طراحی و تحلیل الگوریتم ها

دکتر امیر لکی زاده

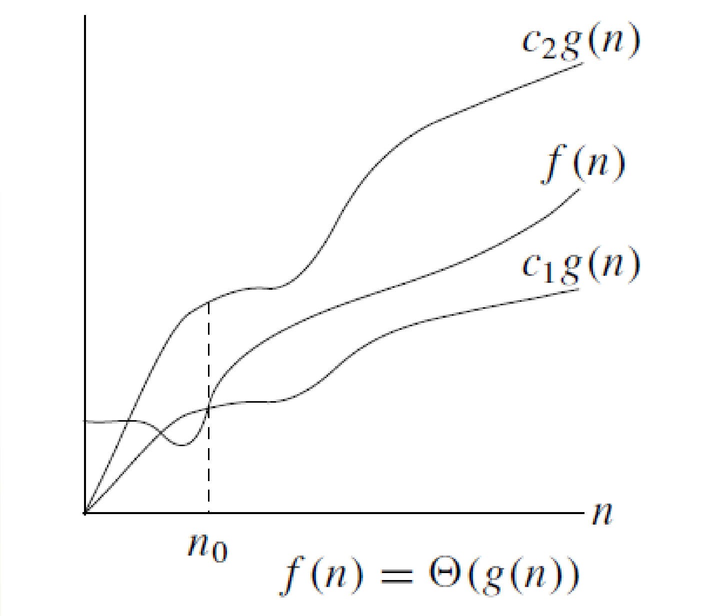
استادیار گروه مهندسی کامپیوتر دانشگاه قم

تحلیل الگوریتم ها

- نماد θ

$f,g: N \rightarrow N$

- $\theta(g(n)) = [f(n) \quad | there\ exist\ positive\ constants\ C_1, C_2, n_0\ such\ that\ \forall n \geq n_o$
$: C_1 g\ (n) \leq f\ (n) \leq C_2\ g\ (n)\ \}$



تحلیل الگوریتم ها

- $f\ (n) \in\ \theta\left(g(n)\right) \rightarrow (f(n) = \theta(g(n))$
- به این معنی است که آهنگ افزایش مقادیر دو تابع f(n) و g(n) به ازای مقادیر به اندازه کافی بزرگ $n \geq n_0$ با هم برابر باشند.
- ثابت کنید $f(n) = \theta(g(n))$

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

$$C_1 n^2 \leq \frac{1}{2}n^2 - 3n \leq C_2 n^2$$

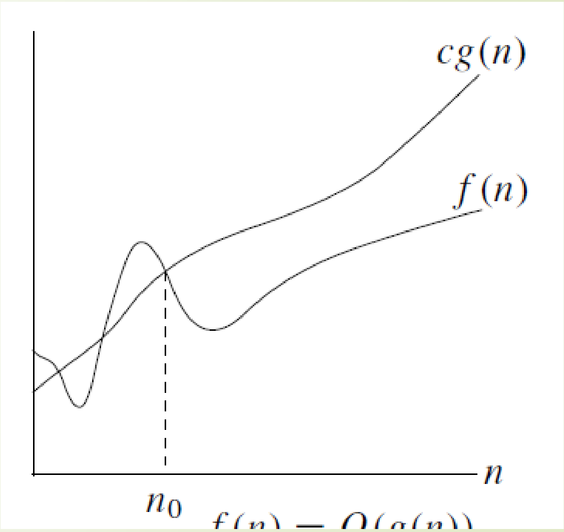
$$C_1 n^2 \leq \frac{1}{2}n^2 - 3n \rightarrow C_1 \leq \frac{1}{2} - \frac{3}{2} : n_0 = 7\ , C_1 = \frac{1}{14}$$

$$\frac{1}{2}n^2 - 3n \leq C_2 n^2 \rightarrow \frac{1}{2} - \frac{3}{n} \leq C_2 : C_2 = \frac{1}{2}$$

تحلیل الگوریتم ها

- ـ نماد O بزرگ: Big_O

- $O\ (g(n)) =\ \{f\ (n)|\ there\ exist\ Positve\ constants\ C, n_0\ such\ that\ \forall\ n \geq n_0 :\ \ f\ (n) \leq c\ g(n)\}$



- $f\ (n) \in O\ (g(n)) \rightarrow f(n) = O\ \big(g(n)\big)$

- g(n) یک کران بالا برای f(n) است و یا آهنگ رشد g(n) بیشتر از f(n) می باشد

تحلیل الگوریتم ها

- $f(n) = 100n + 5 \, , g(n) = n^2$
- $f(n) = O\ (g(n))$

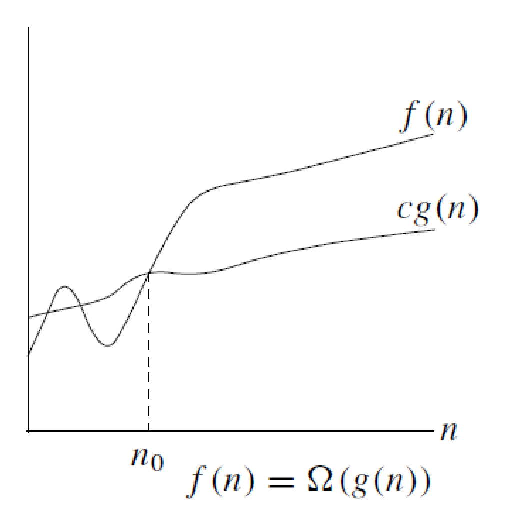
- برای اثبات $Cn^2 \leq 100n + 5$ ارائه یک جفت مقادیر کافی است:$(n_0 = 10^3 ,C = 1)$ یا $(n_0 = 1 ,C = 105)$
- از نماد Big_O برای بیان زمان اجرای الگوریتم در بدترین حالت استفاده می شود.
- اگر T (n) پیچیدگی زمانی یک الگوریتم را نمایش دهد، $T(n) = O(g(n))$ به این معنی است که g(n) کوچکترین کران بالا برای $T(n)$ می باشد یا زمان اجرای الگوریتم در بدترین حالت برابر g(n) می باشد.
- نکته: اگر برای ثابت K داشته باشیم $f(n) = O\ (n)$ گوییم که $f(n)$ محدود چند جمله ای است.

