分类号 密级：

U D C 编号： 10486

武 汉 大 学

硕 士 学 位 论 文

一种聚焦爬虫工具的设计与实现

研 究 生 姓 名：侯 婷 婷

研 究 生 学 号：2009206350020

指导教师姓名、职称：李 兵 教 授

学 科、专 业 名 称：计算机软件与理论

研 究 方 向：网络化软件

二零一一 年 四 月

Dissertation Submitted to

Wuhan University

**The design and realization of a focused crawler**

By

Hou Tingting

Under the Guidance of

Professor Li Bing

April 2011

郑重声明

本人的学位论文是在导师指导下独立撰写并完成的，学位论文没有剽窃、抄袭、造假等违反学术道德、学术规范和侵权行为，否则,本人愿意承担由此而产生的法律责任和法律后果，特此郑重声明。

**学位论文作者签名：**

**二零一一 年 四 月 十 日**

摘 要

随着Internet的飞速发展，互联网上的信息开始以“爆炸式”增长，形成一个庞大的信息海。此时，用户要在此信息海中准确而高效的获得所需的信息并非易事。而且，用户对信息的需求往往针对某个领域，面向特定主题。传统的搜索引擎通常是通过关键字进行查找，而由于人类语言的模糊性以及人的理解差异性问题，用户并不能准确的描述自己需要的信息，因此传统搜索引擎在这些方面都不太能令人满意。而面向主题的垂直搜索引擎，是根据领域把信息细分了，更能找到密切精准的信息，所以它能在满足用户追求专业详细信息方面有着独特的优势。

网络爬虫作为搜索引擎的核心模块，起着至关重要的作用。一个性能良好的搜索引擎往往都有着强大的网络爬虫。聚焦爬虫作为垂直搜索引擎的基础模块，它和传统爬虫相比较为复杂，它是垂直搜索引擎的基础组件，用一定的网页解析算法获得主题相关的链接，舍弃那些没有关联的链接，从而可以一定程度的满足面向主题的个性化的需求。

提出了一种可定制的聚焦网络爬虫。首先，用一系列遵循一定规则的过滤器侦察，根据用户的指定和主题分析进行URL搜索过滤，所用的搜索策略结合了简便的查全率高的深度优先搜索策略和类似HITs的基于链接入度值的排序算法，既保证了主题数据的查全率，也可按主题相关度对页面进行排序；然后，实现了自己的网页解析器，该网页解析器根据HTML的词法规则把HTML源码解析成由标签标记组成的标记序列，然后根据HTML的语法规则把标记序列中的标签解析成一颗以节点为单元的一颗语法树，以供之后的主题建模和主题数据匹配和提取所用。其次，实现了一种面向主题的定制方法。该方法利用用户在某一个样本文档上选择的目标结构及目标数据生成一个模板，然后系统的数据存储模块根据此模板抓取并存储具有相似结构的网页上的相关主题的数据。此算法是建立在同样主题的网页常常有着相似的内容结构的规律的基础上。最后，通过相应的实验，证实了此聚焦爬虫在一定程度上满足个性化的面向特定主题的需求，尤其是在具有领域性专业性的网站上搜集同个主题的主题数据的时候表现出了一定的优势。

关 键 词：垂直搜索引擎，聚焦爬虫，网页解析，数据挖掘

Abstract

With the rapid development of Internet, information on the Internet began to grow in a explosive manner, forming a huge sea of ​​information. At this point, the users are not easy to access to information in the sea of information accurately and efficiently. Moreover, the information the users need is often for a certain area or for a specific topic. Traditional search engines typically search by keyword, and because of the ambiguity of human language, and people’s different understanding the problem, the users do not accurately describe the information they need, traditional search engines are not satisfactory in these areas. The subject-oriented vertical search engines, which divide information according to fields division, are more closely to find accurate information. So it has a unique advantage in meeting users pursuing professional and detailed information.  
 As a core module of search engine, crawler plays a vital role. The search engine having a good performance tends to have a strong crawler. As a based module of vertical search engine, focused crawler is more complex than the traditional crawler. Focused crawler is the basis of vertical search engines, it uses certain analytic method to obtain links which accord with topic and abandon those are not associated with the topic, so that it can meet the needs of the individual requirement to some extent.  
 It presents a focused Web crawler which can be customized. First of all, we use a series of filters which follow certain rules to reconnoiter, and filer the URLs according to user’ s specification and themes analysis. The search strategy used is a simple search strategy combined with the depth-first search strategy which has a high recall rate and a similar HITs strategy which sorts by the hub value of links. Therefore it can ensure the recall rate of the topical data and sort the pages by the degree of relevance to the subject. Then, the paper implements a web page parser, the Web page parser parses HTML source code according to HTML lexical rules to get a token sequence consisted of the tags, then the parser parse the tag sequences according to the syntax rules of HTML to get a syntax tree consisted of units of node, which is used for the subject data matching and extract after the subject modeling. Secondly, the paper provides a method for theme customization. The method uses the target structure and the target data of the sample document the user selected to generate a template, then the system's data storage module captures and stores topic-related data of the pages with similar structure according to the template. This algorithm is based on the law that the pages which belong to the same topical often have similar structure. Finally, the corresponding experiments demonstrate that this focused crawler can meet the individual needs of the specific topic-oriented to a certain degree. Especially, in Web sites which are professional and has territory, the focused crawler shows certain advantages on search the data of the same theme.

**Keyword: vertical search engine, focused crawler, web page parser, data mining**

目 录

[**摘 要** IV](#_Toc292495202)

[**Abstract** V](#_Toc292495203)

[**第一章 绪论** 9](#_Toc292495204)

[1.1研究背景 9](#_Toc292495205)

[1.2研究现状 10](#_Toc292495206)

[1.3本文的工作 11](#_Toc292495207)

[1.4 论文的结构 11](#_Toc292495208)

[**第二章 网络爬虫原理** 13](#_Toc292495209)

[2.1网络爬虫 13](#_Toc292495210)

[2.1.1 网络爬虫组成 13](#_Toc292495211)

[2.1.2 爬虫流程 14](#_Toc292495212)

[2.2聚焦爬虫概述 14](#_Toc292495213)

[2.2.1 爬取目标描述 15](#_Toc292495214)

[2.2.2 URL搜索策略 15](#_Toc292495215)

[2.2.3 网页解析 16](#_Toc292495216)

[2.2.3.1 基于网络结构的分析 16](#_Toc292495217)

[2.2.3.2基于网页内容的分析 17](#_Toc292495218)

[2.3爬虫示例 17](#_Toc292495219)

[2.3.1 Teleport Pro 18](#_Toc292495220)

[2.3.2 Websphinx 18](#_Toc292495221)

[2.3.2.1 Websphinx的组成 19](#_Toc292495222)

[2.3.2.2 Websphinx的操作 19](#_Toc292495223)

[2.3.3 Teleport Pro和Websphinx的比较 20](#_Toc292495224)

[2.4本章小结 20](#_Toc292495225)

[**第三章 系统框架及工作流程** 22](#_Toc292495226)

[3.1系统框架 22](#_Toc292495227)

[3.2模块细解 22](#_Toc292495228)

[3.2.1 搜索过滤模块 23](#_Toc292495229)

[3.2.2 调度和爬取模块 23](#_Toc292495230)

[3.2.3 HTML解析模块 24](#_Toc292495231)

[3.2.4 模板模块 25](#_Toc292495232)

[3.2.5 主题数据提取模块 26](#_Toc292495233)

[3.3本章小结 27](#_Toc292495234)

[**第四章 聚焦爬虫的实现** 28](#_Toc292495235)

[4.1平台与技术 28](#_Toc292495236)

[4.1.1 Java技术及工具包 28](#_Toc292495237)

[4.1.2 MySQL 29](#_Toc292495238)

[4.2 源码视图 29](#_Toc292495239)

[4.3主要算法 31](#_Toc292495240)

[4.3.1 网页搜索算法 31](#_Toc292495241)

[4.3.2 HTML解析器 31](#_Toc292495242)

[4.3.2.1 词法分析器 32](#_Toc292495243)

[4.3.2.2 语法分析器 35](#_Toc292495244)

[4.3.3 主题定制算法 37](#_Toc292495245)

[4.3.3.1 模板通配节点 37](#_Toc292495246)

[4.3.3.2 通配符匹配算法 39](#_Toc292495247)

[4.3.3.3 主题模板匹配 40](#_Toc292495248)

[4.3.4 模板优化 42](#_Toc292495249)

[4.4用户接口 43](#_Toc292495250)

[4.5本章小结 47](#_Toc292495251)

[**第五章 实验设计及结果分析** 48](#_Toc292495252)

[5.1实验设计 48](#_Toc292495253)

[5.1.1 参数设置 48](#_Toc292495254)

[5.1.2 主题定制 48](#_Toc292495255)

[5.1.3 运行结果 50](#_Toc292495256)

[5.2结果分析 51](#_Toc292495257)

[5.3本文聚焦爬虫工具的应用 52](#_Toc292495258)

[5.4本章小结 52](#_Toc292495259)

[**第六章 总结与展望** 53](#_Toc292495260)

[6.1总结 53](#_Toc292495261)

[6.2展望 53](#_Toc292495262)

[**参考文献** 55](#_Toc292495263)

[**致 谢** 58](#_Toc292495264)

[**在读期间参与的科研项目** 59](#_Toc292495265)

第一章 绪论

1.1研究背景

随着网络的诞生及其发展，互联网成为大量信息的载体，浩如烟海的信息在给用户大量选择的同时也为精确查找带来了难度。“人类正在被数据淹没”，却找不到自己需要的信息，正面临“数据丰富，知识贫乏”的境地。如果没有一个有效便利的工具，要在互联网上找到自己感兴趣的知识信息，就如大海捞针。

由于人们对查找自己真正需要的信息的需求越来越多，搜索引擎产生了。搜索引擎根据一定的方法和某种程序，在互联网上搜集信息，并对这些信息进行加工，然后提供给用户进行查找。搜索引擎主要有通用全文搜索引擎和垂直搜索引擎两大类。传统的搜索引擎，其特点是关键字查询，拥有海量的数据，例如：Google，Baidu等。垂直搜索引擎不同于传统搜索引擎，往往专注于特定的搜索领域，其特点是“专，精，深”。

传统搜索引擎因为其自身的特点，其弊端已日益明显：首先，为了达到一定的爬全率，传统搜索引擎往往搜索整个互联网，但是随着网页数量的爆炸性增长，搜索引擎无法获取所有的资源。其次，传统的搜索引擎无法提供给用户日益个性化的需求，因为它要求用户输入关键字，再通过关键字进行查找。但对普通用户来说，对于所要找的信息并不能准确的进行概括描述或者是找到关键的词语，这就使得检索的结果很模糊，不够准确，甚至毫不搭边。而对于专业的研究者或专家而言，其可供选择的查找方式过于单调简单，查找结果也显得泛泛不够精深，常常与欲查找的领域无关。再次，由于互联网更新频繁，传统搜索引擎为了维护庞杂的信息库，更新不够及时，无法满足用户的实时性要求。以上这些都使得传统的搜索引擎越来越难以满足用户的需要。

垂直搜索引擎可以较好的解决以上问题，它是搜索引擎的一种延伸的发展结果，是对网页库中某类专门的信息进行整理，形成特定领域或特定主题的划分。当用户确定主题后，就返回此类主题的内容给用户。由于垂直搜索是针对特定领域的，给某种用户群提供一定信息和服务，具有行业色彩，相比通用搜索引擎的海量信息的庞杂与无序性，显得更加专注，精进和深入[1]。

网络爬虫是搜索引擎的一个重要组成部分，它按照一定的规则，自动抓取网页。传统爬虫从一个或若干个起始网页URL开始，抓取这些网页之后并从其中抽取新的URL，直到满足系统的停止条件而终止抓取。聚焦爬虫的性能直接影响搜索引擎的质量。聚焦爬虫是针对特定领域的专业搜索引擎，其个性化和智能化可以使其只关注某一个领域的文档，能很好的解决通用搜索引擎所面临的问题。

如今，在生活及研究中，人们在网上冲浪查找信息时，常常只关注某一方面的知识，比如“流行歌曲”,“健康饮食”等，只靠关键字搜索，难以查到想要的结果，往往是一些不是很符合要求的网页链接。对于研究者来讲，往往需要的是某一个领域的信息，比如实验室的研究，研究者们常常需要抓取某一个网站或几个网站的某种信息，例如某种服务或某类软件的信息来进行实验和研究，但是用普通的搜索引擎却无法提供这些详细信息，在这种情况下，就需要有一个可定制的聚焦爬虫来完成这些抓取任务。

1.2研究现状

目前，已有很多搜索引擎的搜索引擎产生，而且有各种语言编写的，其中一些已使用的相当普遍，而且具有权威性，如Google，Baidu等。但大多数都是传统的全文搜索引擎，是利用关键字查询的，它的局限性给很多用户带来了不便。

