# 通过Python实现人物关系图谱



## 一、实验目的

通过代码实现针对某一段文章中的解析，得到相对应人物之间的人物关系。使用代码实现是相对于人工而言更加的方便和快捷，减少了很多的人工工作量。

本次实验使用python，对《人民的名义》、《红楼梦》进行人物关系图谱的构建。

## 二、实验思路

### 主要步骤流程

1.准备相对应需要实现人物关系图谱的文本。

2.查询相关文本中的主要角色 实现人物角色字典。

3.通过调库实现相关人物关系，得到人物间的权重等关系。我们认定，如果在一定范围内的文段中，多个人的名字出现，那么可以认定这几个人之间是有联系的。通过多段文字之间的关系，构建节点、图并将出现的次数作为权值。

4.使用软件Gephi绘制关系图，将关系可视化。

### 使用函数库

***jieba***:对文本进行分词操作

***csv***:读写CSV文件，便于后期的人物关系图谱的构建

### 使用软件

·Python 3.7（pycharm、anaconda3）

·Gephi 0.9.2:将前期得到的CSV文件进行后期的可视化

# 三、实现《人民的名义》人物关系图谱

## 主函数部分

#### 构建停用词和替换词

因为在文档中会出现大量的词汇，进行jieba分词后会出现很多干扰项。对此我们需要针对一些词汇进行停用。除此之外，针对文章中可能出现的大量人称的替换词，我们需要在实验中替换回来，这样才可以正确的对固定的人物之间进行计数操作。

并构建好后期可能会用的空字典。

```
stopwords=['吕州','林城','银行卡','明白','白云','嗡嗡嘤嘤',
           '阴云密布','雷声','陈大','谢谢您','安置费','任重道远',
           '孤鹰岭','阿庆嫂','岳飞','师生','养老院','段子','老总']
replace_words={'师母':'吴慧芬','陈老':'陈岩石','老赵':'赵德汉','达康':'李达康','高总':'高小琴','猴子':'侯亮平','老郑':'郑西坡','小艾':'钟小艾','老师':'高育良','同伟':'祁同伟','赵公子':'赵瑞龙','郑乾':'郑胜利','孙书记':'孙连城','赵总':'赵瑞龙','昌明':'季昌明','沙书记':'沙瑞金','郑董':'郑胜利','宝宝':'张宝宝','小高':'高小凤','老高':'高育良','伯仲':'杜伯仲','老杜':'杜伯仲','老肖':'肖钢玉','刘总':'刘新建',"美女老总":"高小琴"}
names={} #姓名字典
relationships ={} #关系字典
lineNames =[] #每段内人物的关系
node=[] #存放处理后的人物
```

#### 读取文件进行预处理

这里先将我们需要程序进行解析的文件导入并使用jieba分词进行分词操作。除此之外，需要将停用词去掉，将已经构建好的替换名进行替换。如果分词后的长度仅为1，说明这个词可能会 有用的情况不多，所以将其删除。

如果在当前段落中可以找到两个人名，那么可以认为这两个人之间有了联系。每当出现一次这样的情况，便增加1，从而最终得到相关的两人之间亲密度之间的权重。如果出现了新人名，那么可以进行记录然后构建关系。

```
def read_txt(path): #读取剧作并分词
    jieba.load_userdict("RMDMY.txt") #加载人物字典
    f=codecs.open(path,'rb') #读取剧作,并将其转换为utf-8编码
    for line in f.readlines():
        poss=pseg.cut(line)  #分词并返回该词词形
        lineNames.append([])  #为新读入的一段添加人物名称列表
        for w in poss:
            if w.word in stopwords:  #去掉某些停用词
                continue
            if w.flag != "nr" or len(w.word) <2 :
                if w.word not in replace_words:
                    continue
            if w.word in replace_words: #将某些在文中人物的昵称替换成正式的名字
                w.word=replace_words[w.word]
            lineNames[-1].append(w.word)  #为当前段增加一个人物
```

```
            if names.get(w.word) is None: #如果这个名字从来没出现过，初始化这个名字
                names[w.word] =0
                relationships[w.word] ={}
            names[w.word] +=1 #该人物出现次数加1
    for line in lineNames: #通过对于每一段段内关系的累加，得到在整篇小说中的关系
        for name1 in line:
            for name2 in line:
                if name1 == name2:
                    continue
                if relationships[name1].get(name2) is None: #如果没有出现过两者之间的关系，则新建项
                    relationships[name1][name2] =1
                else:
                    relationships[name1][name2] +=1 #如果两个人已经出现过，则亲密度加1
```

