

密级： 保密期限：

北京邮电大学

硕士研究生学位论文



题目：问答式社会化搜索系统客户端的研究与实现

学 号：096625

姓 名：王少岩

专 业：通信与信息系统

导 师：龚向阳

学 院：网络技术研究院

2011 年 12 月 29 日

独创性（或创新性）声明

本人声明所呈交的论文是本人在导师指导下进行的研究工作及取得的研究成果。尽我所知，除了文中特别加以标注和致谢中所罗列的内容以外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得北京邮电大学或其他教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示了谢意。

申请学位论文与资料若有不实之处，本人承担一切相关责任。

本人签名：_____ 日期：_____

关于论文使用授权的说明

学位论文作者完全了解北京邮电大学有关保留和使用学位论文的规定，即：研究生在校攻读学位期间论文工作的知识产权单位属北京邮电大学。学校有权保留并向国家有关部门或机构送交论文的复印件和磁盘，允许学位论文被查阅和借阅；学校可以公布学位论文的全部或部分内容，可以允许采用影印、缩印或其它复制手段保存、汇编学位论文。（保密的学位论文在解密后遵守此规定）

保密论文注释：本学位论文属于保密在__年解密后适用本授权书。非保密论文注释：本学位论文不属于保密范围，适用本授权书。

本人签名：_____ 日期：_____

导师签名：_____ 日期：_____

问答式社会化搜索系统客户端的研究与实现

摘 要

随着社交网络服务（Social Network Service, SNS）的高速发展，社会化搜索（Social Search）已经变得越来越流行。问答式社会化搜索系统就是基于社会化搜索的一个问答社区，在这里人们不仅仅可以寻求帮助和分享经验，还可以关注感兴趣的人，流行的话题，甚至是特定的问题，这些关注关系把整个社区有效地联系在一起。该系统提供了类似于 Wiki 的协作编辑功能，所有的内容都由用户来主导和创造，并且提供了投票机制让用户来决定答案的质量，系统还提供了实时搜索机制让用户可以随时搜索自己感兴趣的内容，系统后台的推荐机制通过学习用户的行为，并借助用户的社交关系极大地扩展了每个用户的社交圈和知识面，所有这些机制都给用户带来了全新的体验。

本论文首先对社会化搜索的相关技术进行了介绍，进而重点讲述了客户端开发的相关技术和本系统使用这些技术的原因；之后对基于特定话题的用户影响力计算方法进行了探讨，提出了一个基于 PageRank 算法的改进的模型，该模型综合考虑了用户的问题网络和社交网络来衡量用户的影响力；接下来详细描述了问答式社会化搜索系统的需求分析，架构设计，以及客户端的详细设计与实现，其中重点分析了用户管理、个人信息管理、会话管理、内容管理、实时搜索和关注管理各个子模块的详细设计与实现；最后，通过集成测试和单元测试对系统进行了验证和分析，测试表明了整个系统达到了预期的设计目标。

论文最后总结了作者在研究生期间的工作和成果，提出了系统中还存在的一些缺点和问题，并且对未来的工作进行了展望。

关键词 社会化搜索 影响力计算 专家发现 PageRank 问答社区

RESEARCH AND IMPLEMENTATION OF CLIENT-SIDE IN SOCIAL QUESTION ANSWERING SEARCH SYSTEM

ABSTRACT

With the explosive growth of social network service, social search service is becoming more and more popular. Social question answering (Q&A) search system, which is based on social search service, is a Q&A community for people to seek answers and share experience. The entire community is connected based on the “following” relationship where people tend to follow popular topics, specific questions and other people they are interested in. The system provides collaborative editing features like Wiki where all content is created by users, and provides evaluations of high quality content which note the importance by voting answers. Also real-time searching and recommendation mechanisms which enhance the user experience are brought to this system.

First, this thesis introduces the social search technologies and then focuses on the client-side development technologies and the reasons for using them. Second, a model of measuring the influence of users in specific topic is proposed, and an extension of PageRank is used to iteratively calculate the rank value which taken link structure, user activity and user authority into account. Third, the detailed description of system requirement analysis, architecture design and implementation are discussed. The detailed design and implementation of each sub modules, such as user management, personal information management, session management, content management, real-time searching and following management are described. Finally, integration testing and unit testing of the system are verified and analyzed, which indicating that the entire system achieve the desired design goals.

At last, the thesis summarizes the whole work of the system, which shows there are still some shortcomings and further work in this system, and summarizes the work and achievements of the author during the post-graduate.

KEY WORDS: social search; influence measurement; expert finding; pagerank; question answering community

目 录

第一章	绪论	1
1.1.	课题背景	1
1.2.	课题主要研究内容	2
1.3.	主要工作内容	3
1.4.	论文结构	3
第二章	社会化搜索相关技术	4
2.1.	社会化搜索理论	4
2.1.1.	传统搜索的社会化属性	4
2.1.2.	新型社会化搜索技术	5
2.2.	客户端开发技术	6
2.2.1.	JSP 技术	6
2.2.2.	HTML 和 CSS 技术	7
2.2.3.	JQuery 技术	7
2.3.	Web 通信技术	7
2.3.1.	AJAX 技术	7
2.3.2.	JSON 标准	8
第三章	基于社会化搜索的用户影响力模型研究	9
3.1.	研究意义	9
3.2.	模型背景	9
3.3.	爬虫设计	10
3.3.1.	爬虫背景和相关技术	10
3.3.2.	爬虫架构	10
3.3.3.	Quora 爬虫	11
3.3.4.	话题网络爬虫	14
3.3.5.	SNS 网站爬虫	15
3.4.	数据集分析	16
3.4.1.	Bow tie 模型	17
3.4.2.	Quora 分布	18
3.5.	模型介绍	19
3.5.1.	问题网络构建	19
3.5.2.	社交网络构建	21
3.5.3.	排序引擎	22
3.6.	验证结果	23
3.6.1.	排名结果分析	23
3.6.2.	常用计算方法比较	23
3.6.3.	模型总结	24
第四章	需求分析	26
4.1.	功能需求	26
4.1.1.	用户管理	26

4.1.2.	个人信息管理	27
4.1.3.	会话管理	27
4.1.4.	内容管理	29
4.1.5.	实时搜索	31
4.1.6.	关注管理	31
4.2.	非功能需求	32
4.3.	运行环境	33
4.3.1.	服务器的运行环境	33
4.3.2.	客户端开发环境	33
第五章	设计与实现	35
5.1.	客户端概要设计	35
5.1.1.	整体架构介绍	35
5.1.2.	目录组织结构	36
5.1.3.	客户端结构规范	37
5.2.	客户端的设计与实现	37
5.2.1.	用户管理模块	38
5.2.2.	个人信息管理模块	40
5.2.3.	会话管理模块	42
5.2.4.	内容管理模块	46
5.2.5.	实时搜索模块	53
5.2.6.	关注管理模块	55
第六章	系统测试	58
6.1.	系统测试环境	58
6.2.	系统单元测试	58
6.2.1.	用户管理模块	59
6.2.2.	个人信息管理模块	60
6.2.3.	会话管理模块	60
6.2.4.	内容管理模块	61
6.2.5.	实时搜索模块	62
6.2.6.	关注管理模块	62
6.3.	系统集成测试	63
6.4.	测试结果分析	68
第七章	结束语	69
7.1.	全文总结	69
7.2.	未来工作展望	69
7.3.	研究生期间工作	69
参考文献	71
致 谢	72
攻读学位期间发表的学术论文目录	73

第一章 绪论

1.1. 课题背景

社会化搜索 (Social Search), 是用户在提交查询的时候把用户的社交关系图 (Social Graph) 考虑进来的一种搜索行为, 它区别于传统的搜索引擎, 传统的搜索引擎都是依靠机器学习的算法, 只分析网页的连接结构或者文档的文本结构^[1]。社会化搜索的目的在于通过用户的社会关系图来改善搜索结果, 令搜索的结果和用户的兴趣更加匹配, 从而提供更好的搜索服务和体验。社会化搜索以很多种形式出现, 从分享书签、增加描述性标签到结合人工智能的算法, 在很多地方都可以看到社会化搜索技术的相关应用。传统的搜索引擎, 如 Google、Yahoo 和 Bing 都开始融入社交关系, Google 的 Google Reader 和最新的 Google Plus 应用都充分利用了好友的分享行为, 从而为用户的搜索提供精准推荐, Bing 结合了 Facebook 中丰富的社交关系, 当用户在搜索相关的人或者内容时, Bing 可以把从 Facebook 获取到的好友信息呈现到搜索结果的前边, 让用户最快地找到自己感兴趣的人或者内容。

传统的问答网站已经成为人们寻求帮助和分享经验的重要地方, 在这里, 人们可以提出问题, 等待着其他用户看到这个问题, 如果看到的用户正好擅长该问题, 并且又乐于分享的话, 这个问题就可能被解决。像 Yahoo! Answers 和百度知道, 都是典型的问答社区, 这里已经积累了很多问题和答案, 用户往往可以通过搜索就能找到自己想要解决的问题。但是在传统的问答网站里, 人们往往只是来寻求帮助的, 这些问题经常被热心的志愿者回答; 这样就带来了一些问题, 例如问题回答的周期比较长, 答案不够专业等, 因为很多热心的用户只是为了拿到一些激励的分数才回答问题的。

因此, 为了解决这些问题, 越来越多全新的问答式社会化搜索引擎技术和服务开始出现。Aardvark, 一个基于问答的社会化搜索系统, 在用户提问之后, 后台会分析问题所属于的话题, 从而能够找到更加专业的人, 同时后台还会考虑提问者的社交圈, 找到和用户最亲密的人, 综合考虑这两个因素, 快速地找到一些最适合回答这个问题的人, 从而使得问题在社交网络里更快速、更准确地传播, 答案也更加专业。Quora 的出现更加颠覆了人们对传统搜索的认知, 它把强大的社交关系完美地融合到了问答网站中, 用户和用户之间, 用户和话题之间, 用户和问题之间, 都通过“关注”关系紧密地联系在一起, 让人们在问答中不仅仅获得知识和分享经验, 同时也可以结识感兴趣的人, 扩展社交圈子和知识面。在国内, 也开始出现类似的一些服务, 比如“知乎”和开源的“者也”等, 它们也都开始把社交关系融入到问答网站中, 并且通过名人效应聚合成了更加

专业化的社区，在这里人们往往会讨论一些最流行的话题，因此增加了用户的黏性，使得用户更加乐意来关注最新的讨论。和传统的问答网站相比，新型的问答式社会化搜索社区更强调人的参与和分享，让整个社区的知识传播地更加快捷。

本课题基于“上下文感知的社会化搜索系统”项目，研究并实现一个问答式社会化搜索系统。该系统以体现社会化搜索技术的基本概念为出发点，通过强大的社交元素，提供给用户更加丰富的内容和精准结果，从而给用户带来更好的用户体验。

1.2. 课题主要研究内容

问答式社会化搜索系统是在传统问答系统的基础上，融入了社会化的元素，把问题快速地分发到最适合回答的人那里。人们在这里不仅仅可以寻求帮助和分享经验，还可以关注一些流行的话题，感兴趣的用户，甚至是特定的问题，通过这些关注关系，整个社区被紧密地结合起来。人们除了能够及时地获取到问题的答案之外，还可以看到那些和流行话题相关的问题，好友的动态信息和关注的特定问题的最新进展，通过这些和用户紧密相关的内容，极大的扩展了用户的社交圈和知识面，进而也能够影响到社区里边其他的用户。

问答式社会化搜索系统是完全开放的，它结合了目前常用的社交网络服务和微博客应用，用户可以通过这些站点快速注册或者登录到系统中，同时还可以把自己的社会关系图导入进来，从而快速地找到感兴趣的信息。系统还提供了强大的实时搜索系统，可以在用户搜索时，动态的给用户返回最匹配的信息，包括可能感兴趣的人，最新的流行趋势，以及最热门的相关问题。

本系统还结合了 Wiki 的特点，人们可以贡献自己的知识，通过编辑相关信息和互相协作，极大地调动了用户的积极性，使得用户在这里有一种归属感，不仅仅是寻求帮助和分享经验，而是在维护社区健康成长，使得整个社区能够快速地丰富起来。同时为了激励用户能够更好的分享经验，系统采用了用户投票的机制来评价答案的质量，用户可以对自己满意的答案进行投票，获得赞成票多的答案往往能够显示在问题页面的前边，从而让更多的用户看到该答案。

在本课题的研究中，论文作者发现通过综合考虑用户在网络中的重要性和活跃度，以及用户之间的连接结构可以更好地评估用户在某个领域里的影响力。因此，本文提出了一个改进的 PageRank 算法模型^[2]来迭代计算用户的影响力排名，实验结果表明该模型可以有效地计算问答式社会化搜索系统中用户在某个领域里的影响力。

在项目期间，论文作者对上述研究内容进行了相关调研，并且分析了系统需求，然后设计系统架构设计，并对客户端技术方案进行了详细地调研和研究，参与了系统中各个模块的详细设计和实现，并且负责系统的部署和联调测试。

1.3. 主要工作内容

本课题研究中，论文作者主要负责的工作内容如下：

1. 调研社会化搜索相关技术和实现方案，搭建系统的开发环境和部署环境
2. 参与系统的整体需求分析和各个模块的需求分析
3. 参与系统整体架构设计和各个子模块的通信接口定义
4. 负责客户端主要模块的设计和编码实现
5. 参与服务器端部分模块的设计工作
6. 负责系统部署工作，编写测试用例，进行单元测试和集成测试
7. 负责部分项目文档的撰写工作，包括需求分析，系统设计和详细设计等文档

1.4. 论文结构

本论文的结构安排如下：

- | | |
|-----|--|
| 第一章 | 绪论，介绍本论文的课题背景，课题主要研究内容和作者的工作内容等。 |
| 第二章 | 相关技术，讲述社会化搜索的相关理论，主要包括了传统搜索的社会化属性和新型社会化搜索技术两部分，同时介绍了客户端开发的相关技术，重点介绍本系统采用该技术方案的原因。 |
| 第三章 | 需求分析与相关算法介绍，首先介绍了系统的需求分析，然后介绍基于特定话题的用户影响力的计算模型，重点介绍了社会化搜索社区的特点，同时提出了一个改进的模型来衡量用户的影响力；同时介绍了获取这些数据的方法。 |
| 第四章 | 问答式社会化搜索系统的设计与实现，讲述了系统的整体设计方案，最后对各个模块的实现方案进行逐一阐述。 |
| 第五章 | 系统测试，介绍了系统测试环境，针对各个模块进行单元测试的方法，以及系统的集成测试和运行效果。 |
| 第六章 | 对论文的工作进行了总结，并且介绍了作者在研究生期间的工作成果，最后提出了未来工作的展望。 |

第二章 社会化搜索相关技术

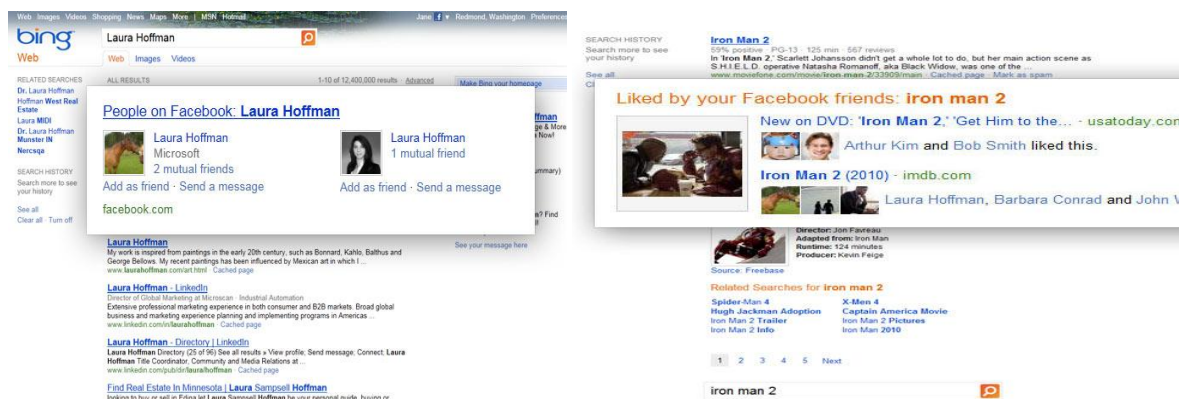
2.1. 社会化搜索理论

2.1.1. 传统搜索的社会化属性

传统的搜索引擎已经不再是用户查询关键词，搜索引擎返回查询结果这样被动地搜索了，传统搜索引擎也开始融入社会化元素，利用用户的查询习惯和好友圈子给用户返回更加匹配的结果。Google 早已经开始从用户的社交圈中来匹配用户的查询结果，如图 2-1 所示，社交圈主要包括了 Gmail 中的 Google Contacts 联系人，Twitter 以及 Google Buzz 中的粉丝和好友，还包括这些好友的好友，也就是二度好友；Google 从好友的网站、博客以及公开的 Profile，好友的状态更新、微博以及评论，还包括 Picasa Web 上的照片和 Google 阅读器上的订阅文章获取内容信息。

图 2-1 Google 的社会化属性^[3]

Bing 搜索引擎更是和全球最大的社交网站 Facebook 合作，如下图所示，用户可以直接在搜索引擎上查询到 Facebook 上的相关好友，以及跟好友相关的内容，搜索引擎和社交网站的结合，一方面可以让用户在搜索的时候搜索到好友的信息，另一方面也会给社交网站带来流量，帮助用户结识有共同兴趣的朋友。

图 2-2 Bing 搜索引擎的社会化属性^[4]

这些新的社会化元素都表明社会化搜索已经成为未来搜索的流行趋势，具有广泛的应用前景。

2.1.2. 新型社会化搜索技术

目前越来越多的新型社会化搜索网站出现在人们的视野中，其中和问答结合的问答式社会化搜索网站更是人们关注的重点。2010 年 2 月，谷歌以 5000 万美元收购了 Aardvark，一个全新的社交问答网站，如下图所示，在这里，人们可以通过 Email、IM、短信和 Web 等方式进行提问，用户的问题会通过后台的社会化搜索引擎转发给最适合回答该问题的一些人那里，这些用户也可以通过 Email、IM 等方式来解答问题；在 Aardvark 中，一切的操作都围绕着提问和回答进行，整个系统非常重视回答的及时性，所以提供了手机客户端的支持，以保证人们能够随时随地的提问和回答；Aardvark 最大的特色就是后台的路由引擎能够帮助用户找到合适的人来解决问题，它通过专业的问题分析找到问题所属的话题列表，然后综合考虑擅长该问题的用户列表和用户的好友列表，来找到最适合回答的一些人。

