بکارگیری تعاریف مرتبهٔ مجانبی در عمل

- توجه کنید که در تعاریف مرتبهٔ مجانبی قیدی مبنی بر Best Case یا Worst Case در تعاریف وجود ندارد.

- در عمل اگر زمان اجرای یک الگوریتم در بهترین حالت برابر $B(n)$ باشد می‌توانید درباره اینکه

یا $B(n) = O(g(n))$

یا $B(n) = \Omega(g(n))$

$B(n) = \Theta(g(n))$

صحبت کنید.

- بطور مشابه اگر زمان اجرای یک الگوریتم در بدترین حالت برابر $W(n)$ باشد می‌توانید درباره اینکه

$W(n) = O(g(n))$ یا

$W(n) = \Omega(g(n))$ یا

$W(n) = \Theta(g(n))$ صحبت کنید.

- از نظر منطق کاربردی، اصولاً تحلیل زمان اجرای یک الگوریتم در بدترین حالت برحسب Ω غیرمعقول است.

- بهمین دلیل معمولاً آنچه برای ما در تحلیل الگوریتم‌ها جالب است تحلیل بدترین حالت برحسب مرتبهٔ O یا Θ است.

- به عبارت دیگر در راستای نشان دادن اینکه یک الگوریتم کارآمد است کافی است نشان دهیم در بدترین حالت از مرتبه $O(g(n))$ است.

- بطور مشابه از نظر منطق کاربردی، تحلیل زمان اجرای یک الگوریتم در بهترین حالت برحسب O، غیرمعقول است.

- معمولاً در راستای نشان دادن ناکارایی یک الگوریتم کافی است نشان دهیم در بهترین حالت از مرتبه $\Omega(g(n))$ است.

مثلاً $B(n) = \Omega(n^2)$.

$$\sum_{i=1}^{n} i^K = \underbrace{1^K + 2^K + \cdots}_{\times} + \underbrace{\cdots + n^K}_{\checkmark} > \lceil n/2 \rceil^K + (\lceil n/2 \rceil + 1)^K + \cdots + n^K$$

$$> \lceil n/2 \rceil^K + \lceil n/2 \rceil^K + \cdots + \lceil n/2 \rceil^K$$

$$> (n/2)\,(n/2)^K$$

$$= \frac{n^{K+1}}{2^{K+1}}$$

$$> \frac{1}{2^{K+1}}\, n^{K+1}$$

$$c = 1/2^{K+1}$$

بکارگیری آنالیز مجانبی در عمل

① برای اثبات $f(n) = \Omega(g(n))$ کافی است ثابت کنید $g(n) = O(f(n))$

② برای اثبات $f(n) = \Theta(g(n))$ کافی است ثابت کنید

$$f(n) = O(g(n)) \quad \text{AND} \quad g(n) = O(f(n))$$

③ مرتبهٔ توابع چند جمله ای. هر تابع چند جمله ای از مرتبهٔ k مثل

$$f(n) = a_k n^k + a_{k-1} n^{k-1} + \cdots + a_1 n + a_0$$

از مرتبهٔ $\Theta(n^k)$ است اگر $a_k > 0$.

4 به ازای اعداد حقیقی $0 \leq a \leq b$ داریم $O(n^a) \subseteq O(n^b)$

به بیان ساده اگر دو الگوریتم یکی از مرتبهٔ زمانی $O(n^a)$ و دیگری از

مرتبهٔ $O(n^b)$ داشته باشیم الگوریتمی کارآمدتر است که درجهٔ

چندجمله‌ای آن کوچک‌تر باشد. $O(n^{1.2}) \subseteq O(n^2)$

5 به ازای اعداد حقیقی $1 < a \leq b$ داریم $O(a^n) \subseteq O(b^n)$

مثال. $O(2^n) \subseteq O(3^n)$

⑥ به ازای هر دو عدد حقیقی دلخواه $a>1$ و $b>1$ داریم

$$\log_a n = \Theta(\log_b n)$$

بهمین دلیل در مبحث آنالیز الگوریتم‌ها، تمام الگوریتم‌های مرتبه زمان اجرای آنها لگاریتمی است مستقل از اینکه پایه لگاریتم چه عددی است در یک خانواده قرار می‌گیرند

