

# 图分析说明文档

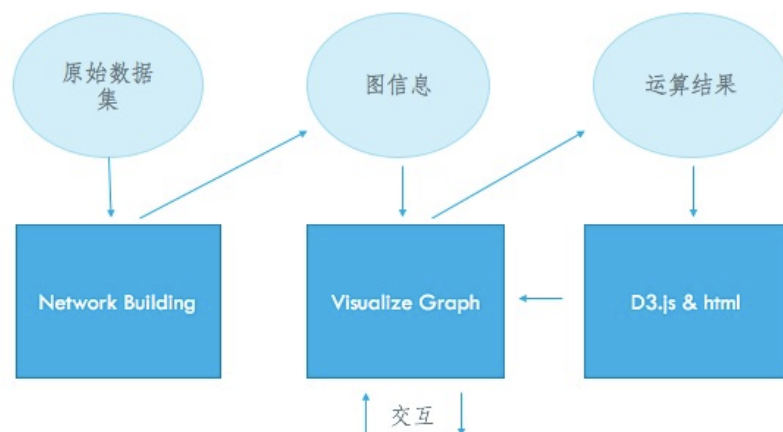
江俊广 2015011584  
毛誉陶 2016013249

## 目录

<b>概述 .....</b>	<b>1</b>
选取数据集	
流程及框架	
运行环境	
<b>必做项说明.....</b>	<b>1</b>
根据用户指定计算最短路径	
根据用户指定计算最小生成树	
根据用户指定（边阈值）计算图的连通分量	
计算节点的中心度	
<b>选做项说明.....</b>	<b>4</b>
网络图的构建	
数据采集	
可视化分析	

## 一、 概述

- 选取的数据集：「豆瓣影评数据」和自采数据集「知乎话题数据」。
- 框架及流程



我们的图分析程序主要由三部分构成。

首先是用C++进行网络图的构建（NetworkBuilding），由于该部分占用较多的时间和空间资源，在得到结果文件（"output/network.csv"）后就不再运行。

然后用C++进行核心算法的实现（VisualizeGraph），并且将计算得到的结果输出到json文件。

最后用QT和D3实现算法的可视化，QT主要用于接受用户的输入，D3实现了图的可视化。

■ 程序运行环境：Mac

## 二、必做项说明（核心算法的实现）

由于全图较为稀疏，因此采用邻接表存储。每个节点存储所有关联边的编号，同时每条边也存储所有关联节点的编号。设总节点数目为 $n$ ，总边数为 $e$ 。

### ■ 根据用户指定计算最短路径

最短路径采用Dijkstra算法，最小生成树采用Prim算法，两种算法本质上都是优先级搜索，不同点在于优先级计算方式不同，因此程序中统一采用优先级搜索PFS进行

实现。

#### 单连通域优先级搜索（PFS）的实现算法

1. 所有节点处于未访问状态，所有节点的优先级数都为最低。维护一个优先级队列，将搜索的起始节点的一个拷贝加入队列。
2. 如果优先级队列为空，则搜索完毕。否则，从优先级队列中弹出优先级最高的节点，如果发现该节点拷贝对应的原节点已经被访问，则舍弃该节点；否则访问原节点。转3。
3. 更新原节点的所有邻居节点的优先级数，并且将那些优先级发生了更新的邻居节点的拷贝加入到优先级队列（此处采用懒惰删除策略，尽管优先级队列中可能出现同一个节点的多个拷贝，但是第2步保证了只对同个节点访问一次，其他拷贝会被自动舍弃）。至此，当前节点访问完毕。转2。

#### 多连通域优先级搜索（pfs）的实现算法

多连通域的优先级搜索（pfs）只要遍历所有的节点，从每个尚未访问的节点出发进行一次单连通域优先级搜索即可，因此总的时间复杂度为 $O(n+e)$ 。

#### 最短路径的 Dijkstra 优先级更新器

由Dijkstra算法知，通过节点 $u$ 去更新节点 $v$ 的优先级的公式如下。

$$\text{节点}v\text{的优先级} = \min \left\{ \text{节点}u\text{的优先级} + \text{边}(u, v)\text{的权重}, \text{节点}v\text{的优先级} \right\}$$

