CSC373— Algorithm Design, Analysis, and Complexity — Spring 2018

Tutorial Exercise 1: Greedy Algorithms

The first two questions below are mainly to ensure that you understand the lecture notes and can fill in portions of the proofs that were omitted in the lectures.

The third problem is to give you practice designing a greedy algorithm and proving its correctness. Consider this problem only if you and your study group finish the first two problems and wish to move on.

Bring any questions you may have to the next tutorial where you will get a chance to discuss these with TAs and other student groups.

1. Interval Scheduling. Make sure you understand the proof of the correctness of the interval scheduling algorithm given on slides 14 through 18 of the lecture notes. There are two questions on the top of p.18 that require some details to be filled in, and we look at those questions here.

First prove that, in case (2b), $J(k+1)$ is compatible with $\{J(j_m)|m = 1, \ldots, r\}$. Recall the notation here is that loop invariant $L(k)$ refers to an optimal solution, $\mathcal{O}$, which has been unpacked as $\mathcal{O} = \{J(j_m) \mid m = 1, \ldots, p\}$, where $p > r$. Hint: A suitable argument is very short but crystal clear.

In addition, prove that $J(k + 1)$ is compatible with $\{J(j_m) \mid m = r + 2, \ldots, p\}$. Hint: Recall jobs $J(k)$ have been sorted by finish time, so $f_1 \leq f_2 \ldots \leq f_n$.

2. Certificate for Minimum Max-Lateness. Consider the problem of minimizing the maximum lateness for a set of jobs, as discussed on pp.20-24 of the lecture notes. The input data has the form $\{(t_i, d_i)\}_{i=1}^{n}$. Here we assume t_i and d_i are all strictly positive integers. Consider the following function,

$$T(d_j) \equiv \sum_{\{i \ \mid \ d_i \leq d_j\}} t_i, \tag{1}$$

where "$\equiv$" denotes a definition. That is, $T(d_i)$ is simply the total of the execution times of all the jobs that have deadlines no later than d_i. Moreover, define $L^* \equiv \max\{0, \max\{T(d_j) - d_j \mid j = 1, \ldots, n\}\}$. Note that L^* can be easily computed given the data; it does not depend on first finding a suitable schedule.

2a) Let P denote any feasible schedule of the given jobs (that is, P specifies the order all jobs are scheduled and no two jobs overlap in time). Show the maximum lateness, $L(P)$, of this schedule satisfies

$$L(P) \geq L^*. \tag{2}$$

2b) Use part (2a) to prove that the greedy algorithm on p. 24 of the lecture notes is correct. Hint: Consider lower bound you obtained in part (2a) in relation to the schedule P generated by the greedy algorithm.

3. Cell Phone Towers Consider the problem of choosing cell phone tower locations on a long straight road. We are giving a list of distances along the road at which there are customers' houses, say $\{x_k\}_{k=1}^{K}$. No towers have yet been built (so no customers currently have service). However, a survey of the road has provided a set of J possible tower locations, $\{t_j\}_{j=1}^{J}$, where these potential tower locations are also measured in terms of the distance along the road. (These indexed lists of house and tower locations might be in any order. You can assume that all these distances are integers.)

Each customer will get service (at home) if and only if they are within a range $R > 0$ of at least one cell phone tower that gets built, that is, the house at x_k will have service iff there exists a tower that has been built at some t_j with $|x_k - t_j| \leq R$.

You can assume that if all the towers were built then every home would get service. However, such a solution could be overly expensive for the phone company. How can we minimize the number of towers that need to be built but still provide service at every house? Note that a suitable solution may still leave some parts of the road without cell service, even though each custormers' house will have service.

3a) Describe a plausible greedy algorithm for choosing the minimum number M of cell phone towers required, along with a suitable set of locations, say $T = \{t_{j(m)}\}_{m=1}^{M}$, such that every house has service. That is, for each $k = 1, \ldots, K$, there must exist an $m \in \{1, 2, \ldots M\}$ such that $x_k \in [t_{j(m)} - R, t_{j(m)} + R]$ (i.e, x_k can get service from tower $t_{j(m)}$).

3b) Prove that your algorithm in (a) is correct. That is, it will select the minimum possible number of towers to be built and identify an appropriate set of locations for these towers.