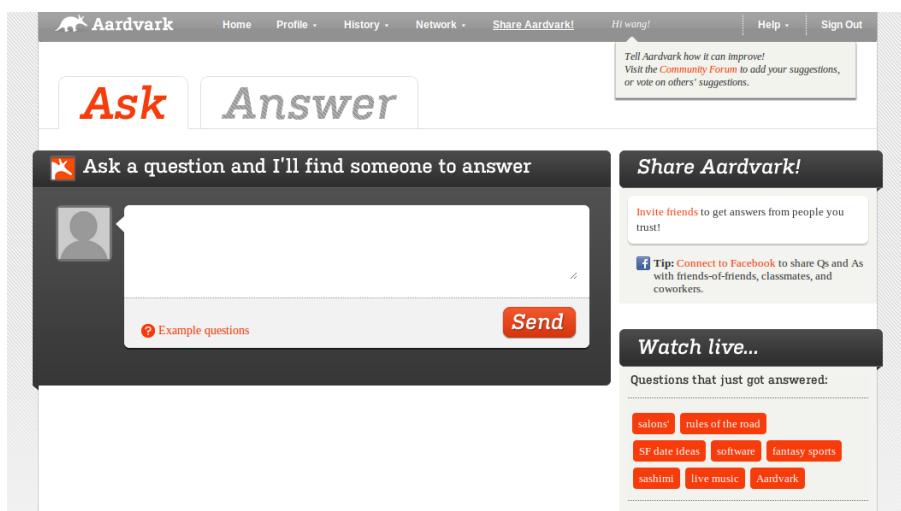
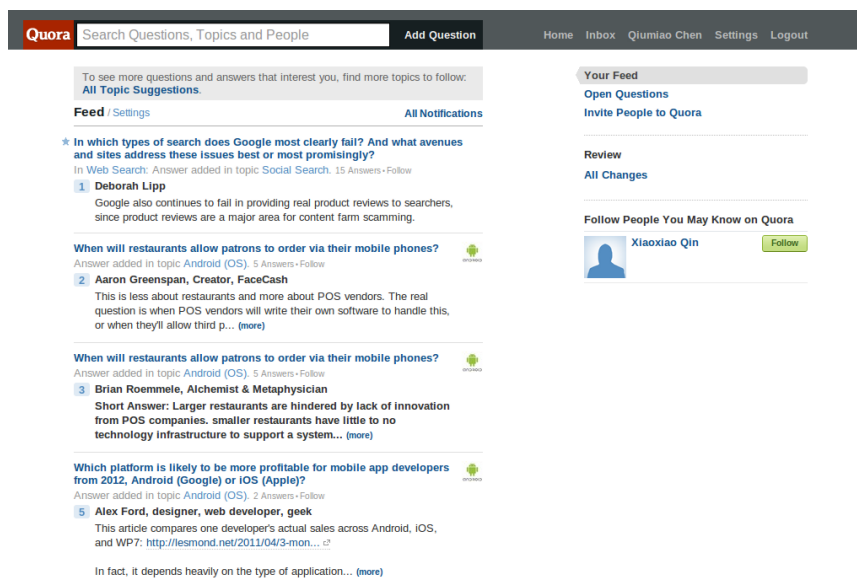


图 2-3 Aardvark 网站^[5]

Quora 是另外一家问答式社交网站，如下图所示，它除了帮助用户解决问题之外，更加重视用户之间的社交关系，用户可以通过关注关系关注话题，人，以及问题。相比 Aardvark 来说，Quora 给了用户更多的选择，使得用户能够获得自己感兴趣的内容；它通过评价机制来判断回答的质量，可以让有用的回答排名靠前，让没有意义的回答排在后边，其搜索话题、问题和用户时的界面交互，以及推送内容的精准性都是值得借鉴的。

图 2-4 Quora 网站^[6]

当前国内也掀起了一阵复制浪潮，从国内第一家社交网站问答系统“啊烦题”，到被业界看好的“知乎”，以及开源的“者也”，社会化问答网站也成为越来越被国人关注的领域，发展前景非常广阔。

2.2. 客户端开发技术

2.2.1. JSP 技术

JSP (JavaServer Pages) ^[7] 是一种动态网页技术，它是基于 HTML、XML 和其它文档的动态生成网页的 Java 技术。本系统之所以选择该技术是因为它有如下的特点：

- 内容和显示分离，和 PHP 以及 ASP 不同的是，JSP 可以使用 HTML 和 XML 来设计页面，使用 JSP 标识来产生动态内容，整个逻辑部分有 JavaBeans 封装，网页的设计人员可以修改页面结构，不影响内容的产生；服务器端通过解析 JSP 来产生内容，并封装成 HTML 或者 XML 返回给客户端。
- 完全使用 Java 开发，服务器端在解析请求的时候，可以使用 Java 提供的框架

来开发，提升了系统的兼容性和易用性。

现在越来越多的大型站点开始摒弃 PHP，而采用 JSP 技术来进行开发，JSP 强大的技术支持和论坛帮助也是本系统选择该技术的原因之一。

2.2.2. HTML 和 CSS 技术

HTML 和 CSS 技术是 Web 开发人员所必备的语言，本系统在开发的时候确保结构与表现形式分离，页面结构内容放在 HTML 文档中，尽量最少的使用样式表，具体的样式表都专门放到以 CSS 结尾的文件中。同时为了保证浏览器的兼容性，系统在设计 HTML 规则的时候采用了语法更严格 XHTML（Extensible HyperText Markup Language，可扩展超文本置标语言），它是基于 HTML 4.0 改进的新语言，具有很好的扩展性。本系统在使用 CSS 的时候，应用了很多 CSS3 的最新技术，在保证兼容性的同时，带来最好的用户体验，比如圆角图片、背景模糊等。

2.2.3. JQuery 技术

JQuery 技术^[8]是 JavaScript 的使用封装，大大简化了 JavaScript 的开发成本。JQuery 提供了强大的选择器，可以用最少的代码获取到页面的各种元素，链式操作可以对一个对象的各种操作连写，不需要重复地获取对象；它还对 DOM 进行了封装，替代了原来复杂的 JS 操作；JQuery 在 Web 开发中一个最主要的特点就是提供了丰富的界面效果和可靠的事件机制，比如页面的渐隐、下拉和闪亮效果等；另外 JQuery 对 AJAX^[9]进行了封装，可以方便地调用；对各大浏览器的兼容性也是本系统选用 JQuery 技术的一个重要原因，开发人员可以使用一些 JQuery 提供的动画效果而不用考虑兼容性问题；JQuery 进一步确保了页面结构和行为相分离，便于维护。

本系统中，所有的动态交互部分都采用 JQuery 代码来实现，并且论文作者在开发过程中制定了一套类似于面向对象的机制来保证开发的规范性，一方面可以让系统的功能模块化，另外一方面也体现了封装的特性。

2.3. Web 通信技术

2.3.1. AJAX 技术

AJAX(Asynchronous JavaScript and XML)的全称是异步的 JavaScript 和 XML，AJAX 确保了网页不需要刷新就可以和后台交互，给前端开发人员带来了更加丰富的空间，提升了用户体验。浏览器只要支持 JavaScript，就可以使用 AJAX 通信，无需要插件的支持，用户操作和服务器通信可以异步化，减少了服务器的压力。本系统在很多地方都用

到了 AJAX 技术，比如“获取更多”，切换标签页等，AJAX 技术可以确保用户停留在本页的同时，获取到新的内容。

系统在设计中尽量使用 AJAX 的通信方式，减少了每次数据传输的字节流大小，并且提升了用户体验，让用户能够感受到 Web 2.0 带来的新特性和功能。

2.3.2. JSON 标准

JSON(JavaScript Object Notation)是一种轻量级的数据通信格式^[10]，适合用于 Web 开发中前后台的通信格式定义，易于解析和生成。目前各大主流编程语言都提供了 JSON 的解析和生成库，包括 C、C++、Java、Lisp、Matlab、PHP、SQL、JavaScript、Ruby 和 QT 等等。它便于前台解析，数据体积小，传输快，在客户端是一个 JavaScript 对象，不需要创建 DOM 节点。

本系统几乎所有的客户端和后台的通信都采用了 JSON 的格式，让系统能够统一起来，另外也能够保证系统的可移植性，当开发手机客户端系统的时候，可以完全使用制定好的通信接口，只需要在客户端重新解析一下就可以了。JSON 目前被越来越多的开发人员使用，也是未来的流行趋势，强大的技术支持也是本系统采用该标准的一个原因。

第三章 基于社会化搜索的用户影响力模型研究

3.1. 研究意义

通过对基于社会化搜索用户影响力模型的研究,一方面可以对目前流行的社会化问答网站的页面结构和用户数据进行分析,从而可以指导帮助我们进行系统整体设计,包括整个网站的组织结构,还包括不同页面的一些布局方案等等;另外一方面,通过计算用户在不同领域的影响力,也为系统后台的推荐机制、问题转发机制以及页面的答案排列顺序等提供了有价值的指导意义。

社会化问答服务是一个新兴的知识社区,该社区结合了传统的问答网站的特点,用户可以在这里寻求帮助、分享经验,所有用户都可以搜索查看这些信息。通过研究特定领域中的用户影响力评估方法,不仅仅可以帮助我们设计整个问答式社会化搜索系统的基本框架,还会得到如下的好处:

- 改善搜索结果,根据回答用户的影响力情况,把影响力更高的用户的内容呈现给用户。
- 随着用户数目的持续增产和问答质量的提高,市场潜力也越来越大,企业可以通过有影响力的用户来进行更有效的市场推广。
- 帮助提问的用户找到相应领域的专家^[11]。
- 随着整个社区的积累越来越丰富,为分类整理这些内容提供了依据。

3.2. 模型背景

文档的重要性^[12]一直是人们研究的重点。在问答网站中,可以通过静态地分析非文字特性来预测文档的质量^[13],比如说答案长度,点击量,回答的总数等等。在一些网站里,用户通过对问题页面中不同答案进行评价来改善页面的布局,好的答案往往排列在前边,这样就可以让用户更快地找到最佳的答案。在社会化问答社区,用户之间的连接结构是很重要的, PageRank 算法可以静态地计算网页的重要性,而基于特定话题相关的 PageRank 算法^[14],可以更加准确地计算网页在某些领域里的重要程度。在微博客^[15]中,用户之间通过关注关系连接起来,并且用户之间具有 homophily 的特性^[16],粉丝多的用户好友往往也多,并且用户在关注一个好友的时候往往是对他的发言很感兴趣。但是在 Quora 这种新兴的社会化问答社区里边,寻求帮助,获取感兴趣的经验才是最重要的,通过对 Quora 的用户数据进行爬取、分析和建模,可以通过用户的专业经验和连接结构来综合评估用户的影响力。

3.3. 爬虫设计

通过对一些规模较大的社交网站进行数据爬取，可以帮助分析算法的合理性，同时也可以对已有算法进行完善，本小节重点介绍了论文作者在研究生期间设计和实现的一些数据爬取的方法，通过分析社会化站点中用户数据的分布情况和隐私结构，利用脚本语言快速地搭建爬虫框架，实现了对多个社会化站点的数据获取。

3.3.1. 爬虫背景和相关技术

整个 WWW (World Wide Web) 的页面有两部分构成: Surface Web 和 Deep Web (或者叫做 Invisible Web, 或者叫做 Hidden Web)^[17]。前者通过 HTML 标签互相连接起来, 可以被传统的搜索引擎收录和索引; 后者 Deep Web 主要存在于动态生成的网站, 传统的搜索引擎没有办法发现这些内容, 因为它们往往是基于一个特定的查询动态生成的, 可以分为以下两类:

1. 私有 Web, 往往需要注册和登录才能看到。
2. 动态内容, 这些动态页面往往需要用户执行某一个操作之后才会产生。

3.3.2. 爬虫架构

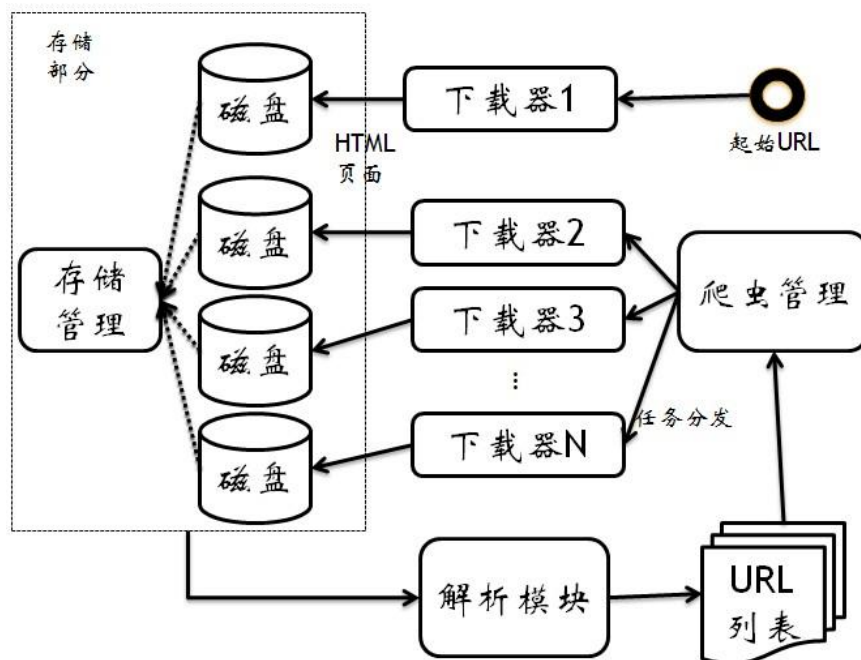


图 3-1 爬虫框架图

定向域名的爬虫也像传统的爬虫一样, 从一个选好的初始 URL 开始爬取, 下载页面, 然后对页面进行保存解析, 图 3-1 展示了爬虫的基本框架, 接下来详细的分析各个

模块:

1. 下载模块, 该模块是获取 Deep Web 信息的核心部分, 获取 Deep Web 内容的方法会应用到各个下载器中, 下载器可以通过浏览器已保存的 cookie 信息跳过登录和验证的过程, 直接获取到 HTML 页面, 并且通过模拟用户交互动作向浏览器发送 AJAX 请求, 从而获取到 Deep Web 的信息, 之后会把整个 HTML 页面保存到本地磁盘上。
2. 存储管理, 整个系统采用分布式管理, 各个下载模块分散在不同的物理机和虚拟机中, 采用分布式管理一方面可以对特定域名的网站并行爬取, 加快数据收集的速度, 另外一方面也可以避免同一个 IP 地址对网站的多次访问而被封禁。下载的 HTML 页面按照一定的命名规则, 以文件的形式保存到磁盘上; 之后解析模块会根据相应的命名规则来访问这些文件。
3. 解析模块, 主要负责分析 HTML 页面, 定位到相应的 URL 连接。Ruby 脚本语言提供了分析 HTML 页面的相应模块, 可以高效准确的定位到相应的 URL 中, 它采用了 DOM (Document Object Model) 来组织各个节点, 同时可以根据相应的 CSS 来筛选各个节点, 这也是选择 Ruby 语言的重要原因。
4. 爬虫管理, 该模块是整个框架的核心模块, 它负责任务的调度和分配, 并且决定接下来要爬取的信息, 同时也会对下载模块进行监控, 当发现某一台物理机或者虚拟机的下载模块出现问题的时候, 进行相应的操作, 比如下载模块因为超时终止时, 可以重启下载模块, 以保证系统的健壮性。

3.3.3. Quora 爬虫

现在越来越多的网站是需要登录之后才可以访问的, 比如说目前流行的社交网站“人人网”和微博客“新浪微博”等, 都是需要登录之后才能看到用户的详细信息。解决这个问题的办法有两个: (1) 分析登录过程的 HTTP 请求, 通过浏览器的插件 (如 Firefox 的 Httpfox 等) 进行抓包, 从而分析登录过程的具体请求, 然后在程序中模拟登录过程进行验证过程; (2) 在脚本语言中直接调用浏览器来执行页面请求, 因为浏览器可以保存用户登录的 cookie 信息, 所以可以跳过登录过程, 直接对页面进行访问, 本文采用后一种方法来解决私有页面的访问问题。

很多页面需要用户动态地交互才能获取到更多的信息, 因为现在越来越多的页面都是通过 AJAX 来动态请求内容的, 这样可以较少每次传输的数据, 同时还可以提升用户体验, 但是却给自动化的爬虫机制带来了很大的难题。我们同样通过在脚本语言中调用浏览器来动态的请求更多的信息, 通过分析页面的 HTML 结构发现, 按钮“MORE”是通过这样的一个动作执行的:

```
b.cell(:xpath, "//div[@class='pager_next action_button']").click
```

首先通过 CSS 定位到该 div 标签，然后执行 JavaScript 的 click 函数，浏览器通过发送特定的请求之后解析从服务器获得的数据，最后显示给用户。在 Ruby 中调用浏览器执行上述动作，需要首先安装 Watir 模块，同时在 Firefox 浏览器中要安装 JSSH 插件来解释具体的 JavaScript 动作。

Quora 爬虫的整个流程如图 3-2 所示：

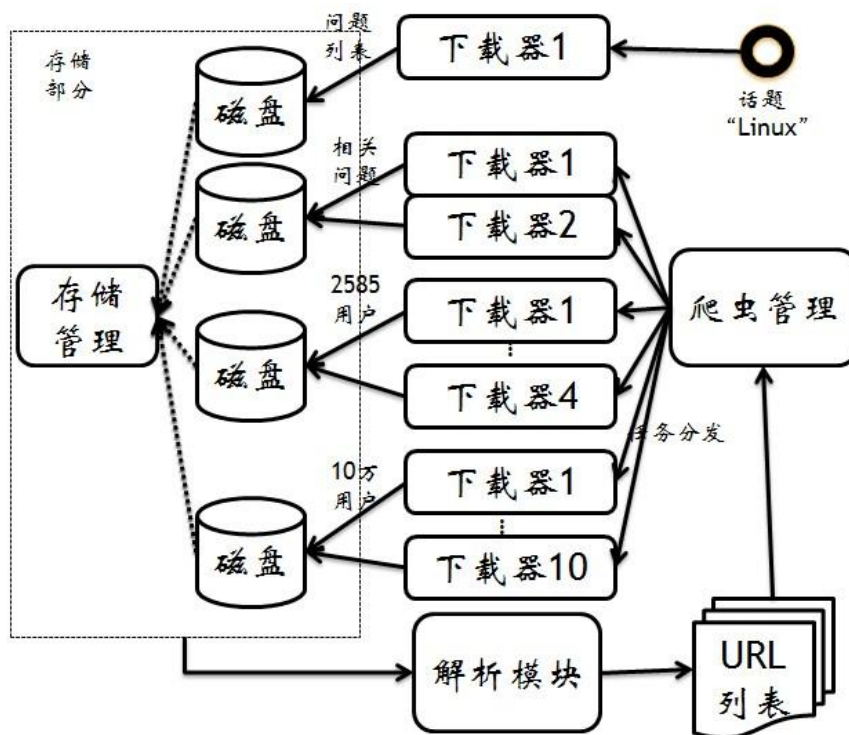


图 3-2 Quora 爬虫框架图

1. 选择一个话题，我们选择的是“Linux”，第一步爬取跟这个话题相关的所有问题列表，把这些列表保存下来，解析出相关的全部问题的 URL。
2. 爬虫管理模块把这些 URL 任务进行分割，分发给 2 个下载模块去下载全部的问题，其中需要获取一些动态的内容，解决方法如前所述。
3. 下载全部的问题之后，解析模块分析这些 HTML 文件，解析出所有相关的用户的 URL 信息，爬虫管理模块进行 URL 分割，交给 4 个下载模块去下载全部 2585 个用户的详细信息。
4. 解析模块对下载的用户详细信息分析，进一步提取出 102733 个用户的 URL，爬虫管理模块对任务进行分发，交给 10 个下载模块进行下载。
5. 存储管理模块对所有的文件进行分析，可以继续下载新的信息，也可以把所需的信息保存到 MySQL 数据库中。

整个爬虫环境在分布在校园网节点中，因为访问公网的带宽资源有限，为了不影响其他人的正常工作，爬虫的下载模块一般在晚上 12 点之后启动，白天会停止下载。每个下载模块在请求 HTML 页面时，以 10 秒的间隔进行请求，避免对该域名造成太大的

压力。三个晚上的时间总共爬取了 102733 个用户的个人信息，接下来又选取了其中的 2585 个用户，对其详细的个人信息进行了爬取，主要包括用户的粉丝和好友列表，以及用户关注的话题列表和用户的 Profile 信息。

