实验三 聚类技术---复杂网络社团检测

学号：19049100002　　姓名:张泽群　　任课老师：马小科

1. 实验内容

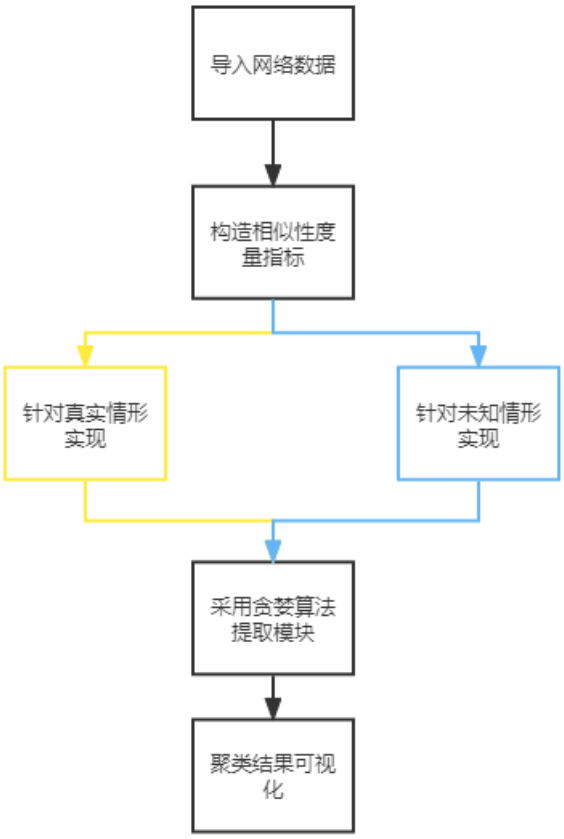
复杂网络是描述复杂系统的有力工具，其中每个实体定义成一个节点，实体间的交互关系定义为边。复杂网络社团结构定义为内紧外松的拓扑结构，即一组节点的集合，集合内的节点交互紧密，与外界节点交互松散。复杂网络社团结构检测广泛的应用于信息推荐系统、致癌基因识别、数据挖掘等领域。

本实验利用马克纽曼提供的真实数据集：跆拳道俱乐部数据由34个节点组成，由于管理上的分歧，俱乐部分解成两个社团。

本实验将数据集转化为邻接矩阵，根据网络结构特征给出节点相似性度量指标，最后用贪婪算法提取社团模块，获得社团聚类结果，最后将聚类结果进行可视化表示。

2. 分析及设计

流程图：



2.1 导入网络数据

我们利用邻接矩阵A来存储网络，0表示无关联，1表示有关联。使用matlab程序进行网络数据的格式转换，gml数据类似json格式，使用正则表达式进行数据的转换即可。

我借鉴了如下网址进行实现 参考实现网址。

2.2 根据网络结构特征给出节点相似性度量指标

对于无权值的图像，我们常用邻域距离来作为相似性度量指标。

给定节点i, 其邻居节点定义为与该节点相链接的所有节点组成的集合，即

$$N(i) = \{j | A_{ij} = 1, j = 1,2,\ldots,n\}$$

给定一对节点（i,j）,其相似性度量可以定义为这个两个节点的邻居节点交集除以邻居节点的并集， 即:

$$S_{ij} = \frac{|N(i) \cap N(j)|}{|N(i) \cup N(j)|}$$

其中| N() |表示集合N中元素的个数。

2.3 采用贪婪算法提取模块

贪婪算法是指在对问题进行求解时，在每一步选择中都采取最优解，从而希望能够导致结果是最优的算法。但不从整体最优上加以考虑，它所做出的仅仅是在某种意义上的局部最优解。

首先定义社团聚类密度的指标，这里我选用了平均相似度距离，其优势在于相对稳定，利用了集合中的全局信息。其公式如下:

$$S(C1, C2) = \frac{\sum_{x \in C1} \sum_{y \in C2} s(x, y)}{|C1| * |C2|}$$

在实现贪婪算法中，首先随机选择一个未聚类的节点作为当前社团C，注意由于节点的随机性，部分初始节点的聚类效果可能很差。然后提取出社团C所有未聚类的邻居节点$N(C)$，并选择使得社团密度降低最小的那个节点v添加到当前社团C，更新当前社团C=C ∪ V。

