hw6-4102

jh4ctf

October 2021

1

We are to show that MST is unique if all edges within the tree are different. When we are done traversing through the graph, the weights should be elements of a set in order. Moreover, the minimal spanning tree X from algorithm that exists such as Prim or Kruskal, has the property that for any other spanning tree X, if we consider the weight in increasing order for X and X:ω_X :(X_1,...$X_(n)$) and ω'_X:(X_1,...$X'_(n)$) and satisfying $\omega_X \leq \omega'_X$ for all nodes and edges. We can transition from X to X' by substituting edges with same or slightly better edges. And if all edges' weights are different then X is strictly better than any other tree since at each step we substitute an edge with another strictly large weight because they have to be different. Thus we can prove the uniqueness.

2

First step is to break down all the possible flight destination into nodes and the waiting time in airports(difference between land time and departure time of next flight) to be the edges. And we need to turn them into a undirected graph with all possible flights and airports. Then we can start Dijkstra's Algorithm with the start node(Start Airport A). Then we run the Dijkstra's to get the shortest path which is intuitively the shortest waiting time in airport. Thus we will have the shortest waiting time in airport for Professor Floryan.

3

So our approach would be using Prim's Algorithm to construct MST for both robots but with a bit adjustments. In the first place, we start both robots from s_1 and s_2 to run Prim's Algorithm to try to form 2 MSTs. And both robots proceed at the same nodes level.

If there is no conflict with its overlapping on the same level, then it is done. If they overlap on the same level then we choose a robot called it as robot a to recede a node(pause for one level and to be one level down). And let the robot B to proceed with the MST and robot A to be one level less optimal. When

both robots reached to the final result MSTs, we go back on the receding node and check if the other robot receded, would both MSTs still be optimal. If not, then choose the other robot to recede which is robot A and proceed with the similar way. Then check if both are optimal for the MSTs.
The worst-case run-time would be caused by re-checking optimization of the MSTs. It would cost extra from O(V*log(V) + E*V) to add on $E^2 * V$to be O(V*log(V) + $E^2 * V$). It goes a bit slower than teh originial