摘 要

信息技术的发展对全世界来说都产生了极大的影响，是当前高技术发展中的主流技术，因为信息在人类生活中无处不在，无时无刻的不影响着人类的生活，对信息的获得和挖掘成为科学界关注的焦点。在互联网发展的初期，信息量和需求量都比较少，那么用户可以很容易的找到自己想要的信息，不过，随着互联网发展越来越迅速，这个事情变得越来越难了。搜索引擎的产生使在互联网上查找信息又变得相对容易了。但是，在海量的数据和异构的信息中存在这大量的网页噪音，这些噪音严重影响了搜索引擎系统的服务质量，降低了搜索结果的准确度，也增加了服务器处理过程的时间和空间上的开销。

首先，本文详细介绍了在网页净化系统实现中所用到的关键技术，主要包括文档对象模型 DOM 的定义及其特点；网页结构，其中包括网页标签树表示，网页如何用网页标签树表示，如何用 DOM 树表示；中文网页分块技术，通过结合 DOM 树以及 HTML 的一些重要标签对网页进行分块同时，本文来列举了中文网页分块的一些规则，结合对这些技术的了解，有助于对本研究课题功能实现的理解。

然后，本论文分析了信息网行业搜索引擎的架构：WebServer&SO，CACHE 管理系统，最新库，数据管理系统，及其网页净化系统。分别对每个子系统的功能做了详细的阐述，并详细介绍了各个子系统之间的关系，并对整个搜索过程做了详细的说明：用户从 Web 端的 CGI(通用网关接口)程序输入查询串，CGI 程序对它进行合理的分割并把查询串传入搜索系统，把查询到相关词的页面传入网页净化系统，最后把净化后的页面通过 Web Server 端的 CGI 程序显示在浏览器中。网页净化系统在整个行业搜索引擎中的作用：通过 Web 端的 CGI 程序显示净化后的结果，并对 CGI 做了介绍。网页净化系统 PageClean 是本文介绍的重点，也是本论文的核心部分。本文详细阐述了 PageClean 系统的架构，实现该系统依据的规则，该算法的主要思想及实现算法的流程。

最后，论文给出了网页净化系统 PageClean 测试方法，并根据测试数据得出结论：网页净化系统 PageClean 无论是在净化速度还是在净化效果上都具有较好的性能，达到了预期的目标。

关键词：网页净化；网页噪声；超文本标记语言；网页结构；万维网

Abstract

The development of information technology has effect on the whole world, it is the main technology during the development of information technology. As it exists nowhere in the human life and effects human life, getting information becomes the focus in the science circles. In the early days, there is less demand of information, so, people can get information that they want easily. But as world wide web develops, it becomes very hard to do. Search engine becomes it easy again. But there is a great deal of web noise in such great mount of webs, it reduces the nicety of search engine, and increases the load of server.

First, the thesis introduces the key technology of the web purification system which includes the definition and characteristic of DOM(Document Object Model) technology; Web structure: denotation of web label tree, how does the web structure denote with web label tree and DOM tree; Web page segment technology: with the DOM technology and some important HTML label, we can segment the web page. Meanwhile this paper specializes some web page segment rules, all that can help you understand the implement of this system.

Then, the thesis analyzes the structure of HuiCong Search Engine: WebServer and SO(Shared Object), Cache, newest database, database and web purify system. and the relation between these systems. The analysis of searching process: the users type the string into CGI from web server. CGI deals with these strings, put them to search engine system, then put all interrelated webs into PageClean system, the result can display in the browser. PageClean is the key part of this thesis. We discuss the arithmetic of this system and rules of implement.

Finally, the thesis discusses the test method of PafeClean, gets conclusion: PageClean system can reach the expected target.

Key words: Web Clean, Web Noise, HTML, Web Structure, WWW

哈尔滨工程大学
学位论文原创性声明

本人郑重声明：本论文的所有工作，是在导师的指导下，由作者本人独立完成的。有关观点、方法、数据和文献的引用已在文中指出，并与参考文献相对应。除文中已注明引用的内容外，本论文不包含任何其他个人或集体已经公开发表的作品成果。对本论文的研究做出重要贡献的个人和集体，均已在文中以明确方式标明。本人完全意识到本声明的法律结果由本人承担。

作者（签字）：徐丹

日期：　　年　　月　　日

第1章 绪论

1.1 课题背景和意义

本课题来源于北京慧聪网行业搜索引擎项目。慧聪网行业搜索引擎是国内最大的行业搜索引擎，每天有数百万的数据流量。其搜索引擎内核采用的是中搜企业级检索内核。慧聪行业搜索引擎采使用的历史库和最新库都是从WEB上抓取的网页。但是，网页上存在大量用户并不关心的信息，如导航条、广告信息、版权信息以及调查问卷等内容，这些信息称之为“网页噪音”。网页噪音导致主题漂移(topic drift)，使同一网页存在多个主题的情况。以整个网页为粒度的信息搜索结果不够准确，必须深入到网页内部，找出网页主题，才能提高信息检索的准确性。搜索引擎对整个页面内容建立索引，因此引入了无关信息。快速准确的识别并清除网页内的噪音内容（我们称之为网页净化）是提高搜索引擎处理结果准确性的一项关键技术。首先，网页净化后，没有了噪音内容的干扰，搜索引擎可以以网页的主题内容为处理对象，从而提高处理结果的准确性。其次，网页净化可以显著简化网页内标签结构的复杂性并减小网页的大小，从而节省后续处理过程的时间和空间开销。因此，网页净化已成为搜索引擎系统预处理环节中一个必不可少的工作[1]。

