Assignment #4

1. Traveling Salesman Problem (TSP)

Date of Performance: 09/07/2024

Date of Submission: 17/09/2024

Student ID: 20220104108

Name: Mashrur Rahman

Group: C1

Algorithm\TSP.cpp

```cpp
#include <bits/stdc++.h>
#include <tuple>
using namespace std;
#define inf 99999
int reduce(vector<int> &row)
{
    int min = inf + inf;
    for (int i = 0; i < row.size(); i++)
    {

        if (row[i] < min && row[i] != inf)
            min = row[i];
    }
    for (int i = 0; i < row.size(); i++)
    {
        if (row[i] != inf)
            row[i] -= min;
    }

    return min;
}

int main()
{
    int n, e;
    cout<<"Node and Edges: ";
    cin >> n >> e;
    cout<<"From To Cost: "<<endl;

    vector<vector<pair<int, int>>> adj(n + 1); // this is for 1 based graph
    for (int i = 0; i < e; i++)
    {
        int u, v, w;
        cin >> u >> v >> w;
        adj[u].push_back({v, w});
    }
    vector<vector<int>> r, main;
    for (int i = 0; i <= n; i++)
    {
        vector<int> t(n + 1, inf);
        r.push_back(t);
    }
    for (int i = 1; i <= n; i++)
    {
        for (int j = 0; j < adj[i].size(); j++)
        {
            r[i][adj[i][j].first] = adj[i][j].second;
        }
    }
    main = r;//this is for saving my cost
    int bound = 0;
    for (int i = 1; i <= n; i++)
    {
        vector<int> t;
        for (int j = 1; j <= n; j++)
        {
            t.push_back(r[i][j]);
        }
        bound = bound + reduce(t);
        for (int j = 1; j <= n; j++)
        {
            r[i][j] = t[j - 1];
        }
    }

    for (int i = 1; i <= n; i++)
    {
        vector<int> t;
        for (int j = 1; j <= n; j++)
        {
            t.push_back(r[j][i]);
        }
        bound = bound + reduce(t);
        for (int j = 1; j <= n; j++)
```

```
            {
                r[j][i] = t[j - 1];
            }
        }

        vector<int> vis(n + 1, 0), ans;
        ans.push_back(1);
        int itr = 1, node = 1;


        while (itr <= n-1)
        {
            vis[node] = 1;
            vector<tuple<int, int,vector<vector<int>>>> temp;

            for (int i = 1; i <= n; i++)
            {
                int bound_temp = bound;
                if (vis[i] == 1)
                    continue;
                if (r[node][i] >= inf)
                    continue;
                vector<vector<int>> t;
                t = r;

                bound_temp = bound_temp + t[node][i];

                t[i][node] = inf;

                for (int j = 1; j < t.size(); j++)
                {
                    t[node][j] = inf;
                }
                for (int j = 1; j < t.size(); j++)
                {
                    t[j][i] = inf;
                }

            for (int j = 1; j <= n; j++)
            {
                vector<int> t_rc;
                for (int k = 1; k <= n; k++)
                {
                    t_rc.push_back(t[j][k]);
                }

                int red = reduce(t_rc);
                if (red >= inf)
                    bound_temp = bound_temp;
                else
                    bound_temp = bound_temp + red;

                for (int k = 1; k <= n; k++)
                {
                    t[j][k] = t_rc[k - 1];
                }
            }

            for (int j = 1; j <= n; j++)
            {
                vector<int> t_rc;
                for (int k = 1; k <= n; k++)
                {
                    t_rc.push_back(t[k][j]);
                }

                int red = reduce(t_rc);
                if (red >= inf)
                    bound_temp = bound_temp;
                else
                    bound_temp = bound_temp + red;

                for (int k = 1; k <= n; k++)
                {
                    t[k][j] = t_rc[k - 1];
                }
```

```cpp
            }

            temp.push_back({bound_temp, i, t});



        }
        sort(temp.begin(), temp.end());

        auto it = temp[0];
        r = get<2>(it);
        ans.push_back(get<1>(it));
        node = get<1>(it);



        itr++;
    }

    int sum = main[ans[ans.size() - 1]][1];
    cout<<"optimal path is: ";
    for (int i = 0; i < ans.size(); i++)
    {
        if (i > 0)
        {

            sum = sum + main[ans[i - 1]][ans[i]];
        }
        cout << ans[i] << "-> ";
    }

    cout << 1 << endl;

    cout<<"And the cost is: " << sum;
}

// 5 11
// 1 2 9
// 1 4 8
// 2 3 4
// 2 5 2
// 3 2 3
// 3 4 4
// 4 2 6
// 4 3 7
// 4 5 12
// 5 1 1
// 5 4 10

// another input

// 4 10
// 1 2 4
// 1 3 12
// 1 4 7
// 2 1 5
// 2 4 18
// 3 1 11
// 3 4 6
// 4 1 10
// 4 2 2
// 4 3 3
```