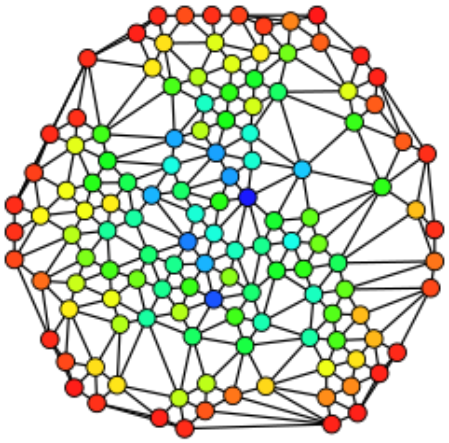
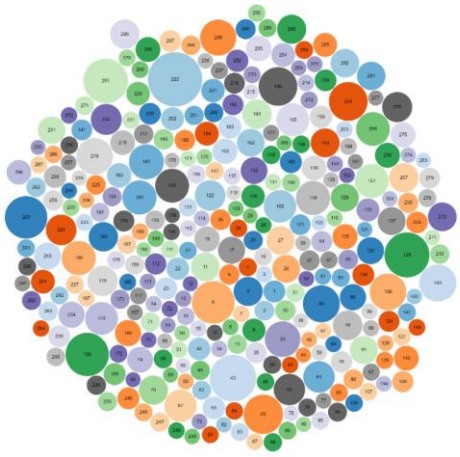
图的可视分析

利用数据进行建模，再使用图论算法对数据进行分析，可以挖掘出很多有用的信息。



图的可视化



中心度可视化

本题目主要有三个部分：（1）自选数据，或者用给定数据，选择方法构建图。（2）实现核心算法，包括基础算法（最短路径，最小生成树，连通分支，中心度等）和提高算法，并从算法得到的结果中挖掘出有用的信息。（3）对结果进行可视化（D3.js / QT / OpenGL 等）展示。本学期将提供 D3.js 的框架或示例代码（9-10 周左右）。

数据包括

(1) 豆瓣电影数据：

movies.csv: 电影数据，包括电影名称，类型，上映日期，影片名等。

comments.csv: 影评数据，包括电影名称，影评人和评分（评分包括从力荐，推荐，还行，较差，很差 5 个等级，对应豆瓣评分的 5 到 1 星）。

movies.cvs 与 comments.csv 里面有不相同的电影。

(2) 学术论文数据：

papers.csv: 包括论文发表的会议，年份，题目，作者等。

不一定要使用全部的数据（比如建模时不用评论，可不用评论数据），但肯定是使用的数据量越大越好。比如使用些复杂度高的算法，可以用部分数据。

(3) 自选数据：可以自选数据，但数据量不宜过小。

数据集大小不作为评分依据，但是过小的数据集不能体现图分析的本质（如果肉眼就看出来，就不用程序分析了）。

(一)图的构建

有不同的构建图的方法，例如，对于豆瓣电影数据可以：

①以电影为结点，使用共同的影人数量、共同的评论者以及评分的相似、类别构建边；

②以影人为结点，使用共同参与的影片等信息构建边；

③以影评人为结点，使用共同评论的影片以及评分构建边。

也可以以上述两种或以上实体作为结点，

核心在于如何构建图，以及边的权重如何设定。

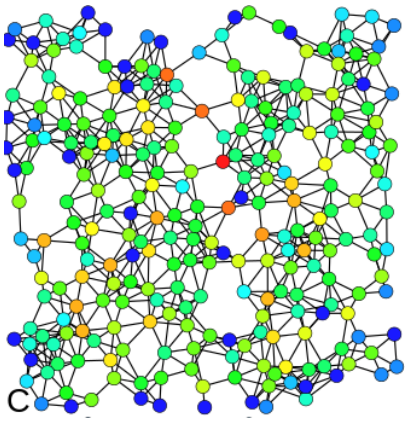
(二)基本算法

基本算法包括最短路径，最小生成树，连通分支和中心度四个算法。

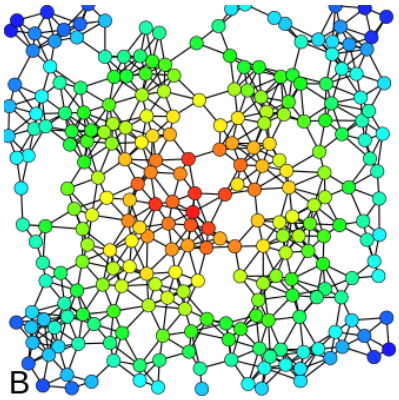
(1) 最短路径：输入任意两个节点，计算出节点间的最短路径，结果中包括路径和权值。

