Problem 1

**General Idea:**

As all nodes in a strongly connected component (SCC for short) have access to each other. My idea is take the problem as a graph, the set of tasks as nodes, and their direct dependencies as directed edge. Use **tarjan algorithm[1]** to find strongly connected components in this graph. If there is only one element in SCC, no extra effect C is needed, however if there are at least two elements in SCC, the extra effect will be **C\*k\*(k-1)/2** , where k is the number of elements in this SCC. Sum up all extra effect C needed for all SCC of this graph and add n\*D, where n is the number of task in this problem is the final result.

**Algorithm:**

Procedure Tarjan(G,V)**[2]**

Input: graph G =(V, E)

Output: set of strongly connected components (sets of vertices)

index := 0

S := empty

for each v in V do

if (v.index is undefined) then

strongconnect(v)

end if

end for

function strongconnect(v)

// Set the depth index for v to the smallest unused index

v.index := index

v.lowlink := index

index := index + 1

S.push(v)

// Consider successors of v

for each (v, w) in E do

if (w.index is undefined) then

// Successor w has not yet been visited; recurse on it

strongconnect(w)

v.lowlink := min(v.lowlink, w.lowlink)

else if (w is in S) then

// Successor w is in stack S and hence in the current SCC

v.lowlink := min(v.lowlink, w.index)

end if

end for

// If v is a root node, pop the stack and generate an SCC

if (v.lowlink = v.index) then

start a new strongly connected component

repeat

w := S.pop()

add w to current strongly connected component

until (w = v)

output the current strongly connected component

end if

end function

**Prove algorithm has linear time[3]**

The Tarjan procedure is called once for each node; the forall statement considers each edge at most twice. The algorithm's running time is therefore linear in the number of edges and nodes in G, i.e. O(|V|+|E|)

Reference

[1][2][3] Tarjan’s strongly connected components algorithm http://en.wikipedia.org/wiki/Tarjan's\_strongly\_connected\_components\_algorithm