因为采用了分布式管理和多下载模块，大大缩减了数据获取的时间，在爬取完所需要的数据之后，可以对全部文件重新分析，把需要的结果存储到数据库中。

具体实现：分为 Ruby 程序模块和 MySQL 程序模块，前者主要是负责爬虫的爬取和下载工作，后者负责分析下载下来的各种类型文件，保存到数据库中。

Ruby 程序模块：

表 3-1 Ruby 相关模块介绍

模块名称	输入	输出	功能	备注
(1) TopicQuestionCrawler.rb	话题名称，如 “Linux”	跟输入话题相关的问题列表原始 HTML 文件	爬取跟该话题相关的全部问题列表	需要联网
(2) TopicQuestionStats.rb	话题名称； 问题列表原始 HTML 文件	跟输入话题相关的问题列表 URL 文件	分析问题列表 HTML 文件，保存全部的问题列表 URL	离线分析，在 (1) 之后进行
(3) QuestionCrawler.rb	话题名称； 问题列表 URL 文件	(2) 中所有的具体问题的 HTML 文件	爬取跟该话题相关的所有具体问题	需要在 (2) 完成之后，需要联网
(4) QuestionStats.rb	话题名称； 所有具体问题 HTML 文件	全部相关用户 URL； 全部相关 Topic 的 URL	分析具体的问题，找到所有相关的用户和话题 URL	在 (3) 完成之后进行，离线分析
(5) UserCrawler.rb	话题名称； 相关用户的 URL 文件	(4) 中相关用户的具体信息 HTML 文件	爬取用户的具体信息，包括粉丝、好友和关注的 Topic 列表	在 (4) 之后进行，需要联网
(6) UserStats.rb	话题名称； 用户信息信息的所有 HTML 文件	跟 (5) 中用户相关的所有用户和话题的 URL 列表	分析所有二度好友和粉丝，以及关注的 Topic 列表，找到最大并集	在 (5) 之后进行，离线分析
(7) AllUserCrawler.rb	用户 URL 文件	用户的首页 HTML 文件	爬取用户的首页信息	在 (6) 之后进行，需要联网

MySQL 模块:

表 3-2 MySQL 相关模块介绍

模块名称	输入	输出	功能	备注
QuestionList.rb	话题名称; 问题列表 URL	question_list 表	把问题列表导入数据库中	使用自增编号
TopicList.rb	话题名称; 话题列表 URL	topic_list 表	把话题列表导入到数据库中	使用自增编号
UserList.rb	话题名称; 用户列表 URL	user_list 表	把用户列表导入到数据库中	使用自增编号
QuestionUser.rb	话题名称; 用户详细信息 HTML 文件	question_asker 表; question_replier 表; question_voter 表	把用户的提问、回答和投票关系导入数据库中	question_voter 表中投票者是给回答者投票的, 和问题无关
UserSocial.rb	用户的粉丝和好友关系 HTML 文件	user_social 表; user_topic 表	把用户的关注关系导入到数据库中	A 关注 B, A 是粉丝, B 是好友
UserHomeStats.rb	用户的首页 HTML 文件	user_profile 表	把用户的统计信息保存到数据库中	所有 102733 个用户

论文中提到的 Quora 爬虫架构更适合于定向爬取某个域名的内容, 整个架构中还有一些需要改善的地方, 比如说添加 DNS 解析模块来加快域名解析速度, 还可以对存储管理使用 NFS 统一管理, 这样可以无差异的存储文件, 减少多个分析模块获取文件的难度等。

3.3.4. 话题网络爬虫

话题网络爬虫的目标是通过爬取 Quora 和知乎网站的话题结构来把话题之间的映射关系建立起来, 一个话题可能有若干个父话题和子话题, 以 Quora 为例, 通过分析网页的结构, 可以发现它的话题组织主要在如下 URL 中体现:

```
http://www.quora.com/+”话题名”+/organize
```

分析 HTML 页面的结构, 可以提取出当前话题的所有父话题和子话题, 通过以下的链式结构来获取到:

```
page.css('div.row.section.p1').each do |section|
  title = section.css('strong').inner_text.split(' ')[0]
```



```

if title == "Parent"
    section.css('a.topic_name').each do |topic|
        pname.push topic.attribute('href').value.gsub('/', '')
        i = i + 1
    end
elsif title == "Child"
    section.css('a.topic_name').each do |topic|
        cname.push topic.attribute('href').value.gsub('/', '')
        j = j + 1
    end
end
end
end

```

把爬取到的父话题和子话题的映射关系保持到 `topic_relation` 表中，同时把没有出现过的话题保存到 `topic_profile` 表的末尾。

整个爬虫采用广度优先策略（BFS），从一个起始话题开始，比如“Linux”，按照前述的方法把没有出现过的话题保存下来，等待进一步地爬取，同时把该话题的父子关系保存下来；分析完该话题之后，把标志位 `visited` 置 1，继续找到第一个 `visited=0` 的话题爬取，直到所有的话题都爬取完毕为止。

“知乎”的话题网络爬虫也是类似的，只不过会有一些特殊情况需要处理，比如说登录次数过多时要输入验证码；并且访问的速度不能太快，避免帐号被封禁。

3.3.5. SNS 网站爬虫

SNS 网站爬虫的目标是中国最大的实名制网站人人网的 3g 站点，整个爬虫也采用广度优先策略进行，通过好友之间的关系来实现遍历。该爬虫采用 Python 语言实现，主要设计到的 Python 模块有：

- `urllib2` —— 发送和接受 http 请求
- `urllib` —— 把字符串转换成 http 请求中的参数格式
- `cookielib` —— 配合 `urllib2`，把 cookie 信息保存下来，这样后续的请求才能持续下去，而不是每次都登录
- `re` —— 正则表达式，分析网页结构

访问的链接主要有以下几类：

- 主页 —— `profile.do`
- 个人信息 —— `details.do`
- 访问足迹 —— `footprint.do`

- 好友列表 —— getfriendlist.do (全部好友要加上参数 f=all)
- 个人状态 —— getminifeed.do (第二页开始 f=0)
- 留言 —— status/replystatus.do

SNS 网站爬虫的第一步也是从登录开始的，因为社交网站都属于 Deep Web 网站，它们的内容不对传统搜索引擎开放，只有经过验证之后才能看到用户的信息。基于研究的目的，首先注册一个帐号，添加几个好友，这几个好友作为最开始的 URL，一方面遍历好友的详细信息，比如说主页、状态和留言等，另一方面遍历好友的好友的 URL，也就是二度好友，把这些好友作为进一步爬取的 URL 主线保存下来，同时也把好友之间的映射关系记录下来；采用广度优先策略进行遍历，整个实验的目标是把所有北邮的用户都爬取下来。实验分成了两个时间阶段，分别采用不同的起始点对全部北邮的学生进行遍历，发现两个实验中的结点个数几乎相等，也就是说整个北邮活跃用户的全集几乎是全连通的。

3.4. 数据集分析

本小节通过对 Quora 的用户数据集进行分析建模，一方面可以了解社会化站点 Quora 的用户特点，另一方面对接下来提到的基于特定话题的用户影响力模型的相关背景进行了介绍。这个数据集的准备时间是从 2011 年 10 月份到 12 月份，数据集总共包括了 102733 个用户的基本信息， $|U_a|=102733$ ，这些基本信息包括用户的提问和回答总数，用户的好友和粉丝个数等等，表 3-3 展示了 Quora 中这些用户的一些基本统计信息，从提问和回答数目来看，整个社区还处在积累阶段，同时也可以看出用户回答的数量几乎是提问数量的两倍，用户的平均粉丝数目和好友数目也几乎相等。在这些用户中我们选取了一个子集 U_t 进行分析， $|U_t|=2585$ ， $U_t \in U_a$ ，并进一步爬取了 U_t 集合中用户的详细信息，包括用户的所有问题和答案列表，用户关注的好友和粉丝列表等，这个子集中的用户都是限定在“Linux”领域中的，这个领域里的所有问题我们用 Q_t 表示， U_t 集合中所有用户关注的话题用 T 来表示。

表 3-3 Quora 数据集统计

	Maximum	Average	Standard Deviation
Questions	29264	2.9838	104.2275
Answers	3800	5.9808	39.0227
Followers	34023	94.1106	370.8265
Friends	64946	83.2404	498.8779

	Maximum	Average	Standard Deviation
Topics	6387	29.5143	67.5916

3.4.1. Bow tie 模型

Quora 的本质是一个问答系统，可以使用 Bow tie 模型^[18]来分析用户的问答情况和社区的特点。Bow tie 模型最早是用来分析 Web 网页之间的连接结构，在这个模型中所有 Web 页面被划分成了四部分：In, Core, Out 和 Others。我们把 Bow tie 模型应用到 U_i 集合上，其中最主要的节点是 Core 里的用户，这些用户构成的图形是强连通的（Strongly Connected Components），通过使用 Tarjan’s 强连通算法^[19]来计算这些用户的集合，Core 中的用户之间往往互相帮助，任意两个用户节点之间都可以通过问题线索连接起来。In 部分的用户经常提问问题，这些问题会被 Core 里的用户回答。Out 中的用户经常回答问题，特别是 Core 提问的问题。图 3-3 和表 3-4 列出了 Quora 中用户在问题线索中的分布情况，并且和 Zhang J.^[20]分析的传统问答社区进行了对比。

通过上边的数据可以发现 Quora 中 In 节点只占了不到 20%，而 Core 和 Out 却占了接近 60%，说明在 Quora 中提问的用户相对较少，大部分用户喜欢回答问题，参与讨论，并且 29.0% 的用户更乐于回答 Core 用户的问题。在传统的问答社区，像 Java Forum，更多的人是来寻求帮助的，只有少数的志愿者来回答问题，但是，在 Quora 中大家更加倾向于对一个问题的讨论，更加乐于分享自己的经验。

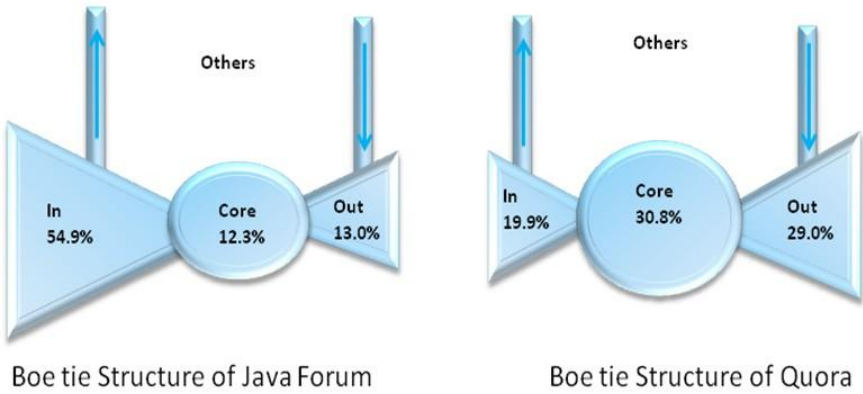


图 3-3 Java Forum 和 Quora 的 Bow tie 结构

表 3-4 Bow tie 比较图

	In	Core	Out	Others
Java Forum	54.9%	12.3%	13.0%	19.8%
Quora	19.9%	30.8%	29.0%	20.3%

3.4.2. Quora 分布

进一步分析 U_a 集合中的用户信息的分布情况，可以了解到用户的整体概况和偏好，

图 3-4 展示了用户在社区中的提问和回答的分布情况，它们都服从 Powerlaw 分布^[21]，从图 3-4 中还可以发现用户的提问数目很明显小于回答的数目，证明了 Quora 是一个偏向于讨论的社区，用户更希望通过提出少量的有价值的问题，让大多数人参与讨论和分享经验。从图 3-4 中还可以发现提问数和回答数目并没有很强的相关性，有的用户喜欢提问，却很少回答问题，这些用户往往希望来这里寻求帮助；但是更多的用户乐于参与讨论，很少提出问题；还有的用户不仅仅喜欢帮助别人，也喜欢提出一些有价值的问题。

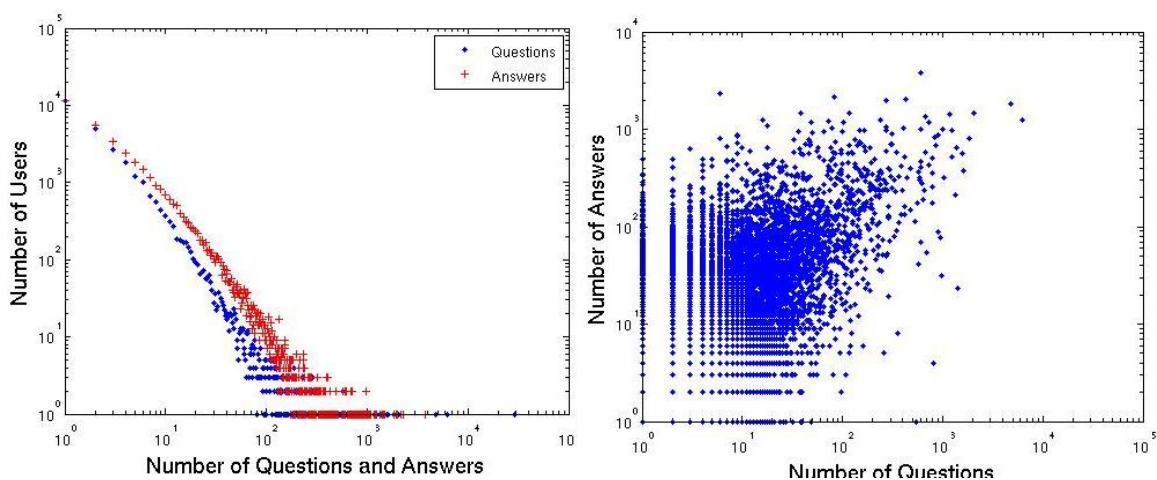


图 3-4 用户提问和回答的分布和散点图

图 3-5 展示了用户的社交关系情况，可以看出用户的粉丝和好友数目的分布除了满足 Powerlaw 分布外，还可以发现用户之间的相互关系，一个用户的粉丝数越多，他的好友数目也会越多，反之亦然。但是正如图 3-5 圆圈中所示，在 Quora 中也有一些用户拥有上万的粉丝，这些人大部分都是名人，比如互联网企业的创始人、出版业的创始人，或者是知名作家等。

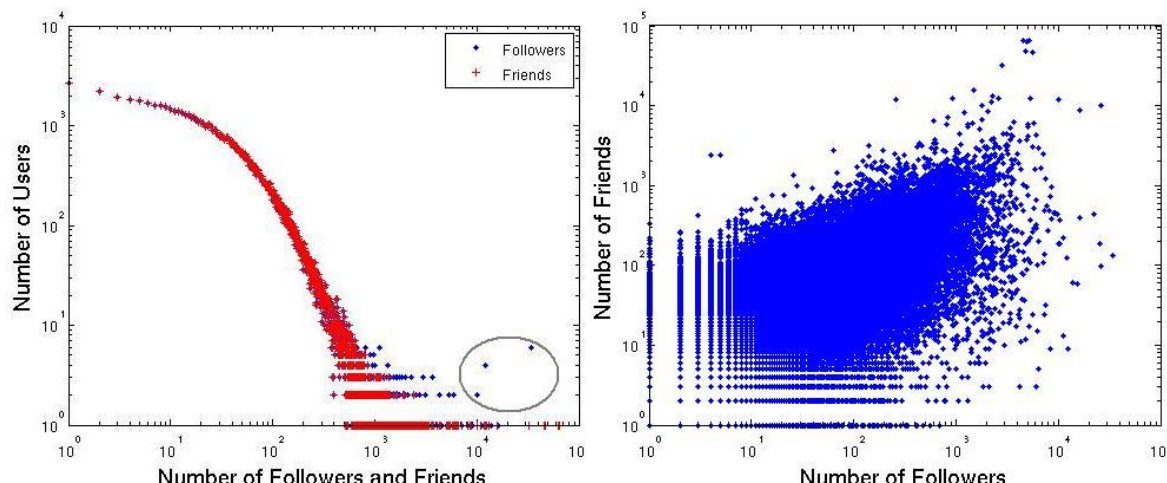


图 3-5 用户粉丝和好友的分布和散点图

3.5. 模型介绍

通过对爬取的数据集进行分析，图 3-6 给出了一个基于特定话题的用户影响力的计算模型，该模型的主要部分如下：问题网络的构建，社交网络的构建和排序引擎，问题网络的构建过程中可以获得用户的活跃度和重要性，社交网络的构建过程中可以获得用户之间的连接结构，通过用户的连接结构和活跃度可以计算出用户影响力的转移概率矩阵，通过转移概率矩阵的迭代和用户重要性作为个性化参数的影响，可以计算出在特定话题的用户影响力排名，接下来详细阐述模型中的主要部分。

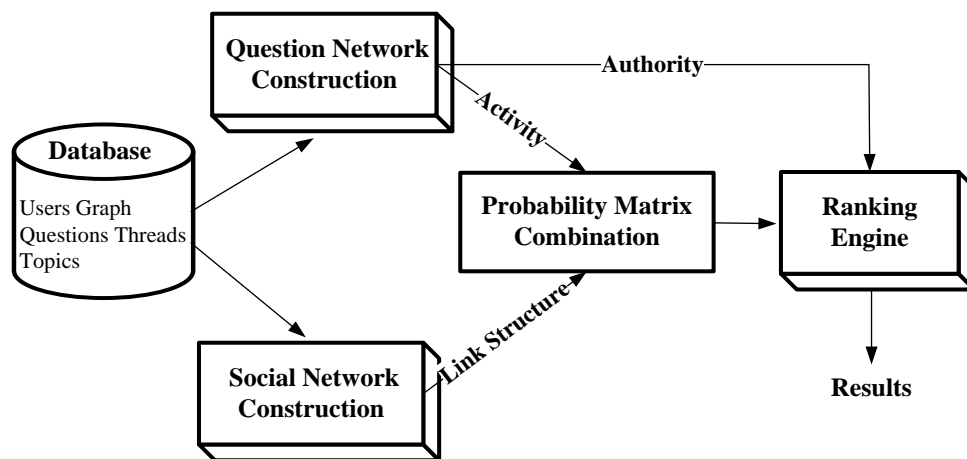


图 3-6 用户影响力计算模型图

3.5.1. 问题网络构建

在问答系统中，用户提出一个问题之后，就生成了一个新的问题线索，其他用户可

以在这个线索上对用户提出的问题进行搜索，还可以对已经有的回答进行投票评价；如图 3-7 所示，在该图中用户 A、C 和 F 分别提出了一个问题，用户 B 和 C 对 A 的问题进行了回答，用户 E 对 C 的问题进行了回答，同时用户 D 和 E 分别对 B 和 C 的答案进行了评价；当用户 B 回答了用户 A 的问题时，可以认为从 A 到 B 形成了一条有向边，同样的道理，用户 D 对用户 B 的答案进行了评价，那么从 D 到 B 也形成了一条有向边。根据用户的提问，回答和评价之间的关系，可以生成一个网络，称之为问题网络。

