实验三 聚类技术---复杂网络社团检测



班级：1403012

学号：14030120075

姓名：张国庆

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。

数据来源：

本实验利用两类数据：模拟数据与真实数据。模拟数据有著名复杂网络学者Mark Newmann所提出，该网络包括128个节点，每个节点的度为16，网络包含4个社团结构，每个社团包含32个节点，每个节点与社团内部节点有k1个节点相互链接，与社团外部有k2个节点相互链接 （k1+k2=16）。通过调节参数k2 (k2=1,2,3,4,5,6,7,8)增加社团构建检测难度。http://www-personal.umich.edu/~mejn/

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。

1. **实验分析及设计：**

本次实验用老师所给的实验要求中的步骤完成。

**步骤一、导入网络数据；**

利用邻接矩阵A来存储网络，其中A\_{ij}表示第i个节点与第j个节点的是否有边相互链接，1表示有，0表示没有

**步骤二、采用贪婪算法提取模块**

随机选择一个未聚类的节点作为当前社团C，提取出社团C所有未聚类的邻居节点。选择使得社团密度降低最小的那个节点v添加到社团C，更新当前社团为。持续该过程知道当前社团的密度小于某个阈值。



**步骤三、采用Cytoscape工具，可视化聚类结果**。

使用cytoscape软件来可视化网络聚类结果。

1. **详细实现：**

上次实验一中采用了c++语言，花费了很长时间编写实验的代码。而这次我选择更加高级的python语言，好简化代码的编写过程，将更多精力放到实验本身。上述所有步骤均用python实现过。下面是详细实现过程。

* 1. **导入网络数据；**
     1. karate数据的导入

karate数据保存在gml文件中，用方括号来标志数据的开始和结尾，内有嵌套。整个数据被组织成类似一个树形结构。所以我用栈来恢复这棵树并建立图。注意：在文件中节点id是顺序的，边和节点也是分开的。按照顺序读取就可以完成图的建立，不过我写的读取方式适用于更复杂的形式（比如乱序，节点和边错杂）的图的组织。

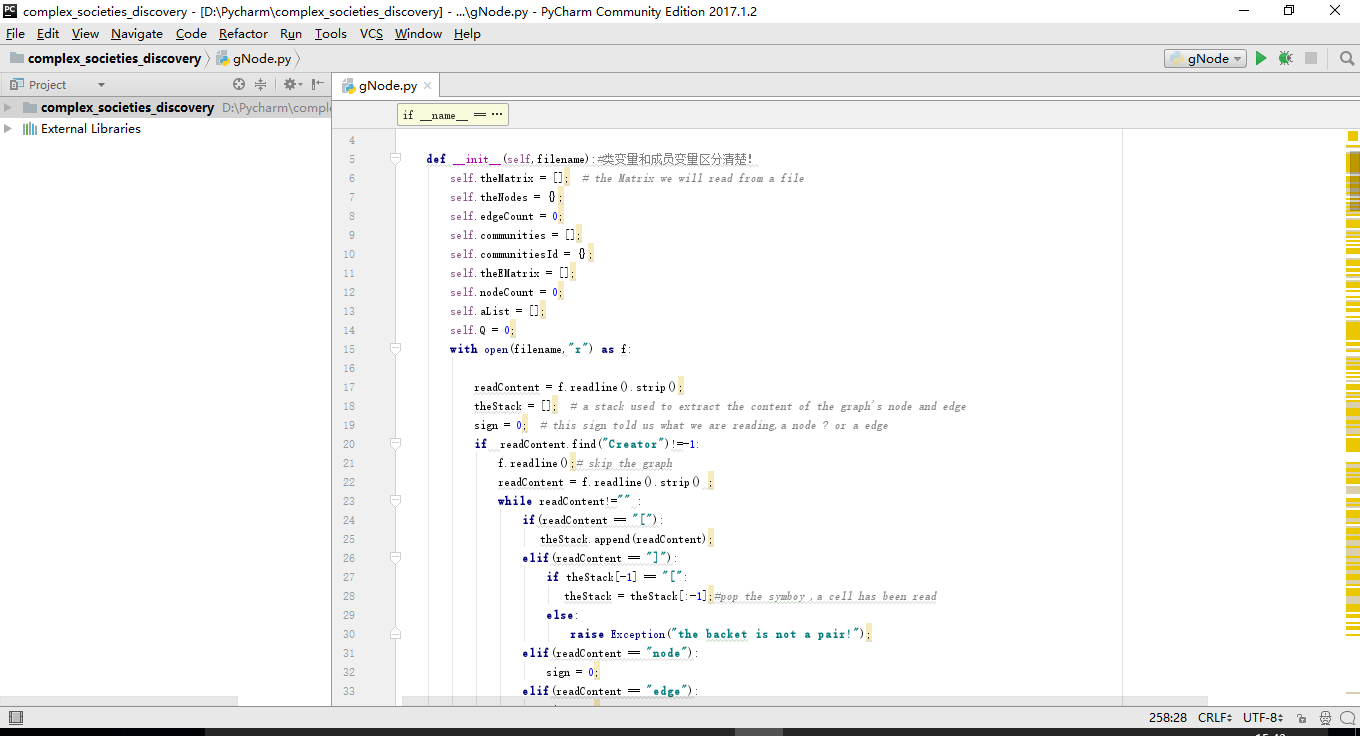
* + 1. Newman数据的导入

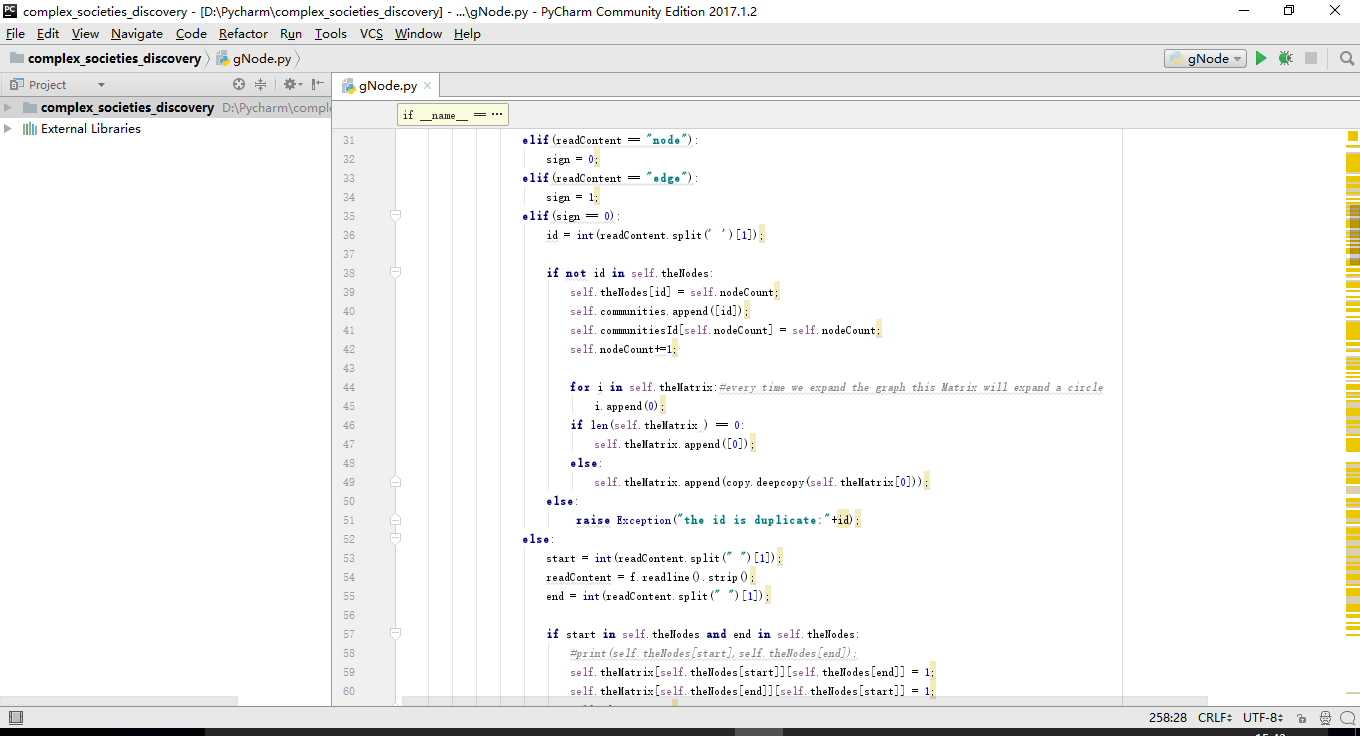
Newman数据保存在txt中，只保存了边。这个数据解读相比上述办法更加暴力一点。只要读取边，并添加两段节点即可。