#### 边的权重说明

网络图构建（NetworkBuilding）中得到了两个用户之间的关系权重 $relation(u, v)$ ，取其中最大者记做 $MaxRelation$ 。

则在计算最短路径或者计算最小生成树中的边权重时采用如下公式。

$$weight(u, v) = \begin{cases} MaxRelation - relation(u, v), & \text{if } (edge(u, v) \text{ exists}) \\ \infty, & \text{else} \end{cases}$$

通过这种方式计算出的最短路径，保证是通过那些关系较为密切的用户找到的一条最短路径，这与我们平常的逻辑较为吻合。

## ■ 根据用户指定计算最小生成树

### 最小生成树的Prim优先级更新器

由Prim算法知，通过节点u去更新节点v的优先级的公式如下。

$$\text{节点v的优先级} = \min \{ \text{节点u的优先级} + \text{边}(u,v)\text{的权重} \}$$

其余算法与计算最短路径相同，不再赘述。

## ■ 根据用户指定（边阈值）计算图的连通分量

用户可以设定两个阈值，一个是用户关系的总阈值 $thread$ ，另外一个为用户评分相似度的阈值 $similarityThread$ 。

每个连通域通过编号 $group$ 来区分。

### 算法

1. 从网络构建后的结果文件中删去所有边的权重小于 $thread$ 或者小数部分小于 $similarityThread$ 的边，构建出一个图。
2. 从某个未访问的节点 $v$ 出发，设定其 $group$ 就是自身的编号。其所有邻居节点的 $group$ 和 $v$ 相同，再从其邻居节点出发进行同样的访问。本质上是深度优先搜寻。搜索完毕后转3。
3. 如果全图存在尚未访问的节点 $v$ ，则转2。否则全图已经访问完毕。

因此总的时间复杂度为 $O(n+e)$ 。

## ■ 节点的中心度

### 介数中心度的算法

1. 初始时所有结点的介数中心度=0
2. 对所有结点 $s$ 进行下列操作：
  - a) 若 $s$ 为孤立点，则跳过；
  - b) 否则，以 $s$ 作为源点，调用Dijkstra算法求得 $s$ 到所有结点（例如结点 $t$ ）的最短路径，每条最短路径 $P_{st}$ 上除了 $s$ 和 $t$ 之外的所有结点的介数中心度加一。
3. 由于介数中心度在路径 $P_{st}$ 和路径 $P_{ts}$ 中被重复计入，因此每个结点的介数中心度除以2。

### 紧密中心度的算法

对所有结点  $s$  进行下列操作：

- a) 若  $s$  为孤立点，则跳过；
- b) 否则，以  $s$  作为源点，调用 Dijkstra 算法求得  $s$  到所有结点（例如结点  $t$ ）的最短路径。求  $s$  到其所在连通域的所有结点最短路径的平均值。该平均值越小，该结点越紧密。

#### 算法效率分析

算法的主要瓶颈在于 Dijkstra 算法，由于使用了优先级队列，因此每次 Dijkstra 算法的时间复杂度都是  $O(n\log n)$ 。共调用  $n$  次，故总的时间复杂度为  $O(n^2\log n)$ 。