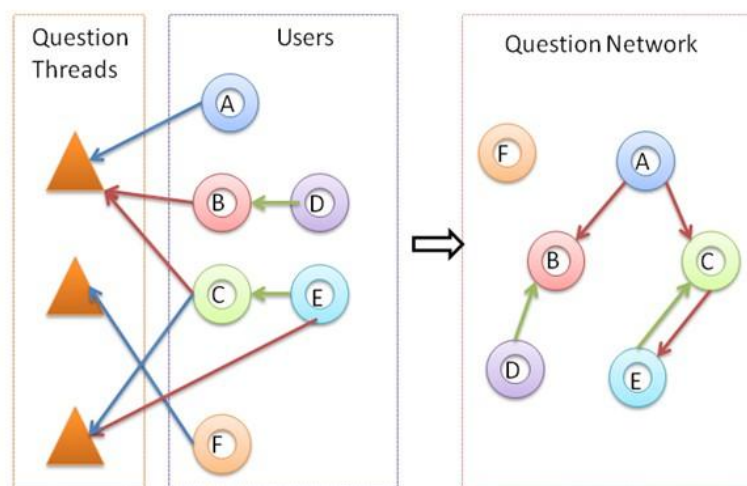


图 3-7 问题网络的构建过程

用户在问题网络中的重要性是衡量一个用户影响力的重要因素，主要包括三个方面：

- 提问的数量总和 $Ask(u)$ ；
- 回答的数量总和 $Answer(u)$ ；
- 获得赞成投票的总数 $Support(u)$ 。

如果一个用户在某一个领域经常提出一些有价值的问题，或者经常参与讨论和分享经验，那么这个用户可能是这个领域很感兴趣，也可能很擅长这个领域的知识。另外用户的投票是衡量一个答案是否有价值，进而证明回答者有经验的一个很重要的指标，在 Quora 中，用户的投票尤其重要，在一个问题页面，获得投票数越多的答案，可以显示在越前边，同时当一个问题中同时出现多个有很多投票的答案时，这个问题也会从普通问题成为常见问题，根据上述分析，定义用户在问题网络中的重要性 $Authority(u)$ 如下：

$$Authority(u) = \alpha \cdot Ask(u) + \beta \cdot Answer(u) + \lambda \cdot Support(u) \quad \text{式 (3-1)}$$

其中参数 α 、 β 和 λ 分别表示了上述三种因素的重要程度，把这些因素线性加权起

来就可以得出用户在问题网络中的重要程度。

按照同样的方法可以定义用户的活跃度 $Activity(u)$ 如下：

$$Activity(u) = \alpha' \cdot Ask(u) + \beta' \cdot Answer(u) + \lambda' \cdot Vote(u) \quad \text{式 (3-2)}$$

其中 $Vote(u)$ 表示用户投票的数量，参数 α' 、 β' 和 λ' 分别表示了上述三种因素的重要程度，把这些因素线性加权起来就可以得出用户在问题网络中的活跃程度。

3.5.2. 社交网络构建

用户通过关注这种关系建立起来的特殊的社交网络也是影响用户影响力的一个重要因素。Quora 中，用户的首页会定期更新很多 Feed，这些 Feed 表明了用户的喜好，这些 Feed 的来源主要有两部分：

- 用户的好友 $Friend(u)$ ；
- 用户关注的话题列表 $T(u)$ 中出现的各种更新，包括新的问题和答案等。

用户关注的好友的一些活动，包括提问，回答和投票等往往可以影响到该用户，同样的道理，用户自己的一些活动也同样会影响到自己的粉丝。因此一个用户的粉丝数越多，那么他的影响力就可能越大。基于数据集 U_t 建立起一个特定领域 t 的有向图 $G(V, E)$ ， V 表示所有用户集合，在所有的边集合中， $e(i, j)$ 等于表示 u_j 是 u_i 的好友， u_i 是 u_j 的粉丝。在有向图 G 上边应用随机冲浪模型如下：在冲浪过程中每一个用户的影响力会以一定的概率沿着 E 中的一条边到达他的好友。我们定义的模型和传统的 PageRank 的区别在于，这个转移概率不仅仅考虑用户之间的连接结构，而且考虑用户的活跃程度。这样就构建起来了一个特定话题的网络关系。在话题 t 中，用户之间的转移概率矩阵 P_t 定义如下：

$$P_t(i, j) = \frac{Activity(j)}{\sum_{k \in B(i)} Activity(k)} \quad \text{式 (3-3)}$$

其中 $Activity(j)$ 表示 u_j 在问题网络的活跃度， $B(i)$ 表示 u_i 的所有好友的集合，

$\sum_{k \in B(i)} Activity(j)$ 表示 u_i 的所有好友的活跃度。

3.5.3. 排序引擎

通过式 (3-3) 得到了用户的转移概率矩阵 P_t ，就可以对 $G(V, E)$ 使用 PageRank 算法进行计算用户在特定领域的影响力 R_t ：

$$R_t = cP_t \times R_t + (1-c)E_t \quad \text{式 (3-4)}$$

其中参数 c 是一个 0 和 1 之间的阻尼系数，在这里设置为 0.85， s 是一个初始向量， E_t 是个性化的向量因子，可以通过设置 E_t 来影响计算的偏向性，定义如下：

$$E_t = Authority'(u) \quad \text{式 (3-5)}$$

$Authority'(u)$ 是对式 (3-1) 行正规化之后的结果，用户在某个特殊领域的问题网络中的重要性可以作为每一个用户的个性化因子，来影响计算的结果。具体的计算过程如下：

Algorithm: Compute Quora Topic-specific PageRank

Input:

1. the transition matrix P_t for topic t
2. the teleportation vector E_t for topic t

OutPut: the rank vector R_t

Procedure:

1. $R_0 \leftarrow s$
 2. loop:
 3. $R_{i+1} \leftarrow cP_t R_i$
 4. $R_{i+1} \leftarrow R_{i+1} + (1-c)E_t$
 5. $\delta = \| R_{i+1} - R_i \|$
 6. while $\delta > \varepsilon$
-

通过式 (3-4) 计算出来的是用户在某一个领域中的影响力，可以根据用户关注的话题列表 $T(u)$ 来计算用户的综合影响力：

$$Inf(i, T) = \sum_{t \in T(u)} c_t \cdot R_t(i) \quad \text{式 (3-6)}$$

$Inf(i, T)$ 表示用户的综合影响力，参数 c_t 是决定用户在各个领域里影响程度的一个权值。

3.6. 验证结果

这一节会通过实验来验证上述模型的正确性，一方面通过对该算法的计算结果进行说明来阐述合理性，另外一方面通过跟一些常用的用户影响力衡量方法进行比较，来验证该模型的性能和作用。

3.6.1. 排名结果分析

表 3-5 展示了通过式(3-4)计算出来的 U_i 集合中用户的影响力的排名, 其中 **Indegree** 表示用户的粉丝数, **Outdegree** 表示用户的好友数, **Authority** 是由式 (3-1) 计算出来的用户在问题网络中的重要性, **Activity** 是用户的活跃度。从中可以发现用户 3 和 4 有很高的 **Indegree** 值, 但是因为他的 **Authority** 值比较低, 所以他们并不是最有影响力的; 相反用户 6 在这个领域的活动比较频繁, 但是他的粉丝数相对来说比较少, 所以影响力也不是最高的。而用户 1 和 2 不仅仅在问题网络中处于重要地位, 粉丝数也都很多, 因此是最有影响力的用户。

表 3-5 “Linux”领域的用户影响力排名列表

Users	Rank Value	Indegree	Outdegree	Activity	Authority
1	0.0948	315	199	204	152
2	0.0367	119	135	286	190
3	0.0314	501	58	41	21
4	0.0300	463	241	48	38
5	0.0278	122	54	70	64
6	0.0273	79	45	104	1152
7	0.0241	50	5	122	106
8	0.0233	62	6	147	160

3.6.2. 常用计算方法比较

简称上述模型为 **QR**, 接下来和一些常用的用户影响力计算方法进行比较:

- **Question Degree**, 简称为 **QD**, 在问答社区中, 往往通过所帮助用户个数来衡量一个用户的影响力, **Quora** 本身也是采用这个方法来计算 **Top Answerers** 的;
- **Follower Degree**, 简称为 **FD**, 用户的粉丝数往往反映了一个用户的受欢迎程度, 在微博客中, 往往会采用这种方法来衡量用户的影响力;
- **Topic-sensitive PageRank**, 简称 **TP**, 根据用户的连接结构来计算用户的影响力,

并且考虑了用户之间的偏好不同，用户的偏好往往是通过这个用户是否关注了该领域来定义的，和它不同的是，QR 算法考虑了用户 in 问题网络中的重要性和用户的活跃程度。

通过使用 Spearman^[22]和 Kendall^[23]两种算法来比较不同的算法之间的相似程度，这两种算法都有一些缺点，Spearman 不能很好的处理那些影响力非常小的用户之间的相似度，因为它通过比较不同用户的先后顺序来计算两种算法的相似度的，由于很多的用户的影响力都比较小，很难比较相互之间的重要程度，Kendall 的算法也有一些问题，它不能考虑了用户之间的差别，因此在进行比较的时候可以增加 Top10 的用户以使结果更加准确。

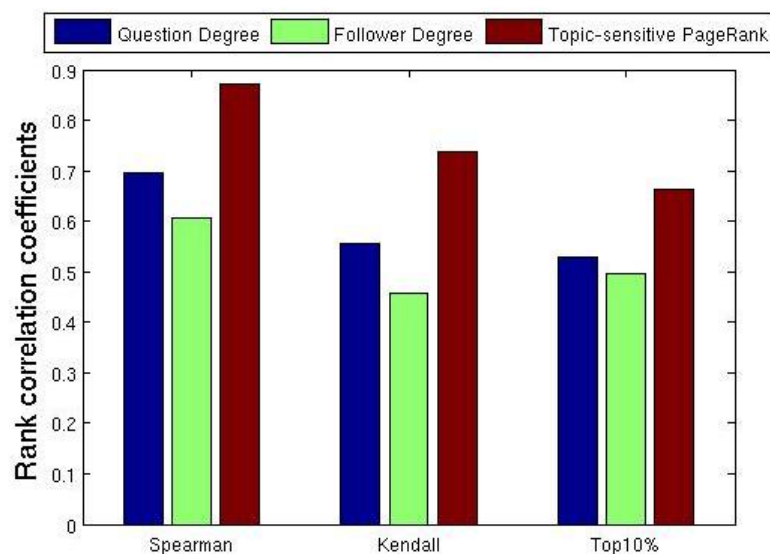


图 3-8 不同计算方法的排名相关度

上图展示了不同算法之间的相似度情况，可以看出用户粉丝数 FD 并不能很好的反映用户的影响力情况，因为名人来到这个社区之后，很快就会有大量的粉丝，但是他们可能并不是很活跃，也不经常参与讨论，或者是不擅长这个领域的知识；用户在问题网络中的重要性 QD 可以较好地衡量用户的影响力，不过也可能会出现一些用户只是比较热心，非常喜欢提问和回答一些新手的问题，但是他们并不是这个领域的专家；从表格中可以看出 TR 可以更好地反映用户的影响力情况，因为它不仅考虑了用户的连接结构，也考虑了用户关注的偏好，但是它没有更精确的考虑到用户在问题网络中的重要性和活跃程度。

3.6.3. 模型总结

在爬取目前最流行的社会化问答服务 Quora 的数据之后，通过对这些数据集进行了分析总结，发现各个用户在问题网络中的重要性有很大的不同，基于此提出了一种改进

的用户影响力的计算方法，该方法结合了社会化问答系统的特点，不仅仅考虑到了由问题线索构成的问题网络里边用户的重要程度，同时也考虑到了社会化问答系统中用户之间的连接结构和用户的活跃程度，使用经典的 **PageRank** 算法进行迭代计算用户的影响力。通过对最终的结果进行分析说明，并且和常用的一些计算方法进行比较之后，进一步验证了本文提出的模型 **QR** 的正确性。目前论文中提到的模型已经开始应用到实际的系统中，该模型不仅仅对整个社会化搜索系统的整体框架设计提供了依据，并且对完善系统的推荐系统和用户信息统计系统发挥了很大的作用。

第四章 需求分析

问答式社会化搜索系统是为用户提供内容服务的系统，一切功能都以用户为出发点，在这个系统中，用户不仅仅可以通过提问和回答进行交流，而且可以通过关注功能来获取到最新的信息，系统还给用户提供了实时搜索功能和推荐功能，这些功能可以帮助用户快速地找到需要的内容。本章通过功能需求、非功能需求和运行环境三部分来详细地描述本系统的相关需求。

4.1. 功能需求

功能需求是本系统中最重要的一部分，它直接关系到用户对整个系统的用户体验，本系统的客户端部分主要有以下模块构成：用户管理、个人信息管理、会话管理、内容管理、实时搜索和关注管理，接下来分别阐述各个模块的功能需求。

4.1.1. 用户管理

用户管理是任何一个社会化 Web 服务的基本功能之一，在问答式社会化搜索系统中，用户管理的作用就显得更加重要，作为一个问答网站，所提供的内容是允许匿名访问的，也就是说访客也可以看到所有的问题页面、用户页面和话题页面；但是只有注册的用户才可以提供内容，包括提问、回答、编辑等；本系统中还维护了用户的实时在线状态，提升了系统的实时性，比如当用户 A 看到用户 B 在线时，用户 A 就更乐意这时候直接向用户 B 提出一个问题，以便得到及时的反馈。

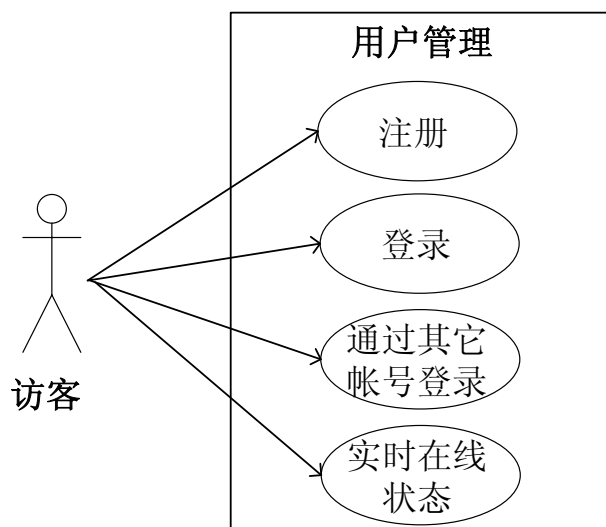


图 4-1 用户管理用例图

如图 4-1 所示，用户管理功能主要包括如下需求：

- 用户注册功能，可以在本系统注册一个帐号
- 直接登录，可以选择保存密码，下次可以跳过登录过程
- 通过其它帐号登录，比如新浪微博、腾讯微博等
- 实时在线状态维护，显示在各个页面的用户头像旁边

4.1.2. 个人信息管理

个人信息管理是系统的一个必要功能，用户通过该功能可以修改个人信息，修改密码，上传头像，通过完善个人信息可以让其他用户更加地了解自己，用户还可以申请更高的权限，比如说星级用户和管理员，具体的功能如图 4-2 所示：

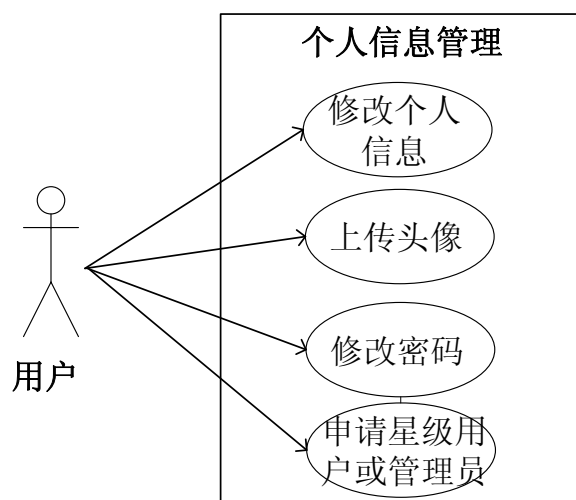


图 4-2 个人信息管理用例图

通过上图可以看出个人信息管理模块主要包括以下需求：

- 修改编辑个人信息
- 上传头像
- 修改密码
- 申请星级用户或者管理员

4.1.3. 会话管理

会话管理模块是整个系统最重要的部分，是用户创造内容最主要的入口，用户可以提出问题，回答别人的问题，评论一个问题，并且还可以对已有的答案进行评价，如下图所示：

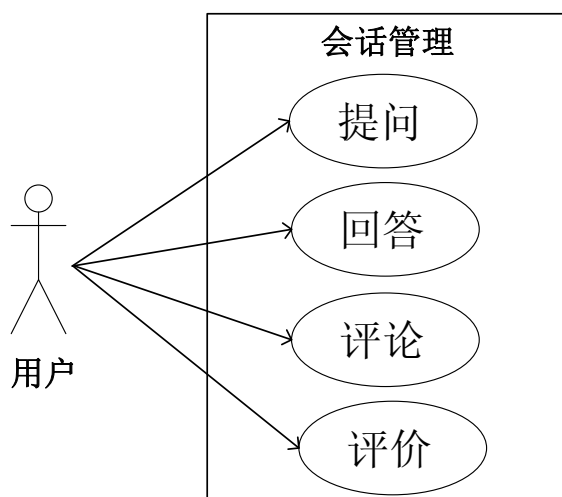


图 4-3 会话管理用例图

通过图 4-3 可以看出会话管理主要有以下需求：

- 提问功能是整个会话管理模块的最重要的功能，因为提问一方面是问答系统里必备的功能，另外一方面也是用户创造内容的最重要的入口。在问答式社会化搜索系统中，提问的时候不仅仅是简单地提出一个问题，用户还可以给这个问题选择上合适的话题，帮助问题进行分类，也方便系统把问题自动推送给最擅长这个话题的人。
- 回答功能是用户帮助别人解决问题，或者是分享经验的重要入口，在问答问题的时候，用户不仅仅可以输入自己的内容，还可以对这些内容进行富文本地编辑，在本系统中提供了如下的编辑功能：加粗、倾斜、下划线、删除线、引用、有序和无序列表等等，通过这些丰富的富文本功能，可以让用户的答案更加有条理。
- 评论功能可以让其他用户和回答者进一步交流，用户可以对这个答案表示感谢，也可以对答案的某些内容提出质疑。但是评论的内容默认是不显示出来的，因为在问答系统中，问题和答案才是整个系统最重要的部分，也是最应该呈现给用户的。当用户对某个答案想进行更多地了解的时候，可以点击评论标签，这时详细的评论信息会以动态的方式呈现给用户，用户可以查看并且添加新的评论信息。
- 评价功能是整个系统中内容组织的重要部分，它让用户参与评价一个答案的质量，当用户觉得一个回答有道理，并且对自己有帮助的时候，他可以对答案投票，可以进行“顶”的操作，收到“顶”操作越多的答案可以显示在越前边，这种方法一方面可以组织答案的排列，让最好的答案呈现在前边，另一方面也可以激励用户去更好地回答问题，因为好的答案可以呈现在前边，可以帮助到更多的人。同样，当一个用户觉得某一个答案不符合问题的要求，或者对自己

和其他用户没有帮助的时候，可以通过“踩”的投票让这个答案显示在下边，避免其他用户浪费时间来浏览这个答案。

4.1.4. 内容管理

内容管理功能部分是呈现给用户的信息，包含用户的首页信息，用户页面信息，问题页面信息和话题页面信息，这些页面之间互相连接起来，共同构成了整个系统的内容呈现部分，如下图所示：

