COT 6405 Introduction to Theory of Algorithms

Topic 2. Algorithm Analysis

Growth rate analysis

- A further abstraction that we use in algorithm analysis is to characterize in terms of growth classes.
  - Matrix multiplication time grows as n^3
  - Linear search time grows as n
  - Insertion sort time grows as n^2

Why is growth rate important?

- Actual execution time assuming 1,000,000 basic operations per second.

Input size	n	$n\lg n$	n^2	n^3	2^n
10	0.00001 sec	3.62e-5 sec	0.0001 sec	0.001 sec	<0.01 sec
100	0.0001 sec	6.52e-4 sec	0.01 sec	1 min	~∞ centuries
1000	0.001 sec	0.00978 sec	1 sec	17.64 min	~∞ centuries
10^4	0.01 sec	0.132 sec	1.692 min	11.76 days	~∞ centuries

Growth "classes" of functions

- O(g(n)) big oh: upper bound on the growth rate of a function;
  - That is, a function belongs to class O(g(n)) if g(n) is an upper bound on its growth rate
- Ω(g(n)) big omega: lower bound on the growth rate of a function
- Θ (g(n)) big theta: exact bound on the growth rate of a function

Determining the growth class

- A function may belong to multiple growth classes
  - For example, a function describing the worst case number of basic operations of an algorithm might be O(n^2) and $\Omega(n\lg n)$
  - If we find example inputs for which the growth rate is n^2, then we can also say $\Theta(n^2)$

Little oh and little omega

- o(g(n)) little oh: used to denote functions that grow more slowly than g(n);
  - For example, 3n + o(n) indicate that it's O(n) with a small leading constant
- ω(g(n)) little omega: denotes functions that grow faster than g(n);
  - Rarely used but included for completeness

Precise definitions of big oh and big omega

- f(n) ∈ O(g(n)) iff there exist c > 0 and n_0> 0 such that f(n) $\leq$ cg(n) for all n $\geq n_0$
- f(n) ∈ Ω(g(n)) iff there exist c > 0 and n_0 > 0 such that f(n) $\geq$ cg(n) for all n $\geq n_0$
- Θ (g(n) ∈ O(g(n)) ∩ Ω(g(n))

