Date: March 9, 2016
Instructor: Professor Lawrence Teitelman

Analysis of Algorithms – CS 700/32
Lecture#6 – March 9 2016
Notes by: Yuqian Zhang

Section 1: Homework

3/9. Q4 $T(1)=1$

$T(n)=T(\sqrt{n})+1$

$n=2^{2^k}$ $\sqrt{n}=2^{2^k/2}=2^{2^{k-1}}$

$T(n)=T(\sqrt{n})+1=T(2^{2^{k-1}})+1$ key to the problem

$S(k)=S(k-1)+1$

$S(1)=T(\sqrt{1})+1=2$

$$\left.\begin{aligned} S(k)-S(k-1)&=1\\ &\vdots\\ S(2)-S(1)&=1\\ S(1)-S(0)&=1 \end{aligned}\right\}\quad \begin{aligned} S(k)-S(1)&=k\\ S(k)&=k+c \end{aligned}$$

$T(n)=S(k)=\log\log n+\square$

Section 2:
Huffman Coding

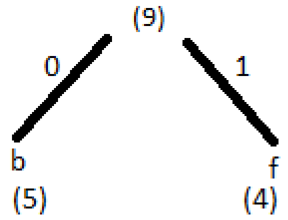
It is one of data compression, but the best compression.
It is an example of greedy algorithm (like **Fraction Knapsack** mentioned before)
Greedy algorithms is where you make locally optimal choice in global optimal choice.
(Uses small grained, or local minimal/maximal choices in attempt to result in a global minimum/maximum. At each step, the algorithm makes the near choice that appears to lead toward the goal in the long-term.)

If you want to represent a letter how much space need?
Old way→ASCII 8 bits
But all things take 8 bits, can we do better than that?
Yes→find relative frequency

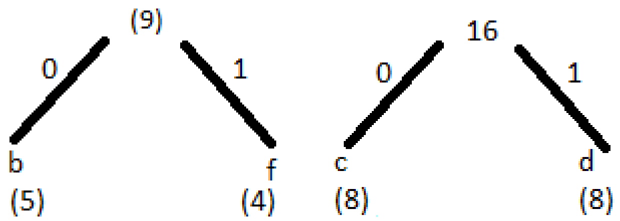
Date: March 9, 2016
Instructor: Professor Lawrence Teitelman

Example: in the text there are 100 characters, and only contains following letters

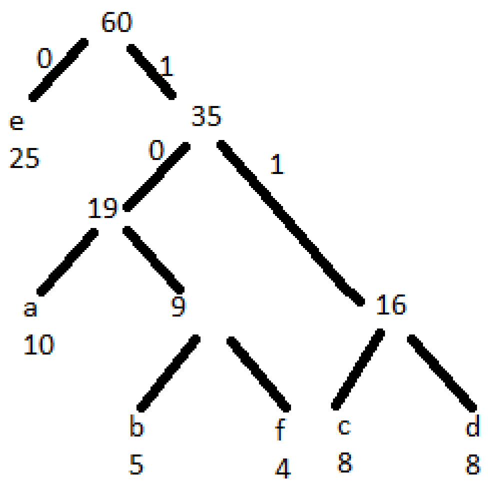
a: 10
b: 5
c: 8 [take 2 smallest and combine them]→
d: 8
e: 25
f: 4



a:10
9 [continue to take 2 smallest]→
c:8
d:8
e:25



……finally we got:



So, if we want to encode 'badbed', it is:
(The length of encoding would be the depth of the leaf)
b- 1010
a- 100
d- 111
b- 1010
e- 0
d- 111

Comparing to ASCII, which using fixed 8 bits to check, Huffman Tree do not implement fixed length. When you are out of leaf, you are in the end of character. The beauty of Huffman coding is no 2 can have 2 different codings have the same prefix and 1 has something more. Thus, that allows you to decode compressed text. (About space, just take it as one of relative frequency)

Most greedy part: taking nest 2 smallest numbers on the list and combine them into sum and go into the bottom.

Q: How does it work (implement) in real coding?
The data structure: **heap**. (Always gets the min on the top) [O(nlgn)]

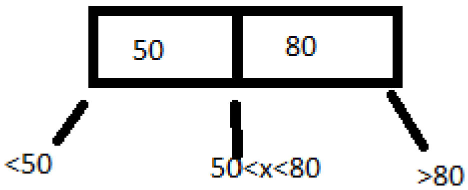
Date: March 9, 2016
Instructor: Professor Lawrence Teitelman

Binary Search Tree

- ### 2-3 Tree

  Hopcraft 1970   O(logn)

  A 2–3 tree is a tree data structure, where every node with children (internal node) has either two children (2-node) and one data element or three children (3-nodes) and two data elements.