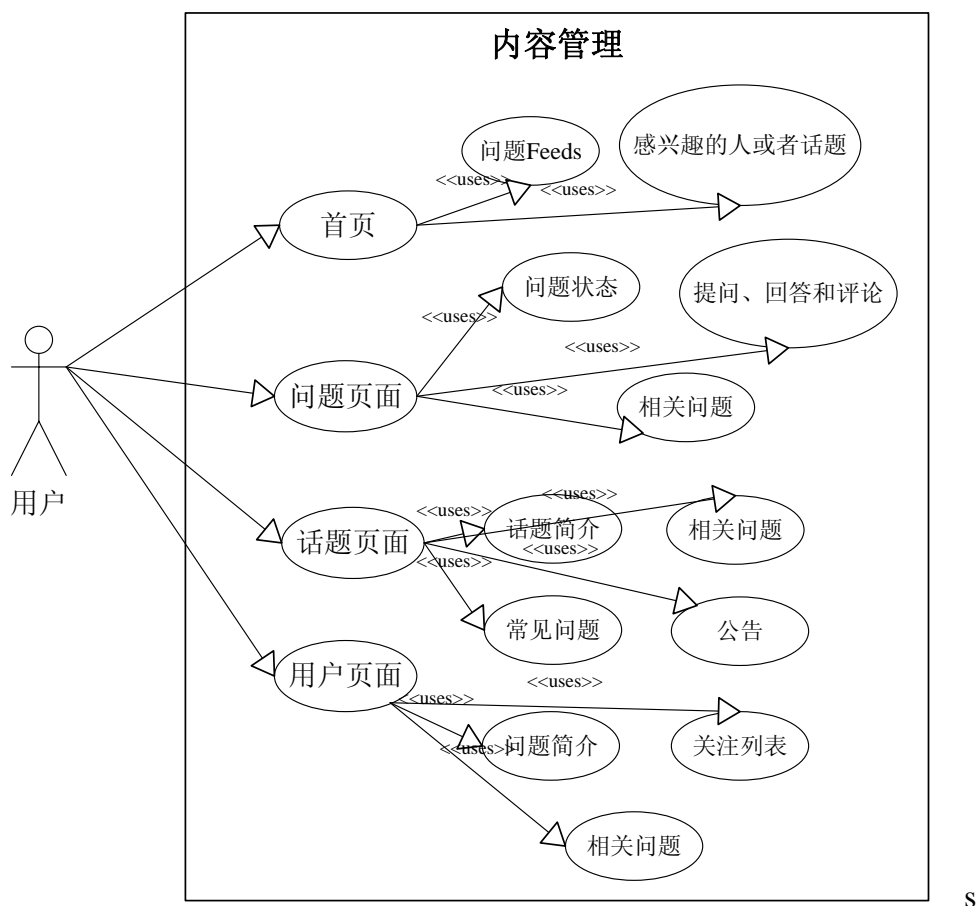


图 4-4 内容管理用例图

通过图 4-4 可以看出内容管理主要有以下需求：

用户首页是用户在登录之后呈现给用户的第一个页面，也是用户停留时间最长的页面之一，在这个页面是主要包含以下的信息：

- 问题 Feeds，这些 Feeds 主要分为两个部分：“我关注的问题”和“推荐给我的问题”，前者是用户关注过的问题，比如说用户提问的问题，和用户主动关注过的问题等，后者是系统推荐给用户的问题，这些问题可能是用户感兴趣的问题，也可能是用户的好友感兴趣的问题。当这些问题的状态有更新的时候，比如有新的答案，新的投票等等，这些问题就会推送到用户的首页，让用户能够及时

的了解到最新的信息。

- 可能感兴趣的人和话题，这是系统对用户的行为习惯进行学习和分析之后，推送给用户的，这个功能可以扩展用户的社交圈，也可以扩展用户的知识面，帮助用户去找到可能认识的人和感兴趣的话题。

问题页面是一个具体的问题线索的展现，在这个页面上主要展现了如下的信息：

- 问题、回答和评论，这是整个系统的最重要的信息，用户可以一目了然的看到想要的信息。在每一个用户头像的旁边会呈现这个用户的在线状态和用户的等级，用户等级分为普通用户、星级用户和管理员。不同权限的用户设置了不同的功能，一方面可以激励用户，另外也可以让系统避免受到 spam 的攻击，提升了系统的安全性。
- 相关问题，呈现了跟这个问题相关的其它问题的列表，这个功能可以扩展用户的知识面，提高用户在整个站点的停留时间。
- 问题状态，显示了这个问题的统计信息，包括创建者、关注人数和点击人数等等，这些信息一方面可以让用户了解到该问题的基本信息，另外还可以帮助系统提供更好地推荐和预测功能。

话题页面呈现了跟这个话题相关的一些信息，话题是整个系统组织结构的一个重要部分，每一个问题都会属于一个或者几个话题，因此话题页面主要显示了跟该话题相关的所有问题，比如一个用户关注了“宽带网中心”，他在“宽带网中心”这个话题页面里就可以浏览所有的相关问题。整个话题页面有如下部分组成：

- 话题简介，包括话题的图片、话题名称和话题的介绍，用户可以编辑这些内容，其中话题介绍部分是富文本编辑的，用户可以更好地组织内容。
- 相关问题，跟该话题相关的所有的普通问题，按照问题提问的顺序排列，最新的问题会呈现在最前边。
- 常见问题，管理员可以把一些普通问题标记为常见问题，常见问题往往是那些用户经常提问，并且有非常完善的答案的问题，这些问题往往会被很多用户浏览查看。
- 公告栏，管理员可以在特定的话题上发表相关的公告信息，这些信息都是具有一定的权威的，主要包括最新的通知信息，该话题的一些流行趋势等等。
- 话题关注列表，显示关注这个话题的一些用户。

用户页面是每个用户所有相关信息的呈现，包括用户参与的问题，用户的好友和粉丝，以及关注的话题等，具体分类如下：

- 用户简介，包括用户的头像，姓名和用户的介绍信息。
- 参与的问题，包括用户提问过的问题列表，用户回答过的问题列表和用户直接向别人提问的问题列表；这些信息都是用不同的标签页组织的，通过 AJAX 动态和后台交互动态获取到的。其他用户还可以在这个页面上向该用户进行提问，

提问的问题会显示在“问他/她”标签页中。

- 粉丝、好友和关注的话题列表,侧边栏会显示这个用户的粉丝列表和好友列表,以及关注的话题列表,同样也是通过 AJAX 动态的从后台获取到的,这样可以避免一次传输太多数据,并且可以使界面更加地简洁。

4.1.5. 实时搜索

当网站已经积累一定的内容,包括问答、用户和话题等之后,实时搜索可以让用户最快的搜索到想要的信息,还可以避免用户重复的提问问题。实时搜索功能几乎存在于系统的每一个页面,用户可以随时随地的搜索信息,用户在搜索框中输入关键字之后,后台可以动态的返回最匹配的信息,如下图所示:

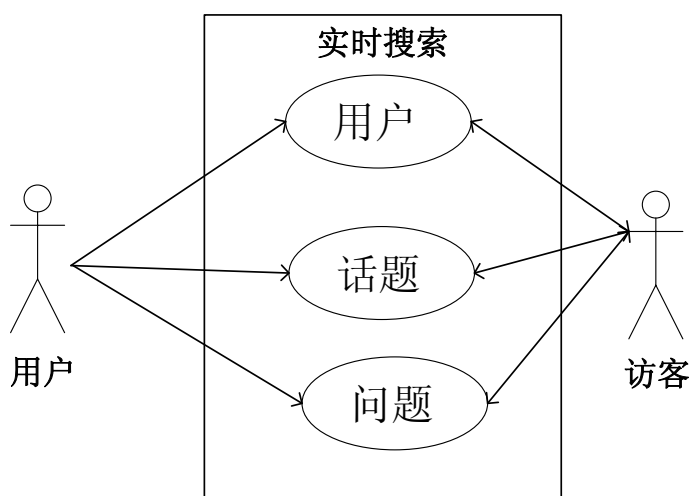


图 4-5 实时搜索用例图

通过图 4-5 可以看出实时搜索主要有以下需求:

- 用户,当输入一个用户名称的若干个相关字符之后,比如“stu”,系统会返回最相关的用户信息“stufever”等,包括用户的头像,名称等。
- 话题,当输入一个话题名称的若干字符之后,比如“宽带”,系统会返回相关的话题列表“宽带网中心”等,包括话题的图片和话题的名称等。
- 问题,用户可能是在搜索一个问题,通过输入可能的关键字,比如“今年招生”,系统会返回相关的问题列表“宽带网中心今年招收多少人?按照去年的情况,要考多少分才能进复试?”等,其中包括问题的名称和问题所属的话题列表。

4.1.6. 关注管理

关注功能是连接整个问答式社会搜索系统内容的最重要的功能,用户通过关注行为,把整个系统紧密的结合在一起,用户可以关注其他用户,关注流行的话题,甚至还可以

关注特定的问题，如下图所示：

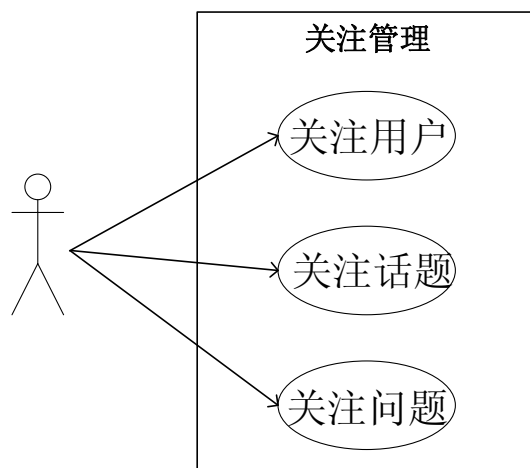


图 4-6 关注管理用例图

通过图 4-6 可以看出关注管理主要有以下需求：

- 关注用户，可以通过点击用户页面的关注按钮来关注一个用户，关注是不需要授权的，关注关系是单向的。关注这个用户之后，这个用户的所有动态，包括提问，回答问题，给问题投票等，都会出现在用户首页的 Feeds 中。
- 关注话题，用户可以随时关注自己感兴趣的，或者是当前流行的话题。关注了该话题之后，所有跟这个话题相关的活动都会推送到用户首页的 Feeds 中。
- 关注问题，当一个用户提出一个问题之后，默认就会关注该问题，其他用户也可以主动关注这个问题，这样当这个问题有最新的动态的时候，会出现在用户首页的 Feeds 中，方便用户及时地了解到最新的答案和评价等。

4.2. 非功能需求

针对 Web 网站的特点，并结合系统的功能需求，下面从系统的性能、可靠性、可扩展性和易用性等方面对系统的非功能需求进行分析。

● 系统性能

作为目前最流行的问答式社会化系统，本系统需要支持 50 万注册用户，以及一定数目的在线用户。这个对系统的总体性能提出了较高的要求，包括最大吞吐量、QPS 等。系统的目标是在支持尽可能多的并发用户的请求下，让每个用户的平均响应时间尽可能的小。

● 可靠性

可靠性是保证用户体验的一个很重要的方面，整个系统应该最大可能的减少崩溃次数，提升容错性，比如说用户在浏览器里输入一个不存在的 URL，这时候要把用户引导

到正确的页面上，而不是只显示网页不存在。同样系统在升级的过程中要向前和向后兼容，以保证新功能能够正常运行。

- 可扩展性

在可靠性时我们已经提到了系统平滑升级的重要性，整个系统要在添加新功能时，不仅仅保证新功能正常运行，还要确保已有的功能不受影响，这就对系统的可扩展性提出了很高的要求，比如数据库表格设计的兼容性，功能模块的复用等等。

- 易用性

Web 网站的设计要确保用户用最小的学习成本来掌握系统的核心功能，在设计的过程中，要对核心功能增加引导界面，帮助用户快速理解和熟悉该功能；同时还要有适当的反馈机制来确保用户的需求能够恰当的执行，另外系统的帮助手册也是理解整个系统的一个不可获取的部分，这些都是在设计的时候需要注意的地方。

4.3. 运行环境

4.3.1. 服务器的运行环境

服务器型号：戴尔 PowerEdge 2950 型服务器

操作系统：Ubuntu 10.04 LTS 发行版

内核版本：2.6.32-21-generic

Tomcat 版本：apache-tomcat-7.0.2

数据库版本：mysql 5.1.41

JDK 版本：jdk1.6.0_21

中文分词器：paoding-2.0.4

整个系统的运行在以 Linux 操作系统为基础的 Tomcat 服务上，不仅可以保证系统的安全性，还可以最大的提升系统的性能和扩展性。

4.3.2. 客户端开发环境

操作系统：Ubuntu 10.04 LTS 发行版

浏览器：Chrome

编辑器：VIM

调试工具：Chrome 开发工具

抓包工具：FireBug

版本控制：OpenSSH

客户端部署了和服务端同样的环境，方便测试和部署。编辑器使用 VIM，开发调试使用 Chrome 自带的开发工具。

第五章 设计与实现

5.1. 客户端概要设计

5.1.1. 整体架构介绍

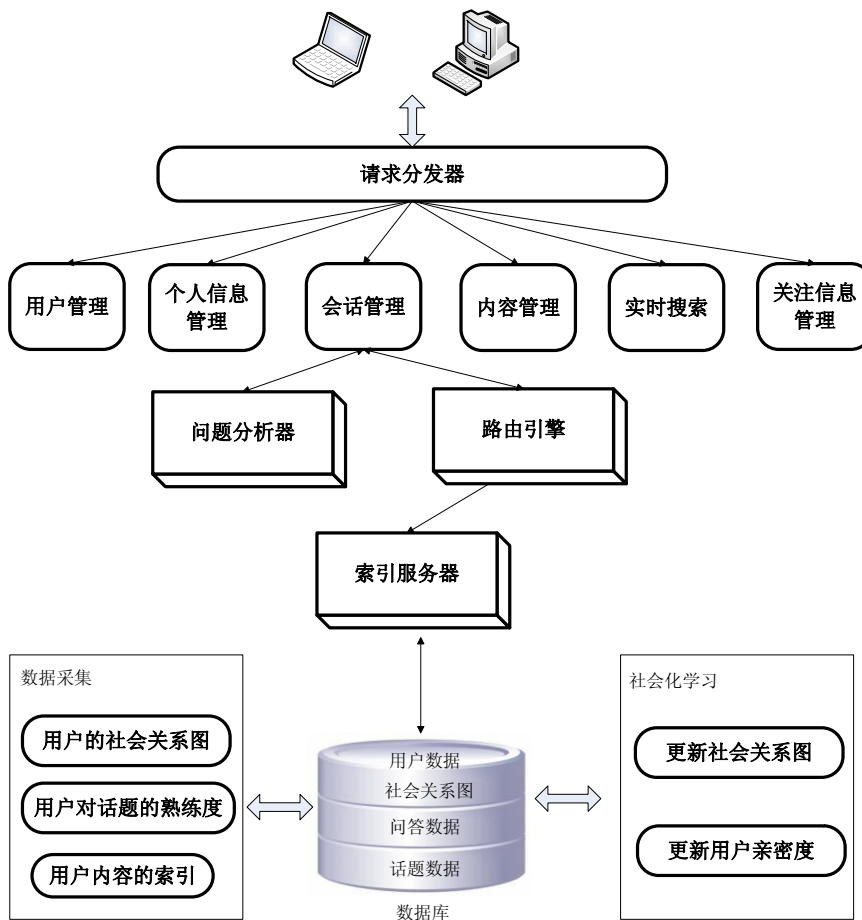


图 5-1 问答式社会化搜索系统架构图

如图 5-1 所示，问答式社会化搜索系统是一个基于 Web 服务的系统，用户通过浏览器访问该服务，通过标准 HTTP 请求和后台服务进行交互，整个系统可以划分为如下的部分：

- 客户端界面，如图中所示，系统的功能主要有用户管理、个人信息管理、会话管理、内容管理、实时搜索和关注信息管理，这些功能都是面向用户的，用户通过进行各种各样的操作来和服务端通信，服务端把结果返回给用户，进而呈现在 Web 网站上，论文的作者全程参与了所有客户端功能的设计和实现，会在后边的文章中详细介绍。

- 服务器端功能，服务器端实现了和客户端对应的功能，通过对用户数据存储、筛选和分类，从而提供给前台所需要的数据，论文作者参与了服务器端部分功能的设计工作。
- 前台和后台的通信，所有上述功能之间的通信接口都需要在开发前定义好，这样客户端和服务端就可以分开开发，互不干扰；本系统的通信接口格式采用了标准的 JSON 格式，保证了系统的通用性和兼容性，论文的作者参与了各个服务接口的制定和标准化工作。
- 数据采集和社会化学习，本系统除了给用户提供必要的功能和服务之外，还需要通过结合已有的社会化站点的数据来完善系统中的内容，并且通过对用户的操作过程进行学习，来提供更加个性化的服务，论文作者对用户影响力的计算问题进行了研究，并提出了第三章所描述的模型，该模型通过学习用户的行为，结合了用户的问题网络和社交网络，来综合衡量用户在某一个领域的影响力。

5.1.2. 目录组织结构

前台文件目录按照不同类别分为如下几类：

- jsp，网页呈现的 HTML 基本框架和通信接口
- js，动态脚本，负责各个页面的动态交互
- css，样式表，负责各页面显示格式
- img，各页面所需要的图片文件

表 5-1 详细列出了系统中客户端部分的文件列表：

表 5-1 系统前台文件列表

所属目录	文件名称	说明
jsp	nav.jsp	所有页面的导航条
	footer.jsp	所有页面的尾注
	register.jsp	注册页面
	question.jsp	问题页面
	home.jsp	用户主页
	profile.jsp	个人信息设置页面
	hidden.jsp	上传头像 iframe 相关
	user.jsp	用户页面
	topic.jsp	话题页面
/	index.jsp	登录页面
js	app.js	共用 JS 库，提供给各个页面基本功能

	ask.js	提问页面相关的 JS 脚本
	facebox.js	弹出框的 JS 脚本，包括提问、发布公告等
	home.js	用户主页相关的 JS 脚本
	jquery.min.js	提供 JQuery 功能的 JS 脚本
	jquery.ui.autocomplete.js	实时搜索自动补全的 JS 脚本
	jquery-ui-1.8.12.custom.min.js	JQuery UI 库，提供很多实用功能函数
	profile.js	个人设置页面的 JS 脚本
	question.js	问题页面的 JS 脚本
	sign.js	注册页面的 JS 脚本
	user.js	用户页面的 JS 脚本
	topic.js	话题页面的 JS 脚本
css	app.css	系统所有的层叠样式表，提供页面布局和显示的基本方式

5.1.3. 客户端结构规范

客户端开发的流程如下：首先和服务端一起定义新功能的通信接口，包括客户端需要请求的服务名称，参数列表，以及返回的结果格式等；然后撰写相应的 JSP 页面，其中包括页面的布局和格式，并且总结相应的 CSS 添加到 app.css 中；接下来，开始通过模拟数据实现动态交互的 JS 脚本，JS 脚本的原则是最小模块化，各个模块互相独立，充分利用公共库的函数避免代码复制现象；最后和服务端进行联调，在实际环境进行功能测试和系统测试。

在 JS 脚本的编写过程中，各个模块都要实现自己的命名空间，封装相应的变量和方法，因为 JS 没有实现封装机制，所以命名空间中的所有方法都是对外可见的，为了区分私有方法和公共方法，通过命名规范来区分，比如 _doLogin() 是一个私有函数，告诉开发人员该函数只能被 Sign 空间内部的其他方法调用，而不应该被作为公共方法来调用；同样对于一些简单的复制和查看操作也提供了 get 和 set 的方法。JS 脚本以单独的文件放在 js/目录下，每一个文件都有自己的命名空间，例如 question.js 主要包括了 Question 命名空间，提供了问题页面交互所需要的功能，它和 jsp/目录下边的 question.jsp 是对应的，为该页面提供相应的动态功能。

5.2. 客户端的设计与实现

在问答式社会化搜索系统里，客户端的开发是整个系统很重要的一部分。系统的目

标是实现一个 Web 网站，对于网站来说，界面的易用简洁和功能强大是用户最关心的，整个客户端采用模块化的开发，接下来分别阐述主要模块的具体设计和实现。

5.2.1. 用户管理模块

5.2.1.1. 界面设计

用户的注册过程分为如下两步：填写基本信息和关注热门用户和话题，填写的基本信息包括姓名、邮箱和密码等，填写基本信息之后就可以在系统中成功注册，之后系统会返回一个热门的用户和话题的列表，用户可以关注自己感兴趣的用户和话题，在热门用户和话题界面用户可以对自己感兴趣的信息进行关注，该界面的设计如下图所示：