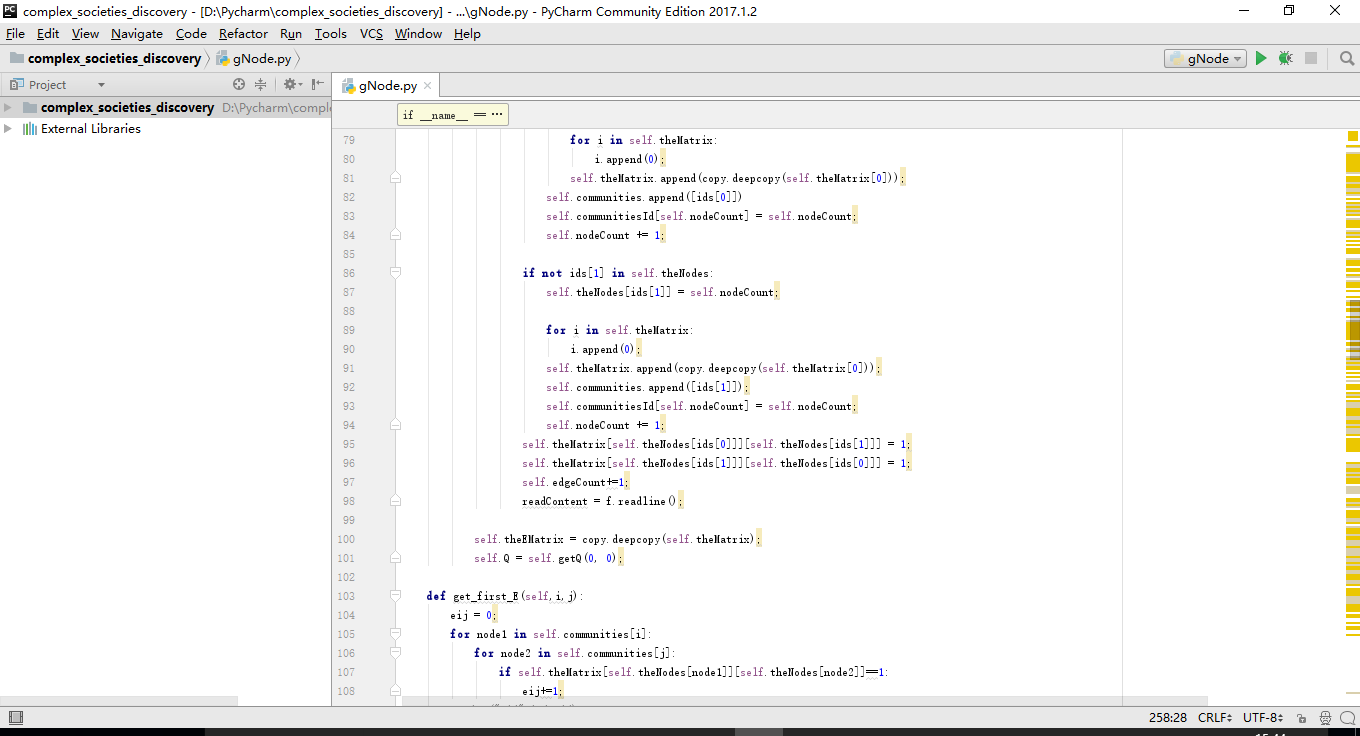
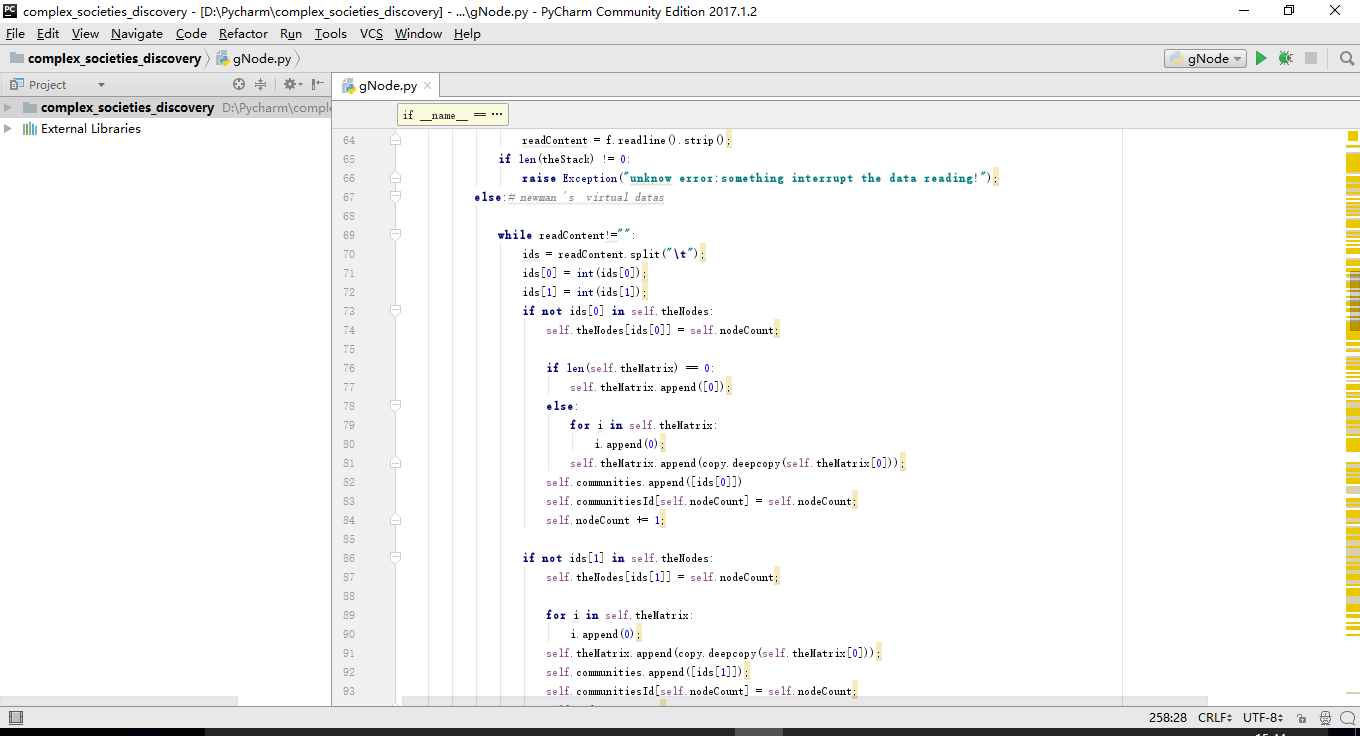
* + 1. 图的组织

采用图的邻接矩阵来表示，方便进行后续的运算。而且这两个图的节点分别是34和128，用邻接矩阵表示，空间也绰绰有余。具体实现是用python中的list，额外需要一个dict来表明节点id和位置的对应。

本次实验的函数都实现在gNode这个类中。而这一步也就是这个类的构造函数\_\_init\_\_函数。具体代码如下：







初始函数较为复杂，因为包含了两种读取方式，并且做了后续计算的相关处理：图的组织，初始化社团等。

* 1. **采用贪婪算法提取模块**

社团发现算法我用的Fast-Newman算法。这个算法借助了贪婪算法的思想，每次合并两个社团，向Q值方向增大最多或者减少最少的方向移动。在实际的网络中,Q的值通常在0.3-0.7之间,Q的值 越大,网络分裂的结果越好。

* + 1. Q值的定义

为了衡量社团聚类结果好坏，Newman提出了Q值（模块度）。下面是q值的定义。newman提出了两个q值的定义。第一次Q值定义如下：

**定义一：**   
　　最早由Newman在文章(1)中给出的模块度的概念：   
　　 Consider a particular division of a network into k communities. Let us define a k ×k symmetric matrix e whose element eij is the fraction of all edges in the network that link vertices in community i to vertices in community j.   
　　 假设网络被划分为 *k* 个社区，那么定义一个 *k*×*k* 的对称矩阵 *e*，它的元素 *eij* 表示社区 *i* 和社区 *j* 之间的边的数量。   
　　   
　　The trace of this matrix *Tre*=∑*ieii* gives the fraction of edges in the network that connect vertices in the same community, and clearly a good division into communities should have a high value of this trace.   
　　矩阵的迹*Tre*=∑*ieii*，也就表示了在相同社区内节点之间的边集合。显然，社区划分的好，也就是社区内部节点之间联系密集，那么该值就越高，这与我们通常对社区的认识是一致的。   
　　作者也指出，仅通过矩阵的迹这个值不能完全反应社区结构，因为如果把网络仅划分成1个社区的话，那么会导致*Tre*=1。

　　So we further define the row (or column) sums *ai*=∑*jeij* , which represent the fraction of edges that connect to vertices in community *i*。   
　　因此又定义了一个值 *ai*=∑*jeij*，表示所有连接到社区*i*的边数量。   
　　于是，作者定义了第一版模块度的概念：

*Q*=∑*i*(*eii*−*a*2*i*)=*Tre*−||*e*2||

　　||*x*|| 表示矩阵 *x* 的所有元素的和。   
　　简单展开一下上面的公式会更清晰：

*Q*=∑*i*(*eii*−*a*2*i*)=∑*ieii*−∑*ia*2*i*=*Tre*−||*e*2||

**定义二：**   
　　Newman在论文(2)中更新了模块度Q的定义，增加了矩阵的表达，感觉更不直观了。   
　　下面是一些定义及推导：   
　　Modularity is the fraction of the edges that fall within the given groups minus the expected such fraction if edges were distributed at random.   
　　模块度 = （落在同一组内的边的比例） 减 （对这些边进行随机分配所得到的概率期望）   
　　为了说明上述公式，对定义如下内容：   
　　假设网络有*n*个节点，有*m*条边，节点*v*的度表示为*kv*。   
　　将网络的邻接矩阵表示为*A*，*Avw*=0 表示节点*v*和*w*之间没有边，*Avw*=1 表示有边。   
　　定义变量*s*，*svw*=1表示*v*和*w*属于同一社区，*svw*=−1表示不在同一社区，那么可以用*δvw*=12(*svw*+1)量化表示*v*和*w*是否在同一社区，如果是则等于1，不是则等于0.   
　　那么上述模块度定义就可以表示为：

