Asymptotic Notation

Dr. K.RAGHAVA RAO

Professor in CSE

KL University

krraocse@gmail.com

http://mcadaa.blog.com

23 March 2012

Introduction:Asymptotic Notation

- Definition: *Asymptotic complexity* is a way of expressing the *main component* of the cost of an algorithm, using idealized units of computational work.
- Consider, for example, the algorithm for sorting a deck of cards, which proceeds by repeatedly searching through the deck for the lowest card.
- A problem may have numerous algorithmic solutions. In order to choose the best algorithm for a particular task, you need to be able to judge how long a particular solution will take to run. Or, more accurately, you need to be able to judge how long two solutions will take to run, and choose the better of the two. You don't need to know how many minutes and seconds they will take, but you do need some way to compare algorithms against one another.

Asymptotic Complexity

- Running time of an algorithm as a function of input size *n* **for large *n***.

- Expressed using only the **highest-order term** in the expression for the exact running time.
  - Instead of exact running time, say Q(n^2).

- Describes behavior of function in the limit.

- Written using ***Asymptotic Notation*.**

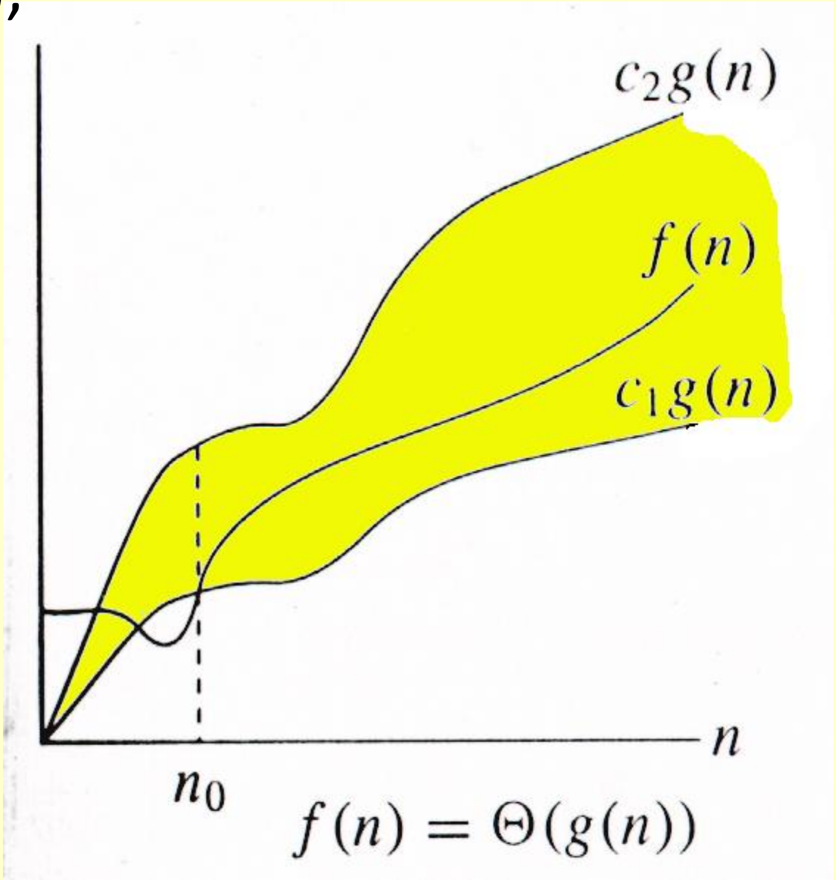
Asymptotic Notation

- **Q, *O*, W, *o*, w**
- Defined for functions over the natural numbers.
  - **Ex:** $f(n) = Q(n^2)$.
  - Describes how $f(n)$ grows in comparison to n^2.
- Define a ***set*** of functions; in practice used to compare two function sizes.
- The notations describe different rate-of-growth relations between the defining function and the defined set of functions.