1.2 课题研究的内容

在Web信息检索领域，通常用两个指标评价一个Web检索系统，检索结果的相关性和检索的速度。

根据噪音内容的粒度大小，Web上的噪音内容可以被分为两类。

全局噪音：全局噪音是指Web上具有较大粒度的噪音内容，它通常包含镜像网站以及近似网页[2]。全局噪音内容不仅影响了Web上信息检索系统（比如搜索引擎）的网页搜集、索引和检索结果排序的质量，也使得Web信息存储系统浪费大量的磁盘空间去保存重复的网页。

局部噪音：局部噪音是指Web网页内与网页主题内容无关的噪音内容，比如：广告、导航条以及版权声明等内容。局部噪音使得应用程序很难确切得

到网页的主题内容，因此它严重的影响了基于网页内容的应用程序。同时，局部噪音很多情况下是伴随着超链出现的，因此，局部噪音也对基于网页间链接关系的应用程序造成影响

在一个页面中常见噪音信息包括这样几类：

1)导航类：为了维持网页间的链接关系，方便浏览者对网站进行浏览而设置的链接。

2)修饰类：为了美化页面而采用的背景，修饰图片，动画等。如站点标志图片，广告条。

3)交互类：为了收集用户提交信息或提供站内搜索服务的表单等。如在线的问卷调查表。

4)其它类：网页中声明的版权信息，创建时间，作者等描述性信息。

如果不去除网页中的噪音内容，那么索引子系统必然对噪音内容也建立索引，从而导致仅仅因为查询词在某张网页的噪音内容中出现，而把该网页作为结果返回，而网页的主题内容可能和这个查询词完全无关。可以看出，噪音内容不仅使索引的规模变大（从而会影响效率），而且还导致了检索准确性的下降。针对这个问题，文献[3]中提出了一个去除网页中噪音内容的方法，该方法首先依据<table>标签构造网页的标签树，进而依据<table>标签将一张网页规划为相互嵌套的内容块；而后，对于使用同一个模板作出的网页集，找出在该网页集中多次出现的内容，作为冗余内容，而在该网页集中共同出现较少的内容块就是有效信息块。实验证明该方法是有效的，但该方法必须局限在基于同一个模板的网页集，而Web上的网页模板不计其数，因此该方法显然不够通用。实际上，任意一张网页，人是比较容易区别其中的噪音内容和主题内容的。这说明我们有可能追求自动识别一张网页中的主题内容和噪音内容而不需要依赖于一个网页集合，这样就可以使去除网页噪音内容的方法更加通用和独立。

在主题搜索领域，大量的广告、导航条等噪音内容会导致主题漂移(topic drift)。这说明传统的主题搜索算法中以网页为粒度构造的网络图（Web Graph）不够准确，必须深入到网页内部将处理单元的粒度缩小，才能提高内容分析的准确性。文献[4]中提出一套方法，首先将网页表示为一棵DOM树结构并找到与主题一致性较高的子树，然后对这些子树作特别的处理，从而来

提高主题提炼的效果。

在网页信息提取领域，自动识别模式的方法必须要从整个网页中提取模式，而不是只针对主题内容进行提取[5]。因此，在净化后的网页上作信息提取不仅可以排除噪音信息对信息提取的干扰，提高信息提取的准确性，而且可以使得网页中的结构简单化，提高信息提取的效率。

从上述分析我们看到，噪音内容对基于网页的研究工作的影响是普遍而严重的，虽然各个领域采用的方法各不相同，但处理的目的都是为了排除网页中噪音内容的干扰，得到真正的主题内容。

参考上述文献中提出的启发式规则，并结合我们自己对HTML网页性质的统计和观察，本文提出了一套更丰富的启发式规则。在这套启发式规则的基础上，借助信息检索领域的方法，结合HTML网页的特点，提出了一种网页净化的方法和相关算法。该方法和算法与前述相关工作相比更为通用，不需要依赖网页模板等附加信息也不需要对待处理网页增加限制条件，比如：属于同一个模板。

