بسم الله الرحمن الرحیم

ساختمان‌های داده

جلسه ۴

مجتبی خلیلی
دانشکده برق و کامپیوتر
دانشگاه صنعتی اصفهان

تحلیل مجانبی

- زمان اجرای دقیق یک الگوریتم معمولا قابل پیش بینی نیست.
- تحلیل مجانبی برای ورودی با اندازه بزرگ
  - تخمینی از مدت زمان اجرای برنامه
  - تخمینی از اندازه ورودی
  - مقایسه کارایی الگوریتم‌های مختلف
  - تمرکز بر بخشی از کد یا الگوریتم که بیشترین تکرار را دارد.
- در این درس ما اغلب رفتار مجانبی الگوریتم‌ها را در نظر می‌گیریم.

نمادهای مجانبی

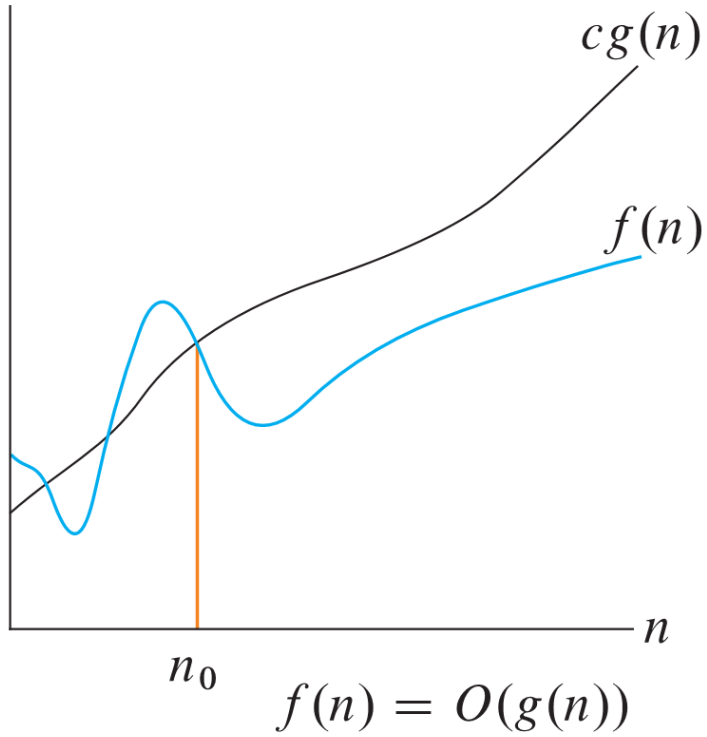
- نماد Big-O.
  - در این کلاس ما اغلب از این نماد استفاده می‌کنیم.

نمادهای مجانبی

- نماد Big-O.

زمان اجرای مجانبی	تعداد عملیات
$O(n^2)$	$150 \cdot n^2 + 1000$
$O(n^2)$	$0.0052 \cdot n^2 - .5\, n - 12.7$
$O(n^{1.5})$	$500 \cdot n^{1.5} - 2^{2000}\sqrt{n}$
$O(n^2 \log(n))$	$41 \cdot n^2 \log(n) + 0.5$

Big-O



$f(n) = O(g(n))$

Def.

We write $f(n) = O\big(g(n)\big)$ if there are positive constants n_0 and c such that for all $n \geq n_0$:

$$f(n) \leq c \cdot g(n)$$

Big-O

Def.

We write $f(n) = O\big(g(n)\big)$ if there are positive constants n_0 and c such that for all $n \geq n_0$:

