CO322 Data Structures and Algorithms - 2019

Lab 5 - Graph ADT

Registration number : E/16/086

Name : A.L.V.H.Dharmathilaka

Transitive Closure

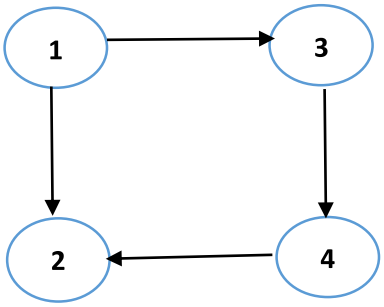
1. Find out what is the Transitive Closure of a graph.

For vertices v and w in given graph G, G* (Transitive closure graph) has an edge from v to w if and only if there is a directed path from v to w in G. This reach-ability matrix is called the transitive closure of a graph. This matrix has following properties.

- Not symmetric
- Supports O(1) reachability queries with O(V^2) space.

2. Manually compute the Transitive Closure for the following graph:

Manuall compute the transitive matrix by looking at the graph :

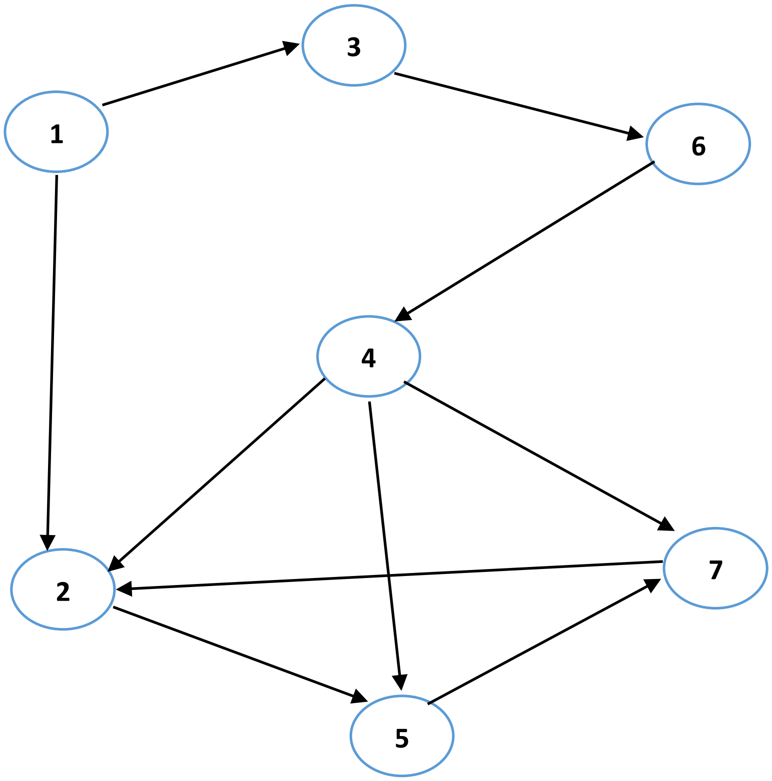


$$\begin{pmatrix} 1&1&1&1 \\ 0&1&0&0 \\ 0&1&1&1 \\ 0&1&0&1 \end{pmatrix}$$

Using warshall's algorithm :

