复旦大学

本科毕业论文(设计)

论文题目：　面向电子邮件的社交网络分析系统的设计和实现

院系：　软件学院

专业：　软件工程

姓名：　张一婧

学号：　12302010076

指导教师：

2016 年 5 月 24 日

目录

摘要

伴随着互联网的发展，人与人的社交关系越来越多地借助网络软件来实现，因此也衍生了诸多社交平台。社交网络平台因为其庞大的用户群体，会产生大量社交关系方面的信息，而对这些信息的研究有着很大的价值。电子邮件作为一种常见的社交网络平台，被大量的各种类型、层次的人群广泛的应用于传递信息，因此其产生的数据具有复杂性、分散性、海量性的特点，对这些数据进行分析同样有着作用和意义。

本文在 Spark 及其图计算引擎 GraphX 技术的基础上，设计并实现一套面向电子邮件的社交网络分析系统。系统向用户提供对电子邮件进行多维度、多方面进行处理分析的功能，其中包括：关系网络构建、社区划分、活跃人分析、联系人最短路径、直接/间接联系人网络和条件筛选等。同时，系统采用 B/S 结构进行构建，并向用户提供处理结果的清晰直观的界面展示。

论文从研究背景、系统需求分析、系统总体设计、系统详细设计和实现、系统测试和总结与展望等部分具体展开。

关键词： 电子邮件；社交网络分析；Spark；GraphX

ABSTRACT

With the development of Internet, maintaining social relationships between people are more and more relying on the network software to achieve, therefore,a lot of social networking platform are produced. Because of its huge user base, social networking platform will produce a large number of social relation information,and it values a lot to study these information.As a common social networking platform,Email is widely used by large quantities of various types and levels of population to transmit information,so the data it generates is complex,decentralized and massive, and the study of there data is also of great value.

This paper designs and implements a social network analysis system, which is based on Spark and its graph compute engine.System provides a multi-dimensional processing and analysis of emails includes networking building, community partition, activeness analysis, shortest path finding, direct /indirect contact network finding and condition filtering etc.System is constructed using B/S structure and it provides a vivid and intuitionistic interface show of the analysis result.

This paper consists of the project background,the system requirements analysis, the summary design of system, the detail design and implementation of system, the system test and the summary and prospect.

Key words:Email; Social Network Analysis;Spark;Graph

第一章 引言

1.1. 研究背景及意义

伴随着网际网络的蓬勃发展，网络时代也迎来了从 Web1.0 到 Web2.0 的转变，形形色色的社交网络平台逐渐走进人们的生活，改变着人们的上网习惯，方便着人们的日常生活。社交网络平台一代代的推陈出新，从最早的 BBS，到现在的微信、微博等平台，人们越来越依赖社交网络来拓宽交际圈，维护人际关系。根据 2016 年 1 月 22 日发布的《中国互联网络发展状况统计报告》，截止 2015 年 12 月，微信的月活跃用户量已达到 6.5 亿，而在 2014 年，这个数字仅有 3.96 亿；截止 2015 年 12 月，微博的月活跃用户量为 2.21 亿，而在 2014 年，这个数字仅有 1.39 亿。其他的社交平台，相较 2014 年，用户量也有了较大程度的提高。

社交网络平台的快速发展，得益于庞大的用户群体。作为网络社会化交往的主要工具，每种社交网络平台都积累了海量的用户数据集[1−2]。如果能对这些数据信息进行合理地提取、整理和分析，将会对用户群体特征和用户之间交往行为的分析起到很大的作用。对用户交互行为和群体特征的研究能够较大程度上方便网络运营商全面掌握用户的特征需求从而提供更好的服务，同样也能有力地帮助有关部门对网络舆情进行合理的控制和干预。

电子邮件是一种常见的社交网络平台，目前已经成为了一种基本的生活通讯工具。随着电子邮件业务的普及，各大互联网公司纷纷提供电子邮件业务，在国内有网易提供的 163、126 邮箱，腾讯公司提供的 qq 邮箱，新浪公司提供的 sina 邮箱等等；在国外有 google 公司提供的 gmail 邮箱，雅虎公司提供的 Yahoo 邮箱等等。用户利用这些邮箱在日常生活和工作中传递信息，进行人际交往，电子邮件和其他的社交网络平台一样，包含着大量对分析用户群体和用户交往行为特征有用的数据信息。对这些数据进行分析同样有着比较大的作用和意义。

然而，电子邮件的使用人群相当广泛，产生的数据量也相当庞大。根据网易 1 月份发布的《2015 年度互联网邮箱使用报告》的统计，仅针对所有的 163 邮箱，日平均接收邮件的数量就达到了 7.71 亿封。如此大的数据量仅依靠人工进行分析处理工作量太过巨大，需要消耗过多的人力资源，同时也无法保证分析的准确性[3-4]。因此，开发一个能自动化的对大量的电子邮件数据进行处理，能直观的展示数据分析的结果，能增加数据分析人员对电子邮件用户情况的感知度并且能方便数据分析人员进行进一步分析决策的电子邮件分析系统，势在必行。

1.2. 国内外研究现状

伴随着社交网络的高速发展，国内外对于社交网络数据分析挖掘的方法越来越重视，对其技术的研究越来越丰富和完善。最新研究成果显示，社交网络数据分析的主要研究方面有用户社区发现、用户影响力排名、用户行为分析等等。

（1） 用户影响力分析

用户在社交网络中的影响力被定义为在社交网络群体中，某个用户的行为引起他人行为变化的能力。早在 1998 年，Larry Page 和 Sergey Brin 等人就创建了网页重要程度 PageRank 计算方法，随后该方法又被广泛用于用户影响力的计算[5-6]。后来又有 Jianshu Weng 针对用户之间话题的相似程度的情况，提出了用户的影响力应该由用户在话题上的影响力总和决定。还有针对微博用户的影响力的研究认为，用户是否为认证用户，用户的粉丝数的多少，用户发布微博的总数量这些因素共同决定着用户的影响力[7]。

（2） 社区发现

社交网络中的用户会因为相同的兴趣爱好、共同的话题或者同样的工作单位等种种原因聚集成为一个社区，社区中的用户之间联系频繁，表现出一些相似的特征。对社区的发现和研究对社交网络的中用户之间交互特征的研究有着重要的意义。社区发现算法主要分为两种，重叠算法和非重叠算法[8]。重叠社区划分的经典算法有 CPM 算法、LMF 算法等；非重叠社区划分的经典算法有 Louvain 算法、Label propagation 算法等[9-10]。

（3） 用户行为分析

用户行为分析的意义在于通过对用户在社交网络上行为的分析，推测用户在现实生活中的人际交往情况，进而研究人类在社会中的各种不同行为。用户在网络上的行为种类有很多，不同的研究目的需要对不同的行为进行分析。一般性的用户行为有发送消息、接收消息、回复消息、评论消息、关注其他用户等等[11-12]。针对不同的用户行为，不同的分析目的，数据获取和处理的方式都有所不同。

1.3. 研究内容与方法

本论文提出了面向电子邮件的社交网络分析系统的实现方法，该系统对电子邮件数据进行深度挖掘分析，对分析的最终结果作出预测，并设计界面对分析结果进行展示。系统采用标准的层次化设计，将整个软件划分为视图层、数据分析

层和数据存储访问层。每层在系统中都有着明确分工，层次之间通过实现定义的各种接口来进行通信。层次内部又划分为多个模块实现不同的功能。此外，本系统采用 B/S 结构，客户无需安装客户端软件，便于软件的分发和维护升级，适应了众多应用客户端分散环境下的运行和维护要求。

研究方法如下：首先查找相关文献了解国内外社交网络平台分析系统以及分析技术的最新进展，大致了解基本研究方法；因为本文是应用型论文，所以在正式开始设计和开发工作之前需要学习相关实现技术，即 Spark 框架和 Scala 语言的相关技术，还有非关系型数据库 Mongodb 的操作方法。通过系统的需求分析和系统的详细设计，为后续的开发做好充分的前期准备工作。继而完成对系统的开发工作。最后对系统进行部署之后通过实验对系统进行验证。

1.4. 章节安排

第一章主要介绍本论文的研究背景、国内外的研究状况和研究的内容与方法。

第二章主要介绍本系统开发所需的相关技术，包括通用并行框架 Spark、图计算框架 Graphx 和函数式编程语言 Scala。

第三章主要讲解本系统的需求分析，包括系统的功能需求、系统的主要流程分析和系统的核心功能分析。

第四章主要介绍面向电子邮件的社交网络分析系统的总体设计，包括软件的结构设计、系统模式和开发技术的选择。

第五章主要介绍本系统的详细设计与实现，包括对数据分析层七个功能模块的详细设计和实现方法。

第六章介绍系统测试，包括功能性测试和性能测试。

第七章主要是论文的结论部分。

第二章 系统相关技术

面向电子邮件的社交网络分析系统对海量电子邮件数据进行深度分析挖掘，系统的开发需要选择合适的技术框架和编程语言。经过前期一系列的调查研究，最终选择 Spark 通用并行框架、GraphX 图计算框架和函数式编程语言 Scala 等技术和语言对系统进行开发工作。

2.1. 通用并行框架 Spark

Spark 是 UC Berkeley AMP lab 所开源的类 Hadoop MapReduce 的通用并行框架，Spark 拥有 Hadoop MapReduce 所具有的优点；但不同于 MapReduce 的是 Job 中间输出结果可以保存在内存中，从而不再需要读写 HDFS，因此 Spark 能更好地适用于数据挖掘与机器学习等需要迭代的 MapReduce 的算法。

基于电子邮件数据数据量庞大的特点，选择 Spark 这种与 Hadoop 相似的通用并行框架进行处理分析是比较好的选择。Spark 是一种开源集群计算环境，相比于其他一些并行数据处理框架，它在某些工作负载方面表现得更加优越。Spark 框架采用内存中的数据结构弹性分布式数据集(RDD)替换了磁盘进行中间处理数据的存储工作，减少了对磁盘的读取次数，因而大幅度地提高了数据处理的速度，并且还能支持对数据进行交互式的查询。除此之外，Spark 还有多语言支持能力，允许使用 Java、Scala 或是 Python 快速编写应用程序，对系统以后和其他不同语言实现的其他功能组件的对接提供便利。

Spark 是在 Scala 语言中实现的，它将 Scala 用作其应用程序框架。Spark 和 Scala 能够紧密集成，其中的 Scala 可以像操作本地集合对象一样轻松地操作分布式数据集[13]。

2.2. 图计算框架 Graphx

对由海量的电子邮件数据构建出的社交关系网络图的分析挖掘需要分布式图计算技术的支持，而 Spark GraphX 就是一个分布式图处理框架。Spark GraphX 基于 Spark 平台，所以天然就是一个分布式的图处理系统。它提供对图计算和图挖掘简洁易用的而丰富多彩的接口，极大的方便了大家对分布式图处理的需求。

图的分布式或者并行处理其实是把图拆分成很多的子图，然后分别对这些子图进行计算，计算的时候可以分别迭代进行分阶段的计算，即对图进行并行计

算。这样可以节约整体计算时间，提高计算效率[14-16]。

2.3. 函数式编程语言 Scala

Spark 是在 Scala 语言中实现的，它将 Scala 用作其应用程序框架，与 Scala 语言能够紧密集成。因此为了能和 Spark 框架中提供的开源内部源代码进行更好的对接，本系统采用 Scala 语言进行开发。

Scala 是一门多范式的编程语言，一种类似 java 的编程语言，设计初衷是实现可伸缩的语言，并集成命令式编程、面向对象编程和函数式编程的各种特性。

Scala 具有的面向对象的本质。与只支持单继承的语言相比，Scala 具有更广泛意义上的类重用。Scala 允许定义新类的时候重用“一个类中新增的成员定义（即相较于其父类的差异之处）”，Scala 称之为 mixin 类组合。除此之外，Scala 还包含了若干函数式语言的关键概念，包括高阶函数（Higher-Order Function）、局部套用（Currying）、嵌套函数（Nested Function）、序列解读（Sequence Comprehensions）等等。

Scala 的开发平台很多，并且各类不同的开发平台都具有各自的特点和优势，而其中值得推荐的是 Ecplise 的开发平台。首先该开发平台是一套开发源代码的平台，通常我们定义他为开源系统，并不涉及相关版权等等。并且其平台本身是一套框架和一系列服务，它是一套开放式的框架，相关的其他功能组件通过标准化的插件模式进行开放式嵌入。当然它本身就包含了大量插件集，其中就有 Scala 开发插件 Scala IDE。

2.4. 本章小结

本章介绍了开发面向电子邮件的社交网络分析系统所需要的主要技术，包括通用并行计算框架 Spark、图计算框架 GraphX 和函数式编程语言 Scala 等。

第三章 系统需求分析

随着互联网的发展，人与人之间的联系越来越多地通过网络进行。人与人的社交关系越来越多地借助网络软件来实现，社交网络平台在人们生活中占据着越来越重要的作用，因此对其产生的大量社交关系方面的信息的研究有着很大价值。

电子邮件作为一种常见的社交网络平台，被大量的各种类型、层次的人群广泛的应用于传递信息，因此其产生的数据具有复杂性、分散性、海量性的特点[17]，不通过统一化地处理和分析，很难从中获得关于使用人的有效信息[18]。为了能够高效快速地从海量电子邮件数据中获取用户需要的信息，并且直观地展示这些与电子邮件以及电子邮件使用人相关的信息，一个面向电子邮件的社交网络分析系统的开发十分必要。

3.1. 功能需求分析

一个面向电子邮件的社交网络分析系统应该具有以下功能:

（1）为用户提供电子邮件数据清晰直观的网络图展示，方便用户进行观察分析。

（2）采用自动方式取代人工方式对数据进行深度挖掘分析。

（3）支持对图中的数据多种不同维度的分析，为用户提供多种角度对数据的理解。

（4）针对不同类型的分析结果进行不同形式的展示，作为分析电子邮件数据的参考，方便用户决策。

（5）降低对使用者的专业技能的要求，提供清晰直观明确的功能按钮，提升工作效率。

（6）实现对电子邮件数据、统计分析结果数据等数据的存储、索引和管理。

3.2. 系统的主要流程分析

整体系统按照数据库配置、关系网络构建、用户功能指定、数据分析、分析结果存储、分析数据展示等多个环节顺序执行。

系统的主要流程如下图 2-1 所示。

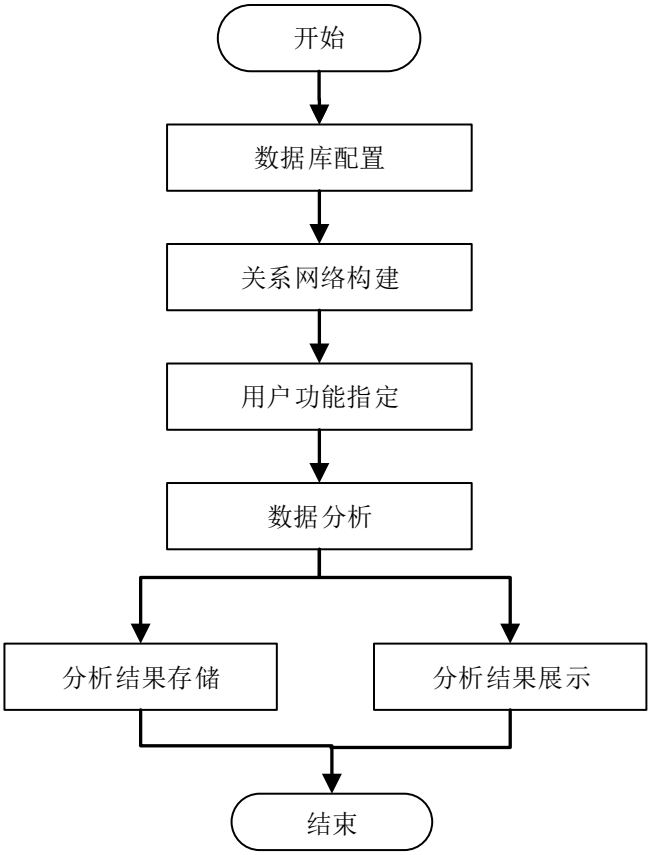

图 2-1 系统流程图

3.3.系统的核心功能分析

3.3.1. 友好的 UI 交互和清晰的可视化展示

基于减小客户端电脑载荷，减轻系统维护与升级的成本和工作量的目的，本系统采取 B/S 架构实现，用户通过客户端浏览器访问系统 Web 页面。系统向浏览器输出的 JSP 页面，由 HTML、图片等资源文件和 JavaScript 脚本组成，向用户提供原始数据或者分析后得出的数据构成的图的可视化展现，并向用户提供功能按钮。

3.3.2. 数据分析功能

面向电子邮件的社交网络分析系统主要对电子邮件中存在的信息进行有目的性地处理并对处理结果进行可视化展示，为用户的决策提供分析后的数据支持。使用本系统，用户希望通过构建网络、综合多角度数据分析等手段，建立并合理的展示电子邮件使用人之间的关系。

