The Research on Focused Crawling Algorithm in Vertical Search Engine

by

CHEN Keqin

A thesis submitted in partial satisfaction of the

Requirements for the degree of

Master of Science

in

Computer Science

in

Central South University of Forestry and Technology

498 Shaoshan South Road, Tianxin District

Changsha Hunan 410004, P.R.CHINA

Supervisor

Professor TAN Junshan

June, 2009

中南林业科技大学

学位论文原创性声明

本人郑重声明：所呈交的论文是本人在导师的指导下独立进行研究所取得的研究成果。除了文中特别加以标注引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写的成果作品，也不包含为获得中南林业科技大学或其他教育机构的学位或证书所使用过的材料。对本文的研究作出重要贡献的个人和集体，均已在文中以明确方式表明。本人完全意识到本声明的法律后果由本人承担。

作者签名：陈可钦

2009 年 6 月 9 日

中南林业科技大学

学位论文版权使用授权书

本学位论文作者完全了解学校有关保留、使用学位论文的规定，同意学校保留并向国家有关部门或机构送交论文的复印件或电子版，允许论文被查阅或借阅。本人授权中南林业科技大学可以将本学位论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

本学位论文属于：

1、 保密□，在年解密后适用本授权书。

2、 不保密□。√

（请您在以上相应方框打“√”）

作者签名：陈可钦 导师签名：

2009 年 6 月 9 日 2009 年 6 月 11 日

摘 要

随着人们对个性化信息检索服务需求的日益增长，面向主题的垂直搜索引擎应运而生。围绕这一社会研究的新热点技术，本文针对垂直搜索引擎中占有重要地位的主题爬虫算法展开研究和讨论。主题爬虫是一种基于主题的信息采集系统，可以从互联网上采集到与主题相关的有用信息，在垂直搜索引擎、站点结构分析等方面取得越来越广泛的应用。

主题爬虫的主要问题是怎样沿着一条好的“路径” 来采集主题相关度高的网页。因此本文研究都专注于主题爬虫的搜索策略，主要分成两部分来讨论——基于 Web 链接结构的策略和基于页面内容的策略。为了充分利用各种文字内容和超链接信息，本文将两者紧密结合在一起，使两种算法可以互相弥补自身的不足。考虑到如何优先采集“ 重要” 网页，本文利用链接重要度权重和主题相关度权重求和计算得到总优先级高低来指导主题爬虫抓取网页。本文研究了主题爬虫系统的基本理论和基本结构，深入分析和探讨了主题爬虫算法，并对算法的两大关键性技术点以及参数进行对比、选择和优化。在页面与主题相关性判定中，引入了文本分类的思想，应用了在自然语言处理中比较成熟的基于向量空间模型的主题相似度计算方法，论文还加入了对链接文本相关度与文本位置权重的考虑。特别地，在 URL 链接的重要性判定过程中，本文在经典的 Page Rank 算法基础上引入类间链接转移概率的概念，即利用类间链接转移概率矩阵来调整 Page Rank 的权值，进而提出了一个基于规则的软主题爬虫的设计方案。这种基于规则的软主题爬行方法借助 Baseline 主题爬虫的架构，应用朴素的贝叶斯分类器并利用主题团间链接的统计关系构造规则找到在一定链接距离内的“未来回报”页面。

最后对该爬虫的性能进行了实验测评，并根据实验结果对该算法的性能进行分析、评价，实验证明本文提出的算法对主题爬虫的爬行收获率有很好的改善并具有很强的穿越隧道的能力。

关键词：垂直搜索引擎；主题爬虫；类间链接；基本主题爬虫；隧道

ABSTRACT

With the ever-expanding information, people become increasingly dependent on vertical search engines. Surrounding the research on this hotspot, the important part of the topic-specific search engine that is called topical crawler is discussed in this paper. Focused crawling is able to crawl particular topical portions of the world wide web quickly without having to explore all web pages. And now, it is more and more widely applied in the fields of topic-specific search engines and analyzing site structure, and so on.

We introduce the structure of rules-based crawler which can download "important" web pages first by counting the sum of URL important value and text relativity value. We give an algorithm we designed. This crawler uses the search algorithm both based on Web hyperlink structure and page content. In the course of the relativity judging between the page content and the topic, the method based on vector space model which is widely applied in the field of the text classification is used. Besides, the influence to weight of anchor text and text locality are taking into account. In the course of the relativity judging between the URL content and the topic, we develop a new method that statistically capture linkage relationships and extract rules among the classes (topics), then guide our focused crawler by using these rules. Meanwhile it has an improvement on Page Rank algorithm.In our work, we started with a focused-crawling approach designed by Soumen Chakrabarti, Martin van den Berg and Byron Dom, called baseline crawler. Building on this crawler, we developed a rule-based crawler, which uses simple rules derived from interclass (topic) linkage patterns to decide its next move.

Initial performance results show that this rule-based Web-crawling approach uses linkage statistics among topics to improve a baseline focused crawler's harvest rate. This rule-based crawler also enhances the baseline crawler by supporting tunneling.

Key Words: vertical search engine; topical/focused crawler; interclass linkage; Baseline focused crawler; tunneling

目 录

1 绪言

1.1 研究的背景和意义

1.1.1 课题的研究背景

当今，因特网的迅速发展、信息量呈指数级增长。搜索引擎是以一定的策略在互联网中搜集、发现信息，对信息进行理解、提取、组织和处理，并为用户提供检索服务，从而起到信息导航的目的，成为网络信息资源检索和利用的主要工具。但随着网络服务的深入和网络信息的迅速膨胀，互联网信息检索技术面临着巨大的挑战。用户需求的信息更为精确化、专业化，尤其是在一些专门化的领域，现存的通用型搜索引擎已不能良好地服务用户，出现了诸多问题。如信息过度重复聚积、垃圾数据混杂、有用信息漏失、信息重复搜索、更新不及时、检索失误等等[1]。一方面是搜索引擎热潮，另一方面，搜索引擎却越来越不适应互联网信息检索的需要。Google、百度等提供的数以万计的查询结果看上去很好，想找到合适的结果却比较难，由传统搜索引擎提供的信息秩序并不能满足互联网发展的要求。

著名学者周立柱[2]总结了通用搜索引擎的局限性：

1）不同领域、不同背景的用户往往具有不同的检索目的和需求，通用搜索引擎所返回的结果包含大量用户不关心的网页。

2）通用搜索引擎的目标是尽可能大的网络覆盖率，有限的搜索引擎服务器资源与无限的网络数据资源之间的矛盾将进一步加深。

3）万维网数据形式的丰富和网络技术的不断发展，图片、数据库、音频/视频、多媒体等不同数据大量出现，通用搜索引擎往往对这些信息含量密集且具有一定结构的数据无能为力，不能很好地发现和获取。

4）通用搜索引擎大多提供基于关键字的检索，难以支持根据语义信息提出的查询。

为了克服通用搜索引擎的以上不足，于是具有针对性强、关注于某个具体领域的垂直搜索引擎应运而生。垂直搜索是针对某一个行业的专业搜索引擎，是搜索引擎的细分和延伸，是对网页库中的某类专门的信息进行一次整合，定向分字段抽取出需要的数据进行处理后再以某种形式返回给用户[3]。尽管垂直搜索很多技术与通用搜索引擎很类似，但是还是有很多自己独特的技术和一些新的需要解决的问题。

1.1.2 研究的目的和意义

垂直搜索引擎的诞生是为了更大程度的提高搜索的“查全率”[4]和“查准率”[5]。垂直搜索引擎也被称为专业或专用搜索引擎，就是专为查询某一个学科或主题的信息而产生的查询工具，专门收录某一方面、某一行业或某一主题的信息，在解决某些实际查询问题的时候比通用搜索引擎更为有效。具体而言，垂直搜索引擎主要通过对行业领域内的信息模型和用户模型进行结构化的搜集或再组织，提供更多、更专业、个性化的行业相关服务。垂直搜索因为其提供了整合深度和广度的行业信息以及更加细致周到的服务，做到了用户要求的专、精、深。因而业界普遍认为垂直搜索将是搜索市场下一个增长点。更有分析人士指出，综合搜索的市场将逐步被垂直搜索所蚕食。

除此现实意义外，本课题的理论意义更值得我们关注。因为本文的主要目的是在互联网上收集用户关心的主题信息，研究范畴属于网络信息采集和数据挖掘领域。

1.2 国内外研究现状及水平

本文大部分的研究都专注于垂直搜索引擎框架下主题爬虫的搜索策略和相关度分析方面，主要是对主题爬虫设计中网页爬行算法进行分析和改进。

现有的主题定制爬行策略和相关算法按照所采用的领域知识和评价链接价值方法的不同，主要分为两大类：基于内容相似度评价的搜索策略和基于 web 结构评价的搜索策略。

前者的主要特点是利用页面中的文本信息作为领域知识指导搜索过程，并根据页面或链接文本与主题（如关键词、主题相关文档等）之间相似度的高低来评价链接价值。这类搜索策略中具有代表性的是 fish search [6]、shark search[7]。fish search 是查询字符串驱动的，该算法仅仅考虑与给定字符串匹配的页面内容的网页以及它们的邻近出链页。shark search 是 fish search 的改进，计算相似度目前比较常用的方法是带权值的 TF-IDF 主题相似度算法——余弦法，将查询串和文档都看成向量，并考虑查询串和文档的长度[8]。

后者的主要特点是利用链接结构信息指导搜索，并通过分析页面之间相互引用的关系来确定页面和链接的价值，通常认为有较多入链或出链的页面具有较高的价值。页面链接分析具有代表性的算法有 Page Rank 和 HITS。

以上介绍的几种都没有运用分类器，仅依赖于基于信息检索的技术来决定相似性。而带有分类器的主题爬行方法有带通道的主题（Focused crawling with tunneling）、语境图（context graph）、Cora、硬聚焦（hard-focus crawling）、

软聚焦（soft-focus crawling）等。

语境图爬虫采用最佳优先启发式搜索策略，并用朴素贝叶斯分类器训练学习层，层表示是与目标类别（layer0）有一定链接距离的页面集合。算法按照分类器的分类分值高低把从 layer-i 页面提取的 URL 插入到 layer-i 队列中，也就是说每一层都保留一个专门的队列[9]。当要决定下一个的爬行页时，爬虫选择 URL 从带最小层标记的第一个非空层弹出，语境图算法支持隧道，但需要构造带后退链接的语境图，开销较大。

Cora 是在计算机科学研究论文这一特定域的搜索引擎。它很依赖机器学习技术，在 Web 上搜索与计算机科学相关的论文，且只能搜索 PS 格式的论文[10]。如果一篇文章含有题名、作者、摘要和参考文献，它就认为是一篇学术论文。然后将 PS 文件转换成文本文件，利用隐式马尔可夫模型来找出题名、作者、摘要和参考文献，并利用统计型文本分类算法将其按照 Yahoo 分类体系进行分类。Cora 思想比较先进，很容易扩展成为其它学科主题的搜索引擎，对 Web 垂直门户网站资源自动建设具有相当重要的意义。但是，Cora 没有在预测 URL 与主题的相关度上作深入研究，也没有对 Web 网页进行采集分析。实际上，由于论文的结构清晰，有很多明显特征，用词规范，所以对 PS 等格式的论文采集的难度要略低于对 Web 网页进行页面主体采集的难度。

硬主题（hard-focus crawling）爬虫的分类器如果发现当前下载页与主题无关，爬虫将不会访问该页面上的任何一个 URL，即算法修剪掉由该页面延伸出去的任何主题发现路径。如果经由分类器返回的网页最高得分的类别不是目标主题或如果评分低于某一阈值（0.5），爬虫将断定该网页与主题无关且停止爬行其出链。因此，该方法可能忽略很多相关网页。

chakrabarti 等提到的软主题(soft-focus)方法，它会把一张抓下来的页面 P 中所有出去的链接 URL 放入待抓列表，这些 URL 的抓取优先级是基于 P 与主题的相关度来计算的[11]。页面 P 与主题的相关度是使用一个分类器来完成。Mukherjea 介绍了一个收集并分析相关网页的系统 WTMS。WTMS 使用向量空间相似度来度量抓取下来的网页与目标主题的相关度，一张网页如果它的与目标主题的相关度越高，那么从它出来的 URL 的优先级越高，放入待抓队列后，将会先被抓取[12]。另外在 fish search 中也采用了类似的思想。这种仅仅利用父页面与主题的相关度来预测子页面与主题的相关度，并以此作为指导的主题爬虫，Altingovde 等称为基本主题爬虫(Baseline Focused Crawler)[13]。

1.3 本论文的主要工作和创新

本文以搜索引擎技术研究为背景，开发了一个基于垂直搜索引擎的主题爬虫

原型系统。首先重点介绍了主题爬虫模型及其重要的算法，对现存主题爬虫算法两大关键技术点的多种选择方案进行了研究对比，总结了各种方案的优缺点和适用度。接着提出了一个改进的基于规则的软主题爬虫算法，该思路是将网页类间链接转移的统计规则和文本相关性的理论结合应用到主题爬虫算法上，然后详细说明了该爬虫算法的实现步骤以及相关设计细节，最后通过实验证明了该算法不仅能采集主题相关的网页，而且能优先采集“更重要” 的主题相关网页。

创新点表现在：

1）给出了 Web 信息搜集中主题搜集策略的主流算法的分析和改进，为其它主题爬虫的开发提供了借鉴。

2）结合算法设计方案和开源项目 Heritrix 提出了主题爬虫的具体实现方法，并基于此方法开发了主题爬虫原型系统。原型系统沿袭了开源项目 Heritrix 易于扩展的实现风格和技术，为系统的二次开发打下了良好的基础。

1.4 本文的结构

第 1 部分为概述，简要介绍了互联网信息检索技术和面临的挑战，目前国内外在该领域的研究状况，以及本文所做的工作。

第 2 部分介绍了垂直搜索引擎的基本概念和通用架构，并对垂直搜索引擎的设计技术做了大体说明。

第 3 部分简要介绍了主题爬虫模型与开源项目的选择，对通用爬虫和主题虫进行了对比，具体阐述了主题爬虫的基本原理、结构和工作流程。

第 4 部分分析了基于 Web 超链接结构和网页文本内容的主题爬虫算法设计的关键点，详细介绍了主题爬虫算法用到的主要技术——网页相关度和链接重要度的计算方法，在此基础上给出了算法描述及优化改进方向，提出了一种新的改进算法——基于规则的软主题爬虫算法并对其设计方案和算法步骤作了详细的论述。

第 5 部分是给出多组对比实验和部分实验数据，最后对实验结果讨论分析。

第 6 部分总结和未来发展展望。

2 垂直搜索引擎研究综述

2.1 垂直搜索引擎

垂直搜索引擎是相对通用搜索引擎的信息量大、查询不准确、深度不够等提出来的新的搜索引擎服务模式，通过针对某一特定领域、某一特定人群或某一特定需求提供的有一定价值的信息和相关服务。其特点就是“专、精、深”，且具有行业色彩，相比较通用搜索引擎的海量信息无序化，垂直搜索引擎则显得更加专注、具体和深入。

用户使用 Google、Baidu 等通用搜索引擎的方式是通过关键字的方式实现的，是语义上的搜索，返回的结果倾向于知识成果，比如文章，论文，新闻等；垂直搜索也是提供关键字来进行搜索的，但被放到了一个行业知识的上下文中，返回的结果更倾向于信息、消息、条目等。比如对买房的人讲，他希望找的是房子的供求信息而不是文章和新闻。

这个特性是他们各自的的技术特点决定的。垂直搜索引擎和普通的网页搜索引擎的最大区别是对网页信息进行了结构化信息抽取，也就是将网页的非结构化数据抽取成特定的结构化信息数据。好比传统网页搜索是以网页为最小单位，而垂直搜索是以结构化数据为最小单位，然后将这些数据存储到数据库，进行进一步的加工处理，如去重、分类等，最后分词、索引再以搜索的方式满足用户的需求。整个过程中数据由非结构化数据抽取成结构化数据，经过深度加工处理后以结构化的方式返回给用户。

举个例子来说明会更容易理解，比如笔记本导购搜索引擎，整体流程大致如下：抓取网页后，对网页商品信息进行抽取，抽取出商品名称、价格、简介……甚至可以进一步将笔记本简介细分成“品牌、型号、CPU、内存、硬盘、显示屏、……”然后对信息进行清洗、去重、分类、分析比较、数据挖掘，最后通过分词索引提供用户搜索、分析、挖掘得到市场行情报告。

垂直搜索引擎的应用方向很多，比如企业库搜索、供求信息搜索引擎、购物搜索、房产搜索、人才搜索、地图搜索、mp3 搜索、……、图片搜索等。几乎各行各业各类信息都可以进一步细化成各类的垂直搜索引擎。现在市面上出现了很多垂直搜索引擎，比如文学搜索、新闻搜索、商品搜索等。根据中国市场研究集团通过调查发现，百度有 20 %的搜索流量同其垂直搜索引擎 MP3 搜索相关[14]。由此可见垂直搜索引擎的重要性。

2.2 垂直搜索引擎架构

通常一个通用搜索引擎的所有模块可以分为两大块：在线部分和离线部分。所谓的在线部分是指不断响应用户搜索请求的部分，这部分也是用户所能接触到的，包括搜索、缓存、网页评分等。而离线部分是为在线部分提供数据准备的部分，这部分包括爬虫、建立索引、链接分析等。垂直搜索引擎在线部分的模块和通用搜索引擎一致，所不同的是离线部分的模块。

一个垂直搜索引擎和通用搜索引擎离线部分最大的不同是爬虫。垂直搜索引擎需要的是一个主题爬虫，它的任务是尽可能的抓取与主题相关的网页，而通用爬虫是抓取所有看到的网页。本文所实现的主题爬虫是垂直搜索引擎的一部分，了解垂直搜索引擎的架构和主题爬虫在垂直搜索引擎中所处的地位，对于理解主题爬虫很有帮助。下面引用一个垂直搜索引擎的通用架构图（如图 2.1）来形象表示两者之间的关系。

另外，垂直搜索引擎还多了一些其他的模块，如页面分类器和信息抽取器。

图中主要部件说明：

1）web 图生成器

对于一个搜索引擎来说，用爬虫把互联网上的网页抓下来并取出网页中的所有链接之后，就是利用这些链接还原一个互联网图，而这个图将在链接分析中用到。还原互联网图的任务就是由 web 图生成器完成的。

2）页面分类器

页面分类是文本分类的一种。文本分类所涉及的问题，是把给定的文本段落(段落或者文件)自动分配到预定的类别中去。由于文本数据的快速爆炸式增长和使用多种方法自动组织和索引大型文本集合的需要，文本分类已经成为一个重要的研究领域。许多标准的机器学习技术已经应用于自动文本分类问题中，学习过程简化为两个简单的步骤：特征提取和在特征向量空间里进行分类学习。在这两个步骤里，特征提取对文本分类得到好的性能非常重要。一旦好的特征提取出来后，几乎任何合理的分类器学习技术都似乎可以执行得很好。