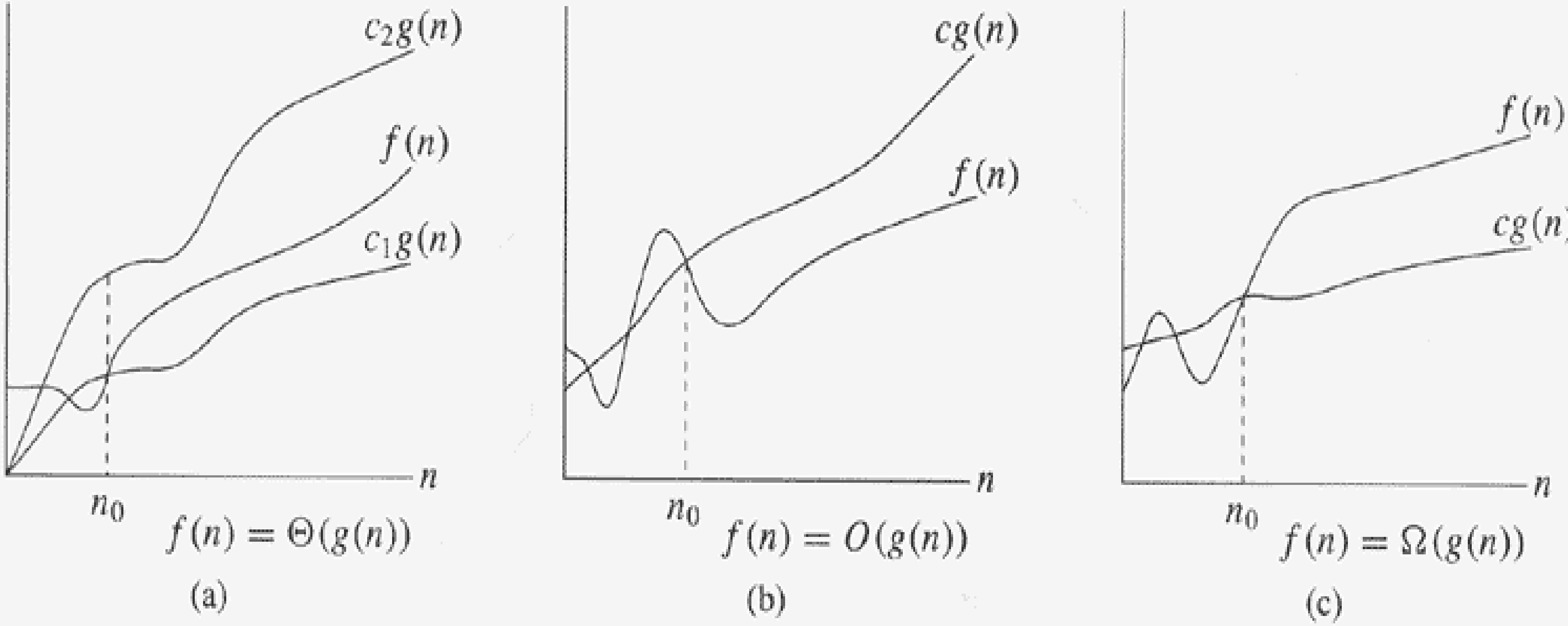

Figure 3.1 Graphic examples of the Θ, O, and Ω notations. In each part, the value of n_0 shown is the minimum possible value; any greater value would also work. **(a)** Θ-notation bounds a function to within constant factors. We write $f(n) = \Theta(g(n))$ if there exist positive constants n_0, c_1, and c_2 such that to the right of n_0, the value of $f(n)$ always lies between $c_1g(n)$ and $c_2g(n)$ inclusive. **(b)** O-notation gives an upper bound for a function to within a constant factor. We write $f(n) = O(g(n))$ if there are positive constants n_0 and c such that to the right of n_0, the value of $f(n)$ always lies on or below $cg(n)$. **(c)** Ω-notation gives a lower bound for a function to within a constant factor. We write $f(n) = \Omega(g(n))$ if there are positive constants n_0 and c such that to the right of n_0, the value of $f(n)$ always lies on or above $cg(n)$.

Exercises

- How do we define that a function f(n) has an upper bound g(n), i.e., f(n) is in O(g(n)) ?

- How do we define that a function f(n) has an lower bound g(n), i.e., f(n) is in Ω(g(n)) ?

- How do we define that a function f(n) has an tight bound g(n), i.e., f(n) is in Θ(g(n)) ?

An example of big oh and big omega

- How to prove n^2 + 2n + lg n ∈ O(n^3) ?

$n^2 + 2n + \lg n \in O(n^3)$

$$
\begin{array}{llll}
\textit{Proof.} \quad n^2 + 2n + \lg n & \leq & n^2 + 2n + n & \text{as long as } n \geq 1 \\
& = & n^2 + 3n & \\
& \leq & n^3 + 3n^3 & (\text{if } n \geq 1) \\
& = & 4n^3 &
\end{array}
$$

This satisfies the definition of $\mathrm{O}(n^3)$ with $c = 4$ and $n_0 = 1$.

Exercises (cont'd)

- Ex1: Prove $n^3 - 10n^2 \notin O(n^2)$
- Ex2: Prove $5n^3 - 3n^2 + 2n - 6 \in \Theta(n^3)$

Exercises (cont'd)

$n^3 - 10n^2 \notin O(n^2)$

Proof. Otherwise there must exist $c > 0$ and $n_0 > 0$ with $n^3 - 10n^2 \leq cn^2$ *for all* $n \geq n_0$.
But then $n^3 \leq (c + 10)n^2$ (for all $n \geq n_0$) and $n \leq c + 10$. The latter is impossible for a given c and all $n \geq n_0$.

Exercises (cont'd)

$5n^3 - 3n^2 + 2n - 6 \in \Theta(n^3)$

Proof.

First show that it's in $\mathrm{O}(n^3)$:

$$\begin{aligned} 5n^3 - 3n^2 + 2n - 6 &\leq 5n^3 + 2n \\ &\leq 7n^3 \qquad \text{when } n \geq 1 \end{aligned}$$

so it's $\mathrm{O}(n^3)$ with $c = 7$ and $n_0 = 1$.

