CSCE 500 Midterm Exam #2

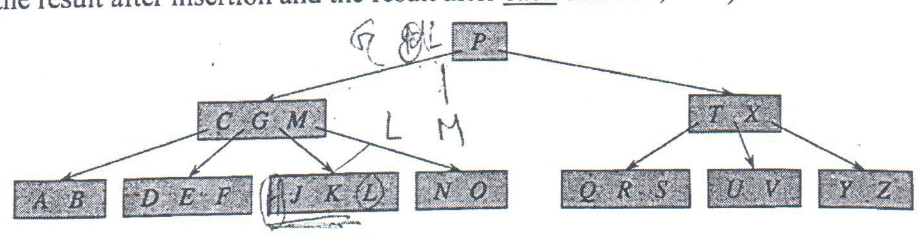
11/4/2015

1. The binary search tree (T) facilitates key search and it involves several operations to maintain the tree property when a node (z) is deleted, as shown in the following pseudo code, TREE-DELETE(T, z), where TRANSPLANT(T, u, v) replaces the subtree rooted at u with one rooted at v. Fill in the two missing statements in the pseudo code. (10%)

```
TREE-DELETE(T, z)
if z.left == NIL
    TRANSPLANT(T, z, z.right)        // z has no left child
elseif z.right == NIL
    TRANSPLANT(T, z, z.left)         // z has just a left child
else // z has two children.
    y = TREE-MINIMUM(z.right)        // y is z's successor
    if y.p ≠ z
        // y lies within z's right subtree but is not the root of this subtree.
        TRANSPLANT(T, y, y.right)
        1.
        2.
    // Replace z by y.
    TRANSPLANT(T, z, y)
    y.left = z.left
    y.left.p = y
```

2. For any n-key B-tree of height h and with the minimum node degree of $t \geq 2$, prove that h is no larger than $\log_t \frac{n+1}{2}$. (Hint: consider the number of keys stored in each tree level.) (10%)

3. Given an initial B-tree with the minimum node degree of $t = 2$ below, show the results (a) after inserting the key of H and (b) then followed by deleting two keys in order: X then P. (show the result after insertion and the result after each deletion; 18%)



4. The recurrence of Procedure CUT-ROD(p, n) is given by $T(n) = 1 + \sum_{j=0}^{n-1} T(j)$, with $T(0) = 1$. Solve $T(n)$. (10%)

5. The problem of optimal parenthesization over a chain of matrix multiplications can be solved by a divide-and-conquer approach recursively. Let $m[i, j]$ denote the minimum number of scalar multiplications needed to compute $A_i \bullet A_{i+1} \bullet A_{i+2} \bullet \cdots \bullet A_j$, with A_k sized as $p_{k-1} \times p_k$ (for $i \le k \le j$), give the recurrence definition of the problem. (8%)

6. Given a set of 4 keys, with the following probabilities, determine the cost and the structure of an optimal binary search tree, following the tabular, bottom-up method realized in the procedure of OPTIMAL-BST below to construct and fill $e[1..5, 0..4]$, $w[1..5, 0..4]$, and $root[1..4, 1..4]$.

matrix multiplication chance

i	0	1	2	3	4
p_i		0.12	0.08	0.14	0.20
q_i	0.05	0.08	0.07	0.16	0.10

(a) Fill in the missing 3 statements in the procedure. (12%)
(b) Construct and fill the three tables, and show the optimal BST obtained. (22%)

```
OPTIMAL-BST(p, q, n)
let e[1..n+1, 0..n], w[1..n+1, 0..n],
        and root[1..n, 1..n] be new tables
for i = 1 to n + 1
    e[i, i-1] = q_{i-1}
    w[i, i-1] = q_{i-1}
for l = 1 to n
    for i = 1 to n - l + 1
        j = i + l - 1
        e[i, j] = ∞
        w[i, j] = w[i, j-1] + p_j + q_j
        for r = i to j
            1. t = e[i,r-1] + e[r+1,j] + w[i,j]
            if t < e[i, j]
                2. e[i,j] = t
                3. root[i,j] = r
return e and root
```

+q

Chance
Huffmancode

*

7. A Fibonacci min-heap involves the procedures of CUT and CASCADING-CUT during a key decrease operation when necessary. Given the following Fibonacci heap, show the resulting heaps after (1) the node with key = 30 decreases its key to 6 and then (2) the node with key = 35 decreases its key to 3. (10%)