Algorithms
Computer Science 140 & Mathematics 168
Spring 2007
Homework 4a
Due Thursday, February 8

You are encouraged, but no longer required, to use LaTeX to typeset your solutions to assignments in this course.

1. **[15 Points] Greed Must be used with Extreme Caution!** In class we talked about the problem of finding the maximum number of non-conflicting courses from a given set of courses. We argued that if the courses are sorted in order of completion time, then the following greedy algorithm is guaranteed to find the maximum number of non-conflicting courses: Choose the course with earliest completion time. Cancel out all courses that conflict with that course. Now repeat the process for the remaining courses.

   Professor I. Lai of the Pasadena Institute of Technology has proposed the following four alternative greedy algorithms for this problem. None of them are correct! To show this, construct a separate counterexample (a set of intervals) for each algorithm below such that the algorithm does not find the optimal solution for your counterexample. Make sure to explain what the algorithm would do on your counterexample and what the optimal solution would be.

   Algorithm 1: Sort the courses by ending time as before. Now, run our original greedy algorithm in the opposite direction. That is, choose the course that ends **latest**. Then cancel out all courses that conflict with that course. Now repeat the process for the remaining courses.

   Algorithm 2: Sort the courses by increasing starting time (rather than ending time). Now, choose the course that starts first. Then cancel out all courses that conflict with that course. Now repeat the process for the remaining courses.

   Algorithm 3: Forget about sorting the courses. Choose a course of shortest duration (that is the course that has the least length). Then cancel out all courses that conflict with that course. Now repeat the process for the remaining courses.

Algorithm 4: Don't sort the courses. Choose a course that conflicts with the fewest other courses, breaking ties arbitrarily. Then cancel out the courses that conflict with that course. Now repeat the process for the remaining courses. (This is the most challenging of the four parts of this problem!)

2. [**15 Points**] **The Millisoft Party Problem Revisited!** On the last homework assignment, Gill Bates challenged you to find a dynamic programming algorithm to solve the Millisoft Party Problem. Now, Gill tells you: "Assigning a coefficient of fun to each employee is too subjective. I just want to find the largest number of people that I can invite such that we never invite an employee and their immediate boss. This is analogous to every employee having the same coefficient of fun."

   In other words, your task is to take as input a tree representing the company hierarchy. The tree need not be binary. Non-leaf nodes may have 1 child, 2 children, 3 children, or more! Your objective is to find an algorithm that computes the largest number of employees (nodes) that can be selected such that no two adjacent nodes (i.e. a node and its child) are chosen.

   Describe a *greedy algorithm* for this problem (in either pseudo-code or clear English), carefully prove its correctness, and analyze its running time.