3）信息抽取器

信息抽取/解析主要是要从文本中抽取出特定的事实信息。这些事实信息应该是用户所感兴趣的，而不只是大量文档集合中与用户需求相关的文档列表。它们是系统预先设定好的领域相关的有限种类的信息。因此，信息抽取不能采用传统搜索系统的统计及关键词匹配等技术，因为这些技术都是把文本看成词的集合，并不会对文本进行深入分析理解。信息抽取往往要借助自然语言处理技术，通过对文本中的句子以及篇章进行分析处理后才能完成。总的来说，信息抽取技

术是一种面向具体任务的实用的文档理解技术。与复杂的自然语言理解技术不同，信息抽取技术通常采用浅层的文本分析技术，提取出设计者关注的特定主题的信息。

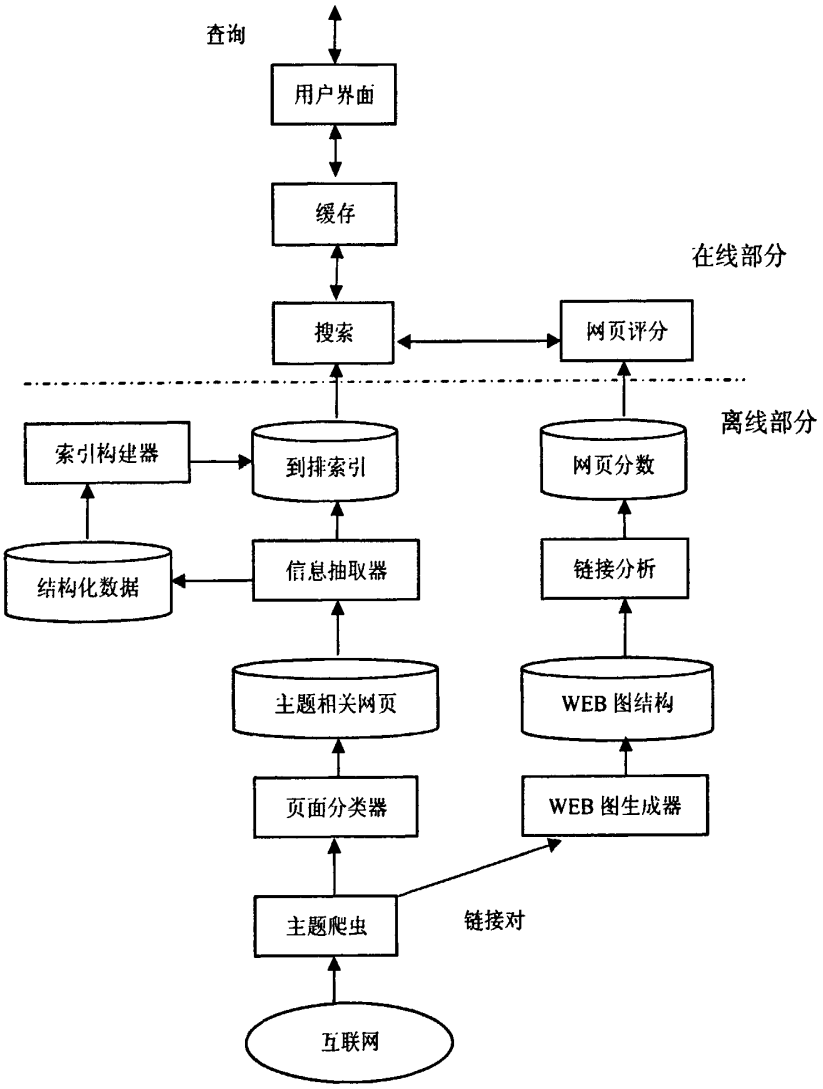


图 2.1 垂直搜索引擎通用架构图[15]

Figure 2.1 Vertical Search Engine System

4）索引构建器

通常现代的搜索引擎应付的用户请求都是关键字查询，对于一个用户的请求，需要扫描所有抓取下来的网页，看看有哪些网页包括这些关键字，并返回给用户这个网页列表。为了提高查询的速度，采用某种数据结构先对这些网页建立索引。目前用的最普遍的建立索引的方法是倒排索引。搜索引擎抓取下来的网页中包括很多单词，一般的逻辑是从网页中找单词，但是这样很慢，解决方法是用单词找网页，即用单词作为索引关键字，每个单词对应的项后面跟着一个网页列表，代表包括这个单词的网页集合，这就是倒排索引。当然搜索引擎的索引数据

结构还包括单词在网页中出现的频率和位置，用频率信息计算单词在网页中的权重，用位置信息生成摘要、组成短语等。

5）网页评分（排序）

网页评分可能是决定一个搜索引擎好坏的最重要部分了。一个好的评分算法能使与用户查询最相关的、最重要的网页排在最前面。所以网页评分的研究内容包括网页与用户查询的相关性度量、高性能的评分算法。

一个商业搜索引擎的评分算法是非常复杂的，要考虑非常多的因素来对一张网页打分。但是最主要的因素有两个：网页的 Page Rank 值，网页与查询的相似度。关于这两个因素论文将在第 4 部分的 4.3 详述。简要地说，一般是用网页 Page Rank 乘以网页和查询的相似值，然后按照这个值对网页进行排序。网页的 Page Rank 值是在链接分析时候计算出来的，可以直接用。而计算网页与查询的相似度一般采用 TF * IDF 方法[16]。还有一个很重要的考虑因素是查询语句中包含的所有单词在网页中是否相邻出现，一般认为这些单词在网页中靠的越近就越有可能是用户想要的网页。而这一点可以用所有单词在网页中出现的位置信息计算出来[17]。

2.3 垂直搜索引擎技术

综合以上对垂直搜索引擎架构的分析可得出，中文垂直搜索大致需要以下技术：信息采集技术——主题爬虫；网页信息抽取技术或元数据采集技术；信息的处理技术（包括重复识别、文本聚类、比较、文本分类、语义相关性分析等）；中文分词；索引。

1）信息采集技术——主题爬虫（Topical/Focused Crawler）

适应特定主题和个性化搜索的主题网络爬虫是垂直搜索引擎的基础和核心。基于垂直搜索的主题网络爬虫在主题搜索中的作用是按照启发式搜索策略从网络中获取主题相关资源，从而在很大程度上缓解了用户使用搜索引擎取得大量无关页面的问题。因此就当今用户对搜索引擎的实际要求来说，对主题爬虫进行研究是十分有用的。本论文主要就是针对垂直搜索引擎中的主题爬虫展开讨论的。其中最重要的问题之一是如何设计一个高效的主题爬虫。

垂直搜索引擎的主题爬虫是相对于通用搜索引擎的通用爬虫而言的。通用爬虫的页面抓取策略相对简单，主要是采用广度优先搜索的思想不停的抓取新的页面，而主题爬虫却是有选择的抓取与特定主题相关的页面。随着垂直搜索引擎的日益流行，作为垂直搜索引擎的关键技术——主题爬虫也显得越来越重要。可以说如果较好的解决了主题爬虫，那么一个垂直搜索引擎也就成功了一半。主题爬虫大致有两个主要问题：一个是 URL 选择策略，表明优先抓取哪些页面；一个是

重访策略，表明什么时候重新抓取页面，看看页面是否发生了变化。本文只解决第一个问题。

2)网页信息抽取技术或元数据采集技术

WEB 结构化信息抽取就是将网页中的非结构化数据按照一定的需求抽取成结构化数据。如房产信息搜索，那就应该抽取出：类型、地域、地址、房型、面积、装修情况、租金、联系人、联系电话……；公司企业信息搜索那就应该抽取出：公司名称、地址、电话、联系人……。

结构化信息抽取有两种方式可以实现，比较简单的是模板方式，还有一种是对网页不依赖的网页库级的结构化信息抽取方式。

web 结构化信息抽取的技术水平是决定垂直搜索引擎质量的重要技术指标。其实 web 结构化信息抽取在百度、Google 早已经广泛应用了，如：百度 MP3、图片搜索、Google 的本地搜索就是从网页库抽取出企业信息，添加到其地图搜索中的,Google 通过这种技术正在颠覆做内容的方式。同样的技术应用还在 qihoo、sogou、购物 shopping 等各种应用中体现。

搜索引擎建立网页索引，处理的对象是文本文件。对于网络爬虫来说，抓取下来网页包括各种格式，包括 html、图片、doc、pdf、多媒体、动态网页及其它格式等。这些文件抓取下来后，需要把这些文件中的文本信息提取出来。准确提取这些文档的信息，一方面对搜索引擎的搜索准确性有重要作用，另一方面对于网络爬虫正确跟踪其它链接有一定影响。

对于 doc、pdf 等文档，这种由专业厂商提供的软件生成的文档，厂商都会提供相应的文本提取接口。网络蜘蛛只需要调用这些插件的接口，就可以轻松的提取文档中的文本信息和文件其它相关的信息。HTML 等文档不一样，HTML 有一套自己的语法，通过不同的命令标识符来表示不同的字体、颜色、位置等版式，提取文本信息时需要把这些标识符都过滤掉。过滤标识符并非难事，因为这些标识符都有一定的规则，只要按照不同的标识符取得相应的信息即可。但在识别这些信息的时候，需要同步记录许多版式信息，例如文字的字体大小、是否是标题、是否是加粗显示、是否是页面的关键词等，这些信息有助于计算单词在网页中的重要程度。同时对于 HTML 网页来说，除了标题和正文以外，会有许多广告链接以及公共的频道链接，这些链接和文本正文一点关系也没有，在提取网页内容的时候，也需要过滤这些无用的链接。例如某个网站有“产品介绍”频道，因为导航条在网站内每个网页都有，若不过滤导航条链接，在搜索“产品介绍”的时候，则网站内每个网页都会搜索到，无疑会带来大量垃圾信息。过滤这些无效链接需要统计大量的网页结构规律、抽取一些共性统一过滤。对于一些重要而结果特殊的网站还需要个别处理,这就需要网络爬虫的设计有一定的扩展性。对于多媒体、

图片等文件，一般是通过链接的锚文本（即链接文本）和相关的文件注释来判断这些文件的内容。

3）中文分词

在搜索引擎和各种语言处理的需要中，分词可以说是最基本的操作。分词是为检索和建立索引服务的。对于计算机，是不可能达到人类的智能的，也不能理解人类语言。但是，由于人类仍然希望计算机能理解人类的语言，并且迫切的希望使用在各种商业和技术领域中，因此提出了计算机形式文法[18]。但是现有形式文法是建立在事先分词的基础上的。对于某些语言，单词之间有特定的符号隔开（一般是空格），所以没有任何分词的困难。而汉语与其他语言都有很大的不同，汉字之间没有空格。如果想继续沿用西方的形式文法理论处理汉语，那么必然涉及到中文分词问题[19]。在系统实现中使用词表分词。词表中文分词的原理，是根据现有词库进行字符串模式匹配，把长的字符串分割为若干个词库中已经存在的词语即可。因此，制作词库成为必须的，词库中词语的选择也要慎重。系统选择的是一个大小为 53301 个中文词语，按照拼音顺序排列的文本格式的词库，词语之间使用回车符隔开。这个词库在使用时，要全部调入内存。为此，使用哈希表来实现。这是因为词库是使用最频繁的公有资源，把词库的调入和查词工作封装到 WordDataBase 类中，这是一个静态类作为公有资源使用，不允许产生多个实例。分词系统同时为 Lucene 索引器[20]、Lucene 查询分析器提供服务。词库选择的好坏，直接影响着 Lucene 的表现。对于 Lucene 来说，是否收入长词语并没有多大关系。因为 Lucene 可以将相对短的词语进行索引，查询时，不会造成什么影响。例如，“中华人民共和国”这个词语并不存在于词库中，而是“中华”，“人民”，“共和国”三个词语存在。在索引时，这三个词语被连续的索引。Lucene 查询分析器将“中华人民共和国”解释为一个短语查询对象（PhraseQuery），由三个 TermQuery 组成，分别是“中华”，“人民”，“共和国”。由于 PhraseQuery 查询要求索引中的词语顺序必须与组成它的 TermQuery 的顺序一致且必须连续，因此在查询时同样能正确的查到。如此一来，似乎可以不再需要词库，直接对每一个汉字做索引就可以了。这样做当然没有任何问题，有些搜索引擎就是这么做的。但是为了查准率，这样做就有一个缺点，即分词中的交叉歧义和包含歧义问题。由于是在无语义的情况下分词，只能作某些字符串运算（实际上是字符串模式匹配）来进行分词，因此出现了不同的分词策略。每一种策略的分词结果可能不同，这完全依赖于词库。所有的分词方法为：前向递增最大匹配分词、前向递减最大匹配分词、前向递增最小匹配分词、前向递减最小匹配分词、后向递增最大匹配分词、后向递减最大匹配分词、后向递增最小匹配分词和后向递减最小匹配分词，共计 8 种方法[21]。系统实现了 4 种前向分词方法，基本上就可以满足需

要。由于现代汉语文章中，充斥着大量的英文单词甚至句子，尤其是计算机方面。鉴于英文是世界上使用最广泛的语言，中文文章中含有另外的语言的可能性不大。英文分词是很简单的，根据空格作为分隔符即可。但是，还有一些特殊的单词必须要考虑。另外涉及到一个重要问题，也是最复杂的一个问题，就是汉字之间的空格如何处理。一般来讲，汉字之间不可能出现空格。在计算机中，有时候为明确区分汉字，也人为地加上空格。处理方法是决不可以把空格当作中文的词语分隔符，因为空格一般恰恰是出现在词语的中间。对于整个分词系统来说，还应该允许用户自由选择需要的词语，即提供过滤功能。系统允许用户设置中文词语，英文词语，中文停止词，英文停止词分别是否要加入结果 Word 列表中。停止词表示一种语言中的大量出现且无关紧要的通用词语，例如助词、叹词和介词等。这些信息预先定义在 WordDataBase 中。对于中文来说，“的”，“地”，“得”等都可以作为停止词[22]。对于英文，则有“this”，“are”，“the”等。

2.4 本章小结

本章从搜索引擎的角度出发，首先对垂直搜索引擎的基本结构、原理和功能进行了阐述，重点介绍了垂直搜索引擎的架构，并对架构中各个重要的模块作了简单介绍，从而引出在垂直搜索引擎中占有最重要地位的主题爬虫。本文所研究的主题爬虫是垂直搜索引擎的一部分。最后对垂直搜索引擎一些重要技术进行了概述，包括网页处理背景知识、超链接分析以及中文分词技术，为下文正式介绍主题爬虫做铺垫。

3 主题爬虫简介

3.1 通用爬虫模型

3.1.1 通用爬虫基本原理

中文搜索引擎的“准”，需要保证搜索的前几十条结果都和搜索词十分相关，这需出“分词技术”和“排序技术”来决定；中文搜索引擎的“全”则需保证不遗漏某些重要的结果，而且能找到最新的网页，这需要搜索引擎有一个强大的网页收集器，一般称为“网络爬虫/蜘蛛”，也有叫“网页机器人”，需要它把网页先抓取下来，按照关键词建立好索引，每次搜索的结果都会直接从搜索引擎建立好索引的数据库中查找，然后把结果返回给访问者。作为搜索引擎的基础组成部分，网络爬虫起着举足轻重的作用，它的性能好坏直接影响着搜索引擎整体性能和处理速度。随着应用的深化和技术的发展，网络爬虫越来越多的应用于站点结构分析、内容安全检测、页面有效性分析、用户兴趣挖掘以及个性化信息获取等多种服务中。

网络爬虫即 Crawler，互联网好比是一个蜘蛛网，Crawler 就是在网上爬行的蜘蛛。网络爬虫在网络中爬行的时候，将 web 上的网页集合看成是一个有向图，从给定的初始 URL 开始，沿着网页中的链接按照一定的策略进行遍历。具体来说，网络爬虫是一个自动提取网页的程序，它为搜索引擎从互联网上下载网页。通用爬虫从网络某一个或多个种子页面开始，读取网页的内容获得初始网页上的 URL，然后通过这些链接地址寻找下一个网页。在抓取网页的过程中，不断从当前页面上抽取新的 URL 放入队列，这样一直循环下去，直到满足系统的一定停止条件为止。

3.1.2 通用爬虫体系结构

根据上一节描述的通用爬虫基本流程，绘制其体系结构图[23]（如图 3.1），其各个模块的主要功能说明如下：

1)页面采集模块：该模块是网络爬虫和互联网的接口，主要作用是通过各种 Web 协议(一般以 HTTP 为主) 来完成对网页数据的采集，然后将采集的页面交出后续模块作进一步处理。

2)页面分析模块：该模块的主要功能是将页面采集模块采集下来的页面进行分析，提取其中满足用户要求的 URL，加入到 URL 队列中。页面链接中给出的 URL 一般是多种格式的，可能是完整的包括协议、站点和路径的，也可能是省略了部

分内容的，或者是一个相对路径。所以为处理方便，一般先将其转化成统一的格式。

3)链接过滤模块：该模块主要是用于对重复链接和无效链接进行过滤。

4)页面库：用来存放已经爬行下来的页面，以备后期处理。

5)URL 队列：用来存放经链接过滤模块过滤得到的 URL，并不断的为页面采集模块提供 URL，当 URL 为空时爬虫程序终止。

6)初始 URL：提供 URL 种子，以启动爬虫。

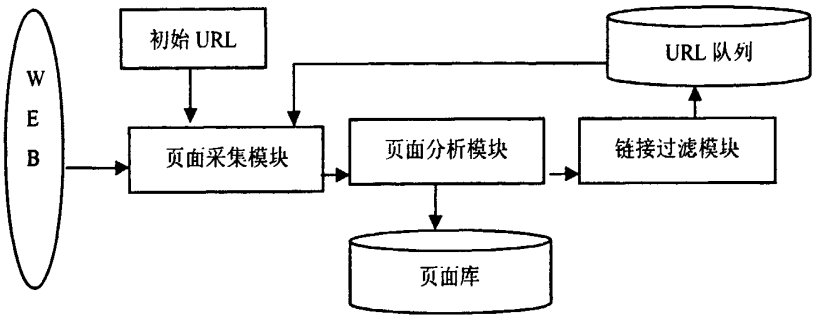


图 3.1 通用爬虫体系结构图

Figure 3.1 System structure of web crawler

3.1.3 爬行方法与选择策略

通用网络爬虫目标就是尽可能多地采集信息页面，而在这一过程中它并不太在意页面采集的顺序和被采集页面的相关主题。这需要消耗非常多的系统资源和网络带宽，并且对这些资源的消耗并没有换来采集页面的较高利用率。主题爬虫则是尽可能快地爬行、采集尽可能多的与预先定义好的主题相关的网页。主题爬虫可以通过对整个网络按主题分块采集，并将不同块的采集结果整合到一起，以提高整个网络的采集覆盖率和页面利用率。