系统提供给用户的功能点用例如图 2-2 所示，包括数据库配置、关系网络构建、社区划分、活跃人发现、联系人最短路径发现、按条件筛选、直接间接联系人网络发现等。

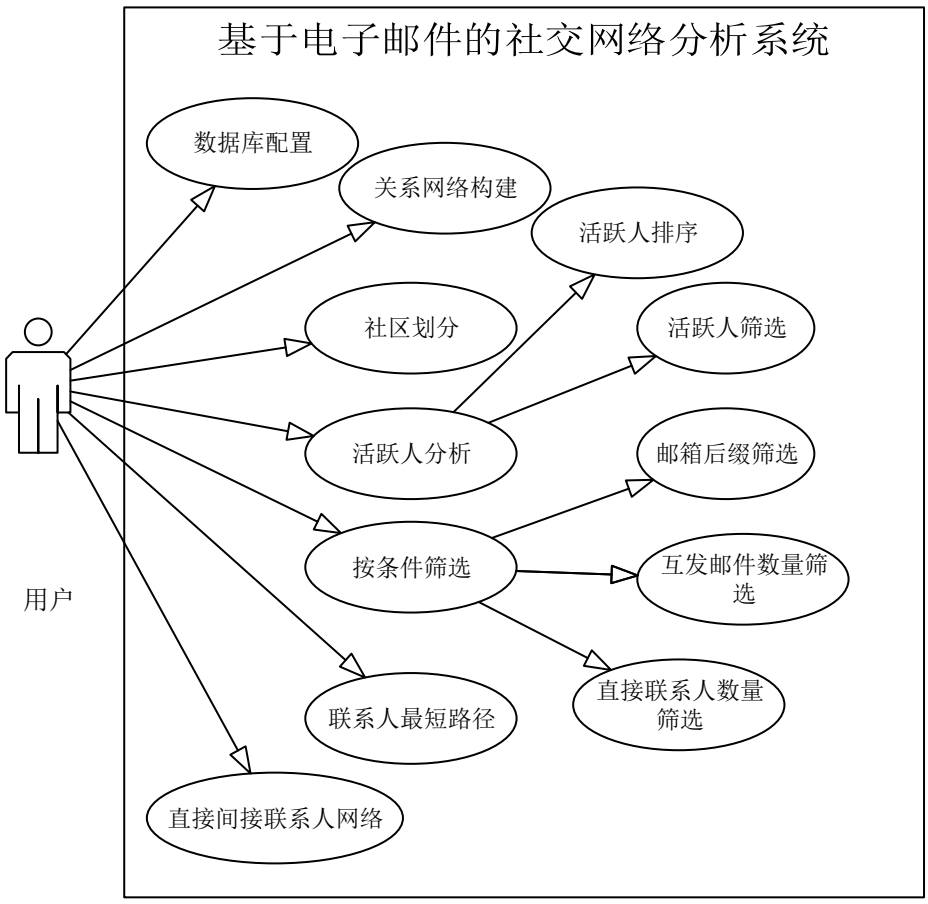

图2-2 用户用例图

3.3.2.1. 数据库配置用例

数据库配置通过提供分析数据所在数据库的连接信息，包括地址、用户名、密码等等，为整个面向电子邮件的社交网络分析系统后续分析提供数据来源。其具体说明如表 2-1。

表 2-1 数据库配置用例说明

用例名：	数据库配置
描述	本用例描述数据库的配置过程，通过输入数据库连接所需要的相关信息，与数据库连接，为系统后续的分析提供数据
前置条件	数据库服务器已启动

后置条件	系统和指定的数据库连接成功
触发条件	用户点击数据库配置按钮
基本流程	（1）用户点击数据库配置按钮 （2）数据库配置页面弹出，页面上显示用户需要填写的信息：数据库地址、数据库名称、用户名、密码 （3）用户输入信息，点击“确定”按钮 （4）如果数据库连接不成功，则会显示“数据库连接不成功”提示；如果数据库连接成功，则系统回到主界面
结束	指定数据库连接成功
注意事项	（1）用户填写的数据库地址必须为服务器上经过处理的电子邮件数据所存储的数据库位置 （2）数据库连接用户有访问等权限

3.3.2.2. 关系网络构建用例

关系网络构建根据数据库中储存的电子邮件和邮箱数据，对数据进行一定程度的处理，构建一个合理的数据关系网络，为后续分析提供具体的分析对象。其具体说明如表 2-2。

表 2-2 关系网络构建用例说明

用例名：	关系网络构建
描述	本用例获取电子邮件数据，构建成一个以电子邮箱为点，以互发邮件关系为边的数据关系网络，为后续分析操作提供具体的对象。同时将构建的网络展示在 UI 界面，让用户对数据和数据之间的关系有更直观的认识。
前置条件	系统和指定数据库连接成功
后置条件	关系网络构建完成
触发条件	用户点击数据导入按钮
基本流程	（1）用户点击数据库导入按钮 （2）调用图构建算法，算法获取指定数据库中的数据，构建一个以电子邮箱为点，以互发邮件关系为边的关系网络 （3）将构建的结果以图的点和边的形式展示在 UI 界面上
结束	关系网络构建成功，构建的网络显示在界面上
注意事项	（1）系统和数据库必须保持连接状态

3.3.2.3. 社区划分用例

社区划分用例根据收发邮件的海量的电子邮箱之间联系紧密的程度，把邮箱

的使用人分割成多个联系紧密的小团体。社区划分结果为后续的活跃人等分析提供更精确的数据源，有利于提高后续分析结果的准确性。其具体说明如表 2-3。

表 2-3 社区划分用例说明

用例名：	社区划分
描述	本用例根据收发邮件的邮箱之间联系紧密的程度，把邮箱的使用者分割成多个联系紧密的小团体。将划分的情况展示在 UI 界面，让用户对结果有直观的认识。
前置条件	关系网络构建完成
后置条件	社区划分完成
触发条件	用户点击社区划分按钮
基本流程	（1）用户点击社区划分按钮 （2）调用社区划分算法，算法对关系网络进行进行社区划分 （3）将划分的结果以图的点和边的形式展示在 UI 界面上，不同社区之间不连通，并弹出窗口显示包含所有社区的列表。 （4）将社区划分的结果存储到数据库中
结束	社区划分完成，社区划分的结果显示在界面上，社区存入数据库中
注意事项	（1）每个社区包含的所有的点和边的集合、社区的名称等信息将被显示在社区列表中，其中社区 ID 由系统随机生成

3.3.2.4. 活跃人分析用例

活跃人分析用例针对用户选择的一个社区，对社区中的邮箱使用人的活跃程度进行排序，并筛选出最活跃的标识出来。活跃的邮箱使用人意味着其和周围的对象联系的次数频繁，并且如果和其有联系的邮箱越活越，其本身的活跃程度也越高。具体说明如表 2-4。

表 2-4 活跃人分析用例说明

用例名：	活跃人分析
描述	本用例针对用户选择的一个社区，对社区中的邮箱使用人的活跃程度进行排序，显示活跃程度由高到低排序的列表，并筛选出最活跃的 10 个邮箱标识在 UI 界面的图上
前置条件	社区划分完成
后置条件	活跃人分析完成
触发条件	用户点击活跃人分析按钮
基本流程	（1）用户点击活跃人分析按钮 （2）系统弹出页面，页面上显示要求用户输入的信息：需要进行活跃人

	分析的社区的 ID （3）调用活跃人排序算法，算法在指定的社区内对其中的邮箱使用人的活跃程度进行排序 （4）调用活跃人筛选算法，从所有邮箱中找出最活跃的 10 个，在 UI 界面的图上标识对应的点出来 （5）弹出窗口显示活跃程度由高到低排序的列表 （6）将社区对应的活跃程度列表存入数据库中
结束	活跃人分析完成，社区对应的活跃程度列表存入数据库
注意事项	无

3.3.2.5. 联系人最短路径发现用例

联系人最短路径发现用例找出用户指定的两个邮箱使用人之间所有的最短联系路径，并将这些最短路径展示在 UI 界面上。两个邮箱之间最短联系路径越短，说明使用人之间的关系可能越紧密。具体说明如表 2-5。

表 2-5 联系人最短路径发现用例说明

用例名：	联系人最短路径发现
描述	本用例找出用户指定的两个邮箱使用人之间所有的最短联系路径，并将这些最短路径构成一张新关系图展示在 UI 界面上，使用户直观地观察到两个邮箱使用人之间的联系路径
前置条件	关系网络构建完成
后置条件	联系人最短路径找出
触发条件	用户点击最短路径按钮
基本流程	（1）用户点击最短路径按钮 （2）系统弹出页面，页面上显示要求用户输入的信息：需要找到最短路径的两个邮箱 （3）调用最短路径发现算法，找出所有的最短路径 （4）构建新的关系图，新的关系图只包含最短路径经过的点和组成最短路径的边，把新的关系图显示在界面上 （7）弹出窗口显示最短路径的数值 （8）将两点之间的所有最短路径存入数据库中
结束	联系人最短路径已找出，所有最短路径存入数据库中
注意事项	（1）最短路径可能不止一条，找出的是所有的

3.3.2.6. 直接/间接联系人网络发现用例

直接/间接联系人网络发现用例找出和用户指定的邮箱使用人有直接或者间接联系的邮箱，并将这些邮箱构成的关系子图展示在 UI 界面上。所有使用这个子图中邮箱的人都可能会和用户指定的邮箱使用人有直接或间接的联系。具体说明如表 2-6。

表 2-6 直接/间接联系人网络发现用例说明

用例名：	直接/间接联系人网络发现
描述	本用例找出和用户指定的邮箱有直接间接联系的邮箱构成关系子图展示在 UI 界面上，使用户直观地观察有哪些邮箱使用人可能和指定的邮箱人有直接或者间接的联系
前置条件	关系网络构建完成
后置条件	直接/间接联系人网络构建完成
触发条件	用户点击联系人网络按钮
基本流程	（1）用户点击联系人网络按钮 （2）系统弹出页面，页面上显示要求用户输入的信息：需要找到直接间接联系人网络的邮箱 （3）调用求一个点所在的连通子图的算法，构建出一个新的关系图，这个新的关系图中除了用户输入的邮箱代表的点，其他点都和这个点有连接路径，不论是直接相连还是间接相连。这个关系图就是这个点的直接间接联系网络 （4）将（3）中构建出来的新的关系图展示在 UI 界面上 （5）将直接/间接联系人网络中的点都存入数据库中
结束	直接/间接联系人网络已构建完成，所有的相关的点都存入数据中
注意事项	无

3.3.2.7. 条件筛选用例

条件筛选用例根据用户指定的条件筛选邮件和邮箱使用人，并且根据筛选后的数据构建关系图展示在 UI 界面上。筛选条件包括邮箱后缀、互发邮件数量和直接联系人数量。具体说明如表 2-7。

表 2-7 条件筛选用例说明

用例名：	条件筛选
描述	本用例根据用户指定的条件筛选邮件和邮箱使用人，根据筛选后的数据构建图展示在界面上

前置条件	关系网络构建完成
后置条件	按条件筛选的关系图构建完成，或者列表结果展示完成
触发条件	用户点击条件筛选按钮
基本流程	（1）用户点击条件筛选按钮 （2）系统弹出页面，页面上要求用户选择筛选条件种类：邮箱后缀筛选，互发邮件数量筛选，直接联系人数量筛选 （3）若用户选择邮箱后缀筛选，则弹出邮箱后缀列表页面，用户可以点击选择其中任意一个邮箱后缀，系统调用邮箱后缀筛选函数，得到一个所有指定邮箱后缀的邮箱构成的关系图 （4）若用户选择互发邮件数量筛选，则弹出页面要求用户输入两个直接联系邮箱之间互发邮件数量的下限，系统调用邮件数量筛选函数，得到一个由所有互发邮件数量大于下限的邮箱作为点，连接互发邮件数量大于下限的两个邮箱的连接线作为边的关系图 （5）若用户选择直接联系人数量筛选，则弹出页面要求用户输入邮箱直接联系的邮箱的个数的下限，系统调用联系人数量筛选函数，得到一个所有直接联系邮箱的个数大于下限的邮箱列表 （6）将（3）（4）构建出来的新的图展示在 UI 界面上，或者将（5）得到的列表弹出页面展示
结束	按条件筛选的关系图构建完成，或者列表结果展示完成
注意事项	无

3.3.3. 数据存储

基于电子数据的海量性的特征，同时考虑到提高数据处理速度和效率的目的，本系统中采用适用于处理大数据量、高并发、弱事务的应用的非关系型数据库 HBase、MongoDB 来储存数据，而且需要根据不同的业务添加对应的业务数据库，储存功能计算后的结果。

3.4. 本章小结

本章主要是从系统的功能需求、系统的主要流程分析和系统的核心功能分析三个方面对系统进行需求分析。其中核心功能分析又从可视化展示、数据分析功能和数据存储这三个方面详细展开。

第四章 系统总体设计

面向电子邮件的社交网络分析系统采用 MyEclipse 作为承载 UI 界面的 JSP 网页的设计开发工具，并且使用 Scala 语言，基于 Spark 这个开源集群计算环境[19]，以 Eclipse 作为开发平台构建类和函数，形成后台计算环境，最终形成 B/S 模式的面向电子邮件的社交网络分析系统。同时采用 MongoDB 这样的分布式开源 NoSQL 数据库作为数据存储工具以适应庞大的数据量和分布式的计算环境特点。本章就将结合这些具体技术的应用，详细地阐述系统的总体设计和各个主要功能的实现方法。

4.1. 系统的总体设计

按照标准的层次化设计[20-26]，面向电子邮件的社交网络分析系统主要按照以下几个模块划分。

（1）视图层：视图层是整个的系统架构的最外层，主要作用是提供和用户进行交互的界面。用户点击界面提供的功能按钮，并且按要求输入数据，数据会被向下传递到数据分析层中对应的功能模块。数据分析层在进行完计算后将结果返回给视图层，视图层对结果进行处理，进行可视化展示。视图层的实现主要依靠 JSP 页面的开发，Eclipse 结合 GraphStream 等图生成插件为 JSP 的开发提供了很好的环境。

（2）数据分析层：数据分析层是整个系统的核心部分，主要作用是实现各个功能用例的业务逻辑和算法，并将业务过程和算法封装，为视图层传递数据时提供接口。数据分析层利用 Spark 这个开源集群计算环境中的 Graphx 图计算框架，针对电子邮件数据特性构建相应的网络，对网络进行多维度的功能性分析。根据功能性需求，数据分析层主要分为七个模块：数据库配置、关系网络构建、社区划分、活跃人分析、联系人最短路径发现、直接/间接联系人发现、条件筛选。

（3）数据存储访问层：数据存储访问层分为两层，数据访问层和数据库层。其中数据访问层采取 DAO 设计模式，为数据分析层提供连接、查询，插入、更新等数据库基本操作接口，数据分析层通过这些指定的接口获取数据。考虑到数据量庞大和内存有限等因素，基于提高数据读取处理效率的目的，在对数据库操作上要求分段、并行并且要保证数据的同步性。数据库层储存各种不同种类用途的数据，总体上分为两类，原始电子邮件数据库和电子邮件业务数据库。

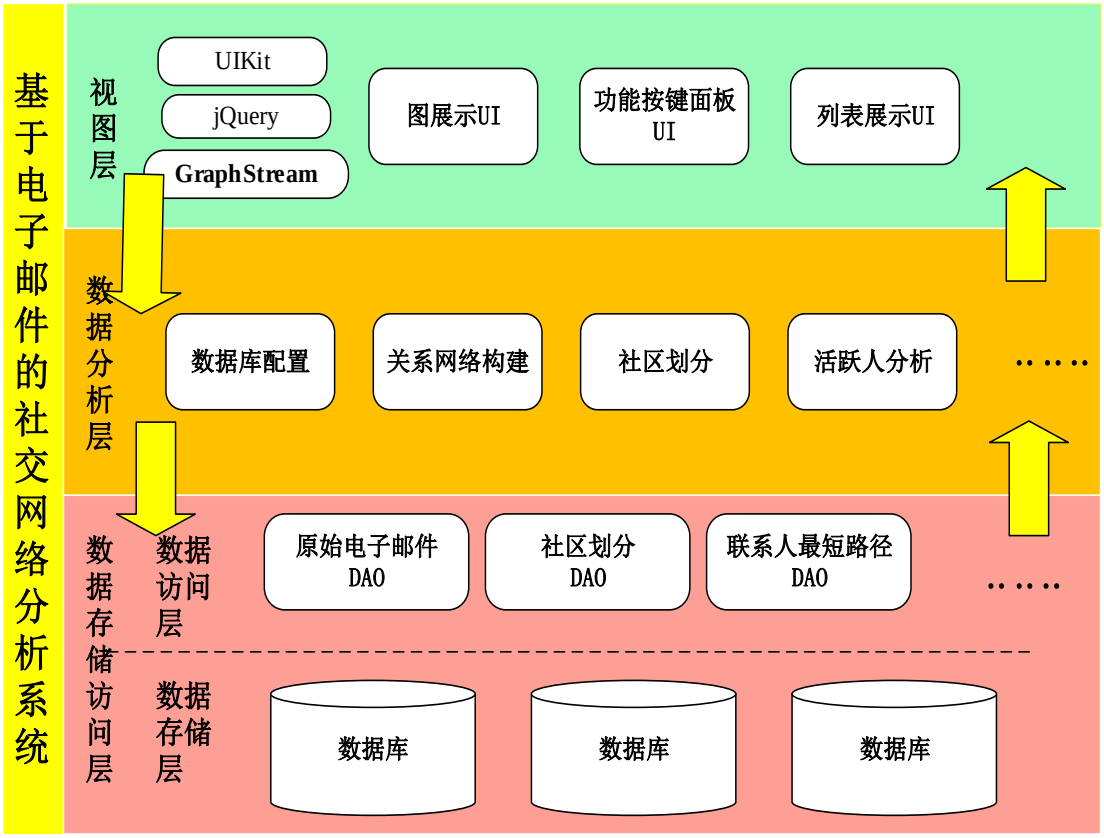

图 3-1 系统架构图

4.2. 软件结构设计

系统的软件结构设计按照标准的视图层、数据分析层和数据存储访问层进行设计。

4.2.1. 视图层

根据第二章的需求分析，视图层主要包含以下主要模块：图展示、功能按键、列表展示。

（1） 图展示 UI：将数据分析层功能模块得到的分析结果数据构建成点和边组成的图展示在界面上，同时为用户提供对图的拖动，放大，旋转等功能，以便用户直观清晰的观察分析结果。

（2） 功能按键面板 UI：功能按键面板上包含各种可以触发对应的数据分析层功能模块进行电子邮件数据分析的功能按钮。用户通过点击这些按钮得到需要的分析结果。功能按键面板中的功能按钮主要有数据库配置按钮，数据导入按钮，社区划分按钮，活跃人分析按钮，最短路径按钮，联系人网络按钮，条件筛选按钮等。

