图分析大作业实验文档

（所在小组：邵钰杉 王泽润 张迁瑞）

一、实验目标

1.网络图的构建

2.核心算法的实现

3.算法可视化

二、实验环境

操作系统：windows10 （版本号 1607）

IDE: visual studio 2015

HTML 编辑：火狐浏览器

编程语言：C++，Java Script

三、组内分工

邵钰杉：数据采集

王泽润：可视化

张迁瑞：算法部分

四、实现功能

（一）算法部分

1.节点间的最短路径

输入任意两个结点，可以输出结点间的最短路径(path:x-y-z)和这条最短路径的权值(weight),使用了 Dijkstra 算法。

2.最小生成树

对于给定的图，可以输出该图的最小生成树，输出结果是最小生成树边的个数和这些边的起点与终点，通过可视化可以得到最小生成树的结构，使用了 prim 算法。

3.连通分量

根据用户输入的边阈值，可以输出图的各个连通分量，这些连通分量的边的权值均大于等于该阈值，使用的算法是广度优先搜索。

4.提高部分

1.将复杂网络划分为两个社团

在这一部分中，我们尝试使用 Kernighan-Lin 算法将给定的网络划分为两个部分，每一部分是一个子社团，代表联系比较紧密的一部分用户。

用 Kernighan-Lin 算法最后得到的最后结果可以做到两个社团内部的边权之和减去两个社团之间的边权之和为最小。但由于这个算法的复杂度较高，所以当需要划分的网络的结点数较多时，划分的速度会变慢很多。经测试，当 n 取 50 以内的数字时，能够较快地返回结果。

（二）数据采集部分

1．算法实现概述

这个爬虫的实现没有调用 scrapy 包，手动实现了爬虫的功能。

思路是这样的，先在 https://movie.douban.com/top250 上获取豆瓣 top250 的 URL，把这 250 个 URL 存到一个 url_list 中去。

之后把对每一个 url_list，进入这部电影的 reviews 部分。获取每一条评价的用户名，评分，以及电影编号，存到 info_list 中去。把这 250 个小的 info_list 整合成一个大的 info_list。

之后再针对这个大的 info_list 做处理，把这个大的 info_list 通过算法转换成题目要

求的形式，最后输出到”movie.txt”中去。

具体的算法实现见代码注释，代码注释详细地写了我的实现过程。

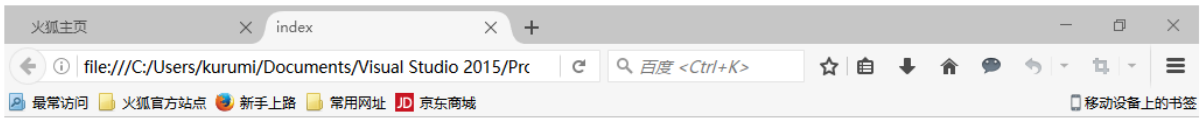
2．遇到的技术难题

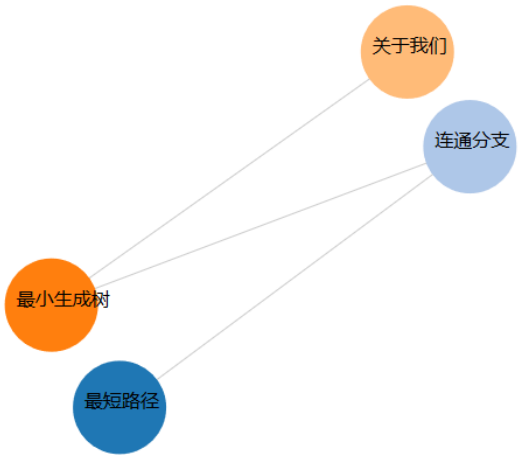
2.1.模式匹配时遇到的难题。爬取每个网页中所需的 url 时，需要进行严格的判断提取，以免爬到了非目标的 url。

2.2.豆瓣的反爬虫机制

豆瓣有反爬虫机制，如果一定时间内多次爬取访问的话会封掉你的 ip。针对豆瓣的反爬虫机制，我采用了更换 User-Agent 和轮换 ip 的方法。使用的 ip 均爬取自网站 http://www.xicidaili.com/nn/1。不过这也会带来一个问题，就是有的 ip 是无效的，这样有时就会造成了爬虫运行不稳定。不过控制每次读取的数量可以能够解决这个问题。

