EE878C High Level synthesis Algorithm for CAD

Homework #1

Student ID : 20023404
Name : Lee, Dongsoo

Title : Searching for all maximum cliques (exact solution)

1. General Description

This program finds all maximum cliques by recursive programming and using matrix saves edge information.

Every vertex, which goes from 0-vertex to final vertex, is assumed that this is included or not, two ways.

When we arrive at final vertex, we check two things. The number of Selected vertices and whether there is another vertex which can be added to this list are calculated.

This way follows 'branch and bounding' algorithm and is simplified by lower bound by checking whether a vertex can be a part of clique(if not, go back upper vertex).

2. Input Format

Firstly, the number of vertices is inputted. And then we can input the edge by two vertex, which can be from 0 to number of vertices minus 1. Because this program assume undirected edge, matrix must fill two blanks by one-time input.

ex)

```
Number of Vertices(Max=50): 5

Vertex is from 0 to 4

Input Undirected Edges [vertex vertex], end -> 99 99

0 2
0 3
1 2
1 3
1 4
2 3
2 4
3 4
99 99
```

3. Output Format

This program shows all maximum cliques by lists of selected vertices.

ex) The result of inputted information above

```
< All Maximum Cliques >

 1 2 3 4
 0 2 3
```

4. Program Source Code

```
-- Source file is attached with this by email
-- Detailed description is shown by comments
-- This program is based on ANSI-C language
-- Tested on Solaris 5.8 OS


//
// EE878C High Level Synthesis Algorithm for CAD
//
// Homework1 : programming (Maximum Clique)
// Stduent ID : 20023404 Name : Lee, Dongsoo
//
// Input Format  : 1. Number of Vertices
//                 2. Edges (undirected) by two vertices
// Output Format : Maximum cliques (list of included vertices)
//

#include <stdio.h>

int edge_table[50][50]; // To save edges
int check_vertex[50];   // To test for each vertex
int n;                  // Number of total vertices

void max_clique(int vertex)
{
    // num : Number of selected vertices
    // num2 : To compare with num for checking maximum clique

    int i,j,num,flag2,num2;

    // Condition for end is when vertex is n
    if(vertex == n) {

        num = 0;
        // Count selected vertices
        for(i=0;i<n;i++)
            if (check_vertex[i]) num++;
        if (num>2) {    // only over two vertices selected

            flag2 = 0;

            // check whether this is maximum clique
            // by test of finding another vertex which can be added
            for(i=0;i<n;i++) {
                if (!check_vertex[i]) { // test vertex
                    num2 = 0;
                    for(j=0;j<n;j++)
                        // count vertices which are connected with test vertex
                        if(check_vertex[j] && edge_table[i][j]) num2++;
                    // when 'num2' is 'num', test vertex can be added
                    // and this is not maximum clique
                    if (num2 == num) flag2 = 1;
                }
            }

            // if this is maximum clique then print the list
            if (!flag2) {
            for(i=0;i<n;i++)
                    if (check_vertex[i]) printf(" %d",i);
                printf("\n");
```

```
            }
        }
        return;
    }

    // test by not selecting this vertex
    check_vertex[vertex] = 0;
    // test next vertex
    max_clique(vertex + 1);

    flag2 = 0;
    // check whether this vertex can be added to current list('check_vertes')
    for (i=0;i<vertex;i++)
        if (check_vertex[i] && !edge_table[i][vertex]) flag2 = 1;
    // if all connected then test by selecting this vertex
    if (!flag2) {
        check_vertex[vertex] = 1;
        // test next vertex
        max_clique(vertex + 1);
    }
    return;
}

main()
{
    int i,j;
    int flag,v1,v2;

    // initializing
    for(i=0;i<50;i++) {
        for(j=0;j<50;j++)
            edge_table[i][j] = 0;
        check_vertex[50]=0;
    }

    printf("\n Homework #1 : Maximum cliques\n");
    printf(" 20023404 Lee,Dongsoo\n\n");
    printf(" Number of Vertices(Max=50): ");
    scanf("%d",&n);
    printf("\n Vertex is from 0 to %d\n",n-1);
    printf("\n Input Undirected Edges [vertex vertex], end -> 99 99\n\n");

    flag = 0;
    while(flag == 0) {
        scanf("%d %d",&v1,&v2);
        if (v1 == 99 && v2 == 99) flag = 1;     // input ending condition
        else {      // save edges with table
            edge_table[v1][v2] = 1;
            edge_table[v2][v1] = 1;
        }
    }

    printf("\n < All Maximum Cliques >\n\n");

    max_clique(0);
    printf("\n End of Program....\n");
}
```