## 三、选做项的说明

### ■ 网络图的构建（以豆瓣影评数据为例）

#### 用户关系的定义

假如两个用户  $A$ 、 $B$  有看过相同的电影，则二者之间存在一条边。

该边的权重=共同看过的电影数目+评分近似程度

由于  $A$  看过的所有电影的平均分介于  $[0, 10.0]$ ，因此定义

$A$ 、 $B$  的评分近似程度

$$= (10.0 - |A \text{ 看过的电影平均分} - B \text{ 看过的电影平均分}|) \times 0.08 + 0.1$$

从而将他们的平均分之差（范围为  $[0, 10]$ ）映射到评分相似度（范围为  $[0.1, 0.9]$ ）

#### 网络图的构建算法

1. 首先从 Movie.csv 中读入电影名称+评分，采用哈希表（QHash）存储电影，保证同样名称的电影只被加入一次。
2. 从 User.csv 中读入电影名称+用户名称，同样采用哈希表存储用户，每个用户看过的电影用集合（QSet）存储。同时将每个用户存入该电影的评论者中。
3. 对所有用户进行一次遍历，分别计算他们各自看过的电影的平均分。
4. 对所有的电影进行一次遍历，将每部电影的所有评论者两两关联。
5. 输出所有的关联边的两个端点和权重。

## 时间复杂度分析

设电影数目为 $N_1$ ,一部电影最多的评论者为 $N_2$ ,全图的边数为 $M$

各个步骤的时间复杂度分别为

步骤	时间复杂度
1	$O(N_1)$
2	$O(N_1 N_2)$
3	$O(N_1 N_2)$
4	$O(N_1 N_2^2)$
5	$O(M)$

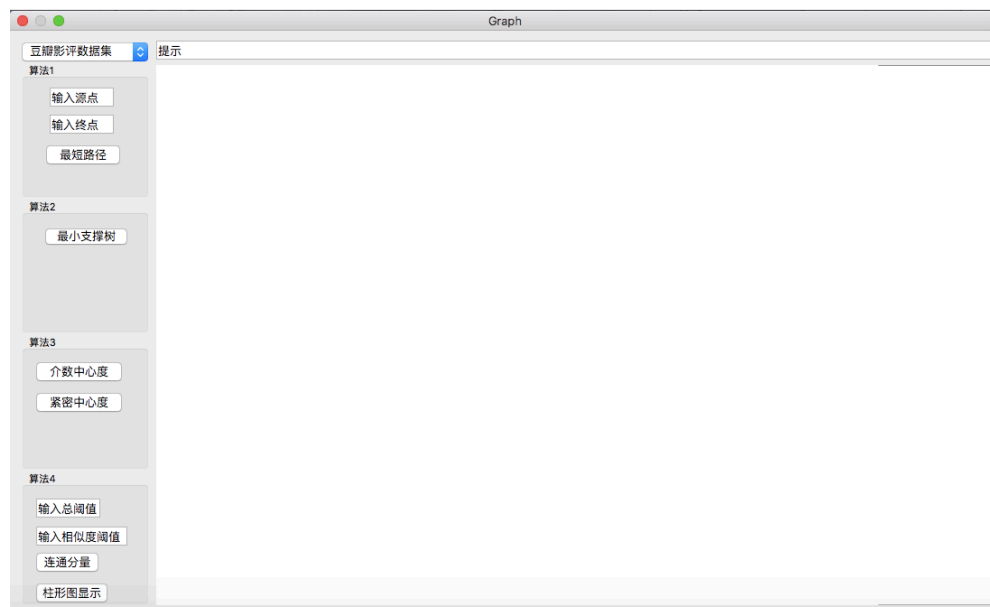
从上表中可以发现，第4步是该算法主要消耗时间和空间的步骤。

## ■ 数据采集

我们用爬虫爬取了知乎的话题和关注用户，在网络构建中话题对应电影，关注话题的用户对应看过电影的用户，取消小数部分的边权。话题共有 1622 个，用户与话题的对应关系共有 32582 对。在可执行程序中可以选择使用哪个数据集。