*Q*=12∑*vwAvwmδvw*−（对这些边进行随机分配所得到的概率期望）

　　∑*vwAvwδvw*/*m* 表示在同一社区内的边的数量占所有边数量的比例， 乘以 1/2，是因为对每条边计算过两次。   
　　   
　　以上这部分计算很清楚，下面主要看那个期望值如何计算。   
　　这里面用到了Configuration Models，大体意思就是，对网络的边进行随机分配，需要将每条边切断一分为二，切断的点我们称作末梢点(stub)，这样*m*条边就会产生*ln*=2*m*=∑*nv*=1*kv*个末梢点，随机的将这*ln*个末梢点进行连接，包括同一节点拥有的末梢点的自连接。这样可以保持每个节点原有的度不变的条件下，可以得到一个完全随机网络。   
　　在该随机网络下，任意两点*v*、*w*连接边数的期望值是：*expvw*=*kvkwln*=*kvkw*2*m*   
　　因此，节点*v*和*w*的实际边数与随机网络下边数期望之差为：*Avw*−*kvkw*2*m*   
　　模块度最终可表示为：

*Q*=1/2*m*∑*vw*(*Avw*−*kvkw/*2*m*)*δvw*(2)

　　定义 *eij*表示连接社区 *i* 和 *j* 的边的总数，*eij*=∑*vwAvw/*2*mδviδwj*，*ai*表示连接 社区*i*的边的总数，*ai*=∑*jeij*=*ki*2*m*，那么模块度可以表示为：

*Q*=1/2*m*∑*vwAvwδvw*−1/2*m*∑*vwkvkw/*2*mδvw*=∑*i*=1*c*(*eii*−*a*2*i*)

可以看到虽然两版定义不同，最后得到的结果是相同的，也非常简单明了。

模块度的取值范围为：[−1/2,1)，有可能得到负值。

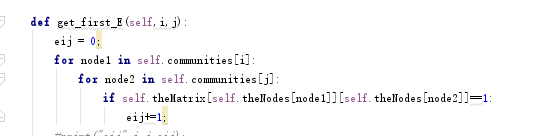
* + 1. Q值的计算

Q值在上述计算中需要累加，似乎有点复杂。虽然图大小不大，但是每次都按照上述步骤计算，时间复杂度很大。计算e需要（n^2），而a值计算需要n次e，再加上Q值计算的累加，即使图不大，也需要较长的时间（Q（n^4））。不过幸运的是我们不必每次都计算这个值。通过计算每次Q的增加值即可。而△Q的计算非常简单。经过分析我们得到下面公式：△Q=eij+eji-2ai\*aj。同时将e值以矩阵形式保存，合并时候的复杂度降为O（m）（m为社团的个数），再将a值保存一个数组，合并只需要一次计算。这样后续计算Q值只需要O（1），每次合并也就需要

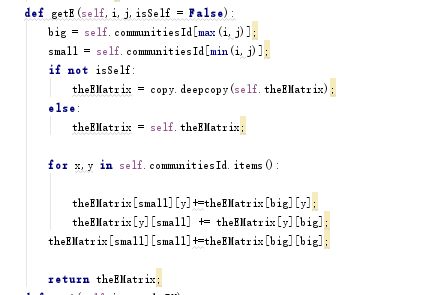
O（m），计算速度大大增加。这就是fast-newman算法的精髓。

Q值计算首先需要第一次对E矩阵的计算，用get\_first\_E函数实现，后续分别需要计算e，用getE实现，返回一个E矩阵，计算a，用getA实现，以及最后getQ。各个函数如下：

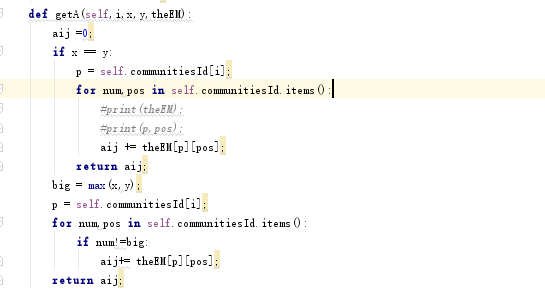
get\_first\_e:



getE:



getA:



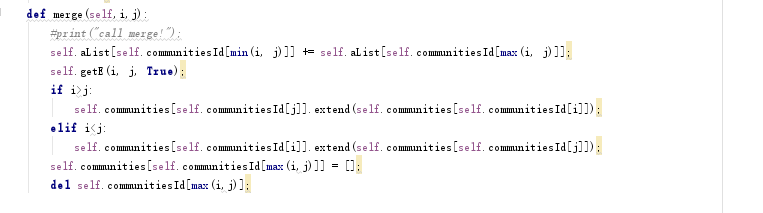
getQ:

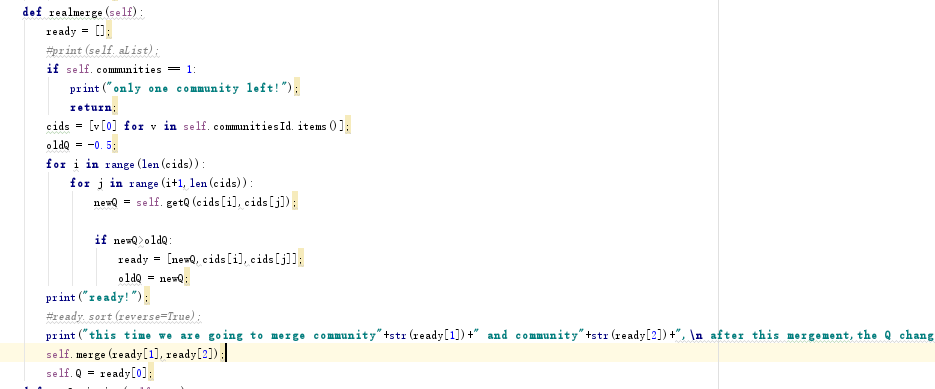


* + 1. 具体算法实现：

从所有现存的社团中选择两个合并，选择Q朝增加最多或者减少最小的方向合并。合并内部用merge函数实现，整个函数被封装为一个getRealMerge函数，代码如下：

merge：





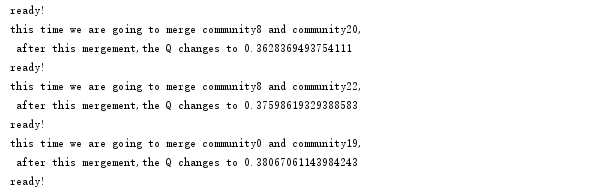
上述代码合并，并且打印相关信息。而对于整体合并，选择一个阈值，比如karate中合并至两个社团停止，Newman合并至4个社团停止，并打印Q值。将结果输出，好配合下一步操作。

* 1. **采用cytoscape工具可视化聚类结果**

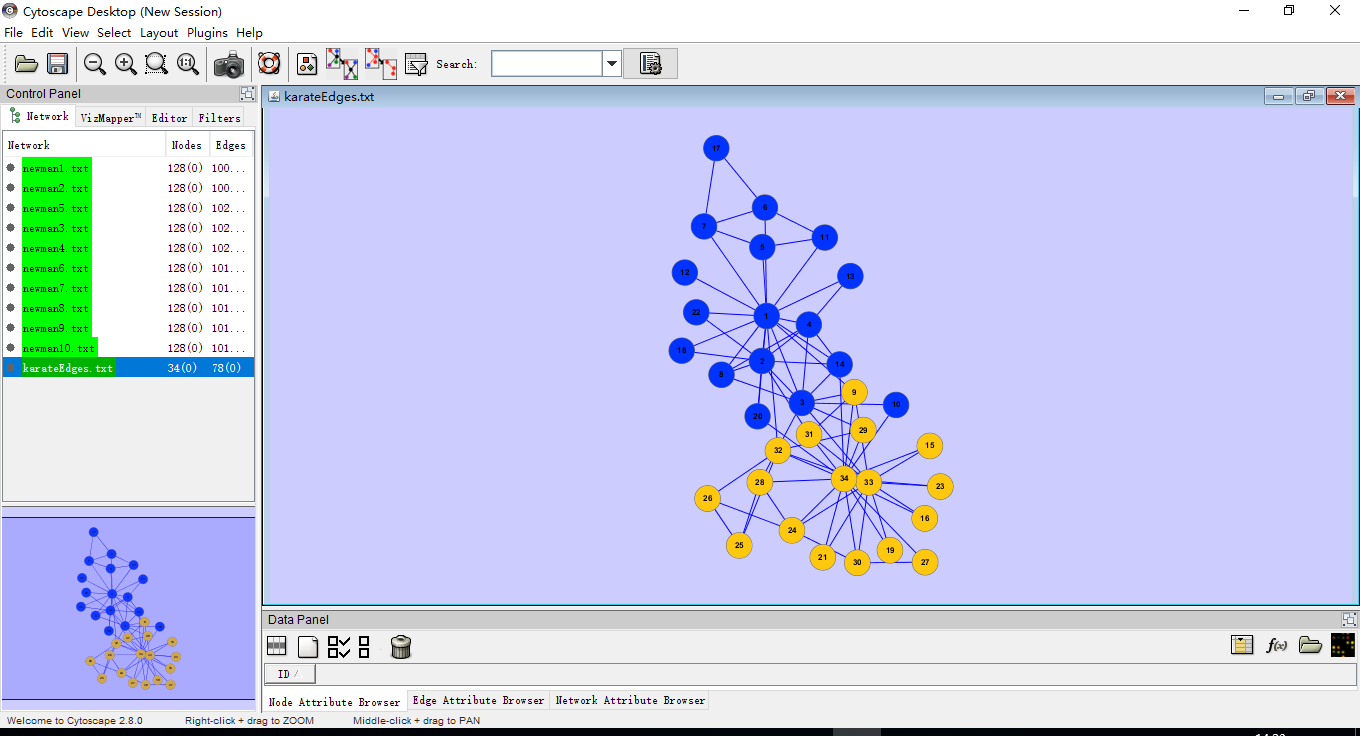
使用cytoscape软件可以清晰画出聚类结果。我们需要一个是图的文件，另一个属性值，标志分类的所属社团。只需简单保存到.txt文档中用tab隔开即可。

