CMSC 341 Data Structures
B-Tree Review

April 28, 2005

1. Define B-Tree.
2. What does it mean to say a B-Tree is **order M**?
3. When describing a B-Tree, what does L represent?
4. Give the pseudo-code for finding a particular element in a B-Tree of order M.
5. Given the drawing of a B-Tree, show the new B-Tree after inserting a given element.
6. Given the drawing of a B-Tree, show the new B-Tree after deleting a given element.
7. Draw a valid B-Tree with $M = 4$ and $L = 3$ containing the integer values 1 - 25.
8. Show the result of inserting the elements 1, 3, 5, 7, 9, 11, 6 into an initially empty B-Tree with $M = 3$ and $L = 3$. Show the tree at the end of each insertion.
9. Given the characteristics of an external storage problem

   (a) The number of items to be stored

   (b) The size (in bytes) of the key for each item

   (c) The size (in bytes) of each item

   (d) The size (in bytes) of a disk block

   design a suitable B-Tree (i.e. calculate appropriate values of M and L).
10. What is the minimum and maximum number of leaves in a B-Tree of height 2 when $M = 3$?

11. The average case performance of the dictionary operations insert, find, and delete is in $O(lg\, N)$ for balanced binary search trees like Red-Black trees. In a B-Tree, the average asymptotic performance for the dictionary operations is in $O(log_M N)$ where M is the order of the B-Tree. Discuss the following

    (a) When M = 2, do the B-Tree and the RB Tree have equivalent asymptotic performance for the disctionary operations? Are there advantages of one over the other?

    (b) B-Tree height is proportional to $log_M N$ indicating that for a given N, a B-Tree of higher order will be shorter than one of lower order. Is this true? If so, why not always choose a very high value for M since the average asymptotic performance of the dictionary operations is in $O(height)$.

    (c) B-Trees find their greatest utility when data are stored externally (on disk rather than in memory). Why is this so?