Then that it's in $\Omega(n^3)$:

$$\begin{aligned} 5n^3 - 3n^2 + 2n - 6 &\geq 5n^3 - 3n^2 - 6 \\ &\geq \tfrac{5}{2}n^3 \qquad \text{when } \tfrac{5}{2}n^3 \geq 3n^2 + 6 \text{ or } n \geq 2 \\ & \qquad\qquad \text{(good enough)} \end{aligned}$$

Exercises (cont'd)

Prove $5n^3 - 3n^2 + 2n - 6 \in \Omega(n^3)$

$\leftrightarrow 5n^3 - 3n^2 + 2n - 6 \geq c\, n^3$ for all $n \geq n_0$

Enforcing an lower bound and we get

$$5n^3 - 3n^2 + 2n - 6 \geq 5n^3 - 3n^2 - 6$$

We thus would like $5n^3 - 3n^2 - 6 \geq c\, n^3$, equivalently

$$(5 - c)n^3 \geq 3n^2 + 6$$

We search for c and n_0, c can be any number less than 5, and without loss of generality, we assume $c = 1$.

We need to find n_0 that allows $4n^3 \geq 3n^2 + 6$, apparently when $n \geq 2$, this inequality always holds, and thus $c = 1$, and $n_0 = 2$.

Exercises (logarithms and exponents)

- Ex 3: $\ln n \in \Theta(\lg n)$
- Ex4: $e^n \notin O(n^t)$ for any fixed t
- Ex5: $e^n \notin O(e^t)$ for any fixed t

Exercises (cont'd)

$\ln n \in \Theta(\lg n)$

Proof. Recall that $\ln n = \log_e n$ and $\lg n = \log_2 n$. Using one of the mathematical identities on the first page, we have

$$\ln n = \frac{\lg n}{\lg e}$$

So $c \lg n \leq \ln n \leq c \lg n$, where $c = \frac{1}{\lg e}$, for all $n \geq 1$, which proves both $\mathrm{O}(\lg n)$ and $\Omega(\lg n)$.

Exercises (cont'd)

- $e^n \notin O(n^t)$ for any fixed t
- $e^n \notin O(e^t)$ for any fixed t

Exercises (cont'd)

- $e^n \notin O(n^t)$ for any fixed t

Proof: Otherwise there exist *c* > 0 and n_0> 0 with

$$e^n \le cn^t \text{ for all } n \ge n_0.$$

But then (taking natural log's of both sides) $n \le \ln c + t \ln n$.

This translates into (divide each side by ln*n*) $\frac{n}{lnn} \le \frac{lnc}{lnn} + t$.

When $n \ge e, \frac{n}{lnn} \le \frac{lnc}{lnn} + t \le lnc + t$ (a constant). On the other hand,

$$\lim_{n\to\infty} \frac{n}{lnn} = \lim_{n\to\infty} \frac{1}{1/n} = \infty$$

Exercises (cont'd)

- $e^n \notin O(e^t)$ for any fixed t

Proof: Otherwise there exist c > 0 and n_0> 0 with

$$e^n \leq ce^t \text{ for all } n \geq n_0.$$

But then (taking natural log's of both sides) $n \leq \ln c + t$.

c is a constant, and thus lnc + t is a fixed value. It is impossible to find an $n_0 > 0$ so that for all $n \geq n_0$, n is less than or equal to a fixed value.

Little oh and little omega

- f(n) $\in$ o(g(n)) iff for all c > 0 there exists n_0> 0 such that 0 $\leq$ f(n) $<$ cg(n) for all n $\geq$ n_0
- f(n) $\in$ ω(g(n)) iff for all c > 0 there exists n_0 > 0 such that 0 $\leq$ cg(n) < f(n) for all n $\geq$ n_0

Limits and notation

- Limits can be helpful in determining the growth rate of functions
  - $\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$ implies $f(n) \in$ o(g(n)), that is, $f(n) \notin \Omega$(g(n))
  - $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$ implies $f(n) \in \omega$(g(n)), that is, $f(n) \notin O$(g(n))
  - $\lim_{n\to\infty} \frac{f(n)}{g(n)} = d > 0$ implies $f(n) \in \Theta$(g(n))

An example of little oh and little omega

- $2^n \in o(3^n)$
- *Proof:* $\lim_{n\to\infty} (2/3)^n = 0$. This means that

$$2^n \in o(3^n)$$

Limits and notation (cont'd)

- Warning: the converses are not necessarily true. Limits may not exist in some cases where growth classes are well-defined.