Termwork 7

Implement All-pairs Shortest Paths Problems using Floyd's Algorithm.

Objective of the Experiment :

- To introduce the concept of dynamic programming
- Present the working of floyd's Algorithm
- To find the shortest path form all nodes to all other nodes in a given graph
- Analyze the Algorithm Complexity.

Theory:

Dynamic programming:

- It is a technique for solving problems with overlapping sub problems
- It suggests solving each of the smaller sub-problems only once and recording the results in a table from which a solution to be the original problem can then be obtained

Transitive Closure :

If there exists an edge between V_i and V_K and also between V_K to V_j then there is a path between V_i and V_j

Graphs:

Is a non-linear data structure consisting of nodes and edges. Consists of finite set of vertices and set of Edges which connect a pair of nodes.

Weight Matrix :

$W(i,j) = 0$ if $i=j$

$W(i,j) = \infty$ if no edge between i and j

$W(i,j) =$ "weight of edge"

Algorithm:

input: Weight Matrix W of a graph with no negative-length cycle

output: The distance matrix of the shortest path's length

```
begin
    D ← W , P[i,j] ← 0
    for k ← 1 to n, do
        for i ← 1 to n, do
            for j ← 1 to n, do
                D[i,j] ← min { D[i,j], D[i,k] + D[k,j] },
    return D
end.
```

Program:

```
#include<stdio.h>
void printMatrix (int D[10][10], int n){
    int i,j;
    for(i=0 ; i<n; i++){
        for(j=0; j<n; j++)
            printf("%d ", D[i][j]);
        printf("\n");
    }
}

void floyds (int D[10][10], int n){
    int i,j,k;
    for (k=1; k<=n; k++){
        printf("With %d as intermediate vertex \n", k);
        printf("Cost Matrix now: \n");
        for( i=1; i<=n; i++)
            for(j=1; j<=n; j++)
                if(i==j)
                    D[i][j] = 0;
                else
                    D[i][j] = min(D[i][j],D[i][k]+ D[k][j]);
        printMatrix (D,n);
    }
}

int minimum (int a, int b){
    return (a<b)? a:b;
}
```

```
int main(){
    int D[10][10], w,n,e,u,v, i,j;
    clrscr();
    printf("\n Enter the number of vertices: ");
    scanf("%d",&n);

    printf("\n Enter the Cost Matrix: \n");
    for(i=1; i<=n; i++)
        for(j=1; j<=n; j++)
            scanf("%d", &D[i][j]);
    printf("\n The initial cost matrix: \n");
    printMatrix(D,n);
    printf("\n The shortest paths are: \n");
    for(i=1 ; i<=n; i++)
        for(j=1; j<=n; j++){
            if(i!=j)
                printf("\n <%d, %d> ===> %d $", i, j, D[i][j]);
        }
}
```

References :

- Kenneth Berman, Jerome Paul, Algorithms, Cengage learning
- Thomas H Cormen, Charles E. Leiserson, Ronal L Rivest, Clifford Stein, Introduction to Algorithms, PHI, 2nd Edition & Onwards.

Conclusion:

- In this Termwork, we learnt about Dynamic Programming, Graphs, Floyd's Algorithm and applied to solve this problem.
- We also learnt ~~computing~~ the basic problem solving techniques.