Θ-notation

For function $g(n)$, we define $\Theta(g(n))$, big-Theta of n, as the set:

$\Theta(g(n)) = \{f(n) :$
$\exists$ positive constants c_1, c_2, and n_0,
such that $\forall n \geq n_0$,
we have $0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)$
$\}$



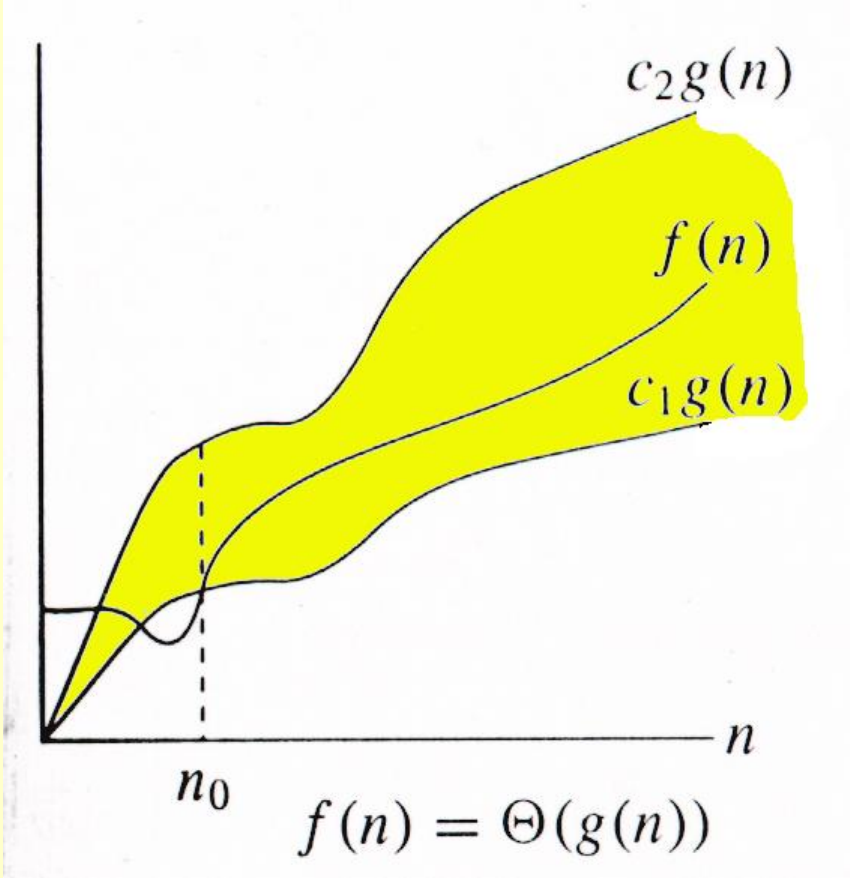
Intuitively: Set of all functions that have the same rate of growth as g(n).

g(n) is an asymptotically tight bound for f(n).

Θ-notation

For function $g(n)$, we define $\Theta(g(n))$, big-Theta of n, as the set:

$\Theta(g(n)) = \{f(n) :$
$\exists$ positive constants c_1, c_2, and n_0,
such that $\forall n \geq n_0$,
we have $0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)$
$\}$



Technically, $f(n) \in \Theta(g(n))$.
Older usage, $f(n) = \Theta(g(n))$.
Both accepted.

$f(n)$ and $g(n)$ are nonnegative, for large n.

Examples

$\Theta(g(n)) = \{f(n)$: $\exists$ positive constants c_1, c_2, and n_0, such that $\forall n \geq n_0$, $0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)\}$

3n+2 = **Θ(n) as 3n+2>=3n for all n>=2 and 3n+2<=4n for all n>=2,**

So c1=3 and c2=4 and n_0=2. So,3n+3= **Θ(n),**

$10n^2+4n+2 = \Theta(n^2)$,$6*2^n+n^2 = \Theta(2^n)$ and

10*log n+4= Θ(log n).