图 5-2 推荐界面设计

在点击左边的话题之后，右边一栏的用户列表会根据左边的话题选择进行相应的调整 and 变化，用户还可以任意的点击图中的关注按钮来关注话题和用户。

5.2.1.2. 后台实现

推荐界面所显示的内容中相关的 JSON 串片段如下图所示：


```
/* var suggestInfo = {
  "hotTopicList":[
    {
      "topicName":"宽带网研究中心",
      "topicId":10,
      "icon":"img/bnrc.gif",
      "isFollowed":1,
      "relateTopicList":[
        {"topicName":"王文东老师","topicId":19,"icon":"img/bnrc.gif","isFollowed":1},
        {"topicName":"龚向阳老师","topicId":25,"icon":"img/topic.png","isFollowed":0}
      ],
      "popularUserList":[
        {"userName":"黄水桂","userId":2,"signature":"黄水桂的个性签名黄水桂的个性签名黄水桂的个性签名黄水桂的个性签名黄水桂的个性签名黄水桂的个性签名",
        "online":0,"userLevel":2,"isFollowed":1},
        {"userName":"王少岩","userId":3,"signature":"王少岩的个性签名","portrait":"img/default.png","online":1,"userLevel":1,"isFollowed":0},
        {"userName":"陈秋苗","userId":4,"signature":"陈秋苗的个性签名","portrait":"img/default.png","online":1,"userLevel":1,"isFollowed":0},
      ]
    },
    {
      "topicName":"网络智能研究中心",
      "topicId":10,
      "icon":"img/bnrc.gif",
      "isFollowed":0,
      "relateTopicList":[
        {"topicName":"王文东老师","topicId":19,"icon":"img/bnrc.gif","isFollowed":1},
        {"topicName":"谢东亮老师","topicId":12,"icon":"img/bnrc.gif","isFollowed":0},
        {"topicName":"龚向阳老师","topicId":25,"icon":"img/topic.png","isFollowed":0},
        {"topicName":"xx老师","topicId":22,"icon":"img/topic.png","isFollowed":0}
      ],
      "popularUserList":[
        {"userName":"黄水桂","userId":2,"signature":"黄水桂的个性签名","portrait":"img/default.png","online":1,"userLevel":1,"isFollowed":1},
        {"userName":"王少岩","userId":3,"signature":"王少岩的个性签名","portrait":"img/default.png","online":1,"userLevel":1,"isFollowed":0},
        {"userName":"陈秋苗","userId":4,"signature":"陈秋苗的个性签名","portrait":"img/default.png","online":1,"userLevel":1,"isFollowed":0}
      ]
    }
  ]
}; */
```

图 5-3 推荐界面的 JSON 数据片段

客户端对返回的 JSON 串进行解析，把解析的数据动态的填充到想要的 HTML 片段中；用户的注册部分如下图的顺序图所示：

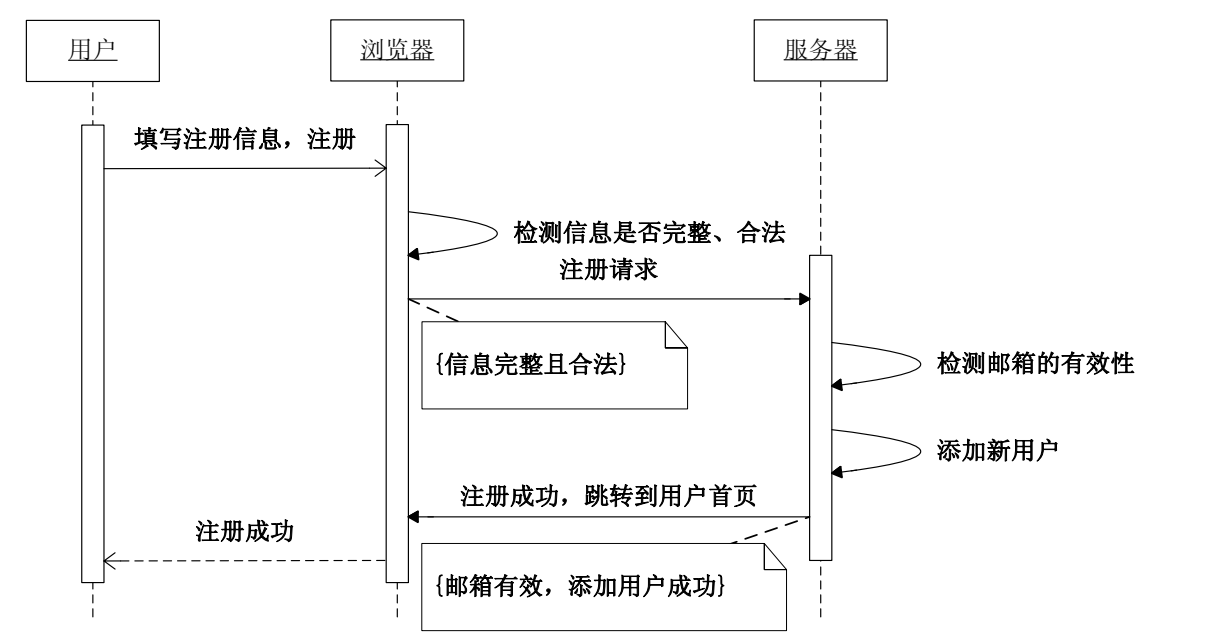


图 5-4 注册的顺序图

其中包含的关键函数如下表所示：

表 5-2 用户管理的属性方法列表

命名空间	Sign //包含注册和登录
名称	

属性说明	_isLoggedIn = 0; //默认是未登录的
方法说明	setLogin: function(login) //属性_isLoggedIn 的 set 方法 getLogin: function(login) //属性_isLoggedIn 的 get 方法 displayLoginPage: function(loginInfo) //设置登录页面的基本属性，包括导航栏的信息 displayRegisterPage: function(registerInfo) //设置注册页面的基本属性，包括导航栏的信息 displaySuggestPage: function(suggestInfo, uId) //设置推荐页面的基本属性，此时用户已经登录，要在导航上设置用户登录信息 _isEmailAddr: function(str) //检查用户输入的 Email 格式的合法性 _doLogin: function() //用户登录的提交过程，需要必要的检查 _doRegister: function() //用户注册的提交过程，需要必要的检查 _displayTopic: function(suggestInfo) //热门话题的交互过程 _displayUser: function(obj, suggestInfo) //热门用户的交互过程

5.2.2. 个人信息管理模块

5.2.2.1. 界面设计

用户不管是否登录，在系统的不同页面中都可以看到系统的导航栏，导航栏的设计如下图所示：

图 5-5 登录前和登录后的导航栏界面设计

从图中可以看出，用户在登录前和登录后都可以搜索信息，但是不同的是，用户在登录前可以点击“随便看看”来浏览系统中的内容，还可以“注册”或者“登录”到本系统中；而登录后的用户可以浏览自己的“首页”和“个人主页”，还可以“设置”个人信息。当用户停留在相应的页面时，导航栏会用突出的颜色来标示用户所在的页面，如上图中所示，用户正停留在“设置”个人信息页面，个人信息界面的设计如下图所示：

图 5-6 个人信息设置页面的界面设计

5.2.2.2. 后台实现

个人信息模块的三个基本功能分别是完善基本信息、修改密码和上传头像，这三个功能彼此是互相独立的，分别对应着不同的后台服务。

- 个人信息，也就是用户的 **profile** 信息，包括姓名、性别、生日、国家等，通过调用 "save_profile.do" 服务，把更新的信息保存到后台，如下图所示：
- 上传头像，最常用的方法就是通过提交表单来实现，不过提交表单之后页面会跳转或者刷新，为了提升用户体验，采用了另外一种方法，该方法利用了 **AJAX** 技术可以动态的和服务器交互的特点，又克服了 **AJAX** 不能传输二进制图像数据的缺点，通过把 **iframe** 框架和表单结合起来，让表单隐藏起来，这样提交了表单之后，并没有刷新页面，而只是刷新了 **iframe**，从而避免了页面的刷新，如下图所示：

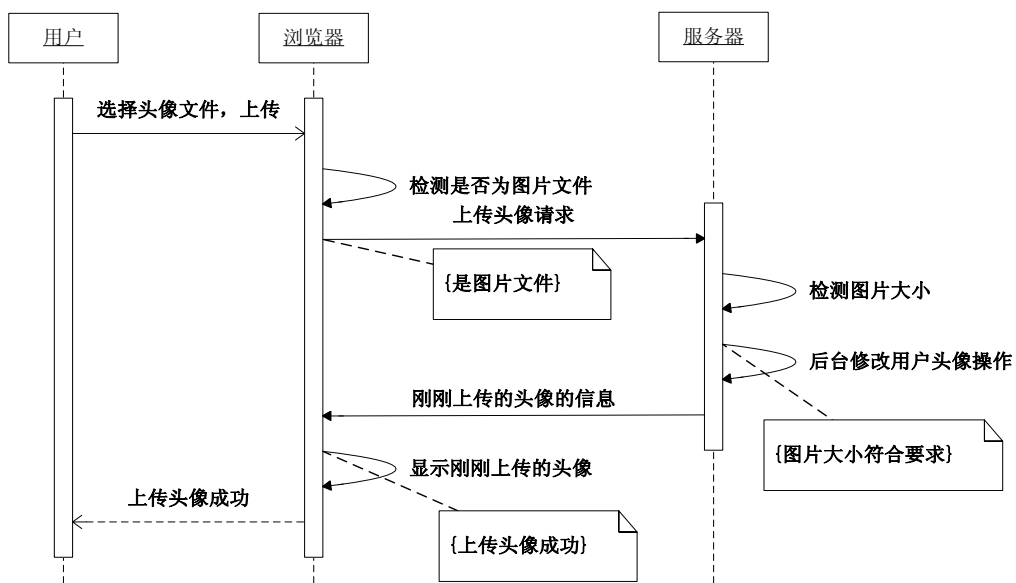


图 5-7 上传头像的顺序图

- 密码设置, 通过和"modify_password.do"服务交互, 来实现用户密码的修改功能。下表阐述了个人信息管理模块的各个功能函数:

表 5-3 个人信息管理的属性方法列表

命名空间名称	Profile //个人信息
属性说明	无
方法说明	displayPage: function(obj, labList) //个人信息页面的初始化函数, 设置相应的页面和脚本函数 displayProfile: function(profileObj) //显示用户的基本信息 uploadCallBack: function(type, url) //上传头像的回调函数, 解析返回的结果, 显示不同的状态 _setProfile: function() //设置个人信息的交互过程 _modifyPassword: function() //修改密码的交互过程, 提交前需要做参数检查 _doUploadPhoto: function() //上传头像的交互过程

5.2.3. 会话管理模块

5.2.3.1. 界面设计

会话管理模块是问答式社会化搜索系统中最基本也是最重要的功能, 包括了用户最

常用的一些功能：提问、回答、评论和评价。提问功能利用了 JQuery 提供的 `facebox.js` 插件，该插件提供了一个类似于 Facebook 界面的轻量级的弹出框，它允许用户载入外部页面、图片和其它任何内容，它还提供了一些基本的界面效果给开发者，比如背景模糊，loading 图片等，该界面的设计如下图所示：

图 5-8 添加问题界面设计

回答功能中用户不仅仅可以输入文本，还可以对文本进行富文本编辑，该功能主要是利用了 JQuery 的 `jquery.qeditor.js` 插件，该插件提供了基本的富文本操作功能，它把用户输入的内容转换成相应的 HTML 结构，通过 HTML 格式显示在界面上；评论功能默认是隐藏起来的，点击评论之后可以动态的显示出来；评价功能提供了“顶”和“踩”两个操作，用户在评价完之后，系统会根据当前答案的投票数来决定该答案的位置，进行动态的调整。

5.2.3.2. 后台实现

和“搜索”功能一样，用户可以在任何界面提出问题，因此整个导航栏的 HTML 片段是各个页面共享的，具体的内容如下所示。其它的页面需要通过下边的代码来引用导航栏的界面：

```
<%@ include file="nav.jsp" %>
```

在 `nav.jsp` 中，除了提供导航栏的功能外，还提供了一下的公共功能：

- Loading——客户端向服务器请求之后的 loading 图标
- Message Box——客户端请求后，返回成功或者失败信息的提示框
- New Ask——提问问题的弹出框
- New Board——发布公告的弹出框
- Modify Icon——修改用户头像或者话题图标的相关界面

```

<div id="header">
  <div class="container">
    <div class="left_wrapper">
      <div id="site_name">
        <a href="/">
          
        </a>
      </div>
      <div id="add_ask">
        <input type="text" autocomplete="off" class="ac_input" />
        <a href="javascript:void(0)">我要提问</a>
        <!--<a href="#" onclick="return Ask.addAsk();">我要提问</a-->
      </div>
    </div>
    <div class="sidebar">
      <div id="user_bar">
        <!--<a class="home" href="#">首页
          <span id="notify_badge" class="badge force-hide">新</span>
        </a>
        <a href="#" class="user">个人主页</a>
        <a href="#">设置</a>
        <a href="#">退出</a>
        <a href="#" class="user">随便看看</a>
        <a href="#">注册</a>
        <a href="#">登录</a-->
      </div>
    </div>
  </div>
</div>

```

图 5-9 导航栏和提问功能的 HTML 片段

会话管理中回答问题的顺序图如下所示：

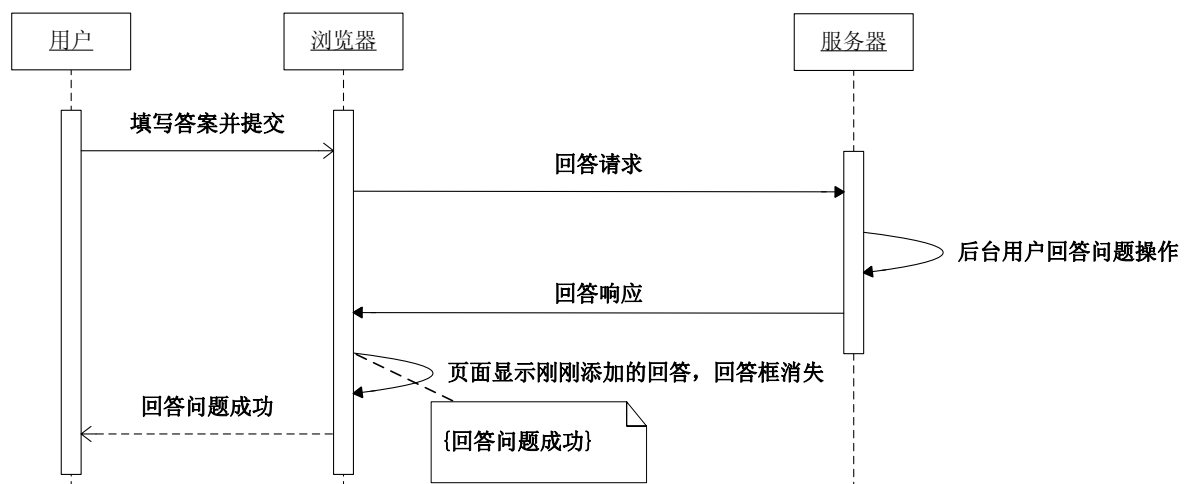


图 5-10 回答问题的顺序图

评论问题的顺序图如下所示：

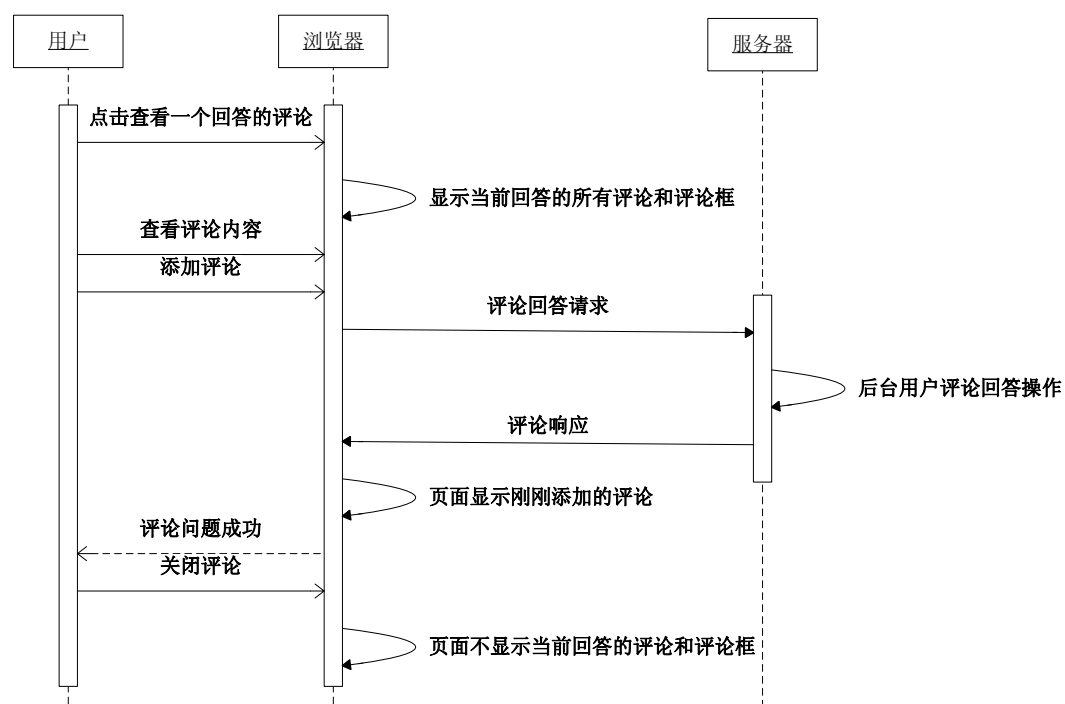


图 5-11 评论问题的顺序图

各个功能的具体方法函数如下表所示：

表 5-4 会话管理的属性方法列表

命名空间名称	Ask //个人信息
属性说明	无
方法说明	initAC: function() //初始化导航栏的交互过程 addAsk: function() //显示提问框，进行初始化 submitAsk: function() //提交问题的交互过程
命名空间名称	Question //回答、评价和评论功能
属性说明	无
方法说明	_doDigg: function(voteBtnDiv, qId, answererId, flag, upNum, downNum) //评价功能的交互过程 _diggSuccessCallBack: function(voteBtnDiv, qId, answererId, flag, upNum, downNum) //评价之后界面的调整过程 _doComment: function(btn, cTextDiv, qId, qTime, answererId, userInfo) //提交评论信息的交互过程

	<div><div>_doTagCommon: function(obj, qId, flag) //标记一个问题为常见问题</div><div>_doRemoveTopic: function(obj, questionId) //删除问题所属的话题</div><div>_doAddTopic: function(questionId) //添加问题所属的话题</div></div>
--	--

5.2.4. 内容管理模块

内容管理模块是和用户最相关的一个模块，用户在大多数时候都在跟该模块打交道，主要包括了首页、问题页面、话题页面和用户页面。这四个页面承载了问答式社会化搜索系统的主要内容，接下来分别描述这些页面的详细设计和实现。

5.2.4.1. 首页

用户首页是用户登录之后跳转到的页面，在这个页面包含了用户最关心的信息，当系统有最新的消息到达时会推送给用户，这些通知主要包含以下几种：

表 5-5 通知类型

通知 ID	通知类型	例子
1	有人关注该用户	陈宽关注了你
2	有人回答了你的问题	陈宽 回答了你的问题 网络技术研究院最近的就业情况怎么样？主要面向哪些行业？工资水平怎么样？
3	有人向你直接提问	陈宽 向你询问 小论文什么时候开始写比较合适？
4	管理员发布新的公告	宋思奇 发表了公告 欢迎 2010 年的师弟师妹们加入宽带网中心这个大家庭！