1. **实验结果：**
   1. **真实网络的计算结果：**

美国空手道的网络被划分为两个社团。最后Q值打印出来如图所示：

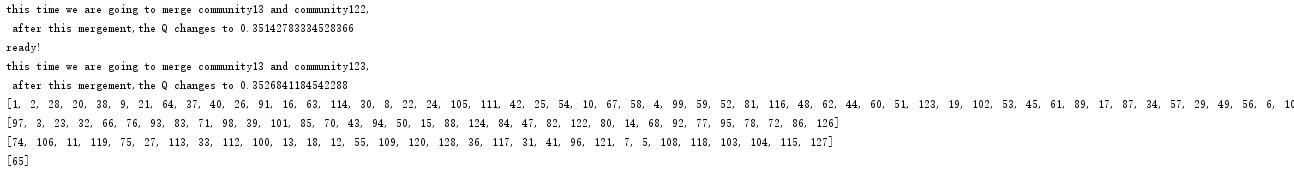


也就是Q值大小为：0.38067左右，符合真实网络的范围（0.3至0.7）。用cytoscape可视化聚类结果如图：

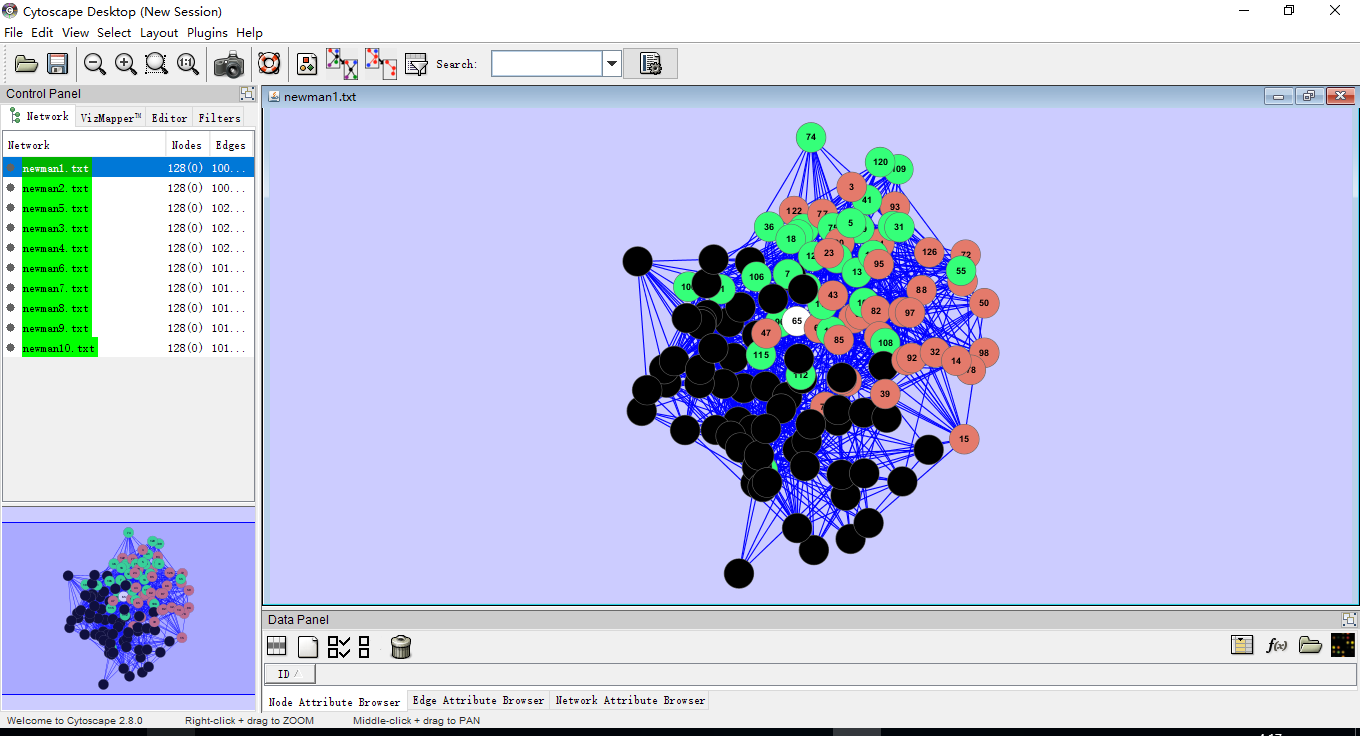


* 1. **虚拟网络的计算结果：**
     1. **newman1的计算结果**

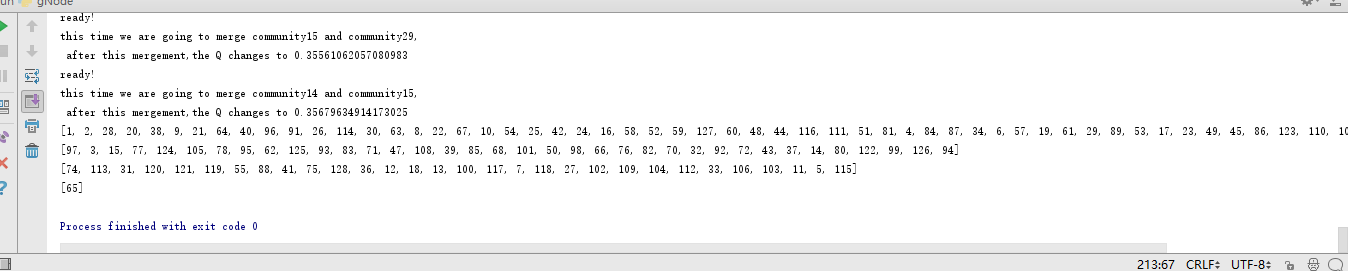
Q值大小：0.35268



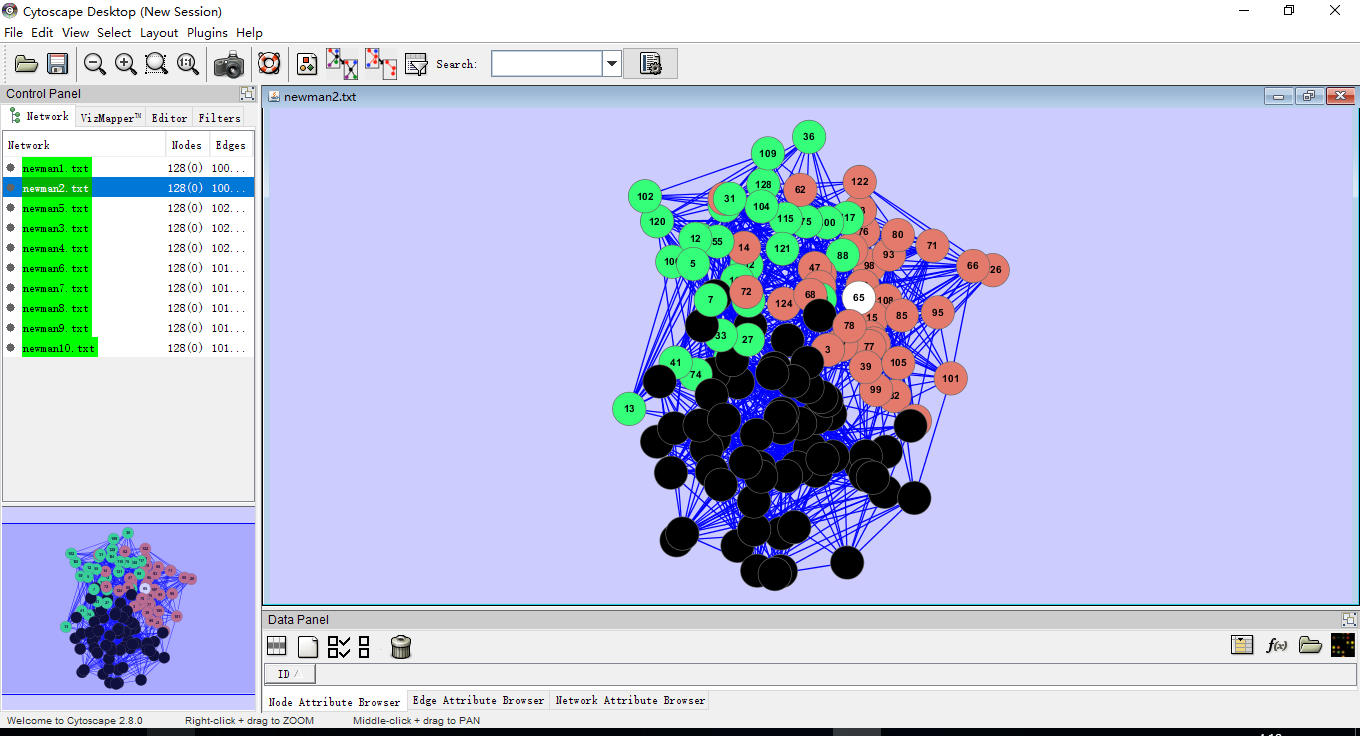
可视化聚类结果：



