Nathaniel Abreu 5/9/19

CS323 Assignment 2: Experiments Scheduling

A). We don't know how each step should be assigned. There are two possibilities for each student. Either they can complete a certain step or not. In the event that a student can complete a step we are concerned with how many more consecutive steps they can complete. For each consecutive step, # of steps − 1 and we record that student with that step. So the sub problems would be student, steps − 1 ~~and if a student can't complete a step then the # of steps is 0~~. Although Greedy Algo's don't always give the optimal solutions, this problem shows an optimal substructure since it assigns most consecutive steps and must contain an optimal solution.

B). A greedy algorithm that could find an optimal way to schedule the students with the steps would be to find the students which have the greatest consecutive steps. This is a greedy solution because by selecting the students with the greatest consecutive steps we are choosing the next student that offers the "most obvious and immediate benefit", since having a student who can do multiple consecutive steps reduces the need of other students for the steps which then reduces the # or amount of switches needed. The algorithm begins by selecting the 1st student checking if they can complete the 1st step. If so, that step is assigned to that student and we continue checking if that student can also complete

the proceeding steps. Once there is a step the Student can't complete, we check all the proceedyStudents to see who can complete that current Step and the next steps consecutively. The Student that offers the greatest consecutive steps including the step the 1st or previous student couldn't complete, is assigned those steps. This process is repeated until either all steps are assigned to a Student ~~[illegible]~~.

D). The runtime of the greedy Algorithm implemented in this problem is ~~O(n²)~~ O(nmn) It is O(nmn) because the while loop runs through n times and within the while loop are two nested for loops that run m times and at n times for the inner loop. Making the run time O(nmn)

E). Greedy optimal Solution Proof:

Defining Solution

int S ← Student 1
for i = 1 to n where n is total steps
    if Step i can be done by Student S then assign Step i to Student S

keep checking further Steps with Student S until Student S does not do a consecutive Step. Then we check the next Students to see who can do the most consecutive steps with that Step.

Prove: ALG is as good as OPT,
ALG choose: $U_1, U_2 \ldots U_k$ k = # of switches
OPT Solution: $V_1, V_2 \ldots V_L$ where $L < K$
Since we need the least amount of switches for the optimal solution

claim: i $(1 \leq i \leq k)$ $U_i \neq V_i$
Since [redacted] our algo is getting the most consecutive steps and at i $U_i \neq V_i$ the algo and opt are either both making a switch or the OPT is switching since it did not find consecutive 1's and the algo doesn't and stays with the same student.
Since OPT [redacted] can do more switches we

[redacted]

replace from the switch point OPT with Algo, so
By Design of our algo we will always be getting the least amount of switches.

Nathaniel Abreu 5/9/19

CS323 Assignment 2: Public, Public Transit

A). An algorithm that can be used for this problem is Dijkstra's Algorithm for a shortest path. We would not use Bellman-ford Algorithm because there are no negative weights in this problem since we are dealing with minutes it takes from one station to another. We begin by adding the value of X to 5:30 to get the start time and we select the vertex that represents station S to be the starting point. We use Dijkstra's Algorithm to compare the costs or minutes it takes to go from U to V for each pair of vertices, however we also take into account and add the minutes of the wait time based on the estimated time of arrival at V and frequency at V to add that to the cost to get the overall cost at each pair of vertices (U, V).

B). The time complexity of Dijkstra's Algorithm is $O(V^2)$ and for the implement of part A it would be $O(V^4)$ since there is an extra array that stores the time and freqencies of the trains at each station.

C). The Algorithm that the method ShortestTime is implementing is Dijsktra's Algorithm.

D). The modifications that are required are check for the arrival times and frequencies of the trains at each station so another array may need to be added to keep track of the frequencies and times.

e). The current complexity is $\mathcal{O}(V^2)$.