CSE332: Data Abstractions

Section 2

HyeIn Kim

Spring 2014

Section Agenda

- Recurrence Relations
- Asymptotic Analysis
- HW1, Project 1 Q&A
- Project 2
  - Introduction
  - Working in a team
  - Testing strategies

Recurrence Relations

Recurrence Relations

- **Recursively defines a Sequence**

  - Example: $T(n) = T(n-1) + 3$, $T(1) = 5$

    ^ Has $T(x)$ in definition

- **Solving Recurrence Relation**

  - Eliminate recursive part in definition

    = Find "Closed Form"

  - Example: $T(n) = 3n + 2$

Recurrence Relations

- **Expansion Method example**

  - Solve $T(n) = T(n-1) + 2n - 1, \quad T(1) = 1$

$$
\begin{aligned}
T(n) &= T(n-1) + 2n - 1 \\
T(n-1) &= T([n-1]-1) + 2[n-1] - 1 \\
&= T(n-2) + 2(n-1) - 1 \\
T(n-2) &= T([n-2]-1) + 2[n-2] - 1 \\
&= T(n-3) + 2(n-2) - 1
\end{aligned}
$$

Recurrence Relations

- **Expansion Method example**

$T(n) \quad = T(n-1) + 2n – 1$

$T(n-1) = T(n-2) + 2(n-1) – 1$

$T(n-2) = T(n-3) + 2(n-2) – 1$

$T(n) \quad = [T(n-2) + 2(n-1) – 1] + 2n – 1$

$\qquad\quad = T(n-2) + 2(n-1) + 2n – 2$

$T(n) \quad = [T(n-3) + 2(n-2) – 1] + 2(n-1) + 2n – 2$

$\qquad\quad = T(n-3) + 2(n-2) + 2(n-1) + 2n – 3$

Recurrence Relations

- **Expansion Method example**

$$T(n) = T(n-1) + 2n - 1$$

$$T(n) = T(n-2) + 2(n-1) + 2n - 2$$

$$T(n) = T(n-3) + 2(n-2) + 2(n-1) + 2n - 3$$

$$\ldots$$

$$T(n) = T(n-k) + [2(n-(k-1)) + \ldots + 2(n-1) + 2n] - k$$

$$= T(n-k) + [2(n-k+1) + \ldots + 2(n-1) + 2n] - k$$

Recurrence Relations

- **Expansion Method example**

$T(n) = T(n-k) + [2(n-k+1) + \ldots + 2(n-1) + 2n] - k$

When expanded all the way down, $T(n-k) = T(1)$

$n-k = 1$, $k = n-1$

$T(n) = T(n-[n-1]) + [2(n-[n-1]+1) + \ldots + 2(n-1) + 2n] - [n-1]$

$= T(1) + [2(2) + \ldots + 2(n-1) + 2n] - n + 1$

Recurrence Relations

- **Expansion Method example**

$$
\begin{aligned}
T(n) &= T(1) + [2(2) + \ldots + 2(n-1) + 2n] - n + 1 \\
&= T(1) + 2[2 + \ldots + (n-1) + n] - n + 1 \\
&= T(1) + 2[(n+1)(n/2) - 1] - n + 1 \\
&= T(1) + (n+1)(n) - 2 - n + 1 \\
&= T(1) + (n^2+n) - n - 1 \\
&= T(1) + n^2 - 1 \\
&= 1 + n^2 - 1 \\
&= n^2
\end{aligned}
$$

Recurrence Relations

- **Expansion Method example** **Check it!**

$T(n) = T(n-1) + 2n - 1, \quad T(1) = 1$

$T(n) = n^2$

$T(1) = 1$	same as 1^2
$T(2) = T(1) + 2(2) - 1 = 4$	same as 2^2
$T(3) = T(2) + 2(3) - 1 = 9$	same as 3^2
$T(4) = T(3) + 2(4) - 1 = 16$	same as 4^2