主题爬虫的设计是以普通爬虫为基础的，实际上是对通用爬虫的功能性的扩充。相比于普通爬虫，一个主题爬虫需要能有效地发现主题相关的文档，并且能通过网页内容和链接结构来指导自身的资源发现过程。因此通用爬虫和基本主题爬虫(Baseline 主题爬虫)的爬行策略有显著不同。

网页的抓取策略可以分为深度优先、广度优先和最佳优先三种。传统的网络爬虫是按照预先设定的广度优先策略、深度优先策略或这两种策略相结合的方式爬行网页。这种爬行方式的特点是爬虫按照预先设定的爬行深度爬行网页，它不会理会网页的内容，当爬虫爬行到给定的深度时，爬虫停止工作。所以它搜集到的信息比较全面，但是它的爬行速度较慢，而且有许多是无效的网页。

比较来说，深度优先在很多情况下会导致爬虫的陷入(trapped)问题，目前常见的是广度优先和最佳优先方法。广度优先缺点在于，随着抓取网页的增多，

大量的无关网页将被下载并过滤，算法的效率很低。最佳优先搜索策略按照一定的网页分析算法，预测候选 URL 和计算文本内容与目标网页的相似度或与主题的相关性，并选取评价最好的一个或几个 URL 进行抓取，它只访问经过网页分析算法预测为“重要”的网页。存在的一个问题是，在爬虫抓取路径上的很多相关网页可能被忽略，因为最佳优先策略是一种局部最优搜索算法，导致整体回报率不高。因此需要将最佳优先结合具体的应用进行改进，以跳出局部最优点。研究表明，这样的闭环调整可以将无关网页数量降低 30%至 90%[24]。

3.2 主题爬虫模型

3.2.1 主题爬虫基本原理

主题爬虫的基本思路是按照事先给出的主题，分析超链接和已经下载的网页内容，来预测下一个要爬行的 URL，保证尽可能地多下载与主题相关的网页、尽可能少下载无关网页。它把整个 Web 从逻辑上看作一个有向图 G = (V , E) ，其中图的节点集 V 表示页面的集合，有向边集 E 表示页面之间的超链接。给定一个目标主题，根据页面内容与目标主题的相关度，节点集 V 可以分为两部分：相关集 V+ 和不相关集 V-，主题爬虫的爬行过程可以看作对一个有向图的遍历过程，即从一组节点(种子节点)出发，尽可能多地搜索到那些属于 V+ 集合的节点，同时尽可能避免搜集到那些属于 V- 的节点。

综上所述，主题爬虫需主要解决以下三个关键问题：

1)怎样判断一个已经下载的网页是否与主题相关？对于已经下载的网页，因为我们可以知道它的文字内容，可以采用传统的文本分类技术来实现。

2)怎样决定 URL 的访问次序？许多主题爬虫是根据已下载的网页的相关度，按照一定的原则，将相关度进行衰减，分配给该网页中的超链接，而后插入到优先级队列中。此时的爬行次序就不是简单的以深度优先或者广度优先为序，而是按照主题相关度大小排序，优先访问相关度大的 URL。不同主题爬虫之间的主要区别也就在于它是如何决定 URL 的爬行次序。

3)怎样提高主题爬虫的覆盖度呢？这个问题要解决的就是如何穿过质量不够好(与主题不相关)的网页得到我们所感兴趣的网页，从而提高主题资源的覆盖度。

3.2.2 主题爬虫体系结构

目前主流的主题爬虫发展到现在，其结构要比原始的复杂得多，也有效得多。但一个主题爬虫一般包括以下三个关键组成部分如图 3.2 所示[25]：

1)页面相关度评价器(页面过滤模块)。该模块主要特点是引入了文本分类的思想。在系统爬行之初，页面相关度评价器根据用户输入的关键字和初始文本信息进行学习，训练一个页面相关度评价模型。当一个页面被爬行下来之后，该页面就被送入页面相关度评价器计算其主题相关度值，若该值大于或等于给定的某阈值，则该页面就被存入页面库，否则丢弃。

2)超链接评价器(链接过滤模块)。该模块是主题爬虫的核心模块，主要用于评估从主题相关页面解析出来的URL与主题的相关度，并提供相关的爬行策略用以指导爬虫的爬行过程。URL的超链接评价得分越高，爬行的优先级就越高。反之，若通过一定的评价策略，发现某链接与主题无关，则将该URL及其所有隐含的子链接一并去除，这个过程称之为剪枝。通过剪枝，爬虫就无需遍历与主题不相关的页面，从而保证了爬行效率。但是，剪枝的行为也可能将潜在的与主题相关的页面也剪掉。因此，超链接评价器所用的评价策略的好坏直接影响着爬虫的爬行效率以及爬行质量。

3)爬行器(页面采集模块)。该模块是任何爬虫都不可或缺的通用模块。该模块承担着连接超链接评价模块和页面相关度评价模块的重任。首先，爬行器从待爬行URL 队列中取出超链接得分最高的URL，将该URL相应的网页爬行到本地，然后，将该页面交出页面相关度评价器处理。在整个爬行过程中，爬行的次序和爬行策略都有超链接评价器提供。

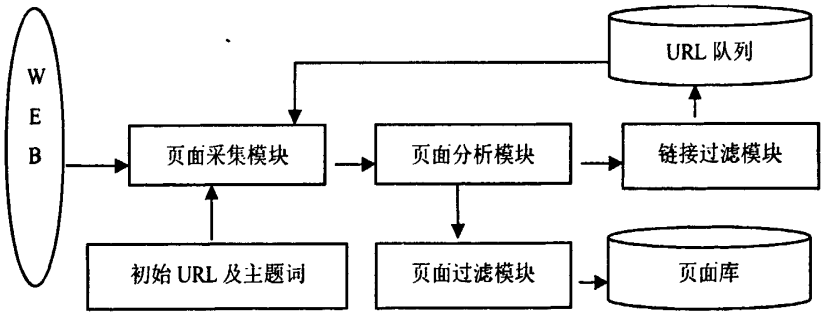


图 3.2 主题爬虫系统结构

Figure 3.2 System structure of topical crawler

3.2.3 主题爬虫工作流程

主题网络爬虫在对网页整个处理过程中，按照事先给出的主题、根据一定的网页分析算法，分析Web中的超链接和已经下载的网页内容、过滤与主题无关的链接、保留主题相关的链接并将其放入待抓取的URL队列中。然后预测下一个要爬行的URL，即根据一定的搜索策略从队列中选择下一步要抓取的网页URL，并重复上述过程，直到达到系统的某一条件时停止。总的原则保证尽可能多地下载与

主题相关的网页、尽可能少地下载无关网页，以此来提高主题爬虫的效率与准确率。

换言之，主题爬虫是按照主题相关度优先的爬行策略爬行网页。主题爬虫有一个主题相关度分析模块，爬虫从网络上抓取到的新网页在爬行之前都要进行主题相关度分析，与初始种子网页相关度最高的网页被优先爬行，而与初始种子网页不相关或相关度不大的网页则被抛弃。因此，与传统网络爬虫相比主题爬虫的工作量大大减少、爬行的准确度大大提高。

由此，在普通爬虫系统基础上，主题爬虫系统对其进行功能上的扩充，增加主题确立模块、优化初始种子模块、主题相关度分析模块和排序模块。主题确立模块用于确立爬虫面向的主题；主题相关度分析模块用来进行网页主题相关度的计算；初始种子模块用于生成目标主题的较好的种子站点，使爬行模块能够顺利的展开爬行工作；主题相关度分析模块是主题爬虫的核心模块，它决定页面的取舍；排序模块是对页面链接的URL的主题相关度进行预测和排序，是决定主题爬虫访问 URL 地址的先后顺序，主题爬虫的爬行策略变为主题相关优先(Topic-First)[26]。其中，初始种子模块和主题确立模块是两个辅助模块，不参与数据流的处理。

主题爬虫系统初始输入，包括种子url、采集范围（可设置范围为本域名下、该目录底下或 WWW 范围三种）、采集深度、采集主题、运行线程数、最大网页大小、网页超时时间以及是否遵守robot exclusion协议等。

本文采用2种方法生成种子页面：①人工指定，即由专家给出相关的种子页面，也称为样板页面(sample URL)；②自动生成，用户指定部分关键词(如“digital library”、“focused crawler”)，并将这些关键词提交给通用搜索引擎(如Google、Baidu) 搜索出的网页，从中选取质量较高的主题URL。

主题爬虫的工作流程结合图3.3说明为：(1)爬虫模块取回网页；(2)调用相关度分析模块，对网页进行相关度分析；(3)爬行模块根据分析的不同结果进行相应的处理；(4)对获得的URL的主题相关程度进行预测并排序；(5)爬行模块从数据库取出等待处理的 URL 继续工作，循环到第一步，直至没有新的URL。

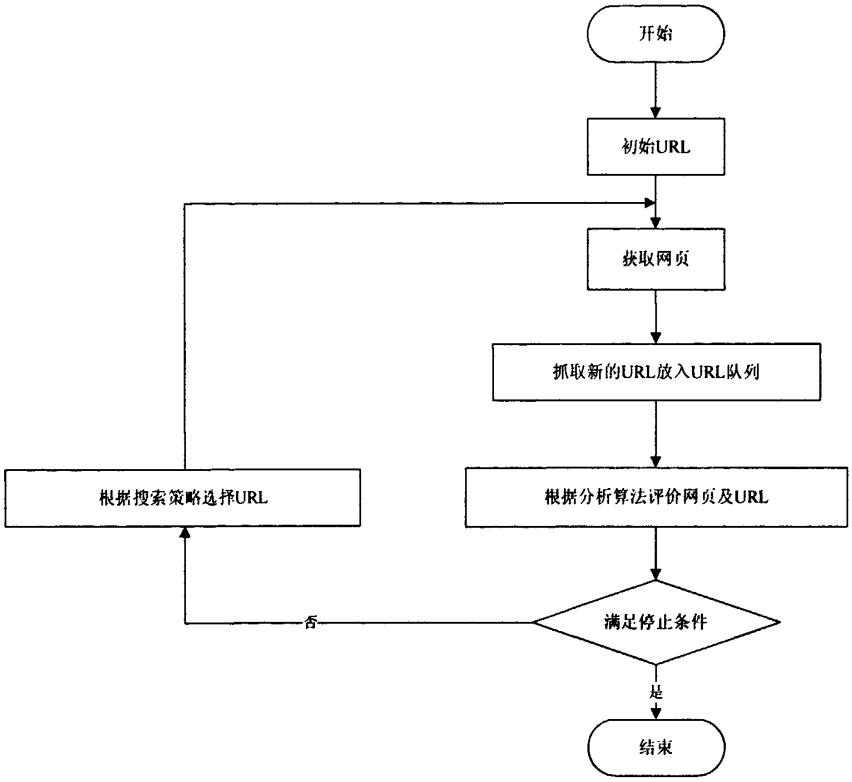


图 3.3 主题爬虫的工作流程图

Figure 3.3 Flow chart of topical crawler

3.3 开源项目的选择

作为大型网络应用系统,主题爬虫的具体实现不仅有着巨大的工作量，并且涉及到很多相关技术的综合应用，任何细节的处理可能都影响着整体的成败。与此相对的是目前存在着很多已成熟的通用爬虫开源项目，它们在系统实现和性能优化方面有着巨大的优势。综合考虑之下，我们认为基于现存开源项目进行主题爬虫的开发是合适的，一切从零开始的作法不仅存在着很大的风险也完全没有必要。

本文提出的基于规则的软主题爬虫的实现是建立在 Heritrix 爬虫[27]之上的，所以介绍这一部分显得很有必要。

3.3.1 Heritrix 架构研究

Heritrix 是由 Internet Archive’s 发起的 Java 开发的一个可扩展并有相关文档的网络爬虫开源项目。Heritrix 最出色之处在于它的可扩展性，在 Heritrix 之上，开发者可以非常方便的建立自己的主题爬虫，可以扩展它的各个组件来实现自己的抓取逻辑。Heritrix 提供了丰富的搜索工具和代码，并且

拥有一个非常灵活的插件机制，我们开发的主题爬虫就是通过修改 Heritrix 实现的。Heritrix 本身相当于一个通用爬虫模型，如果不对其抓取和分析网页的行为进行一定的控制，它是无法达到实际要求的主题网页抓取的。

对 Heritrix 的行为进行控制，需建立在对其架构充分了解的基础上。如图 3.4，Heritrix 主要有三大部件：范围部件，边界部件，处理器链[28]。

1）范围部件：主要按照规则决定将哪个 URI 入队。

2）边界部件：跟踪哪个预定的 URI 将被收集，和已经被收集的 URI，选择下一个 URI，剔除已经处理过的 URI。

3）处理器链：包含若干处理器获取 URI，分析结果，将它们传回给边界部件。

Heritrix 的其余部件有：

WEB 管理控制台：大多数都是单机的 WEB 应用，内嵌 JAVA HTTP 服务器。操作者可以通过选择 Crawler 命令来操作控制台。

Crawler 命令处理部件：包含足够的信息创建要爬的 URI。

ServerCache（处理器缓存）：存放服务器的持久信息，能够被爬行部件随时查到，包括 IP 地址、历史记录、机器人策略。

处理器链（5 个）：

预取链：主要是做一些准备工作。例如对处理进行延迟和重新处理，否决随后的操作。

提取链：主要是获得资源，进行 DNS 转换，填写请求和响应表单。

抽取链：当提取完成时，抽取感兴趣的 HTML，JavaScript，通常那里有新的也适合的 URI，此时 URI 仅仅被发现不会被评估。

写链：存储爬行结果，返回内容和抽取特性，过滤完存储。

提交链：做最后的维护。例如测试那些不在范围内的 URI，提交给边界部件。

本文根据需要主要扩展和定制 Heritrix 的两个部件：

1）向 Heritrix 中添加自己的 Extractor。Heritrix 内嵌的 Extractor 并不能够很好的完成所需要的工作，这不是说它不够强大，而是因为在解析一个网页时常常有特定的需要。比如，可能只想抓取某种格式的链接，或是抓取某一特定格式中的文本片断。Heritrix 所提供的大众化的 Extractor 只能够将所有信息全部抓取下来。在这种情况下，就无法控制 Heritrix 到底该抓哪些内容，不该抓哪些内容，进而造成镜象信息太复杂，不好建立索引。

2）扩展 Frontier Scheduler 来抓取特定的内容。Frontier Scheduler 是一个 Post Processor，它的作用是将在 Extractor 中所分析得出的链接加入到 Frontier 中，以待继续处理。先检查当前链接处理后的结果集中是否有一些属

于高优先级的链接，如果是，则立刻转走进行处理。如果没有，则对所有的结果集进行遍历，然后调用 Frontier 中的 schedule 方法加入队列进行处理。

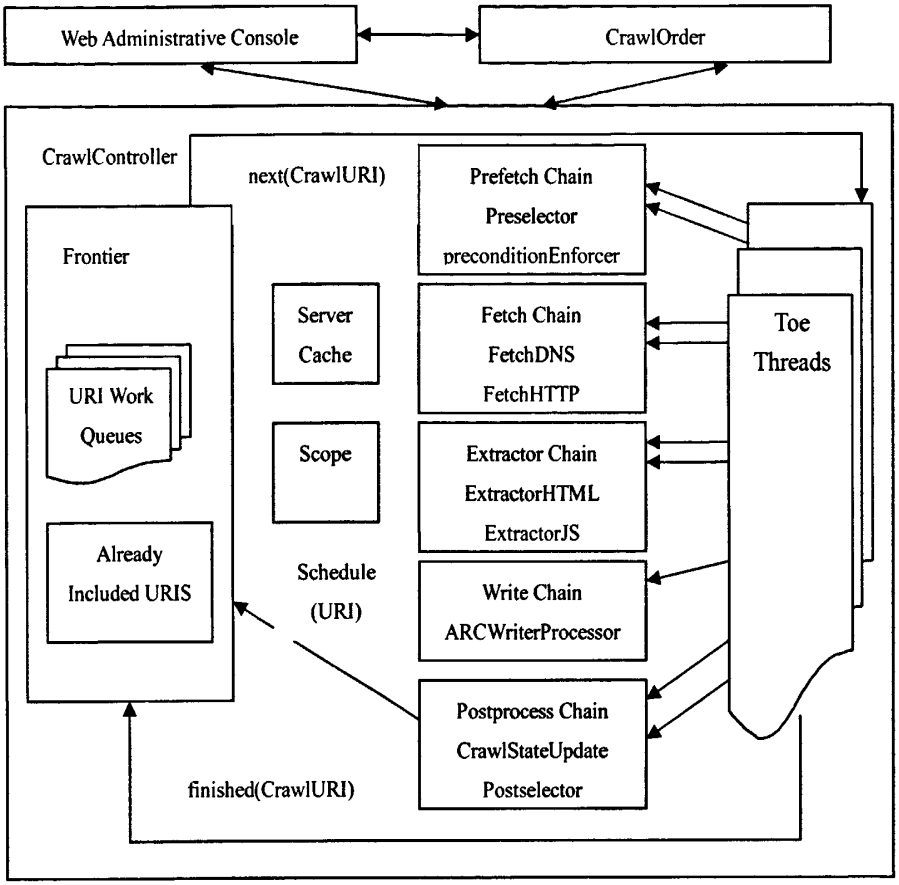


图 3.4 Heritrix 架构示意图

Figure 3.4 Structure of Heritrix

搜索引擎是如此复杂，以至于尽管 Heritrix 本身包含了大量的代码，但是它还是建立在大量其他第三方库之上的。其中最重要的代码库，或者说是页面解析之后的索引工具，这就是 Lucene。Lucene 是一个全文检索代码库，是为 Heritrix 抓取下来的网页提供索引文本和基于关键字搜索的一种工具。下面就简要介绍 Lucene。

3.3.2 Lucene 研究

Lucene 是开放源码的基于 Java 的全文检索引擎，所以在介绍 Lucene 前需提到全文检索的概念。

全文检索是指计算机索引程序通过扫描文章中的每一个词，对每一个词建立一个索引，指明该词在文章中出现的次数和位置，当用户查询时，检索程序就根据事先建立的索引进行查找，并将查找的结果反馈给用户的检索方式。这个过程类似于通过字典中的检索字表查字的过程。全文检索的方法主要分为按字检索和

按词检索两种。按字检索是指对于文章中的每一个字都建立索引，检索时将词分解为字的组合。对于各种不同的语言而言，字有不同的含义，比如英文中字与词实际上是合一的，而中文中字与词有很大分别。按词检索指对文章中的词，即语义单位建立索引，检索时按词检索，并且可以处理同义项等。英文等西方文字由于按照空白切分词，因此实现上与按字处理类似，添加同义处理也很容易。中文等东方文字则需要切分字词，以达到按词索引的目的。