- ### AVL

  AV= Adelson- Velsky

  L = Landis   1967

  Abs(height of (leftSubTree)-height of (RightSubTree))<= 1

  It keep tracking the node height

- ### Red-Black

  Guibas Sedgewick 1978

  Not perfect balanced, but balanced enough, not breaking (lgn)

- ### B-tree

  Bayer McCreight 1972

  A large amount of data

- ### B+ tree

  Like B- tree, but has more children

- ### Trie (Tree/Try)

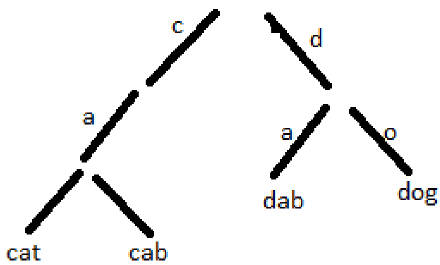
  R.de la Brandais 1959

  Also called prefix-tree

  Example: cat cab dog dab

  Time O(m)

  Maximum length of word



- ### Splay Tree

  A splay tree is a self-adjusting binary search tree with the additional property that recently accessed elements are quick to access again. It performs basic operations such as insertion, look-up and removal in O(log n) amortized time.

  Bring what you'll searching for (close) to the root/top

Date: March 9, 2016
Instructor: Professor Lawrence Teitelman

Section 3: Review

In Order

Sequence

series – arithmetic series ~~a+ad+2ad+3ad~~

$a+(a+d)+(a+2d)+\cdots(a+nd) = (n+1)a+\frac{n(n+1)}{2}d$

$1+2+3+\cdots+n = \frac{n(n+1)}{2}$

– geometric series $c+cr+cr^2+cr^3+\cdots+cr^n = c\left[\frac{r^{n+1}-1}{r-1}\right]$

Date: March 9, 2016
Instructor: Professor Lawrence Teitelman

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

$1^3 + 2^3 + 3^3 + \cdots n^3 = \left[\frac{n(n+1)}{2}\right]^2$

$an^4 + bn^3 + cn^2 + dn + e$

Proof by Induction

base case

inductive hypothesis

inductive step $P(k) \to P(k+1)$

harmonic series $1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n}$

Telescoping

$T(n) - T(n-1) = \boxed{}$ $\qquad \frac{T(n)}{T(n-1)} = \boxed{}$

$T(n-1) - T(n-2) = \cdots$

$\vdots \qquad \vdots$

$T(1) - T(0) = \cdots$

Domain Transformation $T(n) = T(\frac{n}{2}) + \cdots$

$n = 2^k \quad \frac{n}{2} = 2^{k-1}$

$S(k) = S(k-1) + \cdots$

Range Transformation $\frac{T(n)}{3^n} = \frac{3T(n-1)}{3^n} + \frac{\cdots}{3^n}$

$R(n) = R(n-1) + \cdots$

$R(n) = \frac{T(n)}{n}$ OR $R(n) = \frac{T(n)}{n+1}$

$R(n+1) = \frac{T(n+1)}{n+1}$ $\qquad R(n+1) = \frac{T(n+1)}{n+2}$

Expansion

$T(n) = T(\frac{n}{2}) + 1 = (T(\frac{n}{4}) + 1) + 1 = T(\frac{n}{8}) + 3 = T(\frac{n}{16}) + 4$

$= T(1) + \log_2 n$

Date: March 9, 2016
Instructor: Professor Lawrence Teitelman

linear Homogenous Recurrence Equations

$f(0) = f(1) = 1$

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

$x^n = x^{n-1} + x^{n-2}$ $\quad ax_1^n + bx_2^n$

$x^2 = x + 1 \rightarrow$ got 2 unique solutions

$3^n \approx 10^{n/2} = (\sqrt{10})^n$ estimation

$\approx 10^{\frac{n}{\log_3 10}}$ or $10^{n \lg_{10} 3}$

data structures:

stack. (has limit access to the data)

Queue (enQ deQ)

List

Tree $\begin{cases} \text{BST} \\ \text{Heap (Min/Max Heap)} \end{cases}$

(priorityQ)

Find minimum linear $O(n)$ Binary $O(\lg n)$

Selection problem 3 Ways

① presorting → find i^{th} smallest go to position i.

② b) $i\lg n + O(n)$ ← build heap — partial sort

② a). selection sort → find i smallest

③ half quickSort. sorting both sides of pivot

~~lgn Avg~~

Avg → $\Theta(n)$

Worst → $\Theta(n^2)$

median of 5s.

$\Theta(n)$

Sorting

BubbleSort n^2

SelectionSort. n^2

InsertionSort $n^2 \rightarrow$ linear

MergeSort $n\lg n$

QuickSort ~~lgn + O(n)~~ $n\lg n$, n^2 worst

HeapSort $n\lg n$

Date: March 9, 2016
Instructor: Professor Lawrence Teitelman

Each outcome should be in 1 leaf. There are n^2 outcomes cause there are n! leaves.

$n!$ leaves.

each outcome should be in 1 leaf. n^2 outcomes

comparision-based is at least $\boxed{n \lg n}$

Non-comparision-base Sort

Counting Sort → small fixed-range data

Bucket Sort → depends on # of buckets

Radix sort → digit could be base n in element

Master Therom. 3 cases might be useful in test