CS1510 Greedy Problems 1 & 2

Rebecca Negley, Sean Myers

September 2, 2011

1. Consider the following problem:
INPUT: A set $S = \{(x_i, y_i) | 1 \leq i \leq n\}$ of intervals over the real line.
OUTPUT: A maximum cardinality subset S' of S such that no pair of intervals in S' overlap.
Consider the following algorithm:

   Repeat until S is empty.

   (a) Select the interval I that overlaps the least number of other intervals.

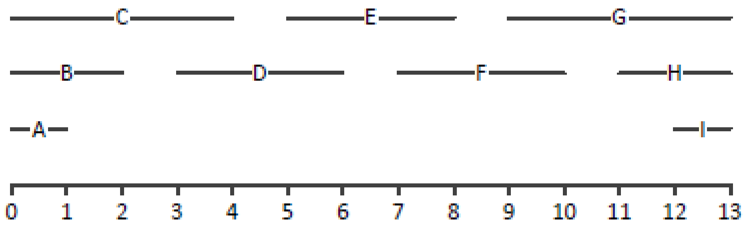
   (b) Add I to the final solution set S'.

   (c) Remove all intervals from S that overlap with I.

   Prove or disprove that this algorithm solves the problem.

   Solution: Consider $S = \{A, B, C, D, E, F, G, H, I\}$, with A, B, C, D, E, F, G, H, and I as defined in the following picture:

   Counterexample 1



   Observe that each interval overlaps at least two other intervals. Therefore, since the algorithm does not specify how to choose between intervals that overlap an equal number of other intervals, we can arbitrarily choose any interval that overlaps only two other intervals first. Suppose we choose E. Then, we remove the overlapping intervals D and F. Every remaining interval now overlaps two other intervals. We choose one of A, B, and C and one of G, H, and I. These selections will lead us to remove all remaining intervals. Hence, the greedy algorithm can produce a subset $S' = \{C, E, G\}$ with cardinality 3. However, we can observe that $S'' = \{B, D, F, H\}$ is an optimal subset and has cardinality 4. Hence, there exists an example were the greedy algorithm produces an incorrect output, so the algorithm does not solve the problem.

2. Consider the following Interval Coloring Problem.

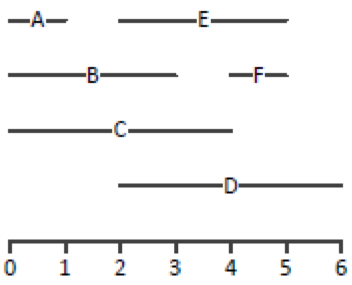
   INPUT: A set $S = \{(x_i.y_i) | 1 \leq i \leq n\}$ of intervals over the real line. Think of interval (x_i, y_i) as being a request for a room for a class that meets from time x_i to time y_i.
   OUTPUT: Find an assignment of classes to rooms that uses the fewest number of rooms.

   Note that every room request must be honored and that no two classes can use a room at the same time.

   (a) Consider the following interative algorithm. Assign as many classes as possible to the first room, then assign as many classes as possible to the second room, then assign as many classes as possible to the third room, etc. Does this algorithm solve the Interval Coloring Problem? Justify your answer.

   Solution: This algorithm does NOT solve the Interval Coloring Problem. Consider the counterexample below:

   Counterexample 2



   We can see that the maximum number of non-overlapping intervals is two. To assign as many possible classes to the first room, we may choose $S' = \{A, F\}$. (Note that this obeys the ends-first algorithm discussed in class.) However, in the interval $(2, 3)$, all of the other intervals overlap. Therefore, if we assigned S' to the first room, we would need four additional rooms to cover the $(2, 3)$ interval, so we would need five rooms. However, if we assign rooms according to the horizontal lines in the picture, we can assign the classes using only four rooms. Thus, the algorithm gives sub-optimal output for this example and does not solve the problem.

   (b) Consider the following algorithm. Process the classes in increasing order of start times. Assume that you are processing class C. If there is a room R such that R has been assigned to an earlier class, and C can be assigned to R without overlapping previously assigned classes, then assign C to R. Otherwise, put C in a new room. Does this algorithm solve the Interval Coloring Problem? Justify your answer.

   Solution: Observe that, if s is the maximum number of intervals that overlap at one particular point in time, you need at least s rooms in the optimal solution. Suppose the greedy algorithm gives more than s rooms. Then, there must be a time when s rooms have been started and are in use, and a class C is being added that would create an $s + 1st$ room. The only possible way for that to happen is if class C overlaps with s other classes at that time (so that each of the other s classrooms is full). It follows then, there are $s + 1$ overlaps. However, this violates the original statement that there are a maximum of s overlaps. Therefore, we reach a contradiction, and the greedy algorithm must use j rooms such that $j \leq s$. Since the optimal solution uses k rooms such that $s \leq k$, we know that $j \leq k$. This gives us that the greedy algorithm is better than or equal to the optimal solution. Since the optimal solution is known to be best, we must have $j = k$, so the greedy algorithm is correct.