Q5. A transitive reduction of a directed graph G (V, E) is any graph G' with the same vertices but with minimum edges such that the transitive closure of G and G' are the same. How can G' be obtained,if G is given? Write the procedure.

We are assuming the given graph G(V, E) to be a directed acyclic graph and that the graph is represented using adjacency matrix. Here, n represents number of nodes in the graph and m represents the number of edges in the graph. Also, we are assuming that the nodes are labeled from 1 to n.

First we will find the "path matrix". This is a n x n matrix in which the entry at (i, j) will be 1 if and only if there is a path from node i to node j, otherwise it will be 0. It is easy to see that such a matrix will represent the transitive closure of the given graph.

Formally, a transitive closure of a graph G is another graph G1 such that there is an edge between node u, v if there is some path from u to v in graph G.

To find the path matrix, we will use warshall's algorithm. It takes the adjacency matrix as input and modifies it to give another matrix which represents the transitive closure of the graph. Its pseudo code is as follows.

```
# adjMat: adjacency matrix of the graph
# n: number of nodes in graph

TRANS_CLOSURE(adjMat, n):
FOR i = 1 to n:
    FOR j = 1 to n:
        IF i == j:
            SKIP
        IF adjMat[j][i] == 1:
            FOR k = 1 to n:
                IF adjMat[j][k] == 1:
                    adjMat[j][k] = adjMat[i][k]
```

The above algorithm modifies the adjacency matrix and now it represents the transitive closure of the given graph. Its time complexity is cubic.

Now, we need to find the transitive reduction of the given graph G.

Formally, the transitive reduction of the graph G is defined to be another graph, G1 such that the transitive closure of G and G1 are same and there is no other such graph with fewer arcs than G1 which satisfies this condition.

The main idea of finding the transitive reduction is to construct a graph which will have the same reachability as the original graph but has as few edges as possible. Note that if there is an edge from node i to node j and also an edge from node j to k, then we can remove the edge(if it exists) from node i to node k because, the two edges; i -> j and j -> k; are enough to form a path from node i to k. Hence the transitive closure of these two edges will be same as the transitive closure of edges i -> j, j-> k and i-> k.

Therefore, we will first create a graph which will have all the edges as contained in the transitive closure of graph G. Then we will remove the "redundant" edges as explained in the above paragraph to have as few edges as possible. The transitive closure can be obtained from warshall's algorithm as explained above. Then the following procedure will be used to remove the redundant edges in the same manner as explained in previous paragraph.

```
# pathMat: path matrix of the graph obtained via Warshall's algorithm
# n: number of nodes in graph

TRANS_REDUCE(pathMat, n):
    FOR j = 1 to n:
        FOR i = 1 to n:
            IF pathMat[i][j] == 1:
                FOR k = 1 to n:
                    IF pathMat[j][k] == 1:
                        m[i][k] = 0
```

The time complexity of this algorithm is also cubic.

Reference:

1. Harry Hsu. "An algorithm for finding a minimal equivalent graph of a digraph.", Journal of the ACM, 22(1):11-16, January 1975