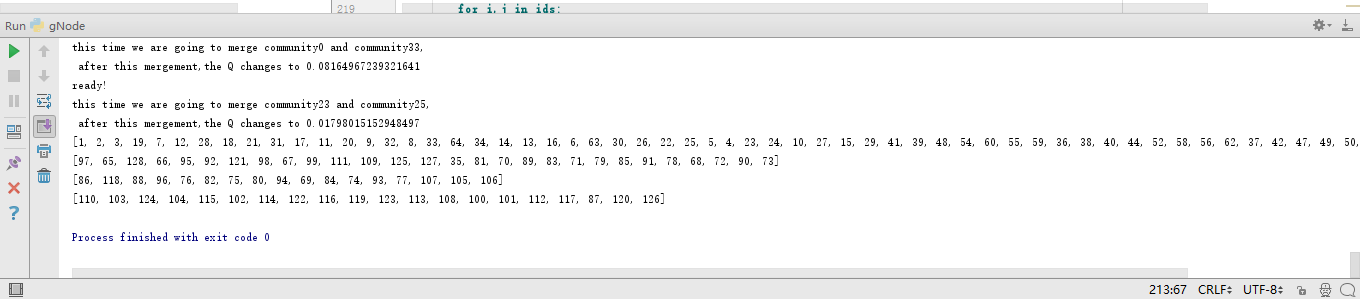
* + 1. **newman2的计算结果**

Q值大小：0.35679

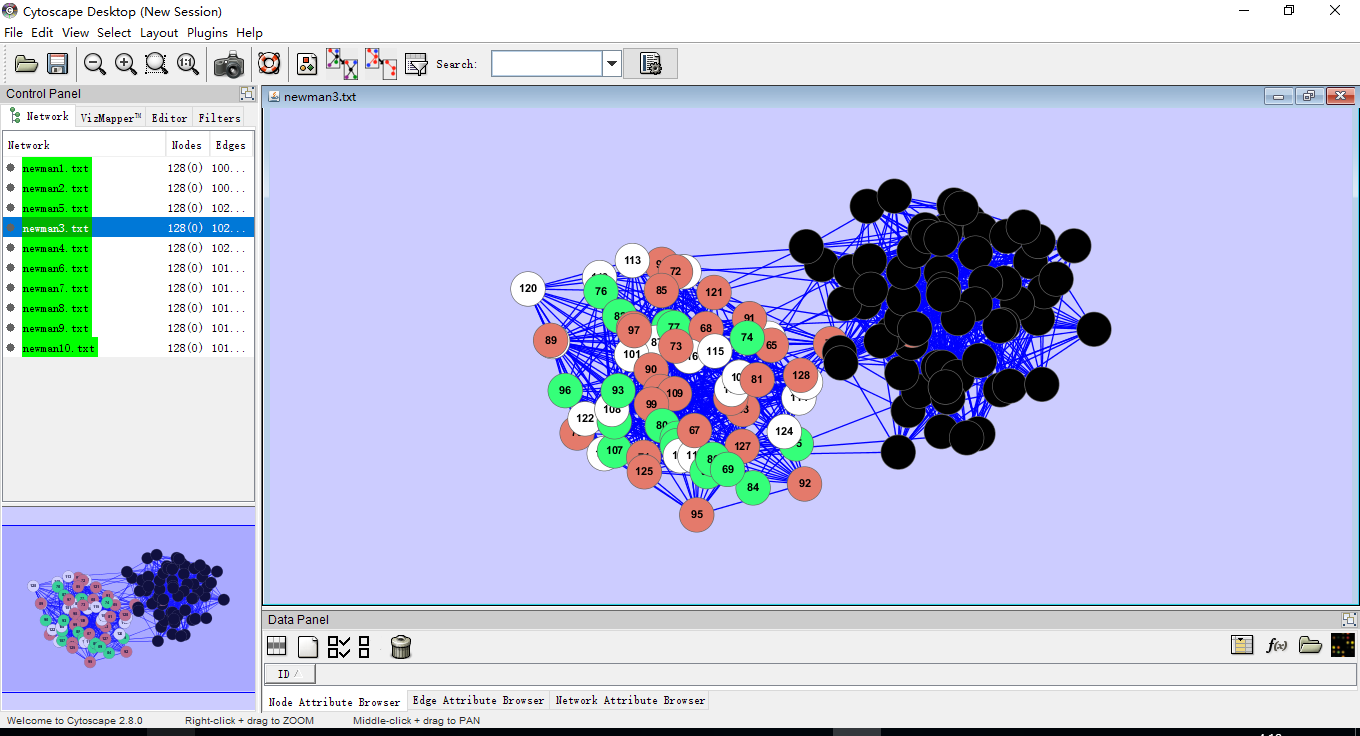
可视化聚类结果：



* + 1. **newman3的计算结果**

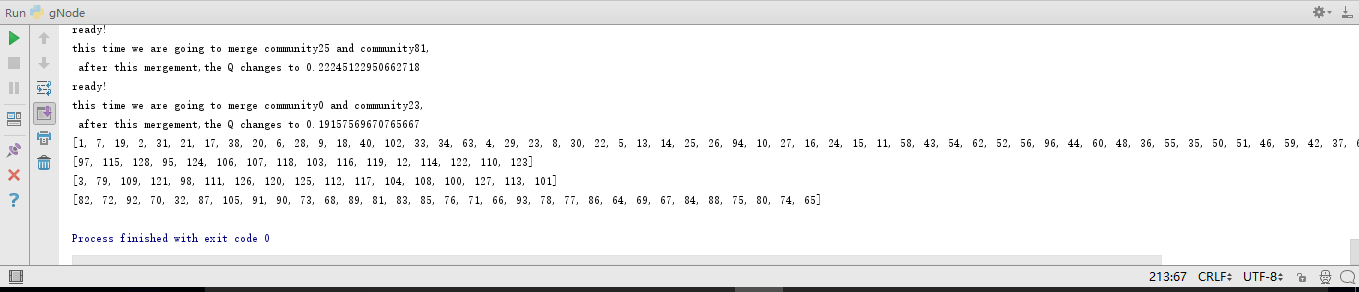
Q值大小：0.01798

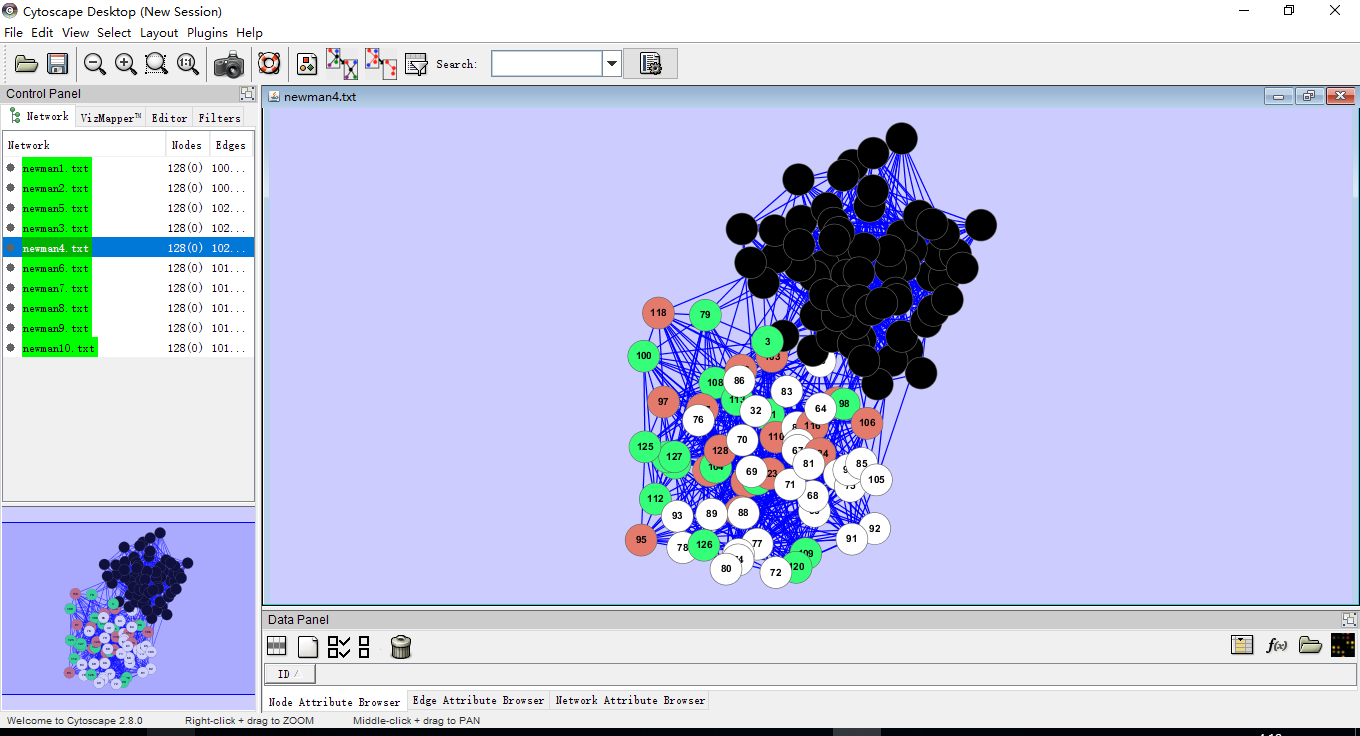
可视化聚类结果：



* + 1. **newman4的计算结果**

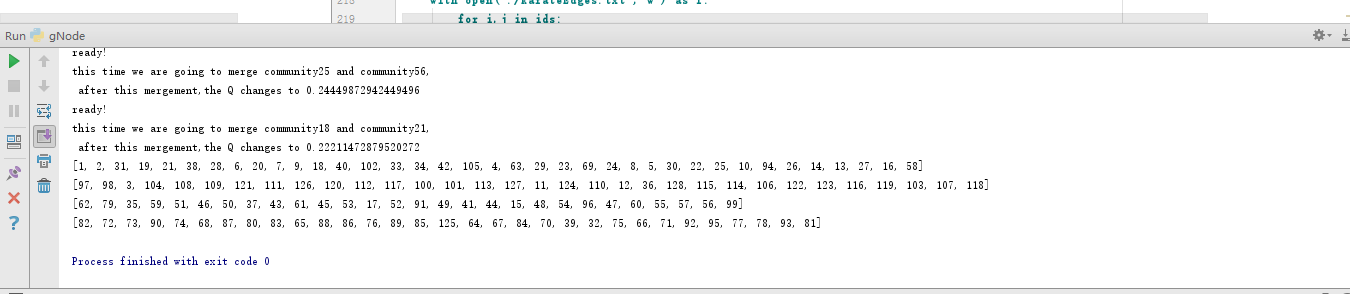
Q值：0.19157



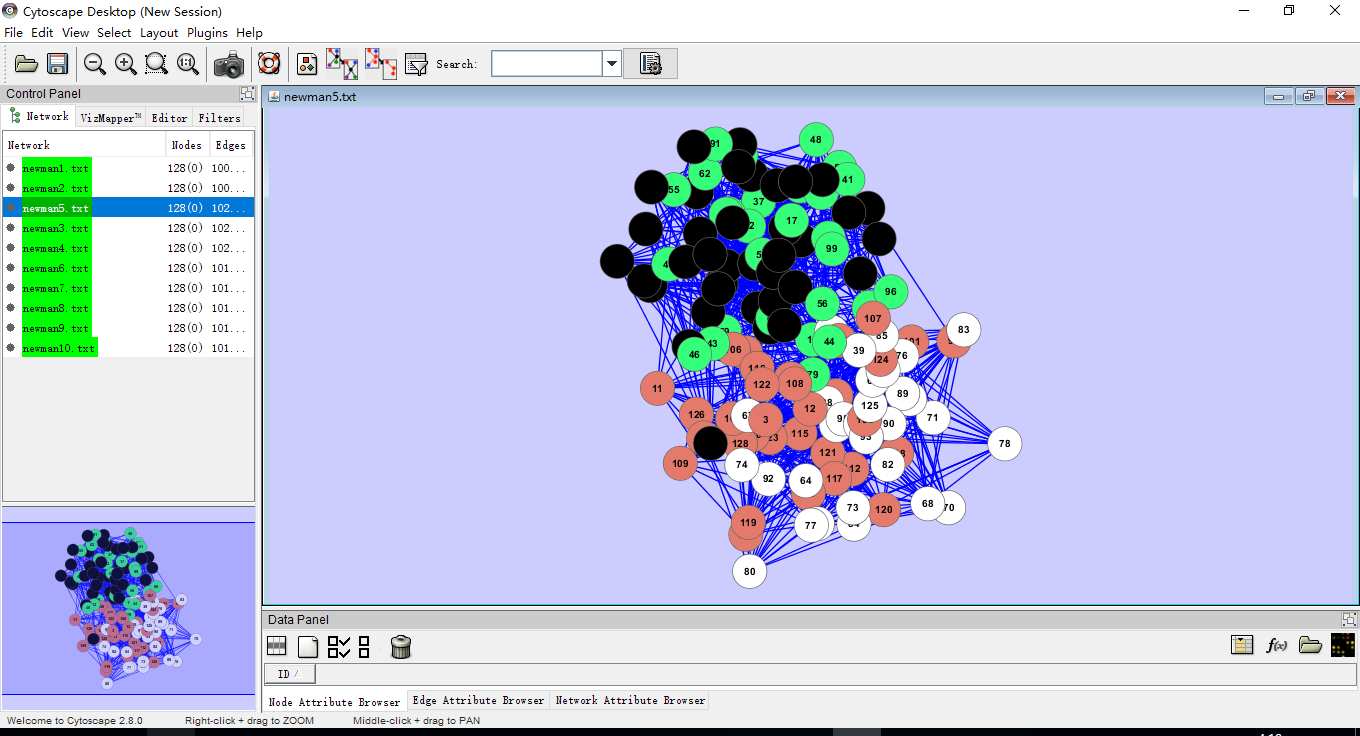
