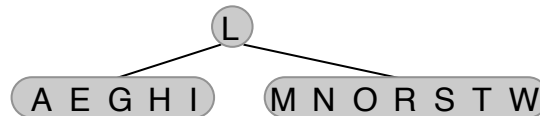


Homework 5

April 1, 2008

Due Date: April 8

1. (20 points) This problem focuses on the B-tree data structure.
 - (a) (5 points) Suppose that you have an application in which you want to use B-trees. The computer you will be using has disk pages holding 2048 bytes. Each tag is 4 bytes long, each reference to child/parent (which is a disk page id) is 4 bytes, each data record reference (which is a disk page id along with an offset within the page) is 8 bytes. For simplicity, for this problem you can assume that a B-tree node just consists of the array of elements, child references, and the parent reference. What value would you select for t ? *Show how you derived it.*
 - (b) (4 points) For the value of t you derived in part (a), if you have an application in which you want to store 1,000,000,000 items in your B-tree, what is the maximum number of disk pages that will be brought into main memory during a search given that the root is kept in main memory at all times?
 - (c) (5 points) Show the B-tree of order 2 (i.e., $t = 2$) that results when inserting the letters of "N,E,W,A,L,G,O,R,I,T,H,M,S" in that order using bottom-up insertion. You are required to show the B-tree just before each split occurs.
 - (d) Consider the following B-tree of order 4 (i.e., $t = 4$).



- i. (2 points) Show the B-tree that results when inserting Y.
 - ii. (1 point) Show the B-tree that results when removing T from the B-tree from part (a).
 - iii. (3 points) Show the B-tree that results when removing W from the B-tree from part (b)
2. (10 pts) Give a high-level description (no code!) of the most efficient algorithm you can take the union of skiplists S_1 and S_2 . (S_1 and S_2 can be destroyed by this method.) Analyze the expected time complexity of your algorithm when there are n_1 items in S_1 and n_2 items in S_2 . Recall that the expected height of a tower is $1/(1 - p)$.
3. (30 points) For the following problem you are to select the data structure(s) to use. The following components should be clearly given in your solutions. Your solution should be at most one page.
 - You should very clearly describe your data structure choice, including all decisions about how the data structure is to be applied (e.g., what is used as the tag, what is the associated data, what is the table size and collision resolution choice for each hash table,...).
 - You should clearly describe how each of the provided operations will be implemented AND discuss the efficiency for each of the operations. You only need to describe any new methods or variations of methods covered in class that you need. You can specify any standard method you are using (e.g., insertion) along with its parameters (e.g., what is given as the tag, and what is given as the associated data), and state its time complexity.

- (a) (25 points) You are to maintain a multimedia document with a set of n video segments where n is about 100,000. Each segment s consists of a beginning time b , an ending time e , and a reference to a video object v . You should assume the the video segment is large and must be stored on disk. (So the video segment location will be stored as a disk location.) You must support the following methods.
- Given a new segment $s = (b, e, v)$ determine if s overlaps any segment currently stored. This should run in worst-case or expected-case $O(\log n)$ time.
 - Insert a new segment $s = (b, e, v)$ into the schedule if it does not overlap any currently stored. (If it does overlap any segment then just report that it cannot be inserted). This should run in worst-case or expected case $O(\log n)$ time.
 - Remove the segment with the earliest beginning time, returning the location (on disk) of the associated video segment. This method must run in worst-case or expected-case $O(1)$ time.
- (b) (5 points) How would you modify your data structure design for the application described in part (b) if $n = 1,000,000,000,000$?
4. (5 points) Show the B+-tree of order 2 (i.e., $t = 2$) that results when inserting the letters of “N,E,W,A,L,G,O,R,I,T,H,M,S” in that order using bottom-up insertion. You are required to show the B+-tree just before each split occurs. Please use the convention that when a B+-tree leaf splits, that the median element is kept with its left child. While this choice is arbitrary, it will make the grading easier to follow it.
5. Consider the digitized ordered collection $\langle ab, abc, ba, bba, c, ca, cbc \rangle$ over the alphabet $\{a, b, c\}$ where each element ends with an end-of-string character.
- (a) (5 points) Show the trie holding the given collection.
- (b) (5 points) Show the compact trie holding the given collection.
- (c) (5 points) Show the compressed trie holding the given collection.
6. (10 points) Consider the following set of two-dimensional set of points
- $$\{(6, 6), (9, 4), (14, 3), (1, 2), (4, 10), (3, 15), (3, 12), (12, 14)\}.$$
- (a) (5 points) Show the k -d tree that results when inserting the above points in that order. Use the convention that the root uses the x -coordinate for branching.
- (b) (5 points) Show the subdivision of the plane the corresponds to the k -d tree you gave in part (a).
7. (10 pts) Let array $A = \langle 14, 10, 12, 7, 4, 8, 9, 5, 6, 2 \rangle$ be a binary heap.
- (a) (1 pts) Show this binary heap in tree form.
- (b) (6 pts) Show the state of the binary heap after inserting 15, then 13, and then 14. Please show the state of the binary heap (in tree form) after each insertion. You can show an intermediate steps if you'd like.
- (c) (3 pts) Show the binary heap (in tree form) that results when performing an `extractMax`.

Extra Credit Problem:

7. (5 pts) Describe an $O(n \log k)$ algorithm to merge k sorted lists L_1, \dots, L_k (sorted in increasing order) into one sorted list, where n is the total number of items in all the lists combined.

Big Hint: Think about the merge step of mergesort and how you can use a tagged priority queue. Be sure to clearly describe what you will use as the tag and associated data.