由于所提供的数据集明确说明只有两个社团，因此我们设计的第一种方法针对于真实情形，等待该过程持续到当前社团的密度小于我们设定的阈值则结束过程，得出聚类结果。

第二种方法是针对未知情况，我们假设事先不知道存在两个社团，因此我们在实现贪婪算法时，一个过程结束时需要重新随机选择一个未聚类的节点作为新的社团，多次重复贪婪选择的过程，直到数据集中的节点都被选择完毕。

2.4 可视化聚类结果

一开始我利用matlab进行绘图实现，但得出的图的视觉效果较差，因此仍然选用采用Cytoscape软件。为使用Cytoscape，我利用matlab导出所需的聚类结果数据，以此实现视觉效果优秀的可视化结果。

3. 详细实现

实验使用的编程语言为*matlab*，编程环境为*MATLAB R2021b*。

3.1 导入网络数据

注：该函数参考了 参考实现网址 中的实现。

```
%% 1.导入网络数据

A = gmlread('karate.gml');
```

3.2 计算相似性指标矩阵

```
%% 2.根据网络结构特征给出节点相似性度量指标

A_size = size(A,1);    % 网络节点个数

S = zeros(A_size,A_size); % 相似性矩阵-领域距离

for i=1:A_size
    for j=1:A_size
        S(i,j) = and_number(A,i,j)/or_number(A,i,j);   % 交集/并集
    end
end
```

3.3 贪婪算法提取模块

注：这里我们把阈值设为0.25，因为经过试验，阈值0.2-0.3得出的聚类效果相对较好。其余阈值时的结果详见'实验结果图'。

第一种实现：详见分析与设计，针对真实情形（已知2个社团)利用贪婪算法，等待算法过程持续到当前社团的密度小于我们设定的阈值则结束过程，得出两个社团的聚类结果。

```
%% 3.采用贪婪算法提取模块

C_all = (1:A_size);     % 定义社团成员全集
C_1 = [];               % 定义社团C1

rand_point = floor(randi(35));   % 选取初始随机点作为当前社团C，不同点得出聚类效果差别可能较大

C_1 = [C_1,rand_point];

C_2 = C_all(ismember(C_all,C_1)==0); % 定义另一个社团C2

now_dens = density_average(S,C_1);   % 利用平均邻域距离作为社团密度

threshold = 0.3; % 设置阈值/可调节

while(length(C_1) <= 32)   % 社团成员无法超过32个

    flag = 0;              % 用于判断是否还存在未聚类的邻居节点
    min_dens_diff = 100;   % 最小密度降低值，初始设为100，即为无穷大
    min_diff_point = 0;    % 最小密度降低节点

    for i=C_1(1:length(C_1))
        for j=C_2(1:length(C_2))

            if A(i,j) == 1 % 提取社团C所有未聚类的邻居节点

                flag = 1;  % 说明存在未聚类的邻居节点

                C_1_temp = [C_1,j];  % 将其放入社团C1
                C_2_temp = C_all(ismember(C_all,C_1_temp)==0); % 从社团C2取出

                temp_dens = density_average(S,C_1_temp); % 求出当前社团密度
                dens_diff = now_dens - temp_dens;        % 求出社团密度降低值

                if dens_diff < min_dens_diff      % 求出社团密度最小降低值和点
                    min_dens_diff = dens_diff;
                    min_diff_point = j;
                end
            end
        end
    end

    if flag == 0  % 如果不存在新的节点与已聚类的节点存在关系，则聚类停止
        break;
    end
```

```
    now_dens = now_dens-min_dens_diff; % 更新当前的社团密度值

    if now_dens >= threshold && min_diff_point~=0   % 社团密度值大于阈值则添
加点
        C_1 = [C_1,min_diff_point];   % 将使得社团模块密度降低最小的那个节点v添
加到社团C_1
        C_2 = C_all(ismember(C_all,C_1)==0); % 更新社团C2
    end

    if now_dens < threshold   % 直到社团密度小于阈值则结束
        break;
    end
end
```