3n+2# Θ(1), 3n+3# $\Theta(n^2)$, $10n^2+4n+2$# Θ(n), $10n^2+4n+2$# Θ(1)

Example

- $10n^2 - 3n = \Theta(n^2)$
- What constants for n_0, c_1, and c_2 will work?
- Make c_1 a little smaller than the leading coefficient, and c_2 a little bigger.
- ***To compare orders of growth, look at the leading term.***
- Exercise: Prove that $n^2/2-3n = \Theta(n^2)$

Example

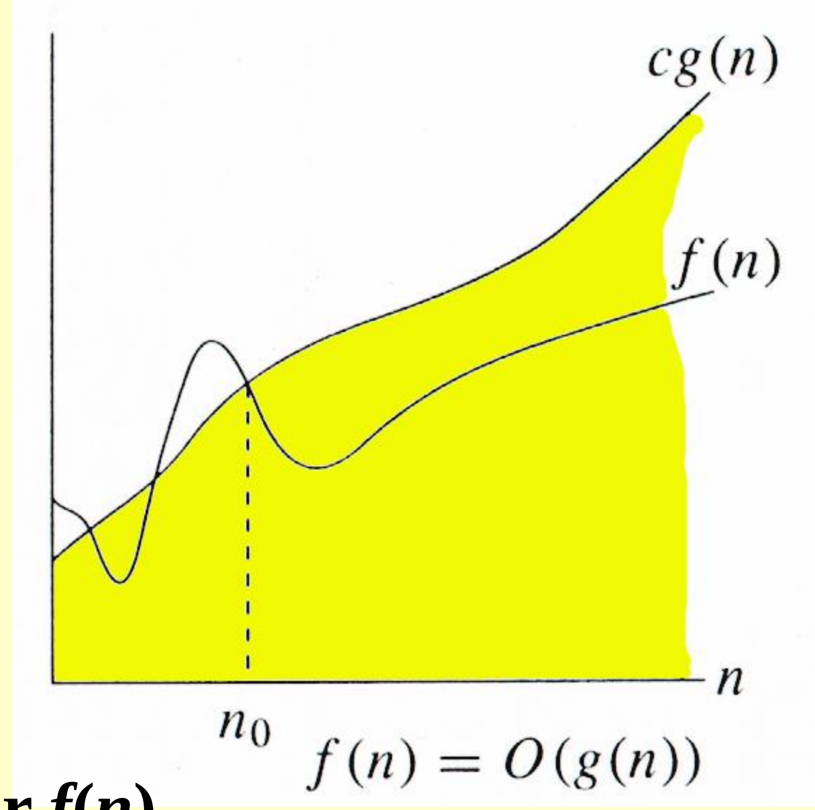
> $\Theta(g(n)) = \{f(n)$: $\exists$ **positive constants** c_1, c_2, **and** n_0, **such that** $\forall n \geq n_0$, $\quad 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n)\}$

- Is $3n^3 \in \Theta(n^4)$??
- How about $2^{2n} \in \Theta(2^n)$??

O-notation

For function $g(n)$, we define $O(g(n))$, big-O of n, as the set:

$O(g(n)) = \{f(n) :$
$\exists$ positive constants c and n_0,
such that $\forall n \geq n_0$,
we have $0 \leq f(n) \leq cg(n)$ $\}$



Intuitively: Set of all functions whose *rate of growth* is the same as or lower than that of $g(n)$.

$g(n)$ is an *asymptotic upper bound* for $f(n)$.

$f(n) = \Theta(g(n)) \Rightarrow f(n) = O(g(n))$.
$\Theta(g(n)) \subset O(g(n))$.

Asymptotic Notation (O)

- Examples
  - $3n+2=O(n)$ /* $3n+2\leq 4n$ for $n\geq 2$ */
  - $3n+3=O(n)$ /* $3n+3\leq 4n$ for $n\geq 3$ */
  - $100n+6=O(n)$ /* $100n+6\leq 101n$ for $n\geq 10$ */
  - $10n^2+4n+2=O(n^2)$ /* $10n^2+4n+2\leq 11n^2$ for $n\geq 5$ */
  - $6*2^n+n^2=O(2^n)$ /* $6*2^n+n^2 \leq 7*2^n$ for $n\geq 4$ */*

