Problem Set 5—Due Friday 3:15 PM, May 25

(30) Problem 1 Consider the problem of planning the fastest plane trip from Sacramento to some far off location (say Shangri-la). Assume that you know for each airport the time each flight is scheduled to leave from that airport and the time it is scheduled to arrive (for simplicity, assume all times are given for Pacific time, so you don't have to worry about correcting for time zones). You may further assume that each flight will be exactly on schedule and that all flights are for one 24 hour period (starting at 10AM).

(a) Describe an efficient algorithm to find the route which gets you to Shangrala in the least time assuming you arrive at the Sacramento airport at 10AM. You may assume that if you arrive at an airport at time t, you can depart on any flight leaving after time t. In solving this problem, you should assume that all flights in your route must take off by 24 hours after your start (that is, by 10 AM, Pacific time, the next day).

(b) If there are a airports and f_i flights leaving airport i, what is the run time of your solution to part a)?

(30) Problem 2 Suppose that you have a communication network with routers and communication lines. Each communication line from router i to router j has a maximum bandwidth $b_{i,j}$ that can be sent through that line, and for a path through our network: $r_1, r_2, \ldots r_k$ the most bandwidth we can send through this path is determined by the minimum bandwidth of an edge on the path: thus the minimum value among the $b_{r_i, r_{i+1}}$ values, for $i = 1, 2, \ldots k - 1$.

Assume the graph of routers (as nodes) and communication lines as directed edges is given, along with the $b_{i,j}$.

(a) Describe an efficient algorithm to find the route from router i to router j of maximum bandwidth ((i, j) are inputs). Hint: one way to solve this problem is using a modification of Dijkstra's algorithm, though there are other ways too.

(b) If there are n routers and m communication lines, what is the running time of your solution?

(c) Now suppose that in addition to the start router i and end router j you also are given a specific bandwidth target B. Describe an efficient algorithm to find the route from i to j with fewest hops that has bandwidth at least B. Also give the run time of your algorithm.

(20) Problem 3 You have a beach house which you are going to rent for the rest of the year. You have solicited bids from interested renters of the form: s_i, f_i, r_i where s_i is the day they want to start renting, f_i is the day they want to finish renting, and r_i is the rent they are willing to pay for their stay.

Only one renter at a time can use the beach house (so if two requested times overlap, only one of them can rent the house). You want to find a set of renters such that no two stays overlap and the total rental income you get is as large as possible.

For example, if the bids are: (Jan 2, March 10, 100), (Jan 10, Feb 7, 60), (Feb. 9, March 25, 90), (March 10, April 10, 80), the best solutions takes the bids of the first and last renters for a total rent of 180. Note: if a rental ends on day f_i the next rental can start on day f_i (like a hotel, rentals end at noon, but don't start until 4PM).

(a) Describe how to use variation on shortest paths to find the best rental choices. Give an efficient algorithm to find the best set of renters and give the run time of your solution if there are b bids.

(b) Now suppose that you have two types of renters: clean and messy. Assume that when a renter submits a bid they also specify which they are (assume everyone is one or the other). After the first rental you may need to clean the beach house before the next renter uses it. If we go from a messy renter to a clean one, you need three days for cleaning (so a new rental starts on day $f_i + 3$ or later). From messy to messy or clean to clean now takes one day. From clean to messy no cleaning is required. Again describe how to find the best set of renters and analyze the running time.

(20) Problem 4 We consider an improved version of the Bellman Ford (BF) algorithm that uses some of the ideas we mentioned in class. We use the same general approach of using $d(v)$ values and the same initialization, but we try and be smarter about deciding which edges to relax.

At a high level we do the following:

INIT // our usual initialization.

Q $\leftarrow s$ // Q holds vertices whose $d(v)$ values have been updated recently.

```
While (Q not empty)
  $u \leftarrow$ Frontof(Q);
  for each neighbor $v$ of $u$
    Relax($u, v$)
    if $d(v)$ was updated by Relax and $v$ not in the Q, ADD $v$ to End of Q.
```

(a) Argue that this algorithm correctly implements finds the shortest path from s to all other vertices in a directed graph with no negative cycles: hint, argue that it does the same useful Relaxes (those that actually change a d() value) that BF does.

(b) What is the worst case run time of this algorithm if every shortest path uses at most k arcs? Justify your answer.

c Extra credit (7) Give a family of graphs that require $\Omega(nm)$ time for this algorithm.