通知系统的界面设计如下所示：

图 5-12 通知系统设计界面

当有新的通知时，导航栏的“首页”按钮旁边会显示“新”这个小按钮，提示用户去查看新的通知消息。如果用户没有及时看到该通知，后台还会在每周的固定时间发邮件通知该用户，以使用户能够及时的了解最新的信息。首页主要分为了三部分：问题摘要，首页导航和推荐，首页的具体界面设计如下图所示：

图 5-13 首页界面设计

其中的推荐模块用到了本文在第三章中讲到的用户影响力计算模型，通过用户影响力的计算，系统后台会把用户最感兴趣的人推荐给用户；同样，用户首页的问题摘要也会根据用户影响力的计算结果进行相应地调整，首页中每个问题的回答都会显示那些较有影响力的人的回答的信息，首页查看问题与公告的顺序图如下所示：

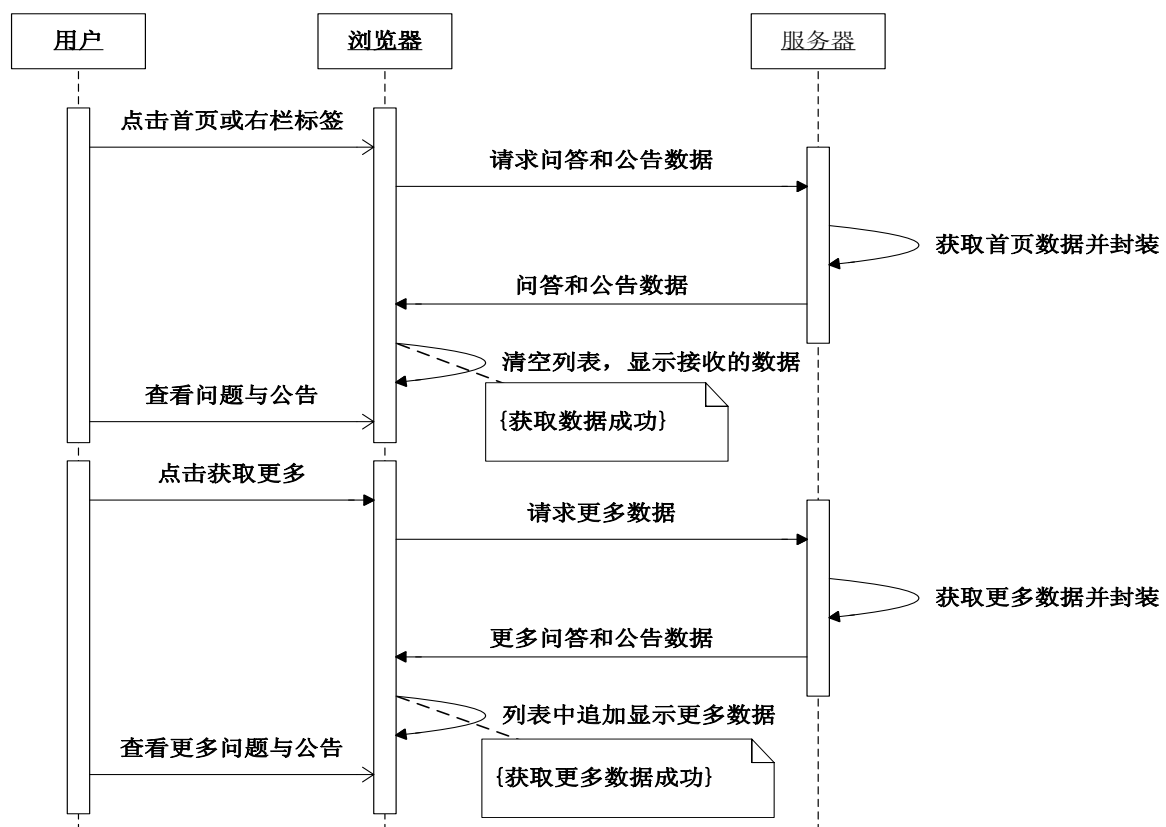


图 5-14 首页查看问题与公告顺序图

用户首页的相关方法函数如下表所示：

表 5-6 用户首页的属性方法列表

命名空间名称	Home //用户首页
属性说明	_pageNo: 1 //当前浏览页
方法说明	displayPage: function(qaListObject) //设置用户首页的基本信息，初始化相关脚本 _displayNotice: function(noticeList) //显示用户通知 _createNotice: function(noticeList) //根据通知类型，创建 HTML 文本 _doNotice: function(obj) //点击通知的处理方法 _displayRightColumn: function(popularUserList, hotTopicList) //右边栏中最热门的话题和用户的显示 _showTags: function() //切换我的首页、我关注的问题和推荐给我的问题 _showPopularUsers: function(userList) //显示热门的用户 _showHotTopics: function(topicList) //显示热门的话题 _getMore: function() //获取更多问题列表

5.2.4.2. 问题页面

问题页面显示每一个具体的问题的详细信息，这里包含了问答式社会化搜索系统中最重要的信息，系统中定义了四种问题类型：

表 5-7 问题类型

问题 ID	问题图标	问题类型	问题例子
1		普通问题	我本科外校通信，想考网研，对编程能力是不是要求比较高啊，现在大三，如果想多做些准备，都应该准备些什么科目呢，怎么练习编程呢，求指导~多谢！
2		常见问题	常见问题：宽带网中心今年招收多少人？按照去年的情况，要考多少分才能进复试？
3		公告	宽带网中心各位老师 2010 年及 2011 年的招生情况汇总
4		直接向其他用户提问的问题	请问王少岩：宽带网中心的复试要准备哪些东西？

问题页面在界面设计中最重要就是把问题相关的信息直观简单地呈现给用户，如

下图所示，左边部分是主要的信息区域，首先会展示一个问题的简要信息，之后会呈现一个一个的答案列表；右边的区域中，在最上方明显的地方，用户可以关注或者取消关注这个问题，下边的部分的相关问题和问题页面统计信息可以增加用户在系统中的停留时间。

图 5-15 问题页面的界面设计

问题页面的内容包括用户的提问、回答、评论和评价信息，以及相关问题和问题状态，具体的方法属性如下表所示：

表 5-8 问题页面的属性方法列表

命名空间 名称	Question //问题页面
属性说明	无
方法说明	<div>displayPage: function(qaObject) //初始化问题页面</div> <div>displayBrief: function(qListObj, isAdmin) //显示问题缩略信息</div> <div>_displayDetail: function(qaObj) //根据不同的问题类型，显示详细的问题信息</div> <div>_showTopic: function(qObj) //显示问题相关的话题，初始化交互过程</div> <div>_showDetailQuestion: function(qaObj) //显示普通问题的具体信息，初始化相关的操作交互，问题的所有者可以删除该问题</div> <div>_showDetailBoard: function(qaObj) //显示公告的具体信息</div> <div>_showQuestionEditIcon: function(qId) //显示问题的编辑按钮，初始化编辑问题功能的交互过程</div> <div>_showAnswerEditIcon: function(qId) //显示回答的编辑按钮，初始化编辑答案功能的交互过程</div> <div>_showDetailAnswerComment: function(qaObj) //显示问题的评论，默认隐藏</div>

	_showOneAnswer: function(isDigged, aObj, qId, qTime, userInfo) //显示具体的一个答案的详细信息，并初始化评论的交互过程 _showOneComment: function(cObj, qTime) //显示具体的评论信息 _showStatus: function(qaObj) //根据不同的问题类型，显示状态图标 _showAddAnswer: function(qaObj) //显示回答框，初始化回答的交互过程 _showRelated: function(relaArray) //显示边栏的相关问题 _showBriefQA: function(oneQAObj, isAdmin) //提供给其它页面来显示问题缩略信息
--	---

5.2.4.3. 话题页面

所有的问题都属于一个或几个话题，通过给问题加上话题标签，一方面可以给问题分类，另一方面也让关注此话题的人能够收到该问题。管理员在话题页面可以发布公告信息，这样关注该话题的用户可以及时的收到最新的信息。具体的函数方法如下：

表 5-9 话题页面的属性方法列表



命名空间 名称	Topic //话题页面
属性说明	_pageNo: 1 //当前用户浏览的页面
方法说明	display: function(topicObj) //初始化话题页面 modifyIcon: function() //修改话题的图标 addBoard: function() //添加公告 submitBoard: function() //提交公告的交互过程 _showTopic: function(qTopic) //显示话题的概要信息，初始化编辑交互过程 _initTabs: function(topicObj) //切换不同的类别，相关问题、常见问题和公告 _displayRightColumn: function(topicObj) //显示右边栏信息 _switchTab: function(topicObj, requestType) //切换不同类别的交互过程 _getMore: function(topicId, isAdmin) //获取更多问题缩略信息 showBriefBoard: function(oneBoardObj) //显示公告的缩略信息，公共方法，其它页面也可以调用 _doUpdateIcon: function() //上传图标的交互过程 uploadCallBack: function(type, url) //上传图标成功的回调函数

5.2.4.4. 用户页面

用户页面不仅仅有用户的提问和回答历史，还有用户的粉丝列表、好友列表和关注的话题列表，在用户的页面里，可以对这个用户直接提问，以便保证问题可以被该用户

及时的看到。整个系统中的用户被定义成了不同的类别，不同类别的用户拥有不同的权限，具体的分类如下：

表 5-10 用户类型

用户 ID	用户图标	用户类型	用户权限
1	无	访客	随便看看，查看用户页面、话题页面和问题页面
2	无	普通用户	除了访客的权限之外，还可以编辑属于自己的内容和公共内容
3		星级用户	和普通用户拥有相同的权限，该类型用户更加权威
4		管理员	某些特殊话题的管理员，除了 3 的权限外，还可以发布公告信息
5	无	超级用户	开发人员，除了 4 的权限外，还可以审核用户提交的星级用户和管理员的申请

具体的方法属性如下所示：

表 5-11 用户页面的属性方法列表

命名空间名称	User //用户页面
属性说明	_pageNo: 1 //当前用户浏览的页面
方法说明	display: function(userObj) //初始化用户页面 showBriefAskToUserQuestion: function(oneATUQ) //显示向别人直接提问的问题的缩略信息，可以被其它页面调用 _showUser: function(userInfo, currentUserInfo, applying) //显示用户的详细信息 _initTabs: function(userObj) //切换不同的显示类型，包括主页、问过、答过和问我 _initAskToUser: function(userObj) //初始化向某人提问的交互过程 _displayRightColumn: function(userObj) //显示右边栏的信息，包括关注按钮、关注的话题个数，粉丝列表和好友列表 _displayFollowCount: function(userObj) //显示粉丝和好友的概要信息 _displayFollowedTopics: function(userObj) //显示关注的话题列表 _displayLatestFollower: function(userObj) //显示最近的粉丝和好友

	<pre> _getRequest: function(userObj, requestType) //显示具体的分类信息，使用 AJAX 获取信息，包括主页、提问、回答、问我、粉丝、好友和话题 _createPeople: function(userObj, peopleArray) //创建用户列表 _createFollowedTopics: function(topicArray, curUserInfo, userInfo) //创建话题 列表 </pre>
--	--

5.2.4.5. 协作编辑功能

协作编辑体现了 Wiki 的精神，不仅可以通过用户贡献来丰富社区的知识，还可以提升用户的积极性和归属感。在问答式搜索系统中，几乎所有的内容都是有用户贡献的，比如说问答、问题所属的话题列表，话题的简介、图片等等。通过编辑问题所属的话题，可以把问题进行分类，让问题能够分发到合适的用户那里；通过编辑话题的概要信息，可以让话题的内容丰富起来，当其他用户浏览到这个话题的时候，可以了解到更加详细的信息。

一个问题的提问和回答，只有所属的用户才可以编辑，用户可以随时编辑自己的问题和答案，让问题描述更加准确，答案更加详细。编辑功能也是通过利用 AJAX 技术动态的和后台交互实现的，当用户编辑问题的时候，需要如下的过程：

```

questionId = $(".ask .title .in_place_edit").attr("question_id");
var args = "questionId=" + questionId;
args += "&questionContent=" + encodeURIComponent(questionContent);
$.ajax({
    type: "POST",
    url: "question_update.do",
    data: args,
    success: function(result) {
        $("#form-question").remove();
        $(".ask .title").show();
        $("#ask_title").text(questionContent);
        btn.removeAttr("disabled");
        App.popMessageBox(" 修改问题成功! ");
        App.loading(false);
    },
    error: function(result) {
        btn.removeAttr("disabled");
        App.popMessageBox(" 修改问题失败! ");
    }
});

```

```
        App.loading(false);  
    }  
});
```

前台通过 POST 方式，把所需要的参数准备好，其中包括问题 ID 和问题的内容，通过调用"question_update.do"服务来和后台交互，通过不同的返回状态来区别是否成功的编辑问题，当成功或者失败的时候，前台会提示用户，给用户不同的响应。这里有一个要注意的地方，当用户提交了之后，要把提交按钮禁用掉，防止用户重复的提交。

同样的道理，用户在给问题添加和删除话题的时候，通过调用"add_question_topic.do"和"remove_question_topic.do"服务来完成交互过程，用户在编辑话题介绍的时候，调用的是"topic_content_update.do"服务。

5.2.5. 实时搜索模块

5.2.5.1. 界面设计

实时搜索可以动态的返回跟用户查询相关的信息，包括相关的用户信息、话题信息和一些问题等，具体的界面设计如下图所示：

图 5-16 实时搜索的界面设计

5.2.5.2. 后台实现

实时搜索功能的前端实现是通过对 JQuery 插件 jquery.ui.autocomplete.js 的修改而实现的，当用户输入字符长度大于等于两个之后，后台在每次用户改变输入的时候都会动态的查询，并且返回最新的匹配结果，如下图所示：

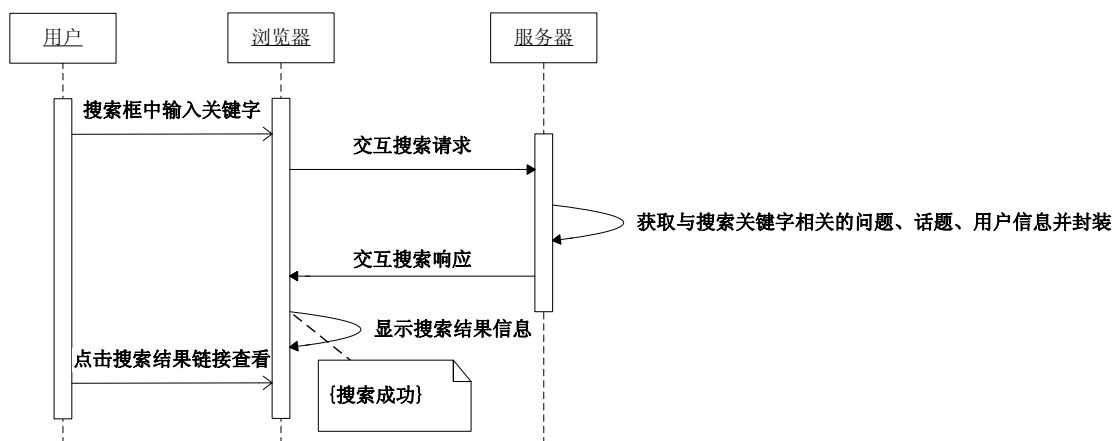


图 5-17 交互搜索顺序图

具体的请求格式如下：

```

$("#add_ask input").autocomplete({
    minLength: 1,
    source: "search.do",
    focus: function(event, ui) {
    },
    select: function(event, ui) {
    }
})
  
```

其中`$("#add_ask input")`是搜索框所属的 DOM 节点，如前边所述，当输入的查询长度大于 1 的时候，前台就会向 `search.do` 发送请求，后台返回定义好的通信格式 JSON 串，包括了可能匹配的用户、话题和问题信息，如下所示：

```

var projects = [
    {
        label: "宽带网中心",
        type: "topic",
        id: 4,
        followCount: 4,
        icon: "img/topic.png"
    },
    {
        label: "宽带通信网这门课难不难？",
        type: "ask",
        id: 2,
  
```



```

        topic: "宽带通信网"
    },
    {
        label: "刘宽带",
        type: "user",
        id: 100,
        followCount: 100,
        icon: "img/default.png"
    }
];
```

其中定义了三种类型："user"表示用户信息，"ask"表示问题信息，"topic"表示话题信息，前台通过解析返回的 JSON 串，把这些信息按照特定的 HTML 格式动态的生成填充到各个页面，在通过 JQuery 动态生成这些 DOM 片段的时候，一定要注意区分不同的类型，并且还要生成相应的 URL，方便用户在点击该链接之后跳转到相应的页面。

5.2.6. 关注管理模块

关注管理模块分为三部分：关注用户、关注话题和关注问题，通过这些关系把整个系统的内容结合到一起，关注功能的顺序图如下所示：

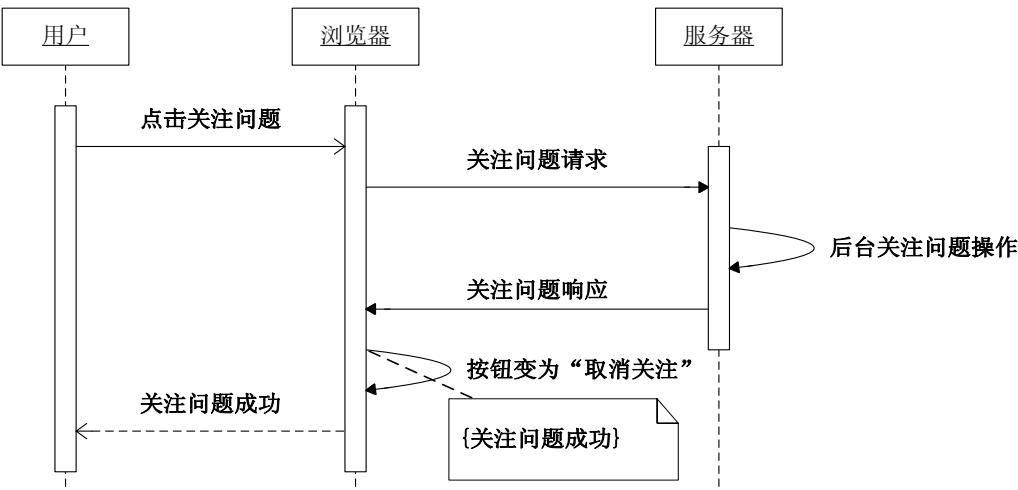


图 5-18 关注问题的顺序图

关注功能的前端实现也是通过 AJAX 的方式和后台交互，分别对应如下的服务：

表 5-12 关注类型

类型	服务名称	关注类别
用户	follow_user.do	0-关注, 1-取消关注
话题	follow_topic.do	0-关注, 1-取消关注
问题	follow_question.do	0-关注, 1-取消关注

关注服务的按钮使用了特殊的 CSS 样式构成, 使用 1 个像素的图片, 以节省带宽, 关注按钮的样式表如下:

```
a.green_button {
    background: #dffb1 url(../img/green_button_bg.gif) bottom left repeat-x;
    border: 1px solid #8CB332;
    border-bottom: 1px solid #648517;
    display: inline-block;
    line-height: 24px;
    padding: 2px 10px 1px 10px;
    -moz-border-radius: 3px;
    -webkit-border-radius: 3px;
    border-radius: 3px;
    text-decoration: none;
    color: #406A24;
    text-align: center;
    text-shadow: 0 1px #D4ED95;
    font-weight: bold;
}
```

关注管理模块对应的属性方法如下表所示:

表 5-13 用户页面的属性方法列表

命名空间 名称	User //用户页面
属性说明	无
方法说明	doFollow: function(obj, flag, userObj) //关注用户的交互过程 _displayFollowButton: function(userObj) //显示关注按钮
命名空间 名称	Topic //话题页面
属性说明	无
方法说明	doFollow: function(obj, flag, curUserObj, userObj) //关注话题的交互过程 _displayRightColumn: function(topicObj) //显示关注按钮

