Suraj Iyer
2021300045
SE Comps A, Batch C

DAA Experiment 9

Aim - Approximate solution for vertex cover problem

Details - A vertex cover of an undirected graph is a subset of its vertices such that for every edge (u, v) of the graph, either ‘u’ or ‘v’ is in the vertex cover. Although the name is Vertex Cover, the set covers all edges of the given graph. Given an undirected graph, the vertex cover problem is to find minimum size vertex cover.

Consider all the subset of vertices one by one and find out whether it covers all edges of the graph. For eg. in a graph consisting only 3 vertices the set consisting of the combination of vertices are:{0,1,2,{0,1},{0,2},{1,2},{0,1,2}} . Using each element of this set check whether these vertices cover all the edges of the graph. Hence update the optimal answer. And hence print the subset having the minimum number of vertices which also covers all the edges of the graph.

It can be proved that the above approximate algorithm never finds a vertex cover whose size is more than twice the size of the minimum possible vertex cover.

The Time Complexity of the above algorithm is O(V + E). The space complexity of this solution is O(V), where V is the number of vertices of the graph. This is because we are using an array of size V to store the visited vertices

Code -

```
#include <stdio.h>
#include <stdlib.h>

int min(int x, int y) { return (x < y)? x: y; }

struct node
{
	int data;
	struct node *left, *right;
};

int vCover(struct node *root)
{
	if (root == NULL)
		return 0;
	if (root->left == NULL && root->right == NULL)
		return 0;

	int size_incl = 1 + vCover(root->left) + vCover(root->right);

	int size_excl = 0;
	if (root->left)
	size_excl += 1 + vCover(root->left->left) + vCover(root->left->right);
	if (root->right)
	size_excl += 1 + vCover(root->right->left) + vCover(root->right->right);

	return min(size_incl, size_excl);
}

struct node* newNode( int data )
{
	struct node* temp = (struct node *) malloc( sizeof(struct node) );
	temp->data = data;
	temp->left = temp->right = NULL;
	return temp;
}
```

```
int main()
{
        struct node *root             = newNode(20);
        root->left                    = newNode(8);
        root->left->left              = newNode(4);
        root->left->right             = newNode(12);
        root->left->right->left = newNode(10);
        root->left->right->right = newNode(14);
        root->right                   = newNode(22);
        root->right->right   = newNode(25);

        printf ("Size of the smallest vertex cover is %d ", vCover(root));

        return 0;
}
```

Output -

```
Size of the smallest vertex cover is
3
```

Conclusion - I have understood the concept of finding the smallest vector cover for a given graph and implemented the same in C programming language.