More Big-Oh Examples

- $7n-2$

  $7n-2$ is $O(n)$

  need $c > 0$ and $n_0 \geq 1$ such that $7n-2 \leq c \cdot n$ for $n \geq n_0$

  this is true for $c = 7$ and $n_0 = 1$

- $3n^3 + 20n^2 + 5$

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

  need $c > 0$ and $n_0 \geq 1$ such that $3n^3 + 20n^2 + 5 \leq c \cdot n^3$ for $n \geq n_0$

  this is true for $c = 4$ and $n_0 = 21$

- $3 \log n + \log \log n$

  $3 \log n + \log \log n$ is $O(\log n)$

  need $c > 0$ and $n_0 \geq 1$ such that $3 \log n + \log \log n \leq c \cdot \log n$ for $n \geq n_0$

  this is true for $c = 4$ and $n_0 = 2$

Big-Oh and Growth Rate

- The big-Oh notation gives an upper bound on the growth rate of a function
- The statement "$f(n)$ is $O(g(n))$" means that the growth rate of $f(n)$ is no more than the growth rate of $g(n)$
- We can use the big-Oh notation to rank functions according to their growth rate

	$f(n)$ is $O(g(n))$	$g(n)$ is $O(f(n))$
$g(n)$ grows more	Yes	No
$f(n)$ grows more	No	Yes
Same growth	Yes	Yes

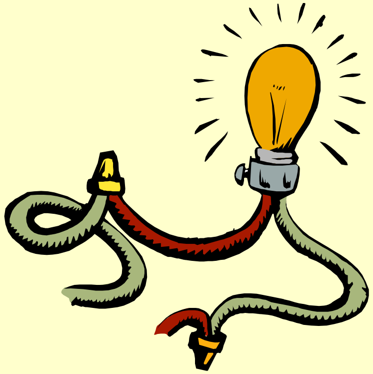
Big-Oh Rules

- If is $\boldsymbol{f}(\boldsymbol{n})$ a polynomial of degree $\boldsymbol{d}$, then $\boldsymbol{f}(\boldsymbol{n})$ is $\boldsymbol{O}(\boldsymbol{n}^{\boldsymbol{d}})$, i.e.,
    1. Drop lower-order terms
    2. Drop constant factors
- Use the smallest possible class of functions
    - Say "$2\boldsymbol{n}$ is $\boldsymbol{O}(\boldsymbol{n})$" instead of "$2\boldsymbol{n}$ is $\boldsymbol{O}(\boldsymbol{n}^2)$"
- Use the simplest expression of the class
    - Say "$3\boldsymbol{n} + 5$ is $\boldsymbol{O}(\boldsymbol{n})$" instead of "$3\boldsymbol{n} + 5$ is $\boldsymbol{O}(3\boldsymbol{n})$"

Relatives of Big-Oh

- **big-Omega**
  - f(n) is $\Omega(g(n))$ if there is a constant $c > 0$ and an integer constant $n_0 \geq 1$ such that $f(n) \geq c \cdot g(n)$ for $n \geq n_0$
- **big-Theta**
  - f(n) is $\Theta(g(n))$ if there are constants $c' > 0$ and $c'' > 0$ and an integer constant $n_0 \geq 1$ such that $c' \cdot g(n) \leq f(n) \leq c'' \cdot g(n)$ for $n \geq n_0$
- **little-oh**
  - f(n) is $o(g(n))$ if, for any constant $c > 0$, there is an integer constant $n_0 > 0$ such that $f(n) < c \cdot g(n)$ for $n \geq n_0$
- **little-omega**
  - f(n) is $\omega(g(n))$ if, for any constant $c > 0$, there is an integer constant $n_0 > 0$ such that $f(n) > c \cdot g(n)$ for $n \geq n_0$

Intuition for Asymptotic Notation



Big-Oh

- f(n) is O(g(n)) if f(n) is asymptotically **less than or equal** to g(n)

big-Omega

- f(n) is Ω(g(n)) if f(n) is asymptotically **greater than or equal** to g(n)

big-Theta

- f(n) is Θ(g(n)) if f(n) is asymptotically **equal** to g(n)

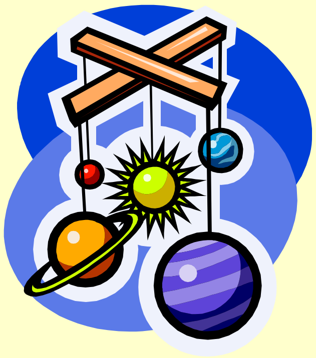
little-oh

- f(n) is o(g(n)) if f(n) is asymptotically **strictly less** than g(n)

little-omega

- f(n) is ω(g(n)) if is asymptotically **strictly greater** than g(n)

