Davie Truong

I have read and agree to the collaboration policy. Davie Truong

Homework Heavy

CMPS 102 – Spring 2017 – Homework 3

Solution to problem 1

V – set of n small projects

E – set of pair of projects (u, v) if u must be completed before v.

Goal: earliest Completion time, IE. Shortest path in terms of weighted vertex and edges

1. **Algorithm for earliest possible completion time**

Modified-Dijkstra (G, s)

Initialize an empty queue Q

Q <- s // the start vertex is pushed on the queue

currentWeight(s) = 0

while (Q is not empty)

u = Q.pop

for (v adjacent projects to u)

if (currentWeight(u) + cost from (u, v) < currentWeight(v))

currentWeight(v) = currentWeight(u) + cost from (u, v)

v.prev = u

if (v is not already in the queue)

Q <- v

**Description**: This is a modified version of Dijkstra’s algorithm. It runs mostly the same keeping track of the shortest weighted paths from the start project. After it completes its run, it returns data containing the earliest possible completion time and the critical points.

**Proof of Correctness:**

For each project n d(n) is the path of the earliest possible completion time from s – n.

Proof: by induction on V

Base: V = 1 is trivial

Induction: assume true for V = K

* Let V be next project added to V, and let (u, v) be the chosen edge.
* The shortest (s-u) path plus (u, v) is an (s-v) path of the minimum chosen path.
* Consider any (s-v) path P.
* Let (x-y) be the first edge in P that leaves s, and let P’ be the sub path to x
* P is already too long as soon as it leaves s.

Runtime: E log V

The while loop runs for every vertex because the algorithm pushes every node adjacent to the start on the queue and it follows the tree of nodes. Finding and updating each adjacent vertex weight take O (log V) time. Thus, the total time will be E Log V, where E represents (v-1) edges.

Space Complexity:

The algorithm stores a matrix data in an adjacency list format.