Analysis of Algorithms - CS 323
Lecture #1 – 02/10/16

Notes by: Henry Shum

Homework Set #1 Review:

1) Define the Upper and Lower Bound

$$n^2 < 1^2 + 2^2 + 3^2 + \ldots + n^2 < n^3$$
$$an^3 + bn^2 + cn + d$$

n=0 $\quad a0^3 + b0^2 + c0 + d = 0$
n=1 $\quad a1^3 + b1^2 + c1 = a + b + c = 1$
n=2 $\quad a2^3 + b2^2 + c2^1 = 8a + 4b + 2c = 5$
n=3 $\quad a3^3 + b3^2 + c3^1 = 27a + 9b + 3c = 14$

Find a common multiple to cancel out the c's for (n=2) – (n=1)
$6a + 2b = 3$
Find a common multiple to cancel out the c's for (n=3) – (n=1)
$24a + 6b = 11$
Find a common multiple to cancel out the b's of
$24a + 6b = 11$ **(1)**
$(6a + 2b = 3) \times 3 \rightarrow 18a + 6b = 9$ **(2)**

$(1) - (2) = 6a + 0b = 2 \rightarrow 6a = 2 \rightarrow a = 1/3$

$6(1/3) + 2b = 3 \rightarrow 2b = 3-2 = 1 \rightarrow b = 1/2$

$1/3 + 1/2 + c = 1 \rightarrow c = 1 - 1/2 - 1/3 = 1/6$

$$n^3/3 + n^2/2 + n/6 = \frac{n(n+1)(2n+1)}{6}$$

2) Base case
   n = 0

Inductive hypothesis:
Assume $1^2 + 2^2 + 3^2 + \ldots + k^2 = \frac{k(k+1)(2k+1)}{6}$
Prove $1^2 + 2^2 + 3^2 + \ldots + k^2 + (k+1)^2 = \frac{k(k+1)(2k+3)}{6}$
Simply:
$$\frac{k(k+1)(2k+1)}{6} + \frac{6(k+1)^2}{6} \rightarrow \frac{(k+1)}{6}\left[k(2k+1) + 6(k+1)\right]$$
$$= \frac{k+1}{6}(2k^2 + k + 6k + 6) = \frac{k+1}{6}\left[(k+2)(2k+3)\right]$$

3) $a + (a+d) + (a+2d) + \ldots + (a+nd)$

add all of the a's: $a(n+1) + d[1+2+\ldots n]$

$$\frac{n(n+1)}{2}$$

$$(n+1)\left[a + \frac{dn}{2}\right]$$

4) $g(n) = c + cr + cr^2 + \ldots cr^n$ telescoping!

$rg(n) = cr + cr^2 + \ldots + cr^{n+1}$

$rg(n) - g(n) = cr^{n+1} - c$

$$g(n) = \frac{cr^{n+1} - c}{r-1} = c\left[\frac{r^{n+1}-1}{r-1}\right]$$

5)

$p(n) = an^3 + bn^2 + cn + d$

0	$p(0) = 0 = d$
1	$p(1) = 1 = a + b + c$
1	$p(2) = 1 = 8a + 4b + 2c$
2	$p(3) = 2 = 27a + 9b + 3c$
	$p(4) = 6$

$a = 1/3$ $b = -3/2$ $c = 13/6$

$p(n) = n^3/3 + 3n^2/2 + 13n/6$

6) Ranking Functions Brake them into groups!

