CMP_SC 3050: Asymptotic growth of functions

Rohit Chadha

September 3, 2014

Announcements

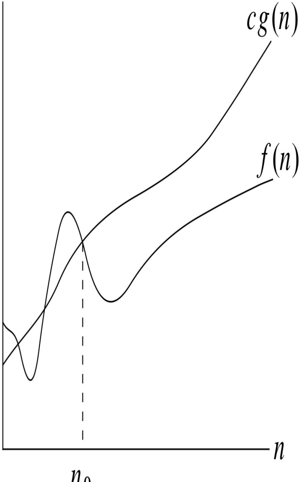
Homework 1 is on Blackboard. Due, September 9, 2014 at 12:29:29 pm.

- Instructor: Professor Rohit Chadha
  Email: chadhar@missouri.edu
  Office: 111 EBW
  Office hours: Wednesday, 2:00 - 4:00 pm
- Teaching Assistant: Kevin Melkowski
  Email: kmm4mc@mail.missouri.edu
  Office: 239 EBW
  Office hours: Friday, 1:00 - 3:00 pm
- Teaching Assistant: Avimanyou Vatsa
  Email: akvhxd@mail.missouri.edu
  Office: 303 EBN
  Office hours: Friday, 3:00 pm - 5:00 pm
- Teaching Assistant: Eric C. Gaudiello
  Office: 239 EBW
  Email: ecgprc@mail.missouri.edu
  Office hours: Tuesday, Thursday 3:30 pm - 4:30 pm

Asymptotic analysis

- Running times are functions of sizes of input
- Our focus is on how the running time grows as input size grows
- Such kind of analysis is called asymptotic analysis (analysis in the limit)
- In this analysis, we can ignore low-order terms and constant factors of functions
- The Θ notation is an example of this analysis (compares functions)
- The other example that you have seen in CMP_SC 2050 is the O (big-oh) notation

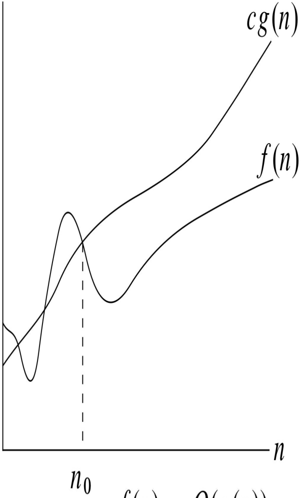
O-notation



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

Informal: Asymptotically, a function $f(n)$ is said to be $O(g(n))$ if in the limit $f(n)$ is smaller than some constant times $g(n)$

O-notation

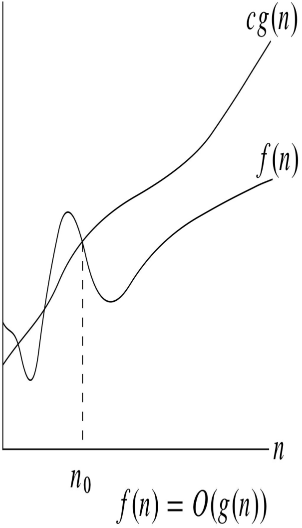


$f(n) = O(g(n))$

Informal: Asymptotically, a function $f(n)$ is said to be $O(g(n))$ if in the limit $f(n)$ is smaller than some constant times $g(n)$

Formally: There exist positive constants c and n_0 such that $0 \le f(n) \le cg(n)$ for all $n \ge n_0$

O-notation



Informal: Asymptotically, a function $f(n)$ is said to be $O(g(n))$ if in the limit $f(n)$ is smaller than some constant times $g(n)$

Formally: There exist positive constants c and n_0 such that $0 \leq f(n) \leq cg(n)$ for all $n \geq n_0$

$O \approx \leq$: Useful to think of O-notation as saying that the function f is less than the function g

Examples

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

Examples

- $2n$ is $O(n)$
- $2n$ is also $O(n^2)$

Examples

- $2n$ is $O(n)$
- $2n$ is also $O(n^2)$
- $2n$ is also $O(n^3)$

Examples

- $2n$ is $O(n)$
- $2n$ is also $O(n^2)$
- $2n$ is also $O(n^3)$
- $n^2 + n$ is $O(n^2)$