Example Uses of the Relatives of Big-Oh



- **$5n^2$ is $\Omega(n^2)$**

  f(n) is $\Omega(g(n))$ if there is a constant $c > 0$ and an integer constant $n_0 \geq 1$ such that $f(n) \geq c \bullet g(n)$ for $n \geq n_0$

  let $c = 5$ and $n_0 = 1$

- **$5n^2$ is $\Omega(n)$**

  f(n) is $\Omega(g(n))$ if there is a constant $c > 0$ and an integer constant $n_0 \geq 1$ such that $f(n) \geq c \bullet g(n)$ for $n \geq n_0$

  let $c = 1$ and $n_0 = 1$

- **$5n^2$ is $\omega(n)$**

  f(n) is $\omega(g(n))$ if, for any constant $c > 0$, there is an integer constant $n_0 > 0$ such that $f(n) > c \bullet g(n)$ for $n \geq n_0$

  need $5n_0^2 > c \bullet n_0 \rightarrow$ given c, the n_0 that satifies this is $n_0 > c/5 > 0$

- $O(1)$: constant
- $O(n)$: linear
- $O(n^2)$: quadratic
- $O(n^3)$: cubic
- $O(2^n)$: exponential
- $O(\log n)$
- $O(n \log n)$

***Figure 1.7:**Function values (p.38)

		Instance characteristic n					
Time	Name	1	2	4	8	16	32
1	Constant	1	1	1	1	1	1
$\log n$	Logarithmic	0	1	2	3	4	5
n	Linear	1	2	4	8	16	32
$n \log n$	Log linear	0	2	8	24	64	160
n^2	Quadratic	1	4	16	64	256	1024
n^3	Cubic	1	8	64	512	4096	32768
2^n	Exponential	2	4	16	256	65536	4294967296
$n!$	Factorial	1	2	24	40326	20922789888000	26313×10^{33}

