图分析大作业文档

刘斌（2014013466） 李书昂（2014013431）

2015年12月30日

1 项目背景

图论在科技、社会上的应用日趋广泛，城市路网，社交网络，引文网络等众多类型的网络中都需要大量用到图论建模后使用相关理论去解决、优化相关问题。因此对图的分析并将分析结果可视化就显得尤为重要。

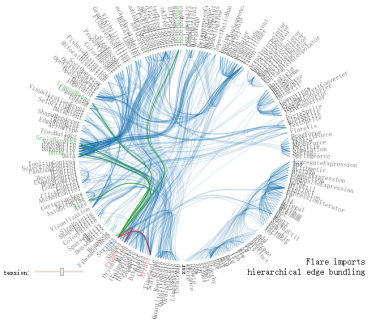

图 1: 引文网络示例

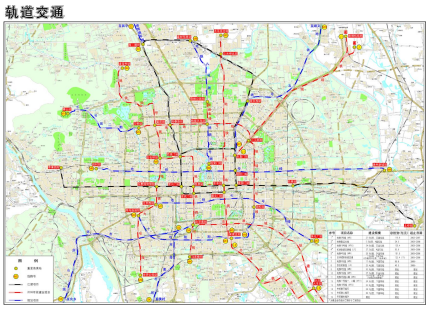

图 2: 城市路网示例

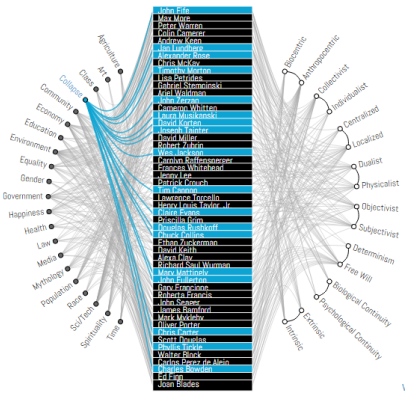

图 3: 概念图示例

2 项目概况

我们完成了包括数据采集，网络图的构建，核心算法的实现，可视分析等所有必做功能和选作功能。其中数据采集自豆瓣，网络图使用json文件储存，核心算法使用JavaScript实现，可视分析使用Html + JS + CSS与用户进行交互。

3 项目文件使用说明

首先运行服务器脚本server.bat（注意该脚本的位置应一直位于项目根目录下），然后在浏览器地址栏输入“$http://localhost:8000/$”即可使用。

4 数据采集

建立节点：访问豆瓣电影($http://movie.douban.com/$)，选择豆瓣电影top250榜单，解析Html，获得当前页面的所有电影的详细信息，建立电影节点

建立边权：进入每一个电影的评论页面获取所有相关的用户评价。构建电影为节点网络（无向图），定义网络中边的含义，如果两个电影节点的评价中有一个相同用户，则边权+1，根据评分的相似程度确定遍全的小数部分（1/(abs(w1-w2)+1)）。如果两个节点之间没有共同的用户，则视为这两个节点不联通。

5 网络图的构建

由于采集下来的数据保存格式为txt，无法方便的读入网页，因此我们使用C++对txt文件进行处理，将其转化为json格式的文件。具体代码请查阅“data/changeDateToJSON/”目录下的changeDateToJSON工程。在网页中，节点被映射为圆，边被映射为直线，且直线的宽度为与边的权值相联系。此处我们使用的权值到宽度的函数是：

$$width = Math.sqrt(weight)/2;$$

6 服务器搭建

由于网页中的数据需在打开网页时从本地加载，因此需要搭建一个简易的服务器以便传输文件。我们使用的是python2.7来搭键服务器，其源代码在源代码根目录下的server.bat中，该代码仅一行：python -m SimpleHTTPServer 8000。

