

Homework 5

April 15, 2003

Due Date: April 22

1. (10 pts) For the following problem, use the directed unweighted graph given by the following adjacency list. Be sure to consider the edges in the given order.

A: B E

B:

C: A E

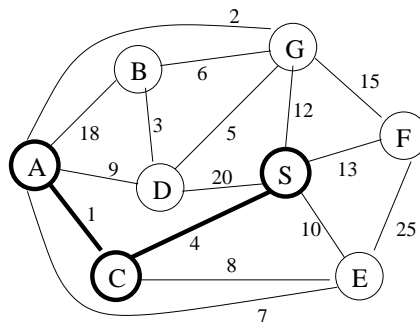
D: C A F

E: B

F: C E B

- (a) For the source vertex $s = F$ what is the order in which the vertices are visited by BFS (breadth first search)? Also, show the breadth-first search tree that you obtain.
- (b) What is the order in which the vertices are visited by DFS (depth first search). **You should assume that the top-level DFS procedure visits the vertices in alphabetical order.** For each vertex give the discovery and finishing time.
- (c) Suppose that this graph is a precedence graph. Using your work above either give a valid order in which to perform the tasks (call them task A, task B, ..., task F) or convince us that there is no valid order.

2. (15 pts) For the weighted undirected graph below, we are in the middle of running Prim's MST algorithm with vertex S as the source. Edges (S, C) and (A, C) which are shown as bold lines, have been added to the spanning tree (i.e. S, A and C have been removed from the priority queue).

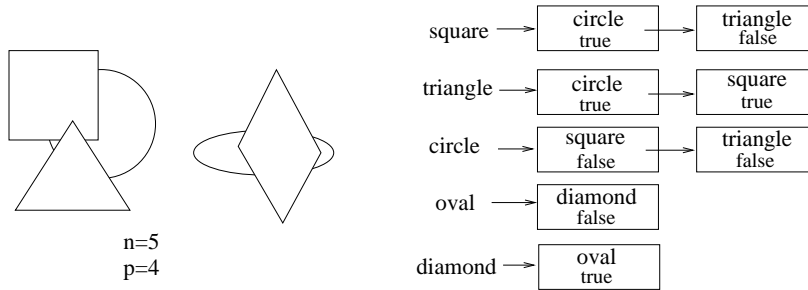


- (a) For vertices B,D,E,F, and G what would be their key and parent in the priority queue? You should be able to answer this by just looking at the graph above. Think about what the key and parent correspond to in Prim's algorithm.
- (b) What is the next edge added in Prim's MST algorithm? After adding this edge, some keys in the priority queue are changed. Give a list of every key in the priority queue that is changed (including the new value) during this execution of the main loop (i.e. up until the next `extractMin`).
- (c) List all additional edges placed in the minimum spanning tree by Prim's algorithm (when run to completion) in the order in which they would be added. Show your work if you want to receive partial credit if your answer is wrong.

3. (20 pts) Consider a variation of the stack ADT that supports the operations PUSH, POP, MINIMUM and EXTRACTMIN. Describe an implementation for which MINIMUM has worst-case cost $O(1)$ and PUSH, POP, and EXTRACTMIN have worst-case cost $O(\log n)$ where n is the number of elements in the ADT.

Be sure to give a clear and complete description of your data structure, and a high-level description (and pseudo-code if you feel it is needed for clarity) for how you will implement each of the four methods. Also you must analyze the worst-case asymptotic time complexity for each of the methods. (*Hint: Can you use the idea of handles to help out here?*)

4. (20 pts) Consider the task of redrawing a set of graphic objects in a drawing program. Assume there is an iterator available to iterate over all graphic objects and for each graphic object x there is a list of objects it overlaps with (which can you iterate over) where each list element has two data fields: the object o that x overlaps, a boolean indicating if x is on top of o . Let n be the number of objects and p the number of pairs of objects that overlap. Below is a simply example showing the lists for each of 5 objects in the picture.