***Figure 1.8:**Plot of function values(p.39)*

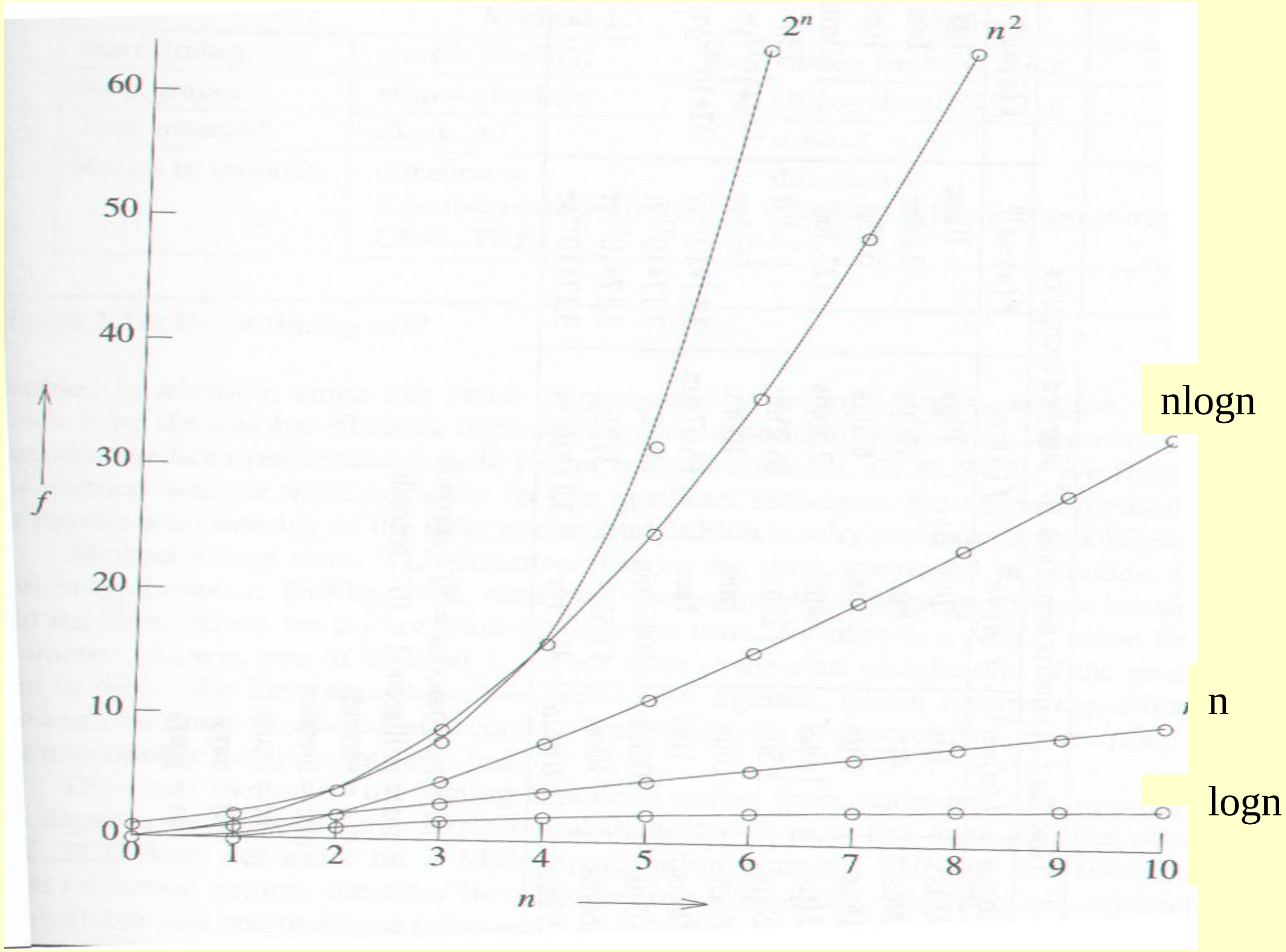


***Figure 1.9:**Times on a 1 billion instruction per second computer(p.40)

	Time for $f(n)$ instructions on a 10^9 instr/sec computer						
n	$f(n)=n$	$f(n)=\log_2 n$	$f(n)=n^2$	$f(n)=n^3$	$f(n)=n^4$	$f(n)=n^{10}$	$f(n)=2^n$
10	.01μs	.03μs	.1μs	1μs	10μs	10sec	1μs
20	.02μs	.09μs	.4μs	8μs	160μs	2.84hr	1ms
30	.03μs	.15μs	.9μs	27μs	810μs	6.83d	1sec
40	.04μs	.21μs	1.6μs	64μs	2.56ms	121.36d	18.3min
50	.05μs	.28μs	2.5μs	125μs	6.25ms	3.1yr	13d
100	.10μs	.66μs	10μs	1ms	100ms	3171yr	$4*10^{13}$ yr
1,000	1.00μs	9.96μs	1ms	1sec	16.67min	$3.17*10^{13}$ yr	$32*10^{283}$ yr
10,000	10.00μs	130.03μs	100ms	16.67min	115.7d	$3.17*10^{23}$ yr	
100,000	100.00μs	1.66ms	10sec	11.57d	3171yr	$3.17*10^{33}$ yr	
1,000,000	1.00ms	19.92ms	16.67min	31.71yr	$3.17*10^{7}$ yr	$3.17*10^{43}$ yr	

μs = microsecond = 10^{-6} seconds
ms = millisecond = 10^{-3} seconds
sec = seconds
min = minutes
hr = hours
d = days
yr = years