（3） 列表展示 UI：某些数据分析层功能模块得到的分析结果数据只能以列表等形式存在，比如活跃人分析模块得到的活跃程度排序。这些结果不能以图的形式呈现，则单独设计列表展示模块，用于展示这些以列表形式存在的结果数据。

4.2.2. 数据分析层

数据分析层由七个不同的的功能模块组成，这些功能模块根据功能性的需求制定。功能模块图如图 3-2 所示。

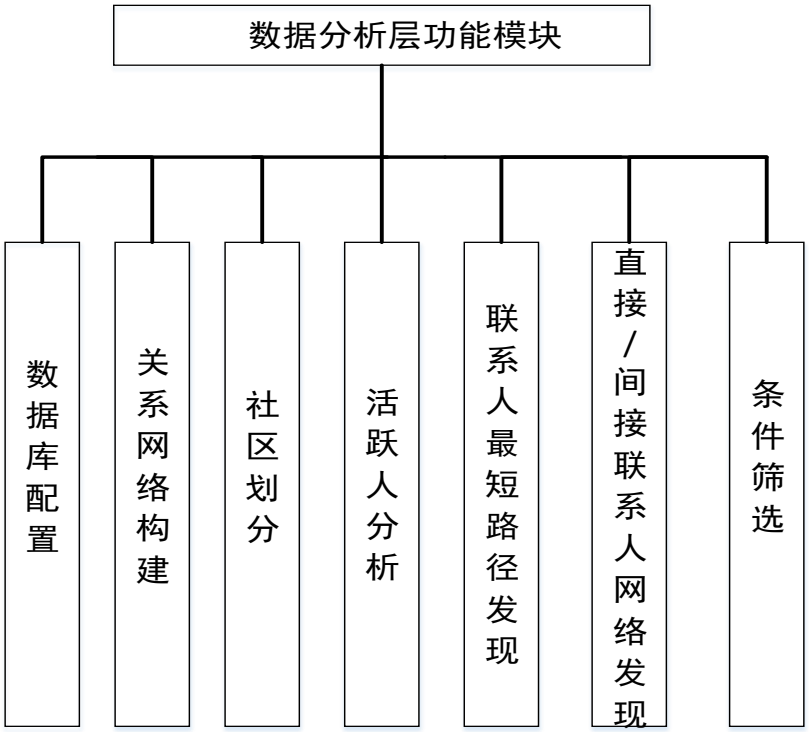

图 3-2 数据分析层功能模块图

下面对数据分析层的各个功能性模块进行详细分析介绍：

（1） 数据库配置：数据库配置模块主要处理与不同的数据库的连接事务。该模块的功能由设计的类 DataBase_Connection 实现，这个类完成和数据库的连接工作并储存数据库的配置信息。

（2） 关系网络构建模块；关系网络构建模块主要处理数据导入和网络图构建的工作。在构建的网络图中，以电子邮箱代表使用人作为点，邮箱之间发送邮件产生关系，邮箱使用人之间的关系由连接点的边表示。

（3） 社区划分模块：社区划分模块主要任务是在构建的关系网络图的基础上对点代表的用户根据联系紧密的程度进行划分。将联系紧密的用户划分到一个社区，联系不紧密的用户划分到不同社区，不同社区之间没有重叠，主

要设计 Community_Partition 类用于处理社区划分的相关问题.

（4） 活跃人分析模块：活跃人分析模块主要任务是在划分的某个社区中找出其中使用人的活跃程度的排序和最活跃的使用人。使用人的活跃程度由两方面决定，一方面，邮箱使用人和周围的对象联系的次数越频繁，其越活跃；另一方面，和当前邮箱使用人有联系的邮箱越活跃，其本身的活跃程度也越高。活跃人分析模块的功能主要由设计的类 Activeness_Finding 实现。

（5） 联系人最短路径发现模块：联系人最短路径发现模块主要任务是找出两个邮箱使用人之间联系的所有最短路程。从其中一个代表邮箱的点到达另一个代表邮箱的点在图上经过的边的个数即是路径的长度。联系人最短路径模块的功能主要由设计的类 Path_Finding 实现。

（6） 直接/间接联系人网络发现模块：直接/间接联系人网络发现模块主要任务是找出和某个邮箱使用人所有有直接或间接联系的邮箱构成的网络。有直接联系指代表邮箱的两个点有边相连，有间接联系之代表邮箱的两个点之间没有直接的边相连，但是有经过其他邮箱的连接路径。联系人最短路径模块的功能主要由设计的类 ConnectionWeb_Finding 中的函数实现。

（7） 条件筛选模块：条件筛选模块的主要任务是根据用户指定的电子邮件和电子邮箱使用人的筛选条件在网络图中筛选出符合条件的点和边构建新的网络图。条件筛选模块的功能主要由设计的类 Condition_Filtering 中的函数实现。

4.2.3. 数据存储访问层

数据存储访问层由数据访问层和数据库层两部分组成。

4.2.3.1. 数据访问层

数据访问层位于数据存储访问层的最外层，实现所有资源信息交互的数据控制。DAO 设计模式[27-29]用于数据访问层的实现，可以很大程度地提高代码的灵活性。在使用 DAO 之前，数据分析层直接访问数据库中的持久化数据。当持久化数据发生改变时，数据分析层也需要跟着发生改变。DAO 的应用解决了此类问题。当底层数据发生改变时，只需要更改 DAO 即可，数据分析层的改动达到最小。数据访问层的具体情况参见图 3-3 展示的逻辑示意图。

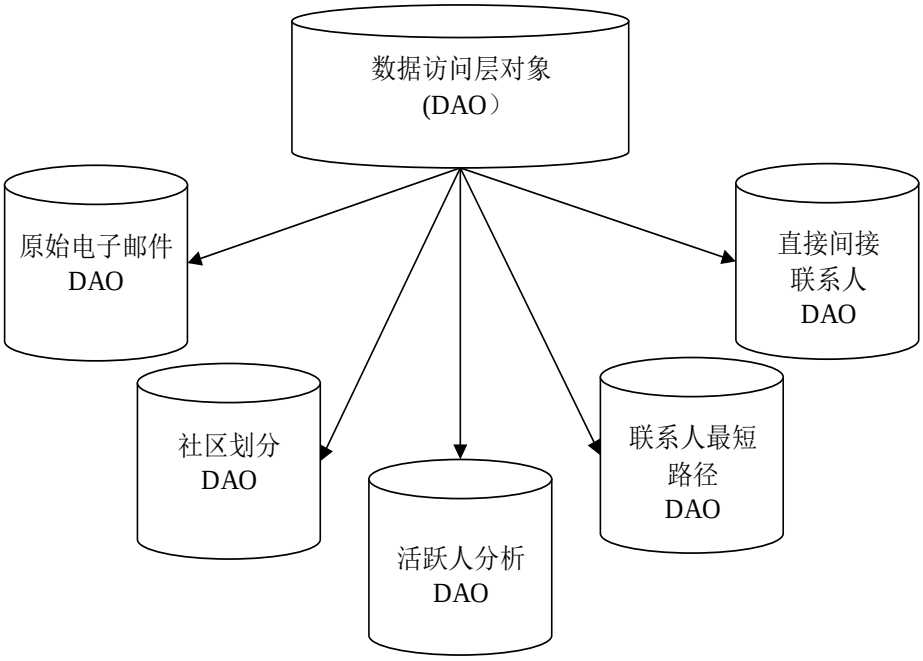

图 3-3 数据访问层逻辑示意图

系统所要用到的 DAO 类有原始电子邮件 DAO、社区划分 DAO、活跃人分析 DAO、联系人最短路径 DAO、直接间接联系人 DAO。这些 DAO 分别实现了各自的接口，用于对数据进行访问。

4.2.3.2. 数据库层

数据库层存储面向电子邮件的社交网络分析系统中数据分析所需的所有数据，主要包括两大类的数据库：原始电子邮件数据库和电子邮件业务数据库。原始电子邮件数据库主要储存所有的用于分析的电子邮件数据。电子邮件业务数据库主要储存各个功能模块分析得到的结果数据以便于进一步的分析和以后的查询工作。

下文将对每个数据库中的数据表进行详细描述。

（1）原始电子邮件数据库

原始电子邮件数据库只包含 Email_Record 数据表，如表 3-1 所示。

表 3-1 Email_Record 表

字段名	类型	备注	说明
Email_ID	Long	邮件 ID	主键
From_Address	String	发件人邮箱地址	

From_Name	String	发件人昵称
To_Address	String	收件人邮箱地址
To_Name	String	收件人昵称
CC_Address	String	抄送人邮箱地址
CC_Name	String	抄送人昵称
Time	Timestamp	时间
Content	String	内容

（2）电子邮件业务数据库

基于必须将系统中某些功能模块得出的结果存储到数据库中的需求，设计所有需要储存结果的功能点都在电子邮件业务数据库中有对应的数据表，则业务数据库中的数据表有储存社区划分结果的 Email_Community 表，如表 3-2 中所描述的。储存活跃人分析结果的 Email_Activeness 表，如表 3-3 中所描述的。储存联系人最短路径的 Email_ShortestPath 表，如表 3-4 中所描述的。储存直接间接联系人网络的 Email_ConnectionWeb 表，如表 3-5 中所描述的。

表 3-2 Email_Community 表

字段名	类型	备注	说明
Community_ID	Long	社区 ID	
From_Address	String	发件人地址	
To_Address	String	收件人地址	
Time	Arrays[Timestamp]	时间	所有邮件发送时间
Content	Arrays[String]	内容	所有邮件发送内容
Email_Number	Int	邮件数量	

表 3-3 Email_Activeness 表

字段名	类型	备注	说明
Community_ID	Long	社区 ID	
Address	String	邮箱地址	
Name	String	使用人昵称	
Activeness_Grade	Double	活跃程度评分	0.0-100.0 之间

表 3-4 Email_ShortestPath 表

字段名	类型	备注	说明
Src_Address	String	路径起点的邮箱地址	
Dst_Address	String	路径终点的邮箱地址	
Shortest_Path	Arrays[String]	路径	所有路径，String 分割为路径上多个邮箱地址
Path_length	Int	路径长度	

表 3-5 Email_ConnectionWeb 表

字段名	类型	备注	说明
Src_Address	String	联系网络的源点	主键
WebContents_Address	Arrays[String]	联系网络中所有的邮箱地址	
Address_Number	Int	联系网络中邮箱的个数	

4.3. 系统的开发环境

系统的开发环境如表 5-1 所示。

表 5-1 系统开发环境配置表

类别	标准配置	最低配置
计算机硬件	CPU:Intel(R) Xeon(R) E5-2630 v3 @2.40GHz	CPU:Intel(R) Xeon(R) E5-2630 v3 @2.40GHz
	内存：4GB	内存：4GB
	硬盘：500G	硬盘：200G
软件	操作系统：Red Hat 6.5	操作系统：Red Hat 6.5
	Eclipse 4.2.0	Eclipse 4.2.0
	Spark 1.6.0	Spark 1.6.0
	MongoDB 3.0.4	MongoDB 3.0.4

4.4. 本章小结

本章对系统的总体设计进行描述。系统采取标准的层次结构，分为视图层、数据分析层和数据存储访问层。层次化的设计能有效的控制层次之间的交互。并杜绝了层次之间不该有的交集，减少了错误的发生。每一层中又划分为不同功能的功能模块，并减弱模块之间耦合性，以便于系统未来的扩展和维护。

第五章 系统详细设计与实现

第四章对面向电子邮件的社交网络分析系统的设计进行了总体上的阐述。系统从架构上分为视图层、数据分析层和数据存储访问层，其中，数据分析层为系统的核心。本章将重点介绍数据分析层中各个功能模块详细的流程设计和算法实现。

5.1. 数据配置模块的设计与实现

数据库配置模块主要处理与不同的数据库的连接事务。下面详细讲解模块中类的设计和流程实现。

5.1.1. 类的设计

数据库配置模块的功能主要由设计的功能类 DataBase_Connection 中的成员变量实现，类图如 4-1 所示。

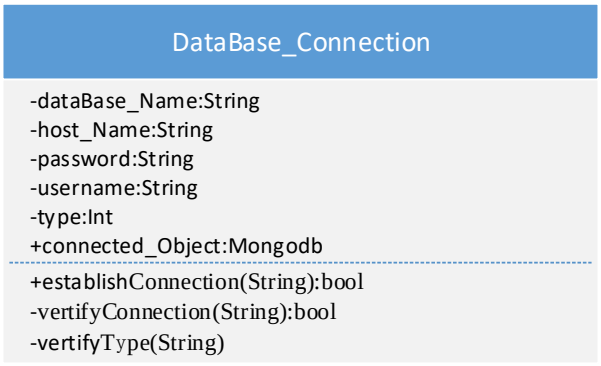

图 4-1 数据配置模块类图

下面是对类图中一些成员的详细解释：

（1） connected_Object：连接成功后的 MongoDB 的数据库连接对象。

（2） type：表示数据库的类型，有两个可能值：1 和 0。0 代表数据库的类型是原始电子邮件数据库，1 代表数据库的类型是电子邮件业务数据库。

（3） establishConnection 函数：解析视图层传递的用户输入的数据库配置信息，并调用 DataBase_Connection 类中的其他函数，对数据库进行连接或验证。

（4） createConnection 函数：建立 MongoDB 的数据库连接对象，如果成功，则输出 true，设置类变量 connectedObject；如果不成功，则输出 false。

（5） vertifyConnection 函数：检验用户名密码的正确性，如果用户名密码正确，则输出 true；如果用户名密码不正确，则输出 false。

（6） vertifyType 函数：检验数据库类型是原始电子邮件数据库还是电子邮件业务数据库，并改变类变量 type 为对应类型。

5.1.2. 功能流程实现

数据配置模块各个对象的调用过程如图 4-2 所示。

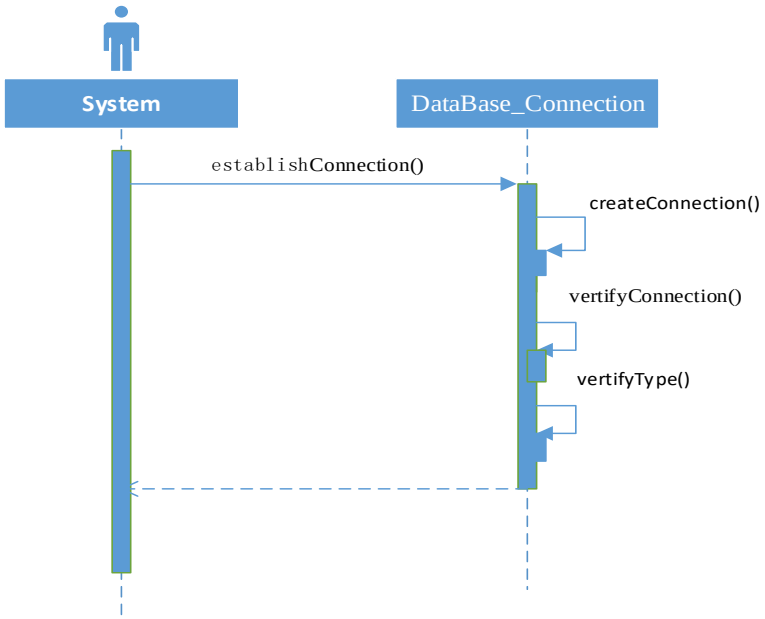

图 4-2 数据配置模块时序图

具体流程如下：

（1） 用户输入数据库配置信息，系统调用 DataBase_Connection 类中的 establishConnection()函数，函数解析输入的配置信息，并调用内部类 createConnection()建立 MongoDB 的数据库连接对象。

