

**CS11001/CS11002**  
**Programming and Data**  
**Structures (PDS)**  
**(Theory: 3-1-0)**

**Growth of Functions**

*Debdeep Mukhopadhyay*  
*IIT Kharagpur*

## Asymptotic Performance

- Exact running time of an algorithm is not always required:
  - When the input size of a problem is very large. Like, in the insertion sort example if the number of elements we had to sort are very large.
  - Then the multiplicative constants and the lower order terms can be neglected.
- ***How the running time of an algorithm increases when the input increases unbounded ?***

## Growth of Functions

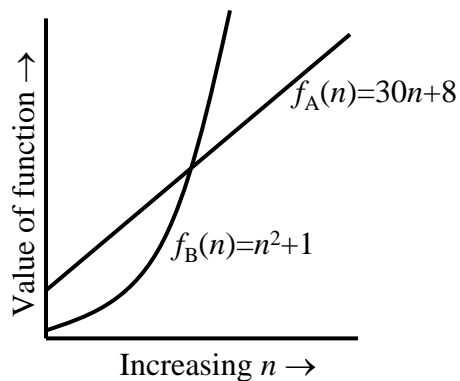
- For functions over numbers, we often need to know a rough measure of *how fast a function grows*.
- If  $f(x)$  is *faster growing* than  $g(x)$ , then  $f(x)$  always eventually becomes larger than  $g(x)$  *in the limit* (for large enough values of  $x$ ).
- Useful in engineering for showing that one design *scales* better or worse than another.

## Growth of Functions

- Suppose you are designing a web site to process user data (e.g., financial records).
- Suppose database program A takes  $f_A(n)=30n+8$  microseconds to process any  $n$  records, while program B takes  $f_B(n)=n^2+1$  microseconds to process the  $n$  records.
- Which program do you choose, knowing you'll want to support millions of users?

## Visualizing Growth of Functions

- On a graph, as you go to the right, a faster growing function eventually becomes larger...



## Definition: $O(g)$ , at most order $g$

Let  $g$  be any function  $\mathbf{R} \rightarrow \mathbf{R}$ .

- Define “at most order  $g$ ”, written  $O(g)$ , to be:
  - $\{f: \mathbf{R} \rightarrow \mathbf{R} \mid \exists +ve\ c, k: \forall x > k: 0 \leq f(x) \leq cg(x)\}$
  - “Beyond some point  $k$ , function  $f$  is at most a constant  $c$  times  $g$  (i.e., proportional to  $g$ ).”
  - We are dealing with asymptotically nonnegative elements of the set
- “ $f$  is at most order  $g$ ”, or “ $f$  is  $O(g)$ ”, or “ $f = O(g)$ ” all just mean that  $f \in O(g)$ .

## Points about the definition

- Note that  $f$  is  $O(g)$  so long as *any* values of  $c$  and  $k$  exist that satisfy the definition.
- But: The particular  $c, k$ , values that make the statement true are *not* unique: **Any larger value of  $c$  and/or  $k$  will also work.**
- You are **not** required to find the smallest  $c$  and  $k$  values that work. (Indeed, in some cases, there may be no smallest values!)

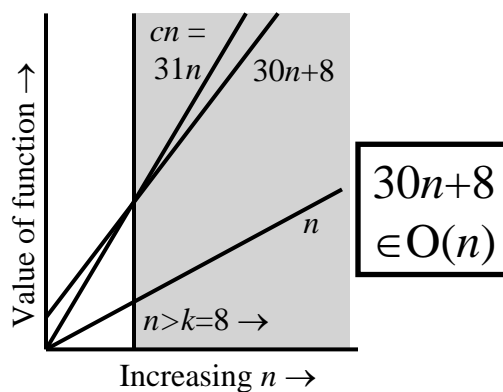
However, you should **prove** that the values you choose do work.

## “Big-O” Proof Examples

- Show that  $30n+8$  is  $O(n)$ .
  - Show  $\exists c, k: \forall n > k: 30n+8 \leq cn$ .
    - Let  $c=31, k=8$ . Assume  $n > k=8$ . Then  $cn = 31n = 30n + n > 30n+8$ , so  $30n+8 < cn$ .
- Show that  $n^2+1$  is  $O(n^2)$ .
  - Show  $\exists c, k: \forall n > k: n^2+1 \leq cn^2$ .
    - Let  $c=2, k=1$ . Assume  $n > 1$ . Then  $cn^2 = 2n^2 = n^2+n^2 > n^2+1$ , or  $n^2+1 < cn^2$ .