垂直搜索因为传统搜索引擎的弊端和它自身特色带来的优势，吸引着一大堆学者对它进行研究。垂直搜索引擎与传统搜索引擎的区别就是它是以网页的结构化的信息为单位进行提取，处理，存储以供索引和查找的，这种方式更能贴合用户的实际需求。作为垂直搜索引擎核心的主题爬虫也因此成为研究的热点。在垂直搜索引擎方面的研究上，出现了通用搜索引擎嵌入垂直搜索引擎的情况，比如Baidu，Google的分类列表，就是用了垂直搜索引擎的行业化，领域化分工的思想。但是垂直搜索引擎并不是简单的分类，它更注重对信息的细分，追求特定领域知识的专业化，深入化探索，这就决定了搜索引擎行业细分化的趋势。

目前已有一系列的一般用途的网络爬虫：RBSE，第一个发布的爬虫，它的两个基础程序：“spider”和“mite”，前者负责抓取队列中的内容存到数据库中，后者是一个浏览器，负责从网络上下载页面。Google Crawler用细节描述，集成了索引处理，它拥有一个URL服务器，用于给爬虫程序发送抓取的URL列表。WebRACE是使用java实现的，拥有检索模块和缓存模块。系统从用户得到下载页面的请求，其最大的特色是，不像大多数的爬虫从一组URL开始，而是连续的接收抓取开始的URL地址[2]。

另外有些开源爬虫，比如Hetrix，Websphinx，Nutch等，这些爬虫都有各自的特色，侧重于不同的方面。同时由于这些爬虫的开源性，也给其他学者提供了研究学习的机会，反过来使爬虫本身也不断改进。但考虑到一些技术竞争性问题，这些开源代码中的核心模块处于保密状态。

聚焦爬虫是一个研究热点，很多学者从不同的角度研究聚焦爬虫。例如，在[3][4][19]中， 是从语义方面出发，提出基于本体的爬虫研究方案，使用一些本体构建工具构建本体，然后利用本体来获得主体化的关键字的提取，从而来确定主题。

这种方法和其他不同之处在于，它是考虑链接的语义性来实现文档相关度的测量与过滤。而[6][31]则是对常见聚焦爬虫通用爬行策略的改进研究。

1.3本文的工作

本文针对目前通用爬虫的缺陷，提出了一种可定制的面向主题的聚焦网络爬虫。其主要的工作如下：

(1) 提出一种基于链接入度值计算的宽度优先的URL搜索过滤策略。该算法为了查全率和搜索简便性，采用宽度优先的搜索策略，但是考虑到此算法没考虑到主题相关度的计算，因此模仿HITs算法，结合进了链接入度值的计算，来对链接按照主题相关度进行排序。

(2) 实现了网页解析器。该网页解析器的词法分析器解析出网页的所有标签，然后用语法分析器对标签序列进行语法分析，产生以节点为单元的一个语法结构树，以供之后的主题定制和模板匹配获得主题数据所用。该解析器考虑到Web上的大部分HTML源码的不规范性，实现了一定的容错性，并具备纠错功能。

(3) 实现了一种面向主题的定制方法。该方法利用用户选择的样本文档的语法结构树的结构和在该样本文档上选择的目标数据生成一个模板，然后系统的主题数据提取模块根据此模板抓取并存储相关主题的数据。此算法是基于某个领域的网站上的主题网页常常有着相似的结构和同样主题的文本的规律。

(4) 通过选取了网站做实验，其结果表明此聚焦爬虫可以根据用户选择的主题查找相关信息，尤其是在这些同样的主题的信息页面有着相似的结构，而且主题信息分散但有规律的时候，此爬虫表现出明显的优势。

1.4 论文的结构

本文的主要内容共有五章。

第一章绪论，介绍了研究的背景即搜索引擎的产生及发展，搜索引擎如今的发展状况，以及网络爬虫的地位作用，然后提出本文所要开展的内容等。

第二章介绍了爬虫的相关知识，包括爬虫的基本知识，已有的爬虫，聚焦爬虫的原理等。这一章为之后的工作做了理论上的准备，也给出了此聚焦爬虫工具实现的总体思路。

第三章是系统框架和工作流程，这一章从总体上介绍本工具的框架，各个模块以及它的工作流程以对此工具有个系统的认识。

第四章介绍聚焦爬虫的实现，这章详细讲解了实现此主题爬虫所用的平台，算法策略，以及源码的层次布局等。

第五章是实验设计及结果分析，这章中通过对一个网站的实际实验操作，来检验此聚焦爬虫工具的思想及功能是否达到预期要求，并通过结果分析来检验其性能好坏。

第六章作为一个总结，指出本文研究的不足，提出应该改进的地方，为以后的工作给出思路。

第二章 网络爬虫原理

2.1网络爬虫

网络爬虫，又称网页蜘蛛和网络机器人，是按照一定的规则，自动抓取网页的计算机程序。爬虫通常从某一个起始页开始抓取网页，读取网页的内容，解析出其中的链接，再通过这些链接寻找新的页面，这样一直循环，直到满足系统的终止条件而停止抓取[7]。

2.1.1 网络爬虫组成

网络爬虫分为通用网络爬虫和聚焦爬虫。

通用网络爬虫的主程序主要由调度器，解析器和资源库三部分组成。调度器主要负责给主程序中的各个爬虫线程分配工作任务。调度器是网络爬虫的中央控制器，它根据系统传过来的URL，分配一线程，启动此线程以调用爬虫爬取网页。解析器负责下载网页，解页面，处理析网页的内容，爬虫的基本工作是由解析器完成的。资源库用于存储下载的网页等资源。

Web

下载模块

URL队列

页面解析器

URL数据库

Web数据库

图2-1 通用网络爬虫的结构图

而聚焦爬虫是在通用爬虫的基础上添加一些主题定制模块，因此它除了调度器，解析器和资源库之外，一般还要有搜索策略和网页及URL的主题相关度评价模块。本文设计的聚焦爬虫结构图见图3-1.

2.1.2 爬虫流程

通用网络爬虫常常是简单的下载页面内容，追求的是对于网络的搜全率，要求数据资源很庞大全面。在过滤方面有些常常是简单的为URL添加关键字进行页面过滤，如文献[16]。而聚焦爬虫针对的是某个领域的数据，相反，追求的是精细而专注的数据，因此它就要求有一定的网页和链接的分析过滤方法。

下面就从通用爬虫和聚焦爬虫的工作流程来进行比较[1]，如图2-2所示。

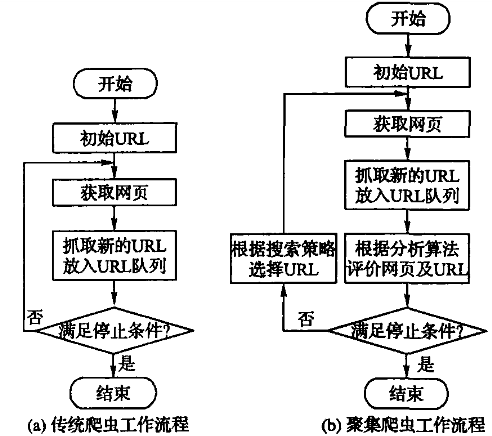


图2-2 传统爬虫和聚焦爬虫的工作流程对比

从图2-2可以看出，传统爬虫和聚焦爬虫的区别，聚焦爬虫需要对网页进行过滤，所以对它所抓取的URL都要根据一定的分析算法进行评价，以过滤符合主题的网页。

2.2聚焦爬虫概述

聚焦爬虫也叫“主题爬虫”，是垂直搜索引擎的核心。主题爬虫建立在普通爬虫的基础之上，在网页处理过程中增加模块实现个性化提取。这些模块包括优化初始种子模块，主题确定模块，主题匹配模块等。聚焦爬虫只下载那些和预定义的主题内容相关的页面，这些预定义的主题需要提供一个样本文档，而不仅仅是使用一些关键字来定义。和那些搜索所有web页面，对所有问题进行解答的通用引擎不同，主题爬虫分析每个页面的链接，并判断这些链接是否是和预定义的主题相关，从而保留主题相关的链接，舍弃无关的链接。其目标就是尽可能的搜集与特定主题相关的网页，避免无关网页的下载，使得保存的页面也由于数量少而更新快。

相对于通用网络爬虫，聚焦爬虫还要解决三个主要问题：如何对爬取的对象进行描述；如何对网页进行解析和处理；URL的搜索方法。

2.2.1 爬取目标描述

聚焦爬虫对爬取目标的描述主要有三种：基于目标网页结构特征，基于目标数据模式，基于领域概念。

基于目标网页结构特征的爬虫，通常需要先指定一个样本页面作为种子，提取其特征后，按照此特征抓取相关的网页。此特征的建立可以是根据网页的内容特征进行制定，也可是根据页面的链接结果来生成。

基于目标数据模式的爬虫，主要是为了抓取具体的数据，它要先建立一定模式的数据样例，在抓取的过程中，匹配网页中所有的数据，提取出符合此模式的数据。

基于领域概念的爬虫，常常关注领域的划分，这种模式都要先建立自己的词典或者词库，以对抓取的内容进行词典查找后使其归入相应的所属的领域。还可以建立本体，从语义角度对抓取的内容进行本体匹配已达到按领域提取的目的。

2.2.2 URL搜索策略

网页的搜索策略通常有深度优先，宽度优先和最佳优先。最常用的是宽度优先和最佳优先[8]。

深度优先策略是指在抓取过程中，当某一个链接被选取后，就先获得此链接指向的网页，并对此页面执行深度优先搜索，也就是在对其他链接进行搜索之前，要先完整的搜索单独的一条链。深度优先策略是尽可能深的搜索Web地图，在深度优先搜索中，如果最新发现的顶点有没有访问的边，就会随着这个边上访问其上的节点，直到节点没有边为止，然后返回到最初的节点，访问下一个没有访问过的边。重复以上的步骤直到所有的节点都找到，此搜索即完成。深度优先策略在信息量不大的时候可以采用，但是互联网发展迅速，数据量急剧膨胀，使得深度优先策略容易让爬虫进入陷阱，所以使用的较少[9]。

宽度优先是指在抓取过程中，当解析某个页面时，遍历此页面的所有链接，获得这些链接所指向的页面，然后再对这些页面继续执行宽度优先算法。该算法比较简单，许多研究者都在聚焦爬虫中使用这种搜索策略，其基本思想就是认为与起始URL在一定的链接距离内与主题相关的可能性很大。

最佳优先搜索策略会对候选URL进行主题相关度比较，选出评价最好的URL进行抓取。这种算法的思想是，它只抓取和目标主题相关的网页，忽略掉没用的网页。但是这种算法会造成网页的漏失，因为被忽略的URL可能正是主题相关的网页，也可能它的内容里有指向主题相关网页的链接。所以最佳优先策略通常要有一定的网页分析算法结合使用，避免局部性[10]。

2.2.3 网页解析

网页分析算法分为基于网络链接结构，基于网页内容和基于用户行为三种。目前，基于网络链接结构和基于网页内容的爬虫较为常见[11]。

2.2.3.1 基于网络结构的分析

基于链接结构的最常见的两个算法是PageRank和HITS算法[12]。

PageRank算法是Google用于网页排名的方法，是用于衡量网页好坏重要的唯一标准。它的基本思想是：如果一个页面被许多其它的页面引用，那么这个页面很可能是个很重要的页面；一个页面如果没有被多次引用，但它被某个重要页面引用，则这个页面也可能是重要的页面；一个页面的重要性被均分并传递给它所引用的页面[14]。一个网页的PageRank值表示在任何给定的时间，有一个随机用户访问此网页的可能性。一个网页的得分递归的依赖于所有引用它的网页的得分。每个网页都会分配它的PageRank值给它所有的外链接。如下为一个页面的PageRank值的计算公式：

C:\Users\Administrator\AppData\Roaming\Tencent\Users\1209498787\QQ\WinTemp\RichOle\WGEO`H$O6[X~5GE34_{T(_V.jpg

（1）

其中，*j*是正在被打分的网页，*in(j)*是指向*j*的网页集合，*out(i)*是*i*的外连接的集合，常量<1，它是一个减震因子，用于表示一个随机用户访问另一个随机网页的概率。PageRank算法常用于指导爬虫高品质的访问网页，它是一个主题无关的方法，必须结合其他的采用了评价相关性的方法一起来对搜索的结果进行排名。

HITS算法是对一个网页的出度和入度进行计算来评价页面的好坏。网页的入度是指指向此网页的链接数，而出度是指此网页中所包含的指向其他的网页的链接数。这两个值分别用Hub和Authority来表示。如果一个网页有很多其他的网页引用它，则这个网页很有可能是个很中心的网页，即它是此网站和此主题内容的中心页面。如果一个网页有很多指向其他页面的链接，则此页面很可能是个权威的门户网站，或者是分类页面，指向每类的具体主题相关的网页。也就是说，Hub和Authority分别代表着页面的中心度和权威度，然后利用这两个值对页面进行排序[20]。

