مرور فصل ۳

- در این فصل راهکارهایی برای توصیف رفتار توابع پیچیدگی (بخصوص در هنگامی که اندازه مسئله بسیار بزرگ است) را ارائه خواهیم کرد. مطالعه رفتار تابع بوسیله مجانب ها (asymptotic) انجام می‌شود.
- در ریاضـــیات، **مجانب** (Asymptotic)به بررســـی **رفتار یک تابع هنگامی که ورودی آن به بینهایت نزدیک می‌شود** می‌پردازد. **تحلیل مجانبی** روشی برای **توصیف رفتار الگوریتم‌ها هنگامی که اندازه ورودی بسیار بزرگ می‌شود** است.
  - توصیف رشد (growth) توابع
  - تمرکز بر روی آنچه مهم است با حذف عبارات مرتبه پایین و عوامل ثابت.
  - نحوه نشان دادن زمان‌های اجرای الگوریتم‌ها.
  - روشی برای مقایسه اندازه توابع:

$$O \approx \leq$$
$$\Omega \approx \geq$$
$$\Theta \approx =$$
$$o \approx <$$
$$\omega \approx >$$

تـــابع پیچیـــدگی یـــا تـــابع زمـــان اجـــراء، بـه تـابعی گفتـه شـده برحسـب انـدازه مسـئله، زمـان اجـرای مسـئله (تعداد دستورات لازم) را مشخص می‌کند. مانند توابع زیر

$\frac{n(n-1)}{2}$ $\qquad$ n^3 $\qquad$ n^2

تابع پیچیدگی همواره **مثبت** و **غیرنزولی** است. با توجه به این نکته آیا توابع ریاضی زیر می‌توانند یک تابع پیچیدگی باشند؟

1) $5n^2 - 100n$ $\qquad$ 2) $3n+7$ $\qquad$ 3) $30 - n^3$

پاسخ

۱) برای $n>20$ تابع پیچیدگی است زیرا اگر n کمتر باشد تابع منفی خواهد بود

۲) درست است

۳) چون تابع نزولی است پس تابع پیچیدگی نیست

O-notation، Ω - notation و Θ – notation

O-notation

یک کران بالا برای رفتار مجانبی یک تابع مشخص می‌کند: می‌گوید که یک تابع سریعتر از یک نرخ معین رشد نمی‌کند. این نرخ بر اساس جمله با بالاترین مرتبه است.

۱. مثال عملی: کدام جمله واقعاً اهمیت دارد؟

فرض کنید تابع پیچیدگی یک الگوریتم $f(n) = n^2 + 100n + 5000$ باشد. این تابع می‌تواند نشان‌دهنده تعداد کل عملیات‌ها (مثلاً مقایسه‌ها یا جمع‌ها) باشد که الگوریتم برای ورودی با اندازه n انجام می‌دهد.

- n^2 ممکن است از یک حلقه تودرتو بیاید.
- $100n$ ممکن است از یک حلقه ساده دیگر بیاید.
- 5000 ممکن است هزینه‌های ثابت راه‌اندازی (setup cost) باشد.

حالا بیایید ببینیم وقتی n (اندازه ورودی) بزرگ می‌شود، سهم هر یک از این جملات چقدر است.

اندازه ورودی (n)	جمله n^2	جمله $100n$	جمله 5000	$f(n)$ (مجموع)	درصد سهم n^2
۱۰	۱۰۰	۱,۰۰۰	۵,۰۰۰	۶,۱۰۰	۱.۶٪
۱۰۰	۱۰,۰۰۰	۱۰,۰۰۰	۵,۰۰۰	۲۵,۰۰۰	۴۰٪
۱,۰۰۰	۱,۰۰۰,۰۰۰	۱۰۰,۰۰۰	۵,۰۰۰	۱,۱۰۵,۰۰۰	۹۰.۵٪
۱۰,۰۰۰	۱۰۰,۰۰۰,۰۰۰	۱,۰۰۰,۰۰۰	۵,۰۰۰	۱۰۱,۰۰۵,۰۰۰	**۹۹.۰٪**
۱۰۰,۰۰۰	۱۰,۰۰۰,۰۰۰,۰۰۰	۱۰,۰۰۰,۰۰۰	۵,۰۰۰	۱۰,۰۱۰,۰۰۵,۰۰۰	**۹۹.۹٪**

همانطور که می‌بینید، وقتی n کوچک است (مثل ۱۰)، جمله ثابت 5000 و جمله خطی $100n$ بخش بزرگی از کل را تشکیل می دهد اما به محض اینکه n به ۱۰۰۰ یا ۱۰,۰۰۰ می‌رسد، جمله n^2 به تنهایی تقریباً **تمام** زمان اجرا را به خود اختصاص می‌دهد.

