**实验三 网络分析典型问题**

**一、 实验目的**

1、了解网络分析的一些典型问题。

2、创意设计典型网络问题的算法。

3、评估算法的效果。

**二、 实验条件**

硬件：计算机。

软件：计算机程序语言开发平台，如C、C++、Java、Matlab。

学生：至少掌握一门计算机程序设计语言，如C、C++、Java、Matlab。

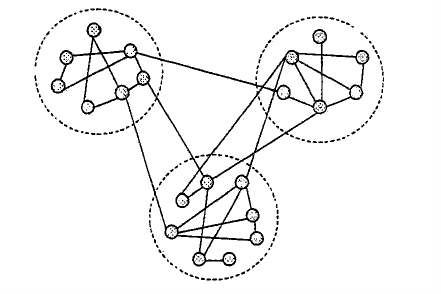
**三、 实验内容及要求**

（1）简介

**复杂网络**： 由数量巨大的节点和节点之间错综复杂的关系共同构成的网络结构。用数学的语言来说，就是一个有着足够复杂的拓扑结构特征的图。复杂网络具有简单网络，如规则网络、随机图等结构所不具备的特性，而这些特性往往出现在真实世界的网络结构中。人类社会中的社交网络、交通网络、电子邮件网络、科学家合作网、电子电路网络等，这些复杂系统都可以用复杂网络的相关特性进行描述和分析，网络中的结点表示系统个体，边表示个体之间的关系。如今，复杂网络的研究是现今科学研究中的一个热点，它已渗透到各个领域，并成为这些领域的重要研究方向。

**社区**：是图的一个子图，相比于图的其他部分，其中包含着密集的节点，或者等价地说，如果图一个子图内部的链接数量高于这些子图之间的链接数，我们就可以说这个图含有社区。

**社区发现**： 相同类型节点之间连接较多，构成一个一个的小社区，不同类型节点之间连接较少，但成为沟通不同社区的重要通道，这种连接的不均匀性表明，网络内部存在一定的自然分化。将一个网络划分成N个社群，使得这些社群构成网络的一个覆盖，并且满足社群的定义。在实践中，任务是找到最大解耦的社群集合。



（2）算法对比

**网络模型：** 网络数据往往是离散的，导致应用于网络的聚类算法倾向于直接使用网络图的属性，图中的节点代表实体，边代表实体之间的联系。一般的社区应该具有的特征为社区内部联系紧密，反映到图中是一个边稠密图，社区之间联系松散，如果把一个社区当做一个节点的话，重新形成的图是边稀疏图。

**凝聚算法：** 这类方法从一个个孤立的节点开始，计算每两个节点的相似度，相似度高则这两个节点在同一个社团。这种方法的问题在于：社区中的核心节点往往具有很高的相似度，但是边缘节点则容易被忽略。

**分裂算法**： 从感兴趣的网络开始，并尝试找到最不相似的连接顶点对，然后去除它们之间的边。通过反复这样做，我们将网络划分为越来越小的组件。我们可以在任何阶段停止该过程，并将该阶段的结果作为网络社区。整个过程可以表示为树形图，描绘了网络连续分裂成越来越小的组。

**对比**： 凝聚算法中，社区中的核心节点往往具有很高的相似度，但是边缘节点则容易被忽略。因此，分裂算法的准确度更高，对于边缘节点的处理更好，不会因为其他原因忽略边缘节点。

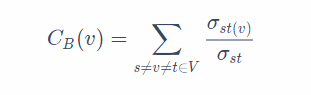
因此，本次采用Girvan-Newman分裂算法。

（3）算法设计

1. 基本概念

**介数和边介数：** 在图论中，介数中心性（betweenness centrality）是基于最短路径的图的中心的度量方法。对于连通图中的每对顶点，在顶点之间至少存在一条最短路径，而一个节点的介数一般是指通过该节点的最短路径数。

**介数中心性的定义**：

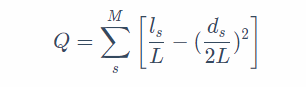


360截图18280504387848

直观上来说，介数中心性反映了节点v作为“桥梁”的重要程度。因此，我们可以定义边介数为经过该边的最短路径数，如图所示，位于社区之间的边的边介数将会很大。同时，可以进一步得到边介数中心性。

**模块度函数Q**： 模块度，是指复杂网络中连接社区内部节点的边所占的比例，与另一个随机网络中连接社区内部节点的边所占的比例的期望值的差值。其中，这个随机网络的构造方法为：保持每个节点的社区属性不变，根据节点度对节点间的边进行随机连接。如果社区结构比较明显，则社区内部连接的边的比例应改高于随机网络的期望值。

