

NAME:

CS 241 Algorithms and Data Structures

Spring Semester, 2003

Final Exam

May 2, 2003

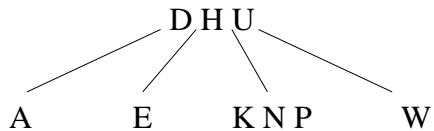
- Do not spend too much time on any problem. The point value approximates the time I expect you to need for the problem. (Note the exam has 110 points.)
  - If you plan to solve a problem using a standard graph algorithm then you should clearly define the graph and then can just state the name of the algorithm you are going to use. If you want to use a variant of a standard algorithm, just describe the variations from the basic algorithm.
  - When asked to show your work in applying an algorithm, show it in the same way you were asked to do so in the homeworks. For example, for bfs, dfs you should give the order in which each vertex was visited and give its parent. For Prim's algorithm, and Dijkstra's algorithm this should include giving the order in which the vertices are removed from the priority queue and giving the parent for each vertex. There should be enough work for us to determine if you properly applied the algorithm to get your solution.
  - Please write up all solutions clearly, concisely, and legibly.
1. (10 pts) For each of the below,  $T_1$  and  $T_2$  correspond to time complexities of two different algorithms. Which one corresponds to the more efficient algorithm (based on the asymptotic complexity). For any recurrences given you can assume that  $T_2(1) = 1$ . You MUST give asymptotically tight solutions to the recurrences given as part of showing your work.

(a)  $T_1(n) = n^2 \log n$ ,  $T_2(n) = 4T_2(n/2) + 10 \log n + n\sqrt{n}$

(b)  $T_1(n) = 10n + n \log n$ ,  $T_2(n) = n^{4/3}$

(c)  $T_1(n) = 100\sqrt{n}$ ,  $T_2(n) = 2T_2(n/4) + \sqrt{n} + \log n$

2. (15 pts) Consider the following B-tree with a minimum branching factor of  $t = 2$ :

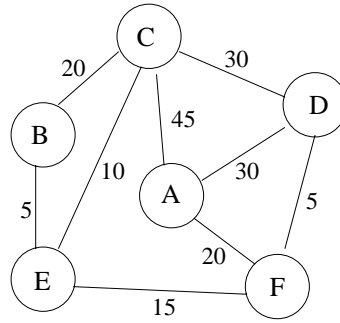


(a) Show the B-tree that results from inserting M into the above B-tree.

(b) Show the B-tree that results from deleting A in what you obtained AFTER inserting M.

(c) Draw a legal red-black tree that corresponds to the ORIGINAL B-tree shown above.

3. (20 pts) In the graph below each vertex represents a street intersection, each edge represents a stretch of road and the weight gives the average time needed to travel that stretch of road.