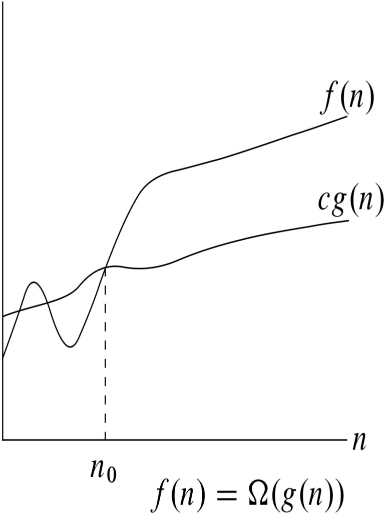
Examples

- $2n$ is $O(n)$
- $2n$ is also $O(n^2)$
- $2n$ is also $O(n^3)$
- $n^2 + n$ is $O(n^2)$
- $\log_2 n$ is $O(n)$

Examples

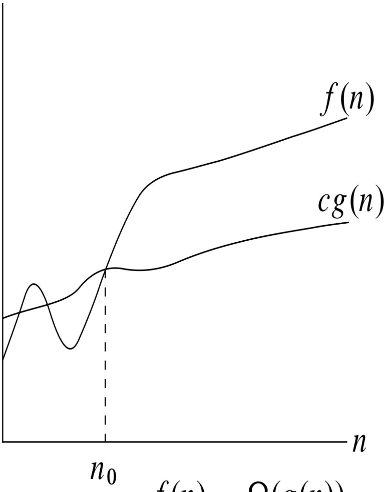
- $2n$ is $O(n)$
- $2n$ is also $O(n^2)$
- $2n$ is also $O(n^3)$
- $n^2 + n$ is $O(n^2)$
- $\log_2 n$ is $O(n)$
- $n \log_2 n$ is $O(n^2)$

Ω-notation



Informal: Asymptotically, a function $f(n)$ is said to be $O(g(n))$ if in the limit $f(n)$ is bigger than some constant times $g(n)$

Ω-notation

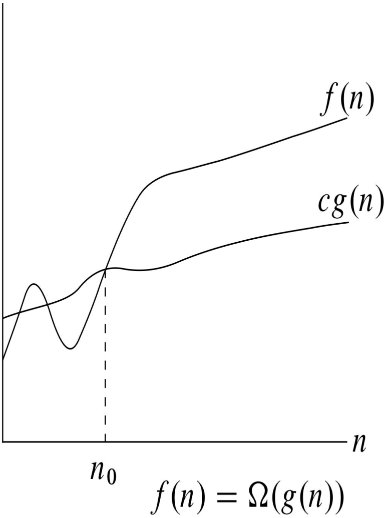


$f(n) = \Omega(g(n))$

Informal: Asymptotically, a function $f(n)$ is said to be $O(g(n))$ if in the limit $f(n)$ is bigger than some constant times $g(n)$

Formally: There exist positive constants c and n_0 such that $0 \leq cg(n) \leq f(n)$ for all $n \geq n_0$

Ω-notation



Informal: Asymptotically, a function $f(n)$ is said to be $O(g(n))$ if in the limit $f(n)$ is bigger than some constant times $g(n)$

Formally: There exist positive constants c and n_0 such that $0 \leq cg(n) \leq f(n)$ for all $n \geq n_0$

$\Omega \approx \geq$: Useful to think of Ω-notation as saying that the function f is bigger than the function g

Examples

- $2n$ is $\Omega(n)$

Examples

- $2n$ is $\Omega(n)$
- $2n$ is NOT $\Omega(n^2)$ but n^2 is $\Omega(2n)$

Examples

- $2n$ is $\Omega(n)$
- $2n$ is NOT $\Omega(n^2)$ but n^2 is $\Omega(2n)$
- $2n$ is NOT $\Omega(n^3)$ but n^3 is $\Omega(2n)$

Examples

- $2n$ is $\Omega(n)$
- $2n$ is NOT $\Omega(n^2)$ but n^2 is $\Omega(2n)$
- $2n$ is NOT $\Omega(n^3)$ but n^3 is $\Omega(2n)$
- $n^2 + n$ is $\Omega(n^2)$