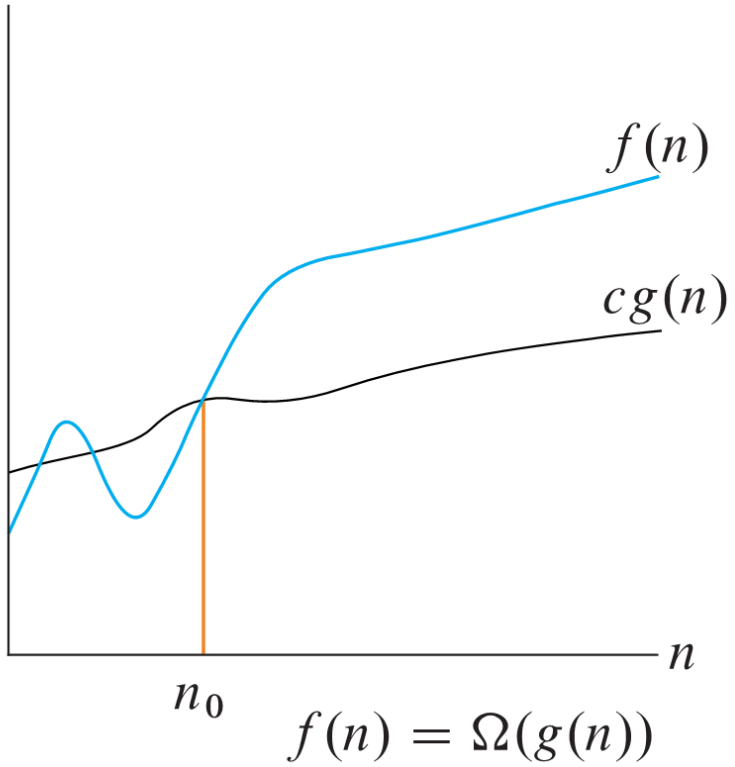
$$f(n) \leq c \cdot g(n)$$

- مثال: $3n^3 + 20n^2 + 5$

$3n^3 + 20n^2 + 5 \leq c \cdot n^3$ for $n \geq n_0$ → $c = 5, n_0 = 20$

$$3n^3 + 20n^2 + 5 = O(n^3)$$

Big-Ω



Def.

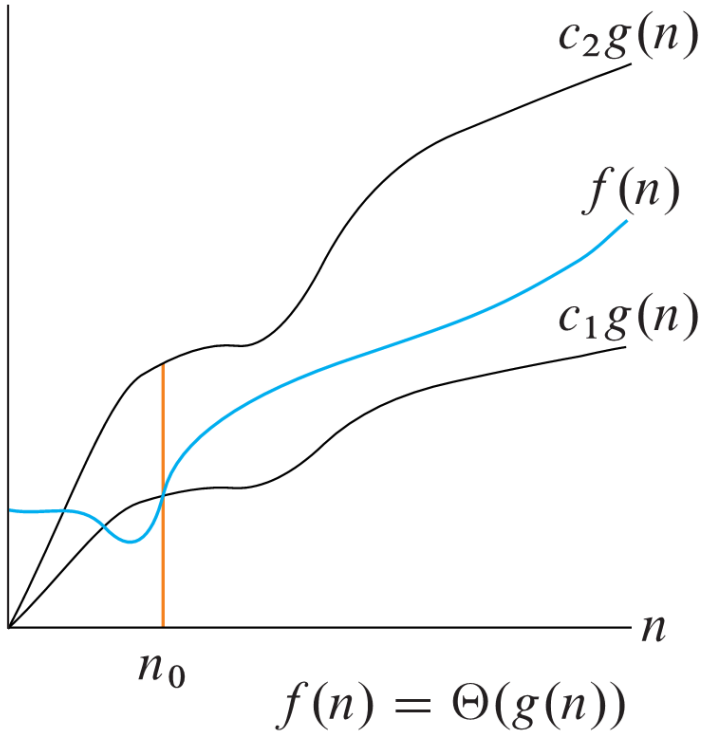
We write $f(n) = \Omega\big(g(n)\big)$ if there are positive constants n_0 and c such that for all $n \geq n_0$:

$$f(n) \geq c \cdot g(n)$$

- مثال:

$$4n^2 + 100n + 500 = \Omega(n^2)$$

Big-Θ



Def.