(三)可视化部分





基于在 JavaScript 中实现以及小组其他成员提供的算法，实现对数据的可视化处理，其中包括：

1.图数据的处理

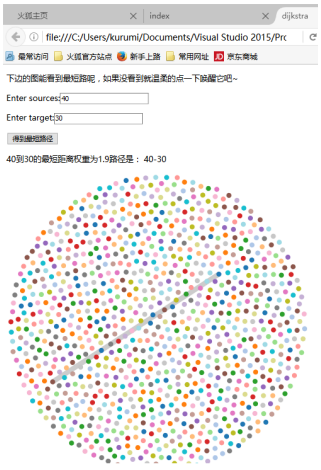
根据小组其他成员提供的算法，生成用于可视化的相应文件，分别为：

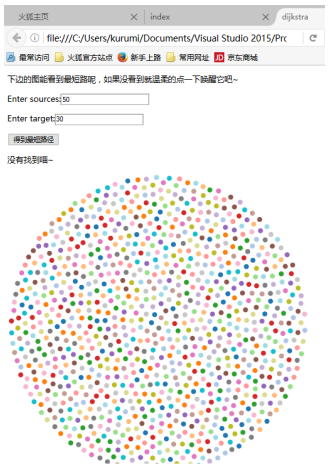
- 记录点信息的 nodes.json 和记录边信息的 edges.json.
- 记录最小生成树边信息的 trees.json.
- 记录连通分量边信息的 links.json.