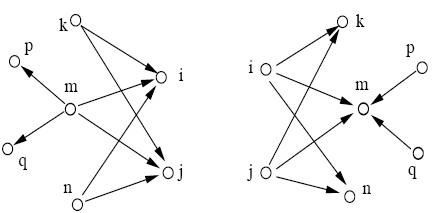


图2-3 HITS算法的出度入度

在图2-3中，每个节点代表一个网页，而每个节点指向其他节点的箭头表示出度，指向节点的箭头表示入度。观察图可知，左图中，m的出度比较大，则它的中心性就很大，而在右图中，m的入度比较大，则它的权威性就比较大。

HITS算法有一定的范围限定，比如以旅游为主题网页，指向另一个网页，如果另一个网页也是以旅游为主题，则它的重要性就很高，但如果指向的另一个网页是烹饪什么的，它的重要性就不一定了。

2.2.3.2基于网页内容的分析

基于内容的网页分析算法通常有两种。第一种方法评价整个网页，每个网页中的链接都给予同样的权重。另一种方法是基于链接的上下文环境的，对于网页中的每个链接，根据其不同的上下文环境赋予其不同的权重。这两种方法都有一些局限性。第一种方法，为了评价整个网页的算法，会使得那些和主题无关的网页也会获得访问的优先权，因为每个链接都有同样的权重。而第二种方法，当评价链接时会评价链接时会由于缺少信息而忽略一些重要的链接[13]。

2.3爬虫示例

在1.2节的研究现状里，列举了一些开源的爬虫和他人所做的研究工作，而本文的工作就是在这些前人研究的基础上进行的改进。在这里就着重介绍一下以下两种开源爬虫：Teleport Pro和Websphinx。

2.3.1 Teleport Pro

Teleport Pro是由美国Tennyson Maxwell公司开发的整站下载工具，它可以把网站上所有的内容，包括纯文本的页面，图形文件以及程序等都下载下来，并在硬盘里形成一个镜像，使得用户可以离线查看所下载网站的内容。Teleport Pro是个通用爬虫的典型实例，它可以同时打开十个进程进行下载，速度很快，而且支持所有类型的文件。

Teleport Pro操作步骤如下：首先得设置起始网页创建一个项目，然后设置项目，可以设置抓取的深度，要抓取的文件类型和大小，运行的时间等，还可以设置关键字来对网页内容进行过滤，可以设置URL过滤条件来过滤无用的网址链接。设置好这些参数后，运行项目就可以了。运行完成后，可以在界面中查看运行结果。

以下示例是设置起始网址为http://www.baidu.com，抓取文件类型为纯文本，运行之后的结果如图2-4所示：

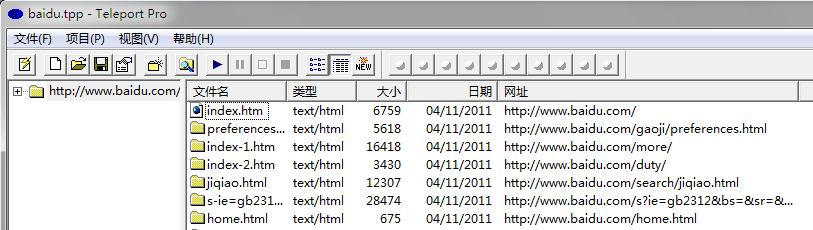


图2-4 Teleport Pro 运行结果

综上，Teleport Pro只提供整站下载，它可以把某个网站上的所有的内容都下载下来，可以很好的提供离线浏览。但是它却不能提供可定制性，以使得用户可以进行具体数据的提取。因此，它不能满足用户不需要整个网站，而是只需要某些数据时的要求，而下面将要介绍的Websphinx则在这方面有所改进。

2.3.2 Websphinx

Websphinx是由java实现的一个基于文本的搜索引擎，它提供了基本的网页下载，并提供了简单的数据匹配模式进行数据抽取。另外，它对网站结构的图形化展示是它的一个特色。

2.3.2.1 Websphinx的组成

它由两部分组成：抓取平台和Websphinx类库。

抓取平台是个图形用户界面，它允许用户进行一些配置，比如是整个网站还是整个互联网等，设置是保存网页还是只提取匹配的数据，通过这个抓取平台，可以把抓取到的网页集合以一个虚拟化的图表来表示，可以保存网页到本地磁盘以供离线浏览，可把网页串联起来作为单一的文档打印，它有一定的文本匹配模式，可以使用户选择某种模式的数据的提取。

Websphinx类库提供了在此基础上进行爬虫开发的支持，它有以下特色[7]：

1. 在一个简单的应用程序框架里用多个线程进行网页提取
2. 有一个对象模型来表示和呈现网页以及链接
3. 提供支持可利用的页面内容的分类器
4. 包容性的解析网页，即有一定的容错性
5. 支持机器人拒绝协议。
6. 提供模式匹配，包括正则表达式，Unix的shell通配符，以及HTML中的Tag标签表达式，正则表达式是由Apache的 jakarta-regexp 正则表达式库的标准提供的。
7. 通常的HTML转换，比如页面串联，把页面存入磁盘，重命名链接等。

2.3.2.2 Websphinx的操作

Websphinx的使用和操作步骤如下：

首先，选择抓取的范围。有如下三个选择：the subtree，the server，the web。

第一个the subtree即是在此网页子集里进行抓取；the server则是在整个服务器上的内容进行抓取；the web则是在整个互联网上抓取所有的内容。

然后输入起始网址。起始网页即此爬虫开始抓取的URL地址。起始网址可以有一个或者几个，一定要是完整有效的URL地址。

接着选择抓取动作：none，save，concatenate，extract，highlight和script。

Save是把网页下载下来保存在磁盘中，要选择保存的目录；concatenate是串联网页，要选择所要串联的文件，并设置串联的选项，系统会把抓取到的网页加在选择的串联文件的内容后面；extract就是模式匹配的集中体现，它要求输入一段HTML Tag表达式，用于匹配所要抽取的数据，其界面如图2-5所示：

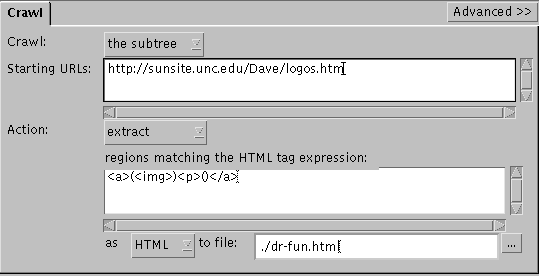


图2-5 Websphinx抽取动作标签表达式图示

以上是基本配置，在高级设置里，还可以设置深度，制定爬行策略，设定线程数量，文件大小，链接和网页等的细节限定等；

设定完成后，运行项目即可。在运行结束后，可以看到图形显示窗口展现了下载内容的链接结构图。在统计窗口中，会有系统和下载内容相关的统计数据。

2.3.3 Teleport Pro和Websphinx的比较

Websphinx和Teleport Pro相比，Websphinx增加了对页面内容的选择过滤模块，它有分类器，可以对页面进行分类；还有模式匹配功能，可以用一些正则表达式进行模式匹配选择所要的数据。在结果显示方面，Websphinx把网站的链接结构以图形化的方式和树形结构呈现出来，使得对整个用户对整个网站的结构可以有清楚的了解。但是究其细节，它的图形化结构呈现在页面复杂度比较低的时候可以很明显，但在网站比较庞大，站内结构错综复杂，链接也纵横交错时，其图形化呈现就基本失去了它的意义。另外，在内容抽取方面，它的匹配很有限，尤其是对页面内容中具体数据的提取，仅仅通过输入一些标签描述进行匹配，这使得用户必须查看页面的源码，查找所需的源码片段，使用起来很不方便。在对抽取的数据的保存方面，Websphinx也没有很好的处理，使得最初的提取数据的目的难以实现，不利于用户尤其是研究者对数据进行研究利用。

2.4本章小结

本章介绍了爬虫的相关知识，包括爬虫的组成，工作流程，并对通用爬虫和聚焦爬虫作了比较。另外，详细介绍了聚焦爬虫的原理、关键技术及其特点。最后列举了两个爬虫示例Teleport Pro和Websphinx，通过介绍示例爬虫的组成、特色、操作步骤、结果展示以及他们之间的比较使读者对爬虫有比较全面的认识，并对聚焦爬虫与通用爬虫的区别有了一定的了解。

第三章 系统框架及工作流程

本章从总体上对系统进行介绍。首先介绍系统的总体框架，使读者对系统的总体构成有个初步了解，然后对每个模块进行介绍，主要是关注模块的功能及与其他模块之间的联系，最后介绍此系统工具的工作流程。

3.1系统框架

由于通用模块是由控制器，解析器和资源库三部分组成，而聚焦爬虫是在通用爬虫的基础上增加主题相关模块来实现的，所以聚焦爬虫比通用爬虫要多出一些模块用于定制主题[14]。

目前，聚焦爬虫主要有三种代表性的体系结构：基于分类器的聚焦爬虫[5]，基于数据抽取器的聚焦爬虫和基于用户学习的聚焦爬虫[15]。本文所研究的聚焦爬虫正是一个侧重于数据抽取的聚焦爬虫，因此采用了基于数据抽取器的聚焦爬虫体系结构。系统框架图如图3-1所示。

样本网页集

URL集

网页资源库

主题数据集

过滤模块

调度模块

模板模块

主题数据提取

爬取模块

页面解析

图3-1 系统框架图

3.2模块细解

由系统框架图3-1可知，此聚焦爬虫的重点是对主题数据的抓取，因此主题定制和模版匹配就是聚焦爬虫的思想体现所在。而模版匹配必须得有一个好的页面解析算法解析网页，才能准确高效的进行匹配，获得所需的数据。本节将详细介绍各个模块的组成和功能等。

3.2.1 搜索过滤模块

搜索过滤模块主要是负责按照一定的搜索策略进行URL的搜索，然后对搜索到的URL进行过滤，获得有用的链接。

搜索到的每个URL都要经过过滤，再决定是否抓取。搜索策略的详细算法在第四章进行详细介绍。

过滤主要是负责URL的过滤，它的任务就是对URL进行主题相关性评价，使得那些和主题相关的链接保留下来，从而舍弃无用的链接。要实现URL的过滤功能，依靠用户的过滤设置和对目标网页的链接结构描述。

过滤设置包括所要下载的文件的类型、大小等，文件类型主要有图形文件（比如jpg、gif、png等）、文本文件（如html、txt、asp等）、可运行文件（如exe、com等）、压缩文件（如zip、rar等）、音频文件（mp3、wav、wma等）、视频文件（avi、mov、wmv等）。过滤还包括URL的深度等参数。

对目标网页的链接结构描述，是指指定所要查找的相关主题网页的链接所具有的共同结构，通常也就是他们都有什么样的链接前缀。这样的策略是基于如下的思想：相同主题的网页通常都有着很大的相似性，包括页面结构及连接结构，他们通常都是从同一个祖先链接（比如指向某个目录页面的链接）通过一层或多层链接而来的[16]。

经过了过滤而得到的URL都放在一个优先队列中，该优先队列按页面入度值进行排序。然后调度模块会给队列中的每个URL分配线程，进行抓取。

3.2.2 调度和爬取模块

调度模块是爬虫的控制器，负责分配线程。系统同时开启的线程数由用户指定，系统根据此设定开启指定数目的线程。对由过滤模块得到的URL候选队列中的每一个URL，调度模块分配给它一个线程，启动爬虫抓取模块进行抓取。

调度模块因为是对线程的管理，所以它要实现数据之间的同步和共享数据的安全问题。比如URL候选队列，每个线程都要访问它来获得一个候选URL进行抓取和解析等，如果一个URL被好几个线程同时获得，会导致对页面的重复提取。所以共享数据的共享和同步很重要，保护好这些共享数据，才能避免重复工作，数据冲突和其它因同步引起的问题。多线程是比较复杂的编程问题，调度模块为了实现多线程，应提供一个线程池， 对每个请求分配一个线程，对共享资源以互斥的方式操作，并避免死锁等问题[17]。

爬取模块根据调度模块的请求，采集一个网页，并更新网页资源库，同时抽取该网页的链接给过滤模块[18]。它的功能比较简单，它主要负责下载网页，网页下载后，要传给解析模块进行解析，然后根据用户的指定（存储与否），来决定是否存储在硬盘里。在网页下载存储过程中，需要解决的一个问题即是编码问题，不同的网页有不同的编码，所以对下载的网页获取它的编码，以便在后续的页面解析中产生乱码问题等。

3.2.3 HTML解析模块

解析模块是爬虫核心模块，它需要对爬取模块爬取的内容进行解析。解析模块主要有词法分析器和语法分析器两部分组成。如图3-2所示：

否

否

否

是

是

是

词单元放入词集合

页面文档

保存语法结构

语法单元是否链接

词法分析顺利完成

语法分析顺利完成

过滤模块

终止

图3-2 网页解析流程图

如图3-2所示，网页解析器主要经过词法分析和语法分析。对由爬取模块爬取的每个网页，词法分析器对其进行分词，保存为一个个的词法单元。

然后语法分析器对词法单元进行语法分析，解析出其语法结构并保存起来。对于每个语法单元，语法分析器会调用一系列监听器来监听语法单元的内容。链接监听器会判断该语法单元是否是一个链接。如果是链接，就将其传给过滤模块以判断其是否和主题相关。对于其它的语法单元，语法分析器会解析出它的语法成分，以供以后的主题建模及模板匹配使用。