We write $f(n) = \Theta\big(g(n)\big)$ if there are positive constants n_0, c_1, and c_2 such that for all $n \geq n_0$:

$$c_1 \cdot g(n) \leq f(n) \leq c_2 \cdot g(n)$$

- مثال:

$$2n^2 - 10n + 8 = \Theta(n^2)$$

Big-Θ

Theorem 3.1

For any two functions $f(n)$ and $g(n)$, we have $f(n) = \Theta(g(n))$ if and only if $f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$. ■

برخی روابط

- $f(n) = O(g(n)) \Leftrightarrow \lim_{n\to\infty} \frac{f(n)}{g(n)} = 0 \text{ or } c$
- $f(n) = \Omega(g(n)) \Leftrightarrow \lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty \text{ or } c$
- $f(n) = \Theta(g(n)) \Leftrightarrow \lim_{n\to\infty} \frac{f(n)}{g(n)} = c \neq 0$

خواص

- برخی از خواص نمادهای گفته شده و روابط بین آنها

- $f(n) = \boldsymbol{O}(g(n)) \Rightarrow g(n) = \boldsymbol{\Omega}(f(n))$
- $f(n) = \boldsymbol{\Omega}(g(n)) \Rightarrow g(n) = \boldsymbol{O}(f(n))$
- $f(n) = \boldsymbol{\Theta}(g(n)) \Rightarrow g(n) = \boldsymbol{\Theta}(f(n))$