Recurrence Relations

- **For Homework**

  Remember to <u>show steps!!</u>

  - Correct answer with no steps gets no credit

  a) Show at least 2 expansions of T(n)

  b) At least 2 representations of T(n), using a)

  c) Writing T(n) in terms of k, using b)

  d) Solve for k (show steps!!)

  e) Plug in k and get the closed form

Asymptotic Analysis

Asymptotic Analysis

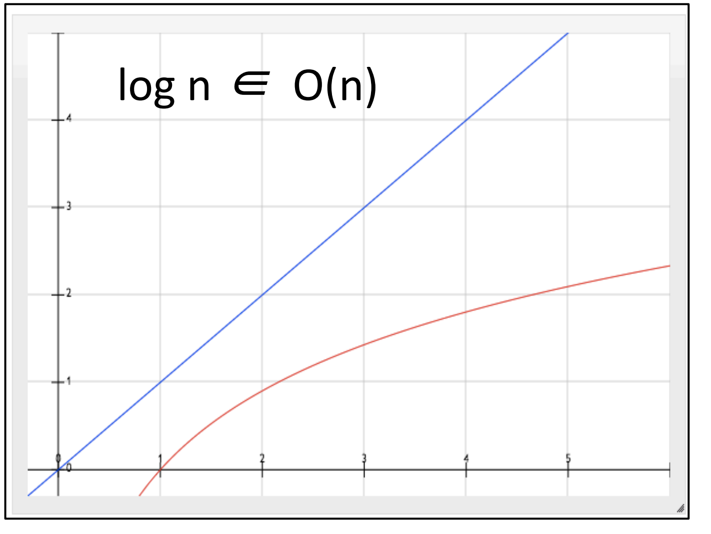
- **Describe Limiting behavior of F(n)**

  - Characterize growth rate of F(n)

  - Use O(g(n)), *Ω*(g(n)), *Θ*(g(n)) for **set** of functions with asymptotic behavior ≤, ≥, ≤ & ≥ to g(n)

- **Upper Bound: O(n)**

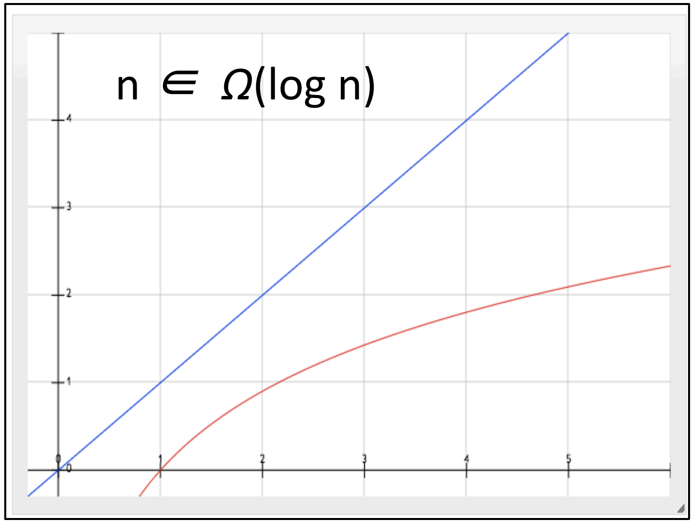
  f(n) $\in$ O(g(n)) if and only if there exist positive constants c and n_0 such that $f(n) \leq c^*g(n)$ for all $n_0 \leq n$