（2） 数据库连接对象建立成功后，establishConnection()函数调用内部类 vertifyConnection()检验用户名密码的正确性。

（3） establishConnection()函数调用内部类 vertifyType()检验数据库类型。

（4） 如果数据库连接成功，通知系统数据库连接成功，否则通知系统数据库连接失败。

5.2. 关系网络构建模块的设计与实现

关系网络构建模块主要负责从数据库中导入电子邮件的相关数据并且根据导入的数据构建出以邮箱为点，以邮件发送关系为边的无向图。下面详细讲解模块中类的设计和流程实现。

5.2.1. 类的设计

关系网络构建模块的功能主要由 DAO_InitialEmail 和 Web_Creation 等设计的类中的成员变量实现，类图如 4-3 所示。

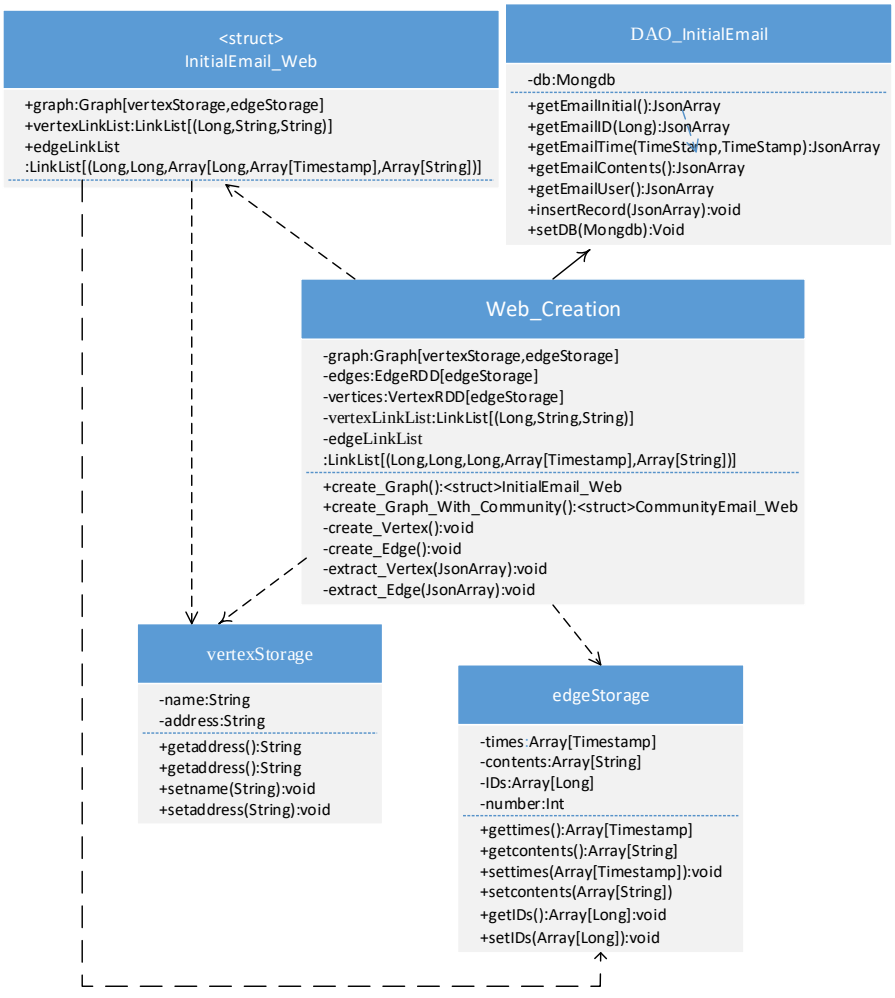

图 4-3 关系网络构建模块类图

下面是对类图的详细解释：

（1） Web_Creation 类：关系网络构建功能类，封装了信息提取处理，点构建，边构建等函数，负责构建电子邮件网络，并向视图层返回图的构建结果。

（2） InitialEmail_Web 结构体:存储处理后的原始邮件数据信息和网络构建的结

果。

（3） DAO_InitialEmail 类：原始电子邮件数据库操作类，封装了对原始电子邮件数据进行读取，存储的函数。

（4） vertexStorage 类：实体类，封装了网络图中的点储存的所有属性。

（5） edgeStorage 类：实体类，封装了网络图中的边储存的所有属性。

5.2.2. 功能流程实现

关系网络构建模块中各个对象的调用过程如图 4-4 所示。

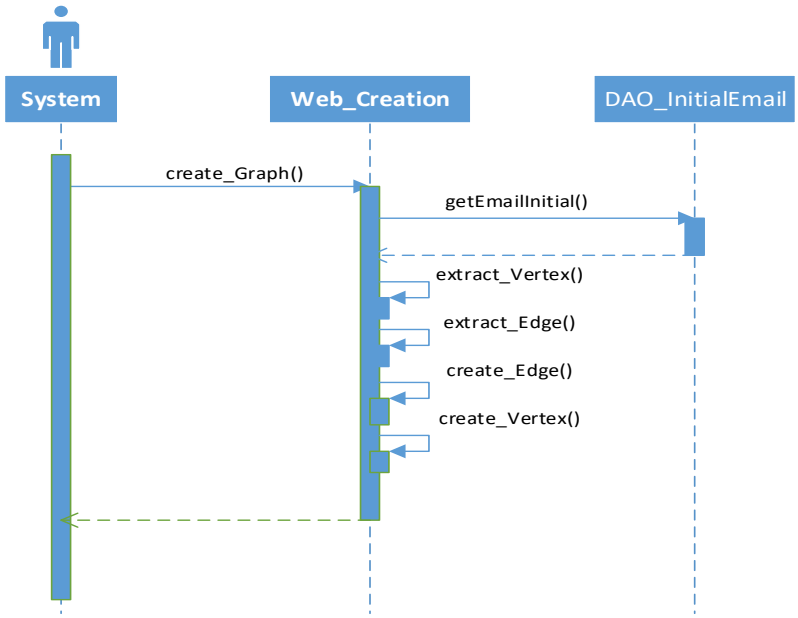

图 4-4 网络构建模块时序图

具体的程序流程如下：

（1） 用户点击数据导入按钮，系统调用 Web_Creation 类中的 create_Graph()函数。create_Graph()函数调用 DAO_InitialEmail 类中的函数 getEmailInitial()获取包含所有字段的原始电子邮件信息。

（2） 得到所有的原始邮件信息后，create_Graph()函数调用类内部函数 extract_Vertex()从原始邮件中提取出其中包含的所有与邮箱有关的信息，包括邮箱的地址和使用人的姓名等，并为每个邮箱指定一个 Long 类型的值作为网络图中这个邮箱对应的点的 VertexId。这个 VertexId 值为邮箱地址字符串的 md5 值。所有的提取出的邮箱信息组成一个新的链表 LinkList[(vertexId，Address，Name)]，存入结构体 InitialEmail_Web 的成

员变量 vertexLinkList。

（3） 调用类内部函数 extract_Edge()，把相同的两个邮箱之间所有互发的邮件都合并起来，生成一个新的链表 LinkList[(fromID，toID，mailID，Times，Contents)] ，存入结构体 InitialEmail_Web 的成员变量 edgeLinkList。其中 fromID 和 toID 为收发邮件的邮箱在网络图中对应的点的 VertexId。

（4） 邮件信息提取完成后，create_Graph()调用 create_Edge()和 create_Vertex()函数，分别构建图的边集 edges 和点集 vertices。

（5） create_Graph()利用构建好的点边集构建图并存入结构体 InitialEmail_Web 的成员变量 graph。最后，将类变量 edgeLinkList 和 vertexLinkList 的值返回给视图层。

5.3. 社区划分模块的设计与实现

社区划分模块主要任务是在构建的关系网络图的基础上对点代表的用户根据联系紧密的程度进行划分。下面详细讲解模块中类的设计和流程实现。

5.3.1. 类的设计

社区划分模块的功能主要由 Community_Partition 和 DAO_CommunityEmail 等设计的类中的成员变量实现，类图如 4-5 所示。

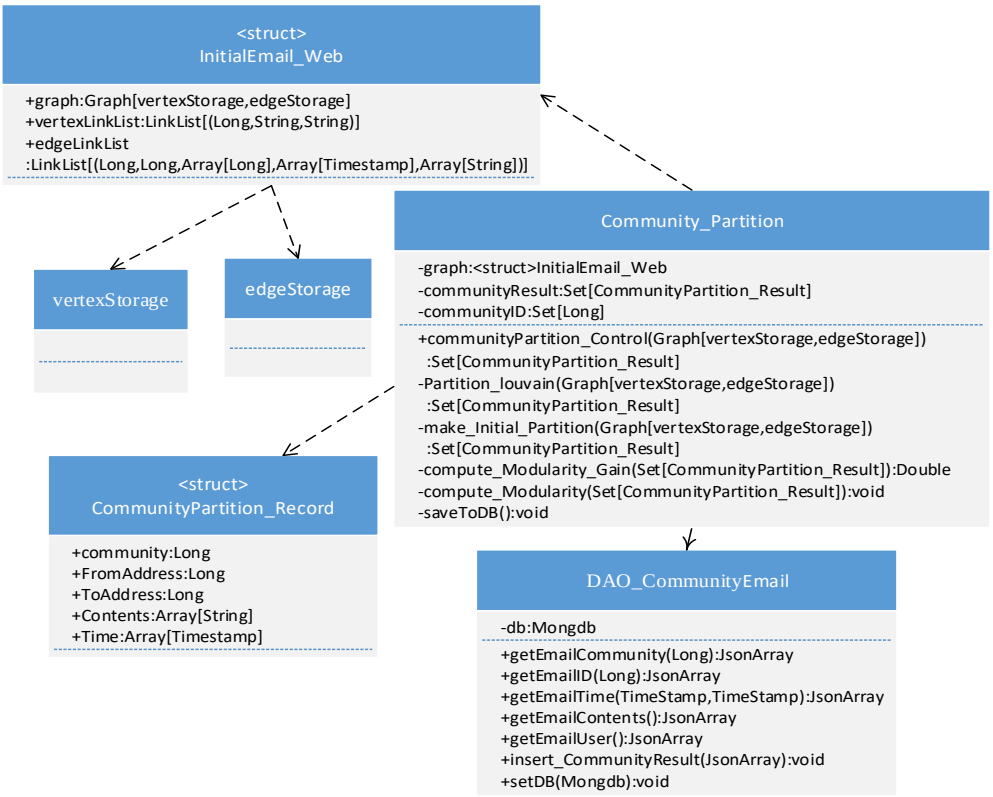

图 4-5 社区划分模块类图

下面是对类图的详细解释：

（1） Community_Partition 类：社区划分功能类，封装了社区划分控制、louvain 社区划分等函数，并向视图层返回社区划分的结果。

（2） CommunityPartition_Record 结构体：存储社区划分后的结果中的一条记录。

（3） DAO_CommunityEmail：业务数据库操作类，封装了对电子邮件业务数据库中 Email_Community 表里数据的存储、读取等操作函数。

（4） InitialEmail_Web 结构体:存储处理后的原始邮件数据信息和网络构建的结果。

（5） VertexStorage：实体类，封装了网络图中的点储存的所有属性。

（6） edgeStorage：实体类，封装了网络图中的边储存的所有属性。

5.3.2. 功能流程实现

社区划分模块中各个对象的调用过程如图 4-8 所示。

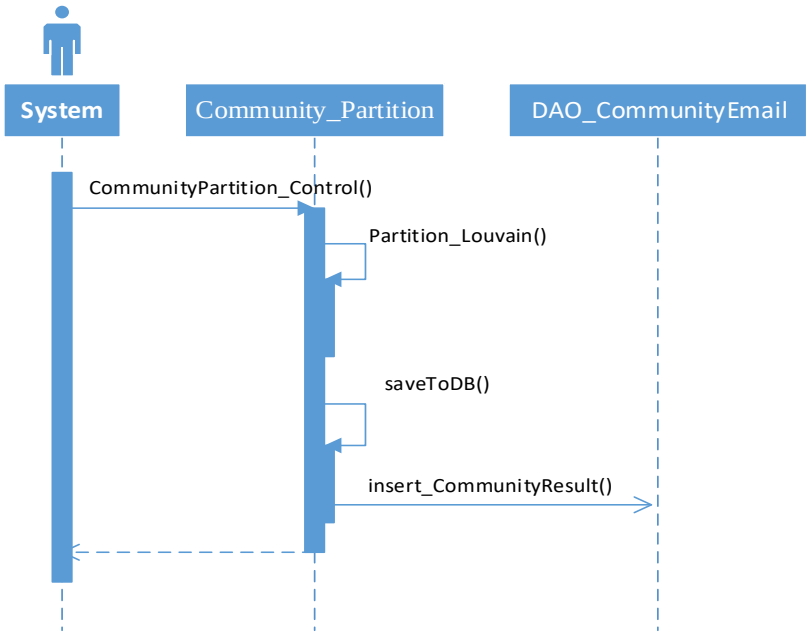

图 4-6 社区划分模块时序图

具体的程序流程如下：

（1） 系统完成关系网络构建模块后，用户点击社区划分按钮，系统调用 Community_Partition 类中的 CommunityPartition_Control()函数。函数 CommunityPartition_Control()调用类内部的函数 Partition_Louvain()。

（2） Partition_Louvain()函数应用 Louvain 算法对网络图进行社区划分。Louvain 算法通过不断改变社区的划分方式达到模块度值最大化。模块度值最大的划分方式被认为是社区划分的最终结果。模块度值是评估一个社区网络划分好坏的度量方法。它的物理含义是社区内节点的连边数与随机情况下的边数之差，它的取值范围是 [-1/2,1)。模块度值越接近 1，表示网络划分出的社区结构的强度越强，也就是划分质量越好。

（3） 网络图的社区划分完成后，CommunityPartition_Control()函数将储存在类变量 communityResult 中的社区划分的结果返回给视图层，并调用类内部函数 saveToDB()。saveToDB()函数把变量 communityResult 中代表收发邮件的邮箱的点的 vertexId 值替换为对应的邮箱的地址字符串，并调用 DAO_CommunityEmail 类中的 insert_CommunityResult()函数将找出的所有的社区存入 Email_Community 表里。

5.4. 活跃人分析模块的设计与实现

活跃人分析模块主要任务是在划分的某个社区中得到其中邮箱使用人的活跃程度的排序和找出最活跃的使用人。下面详细讲解模块中类的设计和流程实现。

5.4.1. 类的设计

活跃人分析模块的功能主要由 Activeness_Finding 和 DAO_CommunityEmail 等设计的类中的成员变量实现，类图如 4-7 所示。

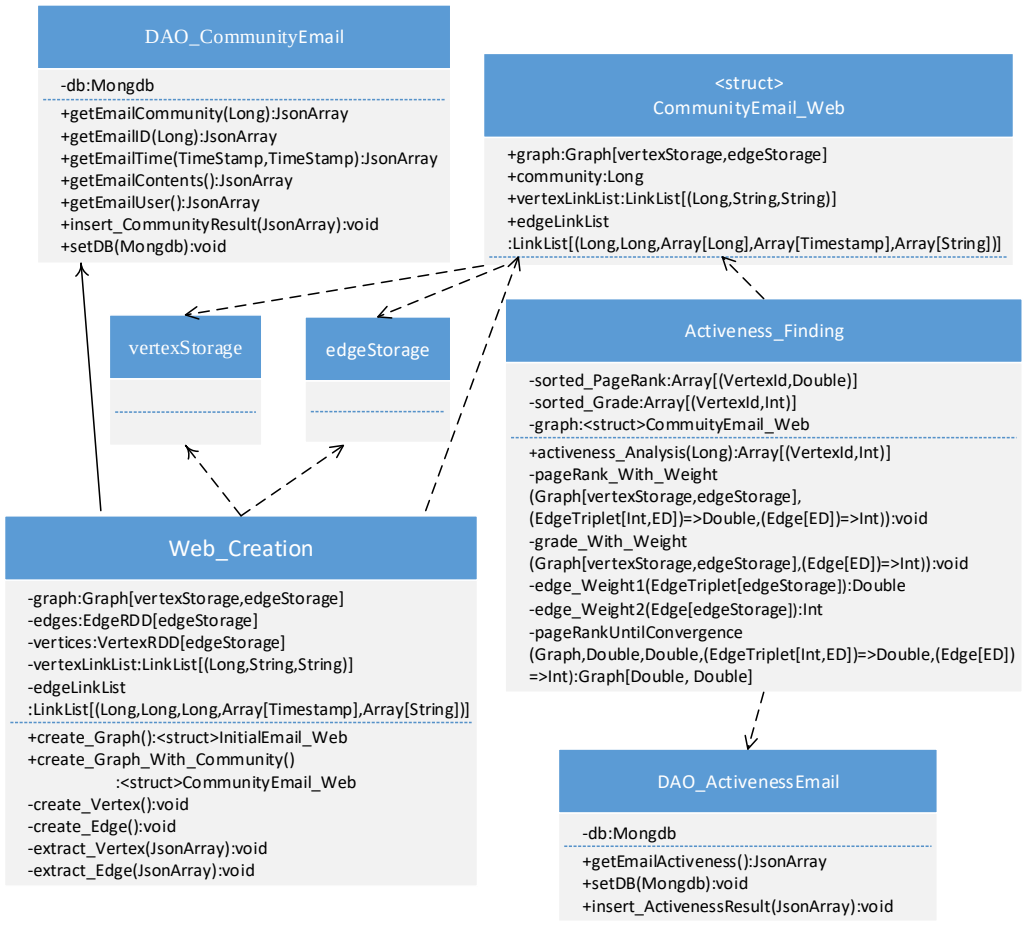

图 4-7 活跃人分析模块类图

下面是对类图的详细解释：

（1） DAO_CommunityEmail：业务数据库操作类，封装了对电子邮件业务数据库中 Email_Community 表里数据的存储、读取等操作函数。

（2） Web_Creation 类：关系网络构建功能类，封装了信息提取处理，点构建，边构建等函数，负责构建电子邮件网络，并向视图层返回图的构建结果。

（3） CommunityEmail_Web 结构体:存储单个社区中的电子邮件数据信息和社区网络构建的结果。

（4） vertexStorage：实体类，封装了网络图中的点储存的所有属性。

（5） edgeStorage：实体类，封装了网络图中的边储存的所有属性。

（6） Activeness_Finding：活跃人分析功能类，封装了 PageRank 计算函数、点总权重计算函数、边权重计算函数等函数，并且存储活跃人分析的结果。

（7） DAO_ActivenessEmail：业务数据库操作类，封装了对电子邮件业务数据库中 Email_Activeness 表里数据的存储、读取等操作函数。

5.4.2. 功能流程实现

活跃人分析模块中各个对象的调用过程如图 4-8 所示。

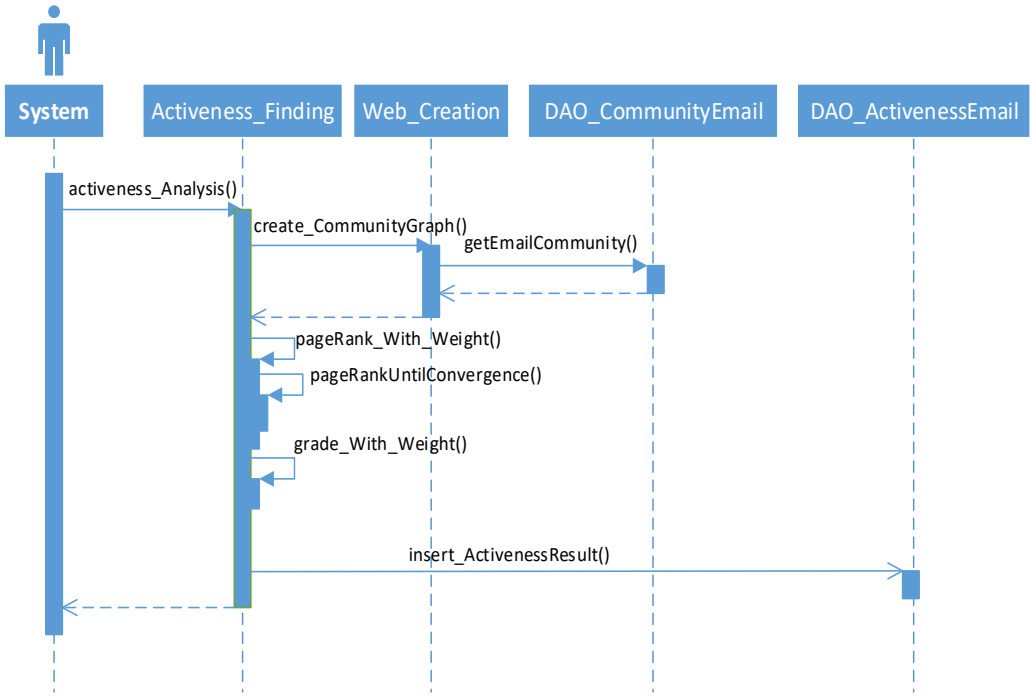

图 4-8 活跃人分析模块时序图

具体的程序流程如下：

（1） 系统完成社区划分模块后，用户点击活跃人分析按钮，系统利用用户输入的社区 ID 调用 Activeness_Finding 类中的 activeness_Analysis()函数。activeness_Analysis()函数接着调用 Web_Creation 类中的 create_CommunityGraph()函数。