实现方法：

通过 jsonmaker.exe 运行形成相应的 json 文件。

2.最短路径的可视化



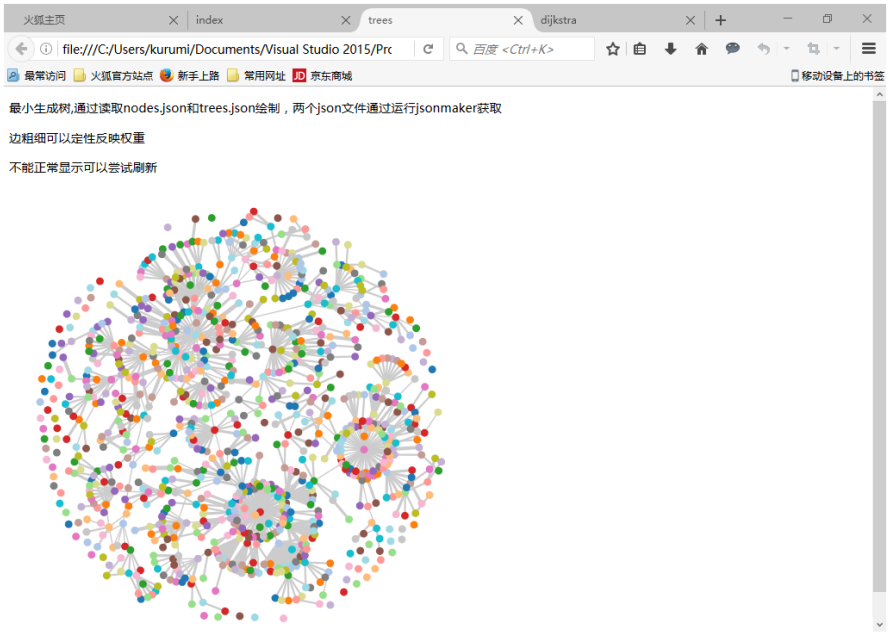


通过在 dijkstra.htm 中输入相应的节点信息，可以给出有无路径及其最短路的权值和路径信息，并在图中予以显示，可以通过鼠标放置在节点上了解节点信息。

实现方法：

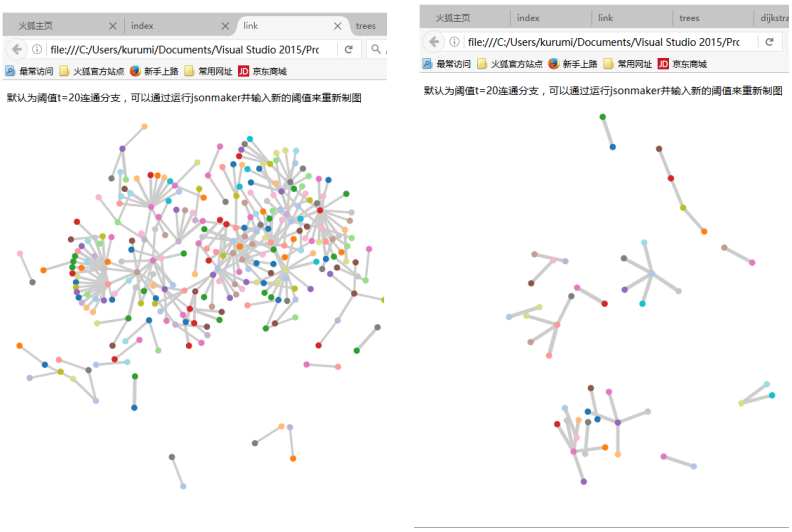
在 html 网页中改编 Dijkstra 算法，读取处理好的结点，边信息，在网页中实现计算，并将结果以 d3.js 提供的力导向图的形式显示在网页中。

3.最小生成树的可视化



通过读取 prim 算法中计算的边信息（存放在 trees.json 中），形成相应的力导向图，以边的粗细定性表示了各边的权重，可以通过鼠标放置在节点上了解节点信息。

4.连通分支的可视化



通过给定阈值计算连通分支后，形成相应文件，通过网页读取形成连通分支，通过条件判断去掉了单点和不需要的边值，可以通过鼠标放置在节点上了解节点信息。

上图分别是 20（左侧）和 30 阈值下的连通分支，可以发现随着阈值变大连通分支变得稀疏了。

5.其他界面设计

利用 d3.js 的功能对数据和界面进行了相应的优化。

四、实验流程

- 核心算法编写
- 数据获取
- 可视化处理

五、操作说明

（一）可视化部分

1.请在火狐浏览器下打开网页，文件中附上了火狐浏览器的安装包

2.打开 index.htm 即可访问所有可视化结果

3.对不同连通分支的可视化效果检查需要先运行 jasonmaker.exe，按照相应提示进行操作即可，再打开 index.htm

4.遇到网页傲娇的情况需要刷新

六、实验结果

包含在文件中

七、实验感想

前端这种东西其实与实现核心算法相比在实际编写中要简单一些，但是我没有学过啊！所以一开始基本都是在学习各种姿势，最后用到的东西也不是很多，而且掌握之后是非常容易的，成果也很直观，完成之后很开心~

在图分析之中，数据的可视化我认为是非常重要的，它所反映出的信息要比其他形式的媒介丰富的多，人对它的接受能力也强的多，用这次对豆瓣电影用户的分析来说，可以直观的看出用户之间的联系紧密程度等，因此是很有意义的。

关于 d3.js 这个数据处理功能真的很强大，还有很多知识值得去挖掘。

一个优秀的前端程序猿起航了~

--王泽润（可视化）

期末考试后实现拓展算法的时候查了许多资料，发现社群分割有许多有趣的算法，其中

一种我还曾经在数据清洗的论文中看到过，那一瞬间感觉图论真是挺有用的，应该好好学，但这时已经是期末考试结束之后了。

本来还想好好在社群分割这方面探索一番的，但是时间不够了，建议下次大作业下学期开学再交。

以及图论将来有缘再见了～

--张迁瑞（算法）

第一次接触爬虫这种东西，感觉还是蛮新奇的。话了大概 30+小时才写出了一个能够成功运行的爬虫里面的内部结构，没想到最后花了更多的时间去应对豆瓣的反爬虫机制。在这个过程中也涨了许多姿势，比如轮换 ip，换 user-agent 啥的。爬虫与反爬虫的博弈也是很有趣的哈。不过我花了好多时间自己实现轮子，结果最后发现早已有人写好了框架，真是欲哭无泪啊。

之后写爬虫的机会应该也是不少，这一次也算是让我入门了，以后也会更加得心应手。

----邵钰杉（爬虫）