Sachin Nair
Svn343
9/19/2022

Programming Assignment 1 Report

A. Pseudocode for Uni optimal

Uni Optimal Algorithm:

While there is a university, u, that is not full and is accepting students:
    Let s be the first student on u's preference list
    If not (u is full) and s is not attending a university:
        U offers S a spot
    Else if U is full or S is attending another university, U2,:
        If S prefers U to U2:
            S withdraws from U2
            S attends U
        Else if S prefers U2 to U:
            S remains at U2

B. Runtime Complexity:
The runtime of this algorithm is O(m*n), where m is the number of universities and n is the number of students. As the outer while loop iterates so long as there is a not full university, the algorithm runs through at least m times given there is at least a minimum capacity of 1 student at each university. Then, the algorithm iterates through the number of spots at each university, which is the same number as the amount of students n, in the matching algorithm. Since the rest of the algorithm determines parings of students to universities based on the length of the preference lists of students and universities and the number of spots at each university. The rest of the algorithm is comparing spots on a preference list which is constant time, until a university is full of desired students. Since the algorithm is a nested loop iterating through m and n, we can confirm our O(m*n) runtime, and since the university is proposing, the algorithm is University-Favored as is the original GS algorithm.

C. Pseudocode for Student Optimal

Student Optimal:

While there is an unpaired student, s, who still has proposals to make and there are universities that are not full:

```
Let U be the first university on student S preference list
    If U is not full:
        S attends U
    Else if U is full:
        Check through the current students attending U
        If U prefers S to any other student, S2,:
            Remove the lowest ranking S2 at U, and offer S a spot
```

D. Runtime

The runtime of this algorithm is O(m*n*S), where m is the number of universities, n is the number of students, and s is the number of spots available for any university. Through the first iteration, we go through any unfilled universities and take an unpaired student out of n students and have him apply to the first preferred university on his preference list (we do this O(n) times. We compare in O(s) to see the lowest ranking student, and do a constant operation to swap the lowest ranking student for student S. We then do this for all the universities in O(m) times, giving us a O(m*n*s) runtime. I'm not sure how I would optimize this algorithm further without using a hash map, (which we were told not to use), but for the sake of the problem I think the runtime is not bad, and usually universities are not student favored, the actual use case of this algorithm is much less prevalent than that of a university optimal one. Lastly, since the students are proposing to the universities, we know that the algorithm will favor them, confirming we have a student proffered algorithm.