این همان چیزی است که در تحلیل مجانبی به دنبال آن هستیم. وقتی $n \to \infty$ (به سمت بی‌نهایت میل می‌کند)، جملات $100n$ و 5000 در مقایسه با n^2 آنقدر ناچیز می‌شوند که می‌توانیم آن‌ها را نادیده بگیریم.

این مشاهده ما را مستقیم به مورد بعدی می‌برد:

برای مثال،

$$f(n) = 7n^3 + 100n^2 - 20n + 6$$

برابر با$O(n^3)$ است، زیرا بالاترین جمله مرتبه $7n^3$ است و بنابراین تابع سریعتر از n^3 رشد نمی‌کند.

تابع $f(n)$ همچنین $O(n^5)$،$O(n^6)$ و $O(n^c)$ برای هر ثابت $c \geq 3$ است.

(امگا بزرگ) notation-Ω

یک کران پایین برای رفتار مجانبی یک تابع را مشخص می‌کند: می‌گوید که یک تابع حداقل به اندازه یک نرخ معین رشد می‌کند. در این جا نیز جمله با بالاترین مرتبه در نظر گرفته می‌شود.

برای مثال،

$$f(n) = 7n^3 + 100n^2 - 20n + 6$$

برابر با$\Omega(n^3)$ است، زیرا بالاترین جمله مرتبه، n^3 ، حداقل به اندازه n^3 رشد می‌کند.

تابع $f(n)$ همچنین $\Omega(n^2)$، $\Omega(n)$ و $\Omega(n^c)$ برای هر ثابت $c \leq 3$ است.

notation-Θ

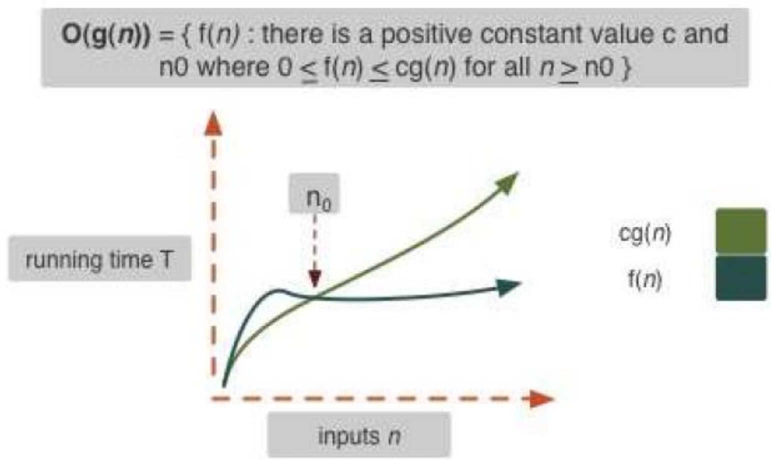
بطور همزمان یک کران بالا و پایین برای رفتار مجانبی یک تابع را مشخص می‌کند. مشخص کننده یک حد و مرز نزدیک از رفتار مجانبی سیستم است. این نماد می‌گوید که یک تابع دقیقاً با یک نرخ معین رشد می‌کند، و مانند نمادهای قبلی بر اساس بالاترین جمله مرتبه خواهد بود.

اگر $O(f(n))$ و هم$\Omega(f(n))$ تابع یکسانی باشد، آنگاه آن تابع، $\Theta(f(n))$ نیز خواهد بود.