$(n \log n)^2$
$\log \log n$
$n^{\sqrt{2}}$
2^n
$(\log n)^2$
$n!$
$2^{2^\wedge n}$
n^e
e^n
$2^{\ln n}$
$\sqrt{2}^n$

logarithmic	polynomial	exponential	
1) $\log \log n$	5) $(n \log n)^2$	8) 2^n	10) $n!$
2) $(\log n)^2$	4) $n^{\sqrt{2}}$	9) e^n	11) $2^{2^\wedge n}$
	6) n^e	7) $\sqrt{2}^n$	
	3) $n^{\ln 2} = 2^{\ln n}$		

$(n \log n)^2 \to n^2 \log^2 n = n^2 \log n \log n$

$2^{\ln n} = \text{Log}_2 (2^{\ln n}) \to \ln n \cancel{\text{Log}_2 2} = 1 \to 2^{\ln n}$

Solving $2^{\ln n}$:

$\log z\,(a)^{\log z\,(b)} = \log z\,(b)^{\log z\,(a)}$

Take the log on both sides -> $(\log z\,(b)) \log z\,(a) = (\log z\,(a)) \log z\,(b)$

$2^{\ln n} = n^{\ln 2}$ (its between 0 and 1)

$\cancel{n^2}$-log n log n *or* $\cancel{n^2}\, n^{.73}$

$\sqrt{1{,}000{,}000}$ $\log_{10} 1{,}000{,}000 = 6$

1,000

$\log_2 2^{2^\wedge n} = 2^n$

7) $f(n) = o(g(n))$

$h(n) = g(n)/f(n)$ -> $\log (g(n)/f(n)) \times f(n)$ -> $\log \log (g(n)/f(n)) \times f(n)$

These functions are asymptotic smaller than the previous one but still bigger than f(n).

<u>Lecture 2:</u>

Time Complexity
Space Complexity

Best Case (Does not give the whole picture of the problem)
Average Case (All possible sets of good and bad cases)
Worst Case (Reach the upper bound of a problem)

Fibonacci Sequence:
$f(0) = 0$
$f(1) = 1$
$f(n) = f(n-1) + f(n-2)$ ← not efficient, it computes a previous computed function (2^n)

We need O(n), bottom up or memory table

Characteristic Equation

$F_n - f_{n-1} - f_{n-2} = 0$
$X^n - X^{n-1} - X^{n-2} = 0$
$X^{n-2} (X^2 - X - 1) = 0$ → quadratic equation - $ax^2 + bx + c = 0$ (2 solutions)

$x = \frac{-b+-\sqrt{b^2-4ac}}{2a}$ -> $\frac{1+-\sqrt{(-1)^2-4(1)(-1)}}{2(1)}$ -> $\frac{1=-\sqrt{5}}{2}$ (Linear Homogenous Equation)

$p\left(\frac{1+\sqrt{5}}{2}\right)^n + q\left(\frac{1-\sqrt{5}}{2}\right)^n$ (Closed Form Fibonacci)

Base Case = 0

$f(0) = 0 = n{=}0 \quad p\left(\frac{1+\sqrt{5}}{2}\right)^0 + p\left(\frac{1+\sqrt{5}}{2}\right)^0 = 0 \rightarrow p+q = 0 \rightarrow p = -q$

$f(1) = 1 \qquad p\left(\frac{1+\sqrt{5}}{2}\right)^1 + p\left(\frac{1+\sqrt{5}}{2}\right)^1 = 1$

$$\frac{p + p\sqrt{5})}{2} + \frac{\overset{-p}{\cancel{q}} - \overset{p}{\cancel{q}}\sqrt{5})}{2} = 1$$

$$\frac{2p\sqrt{5}}{2} = 1 \qquad p = 1/\sqrt{5} \qquad q = -1/\sqrt{5}$$

$f(n) = \frac{1}{\sqrt{5}}\left(\frac{1+\sqrt{5}}{2}\right)^n - \frac{1}{\sqrt{5}}\left(\frac{1-\sqrt{5}}{2}\right)^n$ ← Dominant Term, as **n** goes to infinity goes to 0

produces Fibonacci seq.

$1.6^n \rightarrow (1.6^x)^{n/x}$

Make base 10

$\log_{1.6}10 = \frac{\log_{10}10}{\log_{10}1.6} \rightarrow x{=}4.9$

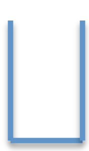
$f(n) \approx 10^{n/5}$

Reviewing Data Structures:

Abstract Data Type (ADT) – Not real

Interface - contract between implementer and promoter.