第二种实现：是针对未知情况，我们假设事先不知道存在两个社团，因此我们在实现贪婪算法时，一个过程结束时需要重新随机选择一个未聚类的节点作为新的社团，多次重复贪婪选择的过程，直到数据集中的节点都被选择完毕。

```
while(~isempty(C_candi))   % 候选集不为空

    if isempty(C_single)
        rand_point = C_candi(floor(randi(length(C_candi))));   % 选取候选集
内随机点作为当前社团C
        C_single = [C_single,rand_point];
        C_candi(C_candi==rand_point)=[];
        now_dens = density_average(S,C_single);
    else
        min_dens_diff = 100;   % 最小密度降低值，初始设为100，即为无穷大
        min_diff_point = 0;    % 最小密度降低节点

        for i=C_candi(1:length(C_candi))
            C_temp = [C_single,i];
            temp_dens = density_average(S,C_temp);    % 求出当前社团密度
            dens_diff = now_dens - temp_dens;         % 求出社团密度降低值


            if dens_diff < min_dens_diff              % 求出使得社团密度最小
降低值和点
                min_dens_diff = dens_diff;
                min_diff_point = i;
            end
        end

        temp_dens = now_dens-min_dens_diff;

        if temp_dens < threshold                  % 社团密度小于阈值则将其划分
为一个社团
```

```
            C_set = [C_set,0,C_single];            % 社团间以0为分界
            club_number = club_number + 1;
            C_single = [];
        else
            C_candi(C_candi==min_diff_point)=[];
            C_single = [C_single,min_diff_point]; % 将使得社团模块密度降低最小的那个节点v添加到当前社团
        end
        if isempty(C_candi)                        % 如果候选集最后为空，则划分最后一个社团
            C_set = [C_set,0,C_single];
            club_number = club_number + 1;
            C_single = [];
        end
    end
end
C_set = [C_set,C_single];
```

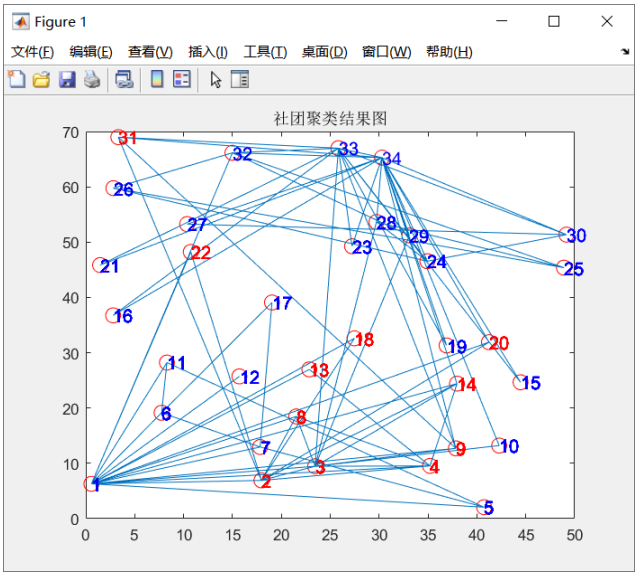
3.4 计算社团模块密度函数值 D

```
% 求取社团模块密度D
module_dens = dens1_number(A,C_1)-
dens2_number(A,C_1)+dens1_number(A,C_2)-dens2_number(A,C_2);
disp(module_dens);
```

3.5 可视化聚类结果

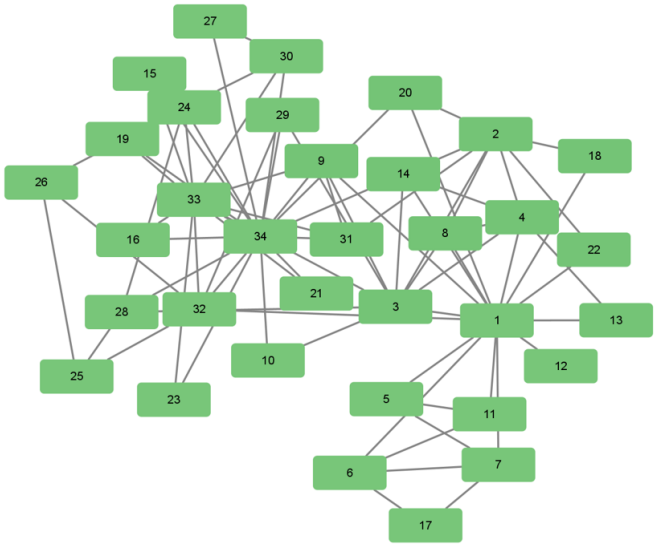
```
% 显示社团聚类结果，由于视觉效果较差，还是使用Cytospace
displayRes(A,C_1);
% 导出画图数据
draw_data(A,C_1);
```