بیان فرمول برای مجانب‌ها

نماد O بزرگ

- در صورتی که تابع f(n) و g(n) داده شده باشد، می گوییم که f(n) از مرتبه O(g(n)) است($f(n) = O(g(n))$) اگر ثابت های مثبت c و n_0ای وجود داشته باشد به صورتی که:
- $f(n) \leq cg(n)$ for $n>n_0$



$2n^2 = O(n^3)$

$2n^2 \leq cn^3$

$(c = 1, n_0 = 2)$

$g(n)$ یک کران بالای مجانبی برای $f(n)$ است. اگر $f(n) \in O(g(n))$،می‌نویسیم

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

(به زودی دقیقاً توضیح خواهیم داد که این علامت تساوی به معنی تعلق است).

مثال : چند نمونه از توابع که دارای پیچیدگی $O(n^2)$ هستند

n^2

$n^2 + n$

$n^2 + 1000n$

$1000n^2 + 1000n$

n

$n/1000$

$n^{1.99999}$

$n^2/ \lg\lg\lg n$

مثال) نشان دهید تابع پیچیدگی زمانی *2n-9* از مرتبه $O(n^2)$ است؟

پاسخ : این کار را در دو مرحله انجام می‌دهیم نخست پیدا کردن n_0 و سپس پیدا یک ثابت C بطوریکه که برای $n \geq n_0$ داشته باشیم $2n - 9 \leq Cn^2$

ما می‌دانیم که تابع *2n-9* باید مثبت باشد بنابراین کمترین مقدار n_0 که می‌توانیم داشته باشیم حداقل *5* است اگر فرض کنیم $n_0=5$ است آن‌گاه با توجه به این که $C \geq \frac{2n-9}{n^2}$ پس می‌توان نتیجه گرفت که C باید بزرگتر از $\frac{1}{25}$ باشد

تمرین) نشان دهید تابع پیچیدگی زمانی $13n^2 + 7n$ از مرتبه $O(n^2)$ است؟

پاسخ : ما می‌توانیم مقدار $n_0=1$ در نظر بگیریم همچنین $13 + \frac{7}{n} \leq C$ $\rightarrow$ $13\, n^2 + 7n \leq Cn^2$ بنابراین مقدار C می‌تواند بزرگتر و مساوی ۲۰ باشد.

تمرین) آیا تابع پیچیدگی زمانی $3n^3 + 7n^2$ از مرتبه $O(n^2)$ است؟

پاسخ)

$$3n^3 + 7n^2 \leq Cn^2$$

$$3n + 7 \leq C$$

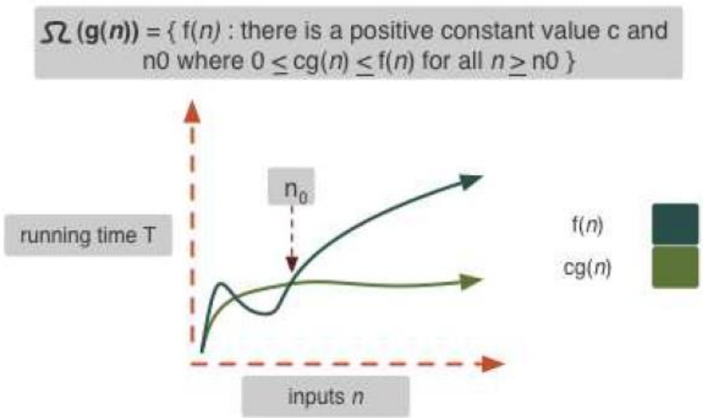
بنابراین ما نمی‌توانیم یک n_0 را طوری پیدا کنیم که برای nهای بزرگتر از n_0 رابطه فوق برقرار باشد.

با توجه به این که مقدار n می‌تواند تا بی‌نهایت ادامه پیدا کند بنابراین ما نمی‌توانیم مقدار ثابتی مانند C پیدا کنیم که رابطه بالا برای آن بر قرار باشد.

نماد امگا (Ω)

در صورتی که تابع $f(n)$ و $g(n)$ داده شده باشد، می گوییم که $f(n)$ از مرتبه $\Omega(g(n))$ است($f(n) = \Omega(g(n))$) اگر ثابت های مثبت c و n_0ای وجود داشته باشد به صورتی که:

$cg(n) \leq f(n)$ for $n>n_0$



$\sqrt{n} = \Omega(\log_2 n)$

$C(\log_2 n) < \sqrt{n}$

$(c = 1, n_0 = 16)$

$g(n)$ یک کران پایین مجانبی برای $f(n)$ است

Example

$\sqrt{n} = \Omega(\lg n)$, with $c = 1$ and $n_0 = 16$.

