EECS 336 Fall 2015
Homework Problem 1.4

This algorithm finds number of distinct TUbor tours from s to t.

Algorithm 1 Find total number of distinct TUbor paths

```
procedure Find-TUbor(s, t)
    Initialize list L[0] to consist of a dictionary {s : 1}
    Initialize layer counter i ← 0
    TUborDone ← false
    while not TUborDone do
        Insert L[i + 1] = {}, an empty dictionary
        for each node u in L[i] do
            for each directed edge e = (u, v) do
                if v = t then
                    TUborDone ← true
                end if
                if L[i + 1][v]does not exist  then
                    Insert L[i + 1][v] = L[i][u]
                else
                    Update L[i + 1][v] = L[i + 1][v] + L[i][u]
                end if
            end for
        end for
        i ← i + 1
    end while
    return L[i][t]
end procedure
```

Claim: The above implementation runs in $O(n+m)$ for a n-node m-edge G

Proof: Let n_u denotes degree of node u, which is number of edges incident to u. Within the inner For loop, time spent considering edges incident to u is $O(n_u)$, so total time is $O(\sum_{u \in G} n_u) = 2m$. Thus total time on edges is $O(m)$. For the outer For loop, list $L[i]$ needs $O(n)$ time to set up. Thus total time is $O(m+n)$.

Claim: The above implementation can obtain correct number of shortest paths given s and t.

Proof:

a) Alg. implementation produces shortest path.

   By contradiction. Suppose there is at least one path with a shorter length n than length of paths from the alg., m, then the supposed shortest length n is at most $m-1$. In each iteration i of the While loop, the search is propagated by one level from $L[i]$ to $L[i+1]$. The

While loop will be terminated at iteration $i = m - 1$ to generate $L[m]$, when t is discovered for the first time. Thus, t is not in any of previous levels of $L[j](j = 0, 1, ..., m - 1)$. Since the shortest path length is at the level of t, then the shortest length is at least m, or $n \geq m$, which contradcits with $n < m$. Thus alg. produces shortest path.

b) The counted number of shorted paths, $L[m][t]$, is correct. Here, m is length of shortest path.

Proof by induction on m:

(a) Base case: $m = 1$, s is directly connected to t. Then $L[1][t] = 1$. Correct.

(b) Induction hypothesis: value in $L[n][t]$ is correct.

(c) Induction step: value in $L[n + 1][t]$ is also correct.

(d) Induction case: Let $v_1, v_2, ..v_k$ be all the vertices that are both in $L[n]$ and incident to vertex u, which is in $L[n + 1]$. According to induction hypothesis, from s to $v_j(j = 1, 2.., k)$, there are $L[n][v_j]$ distinct paths, and from v_j to u there is only one path (v_j, u). so there should be $L[n][v_j]$ paths from s to u through v_j. As all the paths from s to u have to go through one or more v_j's, the total number of shortest paths is $L[n][v_j]$ plus number of v_j's that are incident to u, which is the same as $L[n + 1][u]$ implemented in the alg. at line 12-15. Since t belongs to set of u's in $L[n + 1]$, $L[n + 1][t]$ is also correct number of distinct path for level $n + 1$.