由于现在的浏览器都有一定的容错性，所以很多网页的HTML源码虽能正确显示，但其中很多都有着一定的词法或语法错误。所以，如果要获得这些网页资源并抽取信息，自己实现的HTML分析器就必须也得具备一定的容错能力，能在一定的错误范围内解析出源码相应正确的语法结构[19]。关于本文的HTML分析器如何实现容错功能，其具体算法在4.3.2节的分析算法中有详细说明。

3.2.4 模板模块

模板模块是本文研究的聚焦爬虫的主题模块，它负责主题定制，建立模板。

其建模过程有以下步骤完成：

第一步，用户指定一个样本网页URL，并指定目标网页链接结构，过滤器通过此目标网页链接结构的描述进行链接的过滤，如过滤模块所讲。

第二步，模板模块调用页面解析器对样本网页进行解析，分析出网页的语法结构，语法结构树在面板中展示出，然后系统提取出所有的非空白的叶子节点，即所有的文本节点（包括描述信息，网页链接以及文件名等）。系统把所有的文本节点分行展示给用户。

第三步，用户根据选择文本节点，系统会在语法结构树中找到文本相应的位置，用户可以在语法树中进行选择，可以只选择叶子数据节点，也可以选择一个子树。

第四步，用户对自己选择的每个主题数据（文本节点或语法子树），指定数据的存储模式。

第五步，模板模块根据样本文档的语法结构和用户的目标数据模式建立一个主题模板。

在建模过程中，之所以选择文本节点和页面结构树展示给用户，让用户选择后建模，是在目前的情况下，大多数的研究都集中于对文本数据的研究，文本数据常常包含着大量的信息，这也是最可能需要的资源。而且，这些文本信息往往还包含图形文件或声音、音频等非文本文件的链接，这些链接也可保存下来，以供以后的对非文本方面的研究[20]。

另外，建模依据的样本文档的语法结构是一个森林结构，它有一颗或者多棵树组成。当文档是一个完整的HTML文档时，此文档的语法结构就是一棵树组成。相反，文档不够规范，是由一些HTML片段组成时，它就会由几棵树组成。

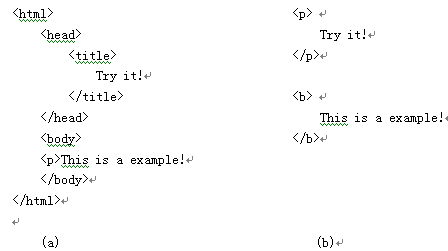


图3-3 文档语法结构树对比

在图3-3中，图（a）是一个完整的HTML，所以它的语法结构树是一个树形结构，其根节点就是html节点，它有两个孩子：head和body。head有一个孩子title，title有一个文本节点。其它节点由此可类推。图（b）是由一些HTML片段组成的，所以它是由两棵树组成的森林。这两棵树的根节点分别是：p和b，它们各有一个文本节点。

3.2.5 主题数据提取模块

在模板模块建立模板之后，主题数据提取模块就使用模板对主题相关的网页进行匹配，从而抓取出相关的主题数据，并按用户指定的数据模式进行存储。

在此模块中，要进行数据的提取，首先要对网页进行结构匹配。在结构匹配过程中，进行主题数据的提取。结构匹配上，要求主题相关的网页可以有相似的语法结构。语法结构相似是指可以由同一棵树通过扩展而来，扩展可以是指树中的某个中间的节点增加了子节点导致树某层节点树增多或者是树的层数增多。

H

A

B

C

F

D

E

G

A

B

C

D

E

F

A

B

C

F

通配符

通配符

图3-4 相似树说明图

在图3-4中，左图是一个模板语法树，右边的两颗语法树都可由此模板匹配出来，中间的语法树的节点D和E与模板树的B节点下的通配符匹配，右图中的H子树(H-DE也与) 模板树的B节点下的通配符匹配，而节点G则与模板树的节点C下的通配符匹配，所以这两棵树都是可以由模板匹配出的主题相关的树。正因为右边两棵树都可由模板匹配出，也就是都可由模板树发展演变而来，所以它们可以叫做相似结构树。

在匹配过程中，通配符匹配的数据即是需要搜集的主题数据，主题数据可以是如图3-4中的中间的图中的叶子节点，即文本节点D、E以及右图中G，也可以是右图中的H子树(H-DE)。

提取出的数据要按一定的模式存储在数据库，而这种存储模式在模板中已介绍，存储模式包括数据的存储位置和存储形式。数据的存储位置是指数据是存储在数据库中，还是存储于文件中，如果是文本节点，默认是存储在数据库中，而子树数据可以以HTML片段存储于数据库中，也可串联而成存储于本地文件中。存储形式是指规定每个数据所属的数据库，数据表和列名等。存储模式指定好后，连接数据库，按数据定义和操作语言按指定的存储模式存储数据即可。

3.3本章小结

本章首先介绍了系统的体系结构，给出了系统的总体框架。然后从各个模块出发介绍了各个模块的功能，模块算法的大致的步骤。在爬虫的组成中，网页解析器是基础模块，它处理网页的内容，提取新的URL，处理标签等，所以在模块详解中，着重介绍了网页分析模块的组成及工作流程等。由于聚焦爬虫是面向主题定制的，本文研究的聚焦爬虫又侧重于主题数据的抽取，所以它相较于通用爬虫多出的主题定制和匹配模块是重点模块，因此，本章对主题定制以及主题匹配的过程也作了详细的介绍。

第四章 聚焦爬虫的实现

通过上一章的介绍，已经了解了此系统的总体构成。下面就详细介绍此聚焦爬虫的具体实现，包括使用的平台和算法，以及搜索策略，源码视图等。

4.1平台与技术

此系统的实现的平台是在eclipse集成开发工具中，用java语言实现的。用户界面的实现主要采用了java的swing包。存储主题数据的数据库使用的是MySQL。

#### 4.1.1 Java技术及工具包

Java语言是一门面向对象的语言，它具有简单性、多线程性、平台无关性、解释性、强壮而安全和可移植性。

Java有如下两种特性使得它方面于本文内容的实现[21]。

1. 网络特性：要想在Web上抓取网页，就必须遵循Web协议，而Web协议又和HTTP协议和TCP/IP协议密切联系。而Java语言考虑了这些问题，它有自己的网络类库，包括套接字类，URL处理类等可以用来处理这些网络协议。
2. 多线程特性：Java的lang包中提供了Thread类和Runnable接口，用于实现程序的多线程。这种特性使得爬虫的控制器模块在进程分配管理等方面可以很好的实现。

在本文中的界面实现中，主要用到了Java的swing包。Swing包是Java类库中的UI工具包，提供给了开发者了创建自己的UI所需要的全部工具。Swing采用了MVC设计模式，即Model-View-Controller模式。Model即模型，它用于业务逻辑的处理，保存业务数据。View即视图，使用户和界面交流。Controller即控制器，它接收到用户的请求后，调用某相应的模型构件处理请求，然后决定用对应的视图显示返回的数据给用户。由于swing采用了这种模式使得它是一个轻量级的组件，支持双缓冲，在界面绘制时可以节省资源消耗。Swing是用纯Java实现的，没有本地代码，因此界面的移植性很好，可以在不同平台上有一致的呈现，并能提供好几种外观的选择[22]。

#### 4.1.2 MySQL

系统最后采集到的主题数据是存储于数据库中，本文采用了MySQL数据库。MySQL是由MySQL AB公司开发的一个小型的关系数据库管理系统，它由于体积小，运行速度快，提供对多种语言的支持和多种数据库连接途径，尤其是它的开源性和免费性使MySQL得到广泛的应用[23]。

为了方面对MySQL的管理和操作，本文系统采用了Navicat图形管理工具。Navicat可以在不同的平台上使用，它的用户界面使用户可以方便的建立数据库，建立数据表，设置数据属性，查看表，记录等。因为本文研究中，用户需要对搜集到的主题数据进行查看，所以Navicat的易操作性使得它成为本文数据库的前端图形界面的选择[24]。

要使得应用程序可以顺利的定义和操作数据库，得有一个驱动程序进行数据库的驱动，然后获得Connection对象来进行与数据库的实际的连接动作[25]。Java连接MySQL数据库的驱动是JDBC驱动，本文的数据驱动所用的程序包是 mysql-connector-java-5.1.15-bin.jar，获得连接的代码如下：

Connection con = null;

Class.*forName*("com.mysql.jdbc.Driver");

con=DriverManager.*getConnection*("jdbc:mysql://localhost:3306/test?useUnicode=true&characterEncoding=GB2312", username, password);

当成功获得Connection对象后，就可以通过createStatement和execute方法来进行数据定义和更新等操作。

4.2 源码视图

根据系统模块的划分，系统的源码实现其核心部分主要由三个包组成，源码视图如图4-1所示。

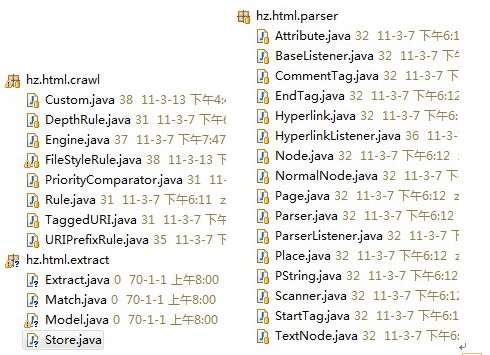


图4-1 源码视图

在crawl包中，包括了搜索过滤模块的功能实现和调度器的实现。其中过滤是由根据一些过滤规则实现的，比如深度规则DepthRule ，文件类型规则FileStyleRule和URI前缀规则URIPrefixRule，还有一些其它的参数都在Custom中指定。经过过滤的URL放在一个优先队列中，调度器Engine会根据用户指定的线程数来启动一些线程，用这些线程依次爬取队列中的URL指向的页面。

核心模块HTML解析器是由parser实现的。词法分析器是由Scanner实现的，它对源码字符串逐个字符的扫描，按照HTML词法规则解析出一个一个的标记（如StartTag、EndTag、CommentTag等），最后源码就成为了一个标记序列。然后，语法分析器Parser按照HTML的语法规则对标记序列遍历进行语法分析，解析出它的语法单元（如NormalNode、TextNode等）。对于解析出的语法单元会调用一些监听器来分析它的语义，比如HyperlinkListener会对监听超链接节点，如果是一个超链接，则会提取此超链接，传给过滤模块进行评价过滤。TitleListener会对监听标题节点已提取标题。

最后要介绍的包extract是重要的主题定制以及主题数据提取模块。其中Model类用于主题模板的建立，Match用于对经过解析器解析的网页进行主题匹配，如果匹配成功，用Extract类提取主题数据，最后Store连接数据库，把主题数据保存到数据库中。

4.3主要算法

4.3.1 网页搜索算法

在2.2节的搜索策略和网页解析的讲解中，介绍了对几种搜索策略和网页解析算法，在搜索策略中，深度优先策略在目前web数据量过于庞大的情况下容易使爬虫陷入陷阱，最佳优先算法常常陷入局部[26]。考虑到爬虫能实现查全率，并提供一定的按相关度对爬取的相关页面排序的功能，PageRank算法与主题无关，所以采取了类似HITs的基于入度hub值排序的宽度优先算法。

算法的伪代码如下：

初始化：用于存放待解析URI的优先级队列 wProcess为空，初始URL放入wProcess中，解析出初始页面中所有的链接links。

While（links中的每个链接link）｛

If（链接link之前没出现过）｛

标记link出现过；

把link加入wProcess中等待解析；

｝

//处理过是指已经抓取解析过

Else if（link已经处理过）｛

则增加link的URI的入度；

｝

//这种情况即link的URI已在wProcess中等待抓取解析处理

Else｛

把link从wProcess中移除；

增加link的URI的入度；

把link加入wProcess中；

｝

｝

｝

对wProcess中的每个URI经过抓取解析处理后得到的链接，又经过此算法的处理，依次循环直到所有的链接都遍历到。

4.3.2 HTML解析器

在3.2.3节的解析模块的讲解中，我们已经了解了网页解析器的组成和工作流程，在这里就详细介绍词法分析算法和语法分析算法。

4.3.2.1 词法分析器

词法分析是指对源程序的字符串进行从左到右逐个字符的扫描，按照一定的词法规则，识别出一个一个的标记token，把源程序变成一个等价的标记序列[27]。对于本文的词法分析来讲，源程序字符串即是HTML的源码文件，词法规则则是HTML文档规范的词法规则，最后识别出的标记即是tag和文本节点。本文的词法分析器是用一个自动机实现的，其状态转换图如图4-2所示。

