $2^n \le c$. However, for any $c > 0$, we can still have a large enough $n > \log_2 c$ such that $2^n > c$. Therefore, by contradiction, $2^{2n} \ne O(2^n)$.

5. Prove Theorem 3.1. (This has been established at page 1.)

6. Prove that the running time of an algorithm is $\Theta(g(n))$ if and only if its worst-case running time is $O(g(n))$ and its best-case running time is $\Omega(g(n))$.

   **Draft.**

   When we say that the worst-case running time of an algorithm is $O(g(n))$, that means for sufficiently large input size $n$, the maximum running time on any input of size $n$ is bounded above by $g(n)$ with a constant factor. This implies that the running time of every input of size $n$ also has the same upper bound. Similarly, from the best-case running time being $\Omega(g(n))$, it means that the running time of every input of size $n$ is bounded below by $g(n)$ with a constant factor. Combine these two then we have $\Theta(g(n))$.

   **Proof.**

   Let $T(I)$ be the running time of an algorithm on a particular input $I$, where the size of the input is $|I| = n$. Denote its worst-case and best-case running time of a input size $n$ as

$$T_{\max}(n) = \max_{|I|=n} T(I)$$

$$T_{\min}(n) = \min_{|I|=n} T(I)$$

   $\boxed{\Longrightarrow}$

   Assume that the running time is $\Theta(g(n))$, then we have $c_1, c_2$ and $n_0$ such that

$$0 \le c_1\, g(n) \le T(I) \le c_2\, g(n), \quad \forall n \ge n_0 \text{ and } \forall I \text{ with } |I| = n.$$

   That is, for all input of size $n \ge n_0$, the worst-case running time is bounded above.

$$\forall I \text{ with } |I| = n, \quad 0 \le T(I) \le c_2\, g(n),$$

$$\Rightarrow 0 \le T_{\max}(n) \le c_2\, g(n)$$

   Thus, $T_{\max}(n) = O(g(n))$.

   Similarly, for all input of size $n \ge n_0$, the best-case running time is bounded below.

$$\forall I \text{ with } |I| = n, \quad 0 \le c_1\, g(n) \le T_{\min},$$

$$\Rightarrow 0 \le c_1\, g(n) \le T_{\min}(n)$$

Thus, $T_{\min}(n) = \Omega(g(n))$.

$\boxed{\Leftarrow}$

Assume that $T_{\max}(n) = O(g(n))$ and $T_{\min}(n) = \Omega(g(n))$, then we have positive constants $c_1, c_2$ and $n_0$ such that for all $n \ge n_0$,

$$0 \le c_1\, g(n) \le T_{\min}(n) \le T_{\max}(n) \le c_2\, g(n),$$

$$\Rightarrow 0 \le c_1\, g(n) \le \min_{|I|=n} T(I) \le \max_{|I|=n} T(I) \le c_2\, g(n),$$

$$\Rightarrow 0 \le c_1\, g(n) \le T(I) \le c_2\, g(n), \quad \forall I \text{ with } |I| = n.$$

Thus, $T(I) = \Theta(g(n))$.

7. Prove that $o(g(n)) \cap \omega(g(n))$ is the empty set.

   **Proof.**

   Assume to the contrary that there existed a $f(n)$ such that

   $$f(n) \in o(g(n)) \quad \text{and} \quad f(n) \in \omega(g(n)).$$

   Then, for any $c > 0$, there would exist positive constants $n_1$ and $n_2$ such that

   $$\begin{aligned} &0 \le c\, g(n) < f(n), \quad \forall n \ge n_1, \quad \text{and} \\ &0 \le f(n) < c\, g(n), \quad \forall n \ge n_2. \end{aligned}$$

   Then we have the contradiction:

   $$c\, g(n) < f(n) < c\, g(n), \quad \forall n \ge \max(n_1, n_2).$$

   Therefore, $o(g(n)) \cap \omega(g(n)) = \varnothing$.

8. We can extend out notation to the case of two parameters $n$ and $m$ that can go to infinity independently at different rates. For a given function $g(n,m)$, we denote by $O(g(n,m))$ the set of functions.

   $$\begin{aligned} O(g(n,m)) = \{f(n,m) \ &: \exists \text{ positive constants } c, n_0 \text{ and } m_0 \text{ such that} \\ &\quad 0 \le f(n,m) \le c\, g(n,m), \ \text{ for all } n \ge n_0 \text{ or } m \ge m_0\}, \end{aligned}$$

   Give corresponding definitions for $\Omega(g(n,m))$ and $\Theta(g(n,m))$.

**Answer.** For any function $g : \mathbb{N}^2 \to \mathbb{R}$,

$$\begin{aligned}
\Omega(g(n,m)) = \{f(n,m) \ :\ & \exists \text{ positive constants } c, n_0 \text{ and } m_0 \text{ such that} \\
& 0 \leq c\, g(n,m) \leq f(n,m), \quad \text{for all } n \geq n_0 \text{ or } m \geq m_0\}, \\
\Theta(g(n,m)) = \{f(n,m) \ :\ & \exists \text{ positive constants } c_1, c_2, n_0 \text{ and } m_0 \text{ such that} \\
& 0 \leq c_1\, g(n,m) \leq f(n,m) \leq c_2\, g(n,m), \\
& \text{for all } n \geq n_0 \text{ or } m \geq m_0\}.
\end{aligned}$$