(2) 最小生成树：给出图中的最小生成树，输出最小生成树的结构。

(3) 节点中心度：



介数中心度



紧密中心度

①介数中心度（Betweenness Centrality）用来衡量某一个节点出现在其他节点对间最短路径上的次数；节点位置越关键，则该指数越大。

②紧密中心度（Closeness Centrality）用来衡量某一个节点，和其它所有节点间最短路径距离之和；节点越中心，则该指数越小。

要求计算图中所有节点的介数中心度和紧密中心度。

(4) 联通分量：

给出无向图中的连通分量，输出的结果中包括每个连通分量的结构。

要求实现其中三个。

注：输出格式和方式无具体的要求，例如你做了可视化界面，可以在可视化界面里面展现；如果只是控制台，就在控制台展示，但请务必在作业文档中明确说明。

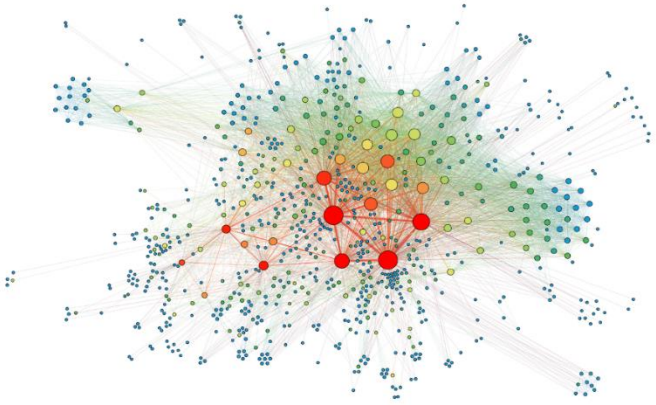
(三)提高算法

对于有余力同学，可以额外自选一些课本外的算法实现，例如 Community Detection 的一些算法。Community Detection 是一种图的聚类算法，即将点划分为不同的 Community, 使每个 Community 内的边尽量多, 不同 Community 之间的边尽量少。[1]是一篇关于 Community Detection 的综述性质的文章，而[2]也给出了 Girvan-Newman Algorithm 的实现方法。当然，提高算法不只局限于 Community Detection，也可以自己选择算法实现。

注：提高算法主要是“尝试”，不要求效果多么好，如果有些算法实现困难，或是时间复杂度高，可以实现简化版本，或者只使用一部分数据实现或展示结果。

(四)可视分析

利用可视化方法分析数据，展示结果。例如输入节点编号，或者点击两个节点，能够可视化其最短路径。推荐使用 D3.js。当数据较多，显示起来比较费劲的时候，比较厉害的同学可以考虑做 Edge bundling 或 Node aggregation（对节点或边进行合并，点击可查看局部结构等），如果时间不够，可以只可视化一部分关键的数据。



分析是指建模并从图中挖掘出信息的能力，比如往届同学的优秀作业中，同学以电影人为节点，根据合作电影，或两人指导电影的相似情况的情况，赋权建图，并将最小生成树算法场景模拟为：“一次电影界的交流会，举办方想召集所有的电影人（导演和演员）前来参加，从任意一个人开始，问如何联系才能最轻易地说服所有人前来参加交流。

D3.js 资料

D3.js 是用动态图形显示数据的 JavaScript 库，一个数据可视化的工具，可方便绘制力导向图。提供了一系列操作网页元素的方法，先选中某个元素（select 方法），然后对其进行某种操作。下面给出一个 D3.js 的例子。

http://bl.ocks.org/eesur/be2abfb3155a38be4de4

其余的资料可参考：

D3 官网：https://d3js.org/

《Getting Started With D3》

《D3 Tips and Tricks》

《Interactive Data Visualization》

《D3.js 数据可视化实战手册》

https://github.com/NickQiZhu/d3-cookbook-v2 有一些例子。

在学习 D3.js 之前需了解 HTML5+CSS3 以及 JavaScript。

此外，还可选择其他可是工具，如 QT 或 OpenGL。

《C++ GUI Programming with Qt4》

《Advanced Qt Programming》

《OpenGL Programming Guide》等。

但是还是建议 D3.js，QT 如果想实现力导向图，似乎还要装 VTK 或其他库，比较麻烦，且本学期将提供 D3.js 的框架或示例代码（9-10 周左右）。

参考文献

1. Newman M E J . Detecting community structure in networks[J]. European Physical Journal B, 2004, 38(2):321-330.
2. Girvan M , Newman M E J . Community structure in social and biological networks[J]. Proc Natl Acad Sci U S A, 2001, 99(12):7821-7826.