Examples

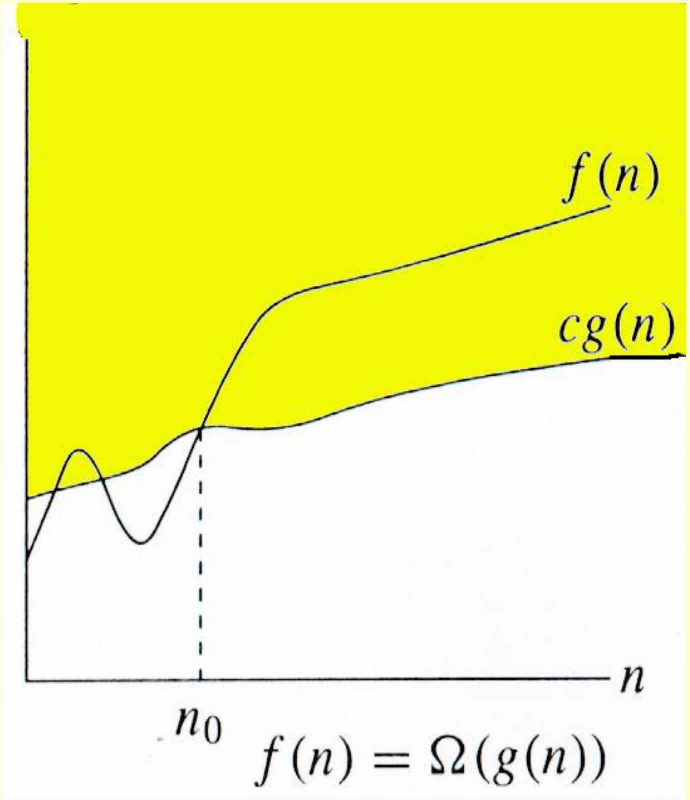
$O(g(n)) = \{f(n) :$ $\exists$ **positive constants** c **and** n_0**,**
such that $\forall n \geq n_0$**, we have** $0 \leq f(n) \leq cg(n)$ $\}$

- Any linear *function* $an + b$ is in $O(n^2)$. **How?**
- Show that $3n^3 = O(n^4)$ for appropriate c and n_0.

Ω -notation

For function $g(n)$, we define $\Omega(g(n))$, big-Omega of n, as the set:

$\Omega(g(n)) = \{f(n)$:
$\exists$ positive constants c and n_0,
such that $\forall n \geq n_0$,
we have $0 \leq cg(n) \leq f(n)\}$



Intuitively: Set of all functions whose *rate of growth* is the same as or higher than that of $g(n)$.

$g(n)$ is an *asymptotic lower bound* for $f(n)$.

$f(n) = \Theta(g(n)) \Rightarrow f(n) = \Omega(g(n))$.
$\Theta(g(n)) \subset \Omega(g(n))$.

Examples

$\Omega(g(n)) = \{f(n) :\exists$ positive constants c and n_0, such that $\forall n \geq n_0$, we have $0 \leq cg(n) \leq f(n)\}$

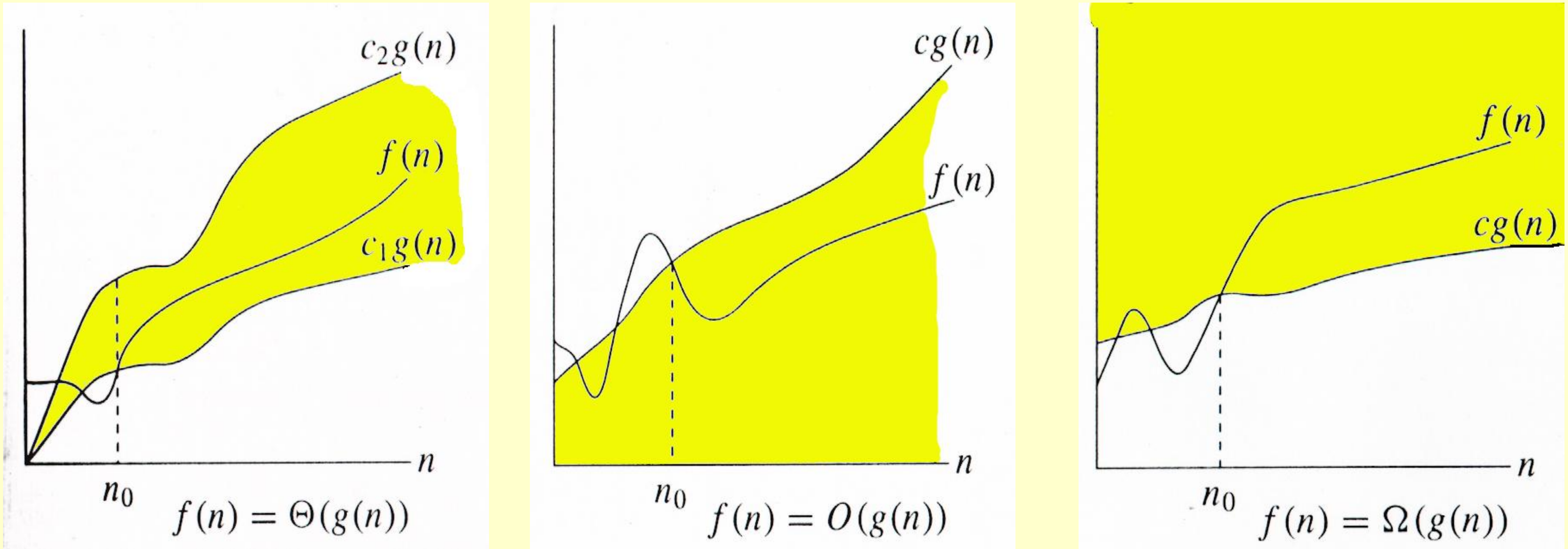
- $3n+2= \Omega(n)$ as $3n+2>=3n$ for $n>=1$
- $3n+3= \Omega(n)$ as $3n+3>=3n$ for $n>=1$
- $10n^2 +4n+2= \Omega(n^2)$ as $10n^2 +4n+2>=n2$ for $n>=1$

