CS 209
Data Structures and Mathematical Foundations

02 / 23 / 2024

Instructor: Michael Eckmann

Today’s Topics

- Questions/Comments?
- Proof by induction
- Asymptotic notation

Proof by induction

- Let me introduce
  - the format of a proof by induction,
  - Appropriate situations when you would use this method of proof
  - and we will apply it to
- Sum of i from 1 to n = n*(n-1)/2

Proof by induction

Use Proof by Induction when trying to show that something (a proposition) is true for all positive integers

Base case: simplest value of n (e.g. n = 1)

show that the proposition is true for that simplest value of n

Inductive Step:

assume the proposition is true for n = k (this is called the inductive hypothesis)

show the proposition is true for n = k+1

Once we do the above the proposition has been proven true for all positive integer values of n by induction.

"There exists" and "for all"

- There exists (written as a backwards uppercase E) – means that (at least) one exists
  - If we need to show that there exists something we get to choose one
- For all (written as an upsidedown uppercase A) – means that something is true for ALL values (possibly subject to a constraint, e.g. all n > 1)
  - If we need to show that something is true "for all", we cannot just choose one value and show it is true for that one, we must show it is true **for all** values (subject to whatever constraint)

Asymptotic notation

- Big O – upperbound
- Big Theta – tight bound
- Big Omega – lowerbound
- Defined on next slide

Asymptotic definitions

$f(n)$ is $O(g(n))$ if $\exists$ $c > 0$ and $n_0 > 0$ $\ni$

$$0 \le f(n) \le c\,g(n) \quad \forall\, n \ge n_0.$$

[we say that $g(n)$ is an UPPER BOUND on $f(n)$]

$f(n)$ is $\Theta(g(n))$ iff. $f(n)$ is $O(g(n))$ and
$f(n)$ is $\Omega(g(n))$.

[$g(n)$ is a TIGHT BOUND on $f(n)$]

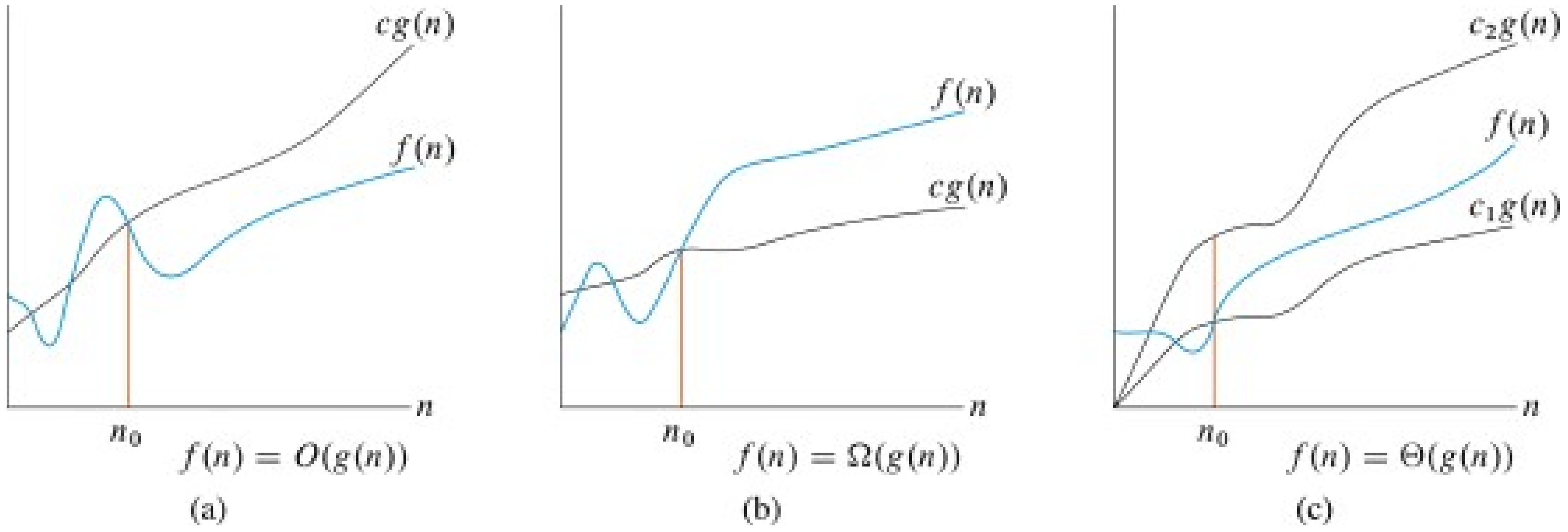
$f(n)$ is $\Omega(g(n))$ if $\exists$ $c > 0$, and $n_0 > 0$ $\ni$

$$0 \le c\,g(n) \le f(n) \quad \forall\, n \ge n_0.$$

[we say that $g(n)$ is a LOWER BOUND on $f(n)$]

Asymptotic notation

- big O
  - When we say f(n) is O(g(n))
  - g(n) is an upper bound on f(n)



Asymptotic notation

- Big O, Big Omega and Big Theta define sets of functions, but we typically say “is” instead of “is in the set of”
- Because we cannot do better than n for findMax, the overall (including best and worst cases) running time of findMax is Big Theta (n) – it is an asymptotically **tight bound**
  - Because we must compare each element to the maxSoFar and since there are n elements we cannot do better than n-1 compares

Asymptotic notation

- Why use asymptotic behavior for efficiency?
  - Ignores small inputs (small values of n) because we may be able to create special purpose algorithms if the input is expected to be small
  - We care only about the efficiency for large n
  - Ignore constants
    - Faster machines can combat constant factors
    - Faster machines cannot combat poor asymptotics

Asymptotic notation

- Let's use the definition of O and Big Omega and Big Theta in an example to determine the asymptotics of some particular function.
- Say f(n) = 3*n + 5
- Is it O(n)?
- Is it BigOmega(n)?
- Is it BigTheta(n)?

Asymptotic notation

- Idea of tight bound vs. not tight bound
- e.g.
- $2*n^2 = O(n^2)$ is asymptotically tight
- $2*n = O(n^2)$ is NOT asymptotically tight (but it is correct to say)
- So, O may or may not be asymptotically tight

Arithmetic Series

Let's prove that the arithmetic sum

1+2+...+n = (n*(n+1))/2

is big $O(n^2)$ on the board (that n squared is an upper bound on the sum)

Let's also prove that it is Big Omega of n^2 (that n^2 is a lower bound on the sum)

Together, they mean that n^2 is an asymptotially tight bound --- that is Big Theta of n^2