و معمولاً پایه را نمی‌نویسیم

$$T(n) = O(\log n)$$

$$\lg n := \log_2 n$$

7 تعریف مرتبه $o(g(n))$

$$o(g(n)) = \{ f : \mathbb{Z}^{\geq 0} \to \mathbb{R} \mid \forall c > 0, \ \exists n_0 \in \mathbb{Z}^{\geq 0}, \ \forall n \geq n_0, \ 0 \leq f(n) \leq c \cdot g(n) \}$$

از تعریف $o(g(n))$ بلافاصله نتیجه می‌شود $o(g(n)) \subseteq O(g(n))$

در نتیجه اگر بتوانیم نشان دهیم $f(n) \in o(g(n))$ آنگاه فوراً داریم

$f(n) \in O(g(n))$

8) با توجه به تعریف مرتبه $o(g(n))$

$$o(g(n)) = \{ f : \mathbb{Z}^{\geq 0} \to \mathbb{R} \mid \forall \varepsilon > 0, \exists n_0 \in \mathbb{Z}^{\geq 0}, \forall n \geq n_0, \ 0 \leq f(n) \leq \varepsilon \cdot g(n) \}$$

اگر فرض کنیم $g(n) \neq 0$ آنگاه تعریف بالا معادل است با: $\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$

نتیجه: اگر $\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$ آنگاه $f(n) = O(g(n))$

۹) به ازای هر $k \geq 0$ و هر $a > 1$ داریم

$$n^k = o(a^n) \Rightarrow n^k = O(a^n)$$

به بیان ساده بر مبنای آنالیز مجانبی سرعت رشد هر تابع چندجمله‌ای کمتر از هر تابع نمایی است