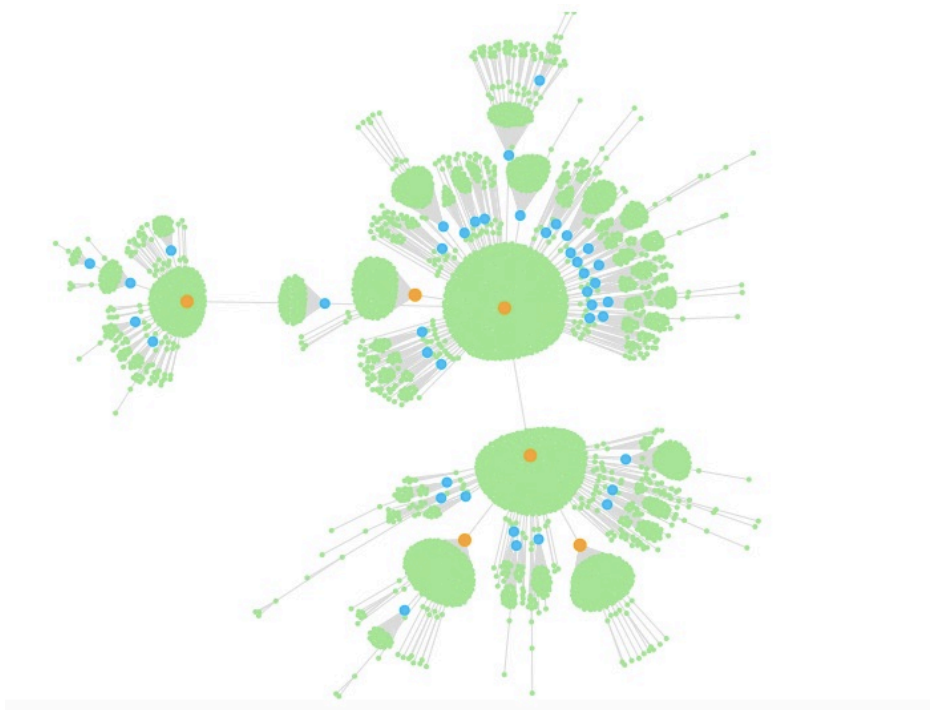
## ■ 可视化分析

### 1. 界面简介



可于左上角选择数据集。

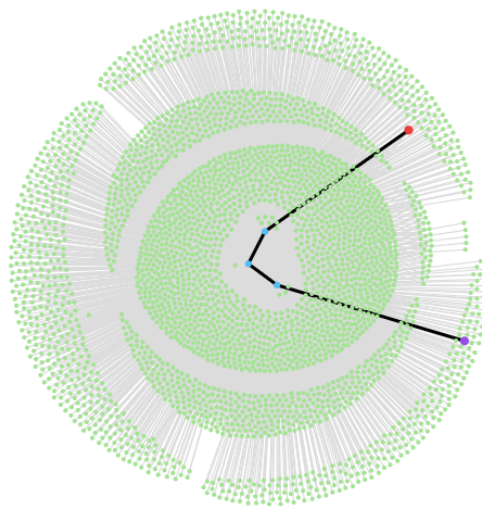
### 2. 最小生成树可视化结果



注：只保留非孤立点的节点以加快d3的显示速度。

橙色点表示度数大于100的点，蓝色点表示度数大于10小于100的点

### 3. 最短路径可视化结果



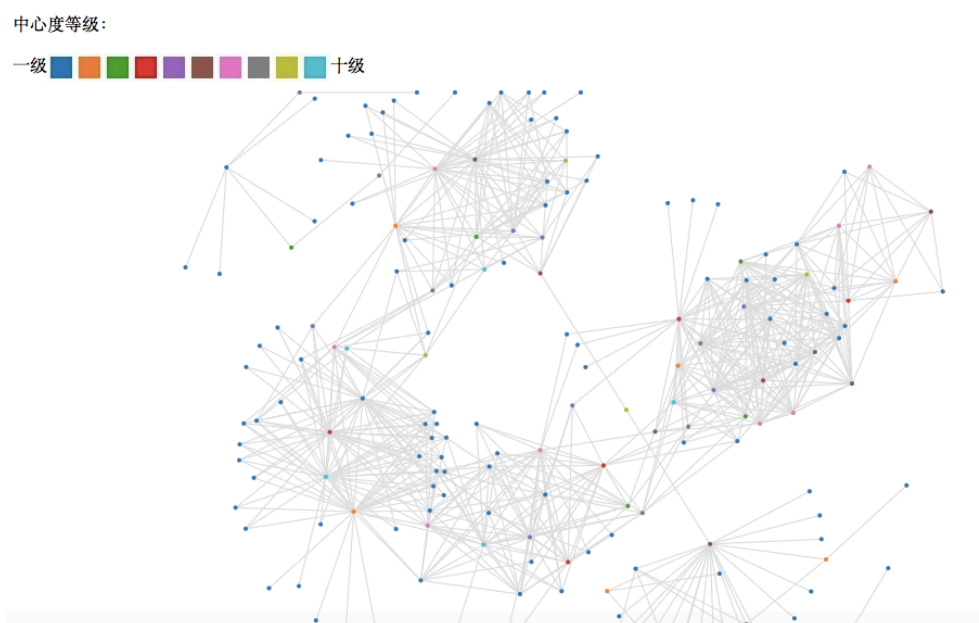
注：只保留最短路径以及和最短路径相邻的节点。（例如源点为4，终点为459时

可以得到类似结果）

