图分析大作业报告书

小组成员：

崔殷庇 201508

李相赫 2016080042

编程环境：

操所系统：Windows 10

编程语言：JavaScript，C++

核心算法：

最短路程：

使用了Dijkstra算法。

最小生成树：

使用了prim算法。

中心度：

我们在求最短路程的过程中记录下了所有节点之间的最短路程，然后直接在计算closeness，betweenness中心都的时候利用了之前计算的两点之间的最短路程的数据，可以说也是利用了Dijkstra算法。

连通分量：

使用DFS，先建立一个节点的数组初始值都设为0，从第一个节点开始一个一个节点的搜索，搜索有边连接的所有节点，搜索到的节点在数组中的值改为搜索次数（搜索次数即为该节点所在的连通支）,如果一个一个搜索时该节点在数组中的值已经不是0，那么就可以直接跳过该节点，搜索下一个节点了。

数据提取及网络构建：

数据提取：

此次大作业的数据利用了助教提供的user.csv，对此数据进行了一点优化。

我们把一个电影当作一个节点，如果一个人看过不同的几个电影，那么这几个电影之间就产生了一条边，权是该节点的重复出现次数。举例子，如果一个人看过电影A，B，C，那么电影A与电影B，电影C之间就产生一条边。为了进行优化，我们把user.csv中的控制用户看过的电影数为2-6。最终剩下一千多个节点和两万个边，在可视化过程中数据太多太卡了，于是利用了其中的一部分数据。

网络构建：

使用了json格式，里面有links和nodes，分别表示节点之间的边和节点的编号，links中有source，target和weight,cost，分别表示两个边的端点,weight和cost指该边的权值。 LinksAll.json和movie\_id.json是数据优化之后的网络构建，但可视化过程中使用的部分数据在data.json里，而且movie\_id.json里写了编号对应的电影名。

GUI使用说明：

界面中有四个按钮，

1.shortest path是最短路程，使用方法为，点击shortest path按钮之后，再点击左图中的两个节点，那么根据搜索结果，图中的边和节点和边会变色。

2.Centrality是中心度，点击该按钮之后下面会出现closeness和betweenness按钮，点击即可。左图中的节点大小会根据要求变化。

3.Connected Component是连通分量，点击之后调节下面的滚动条该边阈值。

4.Minimum Spanning Tree是最小生成树，点击按钮之后，再点击左图中的节点，最小生成树的边会变色（要需要一点时间）。