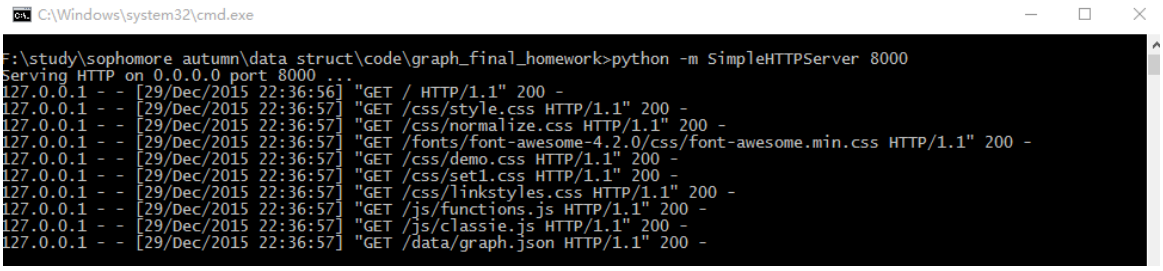

图 4: 简易服务器效果图

7 核心算法的实现

7.1 节点对的最短路径

7.1.1 算法说明

由于此次项目中需要用到大量的最短路径，为了减轻后期计算压力，因此我们并未采用dijkstra算法来计算单源最短路径，而是在页面加载时使用warshall算法计算出所有节点对的最短路径和最短路径权值。在需要使用时直接查询即可。

7.1.2 操作说明

在起点和终点框中输入起点和终点的index后点击确定即可得到最短路径的可视化表达。路径中涉及的点为红色，边为深黑色，未涉及的点颜色不变，边的不透明度变低。左下角将输出该路径的权值和。

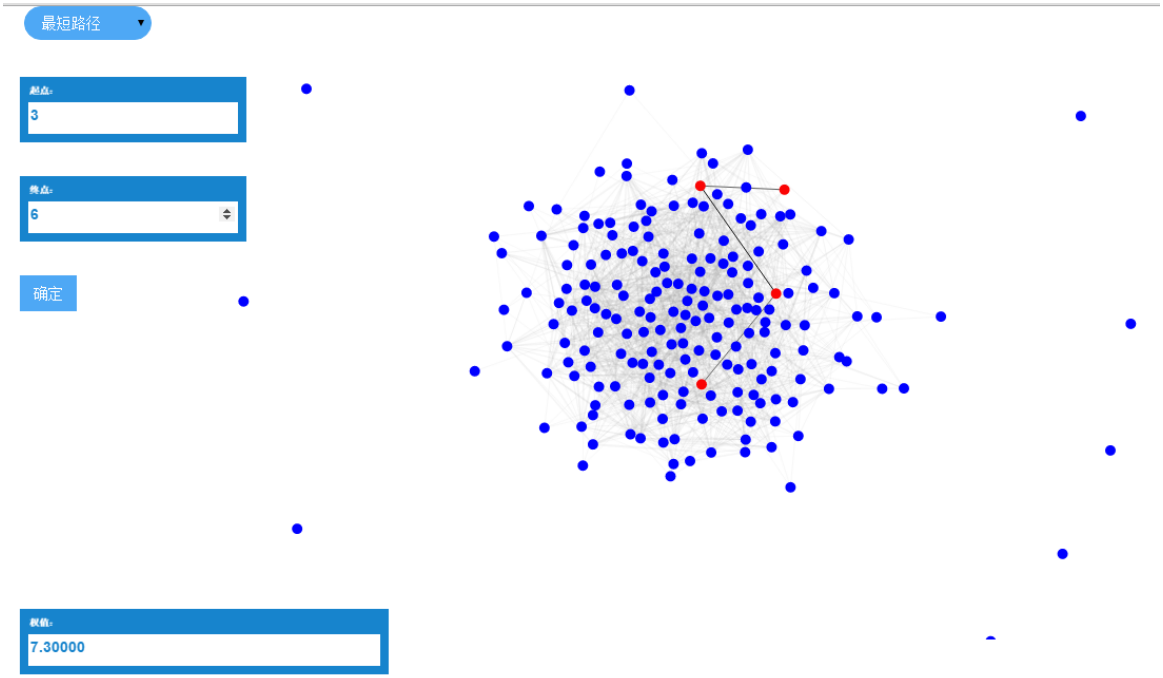

图 5: 节点对的最短路径—效果预览

7.2 最小生成树

7.2.1 最小生成树算法说明

最小生成树使用了prim算法来求解最小生成树。

7.2.2 操作说明

输入根节点，调节最小权值限定，即可得到最小生成树的可视化表达。其中跟节点用黄色标注，其余属于该生成树的节点用红色标注，不属于该生成树的节点颜色不变。树中的边颜色为深黑色，不在树中的边不透明度变低。左下角将输出该生成树的权值和。