Lucene 的贡献者 Doug Cutting 是一位资深全文索引/检索专家。作为一个全文检索系统，在进行检索之前需要建立索引，索引的过程是先读取文章中的词语，然后一一存放在称为倒排索引文件的索引数据库(Index Database)中。索引数据库记录了词语出现的位置、频率等相关信息，以备后面读取。Lucene 并没有规定数据源的格式，而只提供了一个通用的结构（Document 对象）来接受索引的输入，因此输入的数据源可以是数据库、WORD 文档、PDF 文档和 HTML 文档，只要能够设计相应的解析转换器将数据源构造成 Docuement 对象即可进行索引。对于大批量的数据索引，还可以通过调整 Index Writer 的文件合并频率属性（Merge Factor）来提高批量索引的效率。用户输入查询字符串（Query String），然后经过分析器的分析，就会产生一个 Query 对象。真正搜索时，使用 Index Searcher 类的 search 方法，它返回 Hits 对象。通过遍历 Hits 对象的所有文档(document)，就可以找到所有被搜索到的文章（页面）。查询字符串的语法定义为 Query ::= (Clause)*Clause ::= ["+", "-"] [<TERM> ":"] (<TERM> | "(" Query ")")，中间的逻辑包括 and、or、+、-、&&、||等符号，而且还有短语查询和针对西文的前缀/模糊查询等。总的来说，这是其他很多搜索引擎都不具备的功能。通过修改 Query Parser 的语法生成脚本，还可以修改或扩展查询分析器的功能，使它更加适用于中文环境。所有的问题都通过一个额外抽象层来方便以后的扩展和重用，通过重新实现来达到自己的目的，而对其他模块而不需要。可以简单的应用入口 Searcher、Indexer 并调用底层一系列组件协同的完成搜索任务。所有的对象的任务都非常专一，比如搜索过程 Query Parser 分析并将查询语句转换成一系列的精确查询的组合(Query)，通过底层的索引读取结构 Index Reader 进行索引的读取，并用相应的打分器给搜索结果进行打分/排序等。所有的功能模块原子化程度非常高，因此可以通过重新实现而不需要修改其他模块。除了灵活的应用接口设计，Lucene 还提供了一些适合大多数应用的语言分析器实现（Simple Analyser, Standarded Analyser），这也是新用户能够很快上手的重要原因之一。

作为一套开源的全文检索库，Lucene 具有以下优点[30]：

1）索引文件格式独立于应用平台。Lucene 定义了一套以 8 位字节为基础

的索引文件格式，使得兼容系统或者不同平台的应用能够共享建立的索引文件。

2）在传统全文检索引擎的倒排索引的基础上实现了分块索引，能够针对新的文件建立小文件索引，提升索引速度。然后通过与原有索引的合并，达到优化的目的。

3）优秀的面向对象的系统架构，使得对于 Lucene 扩展的学习难度降低，方便扩充新功能。

4)设计了独立于语言和文件格式的文本分析接口，索引器通过接受流(Toker)完成索引文件的创立，用户扩展新的语言和文件格式，只需要实现文本分析的接口。

5）已经默认实现了一套强大的查询引擎，用户无需自己编写代码即可使系统获得强大的查询能力，Lucene 的查询实现中默认实现了布尔操作、模糊查询、分组查询等等。

3.4 本章小结

本章对通用爬虫和主题爬虫进行了详细对比，介绍了主题爬虫的发展历程并对主题爬虫的结构和功能进行了详细阐述，然后介绍了开源爬虫 Heritrix，主要介绍了 Heritrix 的架构。接着还简要介绍了一个跟 Heritrix 紧密相关的开源项目 Lucene。

4 基于规则的软主题爬虫算法

4.1 主题页面在 Web 上的分布特征

从表面上看，Web 上页面的分布杂乱无章，但是经过研究发现，其实主题页面的分布是有规律可循的，大致总结为四个特征：Hub 特性、主题关联特性、站点主题特性、Tunnel 特性。通过对这些特性的研究，在爬虫进行基于主题的爬行过程中以找到一些对链接预测和页面过滤有用的规律。

1)Hub 特性

美国 Cornell 大学的 J.Kleibnegr 发现 Web 上存在大量的 Hub 页面[31]，这种页面不但含有许多出链，并且这些链接趋向于同一个主题。也就是说，Hub 页面是指向相关主题页面的一个中心。另外，他还定义了权威页面(Authority)的概念，指其它许多页面都认为它是相关于某一主题的有较高价值的页面。好的 Hub 页面一般指向多个 Authority 的页面，而好的 Authority 页面会被多个 Hub 页面所指向。根据这个思想，他还提出了 Authorities and hubs 算法，这个算法在后面章节 4.3.2 有介绍。

2)主题关联特性

在 Hub 特性的基础上，又有人提出了主题关联特性的概念。主题关联特性是指页面所包含的链接趋向于指向和该页面同主题的页面，对于链接到某主题页面的页面，它所包含的其它链接指向的页面也趋向于该主题。这实际上源于 Hub 特性[32]，主要是从页面设计者的角度考虑的。页面设计者一般会把本页面指向于与本页面相关的其他页面。

3)站点主题特性

研究人员发现，一个站点趋向于说明一个或几个主题，并且那些关于同一主题的页面较紧密地在此站点内部链接成团，而各个主题团之间却链接较少。这应该与网站设计者的设计思路有关。每个网站在设计时都有目标，而这种目标一般就集中在一个或几个主题中。而网站的浏览者往往也有目标，他们一般也趋向于浏览同一主题的页面。网站设计者为了满足浏览者的实际需要而将相关内容紧密链接。

4)Tunnel 特性

Web 中主题页面团之间往往要经过很多无关链接才能相互到达。这些无关链接就像一个长长的隧道，因此称为“隧道现象”（Tunnel）[33]。主题爬虫在爬行时，Tunnel 的存在将会对爬行的页面质量、覆盖率和准确度都造成极大地影响. 在设计爬虫的搜索算法时，为了提高爬行页面的准确率，需要提高相关性判定阈

值，这样将过滤掉大量的 Tunnel，但同时也会丢掉 Tunnel 另一端的主题团，从而影响覆盖率。反过来，如果为了提高覆盖率而降低相关性判定阈值，就会混入大量的无关页面，从而影响准确率。为了解决这个问题，在设计链接预测算法时通过给被判定为不相关的链接一个再次被选择的机会，这个机会发生的概率一般要大于 Tunnel 出现的估计概率值。

4.2 隧道技术

通过分析上面所述主题页面在 Web 上的分布情况，主题爬虫应从全局着眼为覆盖尽可能多的主题相关网页，其基本思想是认为与初始 URL 在一定链接距离内的网页具有主题相关性的概率很大。因此主题爬行算法需要解决穿越隧道的问题，穿越隧道是指利用启发式规则来解决简单的全局优化问题，也就是说对于那些不相关的网页的超链接可能也有指向主题相关网页的超链接，因此并不能放弃对它的继续搜索，而是进行若干步后才停止在那条路径上的探索。

本文把获得的与主题相关的页面称为“回报”，将可以通过直接计算相关度得到的页面称为“立即回报”，将主题爬虫不能直接获得必须通过间接计算才能得到的页面称为“未来回报”[34]。提高“未来回报率”是本文研究的出发点。

提到“未来回报”页面就是基于一张主题相关的页面可能需要经过几张主题不相关的页面才能到达的假设。按照 soft-focus 的计算方法，子页面的相关性由父页面的相关性计算得到，因此如果一张父页面的相关性很低，那么它的子页面几乎没有机会被抓取下来，这样也就无法到达可能的主题相关页面。为了克服这个问题，Bergmark 等人[35]提出了一个简单的想法——隧道技术，就是给每张待抓页面设置一个距离值，如果其父页面是与主题相关的，那么它的距离为 0，反之，它的距离是它的上一个主题相关的祖先到它的距离。

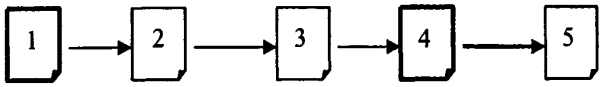


图 4.1 隧道技术图解

Figure 4.1 Demonstration of tunnling

如图 4.1 所示，折角形代表页面，粗折角形代表主题相关页面，细折角形代表主题无关页面，箭头代表一张页面到另一张页面的链接。图中页面 1 的距离值为 0，页面 2 的距离值为 1，页面 3 的距离值为 2，页面 4 的距离值为 0，页面 5 的距离值为 1。系统只抓取距离小于一个阀值的页面。比如如果阀值设为 2，那么页面 3 及其以后的页面将不会被抓取，如果阀值设为大于 2，那么图 4.1 中的所有页面都将会被抓取。所以采用这个方法也能抓取那种需要经过几张主题无关网页才能到达的相关网页。

但是这种方法有一个缺点，如果阀值设置的过大，就跟广度优先搜索没区别；如果阀值设置的过小，很多相关页面也不能到达。

4.3 主题爬虫算法关键点

主题爬虫的主要问题是怎样沿着一条好的“路径” 来采集主题相关度高的网页。论文把大量技术应用到主题爬虫的算法当中，关键点是URL的重要性预测（超链分析）和页面主题相关性判断（文本内容分析）算法。为了充分利用各种文字内容和超链接信息，本文将两者紧密结合在一起，使两种算法可以互相弥补自身的不足，从而提出了一些新的启发策略。本论文就是改进这种文字内容和超链接相结合的方法来指导爬虫的爬行。

4.3.1 页面主题相关性分析可选方案研究对比

主题爬虫主要面向Web 信息，因此主题相关性过滤是指利用页面分析所到的网页正文、链接、链接相关标签属性数据及其它相关内容，对网页进行主题相关判别，从而过滤掉主题无关页面，提高主题爬虫主题信息搜集的准确性。

页面主题相关度的分析是主题爬虫设计的关键。在页面主题相关性的判定过程中，本文引入了文本分类的思想。算法最简单的可以基于关键词进行分析，更深入的可以上升到语义和概念层次。

基于关键词的主题相关度分析的主要思路：首先在领域专家的参与下，确定一组带有权重的能够代表受限领域的关键词，用来表示确定的主题；然后对页面进行关键词提取，通过某种或某些方法计算出主题相关度，从而决定页面的取舍。网页过滤本质上属于文本信息主题判别技术，其基本思想是基于语义内容分析待过滤网页和主题描述之间的相似度。文本信息主题判别是信息检索中一个相对成熟的研究领域。这些方法主要有：全文本扫描、布尔模型[36]、向量空间模型[37]、统计概率模型[38]等。页面主题相关度的计算还有多种方法，例如Naive Bayes[39]、神经网络（Neural Network)[40]等。

全文本扫描虽然有其自身简单直观，使用方便等优点，但是对于大规模信息处理是不合适的。其缺点在于这是一种非常低效的方法，任何字符串的查找都需要对文件进行整篇的遍历。

另一个实现方法相对简单的模型是布尔模型，并且与全文本扫描相比，其计算代价和效率都有着很大的提升。但是布尔模型最主要的缺陷是它不支持设定关键词的相对重要性，在页面主题分析的布尔表达式中，待分析文本中的关键词是不区分性高低的，这就使得其判别准确性不尽人意。

概率模型在文本相关性判别方面的优越性是相当高的，它弥补了其它模型中对关键词之间相互关系的忽略，考虑到了关键词、网页间的内在联系，利用关键词的概率相关性进行网页主题相关性判别。但是对于主题爬虫的应用来说，它有着计算复杂度过高，响应结果过慢的弊病，并且随着获取网页数量的增多，重复进行概率推理网络的修正是必要的，也是耗费极大的。所以概率模型多应用于文本聚类和信息查询。

目前大多数的主题爬虫系统都因为向量空间模型处理能力强和灵活简便的优点而使用它来实现页面的主题相关性判别。相对于布尔模型，向量空间模型使得对关键词中的权重赋值成为可能，从而弥补了布尔逻辑模型将所有关键词视为相同权重的缺陷，使得它的判别准确性大大提高，同时计算复杂度也在合理的范围内。另外，目前大多数的机器学习算法都基于向量操作，采用向量空间模型有利于系统进行其它应用的扩展开发。

综合考虑以上原因，页面主题相关性分析采用了传统的基于向量空间模型算法。这种方法的处理方法比较简单，处理能力较强，是文本分类中常用的一种算法。

文档 D 定义为 D=（T_1，W_1；T_2，W_2；…；T_n，W_n）其中 T_k 是特征项，W_k 是特征项权重。特征项是组成文本内容的基本语言的集合，如字、词、词组和短语等。特征项权重表示特征项在文本中的重要程度。则文档的特征向量为（Wt_1，Wt_2，…，Wt_n）利用 TF-IDF 定义每个特征项的权重为：

$$Wt_k = TFt_k \times IDFt_k = \frac{tf(t_k,p)}{\sum_{i=1}^{n} tf(t_i,p)} \times \log(\frac{N}{n_k}) \qquad \cdots\cdots\cdots\cdots\cdots\cdots (3.1)$$

在一个向量空间模型中，TF 是单词频率的意思，TF_{ij}表示单词 j 在文档 i 中出现的频率。TF 越高说明该词在文章中出现很频繁，能很好的反映文章的主题。IDF 是逆向文档频率的意思，IDFj 表示单词 j 的逆向文档频率 IDFj = log (N/nj)。其中 N 表示所有的文档总数，nj 表示包含单词 j 的文档总数。IDF 越高，说明拥有这个词的文章越少，那么这个词在反映文章的主题时也就很重要。两个文档的相似性就由表示两个文档的向量的余弦值来度量，越接近 1 说明越相近，越接近 0 说明差异越大[41]。

但向量空间模型的 TF-IDF 算法存在着以下问题：算法中的 IDF 函数本质上是为了加大文档的区分度。IDF 函数认为文本频数少的特征项就重要，文本频数多的特征项就无用，这种说法理论依据不足。IDF 函数的简单结构使它不可能更好地反映特征项的重要程度[42]。

目前存在多种基于向量空间模型的训练算法和分类算法。主要有贝叶斯分类

法，支持向量机算法[43]， 线性最小二乘模型[44]，神经网络， K 近邻分类法[15]等。

1）简单向量距离分类法。根据算术平均为每类文本集生成一个代表该类的中心向量，然后在新文本来到时，确定新文本向量，计算该向量与每类中心向量的距离(相似度)，最后判定文本属于文本距离最近的类。

在向量空间模型中，文本和类别都被表示为空间中的一个点向量，文本向量和类别向量之间就存在空间上的距离远近，而这种距离就可以采用向量间夹角的余弦[46]来度量，定义如下：

$$SC(d,c)=\frac{\sum_{i=1}^{n}(w_{di}\times w_{ci})}{\left[\sum_{i=1}^{n}(w_{di}^{2})\times\sum_{i=1}^{n}(w_{ci}^{2})\right]^{\frac{1}{2}}} \quad \cdots\cdots (3.2)$$

2）朴素贝叶斯算法。计算文本属于类别的概率，文本属于类别的几率等于文本中每个词属于类别的几率的综合表达式。计算特征属于每个类别的几率向量。朴素贝叶斯分类算法[47]是一种最常用的有指导意义的方法，它以贝叶斯理论为基础，是一种在已知先验概率[48]与条件概率的情况下的模式识别方法。利用贝叶斯公式通过类别的先验概率和词的分布来计算未知文本属于某一类别的概率。贝叶斯分析方法的特点是使用概率去表示所有形式的不确定性，学习或其他形式的推理都用概率规则来实现。贝叶斯学习的结果表示为随机变量的概率分布，它可以解释为我们对不同可能性的信任程度。

$$P(c_i|d)=\frac{P(c_i)P(d|c_i)}{P(d)} \quad \cdots\cdots (3.3)$$

其中，$P(c_i|d)$为待测文本 d 属于类 c_i 的概率，$P(d|c_i)$ 为类 c_i 中含有文本 d 的概率。计算出文本 d 属于所有类的概率，假设总共有 m 个类，如果在所有 $P(c_i|d)$ $(i=1,2,\cdots,m)$中，$P(c_j|d)$ 的值最大，则文本 d 属于 c_j 类。

3）KNN(K-近邻)算法。在给定新文本后，考虑在训练文本集中与该新文本距离最近(最相似)的K篇文本，根据这K篇文本所属的类别判定新文本所属的类别。

KNN 分类模型的原理如下：首先，对于一个待分类文本，计算它与训练样本集中每个文本的文本相似度，根据文本相似度找出 K 个最相似的训练文本，根据这 K 篇文本的类别来判断待分类文本的类别值[49]。即将待测文本与 K 个文本的相似度对每个类加权平均，求得待测文本与所有类的类别值，类别值最大的类就为文本的归属类。类别值的计算公式如下：

$$W(c_i|x)=\sum_{j=1}^{k}S(x,x_j)p(c_i|x_j) \quad \cdots\cdots (3.4)$$

其中，S (x, x_j)是待测文本与 K 个文本的相似度，x_1-x_k 是训练集中与 x 余弦相似度最大的 K 个文本向量，而 p $(c_i|x_j)$当 x_j 属于类别 c_i 时为 1，否则为 0。其中最重要的是参数 K 的选择，K 过小，不能充分体现待分类文本的特点；而 K 过大，会造成噪声增加而导致分类效果降低。

4）支持向量机。通过某种事先选择的非线性影射把输入向量 X 映射到一个高维特征空间 Z,在这个空间中构造最优分类超平面。

4.3.2 Web 超链结构重要性分析可选方案研究对比

虽然通过以上主题相关度的计算已经能够得到一个比较合理的网页排序,但是为了更有效地提高主题信息搜索的准确率和效率，必须考虑 Web 超链分析。即系统需要对待采集 URL 的重要度进行排序、优先抓取那些重要度高的网页，也可以叫做链接过滤或链接预测。按照高预测值优先采集、低预测值被抛弃的原则对发现的 URL 进行剪枝处理，可以大幅度减少采集页面的数量，极大提高检索结果的质量。

在垂直搜索中网页爬行、关联网页的发现、网页排序优化、检索结果聚类等等都涉及超链分析技术[50]。针对主题搜索网络爬虫而言，如何评价链接对于主题的价值，即链接结构重要度的计算方法，是搜索策略中的关键所在。超链分析能够极大地提高检索结果的相关性。

超链分析算法建立在两个假设之上：①两个网页间存在链接关系表示两个网页之间内容相关；②如果两个网页存在链接关系，那么表明一个网页的作者认为另一个网页是有价值的。Page Rank 算法[51]和 HITS 算法[52]是其中两种影响相当广泛的算法，并在实际中得到了实现和使用。我们主要在此两种算法中进行对比取舍。

(1) Page Rank 算法：

Page Rank 是 google 创始人 Sergey Brin 和 Lawrence Page 发明的用于计算页面重要性的算法，并且应用在其商业化的 Google 搜索引擎中[53]。这种算法可以用一个随机网上冲浪者(surfer)的模型来描述。冲浪者浏览一张页面后，他可以随机选择该页面的一个链接浏览下一张页面，也可以跳到其他页面。这样如果一张页面 P 被很多网页链接，那么 P 被访问的概率也就越大，也就显得 P 越重要。另外如果指向 P 的网页的重要性很高，P 也应该更重要，也就是说 P 的重要性不仅由指向 P 的链接数决定，还由指向 P 的网页本身的重要性决定。