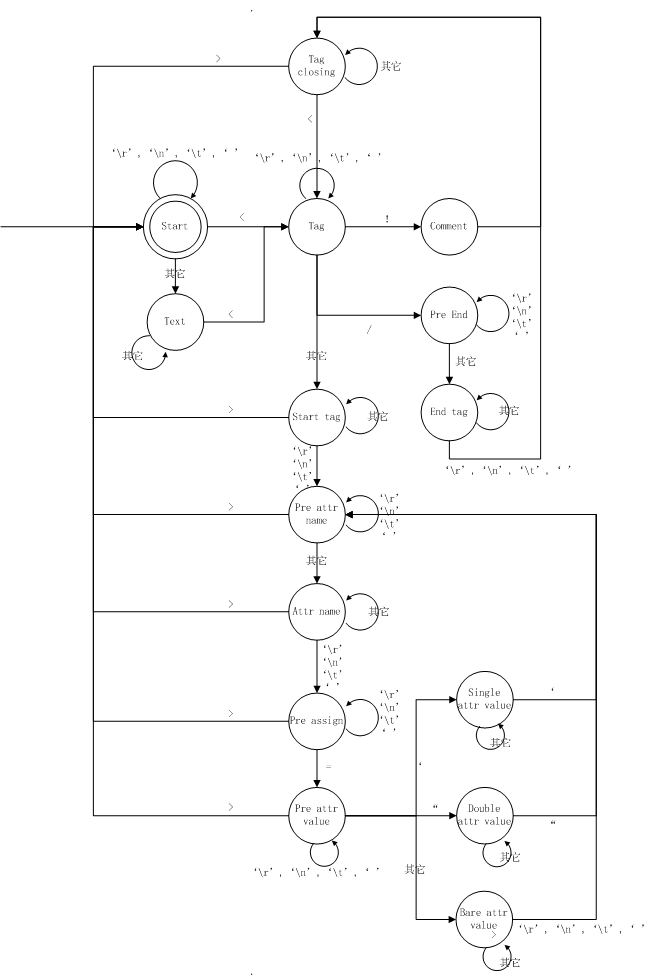


图4-2 词法分析器自动机

该自动机共有15个状态：分别是Start、Tag、Tag closing、Text、Comment、Pre End、Start tag、End tag、Pre attr name、Atrr name、Pre assign、Pre attr value、Single attr value、Double attr value、Bare attr value。

定义‘\r’，‘\n’，‘\t’，‘ ’这些空格符号为space，则词法分析器的状态转换过程说明如下：

Start：自动机的开始状态。如果遇到字符‘<’，则转到状态Tag；如果遇到space则保持开始状态；遇到其他字符转到状态Text。

Tag：表示一个标签的开始，可能是开始标签、结束标签或者注释标签等。如果遇到‘！’，则转到Comment状态；如果遇到‘/’，进入Pre End状态 ；如果遇到space则保持Tag状态；如果遇到其它字符，则转到Start tag状态。

Text：表示在文本节点中。如果遇到‘<’，则转到Tag状态；遇到其它字符，则保持Text状态。

Comment：表示是一个注释标签中。直接跳到Tag closing状态。

Start tag：表示在一个开始标签中。如果遇到；如果遇到‘space则转到Pre attr name状态；如果遇到‘>’，则开始标签结束，转到Start状态；如果遇到其它字符，保持Start tag状态。

Tag closing：表示标签进入预备结束状态。如果遇到‘>’，则标签解析完毕，转到Start状态；如果遇到‘<’，则进入Tag状态；遇到其它字符，保持Tag closing状态。

Pre End：预备结束状态，此状态过滤一些空格符合。如果遇到space则保持Pre End状态；如果遇到其它字符，转到End tag状态。

Pre attr name：属性名识别预备状态，过滤space；如果遇到space保持Pre attr name状态；如果遇到‘>’，则开始标签结束，转到Start状态；遇到其它字符，转到Attr name状态。

End tag：表示在一个结束标签中。如果遇到space转到Tag closing状态；遇到其它字符，保持End tag状态。

Attr name：属性名识别状态。如果遇到space转到Pre assign状态；如果遇到‘>’，则标签解析完毕，转到Start状态；遇到其它字符，保持Attr name状态不变。

Pre assign：赋值准备状态，即尚未遇到‘=’而准备遇到的状态，过滤‘r’，‘\n’，‘\t’，‘ ’等空格符合。如果遇到space保持Pre assign状态；如果遇到‘>’，则标签解析完毕，转到Start状态；如果遇到‘=’，则转到Pre attr value状态。

Pre attr value：在属性值赋值符号‘=’后，识别属性值的准备状态，过滤‘r’，‘\n’，‘\t’，‘ ’等空格符号。如果遇到‘>’，则标签解析完毕，转到Start状态；如果遇到space保持Pre attr value状态；如果遇到‘ “’（双引号）则转到Double attr value状态；如果遇到‘ ‘’（单引号），则转到Single attr value状态；如果遇到其它字符，则转到Bare attr alue状态。

Double attr value：表示双引号引用的属性值识别状态。如果遇到双引号‘ “’，表示双引号引用的属性值识别结束，转到Pre attr name状态；否则，如果遇到其它字符，保持Double attr value状态。

Single attr value：表示单引号引用的属性值识别状态。如果遇到单引号‘ ‘’，表示单引号引用的属性值识别结束，转到Pre attr name状态；否则，如果遇到其它字符，保持Single attr value状态。

Bare attr value：表示没有用引号引起的属性值识别状态。如果遇到space，转到Pre attr name状态；否则，如果遇到其它字符，保持Bare attr value状态不变。

词法分析器有一定的容错性，对于像< html>、<<html>等不合法的词都有一定的纠正和容错性。这使得在如今web上HTML源码有很多词法错误的页面存在的情况下也可以很好的对页面进行分词，使数据抓取工作得以继续。

4.3.2.2 语法分析器

语法分析常用于编译器中，是编译过程的一个逻辑阶段。语法分析器从词法分析器得到一个标记序列，这些标记就是一个个的词法单元，然后语法分析器将这些词法单元组合成各种语法短语，如“程序”“表达式”“语句”等[28]。语法分析器可以验证从词法分析器得到的词法单元串是否可以由源语言的文法生成，从而检验源程序是否编写合乎语法[29]。

在本文的语法分析中，分析的对象是由HTML源码经过解析得到的标记序列，这些标记序列由普通的标签和文本节点组成。语法分析器解析这个序列后，可以生成一颗或者几棵树，每个语法短语都是树的一个节点。语法分析算法步骤如下：

1. 初始化父节点parent为null，创建节点容器vector<node>用来保存所有节点树的根；
2. 从标记序列中获得下一个未处理的token，若没有未处理的token，则转至步骤6；
3. 如果token是一个开始标签，创建相应node节点：若父节点parent不为null，把新建的node节点加到该父节点下，如果node节点不是自我结束标签（自我结束标签是指不需要结束标签的原子标签，如<meta … />、<br/>等），将parent更新为node，否则，不更新parent；若parent为null，表示当前node是根节点，把node加入节点容器vector<node>中，将parent更新为node；
4. 如果token是个结束标签，说明该token要关闭某个节点。当前所有未关闭的节点记录在parent链表中，从parent开始，向上跟踪父节点，逐个与token进行匹配。若父链表中所有节点匹配失败，则认为该token是错误输入造成的，丢弃该token；否则，设匹配成功的节点为match，依次关闭从parent开始到match为止的所有节点（如果上层节点匹配到结束标签，下层未关闭的子节点将默认被关闭，这里假设html代码编写者漏掉了未关闭子节点的结束标签，这样做也与可省略结束标签的标签相吻合，如<p>标签），并且将parent更新为match节点的父节点（根节点的父节点为null）。当关闭某个节点时，触发节点解析完毕事件，调用所有监听者的回调函数，完成相关处理操作。
5. 如果token是文本标签，创建相应文本节点，若parent不为null，将新建的文本节点加到parent节点下，否则，将新建文本节点单独加入到节点容器vector<node>中。触发节点解析完毕事件，调用所有注册的回调函数。
6. 如果parent为null，表明匹配正确，算法终止；否则，当前有一个或多个节点处于未关闭状态，缺少相关结束标签，算法自动自底向上关闭这些节点，调用相应回调函数，算法结束。

原子标签包括<AREA><BASE><BASEFONT><BR><COL><FRAME>

<HR><IMG><INPUT><ISINDX><LINK><META><PARAM>。

语法分析器实现了一定的容错性，可以很好的包容并纠正源码中的语法错误。这些语法错误主要是开始标签与结束标签的不匹配问题，包括某些开始标签没有对应的结束标签进行关闭和有多出的结束标签两种情况，这两种情况的解析示例如下：

源码示例：<html><head><title>Search information！</title></head><body> java，code，programmer!<div>section1<p>section2</div> </div>

正确源码：<html><head><title>Search information！</title></head><body>java，code，programmer!<div>section1<p>section2</p> </div></body></html>

在源码示例中，标签<div>没有正确对应结束标签，使得两个结束标签中的一个成为多余的，而标签<body>缺少对应的结束标签。语法分析器分析后可以纠正这些语法错误，其解析过程如下：

对源码示例的解析：开始遇到开始标签<html>，因为父节点为空，所以把它加入节点容器中，因为没有遇到对应的结束标签</html>，所以<html>为父节点，往下遍历。接下来遇到开始标签<head>，<head>没遇到结束标签，也就是未关闭，所以加到父节点<html>下为其子节点，父节点更新为<head>节点。依次类推，开始标签<title>加到父节点<head>下为其子节点。然后是文本节点，也是数据节点，数据节点是叶子节点，文本节点加到父节点<title>下，父节点依然是<title>，触发监听器序列，继续解析。接下来遇到结束标签</title>，则要寻找与之对应的开始标签并关闭开始标签。从父节点开始寻找，父节点为<title>，其标签名和当前结束标签名一致，即为要找的开始标签，设置<title>对应的节点的结束标签为</title>即关闭之，设置父节点为<title>的父节点即<head>。依次类推，关闭<head>，父节点更新为<html>。然后获得下个标记，即开始标签<body>，依照前面所讲遍历，直到遇到第一个结束标签</div>，此时父节点是<p>，然后往上层查找对应的开始标签，当前父节点<p>名字不匹配结束标签，查找<p>的父节点<div>，匹配成功，则依次关闭<p>和<div>，父节点更新为<body>。接下来的标记是结束标签</div>，往上层查找，一直未找到匹配的开始标签，丢弃。此时标记序列结束，但是父节点还存在，说明有标签未关闭，则从当前父节点<body>开始往上层遍历，关闭所有未关闭的节点<body>和<html>，触发相应的侦听器事件。

4.3.3 主题定制算法

主题定制策略在模板模块中完成，用户首先指定用来匹配网页的主题模板，系统对模板代码进行解析，当爬取模块爬取完一个网页时，先将其保存到磁盘，然后对其解析，抽取出所有超级链接合并到URL集中，最后将解析得到的语法结构与用户注册的主题模板进行匹配，抽取用户感兴趣的数据，存放到数据库中的相应主题表中。

4.3.3.1 模板通配节点

在用户制定主题模板时，需要指定网页中的哪一部分数据是用户要抽取的，这需要引入一种指定方式，为了方便易用，采用通配节点的方式。

在模板网页中，引入自定义节点template-node当作通配节点，即这种节点能够匹配任意内容，该节点包含两个可选属性，一个是name，另一个是max。name表示该节点匹配内容的名称，作为插入到数据库中的主题表的列名；max的值是整数，是name指定的列的大小值。只有指定了name时，max才有意义。在数据库中name指定的列的类型为varchar(max)。若通配节点中，指定了属性name的值，而没有指定max值，则max值取默认值256。 假设存在如下主题模板tmpl1：

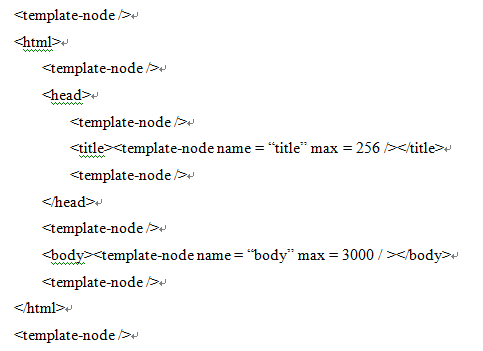


图4-3 主题模板tmpl1

图4-3中可以看出，定义了两个命名通配节点，一个在为title，一个为body，它们的列大小限定为256、3000。上图模板在数据库中对应的表名字为主题名tmpl1，具体定义如表4-1所示：

表4-1 主题模板tmpl1在数据库中对应的表

|  |  |  |
| --- | --- | --- |
| 列名 | 类型 | 用途 |
| ID | int auto\_increment | 自动增加，主键 |
| URI | varchar(1024) | 被匹配网页的URI |
| Title | varchar(256) | 通配节点title对应内容 |
| Body | varchar(3000) | 通配节点body对应内容 |

上述模板中还存在许多没有命名的通配节点，这些节点匹配到的内容将被丢弃，只有命名通配节点匹配的内容才会存入数据库中。定义未命名通配节点的目的是为了使模板更加通用，不会因为无关紧要的结构差异而导致模板匹配失败。假设存在如下图4-4所示的网页：

