CMP 3005 Analysis of Algorithm

PROJECT REPORT

Project:

Writing an efficient algorithm which calculates the maximum clique number in the hamming graph between two vertices.

Methods used in project:

1. distance (int x, int y) :

This method calculates the Hamming distance between integers ‘x’ and ‘y’, which is the number of positions at which the corresponding bits are different.

2. depth_first_search (int z , int&size) :

This method perform a depth-first search (DFS) starting from vertex ‘z’ and updates the size of the current connected component, which is passed as a reference parameter ‘size’.

3. main ( ) :

This is the main method that is executed when the program is run. It reads in ‘a’ and ‘b’ from the user, generates the graph using the ‘distance()’ method , and then finds a maximal clique in the graph using the ‘depth_first_search()’ method. It then prints the size of the maximal clique.

Programming Language:

C++

Libraries are Used in Project:

- iostream
- vector
- cmath

Algorithm is Used in Project:

Depth First Search

A standart DFS implementation puts each vertex of the graph into one of two categories ; Visited and Not Visited

The purpose of the algorithm is to mark each vertex as visited while avoiding cycles.

The DFS algorithm Works as follows:

I. Start by putting any one of the graph's vertices on top of a stack
II. Take the top item of the stack and add it to the visited list
III. Create a list of that vertex's adjacent nodes. Add the ones which aren't in the visited list to the top of the stack.
IV. Keep repeating steps 2 and 3 until the stack is empty.

Complexity of DFS:

The time complexity of the DFS algorithm in this project is O((2^n) ^2). This is because the program generates an adjacent list for a graph with 2^n vertices, and for each vertex it considers every other vertex as a possible neighbor, which takes O(2^n) time. This happens in the for loop. Then the program performs a DFS on the graph, which takes O(V+E) time, where V is the number of vertices and E is the number of edges in the graph. Since the graph has 2^n vertices and each vertex has at most 2^n-1 edges, the overall time complexity is O((2^n) ^n).

```
int distance(int x, int y) {
    int d = 0;
    for (int i = 0; i < a; i++) {
        if ((x & (1 << i)) != (y & (1 << i))) d++;
    }
    return b;
}
```

- Calculates the hamming distance between x and y vertices

```
cout << "Enter a and b: ";
cin >> a >> b;
for (int i = 0; i < (1 << a); i++) {
    for (int j = i + 1; j < (1 << a); j++) {
        if (distance( x: i,  y: j) >= b) {
            adj[i].push_back(j);
            adj[j].push_back(i);
        }
    }
}
```

- Generate the graph

```
int ans = 0;
for (int i = 0; i < (1 << a); i++) {
    if (!vis[i]) {
        int size = 0;
        depth_first_search( z: i,  &: size);
        ans = max(ans, size);
    }
}
```

- Finds a maximal clique