مثال

- با توجه به تابع زیر، کدام گزینه درست است؟

$$T(n) = \frac{1}{2}n^2 + 3n.$$

a) $T(n) = O(n)$

b) $T(n) = \Omega(n)$

c) $T(n) = \Theta(n^2)$

d) $T(n) = O(n^3)$

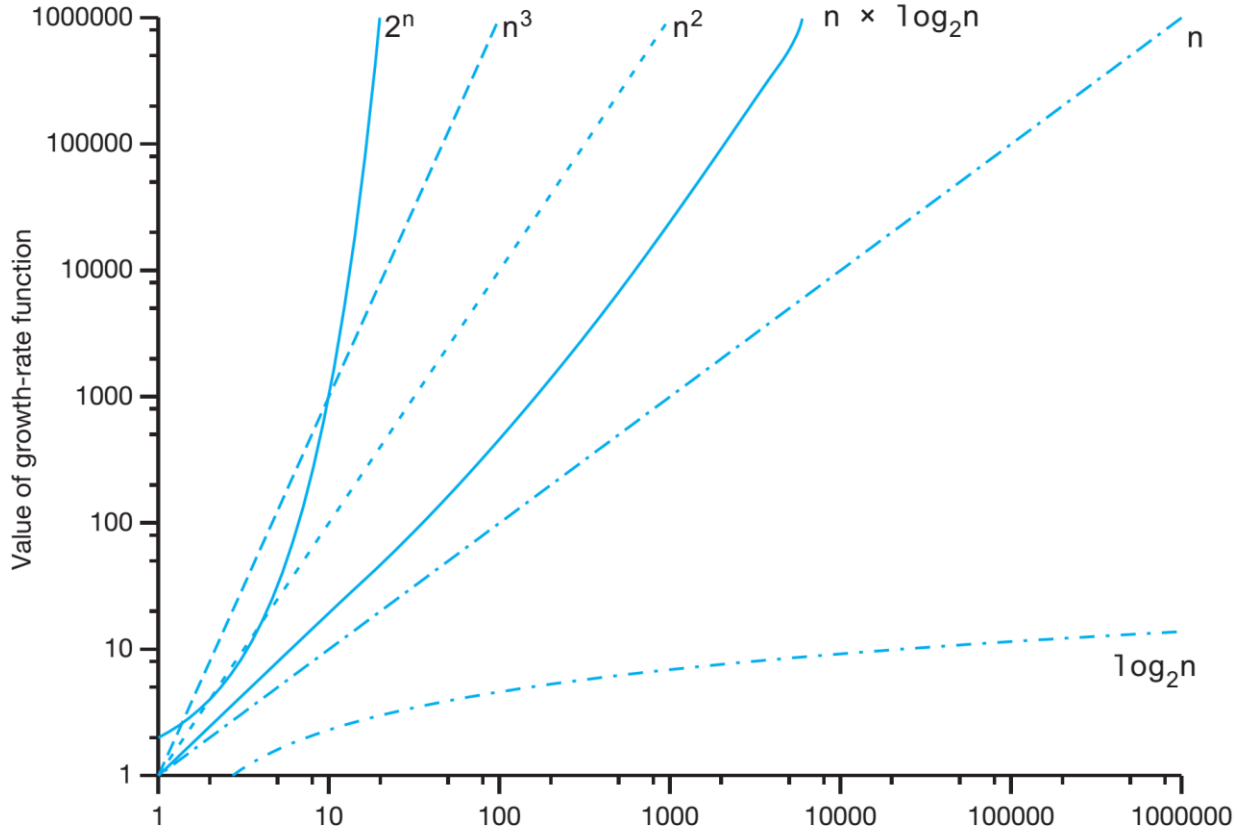
سوال

- آیا درستند؟

- $10^{100} = O(1)$
- $n^2 = O(n^5)$
- $n^2 = \Theta(n^5)$
- $33n^2 + 41n + \log(n) + 14 = O(n^2 + n + 1)$
- $\log(n^2) = O(\log(n))$

برخی نرخ‌های رشد

- $\Theta(1)$: ثابت
- $\Theta(\log n)$: لگاریتمی
- $\Theta(n)$: خطی
- $\Theta(n \log n)$: خطی-لگاریتمی
- $\Theta(n^2)$: مربعی
- $\Theta(n^3)$: مکعبی
- $\Theta(2^n)$: نمایی



زمان مصرفی

- فرض کنید هر عمل ۱ نانو ثانیه

n	$\log n$	n	$n \log n$	n^2	n^3	2^n
8	3	8	24	64	512	256
16	4	16	64	256	4,096	65,536
32	5	32	160	1,024	32,768	4,294,967,296
64	6	64	384	4,096	262,144	1.84×10^{19}
128	7	128	896	16,384	2,097,152	3.40×10^{38}
256	8	256	2,048	65,536	16,777,216	1.15×10^{77}
512	9	512	4,608	262,144	134,217,728	1.34×10^{154}

تحلیل مرتب‌سازی درجی

IUT-ECE

```
INSERTION-SORT(A, n)                                          cost   times
1  for i = 2 to n                                             c1     n
2      key = A[i]                                             c2     n − 1
3      // Insert A[i] into the sorted subarray A[1 : i − 1].  0      n − 1
4      j = i − 1                                              c4     n − 1
5      while j > 0 and A[j] > key                             c5     ∑_{i=2}^{n} t_i
6          A[j + 1] = A[j]                                    c6     ∑_{i=2}^{n} (t_i − 1)
7          j = j − 1                                          c7     ∑_{i=2}^{n} (t_i − 1)
8      A[j + 1] = key                                         c8     n − 1
```

- تحلیل بدترین حالت:

$$T(n) = c_1 n + c_2(n-1) + c_4(n-1) + c_5\left(\frac{n(n+1)}{2} - 1\right) + c_6\left(\frac{n(n-1)}{2}\right) + c_7\left(\frac{n(n-1)}{2}\right) + c_8(n-1)$$

$$= \left(\frac{c_5}{2} + \frac{c_6}{2} + \frac{c_7}{2}\right)n^2 + \left(c_1 + c_2 + c_4 + \frac{c_5}{2} - \frac{c_6}{2} - \frac{c_7}{2} + c_8\right)n - (c_2 + c_4 + c_5 + c_8)\,.$$

تابع مربعی از n

$$T(n) = an^2 + bn + c$$

تحلیل مرتب‌سازی درجی

IUT-ECE

- تحلیل بدترین حالت:

```
INSERTION-SORT(A, n)
1  for i = 2 to n
2      key = A[i]
3      // Insert A[i] into the sorted subarray A[1 : i − 1].
4      j = i − 1
5      while j > 0 and A[j] > key
6          A[j + 1] = A[j]
7          j = j − 1
8      A[j + 1] = key
```

$$T(n) = O(n^2)$$

چند نکته