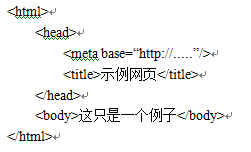


图4-4 实例网页图

若主题模板tmp1没有定义未命名通配节点，那么模板将无法与上述网页匹配，因为head节点下面多出了一个meta节点，这造成了模板结构与网页结构之间无法匹配成功。但是，如果在title节点前面存在一个未命名通配节点，那么meta节点的存在不会造成无法匹配，因为meta节点将与title前面的未命名节点匹配成功，使得整个网页匹配成功。

另一个通配节点叫命名子通配节点，此节点用语指定样本文档中的某个节点之间加入一个命名子通配节点作为其孩子节点，以使得在其它主题网页中此节点有孩子节点而用户又想要这个孩子节点的信息的情况下，模板可以成功匹配出此网页信息。比如在一个表格中，样本文档中的某个<td></td>并没有数据，但是其它网页的这个标签中有数据，而用户需要获得这个数据，那么就在<td></td>中添加一个命名子通配节点即可。

因此，对模板的定义需要谨慎，为了使模板更加通用，匹配到更多网页，需要适当的加入未命名通配节点，以容忍实际网页在未命名通配节点处与模板之间存在结构差异。

4.3.3.2 通配符匹配算法

假设在序列中用\*表示通配符，能够匹配一个或多个单元。模板序列是指在序列中包含一个或多个通配符\*的序列，如何判定一个序列是否匹配给定模板序列，且找出每个通配符匹配的结果。运行算法前，需要对模板进行预处理，将模板中相邻的多个通配符合并为一个通配符，确保不会有相邻的通配符出现。

下面是匹配算法描述（算法描述中从a到b指的是闭区间[a,b]）：

1. 定义变量ifrm、ito、jfrm、jto，ifrm和ito分别表示模板序列未匹配部分的开始、结束位置，jfrm和jto分别表示待匹配序列未匹配部分的开始、结束位置；
2. 初始化ifrm、jfrm为开始位置0，ito和jto分别表示模板序列和待匹配序列的结束位置；
3. 匹配首部非通配单元。若模板序列中ifrm处的单元不是通配符\*：比较模板序列中的ifrm处单元和待匹配序列的jfrm处单元，如果两者不相同，则匹配失败，算法终止；否则，将ifrm和jfrm值都自增一，循环匹配直到模板序列中ifrm处的单元是通配符\*或者匹配完毕；
4. 匹配尾部非通配单元。若模板序列中ito处的单元不是通配符\*：比较模板序列中的ito处单元和待匹配序列的jto处单元，如果两者不相同，则匹配失败，算法终止；否则，将ito和jto值都自减一，循环匹配直到模板序列中ito处的单元是通配符\*；
5. 到此处，模板序列ifrm处和ito处都是通配符\*。如果模板序列未匹配部分只剩下一个单元（即ito和ifrm值相同，模板序列只剩下一个通配符\*未匹配），则，模板序列中ifrm处的通配符匹配待匹配序列中剩下的所有未匹配的部分（即待匹配序列中从jfrm到jto处的所有单元），记录匹配结果，整个匹配过程完毕，算法终止；否则，执行步骤6；
6. 在模板序列中从ifrm+1处往前查找，直到找到下一个通配符，记录其位置next，得到不包含通配符的模板子序列[ifrm+1, next-1]（即从模板序列中的第ifrm+1处开始，到next-1处结束），用len表示子序列长度（len为next – ifrm – 1），在待匹配序列中从jfrm处开始逐步向前搜索模板子序列[ifrm+1, next-1]，若在j处使得待匹配序列的子序列[j, j + len - 1]与模板子序列[ifrm+1, next-1]相同（即在j处搜索成功），将模板序列中的ifrm处通配符匹配待匹配序列中从jfrm到j-1处的所有单元，记录匹配结果，同时，将ifrm更新为next、jfrm值更新为j+len，转至步骤5继续执行，若搜索子序列失败，模板序列和待匹配序列匹配失败，算法终止；

在算法的第6个步骤中，搜索子序列采用的是首次匹配策略。假设模板序列为：\*x{\*xi}\*，花括号表示重复一次或多次。如果子序列能够匹配成功n次，如果采用第i次方略，整个序列能够匹配成功，那么采用第i-1次方略也能够使序列匹配成功，因为在x后紧跟这通配符\*，通配符能够匹配从第i次变到第i-1次时多出来的单元。所以在n次不同的匹配方法中，选择第一次匹配方法，不会有任何匹配问题。唯一的影响是，将多余的、当前通配符可匹配的单元往匀给后面的通配符。其佐证例子如图4-5所示：

E

-

b

c

-

b

c

-

w

-

x

y

z

E

-

b

c

-

b

c

-

w

-

x

y

z

b

c

w

\*

\*

\*

b

c

w

\*

\*

\*

图4-5 通配符策略匹配示意图

4.3.3.3 主题模板匹配

对模板网页进行解析后得到Node序列，对爬取的网页解析后也得到Node序列。需要将爬取网页的Node序列跟主题模板的Node序列进行匹配，抽取用户感兴趣的数据。

Node是一个抽象结构，具体子类别分TextNode和NormalNode。TextNode包含的是文本，没有开始结束标签，因为它是叶子节点，不能包含孩子节点。NormalNode表示正常节点，有开始结束标签，且可以包含孩子节点。判断两个Node是否能够匹配时，需要根据Node的实际类别进行区分对待。

下面给出Node1匹配Node2的算法实现：

Bool match(Node1 , Node2) {

If (Node1是TextNode) {

If (Node2是TextNode)//不比较文本内容，因为算法匹配的是结构

Return true;

Else

Return false;

}

Else ｛ // Node1是NormalNode

If (Node1是通配节点)

Return true;

If (Node2是TextNode)

Return false;

If (Node1节点类型和Node2节点类型不相同)

Return false;

If (Node1和Node2都不包含子节点)

Return true；

If (Node1不包含子节点或Node2不包含子节点)

Return false;

Return match(Node1的子节点序列, Node2的子节点序列);

｝

}

模板节点序列和爬取页面的节点序列匹配算法match(模板节点序列, 爬取页面的节点序列)采用上小节中的通配符匹配算法。只不过在序列匹配算法中，节点之间的匹配用上面的match(Node1 , Node2)算法。这两个函数之间存在间接递归调用关系。

下面给出一个匹配例子对算法进行阐述：

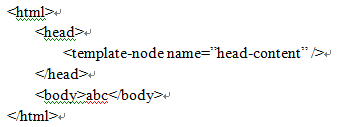


图4-6 模板示例

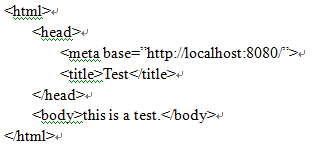


图4-7 网页实例

上面模板图4-6和网页实例图4-7的匹配过程如下：

=>首先匹配模板节点序列[html]和网页节点序列[html]；

=>匹配模板节点html和网页节点html；

=>匹配模板节点html的子节点序列[head, body]和网页节点html的子节点序列[head, body]；

=>匹配模板节点head和网页节点head；

=>匹配模板节点head的子节点序列[template-node]和网页节点head的子节点序列[meta, title];

<=序列[template-node]匹配成功，模板节点title-content匹配网页节点序列[meta, title]；

<=head节点匹配成功

=>匹配模板节点body和网页节点body

=>匹配模板文本节点“abc”和网页文本节点“this is a test.”；

<=文本节点匹配成功

<=body节点匹配成功

<=序列[head, body]匹配成功

<=节点html匹配成功

<=序列[html]匹配成功

从上述匹配过程得知，模板能够成功匹配上述网页实例，通配节点titile-content匹配的内容为<meta base=”http://localhost:8080” /><title>Test</title>。

4.3.4 模板优化

通过对网页结构的观察比较以及一些实验结果分析，发现这样的现象，同一主题的网页虽然格式相似，但总是存在因为页面跟模板相比缺少某个节点（比如表格数据）或者多出某个节点，而导致此页面无法匹配成功的情况，所以为了使模板更加通用，系统需要对用户制定的模板进行一些优化。模板优化过程主要有以下三步：

输入：用户制定的模板。

1. 把所有的文本节点转换为匿名通配节点，以使模板能匹配其它相似的但缺少此文本节点信息的页面。
2. 对于所有可以拥有孩子节点，孩子节点却为空的节点，添加一个匿名通配节点作为其孩子节点，以使得模板可以匹配到其它相似的但在此节点处拥有孩子节点的页面。
3. 对模板的语法结构树进行层次遍历。对每一层的节点，如果第一个节点不是通配节点，则在第一个节点前加一个匿名通配节点；往后遍历此层节点，如果两个相邻的节点都不是通配节点，则在这两个节点之间加一个匿名通配节点；遍历至此层最后一个节点，如果此节点不是通配节点，则在此节点后加一个匿名通配节点。此层遍历完毕，遍历下一层直到遍历完整棵树。

输出：优化后的模板。

4.4用户接口

用户接口是用户和系统进行交互联系和传递信息的渠道，使用户能发送自己请求和信息，从而获得所需要的反应信息在界面中呈现[30]。本文系统的界面主要由主界面和配置界面组成。

主界面由三个面板区组成，分别是Web结构视图面板，模板网页结构面板和Page 源码面板。Web结构视图面板呈现整个网站的站内链接结构，网站的指定是由起始网址的URL决定。模板网页结构面板展示建模所需的样本文档的页面内容结构，是根据样本文档的语法分析得到的一个语法结构树，用户可以在树上进行数据节点或者子树的选择来进行建模。Page 源码面板用于展示样本文档的源码结构，定位节点在源码中的位置以帮助主题建模。

配置面板是用来让客户进行一些参数配置的界面，对于搜索过滤等一些限定必不可少。参数设置主要有起始URL有关设置，样本URL及链接结构描述设置，下载限定，网络礼节等方面的设置。配置面板如图4-8.

C:\Users\Administrator\AppData\Roaming\Tencent\Users\510544021\QQ\WinTemp\RichOle\3DMZVTPDI3SAPK]9%Q4J2PF.jpg

图4-8 配置面板

要想在数据库中建表，保存数据，首先得连接数据库，要输入服务器名称、数据库名、用户名、密码进行数据库连接[31]。另外，要输入开始URL，URL匹配前缀，线程数等参数。



图4-9 系统主界面

如图4-9所示，Web结构视图是对整个网站的链接结构的视图，采用了动态加载，双击某个链接，则此链接指向的页面里包含的所有内链接才被加载，内连接作为此链接节点的子节点展示。

建模过程如图4-10所示，当选择模板中的一个节点时，该节点在源码中的位置就被选中，以使用户精确找到需要的节点以建立模板。当选择一个节点右击时，会出现一个弹出菜单，包括四个菜单项：转换为命名通配节点，转换为匿名通配节点，转换为命名子通配节点，撤销通配节点。

如果一个节点被转换为命名通配节点，则此节点内容将被存入数据库中，需给出一个命名作为该数据所在列的列名。如果节点被转换为匿名通配节点，则此匿名节点的父节点的所有孩子节点全丢弃。

所以一个数据如果是需要的主题数据，则把它转换为命名通配节点，否则，它不是主题数据，而且有可能这个节点在其它网页中又有着扩展出的节点，则可以把它设为匿名通配节点，以使得其它相似结构的网页能丢弃掉和模板不同的部分，使得匹配成功，提取出需要的主题数据。另外，如果模板中没有某个数据，但是其它网页中很可能有这个数据，而用户需要提取这个数据，则在需要此节点的位置添加一个命名通配子节点。

点击“从网址加载模板”输入建模URL，系统获得此URL指向的页面，然后进行建模。

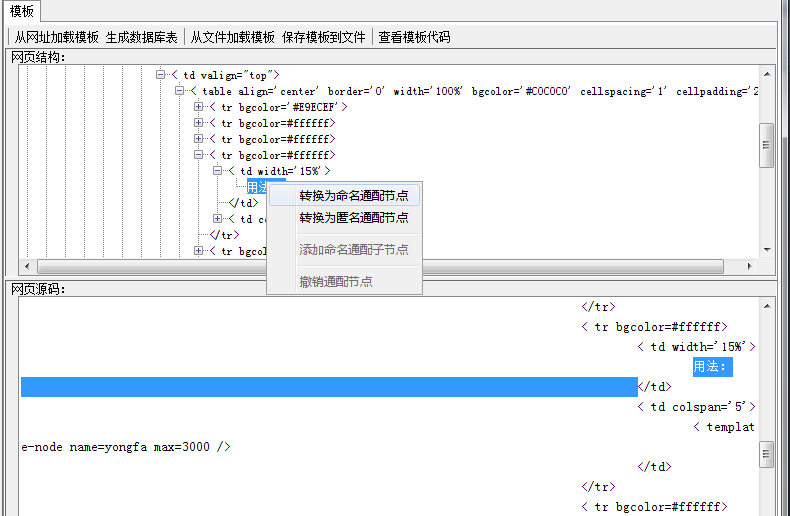


图4-10 建模图示

若节点转换为命名通配节点，其命名格式为名字：长度，如图4-11所示。

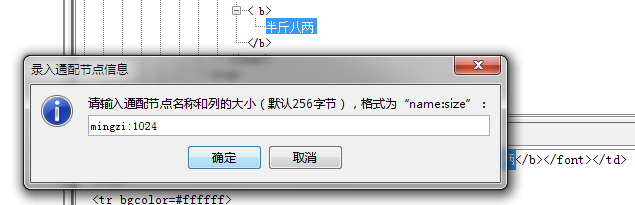


图4-11 命名通配节点命名示例

命名完成后，此节点变成一个模板节点，会包含一个name属性和一个max属性表示长度。若是匿名节点，则没有这两个属性。图4-11中的节点转换为命名通配节点后变成一个命名模板节点，如图4-12所示。

