I have read and agree to the collaboration policy. Lynne Diep.

Lynne Diep
Homework Heavy Grading
Collaborators: Sabrina

Homework 3-1
Due: May 22, 2017

Give an algorithm that takes the inputs above and returns:

(a) For each project u ∈ V , the earliest possible completion time c(v).

(b) A list of the critical projects.

Algorithm (earliest possible completion time) –
Project (E,V) {

Find all projects, p, in E that does not have a project that must be completed before it starts

Define a "start" project with $t_{start} = 0$. This is similar to a source in maximum flow. And add (start, p_1), (start, p_2), …, (s,p_n) to E

Do BFS however:

1. change the array of distances to an array of times L initiates all times to response to t_n
2. If a node is either white, gray visit.
   If a node is black do not visit
3. Have original time array of t_u and overwrite in time array by –

```
        Add parent time
            If new time > old time
                    Time [] = new time
            Else (new time ≤ old time)
                    Keep old time
```

}

Runtime: O(V+E)
Space Complexity: O(v) or O(n) since size of v is n

Algorithm (list of critical projects) –

Every time add parent time, EXCEPT for when parent = start
Add parent node to initial array
If new time > old time
    Overwrite critical node with new parent node
Return list of critical nodes in end

Runtime: O(V+E)
Space Complexity: O(v) or O(n) since size of v is n

Proof of Correctness:
Both algorithms use BFS, so we will prove the correctness of BFS
We use the lemma of how "all vertices of layer i (at distance from i from s) enter the queue before all vertices of layer i + 1.
We prove the lemma by induction –
BC: Let i = 0, this is the vertex s and the claim is clear by the algorithm.
Induction step: Assume the claim hold for i. We see that all i+1 layer vertices enter the queue before i+2 vertices. Let u be a vertex in i+1 layers and w be a vertex in i+2 layers. By definition there are u_1 and w_1 vertices in layers i and i+1, so that $(u_1, u), (w_1, w) \in E$. By induction hypothesis, u_1 will enter the queue before w_1. When scanning u_1, or even before it, u will enter the queue. This implies that u enters the queue before w as the claim follows.
Using BFS allows us to go through all possible completion times, which ultimately returns the earliest possible completion time and list of critical projects by comparing the current time with each new time. If the new time is greater than the current time, than that new time becomes our choice and overrides critical node/completion time with the parent node. Hence the correctness of this algorithm.