MA/CSSE 473 Day 30

Optimal BSTs

MA/CSSE 473 Day 30

Assignment	Old due date	New due date
10	Tuesday, Oct 19	Wednesday, Oct 20
Convex Hull	Thursday, Oct 21	Friday, Oct 22
11	Saturday, Oct 23	Tuesday, Oct 26
12	Tuesday, Oct 26	Thursday, Oct 28
13	Thursday, Oct 28	Wednesday, Nov 3

REMINDER: You may NOT use a late day for HW 12
But you may earn a late day by submitting early.

- **Take-home exam** available by Oct 29 (Friday) at 9:55 AM, due Nov 1 (Monday) at 8 AM.
- **Student Questions**
- Optimal Linked Lists
- Expected Lookup time in a Binary Tree
- Optimal Binary Tree (preview)

Warmup: Optimal linked list order

- Suppose we have n distinct data items $x_1, x_2, \ldots, x_n$ in a linked list.
- Also suppose that we know the probabilities $p_1, p_2, \ldots, p_n$ that each of the items is the one we'll be searching for.
- What is the expected number of probes before a successful search completes?
- How can we minimize this number?
- What about an unsuccessful search?

Q1-2

Examples

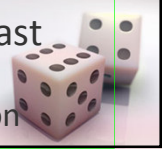
- $p_i = 1/n$ for each i.
  - What is he expected number of probes?
- $p_1 = ½, p_2 = ¼, \ldots, p_{n-1} = 1/2^{n-1}, p_n = 1/2^{n-1}$
  - expected number of probes:

$$\sum_{i=1}^{n-1} \frac{i}{2^i} + \frac{n}{2^{n-1}} = 2 - \frac{1}{2^{n-1}} < 2$$

- What if the same items are placed into the list in the opposite order?

$$\sum_{i=2}^{n} \frac{i}{2^{n+1-i}} + \frac{1}{2^{n-1}} = n - 1 + \frac{1}{2^{n-1}}$$

- The next slide shows the evaluation of the last two summations in Maple.
  - Good practice for you? prove them by inducttion



Calculations for previous slide

```
> sum(i/2^i, i=1..n-1)+ n/2^(n-1);
```

$$-2\left(\frac{1}{2}\right)^{n} n-2\left(\frac{1}{2}\right)^{n}+2+\frac{n}{2^{(n-1)}}$$

```
> simplify(%);
```

$$-2^{(1-n)}+2$$

```
> sum(i/2^(n+1-i), i=2..n) + 1/2^(n-1);
```

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

What if we don't know the probabilities?

1. Sort the list so we can at least improve the average time for unsuccessful search
2. Self-organizing list:
   - Elements accessed more frequently move toward the front of the table; elements accessed less frequently toward the back.
   - Strategies:
     - Move Ahead One Position (interchange with previous element)
     - Interchange with first element
     - Move to Front (only efficient if the list is a linked list)

Q3

Optimal Binary Search Trees

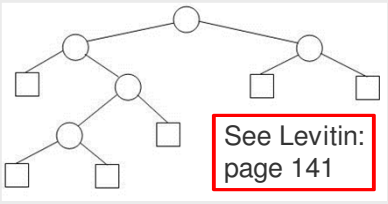
- Suppose we have n distinct data items $x_1, x_2, \ldots, x_n$ (in increasing order) that we wish to arrange into a Binary Search Tree
- This time the expected number of probes for a successful or unsuccessful search depends on the shape of the tree and where the search ends up
- Let y be the value we are searching for
- For i= 1, …,n, let p_i be the probability that y is item x_i
- For i= 1, …,n-1, let q_i be the probability that $x_i < y < x_{i+1}$
- Similarly, let q_0 be the probability that $y < x_1$, and q_n the probability that $y > x_n$
- Note that $\sum_{i=1}^{n} p_i + \sum_{i=0}^{n} q_i = 1$

  but we can also just use frequencies when finding the optimal tree (and divide by their sum to get the probabilities if needed)

Q4

Recap: Extended binary search tree

- It's simplest to describe this problem in terms of an **extended binary search tree** (EBST): a BST enhanced by drawing "external nodes" in place of all of the null pointers in the original tree



See Levitin: page 141

- Formally, an Extended Binary Tree (EBT) is either
  - an external node, or
  - an (internal) root node and two EBTs T_L and T_R
- In diagram, Circles = internal nodes, squares = external nodes
- It's an alternative way of viewing a binary tree
- The external nodes stand for places where an unsuccessful search can end or where an element can be inserted
- An EBT with n internal nodes has ___ external nodes (Prove this by induction– a review from 230)

Q5

What contributes to the expected number of probes?

- Frequencies, depth of node
- For successful search, number of probes is one more than depth of the corresponding internal node
- For unsuccessful, number of probes is equal to depth of the corresponding external node

Q6-7

How many possible BST's

- Given distinct items $x_1 < x_2 < \ldots < x_n$, how many different Binary Search Trees can be constructed from these values?
- Figure it out for n=2, 3, 4, 5
- Write the recurrence relation
- Solution is the **Catalan number** c(n)

$$c(n) = \binom{2n}{n}\frac{1}{n+1} = \frac{(2n)!}{n!(n+1)!} \approx \frac{4^n}{n^{3/2}\sqrt{\pi}}$$

- Verify for n = 2, 3, 4, 5

Q8-14