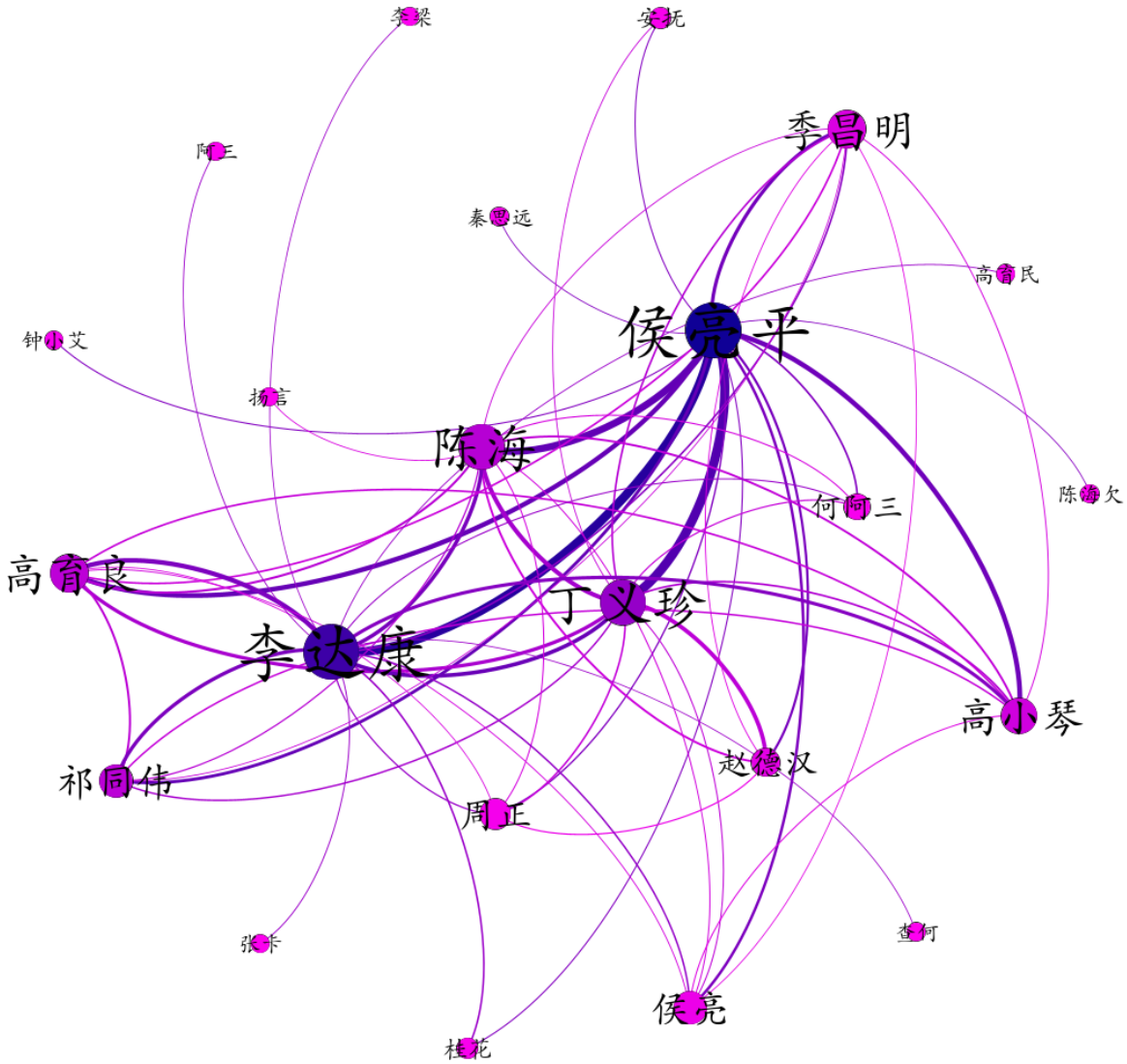
### CSV文件的生成

将之前已经构建好的关系进行再次的处理。因为每个人之间如果有了联系，那么相当于产生了词向量。每个人之间的情况不同，将节点之间连接可以看做是一种联系亲密度的反应，如果其值大，说明两人关系亲密，反之亦反。每个节点所连接的节点数可以看做是这个人的重要性，与更多的节点有连接则说明更加的重要，反之亦然。