Page Rank 算法基于下面 2 个前提[54]：

前提 1：一个网页被多次引用，则它可能是很重要的；一个网页虽然没有被多次引用，但是被重要的网页引用，则它也可能是很重要的；一个网页的重要性

被平均的传递到它所引用的网页。这种重要的网页称为权威（Authoritive）网页。

前提 2：假定用户一开始随机的访问网页集合中的一个网页，以后跟随网页的向外链接向前浏览网页，不回退浏览，浏览下一个网页的概率就是被浏览网页的 Page Rank 值。

Page Rank 算法定义如下：令 u 为一个网页，N (v) 表示从网页 v 向外的链接数目， B(u)表示链接到网页 u 的网页集合，R (u) 表示网页 u 的 Page Rank 值，C 为规范化因子，作用是保证所有网页的 Page Rank 值总和为常量（例如为保证总的 Page Rank 值为 1，可以通过网页 Page Rank 总和的倒数求得）。那么网页 u 的 Page Rank 值可以利用下面的公式计算：

$$R(u) = c\sum_{v\in B(u)} R(v)/N(v) \qquad \cdots\cdots (3.5)$$

这就是算法的形式化描述，也可以用矩阵来描述此算法。设 A 为一个方阵，行和列对应网页集的网页。如果网页 u 有指向网页 v 的一个链接，则，$A_{uv}=1/N_u$（N_u 表示网页 u 向外的链接数目），否则 $A_{uv}=0$ 。设 R 是对应网页集的 Page Rank 值向量，则有 R = cAR ，可见，R 为 A 的特征根为 c 的特征向量。实际上，只需要求出最大特征根的特征向量，就是网页集对应的最终 Page Rank 值。

后来 Sergey Brin 和 Lawrence Page 改进了算法，引入了衰退因子 E(u)，E(U)是对应网页集的某一向量，对应 Page Rank 的初始值，算法改进如下：

$$R'(u) = c\sum_{v\in B(u)} R(v)/N(v) + cE(u) \qquad \cdots\cdots (3.6)$$

其中，$\|R\|_1=1$，对应的矩阵形式为 V' =c(AV' +E)。

Page Rank 算法的最大弱点是仅仅计算待抓网页的重要度值，并没有考虑网页与特定主题相关性，导致容易出现主题漂移问题。即算法无法区分网页中的超链接是和网页主题相关还是不相关，无法判断网页内容上的相似性，因此用这种算法引导的主题爬虫很容易迷失方向，抓下的网页很少是与特定主题相关的。其实这种算法引导的爬虫不能称为主题爬虫，是一种非主题爬虫。Novak 在其文章中指出，实验表明非主题爬虫抓下来的网页中，与主题相关的网页的比率很快就会下降为 0[55]。

(2) HITS 算法

HITS 算法中引入了另外一种网页，称为 Hub 网页，Hub 网页是提供指向权威网页链接集合的 Web 网页，它本身可能并不重要，或者说没有几个网页指向它，但是 Hub 网页确提供了指向就某个主题而言最为重要的站点的链接集合，比如一个课程主页上的推荐参考文献列表。一般来说，好的 Hub 网页指向许多好的权威网页；好的权威网页是有许多好的 Hub 网页指向的 Web 网页。这种 Hub 与

Authoritive 网页之间的相互加强关系，可用于权威网页的发现和 Web 结构和资源的自动发现，这就是 Hub/Authority 方法的基本思想。

HITS（Hyperlink-Induced Topic Search）算法是利用 Hub/Authority 方法的搜索算法，算法流程如下：将查询 q 提交给传统的基于关键字匹配的搜索引擎，搜索引擎返回很多网页，从中取前 n 个网页作为根集(root set)，用 S 表示。S 须满足以下 3 个条件[56]：

条件 1：S 中网页数量相对较小

条件 2：S 中网页大多数是与查询 q 相关的网页

条件 3：S 中网页包含较多的权威网页。

通过向 S 中加入被 S 引用的网页和引用 S 的网页将 S 扩展成一个更大的集合 T。

以 T 中的 Hub 网页为顶点集 V1，以权威网页为顶点集 V2，V1 中的网页到 V2 中的网页的超链接为边集 E，形成一个二分有向图 SG＝(V1，V2，E)。对 V1 中的任一个顶点 v，用 h(v)表示网页 v 的 Hub 值，对 V2 中的顶点 u，用 a(u)表示网页的 Authority 值。开始时 h(v)＝a(u)＝1，对 u 执行 I 操作修改它的 a(u)，对 v 执行 O 操作修改它的 h(v)，然后规范化 a（u），h（v），如此不断的重复计算下面的操作 I，O，直到 a（u），h（v）收敛。

I 操作：　$a(u)=\sum_{v(v,u)\in E} h(v)$

O 操作：　$h(v)=\sum_{u(v,u)\in E} a(u)$　　……（3.7）

每次迭代后需要对 a(u)，h(v)进行规范化处理：

$$a(u)=a(u)\Big/\sqrt{\sum_{q\in V_2}[a(q)]^2}\qquad h(v)=h(v)\Big/\sqrt{\sum_{q\in V_1}[h(q)]^2}\qquad \cdots\cdots (3.8)$$

通过理论分析和算法实际运行结果比较，可以得到两种算法的区别：

Page Rank 算法是对 WWW 的整体分析，通过模拟用户在 WWW 上对网页的随机浏览对每一个网页计算其 Page Rank 值，因此能够很好地应用到整个互联网规模的具体项目中。该算法是独立于用户查询的，是一种离线的机制，因此可以对用户的请求快速地产生响应，而不会为在线的查询过程付出时间上的额外代价。但是，正因为如此，Page Rank 算法也同时存在着一个显著的问题，即相似度的计算不是针对查询的，它所进行排序的网页是预先下载的，如果最重要的网页不在结果网页集中，Page Rank 算法就无法得到它们了。同样，对于用户基于某个特定主题的查询，在返回结果中一些与主题无关的而权重较高的网页将会排在比较靠前的位置。

Hub 算法是对 WWW 的局部分析，是根据特定的查询产生不同的根集，再根据

根集扩充生成基础集，然后计算网页的 Authority 值和 Hub 值，它迭代灵活性强，结果较为精确，但由于都是基于迭代，速度较慢，响应用户查询要进行大量的运算，比较耗费服务器资源，因此计算量比 Page Rank 大。而且该算法是依赖于用户查询的，实时性差。此外，HITS 算法通常在构造的根集合的过程中，会包含有与主题无关的网页。如果这些无关的网页周围存在很多的链接，那么主题漂移(topic drift)[57] 的情况就会发生，即那些最权威的和最中心的网页可能和原来的查询主题没有关系。同时，用 HITS 进行窄主题查询时，可能产生主题泛化问题[48]，即扩展以后引入了比原来主题更重要的新的主题，新的主题可能与原始查询无关。泛化的原因是因为网页中包含不同主题的向外链接，而且新主题的链接具有更加的重要性。

综合以上所述，Page Rank 算法与 HITS 算法相比有一定的优势。因为不难看出，HITS 算法中由 S 生成 T 的时间开销是很昂贵的，需要下载和分析 S 中每个网页的所有链接，并且排出重复的链接。一般 T 比 S 大很多，由 T 生成有向图也很耗时，需要分别计算网页的 A/H 值，计算量 HITS 比 Page Rank 算法大。虽然 Page Rank 算法已经成功地用于 Google 搜索引擎中，但是有一个问题仍然存在，那就是网页中的每个链接的重要性并非都是一样的，而 Page Rank 算法并没有进行区分。

4.4 基于规则的软主题爬行方法

4.4.1 软主题爬行方法

软主题（soft-focus crawling）[58]由 Soumen Chakrabarti、Martin van den Berg 和 Byron Dom 设计的爬行方法，算法计算出给定页面与目标主题的相关度分值，并平均分配这个分值给从这个页面提取的每一个 URL[59]。这样的爬虫称之为 Baseline 主题爬虫。它结合页面内容和链接结构信息来判定下一步要爬行的重要 URL。Baseline 主题爬虫包括一个规范的主题分类结构和一套主题类及相关示例文档的层次结构，基本部件为文档分类器即朴素贝叶斯分类器。先用分类结构和一整套各类别的示例文档对爬虫的朴素贝叶斯分类器部件进行训练。一旦 Baseline 爬虫分类器建构出其内部模型，它就能判定当前抓取网页的主题分类，例如，给定一个网页，分类器返回一个所有类别的名称和网页与该类别的相关度得分的排序列表，然后选择这个分类结构中具有最高的概率分值的那个类别名称作为该网页的主题。网页的主题相关度可以作为其邻近其他网页的相关度的一个指示器，那就是，半径-1 假说[60]。假说认为，如果网页 u 是某一主题相关示例，且有 u 链接到 v，那么 v 也是主题相关的概率高于随机选择的网页是主题相关的

概率。显然，这种假设是 Baseline 主题爬虫的基础，可指导爬虫的网页抓取工作。

Baseline 主题爬虫的一大缺陷是无能力支持隧道。更具体地说，是朴素贝叶斯分类器无法学习判断出当前看似与主题无关的页面，也能最终指向一个高质量的主题相关页面。举例说，现在正需要查找有关神经网络的文章，搜索的结果有可能是出某一大学的主页链接到其计算机科学系主页再到达相关研究者的文章页面。Baseline 主题爬虫可能附加低的相关度分值，丢失未来主题相关的网页，因为当遍历路径长度增加时，某一分类的一个页面不仅可以指向自己同一分类的页面也可以链接到其他不同类别的网页。比如我们观察到主题为“自行车”的网页，会指向“红十字会”和“急救中心”。同样“HIV/AIDS”页面通常也链接到“医院”类页面[61]。

4.4.2 类间链接转移规则爬行方法

为了克服上述缺点，本文总的思路是借助 Baseline 主题爬虫这个架构提出基于规则的软主题爬虫来判定爬行下一步，提高未来回报率。规则来源于主题类间的链接模式，提取在不同的主题分类间抓取到的前向链接的统计关系作为规则引导主题爬虫，即利用基于分类 A 的页面指向分类 B 的页面的概率为 P 进行启发式搜索。先用朴素贝叶斯分类器把当今爬行的页面分好类，然后根据页面的分类情况计算得出分值表示从这个页面到达目标主题的总概率。然后爬虫按着分值高低把从该页面抓取到的 URL 插入优先队列中。

这种方法首先是训练学习出一个类间链接转移概率[62]矩阵，然后再用这个概率矩阵指导主题爬虫。学习类间链接转移概率矩阵主要分为三个步骤，如图 4.2 所示。

(1)从网页主题目录(如 Yahoo)[63]得到一个类别树以及属于每一类的一些网页作为样例，记这些样例为训练集 0，用这些类别和样例训练一个分类器。

(2)对于训练集 0 中的每一类 C，抓取 C 中页面指向的所有页面，抓取后的页面称为训练集 1。把训练集 1 中所有页面用第一步中训练出来的分类器进行分类得到他们的类别。

(3)于是就知道训练集 0 中的各类页面指向其他类页面的分布情况。这个分布情况可以用一个概率矩阵 T_{mn} 来表示，每个元素 T_{ij} 表示从类别为 T_i 的页面链出去的页面其类别为 T_j 的概率。

有了转移概率矩阵，就可以用它来指导主题爬虫。对于一张新抓取的页面，首先用分类器判断它的类别，然后用矩阵 T_{mn} 来找出其所链出去的网页的类别的分布情况。利用矩阵 T_{mn} 可以知道从这张页面链接出去的是与主题类别相关的页

面的概率为多大，并以这个概率来指导主题爬虫，概率越大越优先抓取。

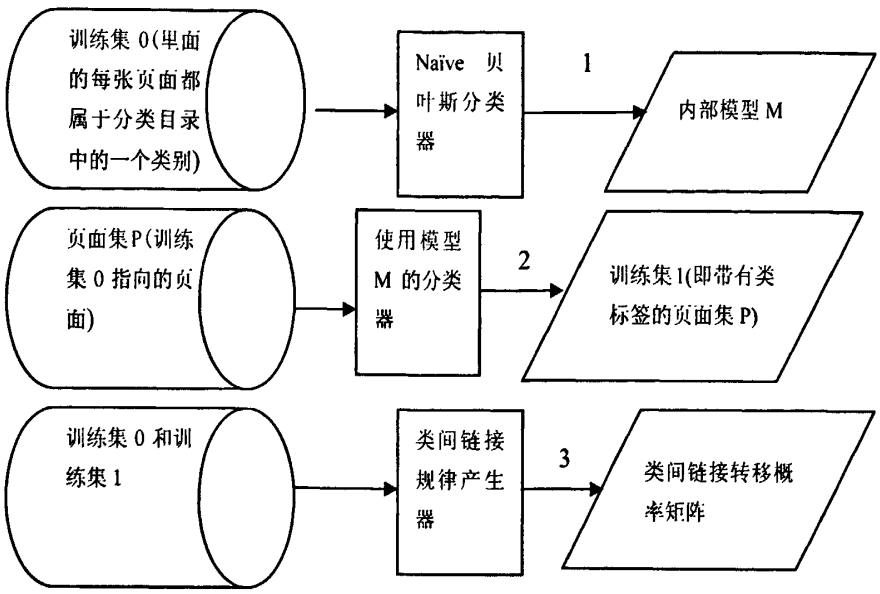


图 4.2 类间链接规律的学习转移概率矩阵的步骤

Figure 4.2 Learnable steps of transfer matrix in interclass linkage patterns

为了实现这个目标，先用分类结构和一整套各类别的示例文档对爬虫的朴素贝叶斯分类器部件进行训练，这种 Web 目录在网上可以找到如 DMOZ 和 yahoo 等网站，这些目录称为 train-0 集合。收集这些相应类别的示例页面所指向的全部网页。这些网页组成 train-1 集合。这时利用已训练好的朴素贝叶斯分类器对 train-1 集合进行分类得到他们真正的分类标注，从而得出 train-0 集合所链出页面的类别分布情况。计算 train-1 集合相关分类的页面个数并产生规则集，即 $T_i \rightarrow T_j$ (X)，表示分类为 T_i 的页面指向分类为 T_j 的概率为 X。当主题爬虫要查找分类为 T_j 的页面时就在其爬行路径上的所有分类为 T_i 的页面上附加概率分值 X 。

为了更好地说明该算法，引入一个简单实例。假设当前的分类结构中包含 4 种类别 ABCD。按上述方法构造 train-0，又假设每一类别分别下载 10 个相关网页构成 train-1，这些新下载页的主题分布情况如下表 4.1。

表 4.1 Train-1 集合中页面按 train-0 中各主题分布情况

Table 4.1 Class distribution of pages fetched into the train-1 set for each class in the train-0 set

A	B	C	D
8 URLs 链接到 B	2 URLs 链接到 A	3 URLs 链接到 A	10 URLs 链接到 D
1 URLs 链接到 C	4 URLs 链接到 B	4 URLs 链接到 B	None
1 URLs 链接到 D	4 URLs 链接到 C	3 URLs 链接到 C	None

由此可以得到分布情况表的类间规则如下表 4.2 所示：

表 4.2 从分布情况表中提取的类间规则

Table 4.2 Interclass rules for the distribution in Table 4.1

A	B	C	D
A→B (0.8)	B→A (0.2)	C→A (0.3)	D→D (1.0)
A→C (0.1)	B→B (0.4)	C→B (0.4)	NA
A→D (0.1)	B→C (0.4)	C→C (0.3)	NA

设定 C 类别为目标主题，其在 train-0 中的页面为种子页面。该种子页面有四个出链，前向链接关系如图 4.3(a)所示。其中 URL5 包含有指向 C 类的链接。

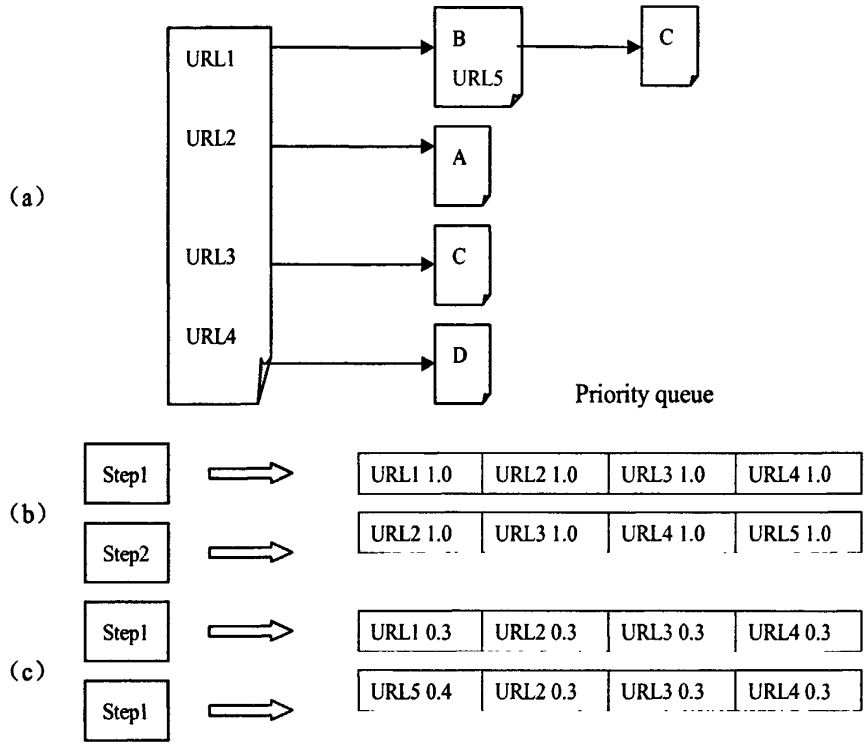


图 4.3 实例图

Figure 4.3 An example scenario

(a)类别 C 的种子页面

(b)Baseline 爬虫的处理步骤

(c)基于规则的软主题爬虫的处理步骤

Baseline 主题爬虫处理过程如图 4.3(b)所示：下载种子页面，提取 4 个超链接并根据种子页面主题相关度（定义为 1.0）把它们插入优先队列，接着弹出 URL1，下载相应页交给分类器评分。根据软主题方法显然 B 类页面的与主题 C 的相关度<1.0，所以爬虫从该页提取到的 URL5 只能插入到优先队列的末端。如果存在其他干扰链接或分类器分配给它的分值相当低的话，URL5 有可能深埋于优先队列中，那它对应的主题相关链接将可能很难召回。

基于规则的软主题爬虫处理过程如图 4.3(c)所示：发现种子页面到另一个主题页面的概率为 0.3，即 C→C (0.3)，所以提取 4 个超链接并赋值相关度为

0.3 把它们插入优先队列，下载 URL1 页面，发现其为 B 类页面。根据规则 B→C（0.4），从该页提取到的 URL5 将插入到优先队列的前端。马上下载其页面并计算相关度，由此得到与主题相关的“立即回报”。