Asymptotic Analysis

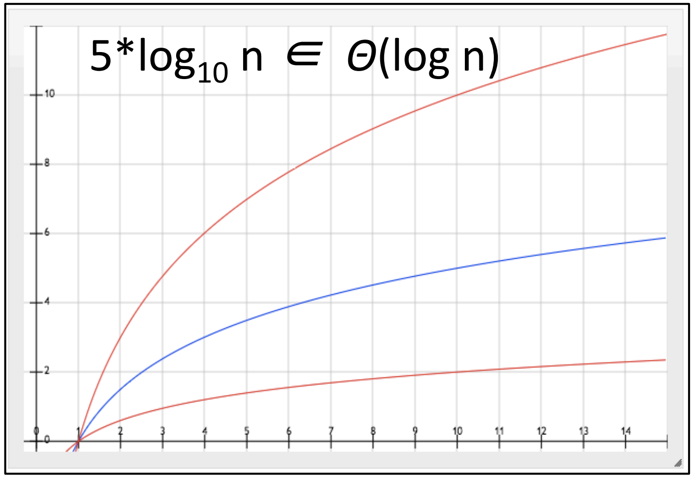
- **Lower Bound: Ω(n)**

  f(n) ∈ Ω(g(n)) if and only
  if there exist positive constants
  c and n_0 such that
  $c*g(n) \leq f(n)$ for all $n_0 \leq n$



- **Tight Bound: Θ(n)**

  f(n) ∈ Θ(g(n)) if and only if
  f(n) ∈ Ω(g(n)) and
  f(n) ∈ O(g(n))



Asymptotic Analysis

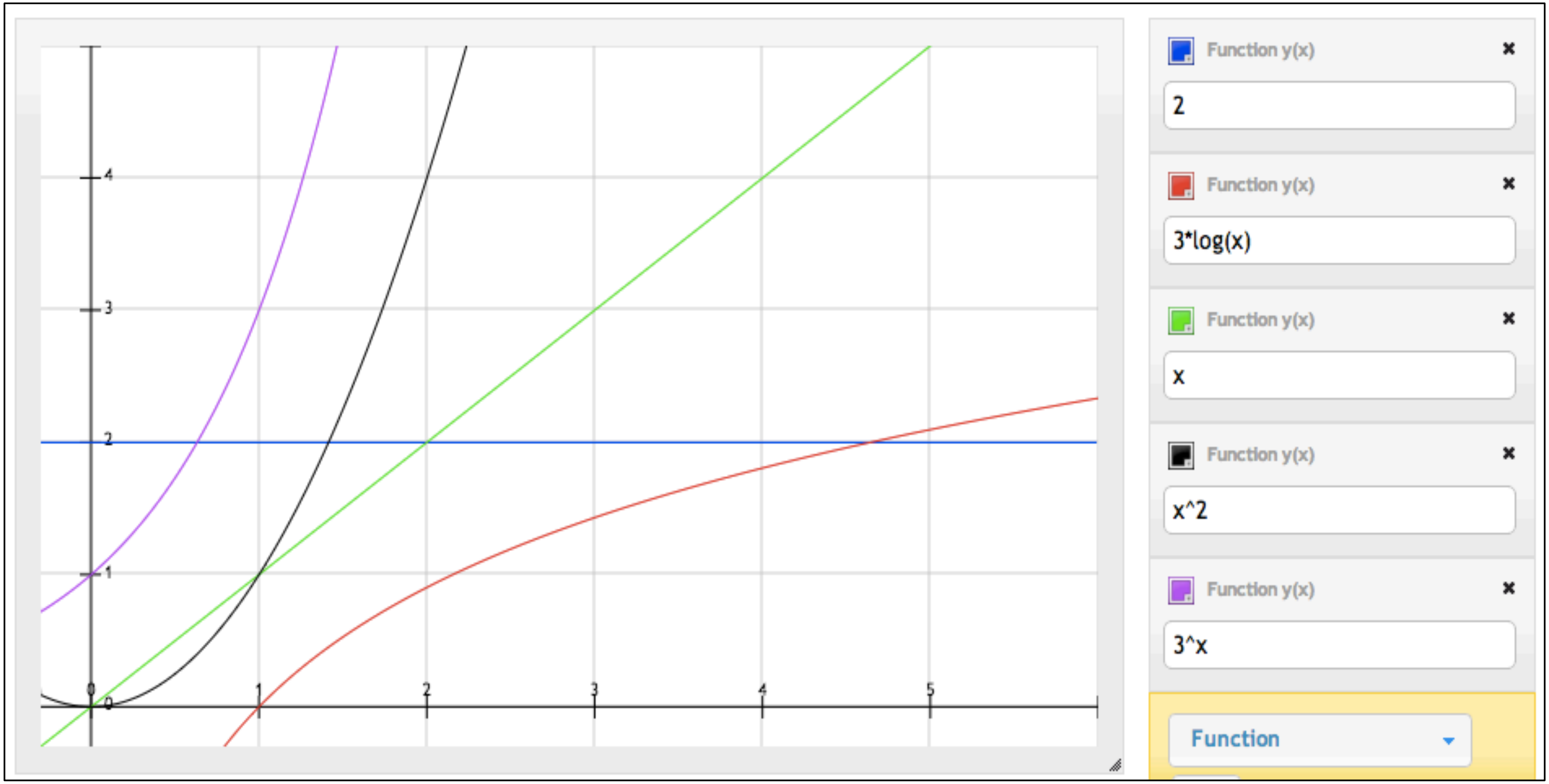
- **Ordering Growth rates (k = constant)**
  - Ignore Low-Order terms & Coefficients