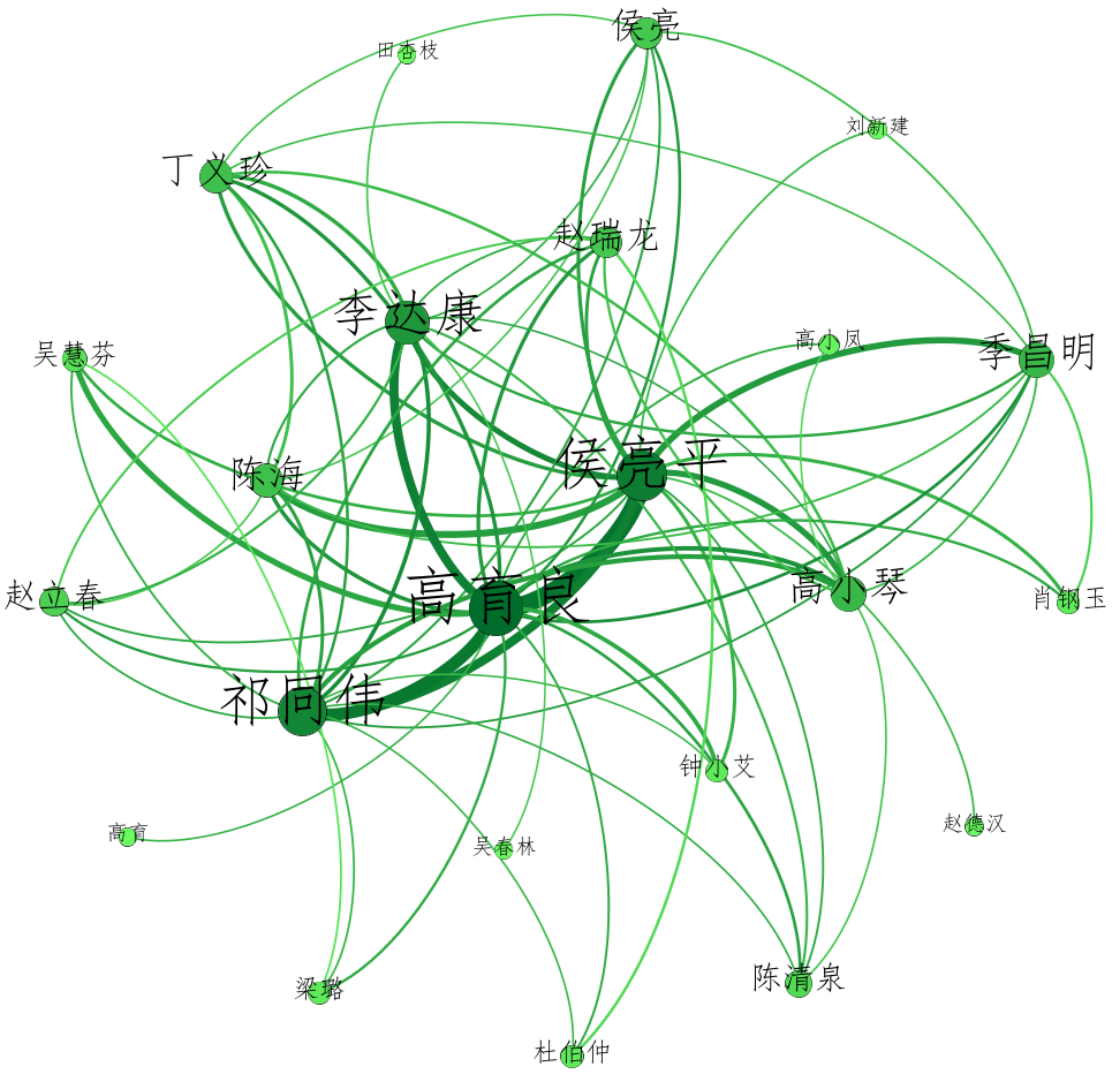
```
def write_csv():
    csv_edge_file = open("edge.csv", "w", newline="")#构建边的关系
    writer = csv.writer(csv_edge_file)
    writer.writerow(["source", "target", "weight","type"])
    for name,edges in relationships.items():
        for v,w in edges.items():
            if w>20:
                node.append(name)
                writer.writerow((name,v,str(w),"undirected"))
    csv_edge_file.close()
    #生成节点文件
    s=set(node)
    csv_node_file =open("node.csv","w",newline="")
    wnode =csv.writer(csv_node_file)
    wnode.writerow(["ID","Label","Weight"])
    for name,times in names.items():
        if name in s:
            wnode.writerow((name,name,str(times) ) )
    csv_node_file.close()
```