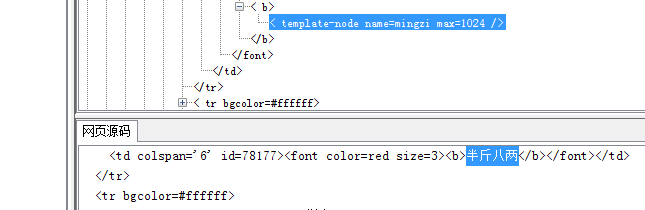


图4-12 命名通配节点图示

对于一个需要添加数据节点的节点，使其添加一个命名通配子节点，如图4-13中，节点<td></td>之间本无数据，所以加上命名子通配节点后，可以匹配其它在此<td></td>之间有数据的节点，并提取出这些数据。

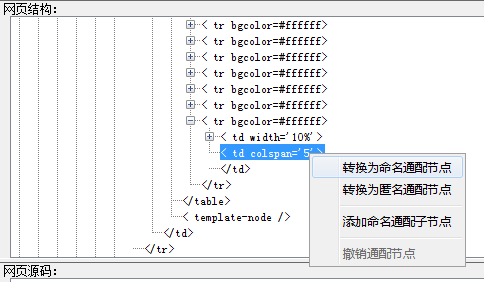


图4-13 命名子通配节点选择

当建模完成后，可以点击“保存模板到文件”按钮把模板保存到文件中，此时的模板只是一个中间模板，也可以点击“从文件加载模板”来对某个模板进行查看。

点击“生成数据库表”在数据库中生成一个数据表。此时点击“查看模板代码”，会出现一个面板，包括数据表和模板代码，如图4-14.

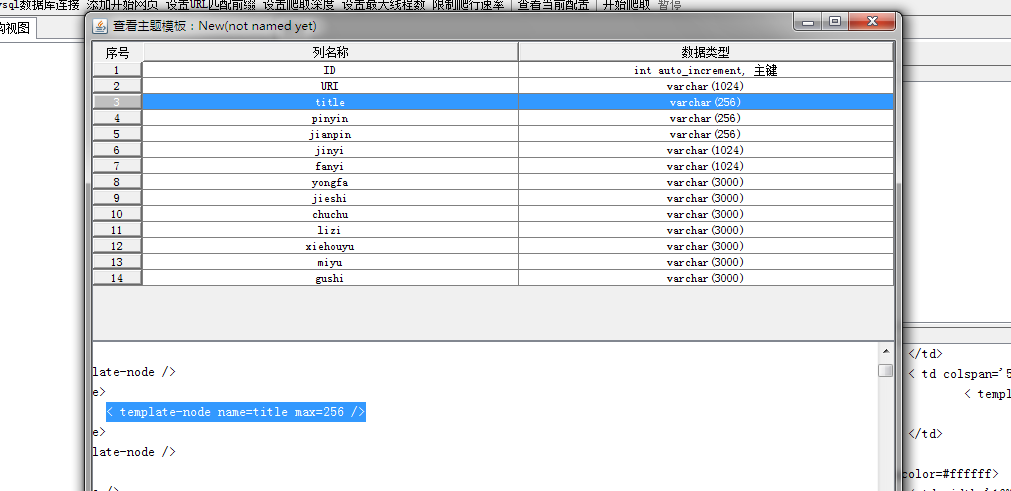


图4-14 查看数据表及模板代码

如果模板确定建立无误，则点击开始爬取按钮，系统即开始抓取数据，在主界面的最下方的状态栏会显示当前的运行状态，包括网页总数，已处理的和待处理的数目，以及线程数。如图4-15所示。



图4-15 系统运行中的状态栏

在抓取过程中，按住暂停按钮可以暂停项目的运行。可以通过Navicat’查看数据库中已匹配的数据。

当项目运行完毕后，状态栏会显示运行完毕，以及总共爬取的数目，如图4-16所示。



图4-16 项目运行结束的状态

4.5本章小结

本章首先介绍了系统的详细的实现，包括实现平台和技术、源码结构、HTML解析算法、主题定制和匹配算法、模板的优化算法，最后介绍了用户接口，展示了系统主界面，并说明了用户操作接口和使用步骤。

第五章 实验设计及结果分析

通过前几章的详细介绍，对聚焦爬虫的原理，系统架构以及实现都有了比较具体的了解，本章就通过一个实际的实验操作来进一步加深对本文设计的聚焦爬虫的认识和了解，并通过对实验结果的分析来验证此系统的功能和性能。

5.1实验设计

为了说明本文实现的聚焦爬虫的功能及其面向主题的思想，本章采用两个不同类型的网站进行实验。这两个网站分别是成语词典在线网站和seekda服务查询网站。

5.1.1 参数设置

其参数设置如表5-1和5-2所示：

表5-1 实验主要参数表

|  |  |  |
| --- | --- | --- |
|  | 实验一 | 实验二 |
| 起始URL | http://cy.5156edu.com/cymore.html | http://webservices.seekda.com/providers/; |
| 样本URL | http://cy.5156edu.com/html4/213.html | http://webservices.seekda.com/providers/galciv2.com/ForumService |
| URL前缀限制 | http://cy.5156edu.com/ | http://webservices.seekda.com/ |
| 深度 | 无限制 | 无限制 |
| 线程数 | 40 | 30 |
| 抓取速度 | 1.5个/s | 5个/s |

5.1.2 主题定制

实验一，选择的是一个在线成语词典网站，该网站目前记录了四万多条成语，对每个成语都有相应的解释，翻译，典故，近义词，反义词等，这些信息都在一个统一格式的表格中，如图5-1所示。

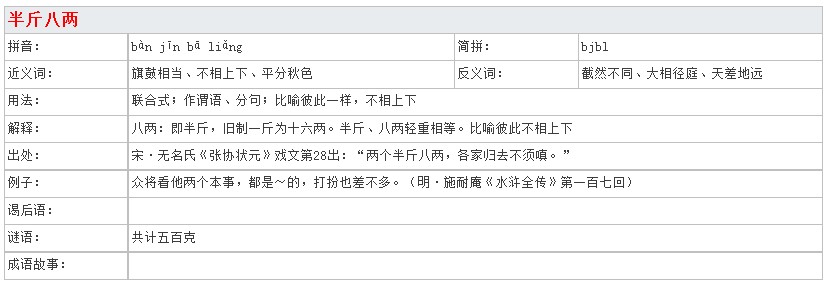


图5-1 成语信息表示例

现在的目标数据即是把所有成语的这些信息抽取出来，所以建立模板时，要把这些信息所在的文本节点设置为命名通配节点。另外由于此成语“半斤八两”中的简拼，反义词等很多信息在其它的成语信息表中都没有，但是有的成语又可能有，而这些信息又被要求抽取出来，所以如果以此成语的页面建模则系统会在缺失信息的地方（如<td></td>中间）添加一个命名子通配节点用于匹配其它拥有此信息的页面。

下面是根据URL：http://cy.5156edu.com/html4/213.html建模后的模板，查看模板以检查模板是否建立正确。如图5-2所示。



图5-2 用户制定生成的中间模板

模板建立完成后，可以保存模板到文件中，保存的模板即是中间模板，如图5-2所示。当点击“生成数据库表”后，系统会对中间模板按优化步骤进行优化，然后在数据库中建立一个表格。开始运行项目，爬虫即开始抓取网页并抽取数据。

实验二，实验的对象是一个服务查找网站，sourceforge.net网站搜集了众多开源软件，对每个软件，都有相应的项目信息和项目开发者信息等。

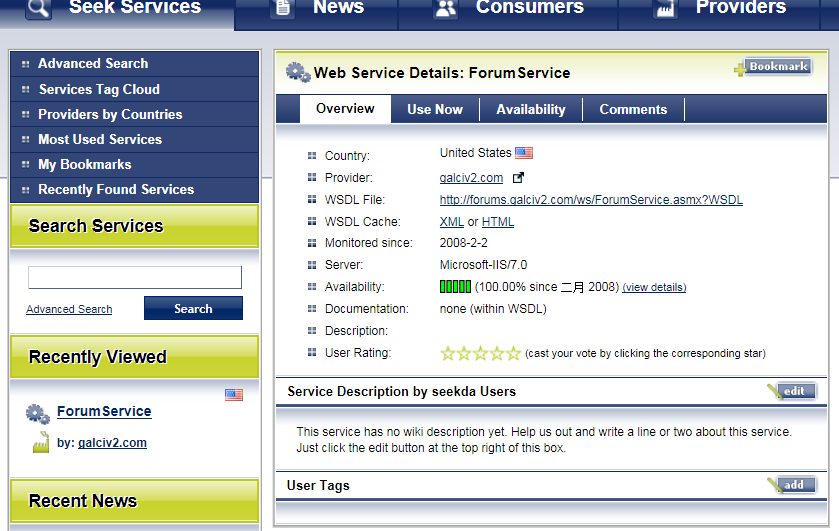


图5-3 service的信息示例图

在实验二中，要提取的就是这些项目的主要信息，包括Overview和Use Now里的所有信息。

5.1.3 运行结果

上述实验结束后，都会在数据库中生成相应的主题的数据，其运行结果的数据记录分别如图5-3，图5-4所示。



图5-3 成语信息数据库记录

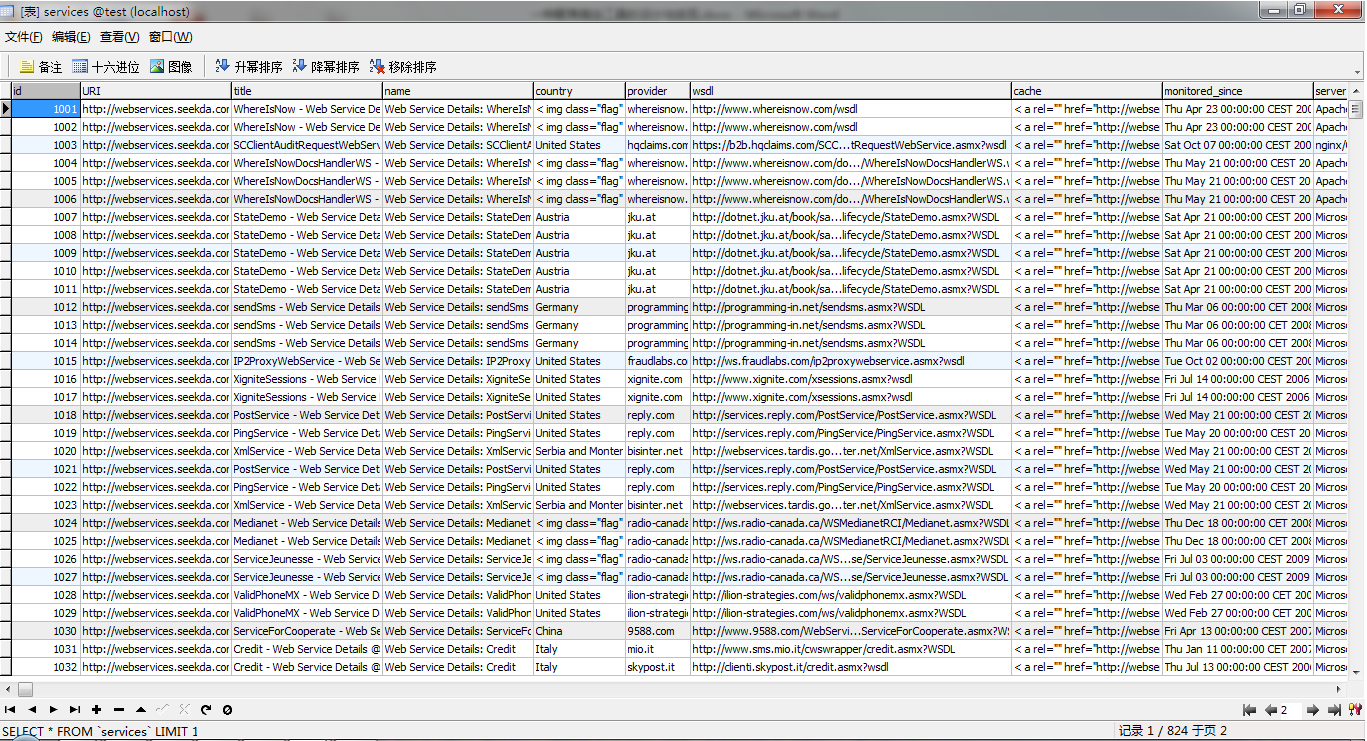


图5-4 service数据库记录

5.2结果分析

对于以上的实验结果分别从网站记录总数，抓取页面总数，抽取数据数目三个数字进行统计比较，比较结果如表5-2所示。

表5-2 实验结果统计表

|  |  |  |  |
| --- | --- | --- | --- |
|  | URL总数 | 匹配数据数目 | 主题数据匹配率 |
| 实验一 | 32794 | 32268 | 98.4% |
| 实验二 | 59986 | 2560 | 4.3% |