可视化聚类结果：

* + 1. **newman5的计算结果**

Q值大小：0.22211

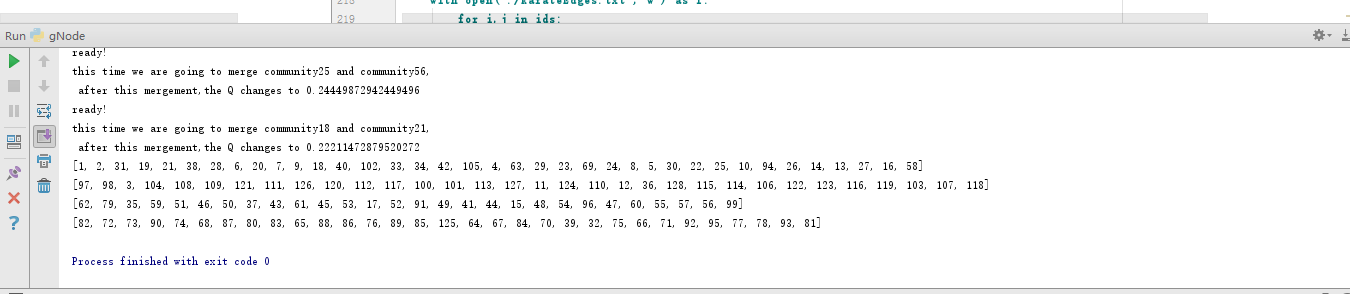


可视化聚类结果：

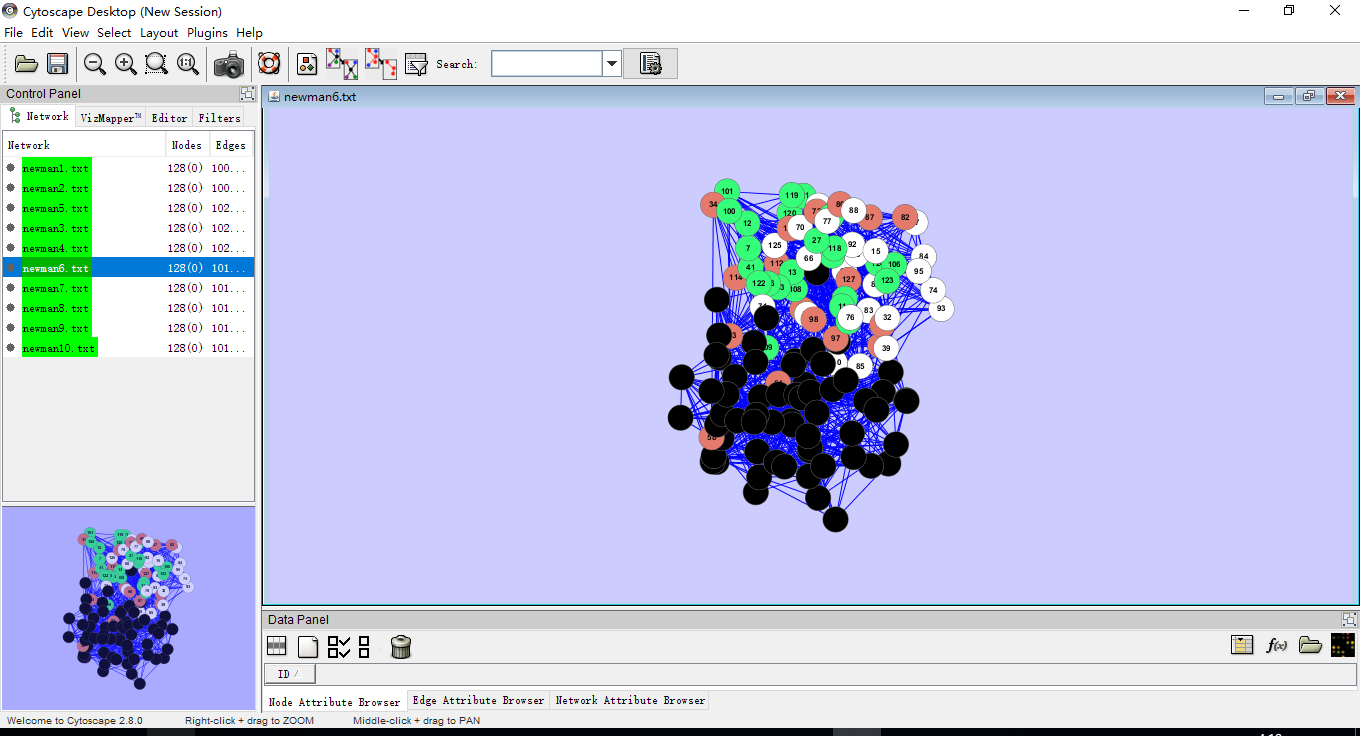


* + 1. **newman6的计算结果**

Q值大小：0.24126

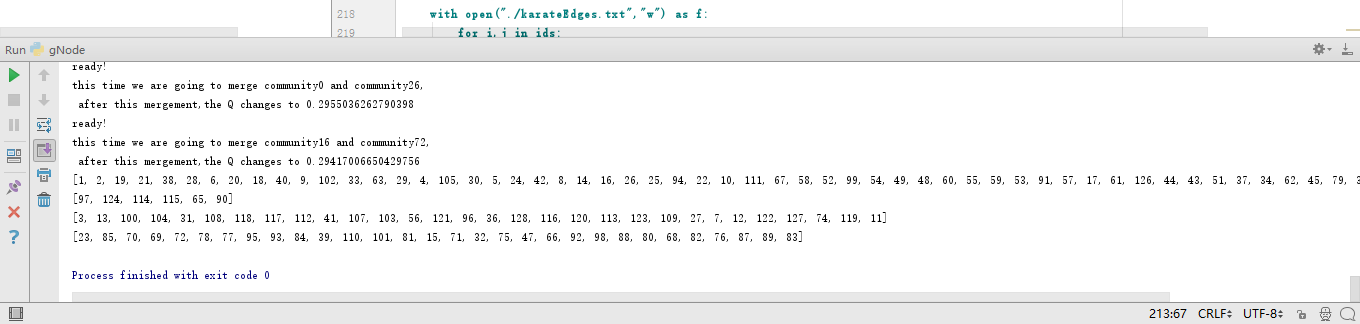


可视化聚类结果：

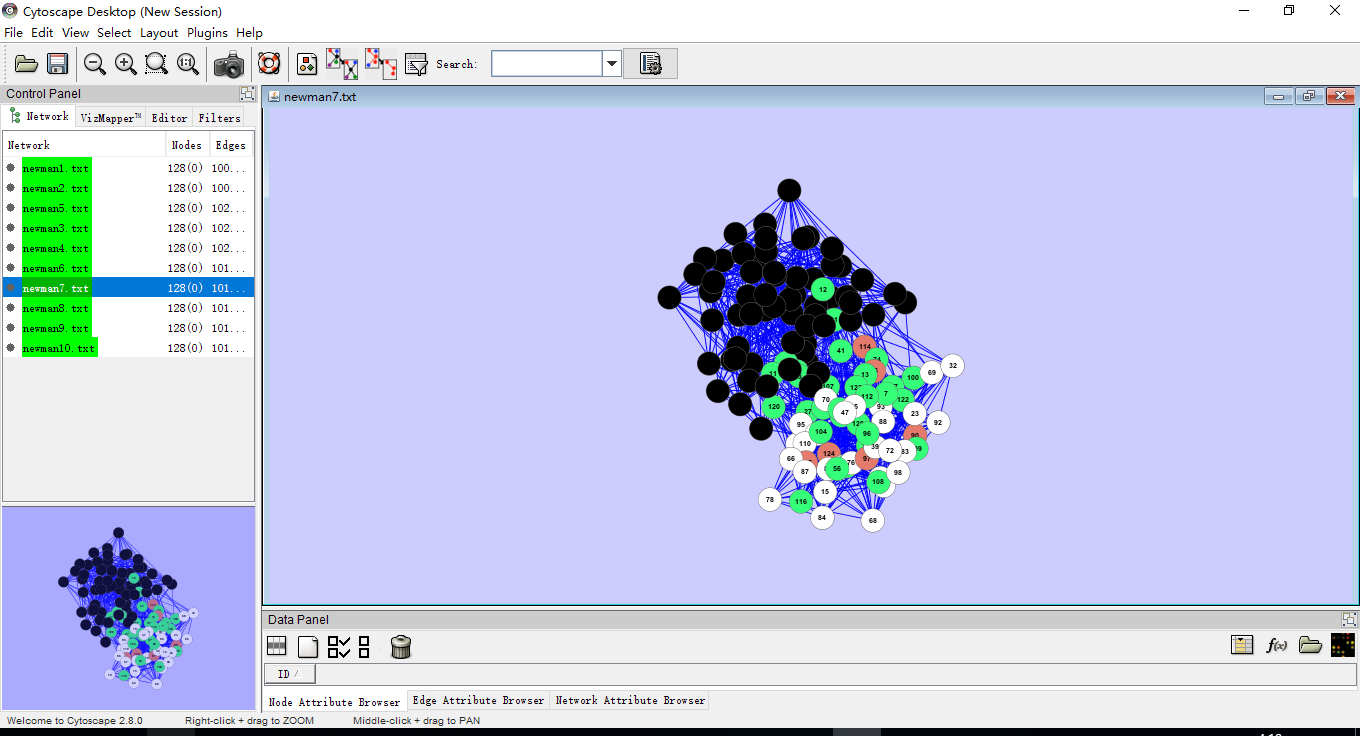


* + 1. **newman7的计算结果**

Q值大小：0.29417

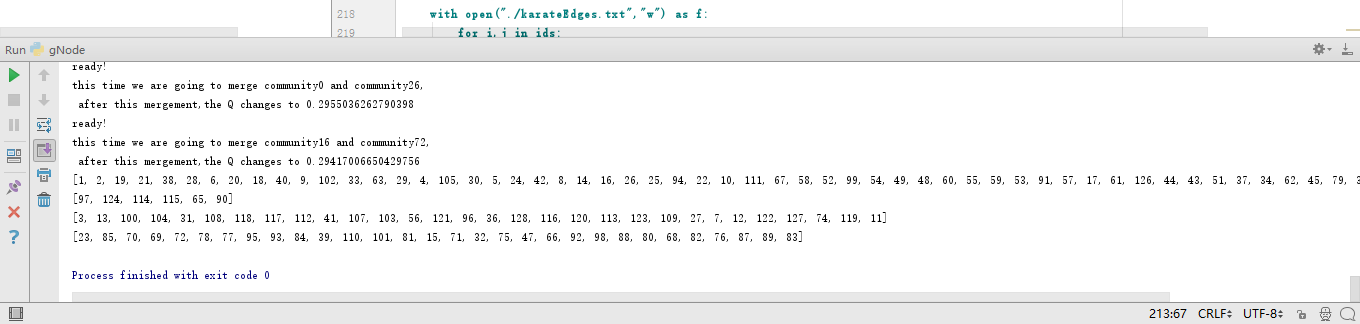


可视化聚类结果：