长路径隧道也能利用规则的传递性解决。如队列弹出 URL2 并下载该页面，发现其为 A 类主题，有直接规则 A→C（0.1），也存在 A→B（0.8）B→C（0.4）可以推理得规则 A→B→C 计算得分为 0.8×0.4=0.32（概率的独立性）。当初始类到目标类存在多条路径，爬虫必须合并这些路径。合并函数有求最大数和求和，这里选择求和函数。合并上述路径，则从该 A 类页面提取的超链接将最后赋值相关度为 0.42 插入优先队列中。同时也存在没有规则的时候因为 train-0 集合和 train-1 集合不可能覆盖所有可能的情况。要求得分函数设计为软主题和基于规则的爬行方法结合的方式。规则可以用嵌套递归的方式，允许嵌套的最大深度 2-3。

规则得分（求规则变量 X）函数描叙如下：

if 存在规则（路径长度不超过嵌套的最大深度）

分值=每条路径上单个规则概率的乘积再求总和

else

分值=分类器计算的页面 P 与主题 T 相关度分值

这种基于规则的打分机制独立于页面与主题的相关度，仅依赖于特定页面分类到目标主题页面的概率。它与 Baseline 朴素贝叶斯分类器相比其回报率更高。

4.5 基于规则的软主题爬虫具体实现

以前面的理论分析为基础，本文设计了一个主题爬虫系统来验证基于规则的软主题爬虫算法，而且该系统能作为一个客户端主题信息采集的工具。该系统在设计的过程中主要参照了 source Forge 开源 Web 爬虫项目 Heritrix 和 Apache Software Foundation 的开源搜索引擎项目 Lucene ，采用 java 语言编程。实验的系统还利用 Bow library [64]和 Sleepycat Software (www.sleepycat.com) 提供的一个开源数据库 Berkeley DB[65]，来存储如 URLs、已下载的页面内容、提取的规则、URL 优先级队列等，以及 Rainbow 文本分类器[66]作为默认的朴素贝叶斯分类器。

结合基于规则的软主题爬虫的上述思想，本文提出了一个解决方案，大致由二步组成：

1）规则学习阶段，主要是从初始的主题目录页面集合中学习出一些类间链接转移规则。

2）分类器实现阶段，该阶段利用从规律学习阶段得到的规则结合文本内容

来指导爬虫进行实际的网页抓取。

4.5.1 类间链接转移规则学习阶段

训练样本数据集主要使用 DMOZ 开源网页目录集进行训练，由人工选取一定数量网页作为训练样本集进行主题的描述。开放目录项目(Open Directory Project ，简称 ODP)，也称为 DMOZ(来源于 Directory.Mozilla.org 的简写)。DMOZ 是一个著名的开放式分类目录集合，以人工编辑管理为主，为搜索引擎提供结果或数据。DMOZ 不同于一般分类目录网站利用内部工作人员进行编辑的模式，而是由来自世界各地的志愿者共同维护与建设的最大的全球目录社区。

训练后即可生成主题特征向量，并得到训练集文档总数(total)、相关度阈值(threshold)、(词、词对应的文档频、词权重）向量(W_1，df_1，Wt_1 ，W_2，df_2，Wt_2，…，W_n，df_n，Wt_n)。其中 W_i 为第 i 特征项（词），df_i 为第 i 特征项的文档频，Wt_i 为经过训练处理后得到的主题的第 i 特征项的权重，n 为特征项总数。

试验步骤如前文所述，先构造 train-0 集合，使用 DMOZ 的分类结构和数据，且该结构中如果 URLs 的个数少于预先设定的阈值（为 150）时去掉叶子节点保留双亲节点，然后得到处理后的树形结构的叶子节点作为本实验规范的类别结构。这个过程将产生 1280 个类别，每一类别约为 150 个 URLs，获取到 119000 个页面构造出 train-0 集合。鉴于时间和资源的有限，只下载 train-0 中 266 个与 science、computer、education 语义相关类别的对应出链页，约 40000 个页面，构成 train-1 集合。实验用的目标主题也将从这 266 个类别中选择一个或多个。

接着 Rainbow 文本分类器对 train-0 数据集进行训练，并在 15 分钟内得到统计模型。然后便用该模型对 train-1 数据集进行了约 30 分钟的分类，最后提取出 4990 个规则。

4.5.2 分类器实现阶段

主题爬虫的核心技术是网页的主题相关性判别与链接重要度预测，通过对待爬网页的主题相关度预测，赋予不同优先级，并对其进行排序、过滤和裁剪，以便集中处理主题相关的网络区域，减少资源开销。所以主题爬虫系统的两个主要部分：网页分类器和链接选择器。网页分类器负责学习与更新主题网页模型，计算网页相关度，并过滤不相关的网页；链接选择器负责预测链接重要程度，并由此动态决定主题爬虫的爬行网页的优先级。

对网页和网页中提取的链接进行分类，系统根据不同的分类要求设计了不同的分类器——classifier，在 Crawler 运行之前，要先设置哪些分类器参与网

页的分类，默认的情况是标准分类器(Standard Classifier 类)参与分类，该分类器不分析网页的内容，只分析网页上包含的链接。注册的分类器按相关度优先级先后对网页进行处理，处理之后会给网页做上不同的标记 Label。在对网页和链接做上标记之后，根据这些标记对网页和链接进行处理，相关度和重要度的加权和等于或超过阀值的网页将被存储，反之该网页将被丢弃。同时对于网页中包含的 http 协议的链接，根据网页的相关度、重要度、链接文本等信息，计算该链接 URL 的下载优先权值。

4.5.2.1 页面主题分类器

页面的主题相关性判别算法流程如下：

(1) 预处理阶段：在信息提取之前，先将描述主题的多个页面进行关键词的提取和加权，从而得到该主题的特征向量及向量的权重。

(2) 对页面的正文进行分词程序分词后，去掉停用词、合并数字和人名等词汇、保留关键词，然后按照关键词在文章中出现的频率，使用 TF-IDF 公式对关键词加权处理。

(3) 对页面标题进行分词，将得到的关键词与网页正文中的关键词进行合并，并加重权于得到的标题关键词上。

(4) 根据设定主题中的特征向量对得到的页面关键词进行调整和扩充。

(5) D1 为主题，D2 为待判别的页面，则：

$$\mathrm{Sim}(d_i, d_j) = \cos\theta = \frac{\sum_{k=1}^{n} w_k(d_i) \times w_k(d_j)}{\sqrt{\left[\sum_{k=1}^{n} w_k^2(d_i)\right]\left[\sum_{k=1}^{n} w_k^2(d_j)\right]}} \quad \cdots\cdots\cdots\cdots (4.1)$$

(6) 根据 Sim(D1 , D2) 值的大小和阈值 d 进行比较：如果 Sim(D1 , D2) 大于等于 d，则表示页面与主题相关，保留到数据库中；否则判为不相关，丢弃该页面。

对于直接影响到向量空间模型判别准确性的关键词权重设置方法，一般将关键词分为内容关键词和标题关键词两类。但可根据实际需要进行了相应优化，对其选取和加权都采取了不同的策略。

因此本文对加权的向量空间模型算法步骤（3）进行了进一步改进。考虑位置权重的不同，即按照文本在文档中的位置不同，一般分为标题、摘要、关键词、正文、结论和超链接等 6 个位置，分别赋予不同的加权系数。本文采用的赋值表如表 4.3 所示，来修正的位置权值。

表 4.3 修正的位置权值赋值

Table 4.3 Weight of locality

位置	标题	摘要	关键字	正文	结论	超链接
权值	1	0.8	0.7	0.4	0.6	0.4

权值 W 公式因此调整如下：

$$W_i = \lambda \times tf_i \times \log\left(\frac{N}{n_i}+0.5\right)+\frac{tf_i}{l_i} \quad \cdots\cdots(4.2)$$

其中 λ 为位置加权系数,表示特征项在文档不同位置的加权处理参数。

同时也考虑利用描述链接的文本内容(anchor text)更加精化结果，大大提高了算法的有效性。链接的文本内容参数在后面算法具体实现步骤中会谈到。

4.5.2.2 链接选择器

前面分析提到在传统的 Page Rank 算法中，网络上每一个超链接赋予了相同的权重。这显然是不太正确的。不过 Page Rank 算法遇到的这种问题，是因为 Page Rank 是纯粹的基于链接分析的算法，没有考虑文本内容，所以在 Page Rank 算法的基础上，本文做了进一步改进，提出结合文本相关性的 Page Rank 算法。对于改进的 Page Rank 算法，根据每个超链接的相关页面的一些链接进行计算，为每个超链接赋予不同的权重，这样就将规则与 Page Rank 算法相结合，大大提高了算法的有效性。

$$PR(j)=(1-d)+d\sum_{i\in B_j}\frac{PR(i)}{|F_i|} \quad \cdots\cdots(4.3)$$

考虑了页面的重要性，加入超链接的权重之后，Page Rank 的迭代公式(4.3)就转化成：

$$PR(i)=(1-d)+d\sum_{j\in B_i}PR(j)W(j,i) \quad \cdots\cdots(4.4)$$

其中，W(j, i)=前面类间链接转移概率算法中规则得分函数所计算的总得分值。

4.5.3 主题爬虫算法具体描述

4.5.3.1 索引结构和队列设计

在介绍算法之前，先介绍一下的 URL 存取结构。

需要三种数据结构：

1)线性数组 url_table,用于存放已经找到的 URL。url_table 包括几百万个项，每一个页面都有一项。这个线性数组 url_table 要使用虚存，将不太常用的

项存放在磁盘中，每个项有两个指针。其中一个指向页面的URL，而另一个则指向页面的标题。

2)页面的 URL 和标题的字符串长度都是可变的，存放他们的数据结构是堆（heap)。“堆”是虚存中的一个很大的非结构化的数据块，页面的URL和标题的字符串可不断地追加到堆的后面。

3)URL 的数量太大，为了查找方便采用第三种数据结构，即散列表（hash table)，其长度为n。任何一个URL经过散列函数散列后都产生一个小于n的非负整数。所有散列函数值相同的URL（不妨设散列值为k)，都链接到以散列码k为标识的一个链表中。保证将某个URL放入url_table的同时也将它放入散列表。散列表的主要作用就是可以迅速确定一个URL是否已经在url_table中。搜索的核心是一个递归过程 process_url，它将一个 URL 字符串作为其输入。过程process_url 首先散列URL，看它是否已经在url_table中，若在就继续散列下一个URL。每一个URL只处理一次。若该URL不在url_table中，则读取该页面。然后将该页面的 URL 和标题复制到堆中，并将指向这两个字符串的指针存入url_table。与此同时，URL也要进入散列表。

将关键词制成索引。对于url_table中的每一个项目，检查其标题，并将所有不在“非用词表”(stop list）中的词挑选出来。所谓非用词表就是这些词不应当制作在索引中，它包含前置词、连接词、冠词等以及一些没有什么价值的词。每选择一个词，就在索引中写一行，包括这个词，以及现在对应的 url_table中的项目。可按词进行检索，索引存放在磁盘上。

主题网页下载过程中还存在的一个问题需要注意，即网页去重[67]。一同源网页去重。由于 Internet 中一些权威网页可能会被其他多个网页所链接，且数据下载器一般是循着解析出的超链接按最佳优先顺序下载链接到的网页，因此这些网页可能会被重复下载。二基于网页内容的去重。有些重要网页可能被多家网站同时转载，虽然它们的URL不同(甚至标题也略有不同，仍应认为是同一个内容，不需重复下载。先对网址URL进行哈希散列来进行同源网页[68]的去重。其次，基于主题概念标引的去重判断可以有效去除来源不同但内容相同的转载网页。

为了能够方便的处理链接重要度和主题相关度的计算，本文使用 5 个 URL队列，每个队列保存着统一处理状态的 URL。(1)等待队列。在这个队列中，URL等待被爬虫处理，新发现的 URL 被加入到该队列中。(2)处理队列。爬虫开始处理 URL 时，优先级高URL被传送到这一队列，为了保证同一个 URL 不能多次被处理，当一个 URL 被处理过后，就被移送到错误队列、抛弃队列或者完成队列。(3)错误队列。如果在下载网页时发生错误，它的 URL 将被加入到错误队列中，一旦移入错误队列，爬虫不会对它作进一步处理。(4)抛弃队列。如果下载网页

没有发生错误，且经过主题相关度的计算小于阈值，则放入该队列，一旦移入抛弃队列，爬虫不会对它作一步处理。(5)完成队列。如果下载网页没有发生错误，且经过主题相关度的计算大于阈值，就要把从中发现的 URL 放入等待队列，处理完毕把它加入到完成队列，到达这一队列将等待排序模块的处理。同一时间一个 URL 只能在一个队列中，这也叫做 URL 的状态。网页 URL 就这样完成从一个状态转换到另一个状态。

4.5.3.2 算法步骤及流程

本文提出的基于规则的软主题爬虫算法步骤及算法描述遵循了设计流程图 4.4。

步骤如下：

1) 初始化种子集合里 url 的 priority 值，将其放到 url 下载对列 fetchQueue 当中

2) 启动下载网页的爬虫线程，通过主题描述文档生成主题向量(当 fetch Queue 不为空且满足用户设置的其他限制时)

3) 下载线程从 fetchQueue 中取 url 并下载网页，解析网页，提取链接。

4) 计算该网页的相关度和重要度

5) 存储主题相关的网页

6) 根据网页的相关度和重要度、链接描述文档等信息计算各子节点的 priority 值

7) 过滤子节点，将剩余子节点的 url 插入到 fetchQueue 中，priority 值大的 url 靠前

8) 计算 fetchQueue 中各个节点还能继续往前访问的深度，转步骤 3

9) 当 fetchQueue 为空或网页达到下载数量或用户在操作界面中按“停止”按钮时结束。

其中两个重要步骤的有如下详细解释。

第 4 步：

网页的相关度=分类器计算的页面 P 与主题 T 相关度分值

网页的重要度=该网页的改进后 Page Rank 值

第 6 步：

url 的 priority 值决定了 url 下载的先后顺序，本算法中子节点 url 的 priority 值的决定因素有：

1) 相关度的权重：该值由两部分决定，一是继承自父节点网页的相关度的值，因为主题提取算法的研究显示，相关网页指向的网页往往也有较高的主题相关度；二是 url 自身的链接文本的主题相关度，链接文本信息往往包含了它所指

向的网页的主题描述，因此是相当重要的。

2）重要度权重：该值也由两部分决定，一是继承自父节点网页的重要度的值，因为层次结构好的网站，网页在网站中的层次越高，该网页往往是越重要的，而且根据 Page Rank 和 HITS 等算法的思想，由重要网页所指向的网页往往也是重要的[68]。二是 url 自身的链接结构。

Priority 计算过程如下：

(1)计算相关度权重，用 textScore 表示：

textScore =FACTOR1 *父节点相关度+ (I-FACTOR1)*链接文本相关度

链接文本相关度=链接文本中包含的主题关键词在主题向量中的权重之和/链接文本包含的总词数

链接文本相关度的计算思想是：链接文本中主题关键词数占的比例越大，链接文本相关度越高；链接文本中主题关键词在主题向量中的权重越大，链接文本相关度越高[69]。

FACTOR1 是两者的权重因子，值在 0 到 1 之间。FACTOR1 值越大表示继承的相关度对计算越起决定作用；FACTOR1 值越小，则表示链接自身的相关度对计算越起决定作用，在实验程序中，该值是可以方便调整的。

(2) 计算重要度权重，用 structureScore 表示：

structureScore =FACTOR2*父节点重要度+ (I-FACTOR2) *链接重要度

链接重要度= $PR(i)=(1-d)+d\sum_{j\in B_i}PR(j)W(j,i)$ 其中 W(j,i)为 Rainbow 分类器训练后规则得分函数所计算的总得分值

FACTOR2 是两者的权重因子，值在 0 到 1 之间。FACTOR2 值越大表示继承的重要度对计算越起决定作用；FACTOR2 值越小，则表示连接自身的重要度对计算越起决定作用。在实验程序中，该值是可以方便调整的。

3）通过重要度权重和相关度权重计算总权重：

Priority=FACTOR*textScore+(1-FACTOR)*structureScore

FACTOR 是两者的权重因子，值在 0 到 1 之间。FACTOR 值越大表示相关度权重对计算越起决定作用，FACTOR 值越小，则表示重要度权重对计算越起决定作用。在实验程序中，该值是可以方便调整的。

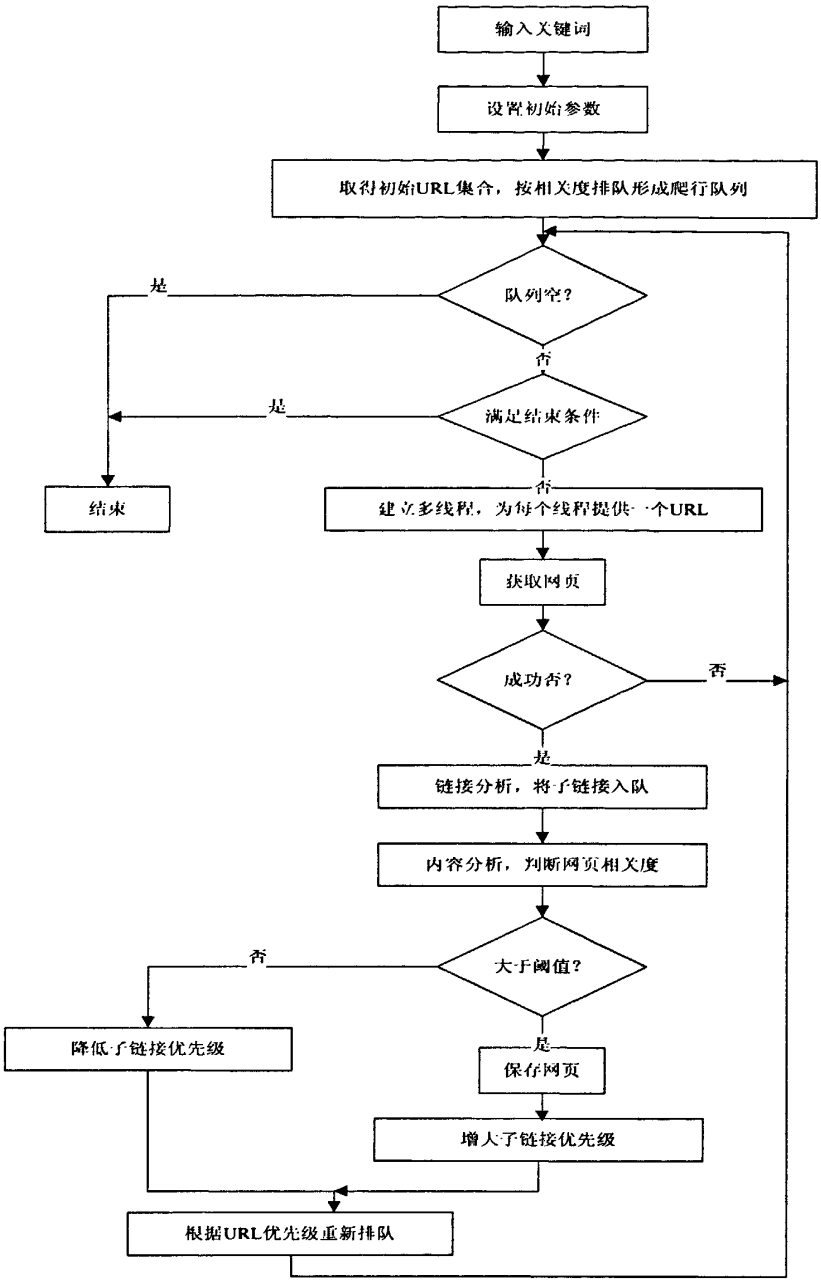


图 4.4 基于规则的软主题算法细化流程演示图

Figure 4.4 Flow chart of rule-based crawler

4.5.3.3 主要的 Java 类

为了实现系统主要流程，也即实现本文提出的主题爬虫算法，本文进行了大量的系统设计和编码工作，算法的主要功能由以下这些 java 类来实现。

1) Work Bench 类：实现了用户操作界面，用户可在该界面中进行主题信息采集的个性化设置，然后启动爬虫，同时可在爬虫采集信息的过程中对其进行控制。该界面通过演示图、URL 列表和统计显示爬虫运行的情况。

