MA/CSSE 473   Day 28 (and 29?)

Optimal Binary Search trees.

1. Optimal linked list order (if we know the probability of search for each item)
    a. Item x_i in list has probability p_i. What is expected number of probes for search?

    b. Example: $p_1 = ½$, $p_2 = ¼$, …, $p_{n-1} = 1/2^{n-1}$, $p_n = 1/2^{n-1}$
    Expected # of probes for best case, worst case:

    c. What if we do not know the probabilities?

2. Optimal binary search tree (for case where we know the probabilities (or frequencies)
    a. For today, we only deal with successful searches.
    b. If $P(x_i) = p_i$, what is the expected number of probes for a search?

    c. Guiding principle for optimization

3. **Example:** consider only successful searches, with probabilities A(0.2), B(0.3), C(0.1), D(0.4).

worst   opposite   greedy

better   best?

4. Should we try exhaustive search of all possible BSTs? How many are there?

n=2 n=3

n=4 n = 5

5. Write the recurrence relation, apply it to n=5 case

6. Solution of this recurrence:

7. Formally, an Extended Binary Tree (EBT) is either
   a. an external node, or
   b. an (internal) root node and two EBTs T_L and T_R
8. The external nodes stand for places where an unsuccessful search can end or where an element can be inserted
9. An EBT with n internal nodes has n+1 external nodes. (We proved this by induction earlier in the term)
10. For successful search, number of probes is one more than the depth of the corresponding internal node.
11. For unsuccessful, number of probes is equal to the depth of the corresponding external node.
12. Optimal BST notation:
    a. Keys are $K_1, K_2, \ldots, K_n$
    b. Let v be the value we are searching for
    c. For i= 1, …,n, let a_i be the probability that v is key K_i
    d. For i= 1, …,n-1, let b_i be the probability that $K_i < v < K_{i+1}$
    e. Similarly, let b_0 be the probability that $v < K_1$,
       and b_n the probability that $v > K_n$
       $$\sum_{i=1}^{n} a_i + \sum_{i=0}^{n} b_i = 1$$
    f. We can also just use *frequencies* instead of *probabilities* when finding the optimal tree (and divide by their sum to get the probabilities if we ever need them). That is what we will do.

13. We want to minimize weighted path length,

$$C(T) = \sum_{i=1}^{n} a_i[1 + depth(x_i)] + \sum_{i=0}^{n} b_i[depth(y_i)]$$

14. You will show by induction (HW 12) that C(T) can be calculated by the recursive formula
    - C(empty EBT) = 0,
    - If T has a root and two subtrees T_L and T_R, $C(T) = C(T_L) + C(T_R) + \Sigma a_i + \Sigma b_i$,
    - where the summations are over all a_i and b_i for nodes in T

15. Consider these Frequencies of vowel occurrence in English A, E, I, O, U

a's: 32, 42, 26, 32, 12

b's: 0, 34, 38, 58, 95, 21

3. Consider these Frequencies of vowel occurrence in English A, E, I, O, U

a's: 32, 42, 26, 32, 12

b's: 0, 34, 38, 58, 95, 21

4. Define the quantities W_{ij} that help with the calculation of the C_{ij}.

5. R_{ij} (an integer) is the index of the best key to use as a root of the optimal tree.

   It is the value of k that minimizes

16. What is the running time of the optimalBST algorithm, as a function of the number of keys?