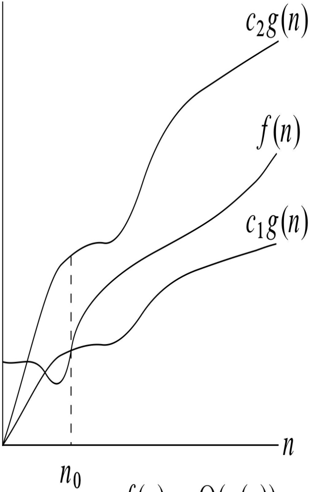
Examples

- $2n$ is $\Omega(n)$
- $2n$ is NOT $\Omega(n^2)$ but n^2 is $\Omega(2n)$
- $2n$ is NOT $\Omega(n^3)$ but n^3 is $\Omega(2n)$
- $n^2 + n$ is $\Omega(n^2)$
- $\log_2 n$ is NOT $\Omega(n)$ but n is $\Omega(\log_2 n)$

Examples

- $2n$ is $\Omega(n)$
- $2n$ is NOT $\Omega(n^2)$ but n^2 is $\Omega(2n)$
- $2n$ is NOT $\Omega(n^3)$ but n^3 is $\Omega(2n)$
- $n^2 + n$ is $\Omega(n^2)$
- $\log_2 n$ is NOT $\Omega(n)$ but n is $\Omega(\log_2 n)$
- $n \log_2 n$ is NOT $\Omega(n^2)$ but n^2 is $\Omega(n \log_2 n)$

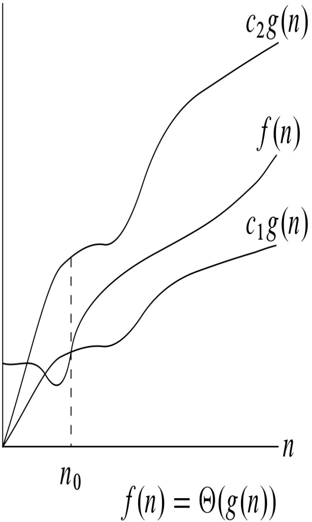
Θ-notation



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

A function $f(n)$ is said to be $\Theta(g(n))$ if $f(n)$ is both $O(g(n))$ and $\Omega(g(n))$

Θ-notation



A function $f(n)$ is said to be $\Theta(g(n))$ if $f(n)$ is both $O(g(n))$ and $\Omega(g(n))$

$\Theta \approx =$: Useful to think of Θ-notation as saying the function f behaves like the function g

Examples

- $2n$ is $\Theta(n)$

Examples

- $2n$ is $\Theta(n)$
- $2n$ is NOT $\Theta(n^2)$ and n^2 is also NOT $\Theta(2n)$

Examples

- $2n$ is $\Theta(n)$
- $2n$ is NOT $\Theta(n^2)$ and n^2 is also NOT $\Theta(2n)$
- $2n$ is NOT $\Theta(n^3)$ and n^3 is also NOT $\Theta(2n)$

Examples

- $2n$ is $\Theta(n)$
- $2n$ is NOT $\Theta(n^2)$ and n^2 is also NOT $\Theta(2n)$
- $2n$ is NOT $\Theta(n^3)$ and n^3 is also NOT $\Theta(2n)$
- $n^2 + n$ is $\Theta(n^2)$

Examples

- $2n$ is $\Theta(n)$
- $2n$ is NOT $\Theta(n^2)$ and n^2 is also NOT $\Theta(2n)$
- $2n$ is NOT $\Theta(n^3)$ and n^3 is also NOT $\Theta(2n)$
- $n^2 + n$ is $\Theta(n^2)$
- $\log_2 n$ is NOT $\Theta(n)$ and n is also NOT $\Theta(\log_2 n)$

Examples

- $2n$ is $\Theta(n)$
- $2n$ is NOT $\Theta(n^2)$ and n^2 is also NOT $\Theta(2n)$
- $2n$ is NOT $\Theta(n^3)$ and n^3 is also NOT $\Theta(2n)$
- $n^2 + n$ is $\Theta(n^2)$
- $\log_2 n$ is NOT $\Theta(n)$ and n is also NOT $\Theta(\log_2 n)$
- $n \log_2 n$ is NOT $\Theta(n^2)$ and n^2 is also NOT $\Theta(n \log_2 n)$

Some interesting properties

Reflexivity

- $f(n)$ is $O(f(n))$. $f(n)$ is also $\Omega(f(n))$. $f(n)$ is also $\Theta(f(n))$.