مثال: $n^{100} = O(2^n)$, $n^{100} = O(1.1^n)$

۱۰) به ازای هر $k > 0$ و هر $a > 1$ داریم

$$\log_a n = o(n^k) \Rightarrow \log_a n = O(n^k)$$

مثال: $\lg n = O(n)$, $\lg n = O(\sqrt[100]{n})$

11

لم. اگر فرض کنیم

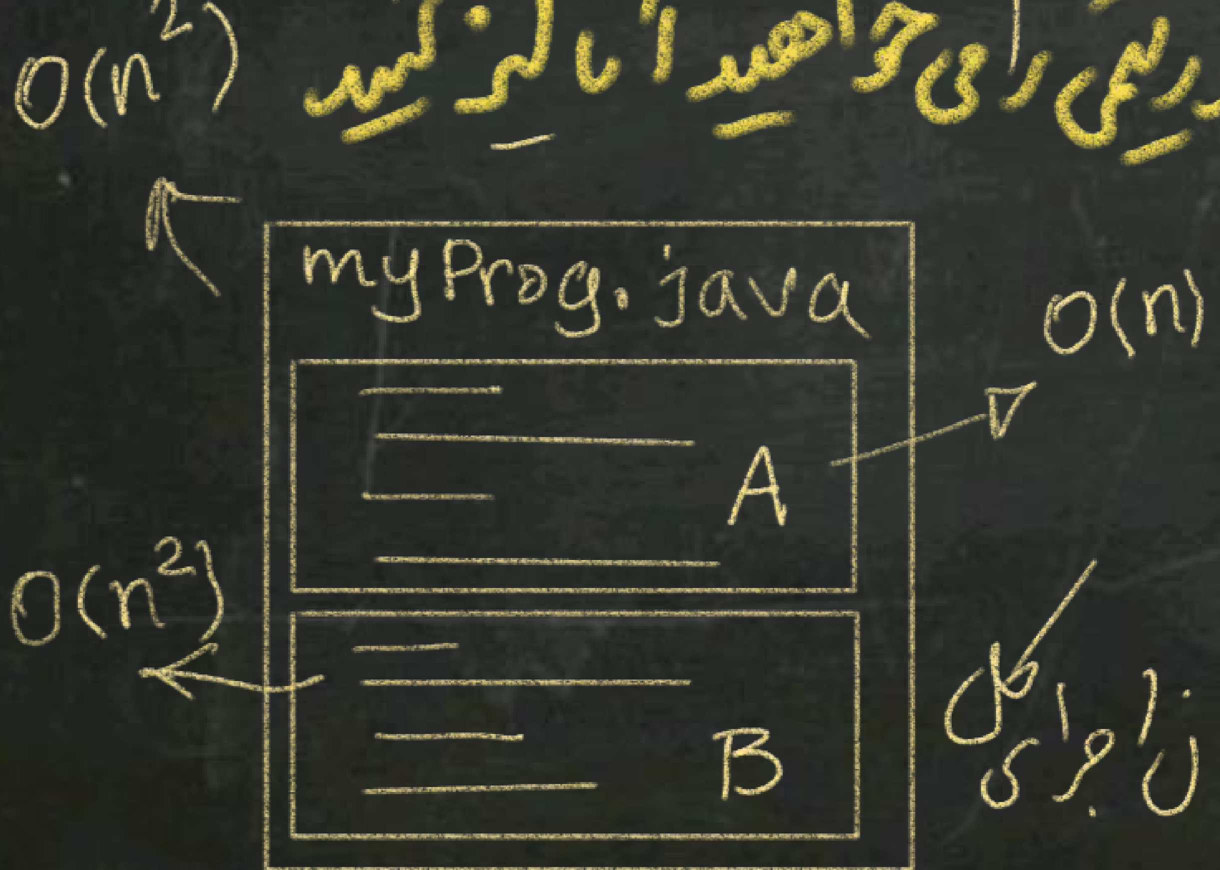
$$f_1(n) = O(g_1(n))$$

$$f_2(n) = O(g_2(n))$$

آنگاه

$$\Rightarrow \quad f_1(n) + f_2(n) = O(\max(g_1(n), g_2(n)))$$

<ins>از جنبه کاربردی</ins>. فرض کنید زمان اجرای الگوریتمی را می‌خواهید آنالیز کنید



که شامل دو پروسه متوالی A و B است

اگر زمان اجرای A از مرتبهٔ $O(g_1(n))$ و

زمان اجرای B از مرتبهٔ $O(g_2(n))$ باشد آنگاه زمان اجرای کل

از مرتبهٔ $O(\max(g_1(n), g_2(n)))$ است.

```
for i = 1 to n
    myFun(i)
```

$\rightarrow O(i^k)$

$T(n) = ?$

$$\sum_{i=1}^{n} i^k = \Theta(n^{k+1})$$

12

for i = 1 to n

myFun(i) → $O(\lg i)$

$T(n) = ?$

$$\sum_{i=1}^{n} \lg i = \lg(n!) = \Theta(n \cdot \lg n)$$

13

$$n! = O(n^n)$$

$$n! \neq \Theta(n^n)$$

But!

$$\log(n!) = \Theta(n \lg n)$$

15 لم. به‌ازای هر $a>1$ ،

$$a^n = O(n!)$$

16 if $f(n) \equiv c$, (c is a constant) $\Longrightarrow f(n) = O(1)$

اما هر تابع از $O(1)$ لزوماً تابع ثابت نیست. مثال:

$$c - \frac{a}{n} \in O(1)$$

17 if $f(n) = O(1)$ and $f(n)$ is increasing $\Rightarrow f(n) = \Theta(1)$