**Q 的函数表达式**：



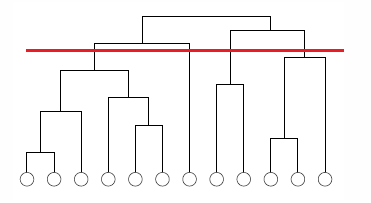
假设已经发现复杂网络的社区结构，M为已发现的社区个数，L 为网络中的边数，l 是社区 s 中节点相互连接的数目，d 是社区 s 中所有节点相互连接数目的和。

上式的物理意义是：网络中连接两个同种类型的节点的边（即社区内部边）比例，减去在同样的社区结构下任意连接这两个节点的边的比例的期望值。如果社区内部边的比例不大于任意连接时的期望值，则有Q=0，Q的上限为1，而Q越接近这个值，就说明社区结构越明显，在实际网络中，Q 值常位于0.3和0.7之间。

1. 基于边介数的社区发现算法

**算法简介**： 计算网络中所有边的边介数，然后寻找边介数值最大的边并移除。在移除时，会改变一些边的边介数。以前经过被移除边的最短路径，现在改为经由其他边，因此必须重新计算边介数，然后再次搜索值最大的边并移除，依此类推。当一条接一条边被移除时，最初连通的网络最终会被分成两部分、三部分等等，直至网络的每一个成为一个社区，是一种分裂算法。

**算法流程**： 算法从顶部开始生成树状图，首先是单个连通网络，然后将其不断划分，直至每个分支只有单个结点为止。在算法运行期间，每个独立的网络结构状态对应于树状图的水平切分。



**算法优势**： 这种算法并不是给出将网络划分成社区的唯一划分，而是给出多种不同程度的划分。范围比较粗的划分，将网络划分为几个大的社区，范围比较细的划分，将网络划分为许多比较小的社区。这种不同程度的划分，可以揭示关于层次网络结构的信息。同时，用户可根据目标社区数来决定进行哪些划分，或者利用模块度函数 Q 给出该算法的最优划分。

1. Girvan-Newman算法

**算法简介**： 如前所述，直观来看，在社区内部节点之间相互连接的边密度较大，因此，通过边介数（连接社区的边的边介数大）来识别社区是一种较为直观的社区发现算法。Girvan-Newman算法即是一种基于介数的社区发现算法。

**算法流程**： 根据边介数中心性从大到小的顺序不断的将边从网络中移除直到整个网络分解为各个社区。

**详细流程**：

（1）计算网络中所有边的边介数；

（2）找到边介数最高的边并将它从网络中移除；

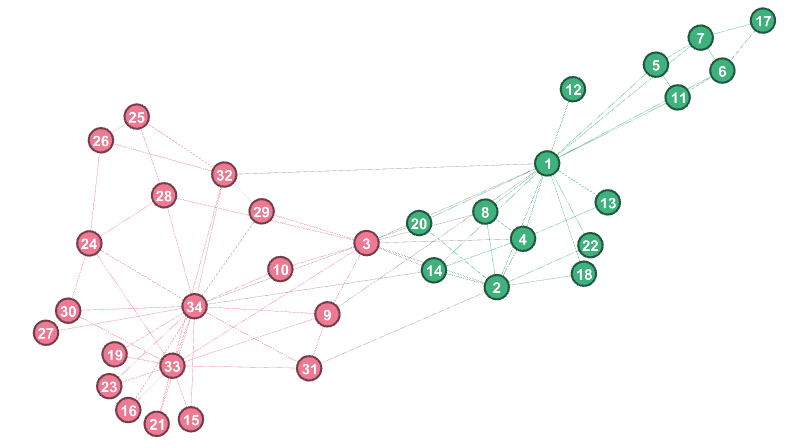
（3）重复步骤2，直到每个节点成为一个独立的社区为止，即网络中没有边存在。

**算法缺点**： Girvan-Newman算法使用模块度函数 Q 寻找合适的社区数，但Girvan-Newman算法中每去除一次边，则计算一下所得社区结构的 Q 值，寻找到 Q 值最大时的社区数量。当网络节点数量太大时，对于 Q 值的计算过于频繁进而导致算法整体效率下降。

**算法改进**： 因为 Girvan-Newman算法的最大缺点为每次都要计算 Q 值，而 Q 值总是要把计算的社区个数相加，因此可以采用贪婪算法来替代每次计算 Q 值的复杂操作，当再次取到的 Q 值变小时，则直接取该当前最大值。因此采用了一种模糊的取值方式来取代之前的精确值，但该方式也相应的不能达到最好效果，但是大体上来看，对于大型网络，该方式的效果比之前的精确的方式更好。

1. 采用数据集

**Zachary karate club网络**： Zachary网络是一个检验不同社区发现算法的经典真实的无权网络，一共有34个节点，78条边。该网络是Zachary在上世纪70年代，用了两年时间观察研究空手道俱乐部34个成员之间的个关系而得到的，如下图所示。在调查过程中，俱乐部的主管和校长产生争执，分裂成了两个团体。