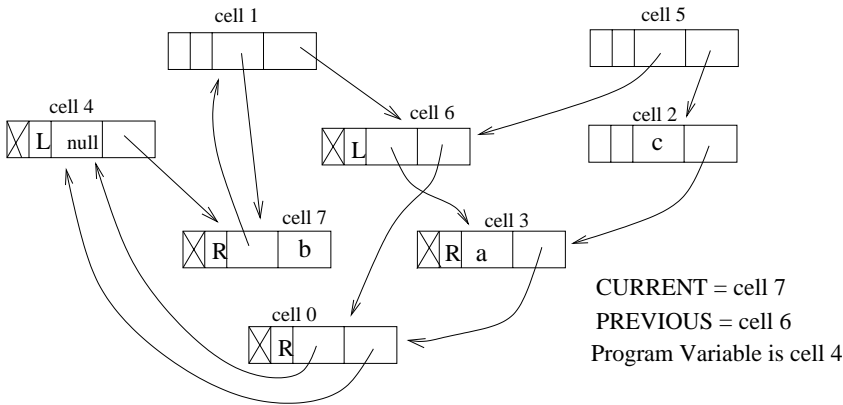


When an object is displayed it is drawn in its entirety and will cover anything already drawn that occupies the same space. You are to select/design the most efficient algorithm you can to determine the order in which to display the objects so they appear correctly (or report that it is not possible to do this while drawing an entire object at a time). Then analyze the worst-case asymptotic time complexity of your algorithm (given that the data structure described above is already built) in terms of n and p . If you build any additional data structures then include that in the time complexity analysis for your algorithm.

5. (20 pts) A search-engine company has decided on the following method to rank web page quality based on an individual's preference. Each individual can give a web page X that they consider to be a very high quality page. For every other web page its quality is defined as the minimum number of hyperlinks that must be followed from page X to get to it. Suppose that there are p web pages being considered with a total of ℓ hyperlinks. You must:

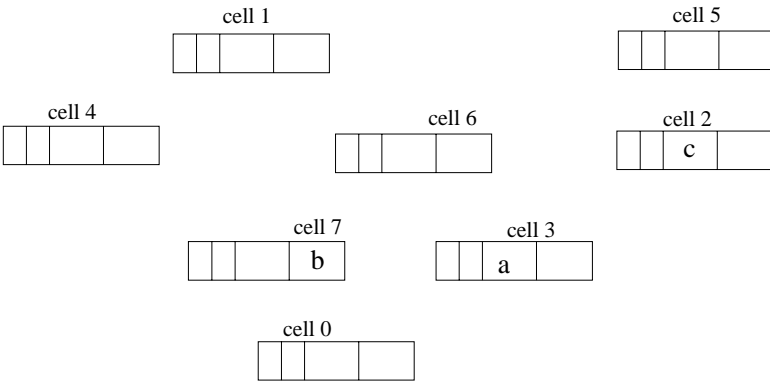
- Formulate this problem as a graph problem. Give enough detail so that given a set of web pages and hyperlinks someone could, without any ambiguities, create the graph. So it should be clear if the graph is directed or undirected, weighted or unweighted, ...
- Select a graph representation under the assumption that p is roughly 1,000,000 and ℓ is roughly 10,000,000 (i.e. on average each web page as 10 links).
- Select/Design the most efficient algorithm you can to compute the quality measure defined above for all p web pages and analyze the time complexity of your algorithm.

6. (15 pts) Below is a pictorial view of memory during the middle of the in-place DFS during phase 1 of the copying collection algorithm. Cells 0, . . . 7 are the from-space and cells 8, . . . , 15 are the to-space. You are to do the following things:
- Complete the list showing the DFS stack.
 - Fill in the links in the graph showing memory as it was before garbage collection began.
 - Using your solution to (b), fill in the pair of tables showing memory AFTER garbage collection is COMPLETED. Cells 0-7 should be shown exactly as they would be (i.e. the old data would be there except when replaced by forwarding pointers).



a) stack head \rightarrow cell 7 \rightarrow cell 6 \rightarrow

b) Show memory BEFORE garbage collection BEGAN



c) Memory (left and right fields) AFTER garbage collection COMPLETES

	left	right		left	right
cell 0			cell 8		
cell 1			cell 9		
cell 2			cell 10		
cell 3			cell 11		
cell 4			cell 12		
cell 5			cell 13		
cell 6			cell 14		
cell 7			cell 15		

Indicate which cells are on the free list at the end of the garbage collection by leaving them blank.

Challenge Problem: (5 points)

Suppose that all edge weights in a graph are integers in the range from 1 to 10. Your task is to think of a new way to implement the Priority Queue ADT for this special case so that Prim's algorithm runs in time $O(m + n)$ where n is the number of vertices and m the number of edges in the graph. Be sure to clearly describe your data structure for the priority queue and clearly describe how `insert`, `extractMin` and `decreaseKey` will be implemented. Then analyze the worst-case time complexity of these three methods and from that analyze the time complexity of Prim's algorithm.

Hint: What is the relationship in Prim's algorithm between the key value for the priority queue and the edge weights.