如图得出的社团聚类结果图并不美观，因此弃用。导出社团聚类结果数据，用于Cytoscape画图。



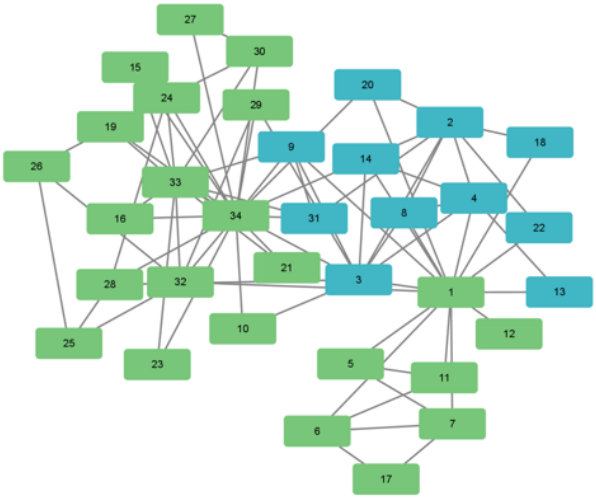
4. 实验结果

1.原始数据图像

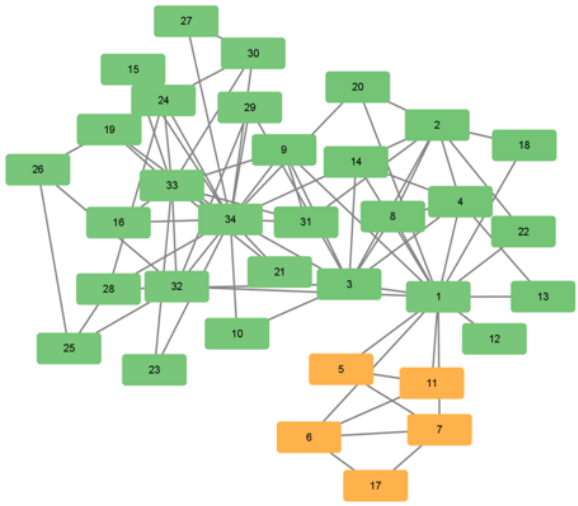


2.实现 ①，阈值=0.25时，通过多次运行程序，主要的存在两种社团模块密度值较好的聚类。其余阈值时的结果详见'实验结果图'。

为[7,6,5,11,17]和[8,4,14,13,2,18,22,20,3,9,31]