$6*2^n +n^2 = \Omega(n^2)$ as $6*2^n +n^2>=2^n$ for $n>=1$.

$10n^2+4n+2=\Omega(n)$ and $10n^2+4n+2= \Omega(1)$

$6*2^n+n^2= \Omega(n^2)$, $6*2^n+n^2= \Omega(n)$ also $6*2^n+n^2= \Omega(1)$

Relations Between Θ, O, Ω



Relations Between Θ, Ω, O

Theorem : For any two functions $g(n)$ and $f(n)$,

$f(n) = \Theta(g(n))$ iff

$f(n) = O(g(n))$ and $f(n) = \Omega(g(n))$.

- I.e., $\Theta(g(n)) = O(g(n)) \cap \Omega(g(n))$
- In practice, asymptotically tight bounds are obtained from asymptotic upper and lower bounds.

Running Times

- "Running time is $O(f(n))$" $\Rightarrow$ Worst case is $O(f(n))$
- $O(f(n))$ bound on the worst-case running time $\Rightarrow$ $O(f(n))$ bound on the running time of every input.
- $\Theta(f(n))$ bound on the worst-case running time $\not\Rightarrow$ $\Theta(f(n))$ bound on the running time of every input.
- "Running time is $\Omega(f(n))$" $\Rightarrow$ Best case is $\Omega(f(n))$
- Can still say "Worst-case running time is $\Omega(f(n))$"
  - Means worst-case running time is given by some unspecified function $g(n) \in \Omega(f(n))$.

Asymptotic Notation in Equations

- Can use asymptotic notation in equations to replace expressions containing lower-order terms.
- For example,

  $4n^3 + 3n^2 + 2n + 1 = 4n^3 + 3n^2 + \Theta(n)$

  $= 4n^3 + \Theta(n^2) = \Theta(n^3)$. **How to interpret?**
- In equations, $\Theta(f(n))$ always stands for an ***anonymous function*** $g(n) \in \Theta(f(n))$
  - In the example above, $\Theta(n^2)$ stands for $3n^2 + 2n + 1$.

Little o-notation

For a given function $g(n)$, the set little-o:

$o(g(n)) = \{f(n)$: $\forall\, c > 0$, $\exists\, n_0 > 0$ such that $\forall\, n \geq n_0$, we have $0 \leq f(n) < cg(n)\}$.

$f(n)$ becomes insignificant relative to $g(n)$ as n approaches infinity:

$$\lim_{n\to\infty} [f(n) / g(n)] = 0$$

$g(n)$ is an ***upper bound*** for $f(n)$ that is not asymptotically tight.

Observe the difference in this definition from previous ones. **Why?**

Little ω –notation

For a given function $g(n)$, the set little-omega:

$\omega(g(n))$ = $\{f(n)$: $\forall\, c > 0$, $\exists\, n_0 > 0$ such that $\forall\, n \geq n_0$, we have $0 \leq cg(n) < f(n)\}$.

$f(n)$ becomes arbitrarily large relative to $g(n)$ as n approaches infinity:

$$\lim_{n\to\infty} [f(n) / g(n)] = \infty.$$

$g(n)$ is a ***lower bound*** for $f(n)$ that is not asymptotically tight.