## 结果展示

通过上述函数可以得到两个.csv文件，使用Gephi软件对其进行引入并生成最后的图片。




上图为《人民的名义》人物关系图，其中人物节点、边均使用大小、颜色不一构成。越偏向蓝色节点越大指的人物联系越多，边越粗说明两人之间的联系越多。该图是使用 剧情概要.txt 得到的图片。除此之外，我尝试使用《人民的名义》全书进行人物关系图谱的制作。得到结果如下图：




通过上图可以看到一个大致相同的一个人物关系图谱的结果，但是和之前的略有不同。在小细节上可以看出小差别。绿色的深浅可以表示出该人物的联系多少，线的粗细可以呈现出联系紧密程度。

**结果分析**

使用两种文本进行测试，可以得出两种人物图谱。在两个图谱中可以看出，使用全书时的结果更加的全面，可能因为故事梗概细节较少，分词结果不是特别好，有个别的人名时不正确的，也会出现小人物出现在图谱中。但是使用全书进行测试时效果会好一些，但是好像还是出现了像是“侯亮平”和“侯亮”两个人物的差别，应该是分词库训练不足的原因。

# 四、实现《红楼梦》人物图谱

## 主函数部分

在该部分调用的函数库依旧是jieba分词库。对《红楼梦》全书的文本进行分词，生成最终的人物图谱。

**人物字典**

因为红楼梦中人物很多，可以先搜索出可能出现的人物，为jieba分词制作一个名字库：name.txt。

黛玉, nr宝钗, nr贾演, nr贾寅, nr贾源, nr贾法, nr贾代化, nr贾代善, nr贾代儒, nr贾代修, nr贾敷, nr贾敬, nr贾钱启, nr张若锦, nr赵亦华, nr钱槐, nr小玄儿, nr隆儿, nr昭儿, nr喜儿, nr住儿, nr寿儿, nr杏奴, nr庆儿, nr王信先儿, nr单大良, nr赵国基, nr单大娘, nr祝妈, nr田妈, nr叶妈, nr许氏, nr何婆子, nr小鸠儿, nr夏婆子, nr柳家的

大致的内容如上图所示。

### 词汇预处理

对已有的人物词典和红楼梦文本进行搜寻。同样的如果不是人物表中词汇或是小于两字词汇进行删除。如果有新的人物进行人物添加。

```
jieba.load_userdict("names.txt")#加载人物表
with codecs.open("红楼梦.txt", 'r', 'utf8') as f:
    for line in f.readlines():
        poss = pseg.cut(line)  # 分词，返回词性
        lineNames.append([])  # 为本段增加一个人物列表
        for w in poss:
            if w.flag != 'nr' or len(w.word) < 2:
                continue
            lineNames[-1].append(w.word)
            if names.get(w.word) is None:
                names[w.word] = 0
                relationships[w.word] = {}
            names[w.word] += 1
```

