Bharatiya Vidya Bhavan's
Sardar Patel Institute of Technology
Bhavan's Campus, Munshi Nagar, Andheri (West), Mumbai-400058-India
(Autonomous College Affiliated to University of Mumbai)

Name	Tejas Jadhav
UID No.	2022301006
Class	COMPS A (B batch)
Experiment No.	10

Aim: To study approximation algorithm by implementing vertex cover problem

Theory:

The Vertex Cover Problem is a classic optimization problem in computer science. It is an NP-hard problem that asks to find the minimum set of vertices in a graph such that all edges are adjacent to at least one vertex in the set. In this experiment, we aim to study approximation algorithms for the Vertex Cover Problem.

- **NP Hardness**

The Vertex Cover Problem is known to be NP-hard, which means that there is no known algorithm that can solve it in polynomial time. Therefore, we need to resort to approximation algorithms to find a solution that is close to the optimal solution.

- **Approximation Algorithm**

An approximation algorithm is an algorithm that finds a solution that is guaranteed to be within a certain factor of the optimal solution. The factor is determined by the function rho(n), which is the maximum ratio of the solution found by the algorithm to the optimal solution for any instance of size n.

To be considered a valid approximation algorithm, it needs to satisfy two conditions. First, it needs to have a polynomial running time. Second, it needs to find a solution that is within a factor of c/c* of the optimal solution, where c is the solution found by the algorithm and c* is the optimal solution.

- **Vertex Cover Problem**

In the Vertex Cover Problem, we are given an undirected graph G = (V, E), where V is the set of vertices and E is the set of edges. A vertex cover is a subset of vertices C ⊆ V such that every edge in E is adjacent to at least one vertex in C. The goal is to find the minimum size vertex cover.

Algorithm:

APPROX-VERTEX-COVER(G)

1 $C = \emptyset$
2 $E' = G.E$
3 **while** $E' \neq \emptyset$
4 let (u, v) be an arbitrary edge of E'
5 $C = C \cup \{u, v\}$
6 remove from E' every edge incident on either u or v
7 **return** C

Code:

```
#include <bits/stdc++.h>
using namespace std;

struct Edge {
    char u;
    char v;

    Edge() {
        u = 0;
        v = 0;
    }

    Edge(char u, char v) {
        this->u = u;
        this->v = v;
    }
};

set<char> vertexCover(vector<Edge>& edges) {
```

```
    set<char> cover;
    vector<Edge> edges_copy = edges;
    srand(time(NULL));

    while (!edges_copy.empty()) {
        int i = rand() % edges_copy.size();
        Edge e = edges_copy[i];
        edges_copy.erase(edges_copy.begin() + i);

        cover.insert(e.u);
        cover.insert(e.v);

        cout << "Adding edge " << e.u << " " << e.v << "\n";
        for (int i = 0; i < edges_copy.size(); i++) {
            if (edges_copy[i].u == e.u || edges_copy[i].v == e.u ||
                edges_copy[i].u == e.v || edges_copy[i].v == e.v) {
                cout << "Removing edge " << edges_copy[i].u << " "
                     << edges_copy[i].v << "\n";
                edges_copy.erase(edges_copy.begin() + i);
                i--;
            }
        }
    }
    return cover;
}

int main() {
    cout << "Enter the number of edges then enter each edge in the format
\"u "
            "v\" where u and v are vertices of the edge.\n";
    int n;
    cin >> n;
    vector<Edge> edges(n);
    for (int i = 0; i < n; i++) {
        char u, v;
        cin >> u >> v;
        edges[i] = Edge(u, v);
    }

    cout << "\nRunning Approximate Vertex Cover Algorithm...\n";
    set<char> cover = vertexCover(edges);
```

```
    cout << "\nVertex Cover: { ";
    for (char v : cover) {
        cout << v << ", ";
    }
    cout << "}\n" << endl;
    return 0;
}

// sample input
// 8
// a b
// b c
// c d
// c e
// d e
// d f
// d g
// e f
```

Output:

```
PS D:\Tejas\clg\daa\Experiment 10\code> ./a
Enter the number of edges then enter each edge in the format "u v" where u and v are vertices of the edge.
8
a b
b c
c d
c e
d e
d f
d g
e f

Running Approximate Vertex Cover Algorithm...
Adding edge d e
Removing edge c d
Removing edge c e
Removing edge d f
Removing edge d g
Removing edge e f
Adding edge b c
Removing edge a b

Vertex Cover: { b, c, d, e, }

PS D:\Tejas\clg\daa\Experiment 10\code>
```

Conclusion:

In conclusion, the Vertex Cover Problem is an NP-hard problem that can be solved using approximation algorithms. The greedy algorithm is a simple and efficient algorithm that provides a guaranteed approximation ratio of 2 for the problem.