（2） create_CommunityGraph()函数调用 DAO_CommunityEmail 类中的函数 getEmailCommunity()获取指定的社区中所有的邮件信息，并且根据这些邮件信息构建出以邮箱为点，以两个邮箱之间所有互发的邮件组成的集合为边的网络图，储存在 CommunityEmail_Web 结构体中。

（3） 网络构建完成后，activeness_Analysis()调用 Activeness_Finding 类内部的函数 pageRank_With_Weight()。pageRank_With_Weight()通过调用函数 pageRankUntilConvergence()计算出网络图中每个邮箱的考虑边权重的 PageRank 值，其中边的权重值为存储的互发邮件的数量。把邮箱按考虑边权重 PageRank 值排序，存入类变量 sorted_PageRank。

（4） activeness_Analysis()调用类内部的 grade_With_Weight()函数，得到一个邮箱使用人的排序，存入类变量 sorted_Grade，排序的依据是和这个代表邮箱的点相连的所有边的权重之和，其中边的权重值仍为存储的互发邮件的数量。

（5） activeness_Analysis()综合类变量 sorted_PageRank 和 sorted_Grade 得出一个邮箱总的活跃程度排序，其中 sorted_PageRank 中的值和 sorted_Grade 中的值在最后结果中的影响程度各占一半。activeness_Analysis()把活跃程度排序结果返回给视图层。

（6） 活跃人分析完成后，activeness_Analysis()调用 DAO_ActivenessEmail 类中的函数 insert_ActivenessResult()将活跃人分析的结果储存入电子邮件业务数据库中的 Email_Activeness 表里。

5.4.3. 重点算子分析

5.4.3.1. PageRank_With_Weight 计算函数

在 Spark 框架里，Graphx 中的 PageRank 类提供计算网络图中的每个点的 PageRank 值的功能接口。但是这些功能接口只提供不考虑边权重的 PageRank 值的计算，即在计算中把每个边的权重都当成 1。PageRank 值计算的原始目的是通过网页之间的超链接关系计算页面在互联网络中的重要程度。同时它也可以应用于其他场景，描述在其他类型的网络图中点的重要程度。在本系统中，PageRank 值可以表示邮箱的活跃程度，代表邮箱的点的 PageRank 的值越大，表示和他有联系的邮箱越多，或者和它有联系的邮箱的活跃程度越高。此时，边的权重就是边中储存的互发邮件的数量，对邮箱的活跃程度还是有一定影响的，因此参照 PageRank 类中的函数 runUntilConvergenceWithOptions() 在自定义的 Activeness_Finding 类中定义函数 pageRankUntilConvergence()，计算考虑边权重的 PageRank 值。因为函数应用的是消息传递迭代 Pregel 的方

式，”UntilConvergence”的意思是不规定迭代次数，直到各个点的 PageRank 值收敛为止。这样 pageRank_With_Weight() 函数可以通过调用 pageRankUntilConvergence()函数计算出网络图中每个邮箱的考虑边权重的 PageRank 值。

（1）基本思想

在 Graphx 图计算框架中函数 runUntilConvergenceWithOptions()中通过网页超链接关系计算页面 PageRank 的步骤如下：

①在初始阶段：根据超链接关系把所有网页构建成一张网络图，开始设置每个页面有相同的 PageRank 值，但是经过若干轮的计算后，每个网页所获得的最终 PageRank 值会被计算出来。网页当前的 PageRank 值会随着每一轮的计算不断更新。

②在每轮计算中，网页的 PageRank 得分都会被更新，计算方法如下:每个页面将其当前的 PageRank 值用平均分配的方式分配到这个页面的所有出链上，这样它的每个出链就获得了相等的分配到的权值。然后每个页面将所有指向该页面的入链所获得的权值求和，作为该页面新的 PageRank 值。当所有的页面的 PageRank 值都被更新了一次后，就完成了一轮 PageRank 计算。

把超链接看作是有向边，将网页看作是点，这样整个互联网就构成了一个有向图，如图 4-9 所示。用一个邻接矩阵 M 就可以表示此时页面之间的关系。如果第 I 个页面存在到一个到第 J 个页面的超链接，则矩阵中的元素 M[i][j]=1，如果不存在超链接，则 M[i][j]=0。

M={ 0, 1, 1, 0,
0, 0, 0, 1,
1, 0, 0, 0,
1, 1, 1, 0}

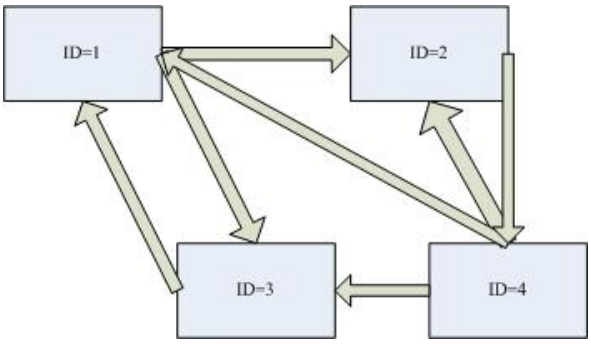

图 4-9 网页超链接关系有向图

M 的第 I 行表示第 I 个网页指向的所有网页，M 的第 J 列表示所有指向 J 的

网页。如果将 M 的每个元素都除于所在行的全部元素之和，然后再将 M 转置（交换行和列），得到 M^T

M^T={ 0, 0, 1, 1/3,
1/2, 0, 0, 1/3,
1/2, 0, 0, 1/3,
0,1, 0, 0 }

M^T[i][j]相当于第 i 个页面能分到 j 页面多少的重要程度，此时得到的结果是假设所有的链接权重是一样的，即迭代时一个页面的重要程度也在它所有链接的边上平均分配。

如果超链接边上的权重不相等，如图 4-10 所示，则改变一个页面的重要程度在它所有出链上的分配方式，即按照权重进行分配。边的权重越大，则页面在这条边上给对面的页面分配的重要程度就越多，反之则越少。这样边的权重在页面重要程度的计算上就起到了其应该起的作用。

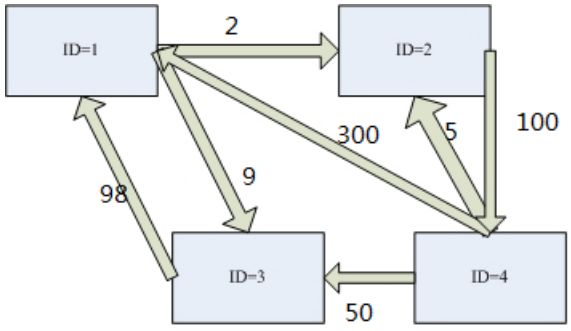

图 4-10 边权重不相等的网页超链接关系有向图

边权重不相等的网页超链接关系对应的邻接矩形为 M1

M1={ 0, 2, 9, 0,
0, 0, 0, 100,
98, 0, 0, 0,
300, 5, 50, 0}

如果将 M1 的每个元素都除于所在行的全部元素之和，然后再将 M1 转置（交换行和列），得到 $M1^T$

$M1^T$={ 0, 0, 1, 60/71,
2/11, 0, 0, 1/71
9/11 , 0, 0, 10/71,
0,1, 0, 0 }

此时 $M1^T$ 相当于考虑边权重的页面重要程度分配矩阵，M^T[i][j]相当于考虑边权重时第 i 个页面能分到 j 页面多少的重要程度。

（2）核心代码和算法流程

重要程度分配矩阵的建立对应于 Pergel 迭代开始之前对网络图的处理过程，在 Graphx 图计算框架中的 PageRank 类的 runUntilConvergenceWithOptions 函数里。代码如下：

```
val pagerankGraph: Graph[(Double, Double), Double] = graph
// Associate the degree with each vertex
.outerJoinVertices(graph.outDegrees) {
(vid, vdata, deg) => deg.getOrElse(0)
}
// Set the weight on the edges based on the degree
.mapTriplets( e => 1.0 / e.srcAttr )
// Set the vertex attributes to (initalPR, delta = 0)
```

其中，mapTriplets(e => 1.0 / e.srcAttr)就是构造一个如 M^T 一样的重要程度分配矩阵。此时如果改变 mapTriplets 的方式，构造一个如 $M1^T$ 一样的重要程度分配矩阵，就相当于改变每个边中储存的重要程度分配比例。改变后的 PageRankUntilConvergence()的部分代码见下:

```
var thing:RDD[(VertexId,Array[Edge[ED]])]
= graph.collectEdges(EdgeDirection.Out);
val sthing = thing.collect
val pagerankGraph: Graph[(Double, Double), Double] = graph
// Associate the degree with each vertex
.outerJoinVertices(graph.outDegrees) {
(vid, vdata, deg) => deg.getOrElse(0)
}
// Set the weight on the edges based on the degree
.mapTriplets( e => {
var sumr = 0.0;
for(i<- 0 until sthing.length){
val ooo = sthing(i);
if(ooo._1==e.srcId){
for(j<-0 until ooo._2.length){
sumr=sumr+f2(ooo._2(j));
}
}
```

```
            }
            f1(e) / sumr} )
        .mapVertices( (id, attr) => (0.0, 0.0) )
        .cache()
    }
```

PageRank_With_Weight()可以调用函数 PageRankUntilConvergence()得到考虑边权重的点的重点程度排序，流程图如图 4-11 所示。

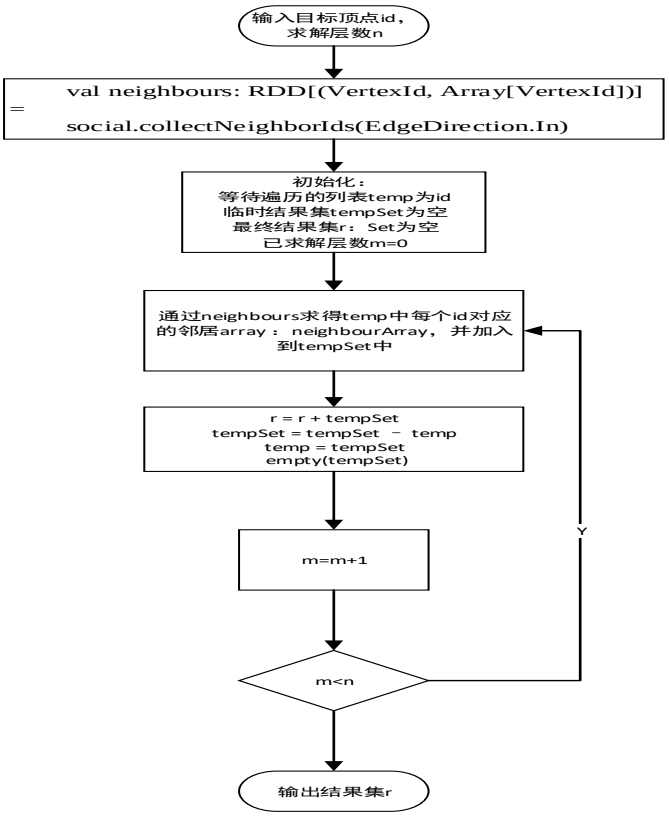

图 4-11 PageRank_With_Weight 函数流程图

5.4.3.2. Grade_With_Weight 计算函数

Grade_With_Weight()函数计算网络图中的每个点所有的出链入链的权重之和，再按照计算出来的值对点进行降序排序。其中边的权重值为边上存储的邮件的数量。网络图中每个点所有的出链入链的权重之和越大，意味着这个邮箱的使用人通过这个邮箱和其他人的联系就越频繁。

具体的函数算法流程图如图 4-12 所示。

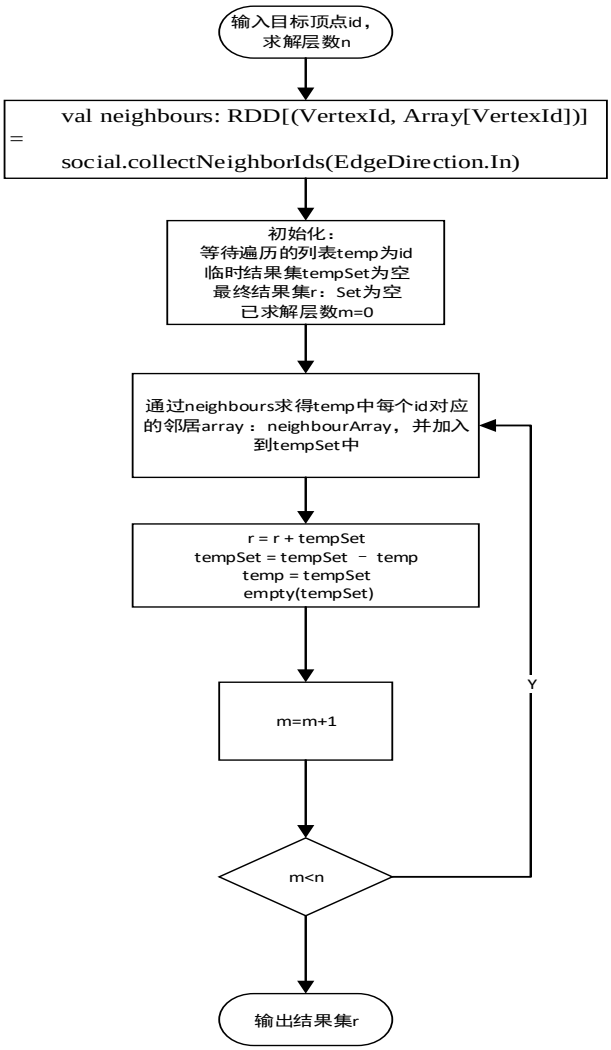

图 4-12 Grade_With_Weight 函数流程图

5.5. 联系人最短路径发现模块的设计与实现

联系人最短路径发现模块主要任务是找出两个邮箱使用人之间联系的所有最短路程。下面详细讲解模块中类的设计和流程实现。

5.5.1. 类的设计

联系人最短路径发现模块的功能主要由 Path_Finding 和 DAO_ShortestPathEmail 等设计的类中的成员变量实现，类图如 4-13 所示。

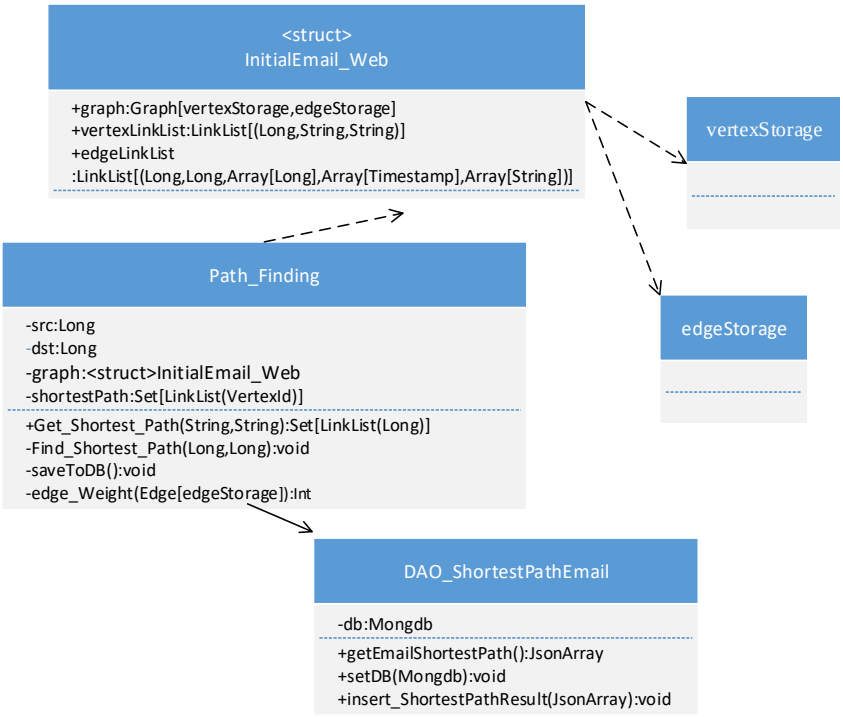

图 4-13 联系人最短路径发现模块类图

下面是对类图的详细解释：

（1） Path_Finding 类：联系人最短路径发现功能类，封装了两点最短路径发现函数，并且存储联系人最短路径发现的结果。

（2） DAO_ ShortestPathEmail 类：业务数据库操作类，封装了对电子邮件业务数据库中的 Email_ShortestPath 表里数据的存储、读取的操作函数。

