# 云南大学数学与统计学院 《算法图论实验》上机实践报告

课程名称: 算法图论实验	<b>年级:</b> 2015 级	上机实践成绩:
<b>指导教师:</b> 李建平	姓名: 刘鹏	专业: 信息与计算科学
上机实践名称: 图的所有连通分支	学号: 20151910042	上机实践日期: 2018-12-31
上机实践编号: 2	组号:	

#### 一、实验目的

- 1. 理解图的连通分支的概念,并能写出求一个图的所有连通分支的算法;
- 2. 学会运用图的遍历算法来解决问题。

## 二、 实验内容

- 1. 写出求一个图的所有最大连通分支的算法;
- 2. 用 C 语言编程实现上述算法。

### 三、 实验平台

Windows 10 Pro 1803:

MacOS Mojave.

### 四、 算法设计

对于一个图G = (V, E),其连通分支所构成的集合是一个图集合,在该集合里,任取一个元素 $G_1$ 都是一个连通图,经过如下(1)、(2)的操作

- (1) 把任何属于集合(如果该集合非空) $G.V-G_1.V$ 的顶点v被添加到 $G_1.V$ ,得到一个新图 $G_1^{'}.V$ ;
- (2)把所有属于G.E的、集合 $G_1.V+\{v\}$ 中顶点之间的、不在 $G_1.E$ 的所有边都添加到 $G_1^{'}.V$ 中,得到一个新图 $G_1''.V$ ;

得到的 $G_1''$ .V是不连通的。换言之,每个连通子图都是最大的,再也无法往其中添加顶点了。

通过对这个"极大性"的考察,可以将广度优先遍历算法或者深度优先算法进行修改,添加条件判断语句,使新的算法可以找到所输入图的所有连通分支。该算法调用了实验报告1中的SEARCH算法。

Algorithm 寻找图的所有极大连通分支,记此算法为SUB-GRAPH

Input 图G的邻接矩阵

**Output** 图*G*的所有连通分支

**Begin** 

Step 1 while  $G \neq \emptyset$ :

```
\begin{split} s &= V.\operatorname{\mathbf{pop}} \\ \operatorname{\mathbf{output}} \ G_{sub} &= \operatorname{SEARCH}(G,\ s) \\ G &= G - G_{sub} \end{split}
```

End

### 五、 程序代码

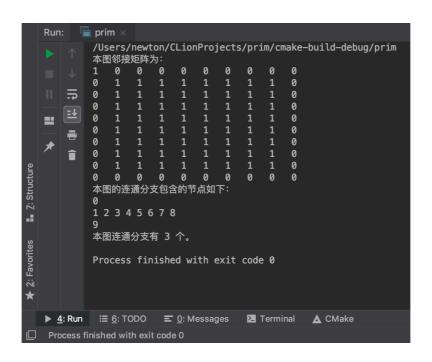
利用 C++语言,对上述算法进行简单实现,并且用一个拥有 10 顶点的图进行测试。

#### 5.1.1 程序代码

```
#include <fstream>
   #include <iostream>
3
   #include <stdlib.h>
4
5
   using namespace std;
6
7
   void visit(int *visited, int num) {
8
       visited[num] = 1;
9
       printf("%d ", num);
10
  }
11
   // 深度优先遍历算法
12
13
   void DFS(int ph[][10], int *visited, int num) {
14
       // visited 是一个数组,表示该节点有没有被访问
15
       if(!visited[num]) {
           visit(visited, num);
16
17
       }
18
       for(int i = num + 1; i < 10; i++) {</pre>
19
20
           if(ph[num][i] == 1) {
21
               if(!visited[i]) {
22
                  visit(visited, i);
23
                  DFS(ph, visited, i);
24
               }
25
           }
26
       }
27
   }
28
29
   int main() {
30
       int ph[10][10];
31
       int visited[10] = {0};
32
       int flag = 0;
33
       char ch;
34
       ifstream OpenFile("tu.txt");
35
       for(int j = 0; j < 10; j++) {</pre>
36
           for(int k = 0; k < 10; k++) {
```

```
37
               OpenFile.get(ch);
38
               ph[j][k] = (int)ch - 48;
39
           }
40
       }
        cout << "本图邻接矩阵为: " << endl;
41
42
       for(int j = 0; j < 10; j++) {
43
           for(int k = 0; k < 10; k++) {
44
               cout << ph[j][k] << "     ";</pre>
45
           }
46
           cout << endl;</pre>
47
       cout << "本图的连通分支包含的节点如下: " << endl;
48
49
       for(int j = 0; j < 10; j++) {</pre>
50
           if(!visited[j]) {
51
               DFS(ph, visited, j);
               cout << endl;</pre>
52
53
               flag++;
54
           }
55
       }
56
57
        cout << "本图连通分支有 " << flag <<" 个。" << endl;
58
59
       OpenFile.close();
60 }
```

#### 5.1.2 运行结果



## 六、 参考文献

[1] 林锐. 高质量 C++/C 编程指南 [M]. 1.0 ed., 2001.

[2]