Data Structure Mid-Term Exam.

Nov. 18, 2011

1. *5%* What is the result of $O(n) + \Theta(n \log n) + O(n \log n)$? And your reasons please.
2. *5%* A is an $N \times N$ upper triangular matrix (entries above the diagonal (including the diagonal) are nonzero, all the others are zero.). Suppose that we would like to store the nonzero terms in a 1D array B. Given i and j, describe the algorithm to retrieve $A[i][j]$, i.e., to answer $A[i][j]$ is zero or to retrieve $B[k]$ that stores the value of $A[i][j]$.
3. *7%* An $r \times l$ sparse matrix stored in an array contains m non-zero terms. Describe the fast transpose algorithm. How much time does each step take?
4. *7%* Calculate the failure function of the pattern $Pat = abababacabab$.
5. (a) *5%* Convert the infix expression $(A - ((B - (C + D)) + (E - A))) * C$ to a postfix expression using a stack.
   (b) *3%* Explain why do we have to define different priorities for the token "(".
6. *8%* Describe the algorithm and the data structure required so that we can "free" a linked list in $\Theta(1)$ time.
7. T is a tree that every node has 2 or 3 children and all the leaf nodes are at the same depth.
   (a) *5%* What is the minimum number of nodes in level i (root is at level 1)?
   (b) *5%* What is the minimum height of T if there are n nodes.
8. Given a set of m relations over n elements, if the relation is symmetric, reflexive, and transitive, we are going partition n elements into equivalence classes.
   (a) *6%* Use the linked list structure to solve the problem. What is the time complexity for the best algorithm? What is the space requirement for the best algorithm?
   (b) *6%* Use the forest representation to solve the problem. What is the time complexity for the best algorithm? What is the space complexity for the best algorithm?
9. *6%* In the Union/Find operation, Union is done by merging two trees. If the **weighting rule for Union** is applied, the height of the resulted tree of n nodes is at most $\lfloor \log_2 n + 1 \rfloor$. Show that the bound $\lfloor \log_2 n + 1 \rfloor$ is tight (the bound is achievable, i.e., there is a tree obtained as the result of weighting rule union has height $\lfloor \log_2 n + 1 \rfloor$).
10. Binary search tree: we start with an empty binary search tree.
    (a) *3%* Draw the resulted tree T if we have the sequence of insertions 9, 15, 6, 4, 12, 19, 17, 22, 14, and 13. Let this tree be T.
    (b) *3%* Put threads into the tree T so that inorder traversal can be done without using a stack.
    (c) *3%* Draw the trees after deleting 14 then deleting 9 from T.
    (d) *6%* Draw the trees after splitting the tree at 14 from T. After splitting, we have three parts, *min*, 14, and *max*. Draw the *min* and the *max*.
11. 6% Show that any binary tree of n nodes has $\lceil n/2 \rceil$ leaves (external nodes). Note that a node has a child actually has an internal node (the child) and an external node.
12. Heap structure:
    (a) *5%* We use an array A to implement a max heap and we don't use the first entry, i.e. $A[0]$. Show that parent of $A[i]$ can be found at $A[\lfloor i/2 \rfloor]$

(b) *5%* Think about this, if you have two max heaps, H_1 and H_2. Keys in H_1 are less than the keys in H_2. Using the implementation mentioned in class, design an algorithm to merge H_1 and H_2 into the max heap H. What is the time complexity of your algorithm?

(c) *6%* This one is hard. Design a heap structure that supports insert arbitrary and delete the kth largest. Describe the data structure you need and describe the algorithm.