（3） InitialEmail_Web 结构体:存储处理后的原始邮件数据信息和网络构建的结果。

（4） vertexStorage：实体类，封装了网络图中的点储存的所有属性。

（5） edgeStorage：实体类，封装了网络图中的边储存的所有属性。

5.5.2. 功能流程实现

联系人最短路径发现模块中各个对象的调用过程如图 4-14 所示。

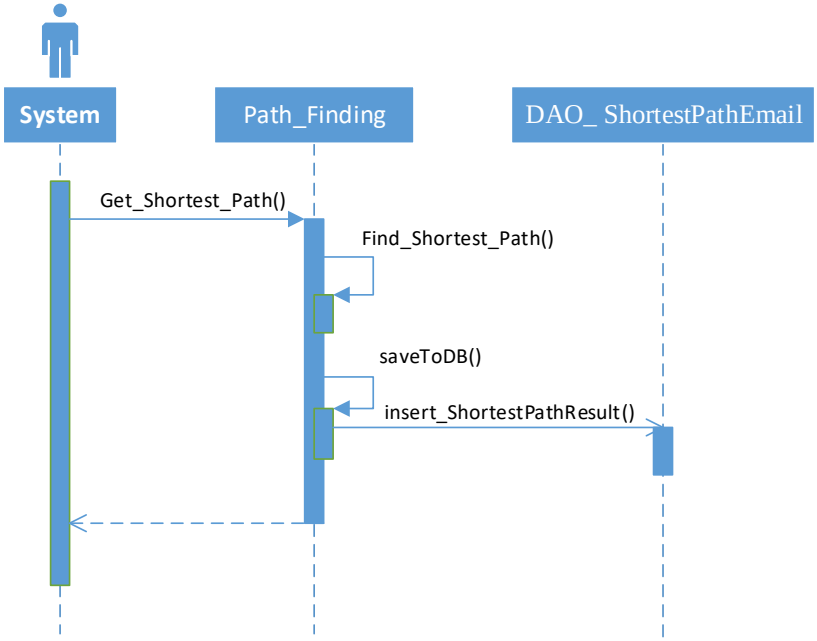

图 4-14 联系人最短路径发现模块时序图

具体的程序流程如下：

（1） 系统完成关系网络构建模块后，用户点击最短路径按钮，系统用需要找到最短路径的两个邮箱的地址字符串调用 Path_Finding 类中的函数 Get_Shortest_Path()，函数用邮箱地址在构建好的网路图 graph 中找到对应的 VertexId，并调用类内部函数 Find_Shortest_Path()。

（2） Find_Shortest_Path()函数找出网络图中两个点之间所有的路径，生成一个新的集合 Set[LinkList(VertexId)],存入类变量 shortestPath。其中一个 LinkList 代表一条最短路径，则变量 shortestPath 就是所有路径的集合。

（3） 所有最短路径都找到后，Get_Shortest_Path()将找到的所有最短路径结果返回给视图层，并调用类内部函数 saveToDB()。saveToDB()把 shortestPath 变量中 LinkList 里的 VertexId 都替换成邮箱地址字符串，并调用 DAO_ ShortestPathEmail 类中的 insert_ShortestPathResult()函数将找出的所有最短路径储存入电子邮件业务数据库中的 Email_ShortestPath 表中。

5.5.3. 重点算子分析

5.5.3.1. Find_Shortest_Path 计算函数

Find_Shortest_Path()函数的主要功能是计算网络图中给定的两个点之间的最短路径，不考虑边的权重值的影响，即每条边的长度都被认为是 1。

目前，求网络图中两点之间最短路径最常用的方法是 Dijkstra 算法。Dijkstra 算法应用了贪心算法模式，是目前公认的最好的求解最短路径的方法[30]。算法解决的是有向图中单个源点到所有的其他顶点的最短路径问题，其主要特点是每次迭代时选择的下一个顶点是标记点之外距离源点最近的顶点。本系统中的 Find_Shortest_Path()函数应用了 Dijkstra 算法的基本思想，在操作方式上进行了一些改动：一般的 Dijkstra 算法只计算最短路径的值，而 Find_Shortest_Path()函数不但需要计算最短路径的值，还需要得出所有的具体路径。

下面简单讲述 Find_Shortest_Path()函数的基本思想：函数计算源点 src 到终点 dst 的最短距离。每个点中都储存有当前的最短距离值和最短路径集合。最开始以源点 src 为标志点，在标志点的基础上延伸计算相邻的点的距离，如果新得出的距离比原距离小，则改变距离值为新得出的距离，并改变储存的最短路径为新得出的路径。如果新得出的距离和原距离相等，则不改变距离值，但是在储存的最短路径集合中加入新得出的路径。如果比原距离大，则距离和路径均不改变。此时挑选距离最小的点，确定这个点距源点的最短路径就是当前值储存的最短路径集合，将其加入 result 集合中。然后以这个新加入的点作为新的标志点，在标志点的基础上延伸计算相邻点的距离，已经确定最小距离的点不再考虑。重复上述过程，直到终点 dst 加入 result 集合中或者连通图所有点都加入了 result 中，则循环结束。如果 src 和 dst 不连通，则得到的结果是一个空集合。如果连通，则得到的结果是一个包含所有最短路径（每个路径用一个 vertexId 的 list 表示）的 Set 集合。

具体的函数的算法流程图如图 4-15 所示。

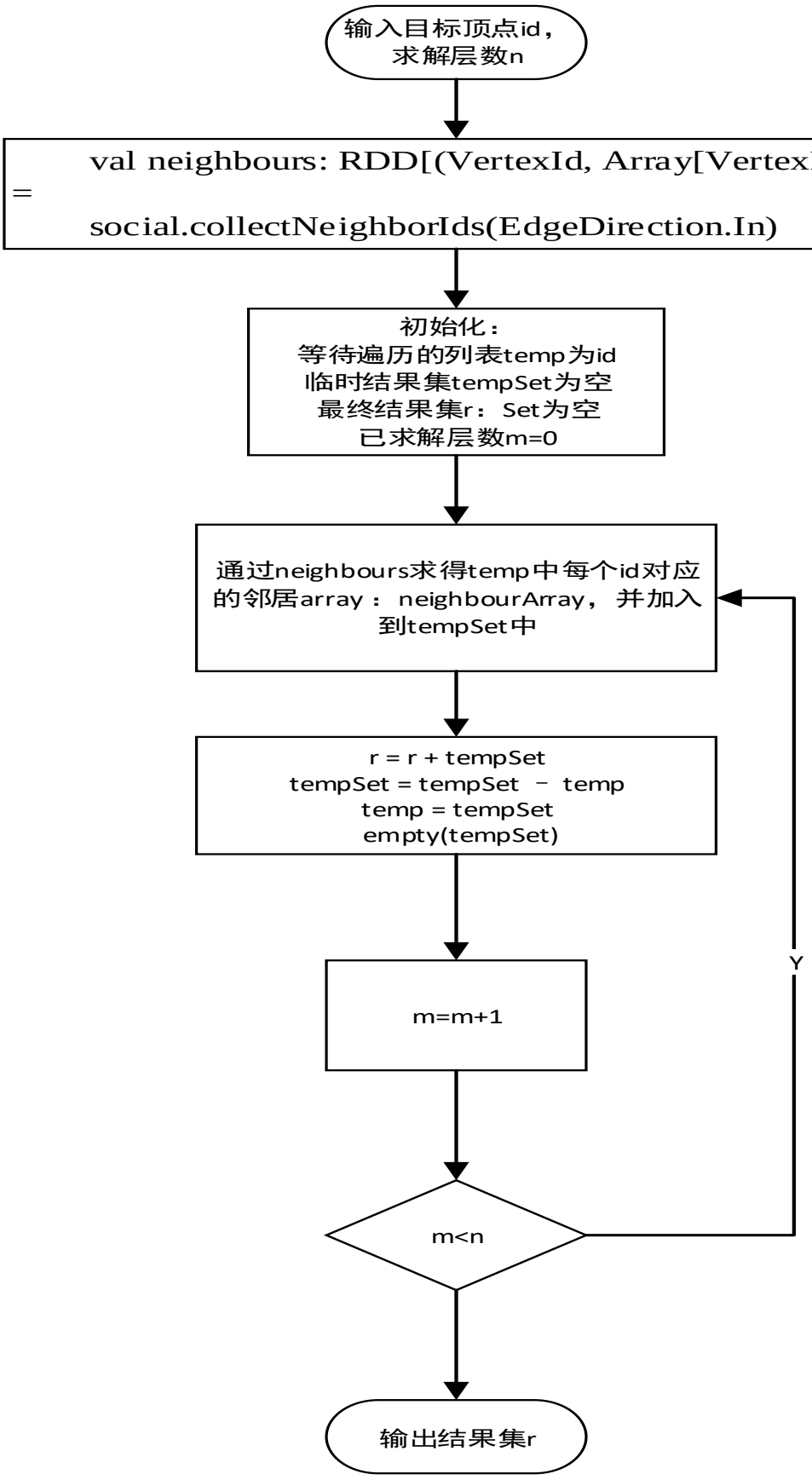

图 4-15 Find_Shortest_Path 函数流程图

5.6. 直接/间接联系人网络发现模块的设计与实现

直接/间接联系人网络发现模块主要任务是找出和某个邮箱使用人所有有直接或间接联系的邮箱构成的网络。下面详细讲解模块中类的设计和流程实现。

5.6.1. 类的设计

直接/间接联系人网络发现模块的功能主要由 ConnectionWeb_Finding 和 DAO_ConnectionWebEmail 等设计的类中的成员变量实现，类图如 4-16 所示。

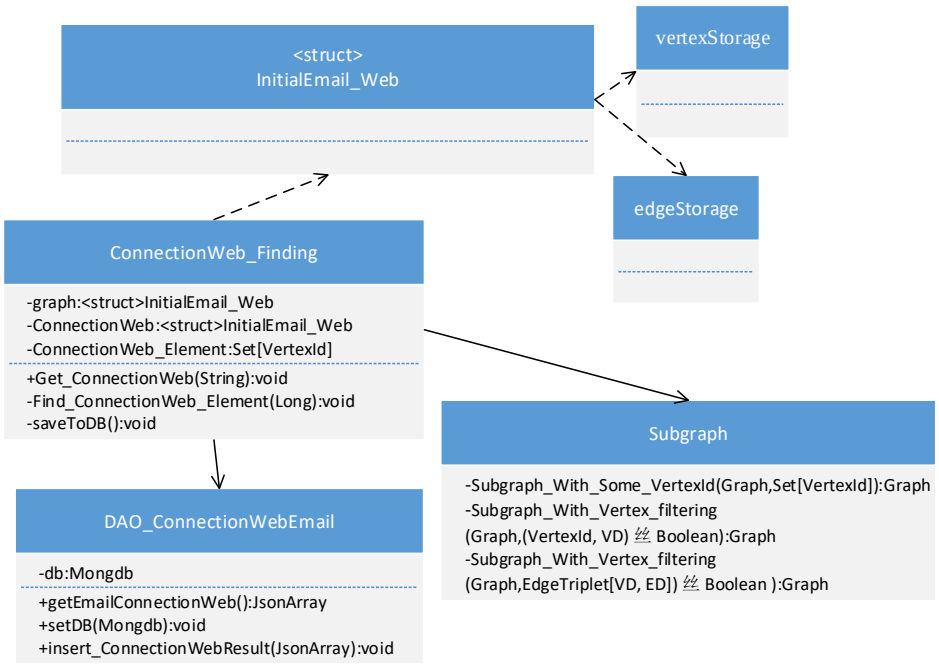

图 4-16 直接/间接联系人网络发现模块类图

下面是对类图的详细解释：

（1） ConnectionWeb_Finding 类：直接/间接联系人网络发现功能类，封装了直接/间接联系人网络发现函数，并且存储网络发现的结果。

（2） Subgraph 类：网络子图构建功能类，封装了几种不同条件下的子图发现函数。

（3） DAO_ConnectionWebEmail 类：业务数据库操作类，封装了对电子邮件业务数据库中的 Email_ConnectionWeb 表里数据的存储、读取的操作函数。

（4） InitialEmail_Web 结构体:存储处理后的原始邮件数据信息和网络构建的结果。

（5） vertexStorage：实体类，封装了网络图中的点储存的所有属性。

（6） edgeStorage：实体类，封装了网络图中的边储存的所有属性。

5.6.2. 功能流程实现

直接/间接联系人网络发现模块中各个对象的调用过程如图 4-17 所示。

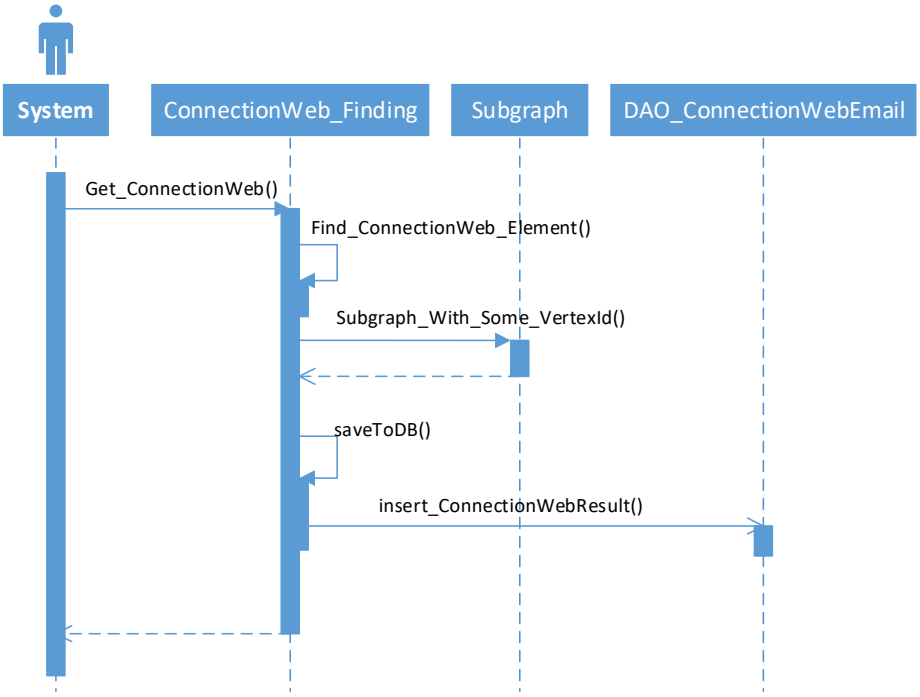

图 4-17 直接/间接联系人网络发现模块时序图

具体的程序流程如下：

（1） 系统完成关系网络构建模块后，用户点击联系人网络按钮，系统用需要找到联系人网络的邮箱的地址字符串调用 ConnectionWeb_Finding 类中的函数 Get_ConnectionWeb()，函数 Get_ConnectionWeb()用邮箱地址在构建好的网路图中找到这个邮箱对应的 VertexId 作为源点，并调用类内部函数 Find_ConnectionWeb_Element()。

（2） Find_ConnectionWeb_Element()函数找出所有包含在源点延伸出的直接/间接联系人网络中的代表邮箱的点，生成一个新的集合 Set[VertexId],存入类变量 ConnectionWeb_Element。

（3） 所有联系人网络中包含的点都找到后，Get_ConnectionWeb()调用 Subgraph 类中的 Subgraph_With_Some_VertexId()函数，得到包含这些的点的联通子图，这个联通子图就是源点的直接/间接联系人网络。Get_ConnectionWeb()函数将联系人网络结果返回给视图层，并且调用类内部函数 saveToDB()。saveToDB()把 ConnectionWeb_Element 变量中 VertexId 都替换成邮箱地址字符串，并调用 DAO_

ConnectionWebEmail 类中的 insert_ConnectionWebResult()函数将找出的直接/间接联系人网络中包含的点都储存入电子邮件业务数据库中的 Email_ConnectionWeb 表中。

5.6.3. 重点算子分析

5.6.3.1. Find_ConnectionWeb_Element 计算函数

Find_ConnectionWeb_Element()函数的主要功能是找出所有包含在源点 src 延伸出的直接/间接联系人网络里的代表邮箱的点。该函数采取 DFS 广度优先搜索方法，主要思想如下：定义集合 result，result 最开始只包含源点 src 的 VertexId。从源点 src 开始，先找到和它有直接的联系的点，把所有这些点的 VertexId 存入集合 result 中。然后再找出和这些新存入的点有直接联系的并且 result 集合中没有的点，再次存入集合 result 中。重复上述过程，直到没有新的点能存入，则循环结束，集合 result 为最终结果。

具体的函数算法流程图如图 4-18 所示。

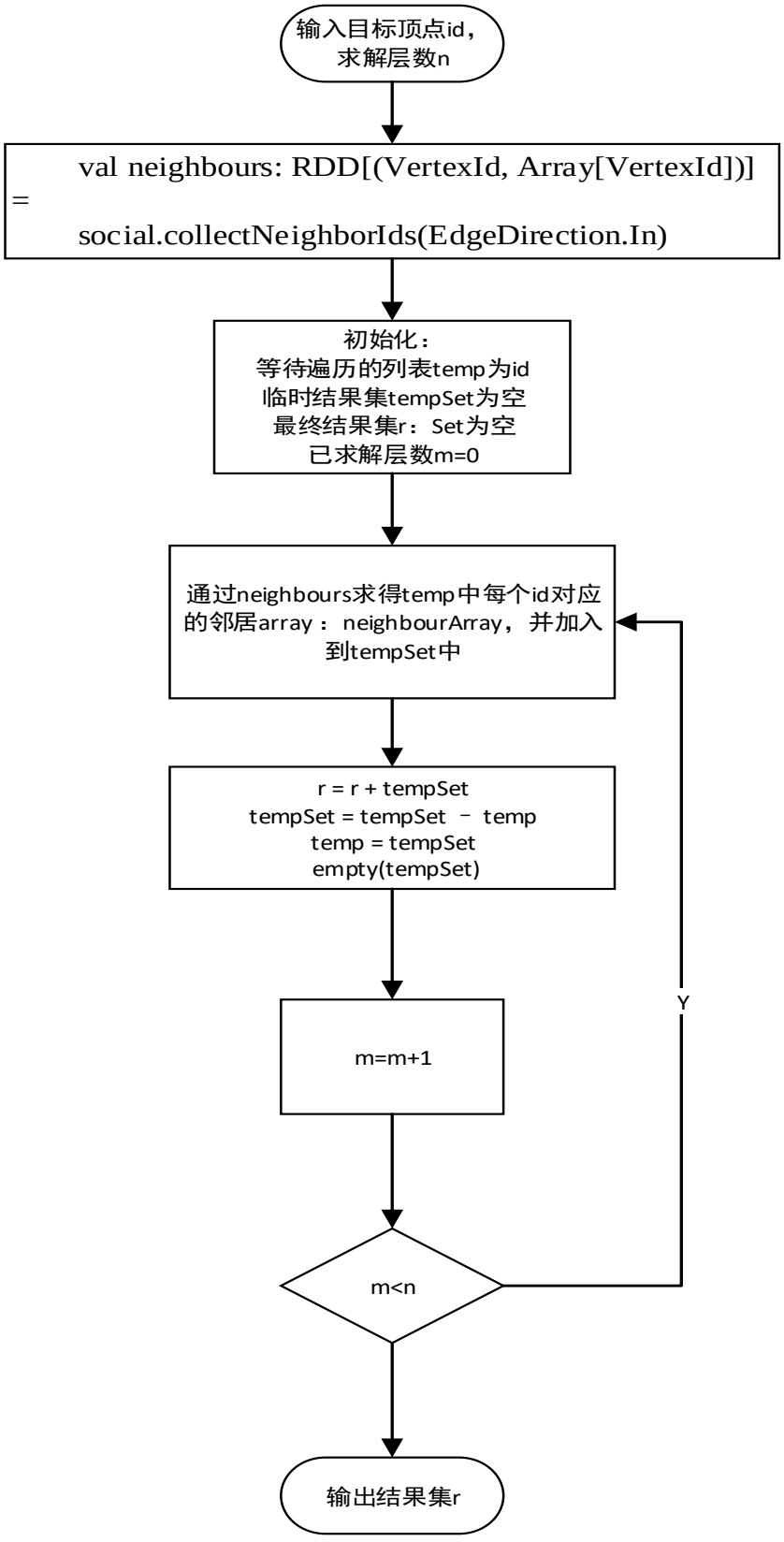

图 4-16 Find_ConnectionWeb_Element 函数流程图

5.7. 条件筛选模块的设计与实现

条件筛选模块的主要任务是根据用户指定的电子邮件和电子邮箱使用人的筛选条件在关系网络图中筛选出符合条件的点和边构建新的网络图。下面详细讲解模块中类的设计和流程实现。