2) 爬虫类：该类描述爬虫的信息，控制整个信息采集流程。系统的其它功能类将直接或间接地被该类调度，实现信息采集。用户对信息采集的控制由该类实现。类中 visit(Page page) 方法根据用户定义的网页处理操作来处理网页，比如保存还是删除网页；shouldVisit(Link link) 根据 link 的权值、协议等信息判断是否下载该 link 指向的网页。

3) Page 类：一个 Page 对象表示一张下载并解析后的网页，该对象包含系统需要的所有该网页的描述信息，如网页标题、网页包含的链接、网页的相关度和重要度等；该类中 download 方法实现网页的下载功能，在下载网页的同时调用 HTML Parser 类来解析网页。

4) Link 类：一个 Link 对象表示一个链接，该对象包含系统需要的所有该链接的描述信息，如链接的 URL、协议、该链接指向的网页、链接文本、域名信息等。该类中包含分析链接 URI 结构的方法。

5) HTML Parser 类：解析 Html 网页，得到描述网页的各种信息，如去除了 html 标记的网页文本，各种链接等等，解析后的信息都保存在 page 对象中。

6) Stardarded Classifier 类：标准分类器，该分类器不分析网页的内容，只分析网页上包含的链接，给链接做上不同标记，如主页链接，同域名链接，协议为 ftp 的链接，图像链接等等。同时它还判断一个链接是否已经被访问过，计算网页包含的链接数和根据链接的信息计算网页的重要度值。

7) My Classifier 类：主题分类器，该类调用 Topic Builder 类建立主题向量，计算网页和该主题的相关度值，还有链接文本和主题向量的相关度值。(注分类器都继承 Classifier 接口，要实现两个方法：一是 Classifier (Page page) 对网页进行分类；二是 priority() 定义该网页 URL 的优先级，分类器按优先级大小先后对网页进行处理，处理之后会给网页做上不同的标记 Label。

8) Topic Builder 类：该类中 buildTopic 方法通过中文分词解析主题文档，用 Hashmap 统计分词结果，去除干扰词，得到主题向量；buildVector 方法建立网页相对该主题的向量模型。ItextRel()方法计算链接文本和主题向量的相似度。

9) Similarity 类：定义各种计算相似度的静态方法。

10) Priority 类：计算 link 的下载优先级。优先级的计算公式在 4.5.3.2 的算法描述当中。实验中 3 个权重因子取值分别为 FACTOR1=0.5；FACTOR2=0.5；FACTOR=0.8。

11) Link Analyzer 类：统计分析互联网上链接结构信息，比如网页包含链接数的分布，某域名底下网页的平均链接数，最大链接数等，这些信息用于给设计链接重要度计算公式作参考。

4.6 本章小结

以何种爬行方式访问 Web 能提高搜索回报率,并提高主题查询的查全查准,为此本章展开了面向主题的网络蜘蛛搜索策略的研究，并在 Baseline 主题爬虫系统的基础上,提出详细介绍了一种改进的基于规则的软主题爬行方法。特别地,在 URL 与主题的相关性判定过程中引入了朴素的贝叶斯分类器并利用主题团间链接的统计关系构造规则能找到在一定链接距离内的“未来回报”页面。

5 实验测试和性能分析

5.1 评价指标

在本文中，采用信息检索领域广泛使用的查准率(precision)和召回率(recall)来评价实验结果。查准率和召回率反映了信息获取系统的效力，也就是衡量信息获取系统获取相关文档同时阻止不相关文档的能力。一般来说信息获取系统的效力越高，其满足用户需要的能力也就越高。其定义如下：

查全率：所有主题网页中被正确判别出来的比率，反映了主题判别的完备程度。

查准率：所有被判别为主题的网页中真正是主题的比率，反映了拒绝非主题样本的能力。

召回率[70,71]是指检索出的相关文档数与检索出的文档总数的比率，衡量的是检索系统(搜索引擎)的查全率。作为本实验的评价指标之一，公式如下：

$$Recall=Nr / Nt \qquad \cdots\cdots (5.1)$$

其中，Nr 表示当前爬行的网页集中主题相关网页数，Nt 表示整个网络中主题相关网总数。如果在爬行了同等数量网页的情况下，算法的目标召回率高，则也说明算法的精确性高。

另外一个评估主题爬虫的性能常用的简单度量为收获率(harvest ratio)或（crawling precision）[72,73]。它是获取页面与目标主题的平均相关度（relavent ratio），公式描述如下：

$$HR=\frac{\sum_{i=1}^{N} Relevance(URL_i,T)}{N} \qquad \cdots\cdots (5.2)$$

其中 Relevance(URL_i,T)是由分类器返回的页面与目标主题的相关度，N 是爬行页面的总数。

5.2 页面主题相关性实验

这里先介绍为这次评测比较建立的两个爬虫：一是广度优先搜索爬虫，一是软主题爬虫。这两个爬虫都是建立在 Heritrix 爬虫之上，由于本文设计的基于规则的软主题爬虫是只抓取属于指定主题列表中的网页，所以为了具有可比性，本文把广度优先搜索爬虫和基本主题爬虫设计成也只限于抓取属于指定主题列表中的网页。为了便于进行评价，预先设定一些种子页面（URLs）和目标页面

（URLs）。除了 DMOZ 上搜集的部分资源作为主题的训练样本网页集外，另外选取一些和训练样本网页不重叠的网页作为目标页面。Google 提供了一些基于 SOAP 和 WSDL 标准的 API，主要包括页面 Cache，检查单词拼写，以及获取链向某页面的 URLs，即 Inlinks 等功能[74,75]。本文就是通过对各个目标 URLs 为参数迭代调用 Inlinks API 来选取种子 URLs。迭代调用的次数可以看成种子到目标之间的距离。最后为每个目标选取相应的一个种子，并且保证从种子到目标的通路没有坏链接。

Heritrix 爬虫原本就是采用广度优先搜索的，所以在 Heritrix 爬虫之上建立实验想要的广度优先搜索爬虫是最自然不过的。只对 Heritrix 爬虫作了尽可能少的更改，仅修改了抓取的种子页面。采用 Naïve Bayes 分类器的软主题爬虫则通过 textScore 公式计算结果作为 URL 的优先级来指导爬虫抓取网页。

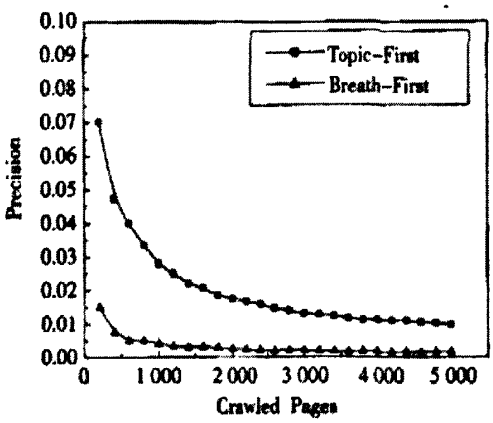
实验参数：实验中选取了 50 个目标 URLs ，并且调用 Google API 获得 50 个种子 URLs。种子到目标的距离是 3 ，即对每个目标 URLs 迭代调用了 3 次。搜索深度= 3(设的较小,为了防止搜索规模过大) ,线程数= 100(要求在网络环境较好的情况下)，阈值 r =0.1，中文分词主要以中科院计算所免费版的分词工具(C 语言开发)为基础。机器配置：P4 2.4G CPU ,内存 1G。实验结果如表 5.1 所示。主题爬虫的精度和耗时可以通过阈值进行调节，阈值设的越大精度就越高，耗时就越小，因为随着阈值的增大，要求主题相关度增大，需要处理的页面就减少了。

表 5.1 实验结果

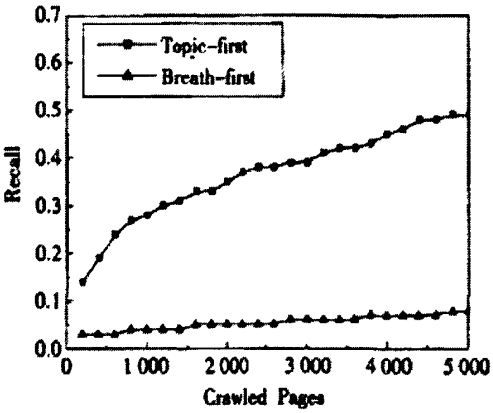
Table 5.1 Comparison of topic crawling and breath crawling

名称总计	软主题爬虫	普通爬虫
提取文档	2644	5406
提取失败	541	1452
拒绝文档	3	5
发现文档	3466	6350
收集数据总数	121464483 字节	246541867 字节
总搜索时间	01352	01694
索引	92	176
实际爬行时间	0633	0247

绘制实验对应的组图 5.1 显示在各自爬行了 5000 个页面后，主题爬虫的爬全率达到 48 %，而以广度优先爬虫只有 10%左右。从上面分析可以看出，采用朴素贝叶斯分类器的软主题爬虫不管是爬准率还是爬全率上都明显优于广度优先的爬虫。



a) 爬准率曲线



b) 爬全率曲线

图 5.1 采用改进分类器的软主题爬虫与广度优先通用爬虫的性能比较

Figure 5.1 Comparison of Soft-focus crawling and Breath-first crawling

5.3 收获率对比实验

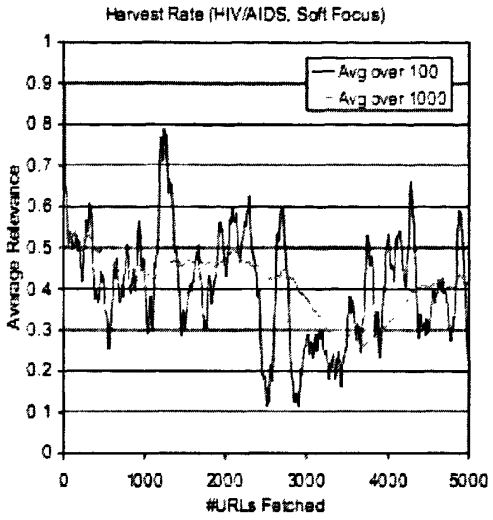
为了检测基于规则的软主题爬虫算法的性能，本文做了两个对比实验。

1)把它与其它两个爬虫程序进行了比较：一个是广度优先的爬虫程序；另一个是 Diligenti 等人（2000 年）开发的传统的软主题爬虫程序[76,77]。

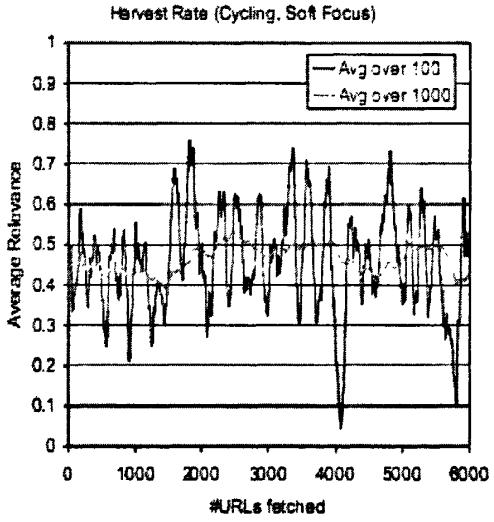
本文选择了“HIV/AIDS”和“Cycling”作为评价的主题，在分类器训练数据的准备中，限制了种子文档集的规模为 30。所有的词汇从 HTML 文档中抽取，但不包括 HTML 的标记和属性。所有的与主题相关的网页通过通用的基于文本搜索引擎所返回的前 30 个结果所生成。

作为实验算法性能的参照，先绘制了软主题爬虫的局部收获率图。这里的局部收获率是最近 100 张或 1000 张抓取的页面中与主题相关的平均比率。从上图可以看出，目标主题“HIV/AIDS”的每 100 张页面的局部收获率维持在 0.1 与 0.8 之间，而每 1000 张页面的局部收获率维持在 0.3 与 0.5 之间。目标主题

“Cycling”的每 100 张页面的局部收获率维持在 0.1 与 0.7 之间，而每 1000 张页面的局部收获率维持在 0.4 与 0.5 之间。



a)主题“HIV/AIDS” 收获率



b）主题“Cycling”的收获率

图 5.2 软主题爬虫的局部收获率图

Figure 5.2 Local Harvast Ratio of soft-focus crawling

接着本文在原来软主题算法的基础上加入了统计类间转移规则的贡献，提升爬虫的主题页面搜集能力。为了反映在这一方面的改进和更清楚地反映系统所采用的算法的性能，做了下面一组对比试验。在这个实验中，每隔 100 页记录下该时刻爬虫所下载的相关页面数和页面总数，并计算出相关率绘制图 5.3。通过 3 次爬行相关率曲线比较发现，广度优先搜索方法收获率维持在 0 与 0.2 之间，而软主题方法平均 100 张页面的收获率维持在 0.2 与 0.7 之间，本文提出的改进算法相比其他两类爬虫有着相当大的优势，并且随着搜集的不断进行，优势成扩大

趋势。另外根据第三次搜集结果可以看出，类间转移概率矩阵对于指导爬虫爬行方向确实有着重要的作用，可以减少无关网页的抓取概率，从而提升爬虫的整体搜集能力。

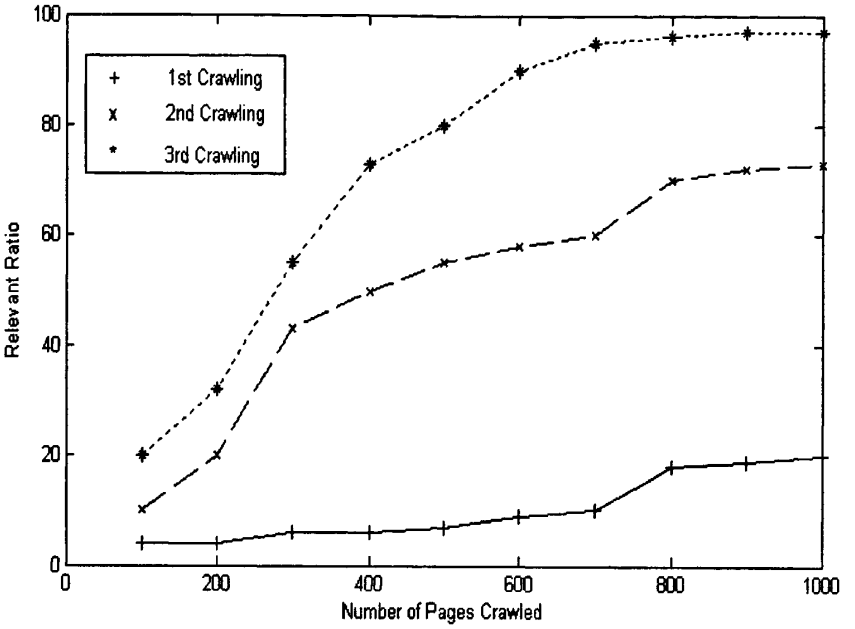


图 5.3 搜索算法的总体收获率对比图

Figure 5.3 Comparison of global Harvest Ratio

1st crawling　　采用广度优先搜索的收获页面相关率曲线图

2nd crawling　　采用软主题方法收获页面相关率曲线图

3rd crawling　　采用改进的基于规则的软主题方法收获页面相关率曲线图

2)最后改进的基于规则的软主题爬虫和 Baseline 主题爬虫单独比较。目标主题为 Top.Science.Physics.Quantum Mechanics，Top.Computer.History，和 Top.Computer.Open Source。它们经过提取的规则条数分别是 41，148 和 212。对于每一类主题构造两个不同的种子集合，分别从 DMOZ(http://dmoz.org/)以及 yahoo 目录(http://dir.yahoo.com/)获取并保证每个集合 10 个种子 URLs 页面。由实验得出的收获率表 5.1 可知，在特定主题和种子集合质量优劣条件下页面链接密度的不同，将导致收获率因主题与种子集合的改变而变化。并且实验结果明显，基于规则的软主题爬虫比 Baseline 主题爬虫性能优越 3%-38%。

表 5.2 基于规则软主题爬虫与 Baseline 主题爬虫收获率比较

Table 5.2 Harvest ratio comparison of the baseline and rule-based crawlers

	种子集 1			种子集 2		
目标主题	Baseline 主题爬虫	基于规则软主题爬虫	改善率 (%)	Baseline 主题爬虫	基于规则软主题爬虫	改善率 (%)
Quantum mechanics	0.28	0.3	7.1	0.25	0.29	16
Computer history	0.29	0.4	38.0	0.36	0.37	3
Open source	0.52	0.56	9.0	0.48	0.61	27

接着为了评估种子集规模大小对页面爬行结果的影响，在 Google 中搜索关键字“open” 和“source”，用其搜索得到的前 50 个结果的 URLs 来构造种子集合。结果发现其收获率与表 5 第 open source 栏用种子集 1 所得数据相近，绘制当获取页面数不断增加时的两种爬虫的对比图如图 5.4 所示。由图可知两种爬虫都成功抓取到主题相关网页，但当获取页面超过 500 个以后，基于规则的软主题爬虫明显优于 Baseline 主题爬虫。