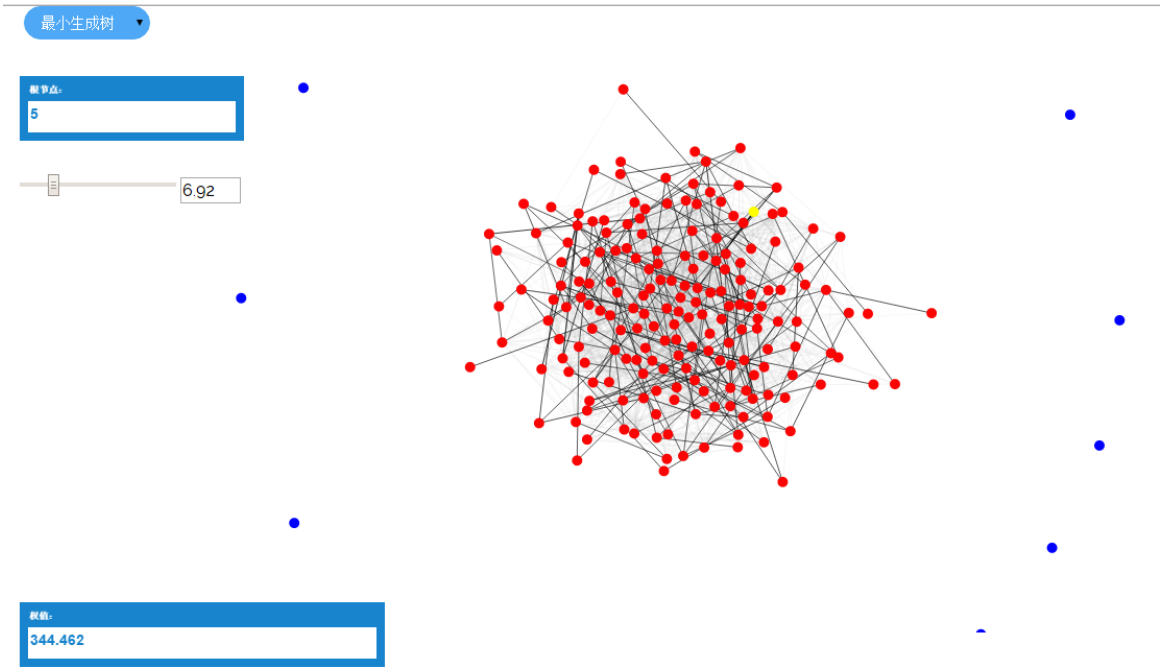

图 6: 最小生成树—效果预览

7.3 中心度

7.3.1 介数中心度算法说明

介数中心度使用warshall算法的结果，遍历所有节点对的最短路径即可获得所有节点对应的介数中心度。

7.3.2 操作说明

当选择中心度时，网页默认显示介数中心度。此外，点击介数中心度按钮即可得到介数中心度的可视化表达。

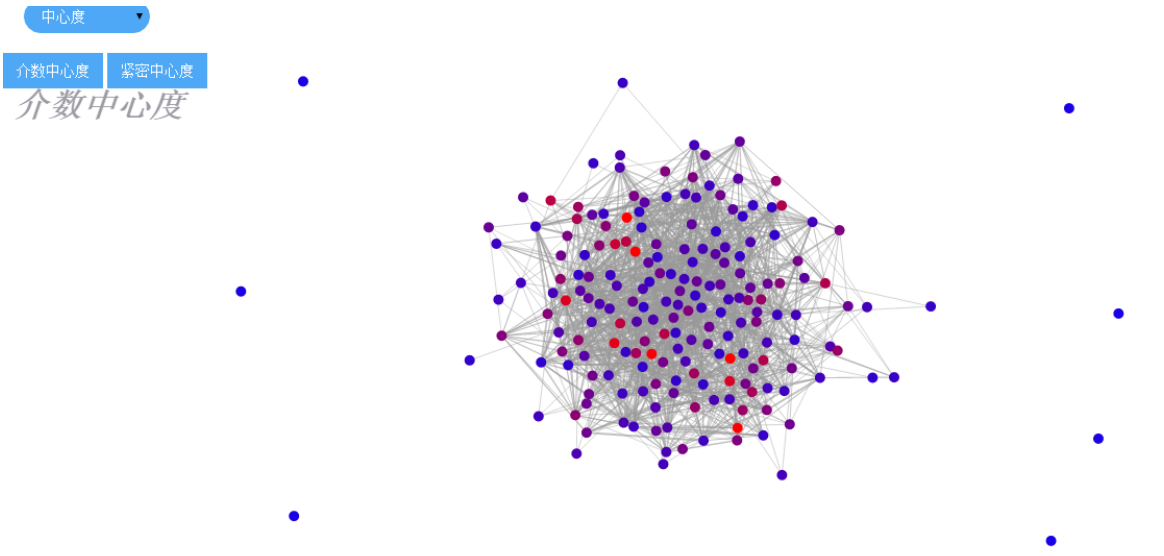

图 7: 介数中心度—效果预览

7.3.3 紧密中心度算法说明

介数中心度使用warshall算法的结果，遍历所有节点对的最短路径即可获得所有节点对应的介数中心度。