5.7.1. 类的设计

条件筛选模块的功能主要由 Condition_Filtering 和 Subgraph 等设计的类中的成员变量实现，类图如 4-17 所示。

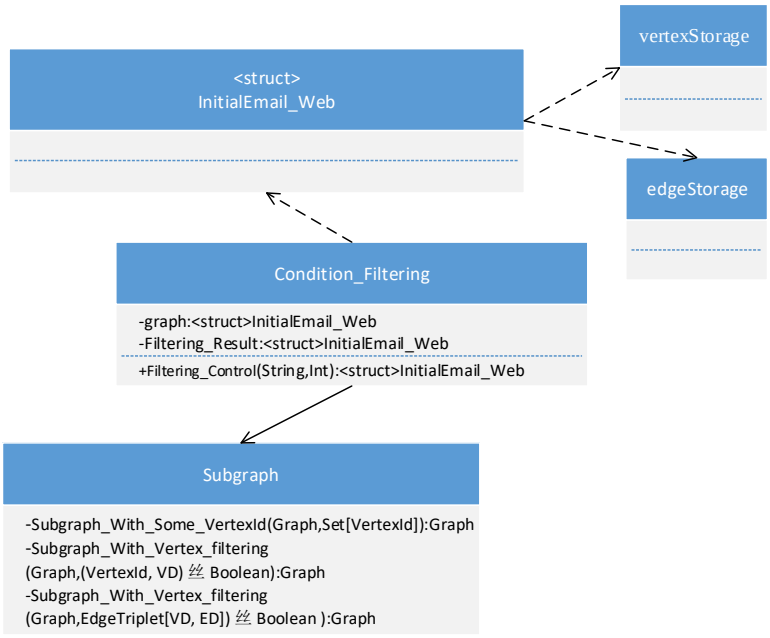

图 4-17 条件筛选模块类图

下面是对类图的详细解释：

（1） Condition_Filtering 类：条件筛选功能类，封装了负责筛选控制和筛选条件构造的函数，并且存储条件筛选的结果。

（2） Subgraph 类：网络子图构建功能类，封装了几种不同条件下的子图发现函数。

（3） InitialEmail_Web 结构体:存储处理后的原始邮件数据信息和网络构建的结果。

（4） vertexStorage：实体类，封装了网络图中的点储存的所有属性。

（5） edgeStorage：实体类，封装了网络图中的边储存的所有属性。

5.7.2. 功能流程实现

条件筛选模块中进行邮箱后缀筛选时各个对象的调用过程如图 4-18 所示。其他两个条件筛选的对象调用过程也类似。

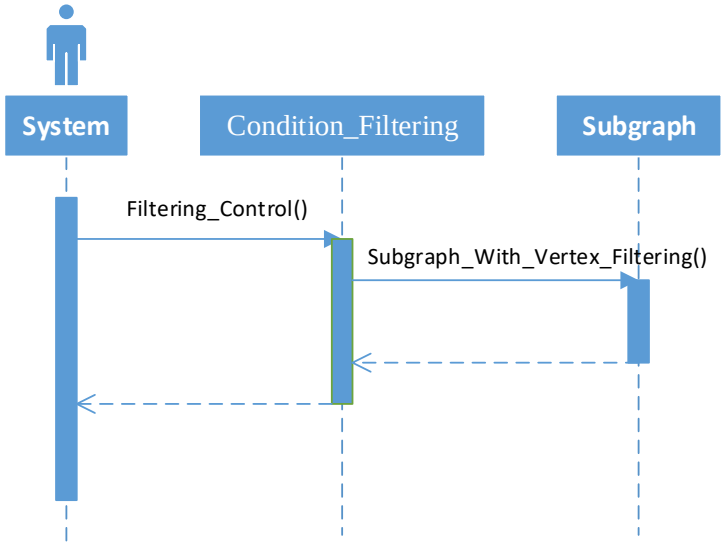

图 4-18 邮箱后缀条件筛选时序图

条件筛选模块中具体的程序流程如下：

（1） 系统完成关系网络构建模块后，用户点击条件筛选按钮，并选择筛选条件的种类。系统利用用户输入的筛选条件和筛选条件的类型调用函数 Condition_Filtering 类中的 Filonttering_Crol()函数。如果筛选条件是邮箱后缀，则调用 Filtering_Control()函数时的第二个参数为 0；如果筛选条件是邮箱间互发邮件的数量，则第二个参数为 1；如果筛选条件是邮箱的直接联系人数量，则第二个参数为 2。

（2） Filtering_Control()函数根据筛选条件的类型，对筛选条件进行解析并构造出对应的条件函数。然后函数 Filtering_Control()用构造出的条件函数调用 Subgraph 类里相应的子图发现函数构建符合条件的子图作为条件筛选的结果：如果筛选条件是邮箱后缀或邮箱的直接联系人数量，则调用函数 Subgraph_With_Vertex_Filtering()。如果筛选条件是邮箱间互发邮件的数量，则调用函数 Subgraph_With_Edge_Filtering()。

（3） 符合条件的子图构建完成后，Filtering_Control()函数将条件筛选后的结果返回给视图层。

5.8. 本章小结

本章对数据分析层各个功能模块的具体实现思路进行了详细介绍，其中包括数据库配置、关系网络构建、社区划分、活跃人分析、联系人最短路径发现、直接/间接联系人网络发现和条件筛选七个功能模块，并对模块中的重点函数进行详细说明。

第六章 系统测试

在投入生产前，对系统进行测试来证实其准确性、可用性和高效性是非常有必要的，本章利用生成的模拟电子邮件数据对面向电子邮件的社交网络分析系统进行功能测试。

6.1. 系统测试环境

系统的服务器测试环境如表 5-1 所示，客户端测试环境如表 5-2 所示。

表 5-1 服务器测试环境配置表

类别	标准配置
硬件	CPU:Intel(R) Xeon(R)
	E5-2630 v3 @2.40GHz
	内存：4GB
	硬盘：500G
软件	操作系统：Red Hat 6.5
	Spark 1.6.0
	MongoDB 3.0.4

表 5-2 客户端测试环境配置表

类别	标准配置
硬件	CPU:Intel(R) Xeon(R)
	E5-2630 v3 @2.40GHz
	内存：4GB
	硬盘：500G
软件	操作系统:WIN7

6.2. 系统功能性测试

本小节中用表格的形式描述对系统进行功能性测试的过程，针对的是七个系统提供给用户的功能点。使用的数据是实验室模拟产生的电子邮件数据。测试数据集中包括 78 个电子邮箱，400 封电子邮件数据。

（1）数据库配置用例测试

表 5-3 数据库配置连接测试用例表

测试用例名：	数据库配置连接测试
测试输入	（1）数据库地址 （2）数据库名 （3）用户名 （4）密码 （5）端口名
测试步骤	（1）点击功能面板中的“数据配置”图标，系统弹出数据库配置窗口 （2）在窗口的对应位置输入正确的数据库地址、数据库名称、用户名、密码和端口名，点击“确定”按钮。 （3）点击新弹出的窗口的“确定”按钮关闭窗口
预期输出	弹出窗口提示数据库连接成功
测试结果	测试通过

下面的图 5-1 是数据配置模块测试时的界面展示。

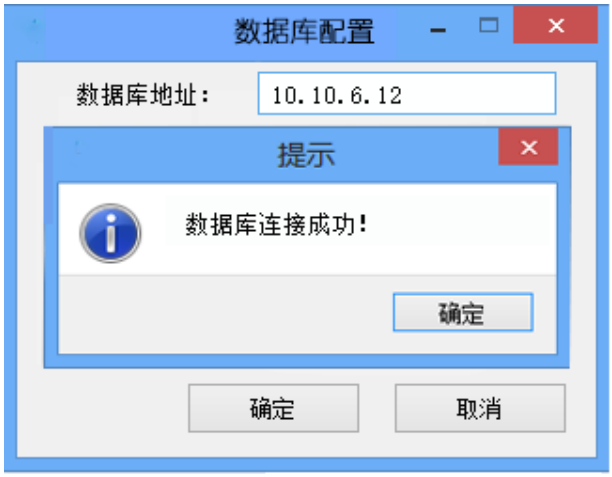

图 5-2 数据库配置界面展示图

（2）关系网络构建用例测试

表 5-4 网络构建测试用例表

测试用例名	关系网络构建测试
测试输入	无
测试步骤	（1）完成数据库配置连接测试后 （2）点击功能面板中的“数据导入”图标
预期输出	构建的好的电子邮箱为点，互发邮件关系为边的网络图展示在界面上

测试结果	测试通过

下面的图 5-2 是关系网络构建模块测试时的界面展示。

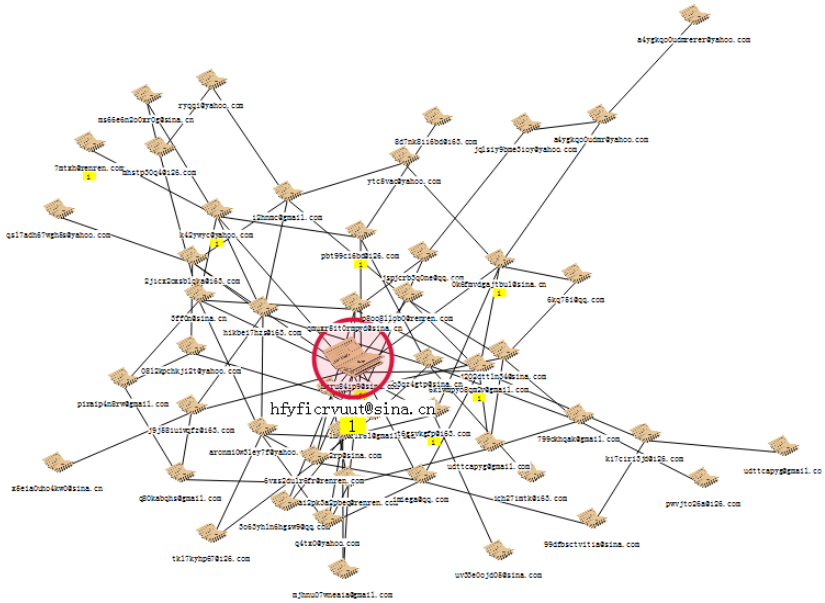

图 5-2 关系网络构建界面展示图

（3）社区划分用例测试

表 5-5 社区划分测试用例表

测试用例名	社区划分测试
测试输入	无
测试步骤	（1）完成关系网络构建测试后 （2）点击功能面板中的“社区划分”图标 （3）点击新弹出的窗口的“关闭”按钮关闭窗口
预期输出	以列表形式展示所有邮箱对应的社区，社区划分的结果在数据库中存储成功
测试结果	测试通过

下面的图 5-3 是社区划分模块测试时的界面展示。

	社区 ID	实体类型	实体标签	实体的出向链接数	实体的入向链接数
1	8490329320	电子邮箱	pmvjto26a@126.com	0	0
2	8490329320	电子邮箱	ryqqi@yahoo.com	0	0
3	8490329320	电子邮箱	3ff0n@sina.cn	0	0
4	8490329320	电子邮箱	o3qz4gtp@sina.cn	0	0
5	8490329320	电子邮箱	ms66e6n2o0xrg@sina.cn	0	0
6	8490329320	电子邮箱	udttcapyg@gmail.com	0	0
7	8490329320	电子邮箱	12hnmc@gmail.com	0	0
8	8490329320	电子邮箱	6kq75i@qq.com	0	0
9	8490329320	电子邮箱	p8oo8llob0@renren.com	0	0
10	8490329320	电子邮箱	q4tx0@yahoo.com	0	0
11	8490329320	电子邮箱	t6gsvkgfp@163.com	0	0
12	8490329320	电子邮箱	q80kabqhs@gmail.com	0	0
13	2645392102	电子邮箱	lmv31flr6l@gmail.com	0	0
14	2645392102	电子邮箱	mzru841p9@sina.com	0	0
15	2645392102	电子邮箱	hikbei7hzs@163.com	0	0

图 5-3 社区划分界面展示图

（4）活跃人分析用例测试

表 5-6 活跃人分析测试用例表

测试用例名	活跃人分析测试
测试输入	（1）社区 ID 值
测试步骤	（1）完成社区划分测试后 （2）点击功能面板中的“活跃人分析”图标，系统弹出窗口 （3）在窗口对应位置输入正确的社区 ID 值，点击“确定”按钮 （4）点击新弹出的窗口的“关闭”按钮关闭窗口
预期输出	以列表形式展示指定的社区中所有的邮箱使用人的活跃程度值排序，活跃人分析结果在数据库中存储成功
测试结果	测试通过

下面的图 5-4 是活跃人分析模块测试时的界面展示。

	社区 ID	实体类型	实体标签	实体的出向链接数	实体的入向链接数	活跃程度
1	2645392102	电子邮箱	qsl7adh67wgh5s@yahoo…	0	0	100.0
2	2645392102	电子邮箱	pbt99ci6bd@126.com	0	0	96.7
3	2645392102	电子邮箱	ai2pk3a2pbeq@renren.…	0	0	90.3
4	2645392102	电子邮箱	6kiwmpyo8qm2v@gmail.…	0	0	81.9
5	2645392102	电子邮箱	799okhqak@gmail.com	0	0	80.2
6	2645392102	电子邮箱	bo5kf6nyqp@163.com	0	0	80.1
7	2645392102	电子邮箱	uv33e0ojd05@sina.com	0	0	79.6
8	2645392102	电子邮箱	imiega@qq.com	0	0	78.3
9	2645392102	电子邮箱	x5eia0uho4kw0@sina.cn	0	0	69.8
10	2645392102	电子邮箱	1ch271mtk@163.com	0	0	65.9
11	2645392102	电子邮箱	aronmi0w3ley7f@yahoo…	0	0	60.4
12	2645392102	电子邮箱	j202dttln34@sina.com	0	0	60.4
13	2645392102	电子邮箱	lmv31flr6l@gmail.com	0	0	60.2
14	2645392102	电子邮箱	mzru841p9@sina.com	0	0	54.3
15	2645392102	电子邮箱	hikbei7hzs@163.com	0	0	50.2

图 5-4 活跃人分析界面展示图

（5）联系人最短路径用例测试

表 5-7 联系人最短路径测试用例表

测试用例名	联系人最短路径测试
测试输入	（1） 两个不同的邮箱地址
测试步骤	（1）完成关系网络构建测试后 （2）点击功能面板中的“最短路径”图标，系统弹出窗口 （3）在窗口对应位置分别输入两个正确的邮箱地址，点击“确定”按钮 （4）点击新弹出的窗口的“确定”按钮关闭窗口 （5）点击控制面板中的“显示所选项”图标。
预期输出	点击“最短路径”图标后，最短路径中包含的点和边在网络图中被标出，系统弹出窗口显示最短路径的长度 点击“显示所选项”图标后，新的只包含最短路径中的点和边的网络图被展示出来。 联系人最短路径结果在数据库中存储成功
测试结果	测试通过

