CS 5633 Analysis of Algorithms – Spring 10

2/11/10

5. Homework

Due **2/18/10** before class

1. **Multi-min (6 points)**
   Consider the following task: Given an unsorted array of n numbers, find the k smallest numbers and output them in sorted order.
   Describe three inherently different algorithms that solve this problem. Analyze their runtimes in terms of n and k (so you should have n and k in the big-Oh notation). Try to find the fastest possible algorithm. Which of your algorithms is the fastest?

2. **Red-black tree rotations (4 points)**
   Find a sequence of numbers which, when incrementally inserted into a red-black tree, causes the following sequence of rotations:

   left, left, right, left.

   You may start with an initially non-empty tree, and you may insert numbers that do not cause any rotations. But there should not be any additional rotations performed.

   Draw the sequence of trees that you obtain after each insertion. For each such tree indicate the node that violates the red-black tree condition, indicate the nodes that participate in the rotation, the type of the rotation, and the subtrees that correspond to each other before and after the rotation.
   Hint: Use a red-black tree demo from the web.

3. **Red-black trees (4 points)**
   The *black-height* of a red-black tree is the black-height of its root vertex.

   (a) What is the largest possible number of internal nodes in a red-black tree with black-height b?

   (b) What is the smallest possible number of internal nodes in a red-black tree with black-height b?

   Justify your answers.

4. **Transforming binary search trees (5 points)**

   (a) Show that an arbitrary binary search tree with n vertices can be transformed into a right-going chain using $O(n)$ rotations.

   (b) Show that an arbitrary binary search tree with n vertices can be transformed into any other binary search tree with n vertices using $O(n)$ rotations.

5. **B-trees (4 points)**

   (a) What is the maximum number of keys that can be stored in a B-tree with minimum degree k and height h? Your answer should depend on k and h.

   (b) The CPU time of B-Tree-Search is $O(k \log_k n)$. Show that if B-Tree-Search is changed to use **binary search** instead of linear search on the key then the CPU time is only $O(\log n)$, which is independent of k.

Related questions from previous PhD Exams

Just for your information. You **do not** need to solve them for homework credit.

1. This problem is concerned with binary search trees and the successors of a node.

   (a) Define "binary search tree".

   (b) Given a node x in a binary search tree, write an *efficient* procedure in pseudocode to find the successor of x. Assume that nodes have fields *left*, *right*, *parent*, and *key*. What is the running time of your procedure in terms of n and h? (Justify your answer.)

   (c) Consider the following procedure to print all n elements of a binary search tree in order: First find the minimum element and then make $n - 1$ calls to your successor procedure. What is the running time of this procedure? (Justify your answer.)

   (d) For a binary search tree, determine the running time of finding a value v and then performing m successive calls to your successor procedure. (Justify your answer.)