Some interesting properties

Reflexivity

- $f(n)$ is $O(f(n))$. $f(n)$ is also $\Omega(f(n))$. $f(n)$ is also $\Theta(f(n))$.

Transitivity

- If $f(n)$ is $O(g(n))$ and $g(n)$ is $O(h(n))$ then $f(n)$ is $O(h(n))$
- If $f(n)$ is $\Omega(g(n))$ and $g(n)$ is $\Omega(h(n))$ then $f(n)$ is $\Omega(h(n))$
- If $f(n)$ is $\Theta(g(n))$ and $g(n)$ is $\Theta(h(n))$ then $f(n)$ is $\Theta(h(n))$

Some interesting properties

Reflexivity

- $f(n)$ is $O(f(n))$. $f(n)$ is also $\Omega(f(n))$. $f(n)$ is also $\Theta(f(n))$.

Transitivity

- If $f(n)$ is $O(g(n))$ and $g(n)$ is $O(h(n))$ then $f(n)$ is $O(h(n))$
- If $f(n)$ is $\Omega(g(n))$ and $g(n)$ is $\Omega(h(n))$ then $f(n)$ is $\Omega(h(n))$
- If $f(n)$ is $\Theta(g(n))$ and $g(n)$ is $\Theta(h(n))$ then $f(n)$ is $\Theta(h(n))$

Symmetry

- If $f(n)$ is $\Theta(g(n))$ then $g(n)$ is $\Theta(f(n))$
- Not true for O and Ω

Some interesting properties

Reflexivity

- $f(n)$ is $O(f(n))$. $f(n)$ is also $\Omega(f(n))$. $f(n)$ is also $\Theta(f(n))$.

Transitivity

- If $f(n)$ is $O(g(n))$ and $g(n)$ is $O(h(n))$ then $f(n)$ is $O(h(n))$
- If $f(n)$ is $\Omega(g(n))$ and $g(n)$ is $\Omega(h(n))$ then $f(n)$ is $\Omega(h(n))$
- If $f(n)$ is $\Theta(g(n))$ and $g(n)$ is $\Theta(h(n))$ then $f(n)$ is $\Theta(h(n))$

Symmetry

- If $f(n)$ is $\Theta(g(n))$ then $g(n)$ is $\Theta(f(n))$
- Not true for O and Ω

Transpose Symmetry

- $f(n)$ is $O(g(n))$ if and only if $g(n)$ is $\Omega(f(n))$

A non-property

There are functions $f(n)$ and $g(n)$ which cannot be compared in asymptotic notation

That is neither $f(n)$ is $O(g(n))$ nor $f(n)$ is $\Omega(g(n))$

An example is $n^{1+\sin n}$ since $1 + \sin n$ oscillates between 0 and 2

Asymptotic bounds for polynomial functions

As we shall continue, we shall be interested primarily in finding upper bound on running times (that is the O-notation)

It will be useful to know bounds for some common functions

Asymptotic bounds for polynomial functions

As we shall continue, we shall be interested primarily in finding upper bound on running times (that is the O-notation)
It will be useful to know bounds for some common functions

Polynomial functions: A function is polynomial if it can be written in the form

$$f(n) = a_0 + a_1 n + a_2 n^2 + \cdots + a_n n^d \text{ where } a_d > 0.$$

Asymptotic bounds for polynomial functions

As we shall continue, we shall be interested primarily in finding upper bound on running times (that is the O-notation)
It will be useful to know bounds for some common functions

Polynomial functions: A function is polynomial if it can be written in the form

$$f(n) = a_0 + a_1 n + a_2 n^2 + \cdots + a_n n^d \text{ where } a_d > 0.$$

- The function f is $\Theta(n^d)$ and hence $O(n^d)$

Asymptotic bounds for polynomial functions

As we shall continue, we shall be interested primarily in finding upper bound on running times (that is the O-notation)
It will be useful to know bounds for some common functions

Polynomial functions: A function is polynomial if it can be written in the form

$$f(n) = a_0 + a_1 n + a_2 n^2 + \cdots + a_n n^d \text{ where } a_d > 0.$$

- The function f is $\Theta(n^d)$ and hence $O(n^d)$
- Also, note that f is $O(n^{d+1})$ but not $\Theta(O^{d+1})$