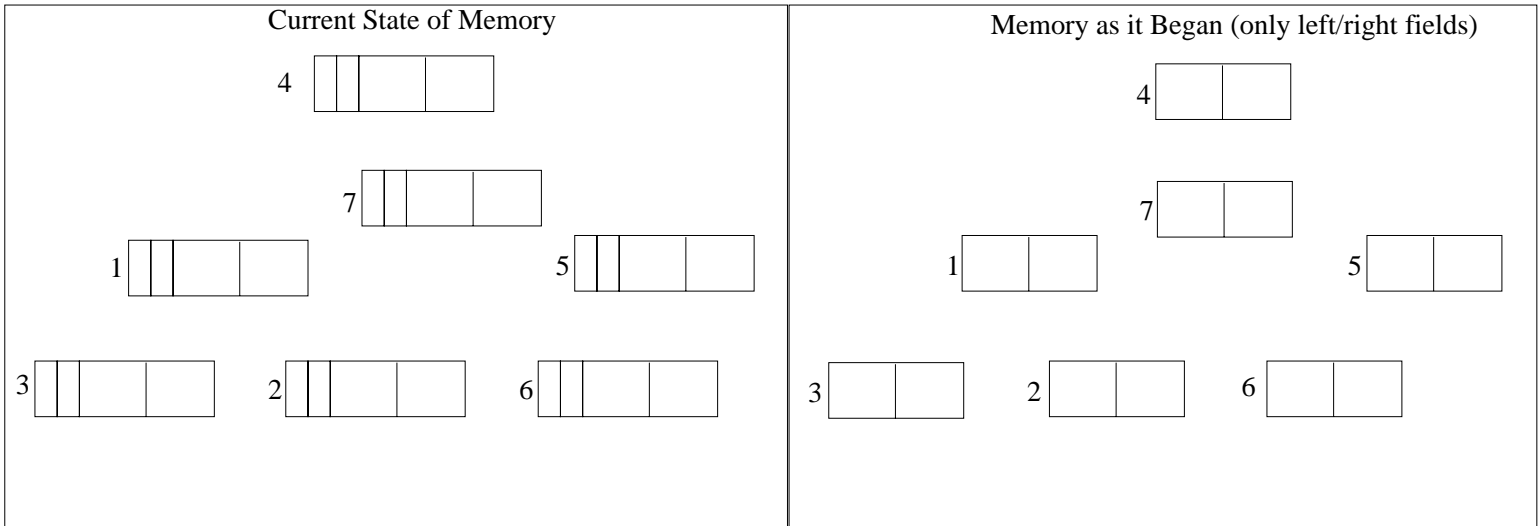
- (a) Suppose you want to find the route to go from intersection A to intersection B that minimizes the number of intersections that you must go through. Describe how you would solve this problem in general and then demonstrate your solution on the above graph. Show your work to enough detail that we can see how you reached your solution.
- (b) Suppose you want to find the route to go from intersection A to intersection B that minimizes time it would take. Describe how you would solve this problem in general and then demonstrate your solution on the above graph. Show your work to enough detail that we can see how you reached your solution.



5. (15 pts) Below is the table showing memory *during* the in-place DFS used by the mark phase of the mark-and-sweep garbage collection algorithm. Note that 0 represents null and “X” is used to indicate the mark bit is set.

Current State of Memory

	mark	back	left	right
1	X	L	0	6
2		R	a	3
3		L	1	2
4	X	R	1	7
5	X	R	c	7
6		L	5	7
7	X	L	1	b



You are to:

- (a) Complete the graphical view of memory as it is shown in the table on the left.
- (b) Current = 5 and previous = 4. With this information (and memory as shown above) determine the DFS stack and clearly show it below.

- (c) Complete the graphical view of memory as it began. Very briefly describe in words below what you did to obtain your solution.

- (d) Which cells will be placed on the free list during the sweep phase?

6. (30 pts) In this problem you will design data structure(s) to support a simple message board system. Let  $n$  be the number of users and  $x_i$  be the number of message in user  $i$ 's message board.

- `postMessage(senderID, receiverID, time, message)` should add the message to the receiver's message board. The sender puts the timestamp on the message and different sender's clocks are not necessarily synchronized. So a message  $m_2$  posted after message  $m_1$  might have a smaller value for the time.
  - `remove(userID, senderID)` should remove all messages in the message board for the user (specified by `userID`) that have been posted by the given sender (specified by `senderID`).
  - `lookup(userID, a, b)` should return an iterator that contains all messages (in sorted) order that have a time value between `a` and `b` that have been posted for the given `userID` that have not been removed. The iterator method that gets the next item must run in  $O(1)$  time.
- (a) For this part you should assume that all needed data structure can be stored in main memory. Describe the data structures you would use to implement the above methods so they run as efficiently as possible. While the highest priority is to make these methods run efficiently, also be as efficient in your usage of space as you can. (In the next part you'll describe the methods – here just describe the data structure(s) used.)

(b) Clearly describe how to implement each method and analyze the time complexity.

- (c) Now suppose that there are billions of users and that there is not room in main memory to even store a single integer per user. How would you modify your solution for this scenario? You can assume that any user has at most 1000 messages in his/her message board.

Problem	Points Possible	Points Received
1	10	
2	15	
3	20	
4	20	
5	15	
6	30	
total	110	