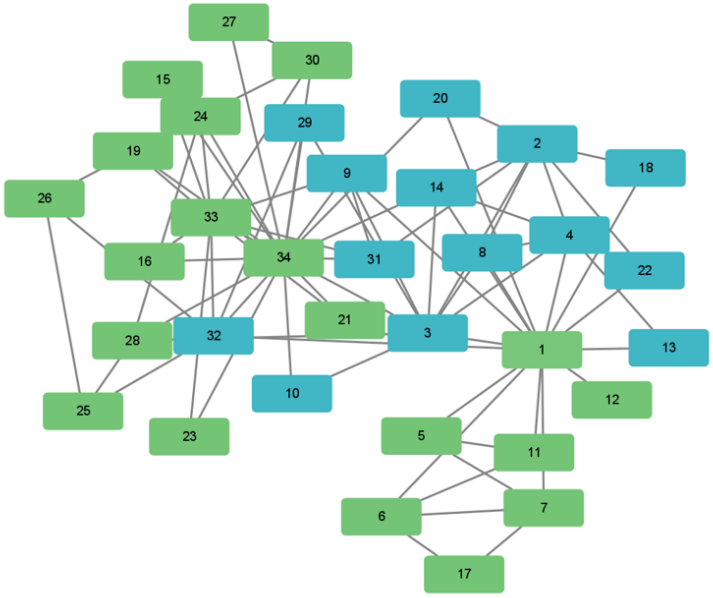
D = 3.8735



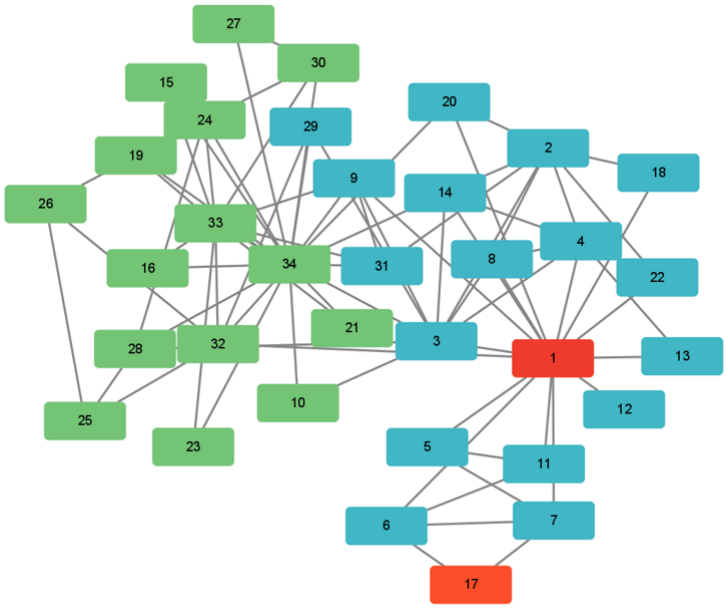
D = 6.1517

3.实现 ①，阈值=0.3时，通过多次运行程序，主要的存在三种社团模块密度值较好的聚类，包括以上两种。

另一种为为[29,32,10,8,4,14,13,2,18,22,20,3,9,31],可见仍包含[8,4,14,13,2,18,22,20,3,9,31]。



4.实现 ②，阈值=0.25时，通过多次运行程序，每次运行主要存在如下聚类结果，运行结果详见'实验结果图'文件夹。



总结：完成设计方案内容，通过调试，整体完成度较高，针对了两种情形实现。

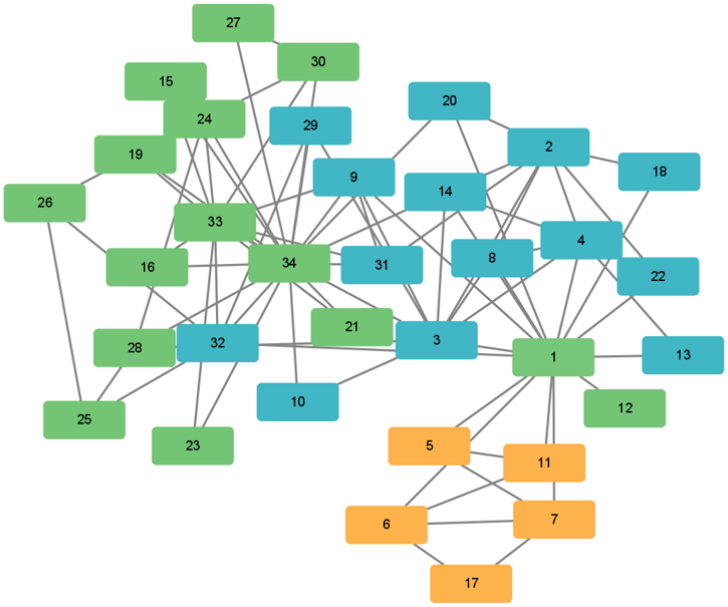
5. 心得体会

通过本次实验的研究学习，我对复杂社团检测算法有了进一步的了解，用程序实现聚类算法让我对课本上的知识也有了亲身实践，理解更为深入。

本程序实现的优点在于构造了针对真实情况（两个社团）和未知（假设多个社团）的方法来实现贪婪算法，有助于验证聚类结果的正确性。

本程序也存在一些不足之处，首先第一种方法是对于随机点的要求较严格，如果不是某些特定的随机点将无法得出较好的聚类结果。

此外，我对实验数据集也有一些疑问，从针对真实情况的聚类结果来看，该真实数据集划分为3个社团应该更合理，但这不符合实际情况，而针对未知情况的聚类结果则更类似于2个社团。这也是我的疑惑所在。



6. 源程序与文件附录

1. **task3_1.m**（针对真实情况，即两个社团进行聚类）
2. **task3_2.m**（针对未知情况，即多个社团进行聚类）
3. **gmlread.m**（用于读取网络数据，.gml文件）
4. **and_number.m**（用于计算相似度S，计算两个节点邻居节点的交集）
5. **or_number.m**（用于计算相似度S，计算两个节点邻居节点的并集）
6. **density_average.m**（用于计算社团密度指标，平均距离）
7. **dens1_number.m**（用于计算模块密度函数D_社团内部）
8. **dens2_number.m**（用于计算模块密度函数D_社团之间）
9. **draw_data.m**（导出画图数据,包括社团点集和边）
10. **displayRes.m**（用于画出社团聚类结果图，但视觉效果较差）
11. **draw_data_edge.xls**（Cytoscape画图数据）
12. **draw_data_points.xls**（Cytoscape画图数据）
13. **test_data.mat**（导入该数据可以获得实验结果图效果）
14. 剩余实验结果截图和cys文件均在'实验结果图'文件夹中