7.3.4 操作说明

点击紧密中心度按钮即可得到紧密中心度的可视化表达。

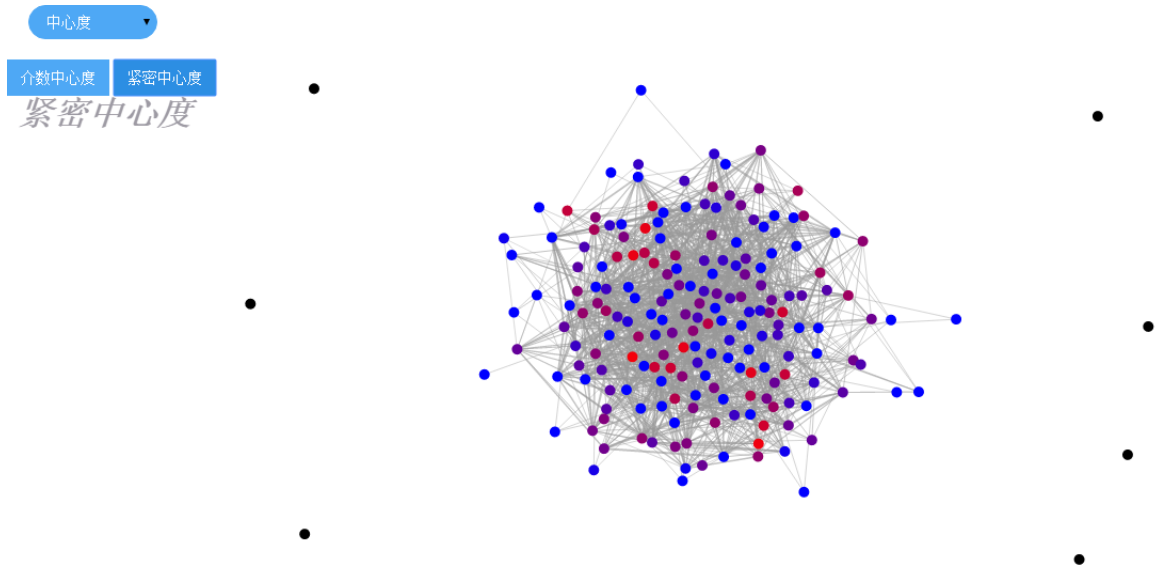

图 8: 紧密中心度—效果预览

7.4 连通分量

7.4.1 连通分量算法说明

连通分量中使用了深度优先搜索算法对所有节点进行了一遍遍历，并在遍历过程中对每一个节点赋予特定的颜色值（将与当前节点连通的节点颜色设置为当前节点的颜色，若无节点与当前节点连通，则重新寻找一个搜索起点，并重新设置一个颜色值。）同时为了处理“阈值较小时边很多会遮盖节点，影响节点的显示，阈值较大时边很少，不容易从图中看出”的矛盾，需要在网页中我们根据将要展示的边的条数来调整边的不透明度。我们采用的函数是