### 构建人物权值

构建人物间的关系时，如果出现某段落出现两个人物名，那么默认其二者之间会产生联系。

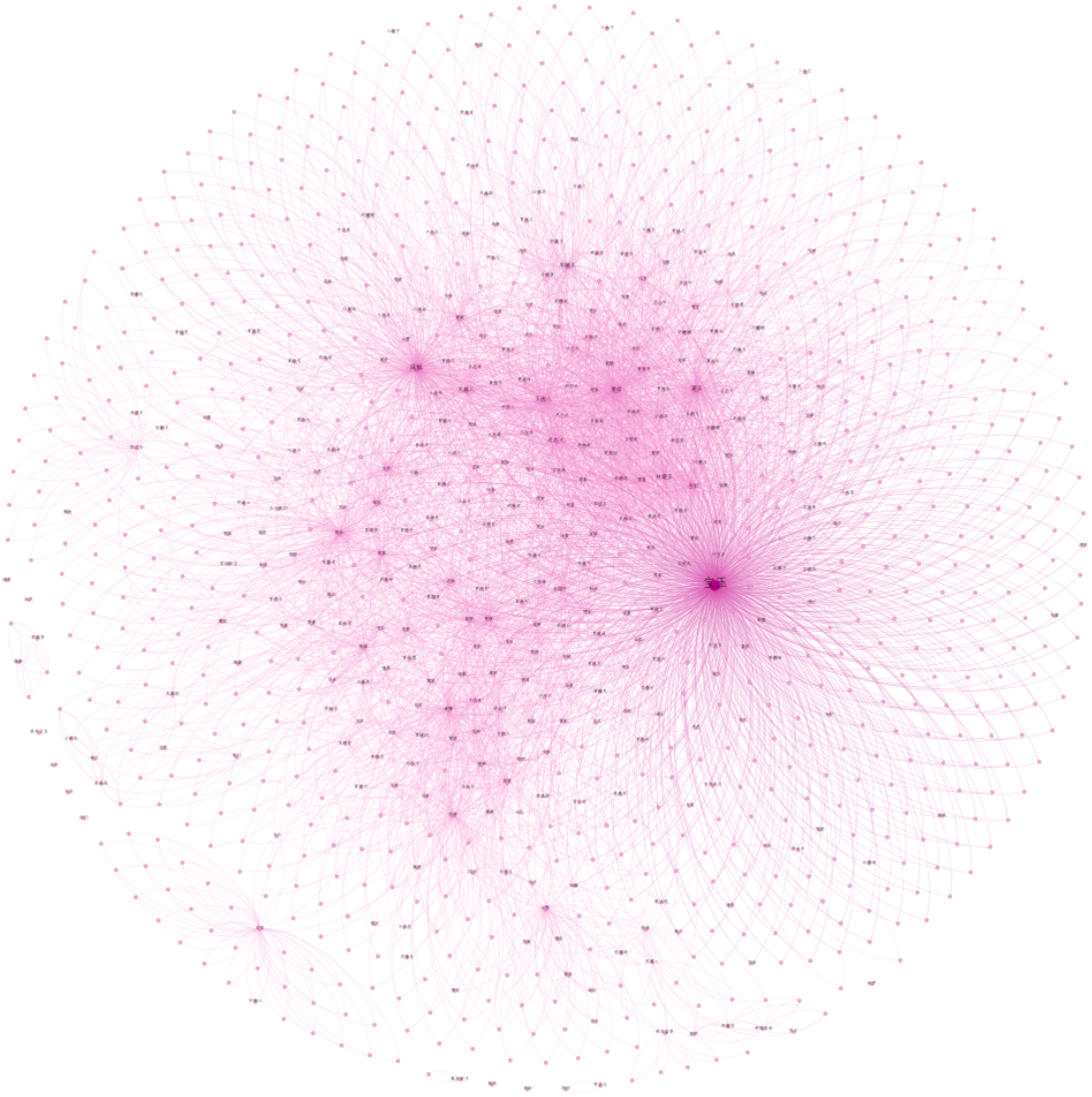
```
for line in lineNames:
    for name1 in line:
        for name2 in line:
            if name1 == name2:
                continue
            if relationships[name1].get(name2) is None:
                relationships[name1][name2] = 1
            else:
                relationships[name1][name2] = relationships[name1][name2] + 1
```

### 产生边、节点文件

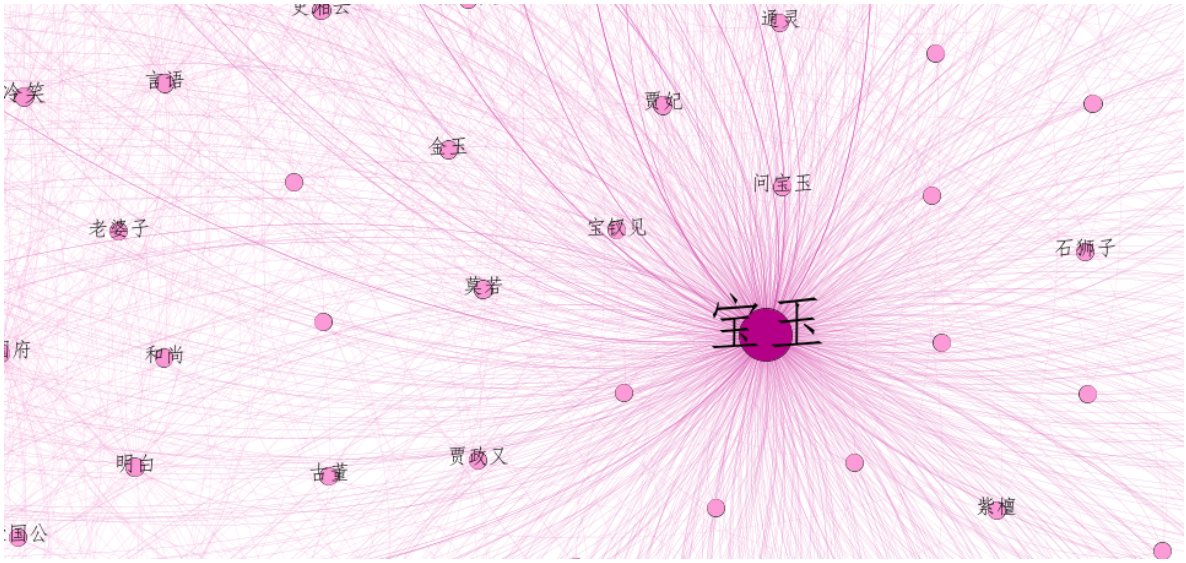
该部分代码与上面的《人民的名义》之间代码类似，但是这里生成的是txt文件。只需在excel中简单操作即可获得相关的csv文件。值得一提的是，在这个部分，我们只选取权值大于10的人物关系进行构建。过小的权值可能是假名字或者是冗余边，需要自动的删除。