Examples of functions in $\Omega(n^2)$:

n^2
$n^2 + n$
$n^2 - n$
$1000n^2 + 1000n$
$1000n^2 - 1000n$
Also,
n^3
$n^{2.00001}$
$n^2 \lg\lg\lg n$
2^{2^n}

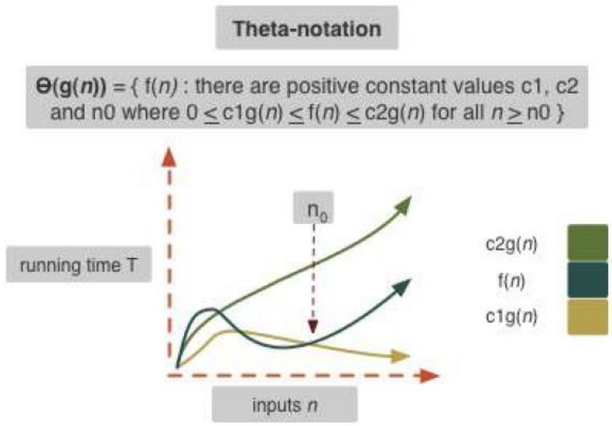
مثال) آیا تابع پیچیدگی زمانی $5n^2 - 11n^2$ از مرتبه Ω (n) است؟

پاسخ) اگر $n_0=1$ باشد، آن‌گاه تابع منفی خواهد شد و این ممکن نیست، بنابراین کوچکترین n_0 که می‌توان در نظر گرفت ۳ است. اگر $n_0=3$ در نظر بگیریم با توجه به این که $C \leq 5n - 11$ باشد پس می‌توان گفت که ثابت C باید کوچکتر یا مساوی از ۴ باشد.

نماد تتا (Θ)

در صورتی که تابع f(n) و g(n) داده شده باشد، می گوییم که f(n) از مرتبه$\Theta(g(n))$است($f(n) = \Theta(g(n))$) اگر ثابت های مثبت c_1 و c_2 و n_0ای وجود داشته باشد به صورتی که:

$c_1g(n) \leq f(n) \leq c_2g(n)$ for $n>n_0$



به عبارت دیگر $\Theta(n)$ مجموعه‌ای از توابع پیچیدگی مانند f(n) است که دارای شرایط زیر باشند:

$$\Theta(g(n)) = \{f(n) |\ \exists\ C_1, C_2, n_0 > 0\ , C_1g(n) \leq\ f(n) \leq C_2g(n)\ \ \forall n \geq n_0\}$$

$$n^2 + 5n + 7 = \Theta(n^2)$$

$C_1n^2 \leq\ n^2+5n+7 \leq C_2n^2$

$C_1n^2 \leq\ n^2+5n+7$

$C_1 = 1\ ,\ n_0=1$

$n^2+5n+7 \leq C_2n^2$

$C_2= 13$

قضیه

$f(n) = \Theta(g(n))$ اگر و فقط اگر $f = O(g(n))$ و $f = \Omega(g(n))$.

ضرایب پیشرو و عبارات مرتبه پایین مهم نیستند.

قضیه: اگر $f(n) = a_mn^m + \ldots + a_1n + n_0$ باشد آنگاه $f(n) = O(n^m)$ خواهد بود.

قضیه: اگر $f(n) = a_mn^m + \ldots + a_1n + n_0$ باشد آنگاه $f(n) = \Omega(n^m)$ خواهد بود.

قضیه: اگر $f(n) = a_mn^m + \ldots + a_1n + n_0$ باشد آنگاه $f(n) = \Theta(n^m)$ خواهد بود.

بررسی خواص هم ارزی روی نمادهای مذکور

خاصیت بازتابی :

$$f(n) \in \Theta(f(n))$$

$$C_1 f(n) \le f(n) \le C_2 f(n) \rightarrow C_1 \le 1 \le C_2$$

بنابراین تعیین C_1 و C_2 مستقل از n_0 است و می توانیم مثلا $C_1=1$ و C_2 را هر عدد بزرگتر یا مساوی ۱ قرار دهیم.