注：URL总数是指本文系统按照宽度优先策略搜索到的URL的总数，其中包括主题无关的网页和无效或过时的URL；而匹配数据数目是指用模板对抓取的网页进行匹配之后成功匹配并抽取的数据的数目。主题数据匹配率=匹配数据数目/URL总数\*100%。

对于实验结果的数据，做了一些分析发现：实验一中，链接结构比较简单，主题相关的页面比较密集，非主题URL指向的页面大多是些目录页面等，所占较少，所以其匹配率高；而在实验二中，链接结构复杂，有大量的主题无关的网页，比如http://webservices.seekda.com/services/tags/tourism，是以tag搜索到的服务列表，是目录页面，另外还有很多服务页面无法获得（如 http:// webservices.seekda.com/providers/edit\_logo/fraudlabs.com ）等原因，导致了主题URL不够集中，使得匹配率很低。

根据以上实验结果的统计可知，本系统在搜集数据时，在一定程度上，可以达到用户预期的效果，而最终可获得的数据记录的多少，依赖于网站的主题网页的可获得网页的多少，对于主题密集的网页，匹配率会比较高，在无关URL上浪费的资源就少。另外，对主题相关的网页的查全率依赖于用户制定的模板的准确性，模板制定的不能太严格，以使模板不够通用，降低查全率，但是也要防止模板过于通用，导致一些主题无关的页面也匹配到。

5.3本文聚焦爬虫工具的应用

通过以上的实验得知，本文的聚焦爬虫工具可以比较具体的获得用户需要的信息，尤其在学习研究等工作中。下面结合本人硕士期间的科研项目说明此工具主要应用的方面，有以下两方面：

一是对原始信息的获取和存储为目的，比如可以抓取服务信息注册在科研项目中实现的服务注册平台上，以供服务查询；或者如实验一中的成语信息的抓取产生一个成语库以供成语的学习等[32] [33]。

二是获得原始信息经过处理得到更多的信息，比如获得mushup的信息或者软件说明文档以供信息分类；或者利用抓取到的某个领域的信息建立领域知识等。

5.4本章小结

本章对两个不同领域的网站进行实验，通过设置一定的参数，运行系统获得相关主题的数据，然后对抓取的结果进行分析，来评价本文系统的功能，最后结合实际的科研项目中的研究介绍了此系统的实际运用领域等。

第六章 总结与展望

6.1总结

随着网络的迅猛发展，互联网上的信息量也急剧增长，如何在这些信息中准确的找到并搜集自己想要的信息，是本文研究的内容。首先，要把包含所需要信息的网页从Web上爬下来，然后再通过对目标数据的描述制定模板，从这些网页中用模板匹配出需要的数据。

本文的研究成果主要有：

1. 经过研究比较了多种网页搜索策略，考虑了这些搜索策略的优缺点，提出一种结合了宽度优先和基于网页粒度的HITS算法。此算法层次遍历网页内的链接，以获得比较高的查全率，其基于链接入度值的排序算法可以使主题网页按主题相关度排序。
2. 实现了HTML解析器，它由词法分析器和语法分析器组成。该解析器有很好的容错性和纠正能力，可以在目前Web上大多网页语法不规范的情况下，也能正确的解析出网页的语法结构。解析器的实现使得对新链接URL的提取和之后的主题定制和主题数据提取变得容易实现。
3. 提出一种主题定制和主题数据匹配提取方法，这种方法建立在相同主题的网页常常有着相似结构的思想的基础上。主题定制包括目标网页内容结构的描述和目标数据模式的描述，而主题匹配采用了模板匹配相似树提取出主题数据的方法实现。
4. 实现了一个具有图形化界面的聚焦爬虫系统，通过本系统，用户可以方便的查看到整个网站的链接结构，可以对样本网页的内容结构以及数据节点的内容进行查看，并通过结构树制定主题模板；提取出的主题数据可以通过Navicat查看MySQL进行查看。

6.2展望

本文实现的聚焦爬虫系统基本上已经按照初期的设计实现了其主要的功能，但总结自己工作，结合着目前Web信息的发展趋势和其它学者对聚焦爬虫的研究工作，本文还在以下方面存在不足，需要在后期的工作中进一步改进：

1. 本文提出的网页搜索策略，是建立在宽度优先的基础上，这种策略肯定会造成对某些和主题毫无关联的网页进行分析，从而浪费了系统资源，影响系统的运行效率。所以在今后的工作中，需要采用某种分析或过滤方法使得搜索算法既能保证查全率，又可提高搜索效率。
2. 本文对主题的描述方面，主要是基于目标网页结构和主题数据模式的，这种算法要求主题网页都要求在结构上有较高的相似性，可以在今后的工作中研究对结构相似度要求降低的匹配算法。而主题数据模式并没有考虑到数据的语义性，所以在今后可以添加语义角度的主题数据的研究内容，可以探讨领域划分等内容。
3. 如今的Web网页上，由于Ajax等技术的发展和应用，网页的动态内容比较普遍，但是本文的研究中对这些工作都还没有涉及，但这些工作却会是以后工作内容中必不可少的一项。

参考文献

1. 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.
2. 齐保元.网络爬虫[DB/OL]. http://baike.baidu.com/view/ 284853.htm, 2006-1 -10/2011-4-20.
3. Lefteris Kozanidis. An Ontology-Based Focused Crawler[J]. Computer Science, 2008, 5038: 376–379.
4. M.Yuvarani，N.Ch.S.N.Iyengar，A.Kannan. LSCrawler: A Framework for an Enhanced Focused Web Crawler based on Link Semantics[A]. The 2006 IEEE/WIC/ACM International Conference[C]. Washington, CS Press,2006.794-800.
5. Wenxian Wang, Xingshu Chen, Yongbin Zou. A focused crawler based on naive Bayes classifier[A]. Third International Symposium on Intelligent Information Technology and Security Informatics[C].Washington, CS Press,2010.517-521.
6. Yulian Zhang，Chunxia Yin，Fuyong Yuan. An application of improved PageRank in focused crawler[A].Fourth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2007)[C]. Washington, CS Press,2007.331-335.
7. Rob Miller. Websphinx[DB/OL]. http://www.cs.cmu.edu/~rcm/websphinx, 2002-7 -8/2011-4-20.
8. Li Peng，Teng Wenda. A focused web crawler face stock information of financial field[A]. Intelligent Computing and Intelligent Systems (ICIS)[C]. Newyork:IEEE,2010.512-516.
9. 周建梁.聚焦爬虫原理及关键技术研究[J].科技资讯，2008,(22):26.
10. 林海霞，司海峰，张微微. 基于Java技术的主题网络爬虫的研究与实现[J].微型电脑应用，2009,25(2): 56-58.
11. 语法分析.http://baike.baidu.com/view/487037.htm#sub487037, 2011-1-20/2011 -4-20.
12. 姜梦稚.基于Java的多线程网络爬虫设计与实现[J].微型电脑应用，2010,26(7): 56-58.
13. 蔡欣宝，郭若飞，赵朋朋，崔志明. Web论坛数据源增量爬虫的[J].计算机工程. 2010.5,36(9)：285-287.
14. 张丽敏.垂直搜索引擎的主题爬虫策略[J].电脑知识与技术,2010.5,6(15)：3962-3963.
15. 杨靖韬，陈会果.对网络爬虫技术的研究[J].科技创业月刊，2010,(10):170-171.
16. 刘磊安，符志强.基于Lucene.net网络爬虫的设计与实现[J].电脑知识与技术， 2010.3,6(8)：1870-1871，1878.
17. 叶勤勇.基于URL规则的聚焦爬虫及其应用[D].浙江：浙江大学硕士学位论文，2007.5.
18. 夏道勋，谢晓尧.基于Web的专用爬虫的研究[J].贵州师范大学学报（自然科学版），2009.8，27（3）：92-95.
19. 杨志伟，王鑫.基于本体的气象领域聚焦爬虫[J].中国管理信息化，2011.2,14(4)：60-62.
20. Pedro Huitema, Perry Fizzano. A Crawler for Local Search[A]. 2010 Fourth International Conference on Digital Society[C]. New York:IEEE, 2010.86-91.
21. Qing Gao, Bo Xiao, Zhiqing Lin, Xiyao Chen, Bing Zhou. A High-Precision Forum Crawler Based on Vertical Crawler[A]. IEEE International Conference on Network Infrastructure and Digital Content[C]. New York:IEEE, 2009.362-367.
22. 孙立伟，何国辉，吴礼发.网络爬虫技术的研究[J].电脑知识与技术, 2010.3,6(15)：4112-4115.
23. 张成奇.支持Ajax的Deep Web爬虫设计与实现[D].上海：上海交通大学工程硕士论文，2009.
24. 罗兵.支持AJAX的互联网搜索引擎爬虫设计与实现[D].杭州：浙江大学硕士学位论文，2007.
25. 刘朋，林泓，高德威.基于内容和链接分析的主题爬虫策略[J].计算机与数字工程， 2009,37(1)：22-24，80.
26. 陈晨.基于主题爬虫的个性化搜索引擎技术研究[J].科技信息，2010,(31):87.
27. Punnawat Tadapak, Thanaphon Suebchua, Arnon Rungsawang. A Machine Learning based Language Specific Web Site Crawler[A].13th International Conference on Network-Based Information Systems[D]. New York:IEEE,2010.155-161.
28. 郑志高，刘庆圣，陈立彬.基于主题网络爬虫的网络学习资源收集平台的设计[J].中国教育信息化，2010.01：36-38.
29. 邹海亮.可定制的聚焦网络爬虫[D].上海：东华大学硕士学位论文，2008.12.
30. 倪贤贵.聚焦爬虫技术研究[D].无锡：江南大学硕士学位论文，2008.04.
31. Li Wei-jiang,Ru Hua-suo,A New Algorithm of Topical Crawler[A]. 2009 Second International Workshop on Computer Science and Engineering[C]. New York:IEEE,2009.443-446.
32. Shan Lin,You-meng Li,Qing-cheng Li. Information Mining System Design and Implementation based on Web Crawler[A]. System of Systems Engineering[C]. New York:IEEE,2008.1-5.
33. Pooja gupta,Mrs.Kalpana Johari. Implementation of Web Crawler[A]. Second International Conference on Emerging Trends in Engineering and Technology [C]. New York :IEEE ,2009.838-843.

致 谢

在本文的完成之际，首先我要衷心地感谢我的导师李兵教授。在两年的研究生学习阶段中，李兵教授一直以严谨的治学态度，平易近人的师长风范影响和指导着我的成长。

在学习和研究上，李老师要求比较严格，他严于律已，对我们也期望比较多。在研究生生活的一开始，李老师就跟我们探讨学术领域划分，指导我们根据个人的兴趣所长制定研究方向和学习计划，并在以后的两年时间里，督促和帮助我们完成学习内容。在生活上，李老师平易近人，对我们细心关怀，时时纠正着我们的错误，并要求我们团结一致，互相帮助，使得整个团队气氛活跃温暖。

而我的论文也就是在这样的团队的帮助关怀下完成的，所以我要感谢李老师带领下的整个研究团队的所有成员。感谢曾诚和潘伟丰以及其他博士师兄师姐对我的指导和关心，特别是曾诚和潘伟丰两位师兄积极乐观的学习和生活态度深深影响和激励着我，他们是我学习的楷模。感谢同门的李杉、陶汉涛、朱子龙、周晓燕、覃叶宜等同学，在一起的研究学习中，我们相互讨论，相互帮助，他们的见解和意见使我获益匪浅，研究生生活变得精彩和充实。

感谢实验室09级的所有同学，在两年的时间里，我们一直团结互助，共同进步。

感谢我的父母，你们一直是我前进的动力和坚强的后盾！谢谢你们一直以来对我的支持和鼓励！

最后，感谢所有关心我、帮助过我的人！

在读期间参与的科研项目

[1] 国家重大基础研究发展计划（国家973计划项目），“需求工程——对复杂系统的软件工程的基础研究（No. 2007CB310800）”，2009-2011．

[2] 国家自然科学基金项目，“基于软件网络的软件度量研究（No.60873083）” ，2009-2011．

[3] 湖北省杰出青年基金项目，“网络化软件复杂性度量及其应用研究（No. 2008CDB351）”，2009-2010．

[4] 国家“863计划”专题课题，“复杂信息资源语义互操作性应用集成的机制与方法（No.2006AA04Z156）”，2009．