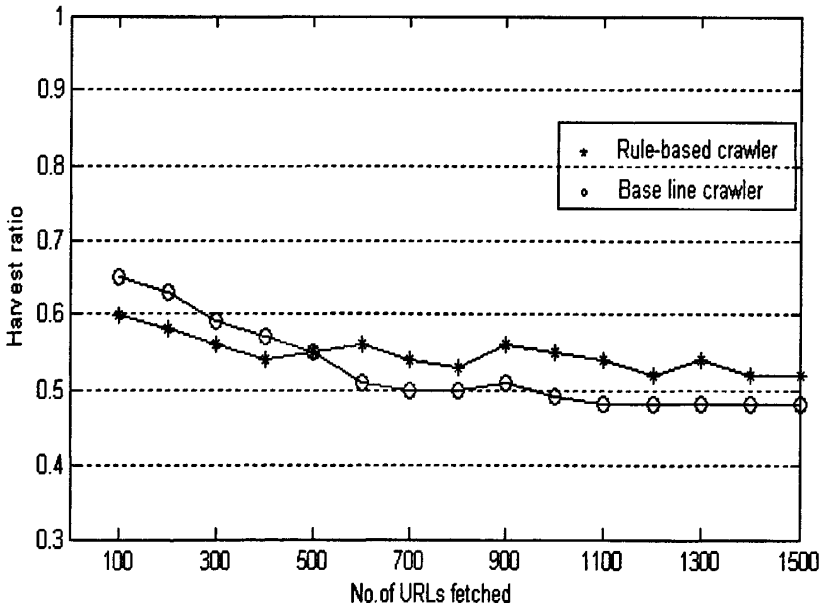


图 5.4 Baseline 主题爬虫和基于规则爬虫的收获率对比

Figure 5.4 Harvest rates of Rule-based crawler and Baseline focused crawler

由实验数据图可以得到这样的结论，本论文所改进的算法有较好的精确度和较高的隧道穿透率，获得的未来回报率很高。虽然有网速等问题的存在，本系统的下载速度也不高，但所得到的准确度应该比单纯追求下载速度更能提高系统的性能。

6 总结和未来发展展望

因特网的迅速发展对万维网信息的查找与发现提出了巨大的挑战。为了满足大多用户提出的与主题或领域相关的查询需求，本文提出了基于垂直搜索引擎主题爬虫的新研究思路和方法。

目前的面向主题的搜索引擎主要采用基于领域知识分析的“启发式”搜索策略，即先通过在线获得的领域知识评价待访问链接的价值，借以推断信息资源的分布情况，然后按最好优先原则选择价值最大的链接进行下一步的搜索。这种搜索引擎搜寻最优行动选择序列的主要困难在于，在整个搜索任务完成之前，网络爬虫对 Web 搜索空间中信息资源的整体分布总是未知的。虽然目前的启发式搜索策略可以借助某些领域知识对信息资源的分布情况做出某种程度的估计，并据此推断出大致的搜索方向，但这种估计存在明显的“近视性”，即估计只能近似地反映信息资源在未搜索空间边缘附近而不是其内部的分布情况。产生近视问题的主要原因是领域知识仅来源于网络爬虫对已搜索空间中的页面文本信息或 Web 结构信息的统计和分析，其本身不但可能是粗糙的、非精确的，而且具有局部性的特点。这种估计的“近视性”表现为按现有的评价方法得到的链接价值仅是一种“局部价值”。因而，按最佳优先策略选出的链接只可能是“局部最优”链接，由此决定的行动也只可能是一种“局部最优”行动； 反之，有些链接虽然表现出的价值不高，但可能预示着更多的“未来”回报，因而可能具有较高的“全局价值”。如何穿过质量不够好(与主题不相关)的网页得到我们所感兴趣的网页，提高主题资源的覆盖度，就是本文研究的重点。

本文展开了对主题爬虫搜索策略和爬行算法的研究，并在 Baseline 主题爬虫系统的基础上，提出了一种基于规则的软主题爬行算法。在页面与主题相关性判定中，引入了文本分类的思想，应用了在自然语言处理中比较成熟的基于向量空间模型的主题相似度计算方法，论文还加入了对链接文本相关度与文本位置权重的考虑。特别地，在 URL 的重要性判定过程中引入了朴素的贝叶斯分类器并利用主题团间链接的统计关系构造规则调整 Page Rank 的权值，找到在一定链接距离内的“未来回报”页面，经由实验证明该方法对主题爬虫的爬行收获率和覆盖率有明显的改善效果。

本论文所设计的爬虫在搜索策略等方面都做了许多尝试， 但因为时间、 精力和水平有限，本论文的研究和下一步所要做的工作仍然有很多。

1)计算网页主题相关性的方法是用文档相似性的判断和关键词比较，可以从基于关键词转化为基于知识而得到进一步发展，对知识有一定的理解与处理能力。

2)在本文所用到的算法中，一些必要参数的设定往往需要根据系统实际运行效果去做适当的调整，但是还没有找到最理想的值，这些需要在今后的实践中不断完善。

3)无论是基于主题的爬虫还是普通的爬虫，对动态网页的获取仍然是一个难点和重点，有待于进一步深入研究。

未来主题网络爬虫的研究主要是围绕如何提高链接主题预测的准确性，降低计算的时空复杂度，以及增加主题网络爬虫自适应性[78,79]这几个方面展开。提高链接价值预测的准确性一直是近年来研究的焦点。将各类评价方法相结合，尤其是基于分类器的主题相关度预测和基于在线训练、反馈的主题爬行方法值得进一步训究。将目前信息检索领域中的概念检索理论[80]应用于链接价值的计算，是一个新的尝试方向。网络爬虫的爬行具有重复性，如何将 Web 动态变化的规律与先前搜索的统计结果相结合，以提高价值计算的准确性，是一个值得研究的问题。降低网络蜘蛛在训练、搜索过程中的计算复杂性，也是有待进一步研究的问题。目前的网络爬虫通常采用固定的搜索策略，缺乏适应性，如何提高网络爬虫的自适应性有待进一步研究[81]。

参考文献

[1]卢敏. 垂直搜索——我专故我在[J]. 软件世界，2006，2：20-23.

[2]张文，吴娟. 垂直搜索：搜索引擎的激情所在[J].电脑爱好者.2006，(10)：110-113.

[3]张京伟.搜索市场新宠儿——垂直搜索[J].电子商务，2007，(5)：12-13.

[4]周赟.垂直搜索引擎Spider技术的研究和应用[D]. 广州：中山人学硕士学位论文，2007：20-23.

[5]李勇，韩亮. 主题搜索引擎中网络爬虫的搜索策略研究[J]. 计算机工程与科学，2008，(30)：202-205.

[6]刘金红，陆余良. 主题网络爬虫研究综述[J].计算机应用研究，2007，(24)：23-27.

[7]严宏伟，何俊. 基于房源分析系统的垂直搜索引擎关键技术的探讨[J]. 中国科技信息，2007，(5)：121-124.

[8]李晓明，闫宏飞，王继民. 搜索引擎——原理、技术与系统[J]. 北京：科学出版社，2005，(7)：2-3.

[9]印鉴，陈忆群，张钢. 搜索引擎技术研究与发展[J]. 2005，31（14）：54-56 .

[10]陈刚，卢炎生. 教育网 BBS 搜索引擎设计与实现[J].微计算机信息.2006，32：98-102.

[11]崔泽永，常晓燕.搜索引擎的 WebRobot 的技术与优化[J] .微机发展，2004，14(4)：100-102.

[12]陈旭春、赵明生. 分布式多搜索引擎系统的研究与实现[J]. 微计算机信息. 2005，20：270-275.

[13]洪光宗，王皓. 搜索引擎 Robot 技术实现的原理分析[J]. 现代图书情报技术. 2002，(1) ：290-292.

[14]李少波，谢庆生. 基于ASP的动态联盟制造资源管理框架研究[J]. 中国机械工程. 2005, 16(6)：502-506.

[15]谢庆生. 我国制造业 ASP 发展的模式与发展策略[J]. 数字制造科学. 2004 1(1)：66-70.

[16]蔡铭，林兰芬，董金祥. 制造资源智能检索系统研究与实现[J]. 计算机辅助设计与图形学学报， 2004， 16(4)：543-548.

[17]张博锋，周传飞，方爱国，等. 基于构件技术的制造资源搜索引擎研究与实现[J]. 2005， 26(9)：2268-2270.

[18]张英杰，曹岩. 基于网络化的制造资源智能管理系统的研究[J]. 西安交通大学学报， 2004， 38(3)：270-273.

[19]何汉武，郑德涛，陈新，等. 面向虚拟企业构造的合作企业搜索方法研究[J]. 计算机集成制造系统， 2005， 11(6)：862-868.

[20]孙茂松，左正平，黄昌宁. 汉语自动分词词典机制的实验研究[J]. 中文信息学报， 2000， 14(1)：20-22.

[21]吴栋,滕育平.中文信息检索引擎中的若干技术[J].计算机应用，2004，24(7)：128-131.

[22]李刚，宋伟，邱哲. 征服 Ajax+Lucene 构建搜索引擎[M]. 北京:人民邮电出版社，2006. 193-308.

[23]刘涌锋. 搜索引擎中主题爬虫算法的研究[D]. 广州：中山大学硕士学位论文，2005：275-282.

[24]黄豫清. 从 Web 文档中构造半结构化信息的抽取器[J]. 软件学报 2000 (1)：270-272.

[25]刘玮纬. 搜索引擎中主题爬虫算法的研究与实现[D]. 南京：南京理工大学硕士学位论文，2006：27-30.

[26]王斐.基于增量反馈和自适应机制的主题爬虫系统的设计与实现[D].南京：南京理工大学硕士学位论文，2005：70-72.

[27]李卫，刘建毅，何华灿等. 基于主题的智能 Web 信息采集系统的研究与实现[J]. 计算机应用研究， 2006，23(2)：163-166.

[28]Eichmann D. The RBSE Crawler-Balancing Effective Search Against Web Load[C]//Proc of the 1st Int' l World Wide Web Conf，1994:113-120.

[29]McBryan O A. GENVL and WWW: Tools for Taming the Web[C]//Proc of the 1st Int' l World Wide Web Conf，1994: 70-90.

[30]Pinkerton B. Finding What People Want: Experiences with the Web Crawler[C]//Proc of the 2nd Int' l World Wide Web Conf，1994.

[31]Cowie J，Lehnert W. Information Extraction[J]. Communications of the ACM， 1999，(1):80-91.

[32]LAWRENCE SGILES L. Accessibility and distribution of information on the Web[J]. Nature，1999，400(8): 107-109.

[33]CHO J， CARCIAM H. The evolution of theWeb and implication for an incremental crawler[C] //Proc of the 26th International Conference on Very Large Databases (NVLDB-00). 2000.

[34]BREWINGTON B E，CYBENKO C. How dynamic is the Web[C] //Proc of

the 9th International World Wide Web Conference. 2000.

[35]MENCZER F, PANT C, RUIZ M E. Evaluating topic-driven Web crawlers[C] //Proc of SIGIR' 01. NewOrleans, Louisiana: [s.n.], 2001: 241-249.

[36]S. Chakrabarti, K. Punera, and M. Subramanyam, "Accelerated Focused Crawling through Online Relevance Feedback," Proc. 11th Int' l World Wide Web Conf. (WWW 02), ACM Press, 2002, pp. 148-159.

[37]CHO J, GARCIAMH, PAGEL. Efficient crawling through URL ordering[J]. Computer Networks and ISDN System s, 1998, 30(1-7): 161-172.

[38] DeBRA P, HOUBEN G, KORNATZKY Y, et al. Information retrieval in distributed hypertexts[C] //Proc of the 4th RIAO Conference. New York: [s.n.], 1994: 481-491.

[39] HERSOVICIM, JACOVIM, MAAREK Y S, et al. The shark-search algorithm: an application: tailored Web site mapping[C] //Proc of the 7th International World Wide Web Conference. Brisbane: [s.n.], 1998: 65-74.

[40]李雪梅. 个性化搜索引擎的研究[J] . 南京大学学报. 2003: 3-4.

[41]杨建林. 基于本体的文本信息检索研究[J]. 南京大学计算机学院. 2006: 2-3.

[42]周立柱, 林玲. 聚焦爬虫技术研究综述[J]. 计算机应用, 2005, 25(9): 1965-1969.

[43]宋峻峰, 张维明, 肖卫东等. 基于本体的信息检索模型研究[J]. 南京大学学报:自然科学版, 2005, 41(2): 189-197.

[44]陈康, 武港山. 基于 Ontology 的信息检索技术研究[J]. 中文信息学报, 2005, 19(2): 51-57.

[45]刘群, 李素建. 基于"知网" 的词汇语义相似度计算[C]. 第三届汉语词汇语义学研讨会论文集. 台北: 2002. 59-76.

[46]潘春华, 武港山. 面向主题的 Web 信息收集系统的设计与实现[J]. 小型微型计算机系统, 2003, 24(12): 2150-2154.

[47]李盛韬, 赵章界, 余智华. 基于主题的信息采集系统的设计与实现[J]. 计算机工程, 2003, 29(17): 102-104.

[48]欧阳柳波, 李学勇, 李国徽等. 专业搜索引擎搜索策略综述[J]. 计算机工程, 2004, 30(13): 36-37.

[49]M. Diligenti et al. "Focused Crawling Using Context Graphs," Proc. 26th Int' l Conf. VeryLarge Data Bases (VLDB 2000), Morgan

Kaufmann, 2000, pp. 527-534.

[50]S. Chakrabarti, M.H. Van den Berg, and B.E. Dom, "Focused Crawling:A New Approach to Topic-Specific Web Resource Discovery," Computer Networks, vol. 31, nos. 11-16, pp. 1623-1640.

[51]曹军. Google 的 PageRank 技术剖析[J]. 情报杂志, 2002, 21(10):15-18.

[52]JingH, BarzilyR, McKeownK, etal. Summarization evaluation methods experiments and analysis[C] // AAAI Intelligent Text Summarization Workshop. Stanford, CA:AAAI, 1998. 60-68.

[53]Muslea I. Extraction Patterns for Information Extraction Tasks:A Survey[C]. AAAI-99 Workshop on Machine Learning for information Extraction, 1999.

[54]Eikvil L. Information Extraction from World Wide Web——A Survey[R]. Norwegian Computer Center, Tech. Rep: 945, 1999-07.

[55]World Wide Web Consortium: The Document Object Model [EB/OL]. http://www.w3.org/DOM, 2004.

[56]Chang Chiahui, Lui Shaochen. IEPAD: Information Extraction Based on Pattern Discovery[C]. Proceedings of the Tenth International Conference on World Wide Web, Hong Kong, 2001-05.

[57] S. Chakrabarti et al. "The Structure of Broad Topics on the Web," Proc. 11th Int' l World Wide Web Conf. (WWW 02), ACM Press, 2002, pp. 251-262.

[58]R Gaizauskas, Y Wilks. Information Extraction: Beyond Document Retrieval. Computational Linguistics and Chinese Language Processing , 1998;(2).

[59]S Brin. Extracting Patterns and Relations from the World Wide Web. International Workshop on the Web and Databases (WebDB' 98) , 1998; (3). 306-315.

[60]C H Chang , S C Lui. IEPAD : Information Extraction Based on Pattern Discovery. In the Proceedings of the Tenth International Conference on World Wide Web , 2001;(6). 456-465.

[61]N Ashish , C A Knoblock. Semi - automatic Wrapper Generation for Internet Information Sources. Second IFCIS Conference on Cooperative Information Systems (Coop IS) , South Carolina , June 1997. 166-175.

[62]Morrison D R. PATRICIA - Practical Algorithm to Retrieve

Information Coded in Alphanumeric. Journal of the Association for Computing Machinery ,1968;15(4).36-45.

[63]Mundluru,D.,Katukuri,J.R,and Celebi,S.Automatically Mining Search Result Records[J].Data Mining 2005.1066-1075.

[64]Zhao H.,Meng W.,WuZ.,Raghavan V.,and Yu C..Fully Automatic Wrapper Generation for Search Engines[A]. In:WWW 2005[C].66-75.

[65]Liu B.,Grossman R.,Zhai Y.Mining Data Records in Web Pages[A].KDD 2003[C].601-606.

[66]ZhaiY.,and Liu B..Web Data Extraction Based on Partial Tree Alignment[A].WWW 2005[C].76-85.

[67]ZhangK.,andShashaD..Tree Pattern Matching.In:Pattern Matching Algorithms [M]. Oxford University Press,1997.6-15.

[68]Justin Park and Denilson Barbosa.Adaptive Record Extraction From Web Pages [A]. WWW 2007[C].56-65.

[69]王琦，唐世渭，杨冬青. 基于DOM的网页主题信息自动提取[J]. 计算机研究与发展，2004， 41(10)：11-15.

[70]李保利，陈玉忠，俞士汶. 信息抽取研究综述[J]. 计算机工程与应用，2003， 39(10)：1-5.

[71]贡正仙，朱巧明，李培峰. 基于相似页面的 WEB 信息抽取系统的实现[J]. 2006，8，(26)：1983-1986.

[72]候震宇.主题型搜索引擎的研究与实现[D].北京：中国科学院研究生院硕士学位论文,2003：2-91

[73]厉亮.主题搜索引擎的探讨[A].李晓明,李星.搜索引擎与Web挖掘进展[C].北京：高等教育出版社,2003：34-40.

[74]李名智.中文搜索引擎：现状、问题及对策[J].大学图书馆学报.1998,(4)：134-140.

[75]林彤.Internet 的搜索引擎[J].计算机工程与应用.2002,(5)：1134-1140.

[76]李盛韬.主题Web信息采集的研究与设计[A].孙茂松,陈群秀.语言计算与基于内容的文本处理[C].北京:清华大学出版社,2003.488-494.

[77] Richard Blum 著.高春荣译 C#网络应用编程[M].北京：电子工业出版社,2003：734-740.

[78]常璐,夏祖奇.搜索引擎的几种常用排序算法[J].图书情报工作,2003(6)：70-73,80

[79]徐宝文,张卫丰.搜索引擎与信息获取技术[J].北京：清华大学出版社,2003.112-115

[80]杨思洛.搜索引擎的排序技术研究[J].现代图书情报技术,2005(1)：43-44.

[81]李景.主要本体表示语言的比较研究[J].数字图书馆,2005,(1)7：114-120.

攻读学位期间的主要学术成果

表 1）发表论文

序号	论文作者	论文题目	期刊名称	发表时间 （卷次、页码）
1	谭骏珊 *陈可钦*	聚焦爬行中网页爬行算法的改进	电脑知识与技术--学术交流	2008 年 12 月 第 4 期：2145-2149 页.

致 谢

本文的诞生是一个漫长过程，从一年前开始接触搜索引擎，到论文的完稿，我得到了很多人的帮助，学到了很多知识。

在这里，我首先要感谢我的导师谭骏珊教授。是谭老师把我引入对信息处理最前沿的领域——数据挖掘的研究，我将里面的思想应用到对搜索引擎的研究当中。在我的研究生阶段，谭老师一直对我谆谆教诲、悉心指导。在此，向谭老师致以最真诚的敬意和感谢！

感谢我们所有实验室的兄弟姐妹，我们一起渡过了美好的时光。

最后， 我要感谢我的父母、家人对我一直的鼓励和支持。

陈可钦

2009 年 6 月

基于垂直搜索引擎的主题爬虫算法的研究

作者：陈可钦
学位授予单位：中南林业科技大学