## Big-O example, graphically

- Note  $30n+8$  isn't less than  $n$  anywhere ( $n > 0$ ).
- It isn't even less than  $31n$  everywhere.
- But it is less than  $31n$  everywhere to the right of  $n=8$ .



### Definition: $\Theta(g)$ , exactly order $g$

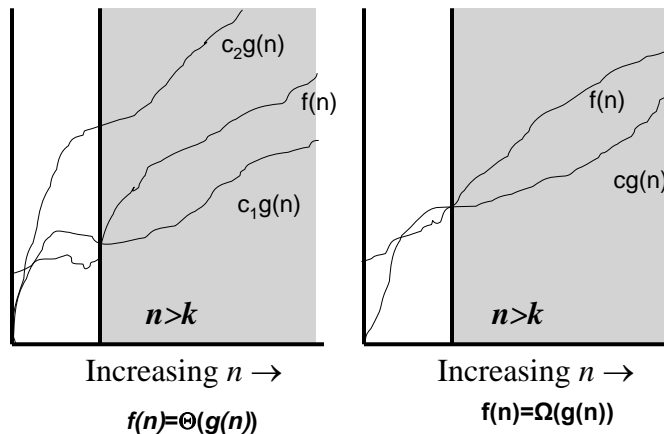
- If  $f \in O(g)$  and  $g \in O(f)$  then we say “ $g$  and  $f$  are of the same order” or “ $f$  is (exactly or tightly) order  $g$ ” and write  $f \in \Theta(g)$ .
- Another equivalent definition:  
 $\Theta(g) \equiv \{f: \mathbb{R} \rightarrow \mathbb{R} \mid \exists +ve c_1 c_2 k \forall x > k: 0 \leq c_1 g(x) \leq f(x) \leq c_2 g(x)\}$
- “Everywhere beyond some point  $k$ ,  $f(x)$  lies in between two multiples of  $g(x)$ .”

### Definition: $\Omega(g)$ , at least order $g$

Let  $g$  be any function  $\mathbb{R} \rightarrow \mathbb{R}$ .

- Define “at most order  $g$ ”, written  $O(g)$ , to be:  
 $\{f: \mathbb{R} \rightarrow \mathbb{R} \mid \exists +ve c, k: \forall x > k: f(x) \leq cg(x) \geq 0\}$ 
  - “Beyond some point  $k$ , function  $f$  is at least a constant  $c$  times  $g$  (i.e., proportional to  $g$ ).”
- “ $f$  is at least order  $g$ ”, or “ $f$  is  $\Omega(g)$ ”, or “ $f = \Omega(g)$ ” all just mean that  $f \in \Omega(g)$ .

## Graphical Representation



## An Example of Tight Bound ( $\Theta$ )

- Prove  $f(n) = \frac{1}{2}n^2 - 3n = \Theta(n^2)$
- In order to prove this we require constants:  $c_1$  and  $c_2$  s.t. :
  - $c_1n^2 \leq \frac{1}{2}n^2 - 3n \leq c_2n^2$ , for all  $n \geq n_0$
  - $c_1 \leq \frac{1}{2} - \frac{3}{n} \leq c_2$ , for all  $n \geq n_0$

n	1	2	3	4	5	6	7	8
f(n)	-5/2	-1	-1/2	-1/4	-1/10	0	1/14	>1/14

Set  $n_0=7$ ,  $c_1=1/14$ ,  $c_2=1/2$ .

It is not important to have an unique value, what is important that one set of values exist.

## For this class

- We shall be using the O-notation in the class frequently
- Point to be kept in mind: *If running time is  $O(n^2)$  => there is a function  $f(n)$  that is  $O(n^2)$  s.t. for any value of  $n \geq n_0$ , no matter what particular input of size  $n$  is chosen, the running time for that input is bounded from above by the value  $f(n)$ .*

## Next Day Recurrences