نکته:

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

$$f(n) \in \Omega(f(n))$$

ب) $f(n) \in O(f(n))$

$f(n) \le c f(n) \Rightarrow 1 \le c$ ✓✓

ج) $f(n) \in \Omega(f(n))$

$c f(n) \le f(n) \Rightarrow c \le 1$ ✓✓

خاصیت تقارن

$$f(n) \in \theta(g(n)) \iff g(n) \in \theta(f(n))$$

برای اثبات طرفین، از فرض و طرف دیگر را حکم در نظر می گیریم و از فرض به حکم می رسیم

پس فرض می کنیم طرف اول برقرار است پس c_1 ، c_2 ، n_0 وجود دارد به ازای $n \ge n_0$

فرض $\Leftarrow$ $c_1 g(n) \le f(n) \le c_2 g(n)$

حال بایستی ثابت کنیم به ازای $n \ge n_0'$ ، c_1' ، c_2' ، n_0' وجود خواهد داشت

حکم $\Leftarrow$ $c_1' f(n) \le g(n) \le c_2' f(n)$

$1 \Rightarrow g(n) \le \frac{1}{c_1} f(n) \Rightarrow 1/c_1 = c_2'$

$2 \Rightarrow \frac{1}{c_2} f(n) \le g(n) \Rightarrow 1/c_2 = c_1'$

تمرین) آیا O و Ω خاصیت تقارنی دارند؟

$n \in O(n^2)$ ولی $n^2 \notin O(n)$

$n^2 \in \Omega(n)$ ولی $n \notin \Omega(n^2)$

پس می‌توانیم برای رد کردن آن مثال نقض بیاوریم

پس Θ خاصیت تقارن را دارد ولی O و Ω خاصیت تقارن را ندارند

خاصیت تعدی

$$f(n) \in \theta(g(n))\,, g(n) \in \theta(h(n)) \;\rightarrow\; f(n) \in \theta(h(n))$$

خاصیت تعدی:

$$f(n) \in \theta(g(n))\,, g(n) \in \theta(h(n)) \Longrightarrow f(n) \in \theta(h(n))$$

فرض

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

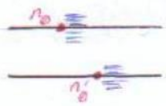
$\forall\, n \geq n_0' : \underbrace{c_1' h(n) \leq g(n)}_{1} \leq \underbrace{\;}_{} \underbrace{g(n) \leq c_2' h(n)}_{2}$

حکم $\forall\, n \geq n_0'' \quad c_1'' h(n) \leq f(n) \leq c_2'' h(n)$

$1 \Rightarrow \times c_1 \Rightarrow c_1 c_1' h(n) \leq c_1 g(n) \;,\; c_1 g(n) \leq f(n) \Rightarrow c_1 c_1' = c_1''$

$2 \Rightarrow \times c_2 \Rightarrow c_2 g(n) \leq c_2 c_2' h(n) \;,\; f(n) \leq c_2 g(n) \Rightarrow c_2 c_2' = c_2''$

فرض



$\Longrightarrow n_0'' = Max\{n_0, n_0'\}$ یا جمع یا ضرب n_0 , n_0' نیز می‌توانند باشند.

اثبات خاصیت تعدی برای O و Ω نیز به راحتی امکان پذیر است.

نکته :

$$\text{if } f(n) \in O(g(n)) \Longleftrightarrow g(n) \in \Omega(f(n))$$

$$\text{if } f(n) \in O(g(n)) \text{ and } f(n) \in \Omega(g(n)) \Longleftrightarrow f(n) \in \Theta(g(n))$$

نمادهای مجانبی و زمان اجرا

باید در اســـتفاده از نماد مجانبی برای مشـــخص کردن زمان اجرا دقت کرد. نماد مجانبی توابع را توصـــیف می‌کند که به نوبه خود زمان‌های اجرا را توصیف می‌کنند. باید دقت کرد که کدام زمان اجرا مشخص شود.

برای مثال، زمان اجرای بدترین حالت برای insertion sort برابر با $O(n^2)$، $\Omega(n^2)$ و $\Theta(n^2)$ اســـت؛ همه صـــحیح هســـتند. ترجیحاً از $\Theta(n^2)$ اســـتفـاده می‌کنیم چون دقیق‌ترین اســـت. زمـان اجرای بهترین حـالـت برای insertion sort برابر بـا $O(n)$، $\Omega(n)$ و $\Theta(n)$ است؛ ترجیحاً $\Theta(n)$.