$O(k)$	constant
$O(\log n)$	logarithmic
$O(n)$	linear
$O(n^k)$	polynomial
$O(k^n)$	exponential ($k > 1$)

Increasing Growth rate

Asymptotic Analysis

- **Ordering Growth rates**



Asymptotic Analysis

- **Ordering Growth rates (k, b = constant)**
  - $\log^k n \in O(n^b)$ if $1 < k$ & $0 < b$
  - $n^k \in O(b^n)$ if $0 < k$ & $1 < b$

- **Ordering Example**

$2n^{100} + 10n$	n^{100}	4
$2^{n/100} + 2^{n/270}$	$2^{n/100}$	5
$1000n + \log^8 n$	n	3
$23785n^{1/2}$	$n^{1/2}$	2
$1000 \log^{10} n + 1^{n/300}$	$\log^{10} n$	1

Asymptotic Analysis

- **Proof Example 1: f(n) $\in$ O(g(n))**

  - Prove or disprove nlog n $\in$ O(3n)

nlog n $\in$ O(3n), then by definition of Big-O
nlog n $\leq$ c*(3n), for $0 < c$ && $0 < n_0 \leq n$
(1/3)log n $\leq$ c

but as $n \rightarrow \infty$, $\log n \rightarrow \infty$
Finite constant c always greater than log n cannot exist, no matter what n_0 we choose

nlog n $\notin$ O(3n)

Asymptotic Analysis

- **Proof Example 2**
  - Prove or disprove
    If $f(n) \in O(g(n))$ and $h(n) \in O(k(n))$,
    then $f(n) + h(n) \in O(g(n) + k(n))$

Homework 1 Q&A

- **Solutions are NOT posted**

  - Any questions?

