Lecture 3

Asymptotic Notation and Recurrence Relations

Tim LaRock
larock.t@northeastern.edu
bit.ly/cs3000sylabus

May 6, 2020

Business

- Piazza is open. Please use it!!
- Email me if you are not in Canvas and/or Piazza at this point.
- Homework 1 is out as of last night. Due Monday night 11:59PM Boston time.
  - There is a mistake in question 4. The condition of the while loop in FancyCounting should read "while more than 1 person is standing". I will update the LaTeX and PDF files after the lecture.

More Business

- TA office hours are being worked out and loaded in to Piazza under Staff → Resources. Thanks for bearing with us.
  - Please email the person whose office hours you plan to attend beforehand to (1) let them know what you would like to talk about and (2) let them know to expect someone in case there is miscommunication about Zoom.
  - You do not need to be super detailed, but it will help us to know in advance what you want to talk about.

Today

- Asymptotic Analysis Notation and Meaning
- Proving Recurrences
- Recursion Trees

$O, o, \Omega, \omega,$ and Θ walk in to a bar...

- Asymptotic analysis is a powerful framework that allows us to reason about many different-but-related things
- Today I will define all of the notations, but keep in mind we will mostly be interested in big-O

Big-O: how big is it really?

- In words: Big-O and little-o notation asymptotically bound functions from above, meaning they refer to *upper bounds* on the asymptotic behavior of the function
- "How big can this function get?"
- Big-O definition:

$$O(g(n)) = \{f(n) : \text{there exist positive constants } c \text{ and } n_0 \text{ such that } 0 \leq f(n) \leq cg(n) \text{ for all } n \geq n_0\}$$

A Note on Notation

$$O(g(n)) = \{f(n) : \text{there exist positive constants } c \text{ and } n_0 \text{ such that } 0 \leq f(n) \leq cg(n) \text{ for all } n \geq n_0\}$$

Note: $O(g(n))$ is a *set*, but we usually don't write $f(n) \in O(g(n))$. Rule of thumb:

- If the asymptotic term is alone on the right hand side of the equation, e.g. $2n^2 = O(n^2)$, the equal sign is equivalent to set membership
- If the asymptotic term appears in the equation, e.g. $T(n) = 2n^2 - O(n)$, the term is a stand in for "some function bounded by $O(n)$."

Let's draw a picture

$$O(g(n)) = \{f(n) : \text{there exist positive constants } c \text{ and } n_0 \text{ such that } 0 \leq f(n) \leq cg(n) \text{ for all } n \geq n_0\}$$

O and o

- The definition of little-o is very similar, but it denotes bounds that are not *tight*

$o(g(n)) = \{f(n)$: for any positive constant c, there exists a constant $n_0 > 0$ such that $0 \leq f(n) \leq cg(n)$ for all $n \geq n_0\}$

What makes a bound "tight"?

- An upper bound is *tight* if it is the smallest function that provides an upper bound
- For example: If a function is bounded by n^2, it is also true that it is bounded by 2^n (related to a homework problem!). 2^n would be a *loose* upper bound to the function.
- We use capital letters (O, Ω, Θ) to denote tight bounds, and lower-case letters (o, ω, θ) to denote bounds that have not been shown to be tight
- We will primarily concern ourselves with tight bounds in this class

O and o

- The definition of little-o is very similar, but it denotes bounds that are not *tight*

$o(g(n)) = \{f(n)$: for any positive constant c, there exists a constant $n_0 > 0$ such that $0 \leq f(n) \leq cg(n)$ for all $n \geq n_0\}$

$$O(g(n)) = \{f(n) : \text{there exist positive constants } c \text{ and } n_0 \text{ such that } 0 \leq f(n) \leq cg(n) \text{ for all } n \geq n_0\}$$

- Intuitively:
  - In a tight upper bound, the function $f(n)$ "follows" or "scales proportionately with" the bounding function $g(n)$
  - In a loose upper bound, the function $f(n)$ is "left behind" because $g(n)$ grows more quickly such that in the infinite limit $\frac{f(n)}{g(n)}$ goes to 0

Quick Question

Is $O(n^2)$ a tight bound for BubbleSort?

Ω and ω

- In words: Ω-notation asymptotically bounds a function from below

$$\Omega(g(n)) = \{f(n) : \text{there exist positive constants } c \text{ and } n_0 \text{ such that } 0 \le cg(n) \le f(n) \text{ for all } n \ge n_0\}$$

$$\omega(g(n)) = \{f(n) : \text{for any positive constant } c\text{, there exists a constant } n_0 > 0 \text{ such that } 0 \le cg(n) \le f(n) \text{ for all } n \ge n_0\}$$

Let's update our picture

$$\Omega(g(n)) = \{f(n) : \text{there exist positive constants } c \text{ and } n_0 \text{ such that } 0 \leq cg(n) \leq f(n) \text{ for all } n \geq n_0\}$$

Θ

- In words: Θ-notation asymptotically bounds a function from above *and* below

$$Theta(g(n)) = \{f(n) : \text{there exist positive constants } c_1,\ c_2, \text{ and } n_0 \text{ such that } 0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \text{ for all } n \geq n_0\}$$

Let's update our picture again

$$\Theta(g(n)) = \{f(n) : \text{there exist positive constants } c_1,\ c_2, \text{ and } n_0 \text{ such that } 0 \le c_1 g(n) \le f(n) \le c_2 g(n) \text{ for all } n \ge n_0\}$$

Summary

- Asymptotic analysis is a powerful and flexible framework for reasoning about functional growth
- Capital symbols O and Ω represent *tight* bounds
  - Tight upper bounds "follow" or "scale proportionately with" the bounding function
  - Loose upper bounds are "left behind" because the bounding function grows more quickly
- We will be almost always interested in the big-O, worst-case runtime of an algorithm

That's it!

- It is my birthday today, my gift to me (and thus you) is a relatively short lecture
- Please take a look at the homework and start asking questions on Piazza!
- Suggested reading for next time: Finish Erickson Chapter 1 (same as yesterday)
- Next time: More Divide and Conquer
  - Proving Recursions
  - Recursion Trees
  - More