$$opacity = 1/(1 + Math.log(link\_count + 1))$$

其中$opacity$为边的不透明度，$link_count$为将展示的边的数量，大致范围为0到1200。经过该函数的映射，边的不透明度的范围约是0.1到1，且边的数量越少，不透明度越高，能达到较好的展示效果。

7.4.2 操作说明

拖动滑动条改变最小阈值t或直接在输入框中输入阈值t即可得到阈值为t时的连通分量可视化表达。其中相同连通分量的颜色相同，不同连通分量的颜色一般不同（不过当连通分量的数目大于20时便会重复）。同一个连通分量内的边与两端顶点的颜色相同。

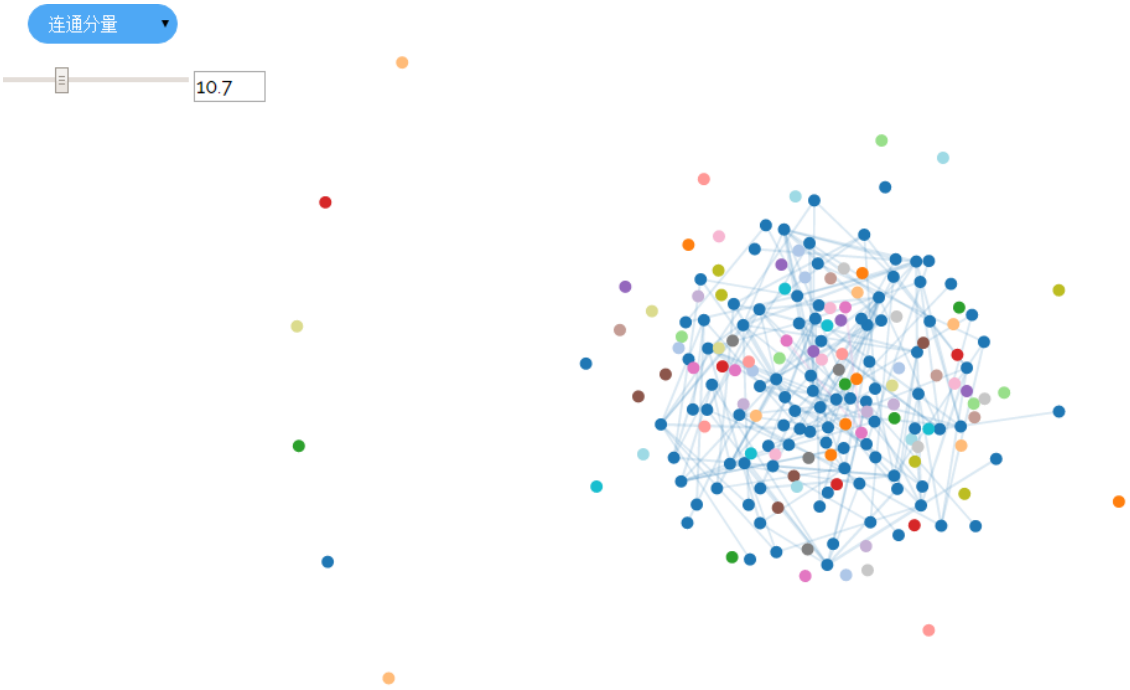

图 9: 连通分量—效果预览

8 总结

感觉图论大作业虽然分值不高，但真的很有难度，也可以让自己学到很多东西。我们学会了使用python写爬虫获得数据，学会了使用d3展示图以及基本的交互，还学习了html的各种控件的使用，css的写法。同时用js实现相关算法，并且进行展示。感觉几天的刷夜，可以做出来一个如此漂亮的界面，非常开心，也有非常大的收获！