下面的图 5-5 是联系人最短路径模块测试时的界面展示。

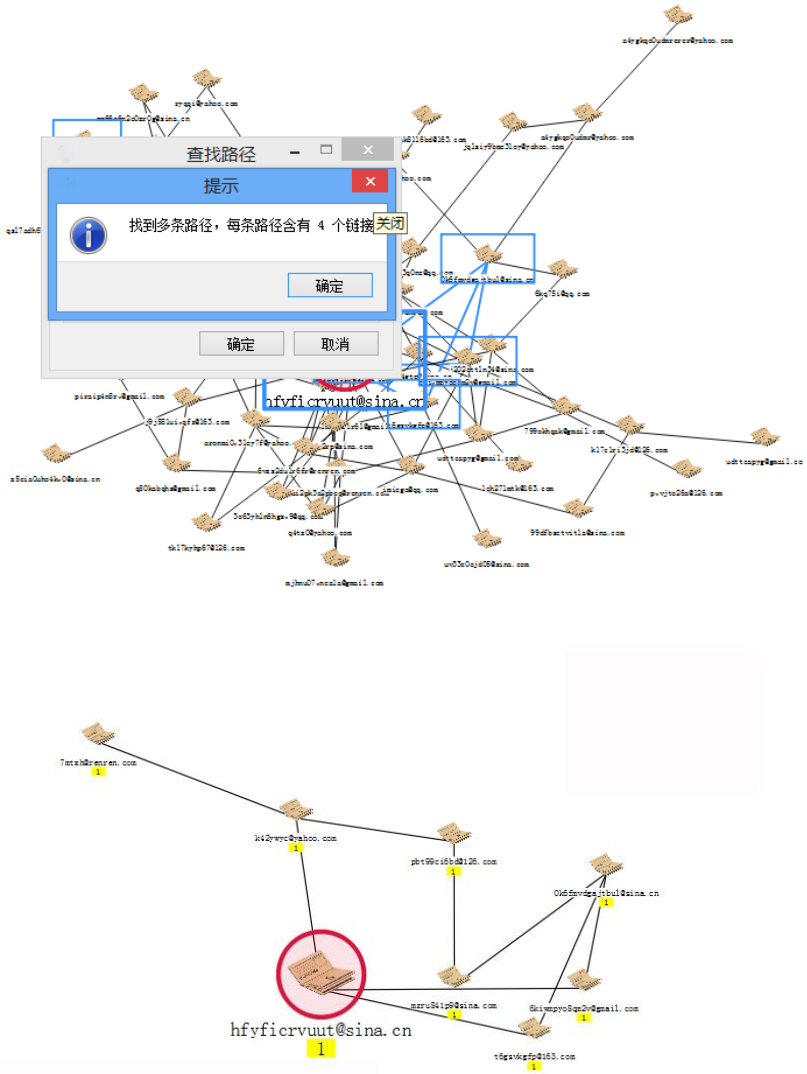

图 5-5 联系人最短路径界面展示图

（6）直接/间接联系人用例测试

表 5-8 直接/间接联系人网络构建测试用例表

测试用例名	直接/间接联系人网络构建测试
测试输入	（1）一个邮箱地址
测试步骤	（1）完成关系网络构建测试后 （2）点击功能面板中的“联系人网络”图标，系统弹出窗口 （3）在窗口对应位置输入正确的邮箱地址，点击“确定”按钮 （4）点击新弹出的窗口的“确定”按钮关闭窗口

	（5）点击控制面板中的“显示所选项”图标。
预期输出	点击“联系人网络”图标后，直接/间接联系人网络中包含的点和边在网络图中被标出，系统弹出窗口显示包含的点和边的数量 点击“显示所选项”图标后，新的只包含直接/间接联系人网络中的点和边的网络图被展示出来。 直接/间接联系人网络构建的结果在数据库中存储成功
测试结果	测试通过

下面的图 5-6 是直接/间接联系人网络模块测试时的界面展示。

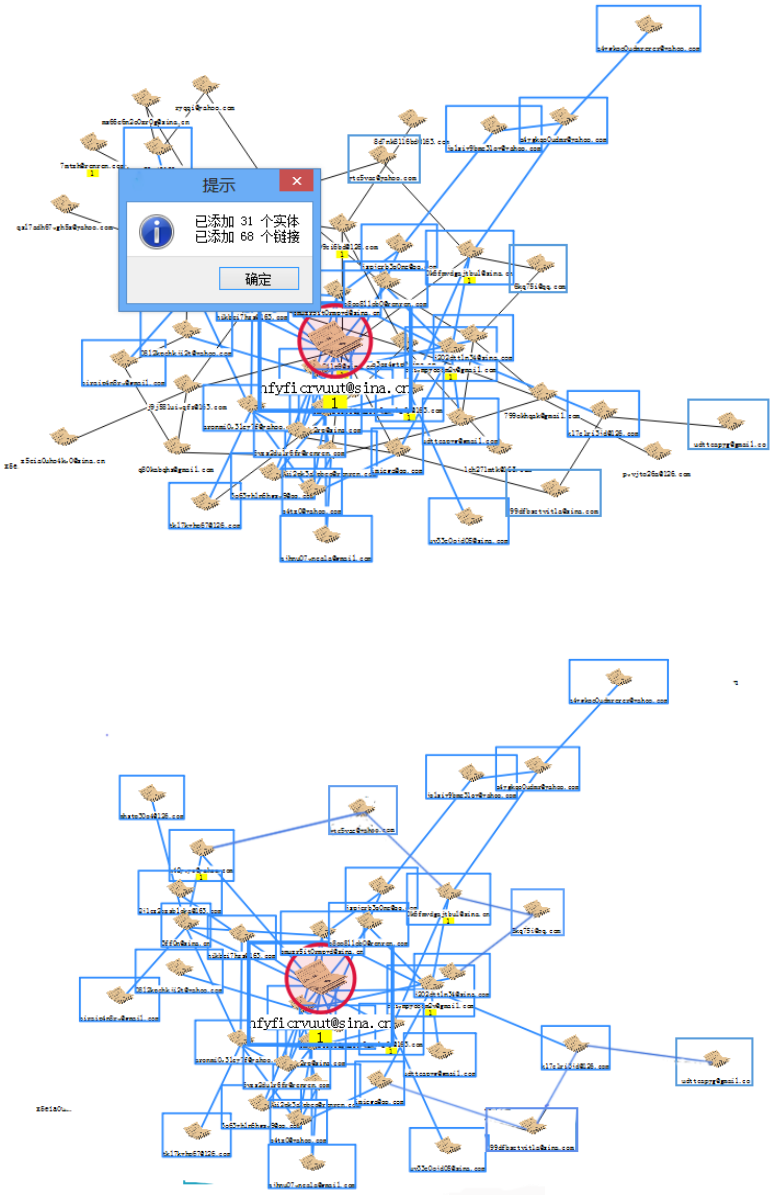

图 5-6 直接/间接联系人网络界面展示图

（7）条件筛选用例测试

表 5-8 条件筛选测试用例表

测试用例名	条件筛选测试
测试输入	（1）筛选条件，一个邮箱后缀名 （2）筛选条件，一个整数值作为互发邮件数量的下限 （3）筛选条件，一个整数值作为直接联系的邮箱的个数的下限
测试步骤	（1）完成关系网络构建测试后 （2）点击功能面板中的“条件筛选”图标，系统弹出窗口 （3）在窗口中选择“邮箱后缀”标签，并在对应位置的菜单中选择其中一个邮箱后缀，点击“确定”按钮 （4）再次点击功能面板中的“条件筛选”图标，系统弹出窗口 （5）在窗口中选择“互发邮件数量”标签，并在对应位置的菜单中输入一个大于 1 的整数值，点击“确定”按钮 （6）再次点击功能面板中的“条件筛选”图标，系统弹出窗口 （7）在窗口中选择“直接联系的邮箱的个数”标签，并在对应位置的菜单中输入一个大于等于 1 的整数值，点击“确定”按钮
预期输出	每个筛选后的结果都是符合筛选条件的点或边在网络图中被标出
测试结果	测试通过

下面的图 5-7 是条件筛选模块测试对邮箱后缀筛选时的界面展示。

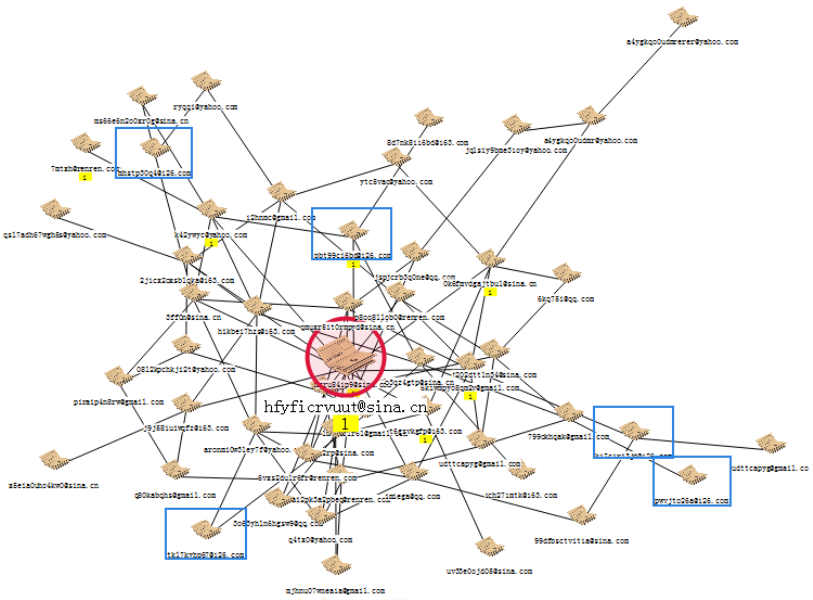

图 5-7 条件筛选邮箱后缀筛选界面展示图

6.3. 系统性能测试

本小节中针对关系网络构建、社区划分、活跃人分析、联系人最短路径和直

接/间接联系人网络发现等功能模块进行性能方面的测试。使用的数据是实验室模拟产生的电子邮件数据，数据集划分为 1 千封电子邮件、1 万封电子邮件和 10 万封级三个大小等级进行测试。

（1）**关系网络构建模块性能测试**

电子邮件数据集	1 千封	1 万封	10 万封
响应时间(min)	0.02	0.05	0.24

（2）**社区划分模块性能测试**

电子邮件数据集	1 千封	1 万封	10 万封
响应时间(min)	0.12	0.35	2.08

（3）**活跃人分析模块性能测试**

电子邮件数据集	1 千封	1 万封	10 万封
响应时间(min)	0.42	0.57	1.04

（4）**联系人最短路径模块性能测试**

电子邮件数据集	1 千封	1 万封	10 万封
响应时间(min)	0.06	0.66	9.71

（5）**直接/间接联系人网络发现模块性能测试**

电子邮件数据集	1 千封	1 万封	10 万封
响应时间(min)	0.06	0.89	14.60

在分别对三个级别的数据集进行测试后，观察上述测试结果，随着数据量的增加，联系人最短路径模块和直接/间接联系人网络发现模块的消耗时间有着大幅度地增加。针对联系人最短路径模块的性能测试显示，1 万封电子邮件数据花费时间是 1 千封电子邮件数据集的 11 倍，10 万封电子邮件数据花费时间是 1 万封数据集的 15 倍。而针对直接/间接联系人网络发现模块的测试显示，1 万封电子邮件数据花费时间是 1 千封电子邮件数据集的 14 倍，10 万封电子邮件数据花费时间是 1 万封数据集的 16 倍。与上述情况相反，活跃人分析模块表现出来的性能并没有随着数据集的增大发生和上述两个模块类似的剧烈的变化。这与活跃人分析模块实用的迭代 Pregel 算法有着密切的关系。上述测试结果，基本符合预

期的功能结果。

6.4. 本章小结

本章对系统的测试进行了说明，通过测试数据集以及测试环境的说明，并用图片和表格的方式展示测试的结果，最后对这些测试结果进行分析说明，充分验证了系统功能的正确性、可用性，高效性。

第七章 总结与展望

7.1. 全文总结

针对对大量的电子邮件数据进行自动化处理，并直观地展示数据分析的结果的需求，本文设计并实现了一个面向电子邮件的社交网络分析系统。系统基于B/S 架构，保证客户端电脑不需要承担太大的载荷，而且减轻系统维护与升级的成本和工作量。该系统能够对电子邮件数据进行处理挖掘得到电子邮箱用户群体特征和交互行为分析的结果，提供关系网络构建、活跃人分析、联系人最短路径发现、直接/间接联系人网络发现和条件筛选等主要功能。系统同样提供展示界面，使得使用该系统的数据分析人员可以简单便捷地操作系统完成数据的分析工作，并清晰直观地观察分析的结果。性能测试结果表明，该系统的性能基本符合预期，能够满足对海量电子邮件数据进行高效快捷地分析处理的需求。

7.2. 展望

本论文所提出的面向电子邮件的社交网络分析系统为很多需要对电子邮件用户群体特征和交互行为进行获取分析的企业部门提供了简单实用的处理工具，节省了对电子邮件数据进行统计分析的人力时间成本。

然而，由于面向电子邮件的社交网络分析系统需要针对用户行为状态进行多方面的分析，而且功能的需求并非一成不变的，同时由于本系统的需求分析、设计、实现等相关时间也相对有限，因此在整个系统的设计方案方面可能有遗漏的地方，实现方法也可能不是最优化的。而且本系统截至目前的所有的测试数据均为实验室模拟生成的电子邮件数据，并未在更广泛更全面的各种的真实环境下进行测试，所以整个系统可能存在不完善的地方。目前系统运行良好，但是还需要在更大数据集下进行进一步的压力性能测试。并且如果未来有其他方面的分析需求，系统也会进一步进行功能方面的完善工作。

参考文献

[1]程学旗, 靳小龙, 王元卓, 等. 大数据系统和分析技术综述[J]. 软件学报, 2014, 25(9): 1889-1908.

[2]郭永建，郑麟，郭杰.大数据时代背景下的海量电子邮件分析[J].警察技术，2014(1),3(1):25-29.

[3]王强,李俊杰,陈小军,黄哲学,陈国良. 大数据分析平台建设与应用综述[J].集成技术, 2016(3), 5(2):2-18.

[4]蔡伟鸿，汤立浩.电子邮件分析系统的设计[J].汕头大学学报，2014,19(2):75-78.

[5]李稚楹，杨武，谢治军. PageRank 算法研究综述[J].计算机科学，2011(10), 38(10A):186-188.

[6]仇丽青, 陈卓艳. 基于 Pagerank 的社会网络关键节点发现算法[J].软件导刊，2008(8), 13(8):48-49.

[7]马俊,周刚,许斌,黄永忠.基于个人属性特征的微博用户影响力分析[J].网络与通讯技术.2013,30(8):2483-2487.邓波, 张玉超, 金松昌,等.

[8]张震.电子邮件网络社区发现技术研究[D].华中科技大学.2013:1-64.

[9]郑浩原.在线个人社交网络可视分析系统中的社区挖掘.暨南大学,2012:1-72.

[10]邓波,张玉超,金松昌.基于 MapReduce 并行架构的大数据社会网络社团挖掘方法[C].中国计算机学会 ccf 大数据学术会议. 2013.

[11]肖云鹏.在线社会网络用户行为模型与应用算法研究[D].北京邮电大学,2013:1-126.

[12]张彦超,刘云,张海峰等.基于在线社交的信息传播模型[J].物理学报,2011,60(5):050501.

[13]李文栋.基于 Spark 的大数据挖掘技术的研究与实现[D].山东大学,2015:1-67.

[14]Gonzalez J E, Xin R S, Dave A, et al. Graphx: Graph processing in a distributed dataflow framework[C]//11th USENIX Symposium on Operating Systems Design and Implementation (OSDI 14). 2014: 599-613.

[15]Zhou W, Han J, Gao Y, et al. An efficient graph data processing system for large‐scale social network service applications[J]. Concurrency and Computation: Practice and Experience, 2014.

[16]Xin R S, Crankshaw D, Dave A, et al. GraphX: Unifying data-parallel and graph-parallel analytics[J]. arXiv preprint arXiv:1402.2394, 2014.

[17]王文生.电子邮件分析取证系统的设计与实现[D].华南理工大学,2011:1-78.

[18]乔少杰，唐常杰，彭京等.基于个性特征仿真邮件分析系统挖掘犯罪网络核心[J].计算机学报，2008(10),31(10):1795-1803.

[19]易成岐，鲍媛媛，薛一波．社会网络大数据分析框架及其关键技术[J]．中兴通讯技术, 2014(1):5-10.

[20]Erich Gamma，Richard Helm，Ralph Johnson.设计模式：可复用面向对象软件的基础（第 1 版）[M].北京：机械工业出版社,2007.

[21]杨成.基于 MapReduce 的社会网络分析系统的研究与实现[D].北京邮电大学,2010:1-72.

[22]谢美英.基于社会网络分析的银行用户管理系统的设计与实现[D].湖南大学,2010:1-58.

[23]汪博洋，王朝坤，张君.一个新颖的社交网络分析系统[J].计算机研究与发展,2015,52(Suppl.):134-158.

[24]丁晓冬.用于网络监管的社交网络可视化分析工具的设计和实现[D].上海交通大学,2013:1-79.

[25]刘晓曼.社交网络数据获取与结构分析系统的设计与实现[D].安徽大学,2014:1-69.

[26]杨寅．X-RIME：基于 Hadoop 的大规模社会网络分析.中国科技论文在线(http://www.paper.edu.cn).

[27]魏勇,唐文彬,郭梅.基于 DAO 模式的 J2EE 应用程序的数据库访问设计[J]. 计算机应用, 2003(S2):456-469.

[28]陈秋容,刘丹.基于 B/S 架构的质检机构客户管理系统的设计[J]. 福建质量管理, 2016(03):234-241,.

[29]欧阳宏基,解争龙,黄素萍.一种基于 DAO 设计模式与 Hibernate 框架的数据持久化模型[J]. 微计算机应用,2009(03):25-66.

[30]Thomas H.Cormen，Charles E.Leiserson，Ronald L.Rivest，Clifford Stein.算法导论（第二版）[M]．北京：机械工业出版社,2006.

致谢

值此论文完稿之际，首先谨向尊敬的导师岳银亮老师和徐迎晓老师致以崇高的敬意和诚挚的谢意。本次论文写作从立题、搜集资料、起草到最后成文，期间两位老师在论文选题、论文写作、学术研究等方面都给了我极大的帮助、鼓励和关怀。他们对待科研极其严谨和朴实，哪怕是毕业设计论文中的每一处标点，他们都仔仔细细的为我检查，使我获益良多。老师们严谨求实、认真负责、以身作则和渊博的学识使我受益匪浅。在此再次向两位老师对本人论文指导表示衷心的感谢！

感谢在复旦大学读书期间授予我知识的所有老师，以及所有在我求学过程中给予我支持和帮助的同学们和朋友们，大家曾经给予的鼓励、帮助和支持我永远会铭记在心，将激励着我继续向新的人生历程迈进。

感谢父母多年来的理解和支持，他们无私的爱激励着我克服学习和生活中的一个又一个困难，不断前进，在此向他们表达深深的敬意！