* + 1. **newman8的计算结果**

Q值大小：0.35065

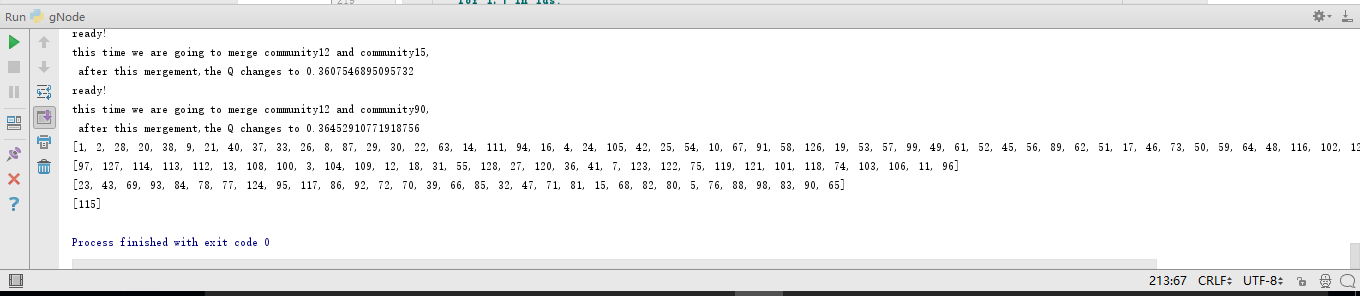


可视化聚类结果：

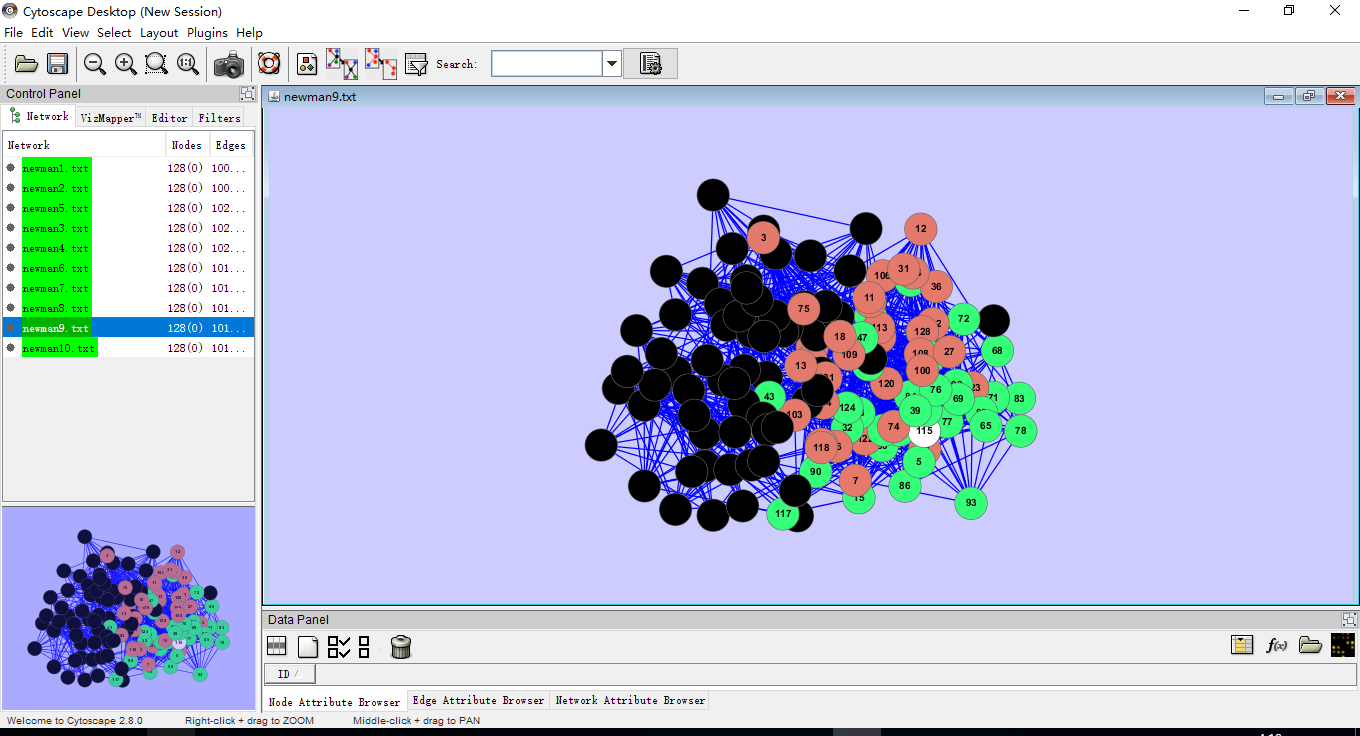
![C:\Users\无聊时的自娱自乐\AppData\Roaming\Tencent\Users\3076227974\TIM\WinTemp\RichOle\)5`QIH{~A[D)](B@81R{X]W.png](data:image/png;base64,)

* + 1. **newman9的计算结果**

Q值大小：0.36453

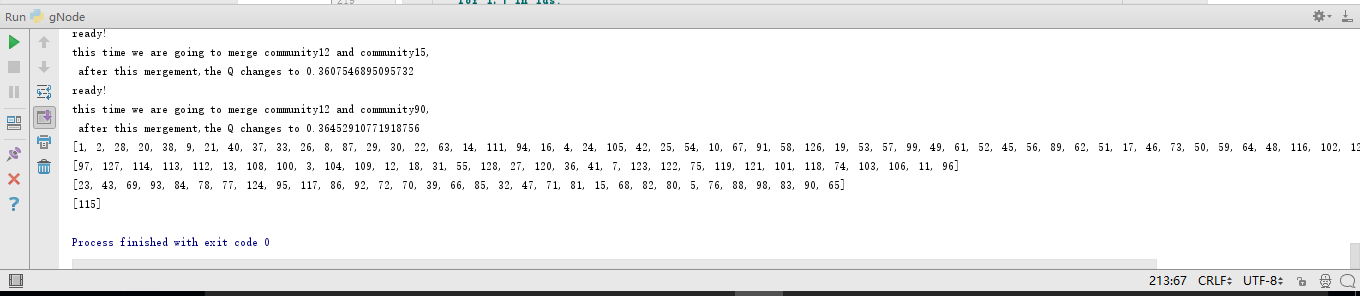


可视化聚类结果：

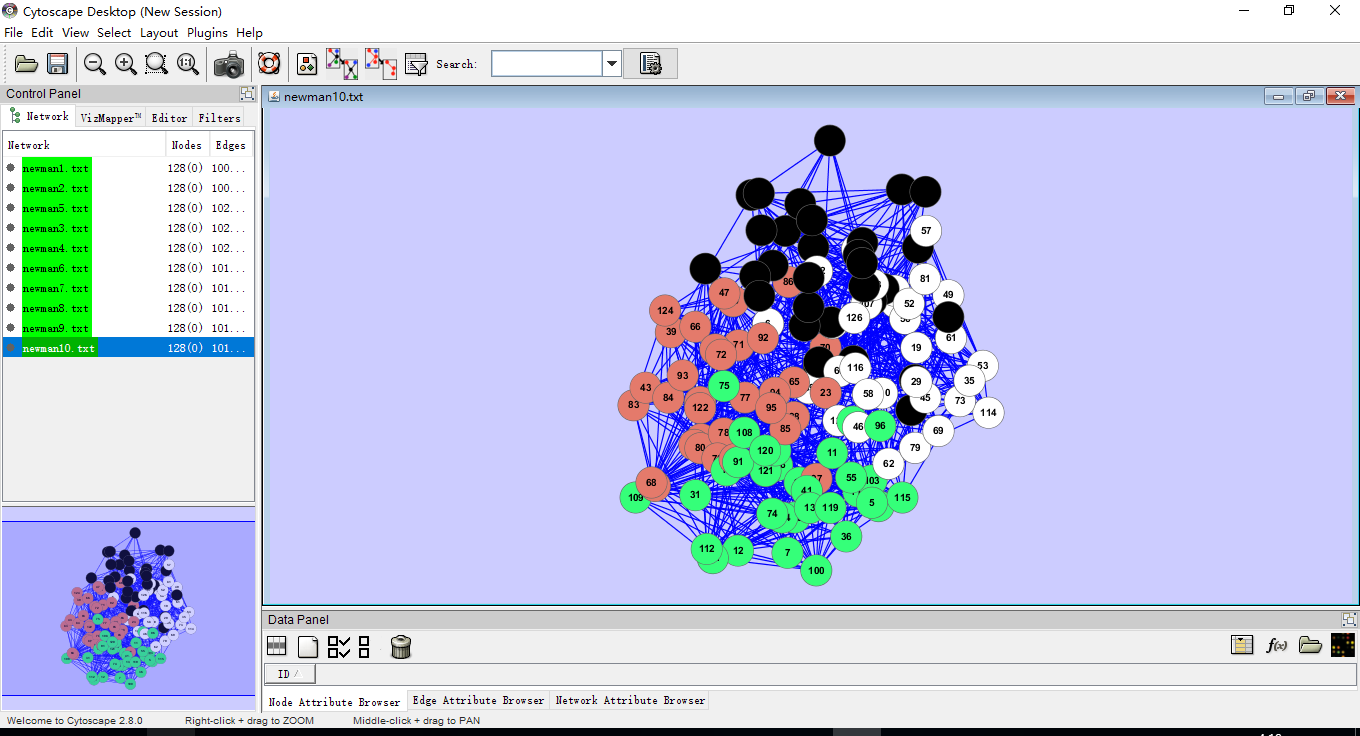


* + 1. **newman10的计算结果**

Q值大小：0.37354



可视化聚类结果：



实验结果打印的图的节点以及属性（也就是导入cytoscape的数据）打包到附件中发送。其中community1-10是虚拟网络的聚类结果，karate相关文件放在karate文件夹中。想要呈现上述图像，先导入Newman2008中的图数据，再导入community文件（用属性文件导入），再调节节点颜色即可。

1. **心得体会：**

对于社团发现算法的学习和了解，我更加深入地体会到了数据挖掘的魅力。如何从海量数据中找到各个网络节点之间的关系，对推荐，社交等等都非常有用。同时我也知道了自己知识水平的不足，也了解到了数据挖掘算法的魅力。今后我会更加努力学习数据挖掘以及机器学习相关内容，做更多有意义的事情。