اما نمی‌توان گفت که زمان اجرای insertion sort برابر با $\Theta(n^2)$ اســت، بدون ذکر **"بدترین حالت"**. حذف این مورد به معنای یک بیان کلی است که همه موارد را پوشش می‌دهد، و insertion sort در همه موارد در زمان $\Theta(n^2)$ اجرا نمی‌شود.

می‌توان این بیان کلی را داشــت که زمان اجرای insertion sort برابر با $O(n^2)$ اســت، یا اینکه $\Omega(n)$ اســت، زیرا این زمان‌های اجرای مجانبی برای همه موارد صحیح هستند.

برای merge sort، زمان اجرای آن در همه موارد برابر با $\Theta(n\lg n)$ است، بنابراین می‌توان از ذکر مورد خاص صرف نظر کرد.

خطای رایج: اشـتباه گرفتن نماد O با نماد Θ با اسـتفاده از نماد Θ برای نشـان دادن یک کران مجانبی تنگ. نماد O فقط یک کران بالای مجانبی می‌دهد.

یک الگوریتم که در زمان $\Theta(n)$ اجرا می‌شود، در زمان $O(n^2)$ نیز اجرا می‌شود.

اگر واقعاً منظور یک کران مجانبی تنگ اســت، باید از نماد Θ اســتفاده کرد. از ســاده‌ترین و دقیق‌ترین نماد مجانبی که قابل اعمال اســت اســتفاده کنید. فرض کنید زمان اجرای یک الگوریتم $\frac{3n^2+20n}{4}$ باشــد. بهترین کار این اســت که بگوییم $\Theta(n^2)$. همچنین می‌توان گفت $O(n^3)$ ، اما این کمتر دقیق است. می‌توان گفت $\Theta(3n^2 + 20n)$ ، اما این ترتیب رشد را مبهم می‌کند.

چند نکته درباره استفاده از نمادهای مجانبی در معادلات

- اگر در معادلات علامت مجانبی یا همان کران اســتفاده شــده باشــد آنگاه علامت مســاوی (=) که در معادلات اســتفاده می‌شود بیشتر به معنی تعلق داشتن است

- نمادهای مجانبی نشان دهنده یک مجموعه از تابع‌ها (تعدادی تابع) هستند. مثلا O(n) نماینده و نشان از توابعی است که پیچیدگی آنها خطی می‌باشد. پس اگر در یک معادله ظاهر شود، توابع خطی زیادی را می‌توان جایگزین آن کرد.

برای مثال $2n^2 + 3n + 1 = 2n^2 + \Theta(n)$ یعنی تابع مانند f(n) وجود دارد بطوری که

$$2n^2 + 3n + 1 = 2n^2 + f(n)$$

حال ما می توانیم f(n) را هر تابع متعلق به O(n) مانند 3n+1 انتخاب کنیم.

مهم نیست که نماد مجانبی (که یک تابع ناشناس است) در سمت چپ چگونه انتخاب شده‌اند، راهی برای انتخاب یک تابع در سمت راست وجود دارد تا معادله معتبر شود. برای مثال:

$$2n^2 + \Theta(n) = \Theta(n^2)$$

این معادله را اینگونه تفسیر کنید که برای همه توابع

$$f(n) \in \Theta(n)$$

، تابعی

$$g(n) \in \Theta(n^2)$$

وجود دارد به طوری که

$$2n^2 + f(n) = g(n)$$

می‌توان به صورت زنجیره‌ای نوشت: $2n^2 + 3n + 1 = 2n^2 + \Theta(n) = \Theta(n^2)$

تفسیر بیشتر این که:

معادله اول: تابعی $f(n) \in \Theta(n)$ وجود دارد به طوری که $2n^2 + 3n + 1 = 2n^2 + f(n)$

معادله دوم: برای همه

$$g(n) \in \Theta(n)$$

(مانند $f(n)$ که برای معتبر کردن معادله اول استفاده شد)، تابعی $h(n) \in \Theta(n^2)$ وجود دارد به طوری که

$$2n^2 + g(n) = h(n)$$