Asymptotic bounds for logarithmic function

Logarithms: Recall $\log_b n$ (logarithm of n to the base b) is the number y such that $b^y = n$

- Base does not matter. That is for each $b > 1$, the function

$$\log_b n \text{ is } \Theta(\log_2 n)$$

From now on, we shall just write $\log n$ and ignore the base

Asymptotic bounds for logarithmic function

Logarithms: Recall $\log_b n$ (logarithm of n to the base b) is the number y such that $b^y = n$

- Base does not matter. That is for each $b > 1$, the function

$$\log_b n \text{ is } \Theta(\log_2 n)$$

  From now on, we shall just write $\log n$ and ignore the base

- For each $x > 0$, the function

$$\log_b n \text{ is } O(n^x)$$

Asymptotic bounds for logarithmic function

Logarithms: Recall $\log_b n$ (logarithm of n to the base b) is the number y such that $b^y = n$

- Base does not matter. That is for each $b > 1$, the function

$$\log_b n \text{ is } \Theta(\log_2 n)$$

  From now on, we shall just write $\log n$ and ignore the base

- For each $x > 0$, the function

$$\log_b n \text{ is } O(n^x)$$

- For each $x > 0$, the function

$$\log_b n \text{ is NOT } \Theta(n^x).$$

  In other words, $\log_b n$ grows much slowly than n^x

Asymptotic bounds for exponential function

Exponential:

- For each $r > 1$ and $x > 0$, the function

$$r^n \text{ is } \Omega(n^x)$$

- For each $r > 1$ and $x > 0$, the function

$$r^n \text{ is NOT } \Theta(n^x).$$

In other words, r^n grows much faster than n^x

Some common running times we shall encounter

By an $O(f(n))$ algorithm, we shall mean an algorithm whose worst-case running time is $O(f(n))$

Some common running times we shall encounter

By an $O(f(n))$ algorithm, we shall mean an algorithm whose worst-case running time is $O(f(n))$

- Linear Time: These are algorithms whose worst case behavior is $O(n)$. Examples: Sequential searching in an array, finding maximum in a list etc ...

Some common running times we shall encounter

By an $O(f(n))$ algorithm, we shall mean an algorithm whose worst-case running time is $O(f(n))$

- Linear Time: These are algorithms whose worst case behavior is $O(n)$. Examples: Sequential searching in an array, finding maximum in a list etc ...
- Quadratic Time: These are algorithms whose worst case behavior is $O(n^2)$. Examples: Insertion sort, selection sort

Some common running times we shall encounter

By an $O(f(n))$ algorithm, we shall mean an algorithm whose worst-case running time is $O(f(n))$

- Linear Time: These are algorithms whose worst case behavior is $O(n)$. Examples: Sequential searching in an array, finding maximum in a list etc ...
- Quadratic Time: These are algorithms whose worst case behavior is $O(n^2)$. Examples: Insertion sort, selection sort
- $O(\log n)$: Example: Binary search

Some common running times we shall encounter

By an $O(f(n))$ algorithm, we shall mean an algorithm whose worst-case running time is $O(f(n))$

- Linear Time: These are algorithms whose worst case behavior is $O(n)$. Examples: Sequential searching in an array, finding maximum in a list etc ...
- Quadratic Time: These are algorithms whose worst case behavior is $O(n^2)$. Examples: Insertion sort, selection sort
- $O(\log n)$: Example: Binary search
- $O(n \log n)$: Example: Merge Sort

Some common running times we shall encounter

By an $O(f(n))$ algorithm, we shall mean an algorithm whose worst-case running time is $O(f(n))$

- Linear Time: These are algorithms whose worst case behavior is $O(n)$. Examples: Sequential searching in an array, finding maximum in a list etc ...
- Quadratic Time: These are algorithms whose worst case behavior is $O(n^2)$. Examples: Insertion sort, selection sort
- $O(\log n)$: Example: Binary search
- $O(n \log n)$: Example: Merge Sort
- Cubic time:These are algorithms whose worst case behavior is $O(n^3)$

Summary

Study asymptotic behavior of running times (Chapter 3 of the book)

$$\begin{array}{ccc} O & \approx & \leq \\ \Omega & \approx & \geq \\ \Theta & \approx & = \end{array}$$

- While we will focus mainly on running times, we can also analyze the extra space used by an algorithm using the same asymptotic notation