Pablo Acuna
CSCI 4020
Anshelevich

Computer Algorithms Homework 2

1. Prove the conjectures:

a. The minimum spanning tree of G, with respect to the edge weights a_e, is a minimum-altitude connected subgraph

b. A subgraph (V , $E^{'}$) is a minimum-altitude connected subgraph if and only if it contains the edges of the minimum spanning tree.

Proof by Contradiction: Suppose we have a cut $(S, V - S)$ of V, there is an edge $e = (u, v)$ which is the edge with the minimum altitude value across cut $(S, V - S)$, and one minimum-altitude connected subgraph (V, E') does not contain e. Suppose the winter-optimal path between u and v in E' is P. Then P must contain at least one edge e' that is across the cut. Since e is the minimum-altitude edge across the cut and edge e and egde $e^{'}$ make a cycle, we have $a_{e'} > a_e$, and the height of P is larger than or equal to $a_{e'}$, therefore larger than a_e, but the height of winter-optimal path in the full graph (V, E) is less than or equal to a_e because edge e makes one path between u and v. This means the height of winter-optimal path in (V, E') is greater than it is in the full graph, which contradicts the definition of minimum-altitude connected subgraph.

Since the minimum spanning tree is composed of edges with minimum altitude values across cuts, directly yields the result. Every minimum-altitude connected subgraph contains all the edges in the minimum spanning tree and because adding edges to a minimum-altitude connected subgraph always yields another minimum-altitude connected subgraph, so we also have the result. Every supergraph of the minimum spanning tree is a minimum-altitude connected subgraph. Both conjectures in the problem are true.

Pablo Acuna
CSCI 4020
Anshelevich

Computer Algorithms Homework 2

2.

a. Show that the following algorithm does not correctly solve this problem, by giving an instance on which it does not return the correct answer. In your example, say what the correct answer is and also what the algorithm above finds.

Month	1	2	3	4
NY	1	1	2	3
SF	2	2	1	1

With an M = 100, the algorithm in the book will return to you the minimum cost sequence of [NY, NY, SF, SF]. The cost of this sequence would be 104. This is not an optimal solution. The optimal solution is [SF, SF, SF, SF] with cost of 6.

b. Give an example of an instance in which every optimal plan must move (i.e., change locations) at least three times. Provide a brief explanation, saying why your example has this property.

Month	1	2	3	4
NY	20	70	25	120
SF	100	10	90	30

With M = 10, the optimal solution sequence is [NY, SF, NY, SF] with a cost of 105. The property that the example has is a low cost of moving plus cost at the moving location be far less than potato staying in the same city.

c. Give an efficient algorithm that takes values for n, M, and sequences of operating costs N_1, ..., N_n and S_1,...,S_n, and returns the cost of an optimal plan.

Subproblems: L[i][1] = Optimal solution when starting at NY at the ith month
L[i][2] = Optimal solution when starting at SF at the ith month

Recursion Relation: L[i][1] = min(L[i-1][2] + M, L[i-1][1] + N_i)
L[i][2] = min(L[i-1][1] + M, L[i-1][2] + S_i)

This relation works because the optimal solution will be the minimum between staying put at a location or moving and adding the cost of moving, M, to the optimal solution for the last month at the other city.

Psuedo Code:
L[n][2]
L[1][1] = N_1
L[1][2] = S_1
for i = 2 to n
    L[i][1] = min(L[i-1][2] + M, L[i-1][1] + N_i)
    L[i][2] = min(L[i-1][1] + M, L[i-1][2] + S_i)
return min(L[n][1], L[n][2])