1 Greedy Algorithms (30 Points)

Suppose that a company has recently aquired a new piece of software and needs to train its employees on how to use this software. Not everyone is available to meet at the same time, and so the company will need to offer several meetings. The company asks each employee to provide a time interval for when they are available to meet. For employee i, this window must have the form $[e_i, l_i]$ where $e_i < l_i$. This interval represents that employee i can attend a meeting that begins anytime after early time e_i and before l_i.

Once we have the intervals from each of the employees, we want to compute a set of meeting times so that every employee can attend at least one meeting. Since the meetings will cost the company money, we would like to minimize the number of meeting times that still allows each employee to attend a meeting.

Give a greedy algorithm (English is okay) that will compute an optimal solution for this problem. Argue why your solution is guaranteed to be optimal.

2 Amortized Analysis(30 Points)

Recall the dynamic array data structure that was asked in class. If our data structure currently has size k, then we double the size to $2k$ once we receive operation $k+1$ and copy all of the elements from the old data structure to the new one. Suppose that instead of doubling to size $2k$, we quadruple to size $4k$. This question will consider the amortized cost of an operation in this data structure.

1. Perform an aggregate analysis to obtain a bound on the sum of the cost of n operations.

2. Use the accounting method to determine the amortized cost of a single operation.

3 Graph Algorithms (20 Points)

In class we saw that a directed acyclic graph (DAG) could be used to represent precedence, such as courses and prerequisites, where each course is a vertex and we have a directed edge from u to v if course u is a prerequisite for course v. The graph is acyclic or else we would not be able to take some of the courses.

Now suppose our classes c have prerequisites such that you should take $k-1$ out of k courses prior to taking c. For example, a course c_1 may have a prerequisite that says you should take 1 out of as set of 2 courses prior to taking c_1, and perhaps we have a course c_2 that says you should take 2 out of a set of 3 courses prior to taking c_2. In otherwords, it is okay to not have taken one class from the set of prerequisites prior to taking a course.

In this setting, the notion of a topological sort should change. Before, all arrows should point to the right when listing the courses from left to right in sorted order. Now we allow for at most one arrow to point to the left (this arrow represents the one course that wasn't taken from the set of prerequisites). Give an algorithm to compute a topological sort in this setting.

4 Minimum Spanning Trees (20 Points)

Remember that MSTs provide a cheap way of connecting nodes together in a network so that there is a path connecting each pair of nodes. This path allows for message passing. We had always considered this problem in the context of undirected graphs. Suppose now that the graph is directed (messages can only be passed in one direction along a link), and we have a hub node h in the graph that needs to send messages to the other nodes in the graph. Consider computing a minimum cost spanning tree in a directed graph such that there is a directed path from h to each of the other nodes so that the sum of the edge weights in the graph is minimized. Give an algorithm for this problem and argue why it is correct.