Stack – is an abstract data type, what does it offer? What does it have?

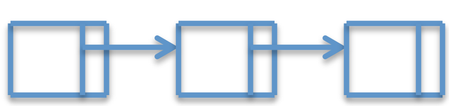


Operations that a stack support

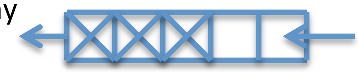
Push, Pop, top (), isFull, isEmpty, numberofElements, capacity,

Queue-

Linked List – pointer to the next , Contains: (enqueue, dequeue, isEmpty, isFull,peek)



Array



-n operations, end of array is running out of space
-an element deletion in the middle is a problem

Circular implementation, maintains pointers at the beginning and end

List

Insert(key,postion)	getkeys
Delete(key)	sort
Delete(position)	isEmpty
Iteration	isFull
next	numberofElements
prev	search
	swap/arrange

<u>Different Types of Trees</u>

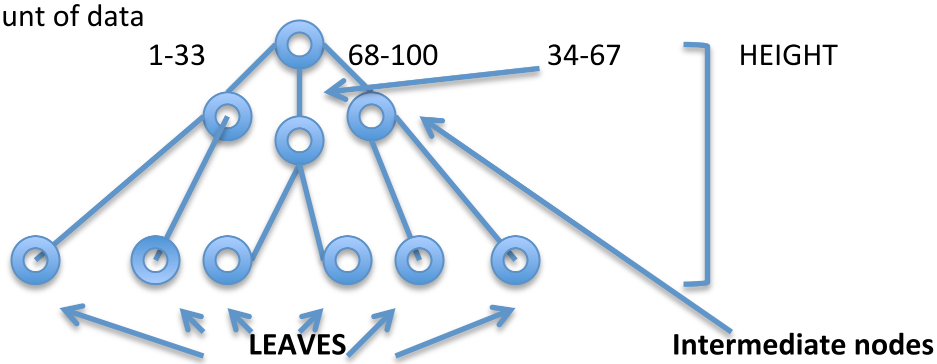
Trees	Node	root
Binary trees	(vertex)	
k-ary trees	Children	

Binary trees (node can have up to 2 children)
K-ary trees (node can have n amount of children)

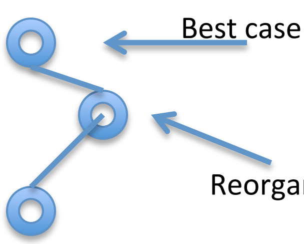
Dictionary - where we look up things

<u>Worst Case</u>

N amount of data
$\log_k n$



A snake



Rotation – move things around to form a complete tree.

<u>Traversal</u> – Getting data in certain order
-depth first search (dfs) – look at root to one child then to root and look at other child
-breadth first search (bfs) – level order searching.

Use a queue to implement the order.
-preorder – root, preorder(left), preorder(right)
-postorder – postorder(left), postorder(right), root
-inorder – inorder(left), root, inorder(right)
-reverse order – reverseorder(right), root, reverseorder(left)

<u>Time complexity</u>
Upper Bound? Worst case = n elements and (n-1) for each individual $\approx n^2$
Lower Bound? n = elements

-A optimal time for a tree is Log n or less

How to get a tree to log n?

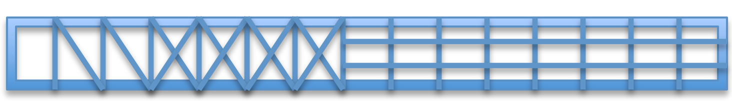
<u>Heap (priority queue)</u>
-A type of binary tree
-Children will be bigger than it's parents
-Look up the smallest element
-Bubble Up / Bubble Down

-delete-min (deletes smallest element)
-delete-max (deletes largest element)

<u>Implementation of trees</u>
-Linked List

-*Array (Cons)* – space is wasted, has to be a full balanced tree, and the kind of data we are working with.

2^L L=level



root