تحلیل الگوریتم ها

- نماد Big – Omega (Ω):

- $\Omega\big(g(n)\big) = \{f(n) \quad | ther\ exist\ positvie\ constant\ C_0\ , n_0\ such\ that\ \forall\ n \geq n_0 \qquad f(n) \geq C\ (g(n)\}$

- $f\ (n) \in \Omega(g(n)) \rightarrow f(n) = \Omega\big(g(n)\big)$



- g(n) یک کران پایین برای f(n) است و یا آهنگ رشد f(n) بیشتر از g(n) می باشد

تحلیل الگوریتم ها

- نماد Big – Omega (Ω):

$$\boldsymbol{f(n) = \Omega\,(g(n))}$$

- $g(n)$ یک کران پایین برای $f(n)$ است (آهنگ رشد $g(n)$ کوچکتر از $f(n)$ می باشد).
- از نماد Big_omega برای بیان زمان اجرای الگوریتم در <u>بهترین حالت</u> استفاده می شود.

- اگر T (n) زمان اجرای یک الگوریتم باشد. آن گاه $T(n) = \Omega(g(n))$، به این معنی است که تابع $g(n)$ بزرگترین کران پایین (زمان اجرای الگوریتم در <u>بهترین حالت</u>) می باشد.

تحلیل الگوریتم ها

- نماد o – Small

- $o\big(g(n)\big) = \{f(n) \quad | for\ any\ positive\ constant\ C, thete\ exits\ n_0 > 0$
  $such\ that\ \forall\ n \geq n_0\ f(n) < cg(n)\}$
- $f(n) = o\big(g(n)\big) \Leftrightarrow f(n) = \ O(g(n)), f(n)\ \neq\ \theta\big(g\ (n)\big)$
- $f(n) = o\ \big(g(n)\big) \Rightarrow f(n) = O\ \big(g(n)\big)$

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

تحلیل الگوریتم ها

- ـ نماد Small – Omega (ω)

- $\omega\left(g(n)\right) = \{f(n) \quad | for\ any\ positive\ constant\ C, thete\ exits\ n_0 > 0$
  $such\ that\ \forall\ n \geq n_0\ f(n) > cg(n)\}$
- $f(n) = \omega\left(g(n)\right) \Leftrightarrow f(n) = \Omega(g(n)), f(n) \neq \theta\left(g\ (n)\right)$
- $f(n) = \omega\left(g(n)\right) \Rightarrow f(n) = \Omega\left(g(n)\right)$

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} = \infty$$

تحلیل الگوریتم ها

Transitivity:

$$f(n) = \Theta(g(n)) \text{ and } g(n) = \Theta(h(n)) \text{ imply } f(n) = \Theta(h(n)),$$
$$f(n) = O(g(n)) \text{ and } g(n) = O(h(n)) \text{ imply } f(n) = O(h(n)),$$
$$f(n) = \Omega(g(n)) \text{ and } g(n) = \Omega(h(n)) \text{ imply } f(n) = \Omega(h(n)),$$
$$f(n) = o(g(n)) \text{ and } g(n) = o(h(n)) \text{ imply } f(n) = o(h(n)),$$
$$f(n) = \omega(g(n)) \text{ and } g(n) = \omega(h(n)) \text{ imply } f(n) = \omega(h(n)).$$

تحلیل الگوریتم ها

Reflexivity:

$$f(n) = \Theta(f(n)) ,$$
$$f(n) = O(f(n)) ,$$
$$f(n) = \Omega(f(n)) .$$

Symmetry:

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

Transpose symmetry:

$$f(n) = O(g(n)) \quad \text{if and only if} \quad g(n) = \Omega(f(n)) ,$$
$$f(n) = o(g(n)) \quad \text{if and only if} \quad g(n) = \omega(f(n)) .$$

تحلیل الگوریتم ها

$$
\begin{array}{lcl}
f(n) = O(g(n)) & \approx & a \le b\,, \\
f(n) = \Omega(g(n)) & \approx & a \ge b\,, \\
f(n) = \Theta(g(n)) & \approx & a = b\,, \\
f(n) = o(g(n)) & \approx & a < b\,, \\
f(n) = \omega(g(n)) & \approx & a > b\,.
\end{array}
$$

تحلیل الگوریتم ها

- نمادهای θ,ω,Ω,O,Q نمی توانند روی همه توابع تعریف شوند. (مثل توابع نوسانی)
- $f(n) = n^2$ $\qquad$ $g(n) = n^{1+sin(n)}$
- اثبات کنید:
- $P\ (n) = \sum_{i=0}^{K} a_i n^i$ $\qquad$ $P(n) = \theta(n^K)$
- $C_1 n^k \le a_0 + a_1 n + \cdots + a_k n^k \le C_2 n^k$

تحلیل الگوریتم ها