红点为起点，紫点为终点，蓝点为最短路径上的点，加粗的边即最短路径。

若输入的两点之间无路径，则会在提示框提示。

#### 4. 中心度显示结果

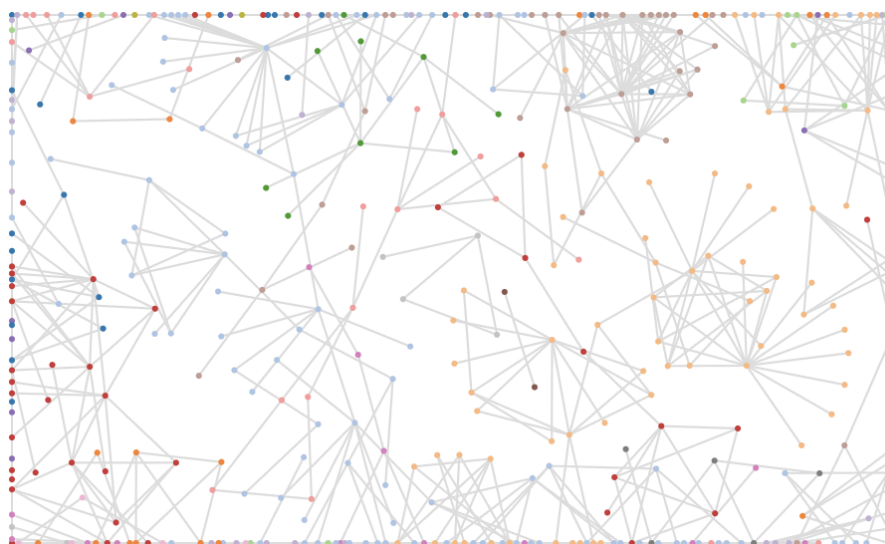


将中心度划分成十个等级，用不同颜色表示，使点的紧密度直观可见。

#### 5. 给定阈值下的连通分量可视化结果

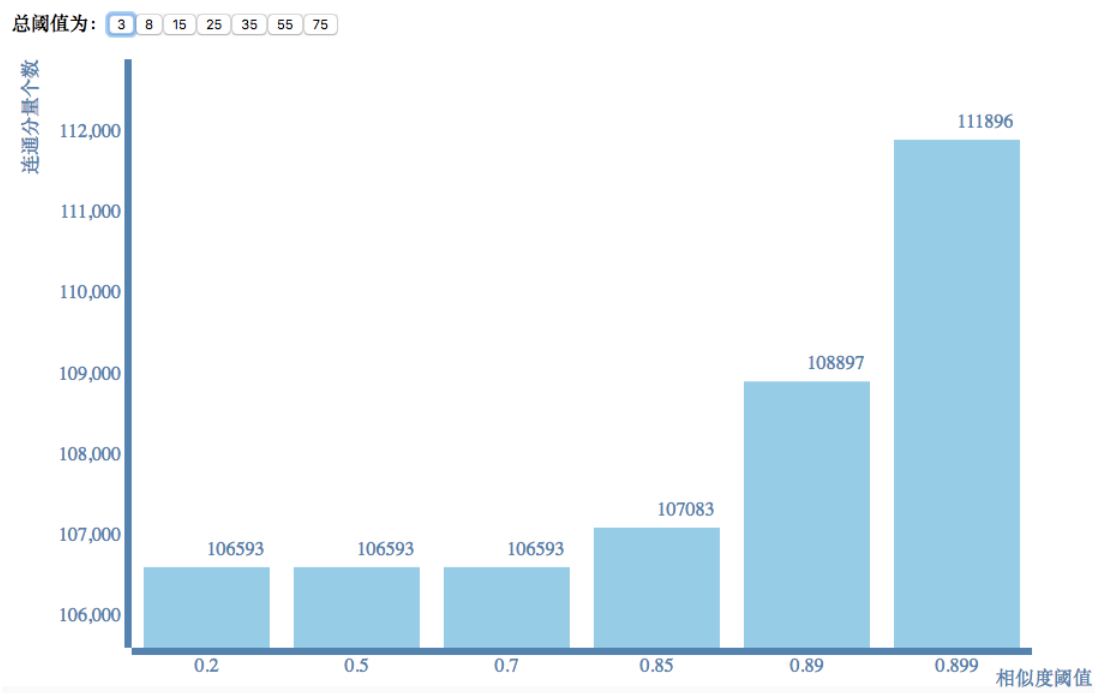


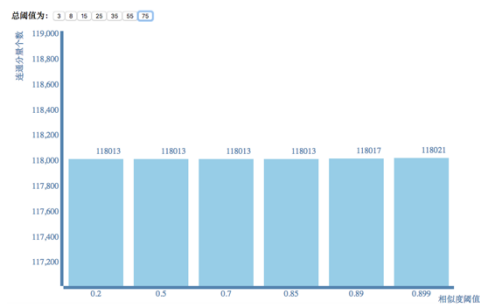
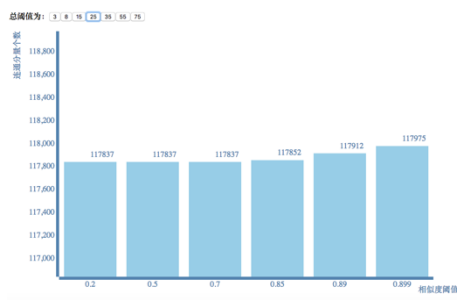




第一张图为阈值较大时的结果，第二张图时阈值小一些的结果。不同颜色代表不同连通支。（当相似度阈值在0.895至0.9之间，总阈值大于20时，可以得到较多的非平凡连通分支，其余情况非平凡连通分支数较少）

## 6. 连通分量随阈值 $t$ 变化情况





柱形图显示在总阈值给定（通过按钮进行

选择）的情况下相似度阈值与连通分量之间的关系。总阈值（整数部分）主要影响两个用户之间共同看过的电影数目，相似度阈值（小数部分）影响两个用户看过的电影的评分相似度。

由图可得：

1. 相同总阈值情况下，相似度阈值越大，连通分量个数越大；
2. 相同相似度阈值下，总阈值越大，连通分量个数越大；
3. 总阈值小的情况下，相似度阈值影响更大。
4. 总阈值在大于8之后，连通分量个数随总阈值变化程度较小。