Practice Questions on Greedy Algorithms

These questions are good preparation for the topic of greedy algorithms, which will be included on Test 2. They are not to be handed in. Some hints follow the questions, but don't look at them unless you are stuck after working on the problem for a while.

1. In class, we considered the following activity scheduling problem. Activity i has start time s_i and finish time f_i. The goal is to select the maximum number of activities with non-overlapping time intervals.

The greedy algorithm in class considered activities in order of their finish times, and chose an activity if it does not conflict with any activity already chosen. We showed this always finds an optimal solution.

Consider the following alternative greedy algorithms for this problem. Either show that there is a set of activities for which the algorithm does not find an optimal solution, or show that it always generates an optimal solution.

(a) Consider activities in order by *duration* and choose an activity if it does not conflict with any activity already chosen.

(b) First, throw away any activity that completely includes another activity. (I.e., if $s_i \leq s_j < f_j \leq f_i$, then throw away activity i.) Then, consider activities in order by their start time and choose an activity if it does not conflict with any activity already chosen.

(c) Sort the activities according to the number of other activities they conflict with. Then consider the activities one by one

2. How can you implement the correct approach(es) in Question 1 efficiently?

3. Alviine organizes timetables at a fitness centre. There are n different activities $a_1, \ldots, a_n$ to schedule. There are two timeslots to use for the activities. Alviine has a list of members who are interested in each activity. Ideally, activities should be scheduled so that, if any member is interested in two activities, those two activities are scheduled in different timeslots (so that the member can attend both of them). We call such a schedule of all the activities a *legal* schedule. Explain how to use one of the greedy algorithms discussed in class to figure out whether a legal schedule exists.

4. In class, we discussed the problem of scheduling unit-time jobs with deadlines and profits. There are n jobs, each of which takes one hour to complete. Job i has deadline d_i and profit p_i. If a job is completed before its deadline, then you get the profit p_i. If it is completed after its deadline, you get nothing. The goal is to find a schedule that yields the maximum total profit.

Generalize the algorithm discussed in class to the case where two workers are available to complete jobs. Your solution should specify which worker does each job and when. Prove that your generalized algorithm always generates an optimal solution.

Hints

1. Only one of the three actually works.

To show that one of the strategies fails, start with a set of activities that looks like this:

```
xxxxxxx  xxxxxxxx  xxxxxxx  xxxxxxxxx
     xxxxxx    xxxxxxx   xxxxx
```

and then add additional copies of some of the activities to force the algorithm to choose the middle activity on the second row (which will mean at most 3 activities can be chosen, whereas the optimal solution has 4 activities.

2. As mentioned above, there is only one correct approach in Question 1. One way to implement it efficiently is to start by sorting the activities by start times. The total runnning time of your algorithm should be $\Theta(n \log n)$.

3. The algorithm works for two timeslots, but not three. Think about how to formulate the problem as a graph problem.

4. The algorithm is quite similar to the one done in class. To prove the generalized algorithm is correct, consider how to modify an optimal schedule when the greedy algorithm assigns a job to a different worker than the optimal schedule does. (It will be a bit similar to the 2-colouring proof.)