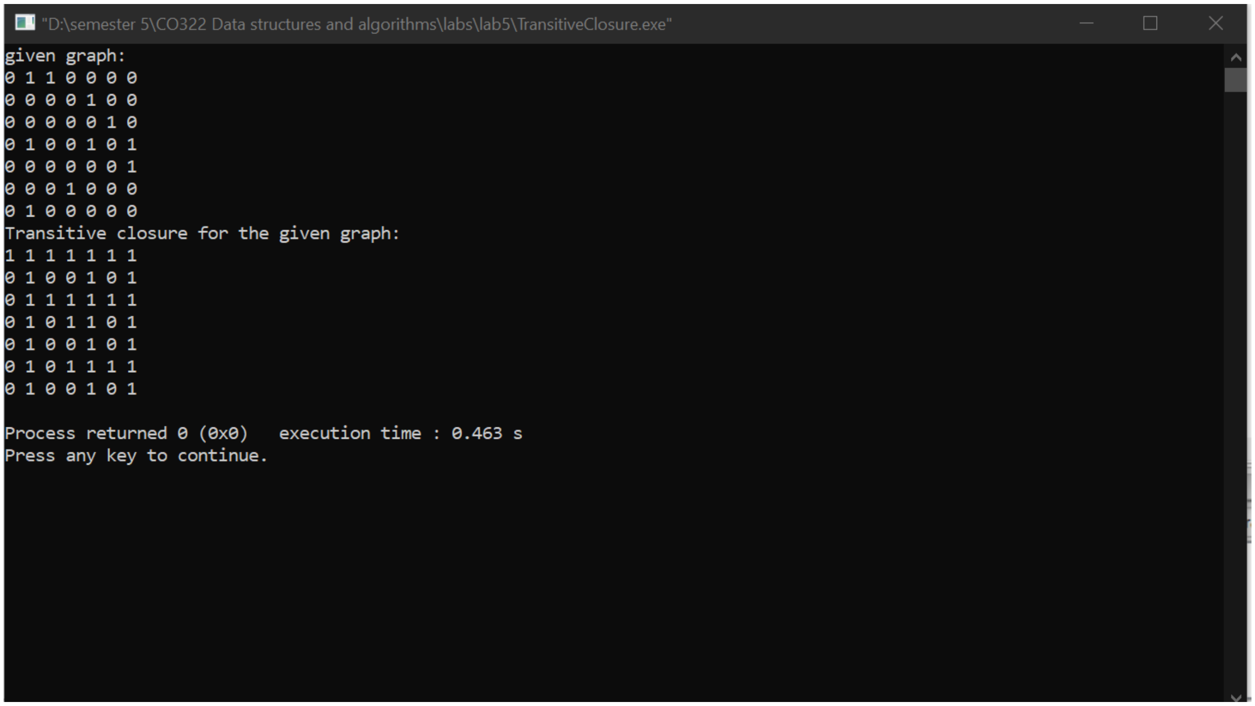
$$\begin{matrix} 1&1&1&0 \\ 0&1&0&0 \\ 0&1&1&1 \\ 0&1&0&1 \end{matrix} \rightarrow \begin{matrix} 1&1&1&0 \\ 0&1&0&0 \\ 0&1&1&1 \\ 0&1&0&1 \end{matrix} \rightarrow \begin{matrix} 1&1&1&0 \\ 0&1&0&0 \\ 0&1&1&1 \\ 0&1&0&1 \end{matrix} \rightarrow \begin{matrix} 1&1&1&1 \\ 0&1&0&0 \\ 0&1&1&1 \\ 0&1&0&1 \end{matrix} \rightarrow \begin{pmatrix} 1&1&1&1 \\ 0&1&0&0 \\ 0&1&1&1 \\ 0&1&0&1 \end{pmatrix}$$

3. Based on the Graph Traversal algorithm discussed in the class, write a C program to compute and print the Transitive Closure of a given graph. Use the following graph to test your program:



Code to test the given graph:

```
#include <stdio.h>
#define S 7

//Initially the matrix size has defined to 7x7
//print a Matrix
void printMatrix(int graph[S][S]){

    for (int i = 0 ; i < S ; i++){

        for ( int j = 0 ; j < S ; j++){
           printf("%d ",graph[i][j]);
        }
        printf("\n");
    }

}
//Calculate transitive closure using warshall’s algorithm
void transitiveClosure(int graph[S][S]){

    int reachMatrix[S][S];

    //copying the current matrix to the Transitive closure
    for (int i = 0; i < S; i++)
        for ( int j = 0; j < S; j++)
            reachMatrix[i][j] = graph[i][j];

    for (int m = 0 ; m < S ; m++){
        for (int i = 0 ; i < S ; i++){
            for ( int j = 0 ; j < S ; j++){
                if(i==j){
                   reachMatrix[i][j] = 1; //diagonal elements are set to one
                }
                else{
                    //set reachMatrix[i][j] =1 if i to j has a directed path,0 otherwise.
                   reachMatrix[i][j] = (reachMatrix[i][m] && reachMatrix[m][j]) ||
reachMatrix[i][j];
                }
            }
        }
    }
    printf("Transitive closure for the given graph:\n");
    printMatrix(reachMatrix);
}

int main(){

    int graph[S][S] = { {0, 1, 1, 0, 0, 0, 0},
                    {0, 0, 0, 0, 1, 0, 0},
                    {0, 0, 0, 0, 0, 1, 0},
                    {0, 1, 0, 0, 1, 0, 1},
                    {0, 0, 0, 0, 0, 0, 1},
                    {0, 0, 0, 1, 0, 0, 0},
                    {0, 1, 0, 0, 0, 0, 0}
                   };
    printf("given graph:\n");
    printMatrix(graph);  //print given matrix
    transitiveClosure(graph); //get the trasitive closure




return 0;
}
```


"D:\semester 5\CO322 Data structures and algorithms\labs\lab5\TransitiveClosure.exe"

```
given graph:
0 1 1 0 0 0 0
0 0 0 0 1 0 0
0 0 0 0 0 1 0
0 1 0 0 1 0 1
0 0 0 0 0 0 1
0 0 0 1 0 0 0
0 1 0 0 0 0 0
Transitive closure for the given graph:
1 1 1 1 1 1 1
0 1 0 0 1 0 1
0 1 1 1 1 1 1
0 1 0 1 1 0 1
0 1 0 0 1 0 1
0 1 0 1 1 1 1
0 1 0 0 1 0 1

Process returned 0 (0x0)   execution time : 0.463 s
Press any key to continue.
```