Project 2

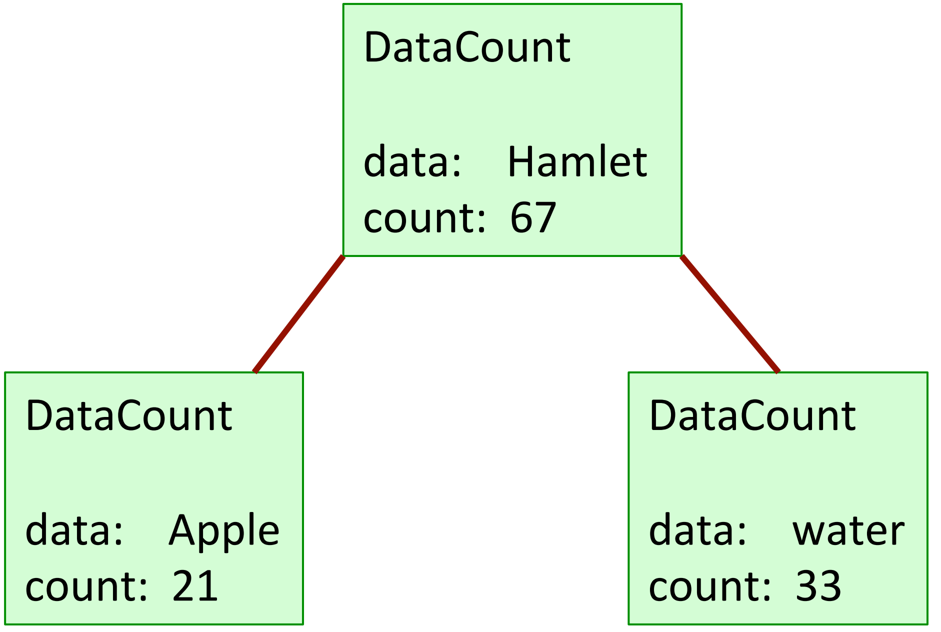
shake-n-bacon

Project 2

- **Word Frequency Analysis**

- **Phase A:** Implement 3 ADTs
  - Due about 2 weeks from now (Thursday April 24th)
  - Word frequency analysis using different DataCounters

- **AVLTree**
- **MoveToFrontList**
- **FourHeap**
- **Heap Sort**



Project 2 – Find Partner

- **Form a 2 person team**
- **Not required, but greatly encouraged**

  - Learn from each other
  - Check each other's style (large fraction of grade!)
  - Collaborate working experience
    (You can put it on your resume!)

- **Many had hard time meeting deadline**

  - Don't be spoiled by project 1!

Project 2 – Find Partner

- **Form a 2 person team**

  - Use discussion board to find partner

    https://catalyst.uw.edu/gopost/area/rea2000/129662

  - Complete catalyst survey to form team
    by next Friday April 18th
    (Only one survey for a team!)
    https://catalyst.uw.edu/webq/survey/kainby87/231804

  - Anyone wants a partner but don’t have one yet?

Project 2 – Sharing Code

- **Version Control System**
  - Git, Mercurial, SVN, CVS, Perforce...
  - Lots of choices (Use whatever you want):
    http://en.wikipedia.org/wiki/Comparison_of_revision_control_software

- **Some posts to help choosing one**
  - Choosing a Version Control: Beginners Tour
    http://www.codeproject.com/Articles/431125/Choosing-a-Version-Control-System-A-Beginners-Tour
  - Review of 7 Version Control Systems
    http://www.smashingmagazine.com/2008/09/18/the-top-7-open-source-version-control-systems/
  - What is your favorite version control system?
    http://programmers.stackexchange.com/questions/940/what-are-your-favorite-version-control-systems

Project 2 – Sharing Code

- **Repository:** Where you store your code

  - Your machine
  - Shared directory in attu (you need to request one)
  - Web based repository (should be private!)

  Bitbucket: https://bitbucket.org/
  GitHub: https://github.com/

Project 2 – Sharing Code

- **Useful tutorials**

  - Bitbucket 101
    https://confluence.atlassian.com/display/BITBUCKET/Bitbucket+101

  - Git
    https://www.atlassian.com/git?utm_source=cac-bitbucket-1&utm_medium=banner&utm_content=visual-git-guides&utm_campaign=git-tutorial

  - EGit (Git with Eclipse)
    http://www.vogella.com/tutorials/EclipseGit/article.html
    http://wiki.eclipse.org/EGit/User_Guide

- **Easy & Quick Code Sharing**

  - If you don't want to bother having version control
    http://www.smashingapps.com/2011/05/25/ten-best-collaborative-sites-for-quick-code-sharing.html