命名空间 名称	Question //问题页面
属性说明	无
方法说明	<code>_doFollow: function(qId, flag, curUserObj, readNum, followNum) //关注问题的交互过程</code> <code>_showStatus: function(qaObj) //显示关注按钮</code>

第六章 系统测试

本章详细描述系统的测试工作，首先介绍测试环境，之后分别对各个模块进行单元测试，并展示运行效果；然后对系统进行集成测试，最后对测试结果进行分析和总结。

6.1. 系统测试环境

Web 测试主要是通过浏览器来访问后台系统，为了保证测试的全面性，测试环境包括了如下的浏览器：

- IE 系列浏览器，包括 IE7，IE8 和 IE9，本系统不兼容 IE6 浏览器，因为 IE6 安全性能差，漏洞很多，并且不支持 HTML5 和 CSS3，其解析不符合 W3C 标准。
- Chrome 浏览器，本系统的开发环境主要是使用的该浏览器。
- Firefox 和 Opera，针对较新版本的浏览器。

测试上述浏览器的时候，也会对不同的操作系统进行测试，因为 Windows 和 Linux 的同一种类型的浏览器在有些情况下显示也会有差异。

在测试的时候，要清空浏览器的缓存，避免有些 JS 文件使用旧的缓存文件。

本系统考虑到用户使用浏览器的差异性，对使用不符合标准规范的浏览器的用户进行提示升级，已确保用户达到最好的使用效果，如下图所示：



图 6-1 浏览器版本过低提示图

6.2. 系统单元测试

在各个模块的开发过程中都伴随着单元测试，在一个模块的开发过程中，客户端和服务端首先要制定好该功能相关的通信接口，这样客户端在开发该功能的时候就可以通

过测试桩——模拟测试环境，通过伪数据来测试。结束第一步测试之后，要和后台进行实际的联调，通过真实环境的测试来确保运行的正确性，下边通过对各个模块进行测试来讲述系统单元测试的步骤。

6.2.1. 用户管理模块

- 功能性测试，如下表所示：

表 6-1 用户管理功能性测试

用例编号	测试用例	预测结果	测试结果
1-1	一个访客进行注册过程第一步，填写注册信息，然后点击注册	注册成功调转到推荐页面	符合
1-2	一个访客使用人人帐号注册，按照顺序授权，验证	注册成功调转到推荐页面	符合
1-3	一个访客使用新浪微博帐号注册，按照顺序授权，验证	注册成功调转到推荐页面	符合
1-4	一个访客使用新浪微博帐号注册，按照顺序授权，验证	注册成功调转到推荐页面	符合
1-5	在推荐页面关注几个热门话题和用户，然后点击完成	跳转到用户首页	符合
1-6	已注册用户登录	登录成功跳转到用户首页	符合
1-7	已注册用户通过人人帐号登录	登录成功跳转到用户首页	符合
1-8	已注册用户通过新浪微博帐号登录	登录成功跳转到用户首页	符合
1-9	已注册用户通过腾讯微博帐号登录	登录成功跳转到用户首页	符合

- 容错性测试，如下表所示：

● 表 6-2 用户管理容错性测试

用例编号	测试用例	预测结果	测试结果
1-10	一个访客进行注册，填写的邮箱已经被注册过	返回“该邮箱已经被注册过”的错误	符合
1-11	一个访客进行注册，两次输入的密码不一致	返回“输入密码不一致”的错误	符合
1-12	用错误的用户名和密码登录	重新回到登录页面，返回“该用户不存在”的错误	符合

- 稳定性测试，在并发用户超过 100 个的情况下，会出现响应时间过长的问題，通过分析发现跟服务器的硬件性能有关，如果系统需要支持更多的并发用户，

除了要进行软件上的优化之外，还要适当地提高硬件的性能。

6.2.2. 个人信息管理模块

- 功能性测试，如下表所示：

表 6-3 个人信息管理功能性测试

用例编号	测试用例	预测结果	测试结果
2-1	修改用户资料并保存	系统返回“保存个人信息成功”	符合
2-2	修改密码并提交	系统返回“更改密码成功”	符合
2-3	上传头像，选择本地文件并上传	系统返回“上传头像成功”	符合

- 容错性测试，如下表所示：

表 6-4 个人信息管理容错性测试

用例编号	测试用例	预测结果	测试结果
2-4	不填写姓名，保存用户资料	返回“姓名不能为空”的错误	符合
2-5	用同样的密码作为修改的新密码	返回“旧密码与新密码相同”的错误	符合
2-6	上传头像不符合系统要求，比如没有选择文件，上传其它格式的文件，上传大于 1M 的文件等	返回“请选择需要上传的头像文件!”、“请选择 JEP、JPEG、GIF 或 PNG 格式的头像文件上传!”、“请选择小于 1M 的头像文件上传!”等错误	符合

- 稳定性测试，同用户管理模块一样，并发请求过多的时候，系统响应时间会变长。

6.2.3. 会话管理模块

- 功能性测试，如下表所示：

表 6-5 会话管理功能性测试

用例编号	测试用例	预测结果	测试结果
3-1	提问，填写问题标题，点击提问	跳转到新提问的问题页面	符合
3-2	回答问题，并进行富文本编辑	系统返回“回答成功”，并闪亮显示新的回答	符合

3-3	评论某一个答案，点击评论	系统返回“评论成功”，并闪亮显示新的评论	符合
3-4	“顶”或者“踩”某一个答案	系统返回“评价成功”，并根据新的排名调整该答案的位置，闪亮显示	符合

- 容错性测试，如下表所示：

表 6-6 会话管理容错性测试

用例编号	测试用例	预测结果	测试结果
3-5	提交空的问题	返回“请输入一个问题”的错误	符合
3-6	提交空的答案	返回“请输入一个回答”的错误	符合
3-7	提交空的评论	返回“请填写评论内容”的错误	符合
3-8	重复“顶”或者“踩”	返回“您已经进行过评价，不能重复评价”的错误	符合

- 稳定性测试，同用户管理模块一样，并发请求过多的时候，系统响应时间会变长。

6.2.4. 内容管理模块

- 功能性测试，如下表所示：

表 6-7 内容管理功能性测试

用例编号	测试用例	预测结果	测试结果
4-1	浏览用户首页，点击侧边栏的切换标签	获取到不同的 Feeds	符合
4-2	浏览问题页面，查看评论信息	获得详细问题信息和评论信息	符合
4-3	浏览话题页面，点击切换标签	获取到不同的问题信息，	符合
4-4	浏览用户页面，点击切换标签，粉丝和好友概况	获取到不同的问题信息，好友和粉丝列表等	符合
4-5	不同页面之间的切换，点击话题到话题页面，点击用户到用户页面，点击问题到问题页面	跳转到相应的页面	符合
4-6	编辑问题所属的话题信息，添加或者删除话题	系统返回“添加话题成功”或者“删除话题成功”	符合
4-7	编辑话题的概要信息	系统返回“修改成功”	符合

4-8	上传或者修改话题图标	系统返回“上传成功”	符合
-----	------------	------------	----

- 容错性测试，内容管理模块主要是查看用户的各种信息，多次刷新页面，看一看是否和预期结果相同，经验证没有问题。

● 表 6-8 内容管理容错性测试

用例编号	测试用例	预测结果	测试结果
4-9	给问题添加同样的话题	返回“此话题已经添加过了，请选择其它话题!”的错误	符合
4-10	发布空的公告信息	返回“公告不能为空”的错误	符合
4-11	上传图标不符合系统要求，比如没有选择文件，上传其它格式的文件，上传大于 1M 的文件等	返回“请选择需要上传的图标文件!”、“请选择 JEP、JPEG、GIF 或 PNG 格式的头像文件上传!”、“请选择小于 1M 的头像文件上传!”等错误	符合

- 稳定性测试，同用户管理模块一样，并发请求过多的时候，系统响应时间会变长。除此之外，用户在登录成功跳转到首页的过程中，后台会返回 JDBC 相关的错误信息，由于该问题出现的不确定性，并且概率较低，不容易复现，是系统下一步需要解决的问题。

6.2.5. 实时搜索模块

- 功能性测试，如下表所示：

表 6-9 实时搜索功能性测试

用例编号	测试用例	预测结果	测试结果
5-1	输入查询关键词	系统返回匹配的结果	符合
5-2	点击返回结果中的用户、话题或者问题	跳转到相应的页面	符合

- 容错性测试，实时搜索模块主要是查询各种信息，多次查询，看一看是否和预期结果相同，经验证没有问题。

6.2.6. 关注管理模块

- 功能性测试，如下表所示：

表 6-10 关注管理功能性测试

用例编号	测试用例	预测结果	测试结果
6-1	关注或者取消关注一个用户	系统返回“关注成功”或者“取消关注成功”	符合
6-2	关注或者取消关注一个话题	系统返回“关注成功”或者“取消关注成功”	符合
6-3	关注或者取消关注一个问题	系统返回“关注成功”或者“取消关注成功”	符合

- 容错性测试，关注管理模块交互比较简单，最重要的就是避免多次重复请求，客户端通过禁用按钮提交属性来实现。

6.3. 系统集成测试

通过系统的集成测试，可以验证各个模块之间的协调和连贯程度，保证系统的整体性。接下来通过一个场景来测试系统的整体性，同时又保证各个模块可以被验证到。

用户“陈宽”注册并登录到系统中，注册过程分为两部分，第一部分是填写基本信息，另外本系统也可以通过其他平台的账号注册，包括人人账号、腾讯微博和新浪微博，登录过程如下图所示：



图 6-2 注册过程，填写基本信息

注册的第二部分是关注热门的话题和用户，图 6-3 左边部分是系统推荐的热门话题，因为本系统是一个以考研问答为主要主题的系统，所以各个中心自然成为了系统推荐的热门话题，例如：“宽带网研究中心”和“网络管理研究中心”等；右边是系统中的一些热门用户，这些用户往往都比较活跃，或者是一些管理员等。在这个界面用户可以很方便的关注这些热门的话题和用户，以便系统推送相关的信息。



图 6-3 注册过程，推荐热门话题和用户

用户可以在系统导航的搜索框中输入感兴趣的信息，后台会根据系统的输入动态地返回相关的信息，例如图 6-4 所示，用户搜索了“宽带网中心的复试需要准备什么内容？”，在搜索的过程中，系统会根据用户的输入调整返回的信息。如下图所示，系统在输入“宽”的时候，系统会返回叫做“陈宽”的用户，并且还返回了话题“宽带网研究中心”等；当用户进一步输入“宽带”两个字的时候，“陈宽”这个用户信息就会消失掉，系统会返回更多“宽带”相关的话题和问题，比如话题“宽带通信”等；当用户在输入完自己想要提问的问题的时候，就会发现已经有用户提问过类似的问题，用户可以直接点击该问题就会跳转到该问题的页面上。

图 6-4 实时搜索功能

和实时搜索类似，用户可以随时在导航栏中点击“我要提问”按钮进行提问，如下图中，“陈宽”提了一个考研分数相关的问题，并且选择了“宽带网研究中心”作为这个问题的话题，这样系统就可以把这个问题分发给更加合适的人来回答。

图 6-5 提问过程

通知系统可以让用户及时地了解最新的信息，如图 6-6 所示，过了一段时间后，“宋

思奇”回答了“陈宽”提出的问题，这个时候在“陈宽”的首页会显示这条通知信息，陈宽就可以及时的看到该通知，点击相应的问题链接就可以跳转到问题页面，对相关的答案进行查看。



图 6-6 通知系统

问题页面是系统中重要的页面，在该页面呈现了跟问题相关的一切信息，如图 6-7 所示，陈宽在相关的问题页面看到了最新的答案，并且通过评价功能对管理员“宋思奇”进行感谢。



图 6-7 评论功能

系统的主要页面结构还包括用户的首页，该页面是用户登录之后首先看到的页面，用户可以随时地看到最新的信息，该页面的右下角包括系统推荐的一些热门话题和用户，如下图所示：



图 6-8 用户首页

在本系统中，话题是系统中的重要组成部分，问题是属于一些话题的，下图展示了一个话题“网络技术研究院”的详细信息。



图 6-9 话题页面

个人信息管理页面可以对用户信息进行完善，如下图所示，“陈宽”编辑了个人信息，上传了头像，并且更新了签名信息，同时他还关注了感兴趣的话题和用户，例如：“保研”和“考研”等话题。



图 6-10 用户主页

6.4. 测试结果分析

通过各个模块的单元测试，可以看出系统的各个子模块基本实现了需求分析中提出来的功能，并且系统的集成测试也说明了本系统已经可以投入到线上使用。经过在线用户的使用和反馈，发现系统中还有一些需要完善的细节，并且系统中还缺少一些必要的引导信息让用户能够更加方便的找到想要的信息，这些都是下一步需要解决的问题。

第七章 结束语

7.1. 全文总结

本论文基于“上下文感知的社会化搜索系统”，设计并实现了一个问答式社会化搜索系统，在该系统中用户不仅仅可以寻求帮助和分享经验，还可以关注流行话题，专业人士和特定问题。该系统融入了 Wiki 的协作编辑精神和用户投票参与的评价机制，还提供了实时搜索机制和推荐机制。

论文首先对问答式社会化搜索系统的相关理论和客户端技术进行介绍；之后详细描述了问答式社会化搜索系统的需求分析，并对基于特定话题的用户影响力计算方法进行了探讨；接下来描述了系统的架构设计和详细设计与实现；最后，通过集成测试和单元测试两方面来对系统进行了验证和分析，表明了整个系统达到了预期的设计目标。

7.2. 未来工作展望

问答式社会化搜索系统是当前流行的最新技术，它融合了很多先进的技术理念，但是同时也会给用户带来一些学习成本，如何更加有效的引导用户融入到这个社区中，如何帮助用户快速的学习这些新的功能，都是本系统需要改善的地方；另外系统需要结合机器学习，智能搜索等自动化技术，提升系统的学习能力，以便提供给用户更好的服务。

7.3. 研究生期间工作

在研究生期间，论文作者参与的工作主要有：

一、参加项目“移动社交网络系统”

- 负责好友模块和即时通信模块的开发工作
- 负责部分模块的测试和项目文档的撰写

二、参加项目“问答式社会化搜索系统”

- 负责 4 人项目团队的协调工作
- 参与前期的技术调研和可行性分析
- 参与系统的需求分析
- 参与系统总体架构的设计和详细设计
- 负责客户端主要模块的设计与实现
- 负责系统的部署和版本控制工作

三、在 IEEE ICC Workshop 上提交一篇学术论文, “An Improved Topic-specific Influence Measurement Model for Social Question Answering Community”

参考文献

- [1] Social Search 定义 http://en.wikipedia.org/wiki/Social_search
- [2] L. Page, S. Brin, R. Motwani and T. Winograd, “The PageRank Citation Ranking: Bringing Order to the Web”, Stanford Digital Library Technologies Project, 1998.
- [3] Google 搜索引擎 <http://www.google.com>
- [4] Bing 搜索引擎 <http://www.bing.com>
- [5] Aardvark 网站 <http://www.vark.com>
- [6] Quora 网站 <http://www.quora.com>
- [7] JSP 定义 <http://zh.wikipedia.org/zh/JSP>
- [8] JQuery 介绍 <http://en.wikipedia.org/wiki/JQuery>
- [9] AJAX 介绍 [http://en.wikipedia.org/wiki/Ajax_\(programming\)](http://en.wikipedia.org/wiki/Ajax_(programming))
- [10] 介绍 JSON <http://www.json.org/json-zh.html>
- [11] D. Horowitz, S. K. Kamvar, “The Anatomy of a Large-Scale Social Search Engine”, Proceedings of the 19th International Conference on World Wide Web, WWW '10, 2010, pp. 431-440.
- [12] D. M. Strong, Y. W. Lee, R. Y. Wang, “Data quality in context”, Communications of the ACM, vol. 40, no. 5, May 1997, pp. 103-110.
- [13] M. Cha, H. Haddadi, F. Benevenuto, K. P. Gummadi, “A few bad votes too many? Towards robust ranking in social media”, in AIRWeb 2008: Proceedings of the 4th International Workshop on Adversarial Information Retrieval on the Web, 2008, pp. 53-60.
- [14] T. H. Haveliwala, “Topic-Sensitive PageRank”, in WWW' 02: Proceedings of the 11th international conference on World Wide Web, New York, 2002, pp. 517-526.
- [15] A. Java, X. Song, T. Finin, B. Tseng, “Why We Twitter: Understanding Microblogging Usage and Communities”, in WebKDD/SNA-KDD '07: in Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis, New York, 2007, pp. 56-65.
- [16] J. Weng, E.-P. Lim, J. Jiang, and Q. He, “Finding Topic-sensitive Influential Twitterers”, in WSDM: Proceedings of the international conference on Web search and web data mining, 2010, pp. 261-270.
- [17] S. Raghavan, M. H. Garcia, “Crawling the Hidden Web”, Proc. 27th VLDB Conf., 2001, pp. 129-138.
- [18] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, J. Wiener, “Graph structure in the Web”, Computer Networks, vol. 33, no. 1, June 2000, pp. 309-320.
- [19] R. Tarjan, “Depth-first search and linear graph algorithms”, SIAM journal on computing, vol. 1, 1972, pp.146-160.
- [20] J. Zhang, M. S. Ackerman, L. Adamic, “Expertise Networks in Online Communities: Structure and Algorithms”, 16th International World Wide Web Conference, WWW 2007, New York, pp. 221-230.
- [21] A. Clauset, C. R. Shalizi, M. E. J. Newman, “Power-law distributions in empirical data”, SIAM Review, vol. 51, no. 4, 2009, pp. 661-703.
- [22] C. Spearman, “‘Footrule’ for measuring correlation”, British Journal of Psychology, vol. 2, 1906, pp. 89-108.
- [23] M. Kendall, “A new measure of rank correlation. Biometrika”, vol. 30(1-2), 1938, pp. 81-93.

致 谢

我首先要感谢我的导师龚向阳教授，在研究生生活期间，无论是学习还是生活方面，龚老师都给予了我很大的帮助；龚老师认真的工作态度和低调的为人处世方法，给我带来了很大的影响，为我在以后的工作和生活中树立了榜样，并且这些知识经验将会是我一生的财富。在此谨向龚老师致以我最真诚的感谢。

在就读研究生期间，王文东教授和崔毅东副教授在科研项目上给予了我很大的支持和帮助，两位老师经常和我们探讨学术问题，指导我的学习，在生活上也教会了我很多做人的道理。在此向两位老师表示最真诚的感谢。同时，阙喜戎副教授也给了我很多的帮助和指导，特别是在生活上，阙老师兢兢业业，帮助我解决了很多困难，在这里也要谢谢阙老师的帮助。

感谢李松、黄川、王茜、田野和韩闻文等师兄师姐，他们在学术上给予了我很大的帮助和启发，谢谢他们无私的帮助。

感谢项目组的同学，特别是陈秋苗、郭亮、宋思奇和黄水桂等同学。我们在项目开发和学术研究中，互相支持，互相帮助，顺利的完成了课题任务。在他们身上我学到了很多新的知识，并且也结下了深厚的友谊。

感谢宽带网中心的所有老师和同学们，谢谢研究生期间你们的陪伴。

此外，我要感谢家人在研究生期间对我的支持，他们辛勤的努力和汗水使得我有今天的成绩，他们的关心和鼓励是我一生中最宝贵的财富。

最后，非常感谢在百忙中悉心评审论文的诸位专家教授。

攻读学位期间发表的学术论文目录

- [1] 王少岩，基于特定域名的 Deep Web 爬虫的设计与 Ruby 实现，发表于中国科技论文在线，署名单位为北京邮电大学