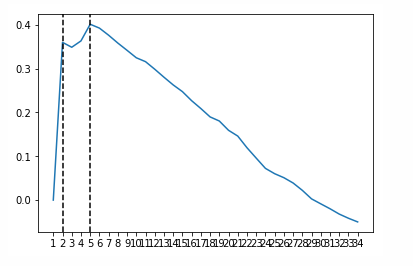
1. 实现

**实现方式**： 这里使用 Python 的 NetworkX 包以及 Matplotlib 包来完成对于 Girvan-Newman 算法的演示，以及Gephi。

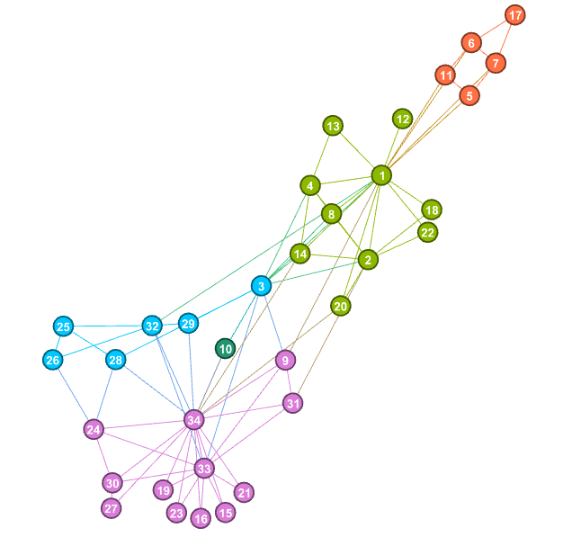
NetworkX是一个用于创建，操作和研究复杂网络的结构、动态和功能的Python包。Matplotlib是一个Python 2D绘图库，利用它完成模块化函数Q与划分社区数目的关系曲线，之后使用Gephi作为网络的可视化工具，将GML文件中的图中划分的社区表示出来。

**环境需求**： Python-3.6.3、NetworkX-2.0、Matplotlib-2.1.0

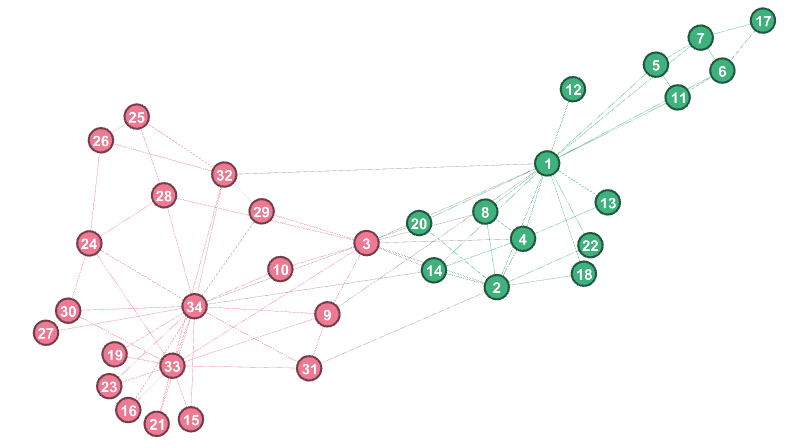
**运行效果**： 可直接运行Girvan-Newman.py，输出格式为GML的文件，以及其他信息，Zachary’s karate club网络Girvan-Newman算法划分社区数与Q值的关系：



当采用精确 Q 值时的效果：



当采用优化的模糊取值时的效果：



**效果对比结论**： 当数据集数量比较小时，采用原始的 Girvan-Newman 算法效果更好，但当数据集数量逐渐变大时，算法的效率会因为时间复杂度太大而逐渐变得很低，进而导致效果越来越差。

**四、 思考题**

1、影响算法效果的关键因素有哪些？

A. 数据集规模： 当数据集较小时，建议采用 Girvan-Newman 算法对社区进行发掘；当随着数据集的不断增大，建议采用对于 Q 值的模糊处理的方式简化计算，进而提高算法效率，避免因时间复杂度过高而导致算法无法计算出结果。

B. 数据集对象间的联系复杂度： 当数据集间数据的复杂度较低时，算法的计算次数会大大降低，进而效率会大大提高。

1. 如何评估算法的效果？
2. 运行时间： 程序需要在规定时间内解决问题，因此考虑算法的时间成本也是很重要的，过高的运行时间在特定环境下是不被接受的。
3. 输出结果： 输出结果往往是我们最关心的评估标准，因为这是解决问题的关键。一个算法如果不能正确的解决问题，那么这个算法就是无用的。
4. 存储要求： 程序还对存储提出了要求，如果程序要对中间结果不断进行大量的存储，当计算机不能满足要求时，这也是不可接受的。