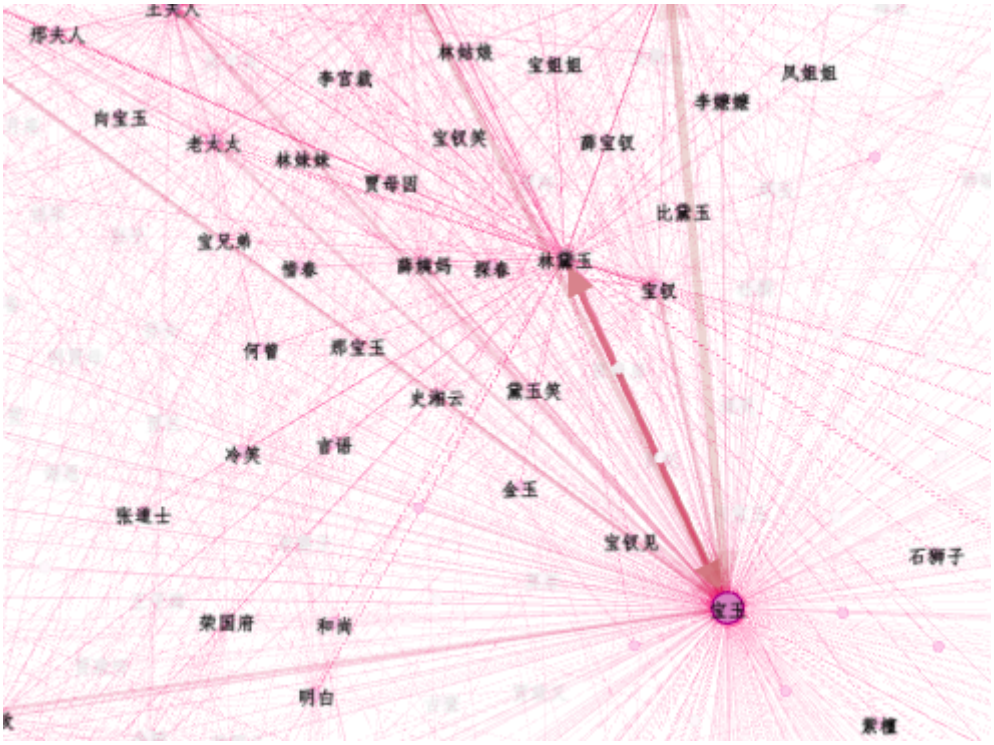
## 结果展示

通过上述得到两个.csv文件，使用Gephi软件对其进行引入并生成最后的图片。



可以看出是一个很庞大的人物关系网络。将部分放大可以看到如下画面：






**结果分析**

可以从上面的图看出，通过这个方式已经可以大致的得到《红楼梦》中的人物关系图，但是细看会发现还是有问题，因为分词系统和姓名库的不完全充足，可以看到很多明显不是人名的词汇出现在了关系图中。

我认为解决的方法可以是构建更加好的人名库，但是相对的也会耗费更多的人力。针对已有的库更明确的词汇的分类，但可能需要更多的训练。

# 五、总结

针对某个小说文本的人物关系图谱的制作，如果一个文本越详细，可能得到的关系会越清楚。除此之外，根据人物姓名库的清楚和丰富程度也会对最后的结果产生影响。越清晰的姓名库会带来更加清楚的人物关系图谱，但是前期的人工工作量也会加大很多。应当合理结合两者的优劣进行分图谱的制作。