**实验报告**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **课程名称** | 《算法分析与设计》 | **实验日期** | 2021 年3月 8日 至2021 年 3 月 15日 | | | | |
| **学生姓名** | 王海轩 | **所在班级** | 计算机195 | | | **学号** | 2019212212194 |
| **实验名称** | 采用Prim算法构造最小生成树 | | | | | | |
| **实验地点** | 勤园13-208 | | | **同组人员** |  | | |

# 问题

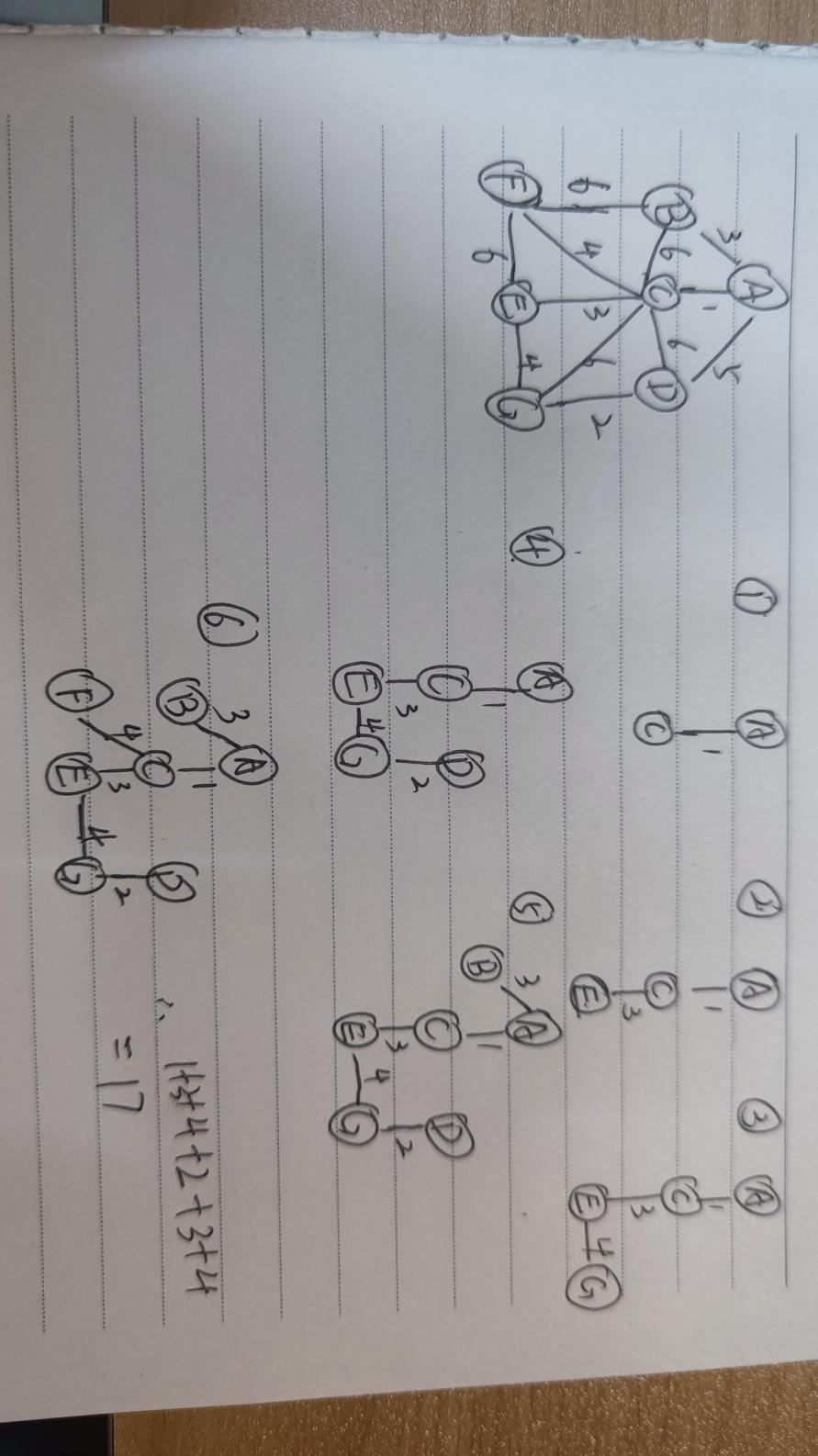
[描述算法问题，首选形式化方式（数学语言），其次才是非形式化方式（日常语言）]

在带权连通图中，每次找与之连线权值最小的顶点，将该点加入最小生成树集合中并构造，直到最后得到一颗最小生成树。  
注意：相同权值任选其中一个即可，但是不允许出现闭合回路的情况。

在下图G<V,E>中，有7个顶点V1-V7，12条边e1-e12

# 解析

[问题的理解和推导，可用电子版直接在此编写，也可用纸笔推导，拍照嵌入本文档]



# 设计

[核心伪代码]

int prim()

{

v[1] = 1;//存入第一个节点

for (int i = 2; i <= N; ++i)

{

v[i] = 0;//其他节点表示为未存入

d[i] = mp[i][1];//权值赋值

}

int sum = 0;

for (int i = 1; i <= N - 1; ++i)

{

int temp = INT\_MAX, pos;

for (int j = 1; j <= N; ++j)

{

if (!v[j] && d[j] < temp) {

temp = d[j];

pos = j;

}//找到数值最小并且尚未存入数组v的权值

}

v[pos] = 1;//计入数组v代表已连通

sum += d[pos];//权值累加

for (int j = 1; j <= N; ++j)//遍历

{

if (!v[j] && d[j] > mp[pos][j] && mp[pos][j] != INT\_MAX)//若未连通并且与pos距离更近，更新数组d

{

d[j] = mp[pos][j];

}

}

}

return sum;

}

# 分析

[算法复杂度推导]

# 源码

[github源码地址]

https://github.com/whx116/lab1-1.git