本文的方法已被应用到慧聪网行业搜索引擎系统中。在行业搜索引擎，使用净化后的网页进行索引，查询，分类效果得到普遍的提高。

1.3 国内外研究现状

在网页噪音去除的工作中，可以看到两类情形：

（1）基于一个或多个网站中的页面集进行页面的模板检测，把为了生成页面而在网站中使用的模板作为噪音由页面中去除[10]。

（2）基于单一页面的处理，根据所处理页面的 DOM 结构，可视信息等应用一些启发性规则对页面内的噪音去除。

在第一类情形中：文献[6]中提出信息块（CONTENT BLOCK）概念，对新闻网站中的页面进行处理。以页面中 TABLE 标记作为处理元素，将页面分割成块，然后由页面集计算出各块的信息熵值，当熵值小于阈值时，认为此块为信息块。此种方法将页面看作可由 TABLE 分割的集合，并假设已知块在页面集中分布的先验知识，这种假设对于其它类型页面很难成立。而 Liu 等[7]文中，根据页面的 DOM 结构，构造 STYLE TREE，进行同一网站内页面模板的检测，以排除各页面内的噪音，提高了 WEB 挖掘的结果，但对从任意网站下

载的页面集或第二类情形无效。第二类情形中：文献[8]提出了根据布局信息对页面分块的算法，以消除噪音对分类的影响，但其在分块过程中采用的页面布局算法对于 HTML 规范中的框架，图层及 CSS 不支持，因此页面布局算法并不总能反映实际的页面布局。同时分块中主要依赖 TABLE 标记作为分块的主要参考标记，因此对于没有用 TABLE 做布局的页面不适应。此外，文献[9]利用页面的 DOM 结构，依据一些规则，对页面内一些元素进行了裁减，但对于链接丰富的 HUB 页面，如 MSN，造成页面中 HUB 链接被删除。文献[10]提出了 VIPS（基于视觉的 Web 页面分页算法）算法，将页面转化成内容结构。VIPS 基于页面的可视化信息来进行页面划分，并定义其内聚度。算法通过可视块抽取，分隔条检测及构造内容结构三步迭代进行，以各块的内聚度值与预定义的终止条件比较作为迭代的终止条件。此算法对页面进行细致地划分，但并未对噪音数据进行进一步的处理。

1.4 课题实现环境

本课题是利用北京慧聪网行业搜索引擎所提供的网页库进行实验的。该网页净化系统用 C/C++语言在 Windows 操作系统、FreeBSD Unix 操作系统系统，开发平台上开发的，使用的工具是 Visual C++ 6.0，GCC。

1.4.1 硬件环境

● UNIX 平台(双机备份)

Server：IBM R6000

MEMORY： 8 GB

DISK： 80 GB

Client： WINDOWS 2000 professional

MEMORY： 1 GB

DISK： 10 GB

1.4.2 软件环境

系统采用 C/C++实现，支持在多种操作系统环境下运行。

1） 支持的操作系统

Windows 2000 Professional、Windows2000 Server

UNIX：FreeBSD

2) 支持的Glib软件包

Glib 2.2.1以上

3) 支持的数据库系统

ORACLE 8i(8.17版本以上)

4) 支持的浏览器

Netscape：适用于所有平台，包括Windows和UNIX

IE 6.0以上：适用于Windows平台

5) 支持的WebServer

Apache 1.3以上，Tomcat3.3以上

1.5 论文组织

本论文对网页净化系统的设计和实现工作加以总结，论述了开发该系统所使用的关键技术；该系统在信息网行业搜索引擎架构中的位置，它与各个子系统之间的关系；并对其中的算法进行详细的描述，满足系统需求。

本论文组织结构安排如下：

第1章为绪论部分，介绍了本课题的研发背景及研究意义，明确了本课题研究的内容，给出了课题的软硬件实现环境，同时，介绍了本论文的组织结构，以给人清晰的条理。

第2章详细介绍了在网页净化系统实现中所用到的关键技术，主要包括网页结构：网页标签树表示，文档对象模型DOM树定义及其特点，网页如何用网页标签树表示，如何用DOM树表示；网页结构分块技术，根据HTML的一些重要标签对网页进行分块。对这些技术的了解，有助于对本研究课题功能实现的理解。

第3章介绍了信息网行业搜索引擎的整体架构和各个子系统；网页净化系统在行业搜索引擎架构中的位置，它与各个子系统之间的关系；搜索引擎的检索过程及其Web端的CGI程序设计。

第4章给出了信息网网页净化系统的详细设计方案及其实现，其中详细介绍了网页净化系统PageClean系统的算法实现依据的规则，算法的思想，算法的流程以及在联机的情况下对网页净化的实现。

第 5 章对该网页净化系统进行了详细的测试和分析。其中介绍了测试的数据集，网络环境，软件环境，硬件环境。详细介绍了网页净化系统 PageClean 的净化速度和净化效果，并对结果进行了分析，提出了改进的优化策略。

其中第 3 章、第 4 章和第 5 章分别是本系统的需求、设计和具体实现及测试部分，为本文的主题。

在结论部分对本文的工作做了一个总结。

第2章 网页结构分析及内容分块技术

这一章将介绍在实现本课题研究的系统所需要的关键技术，主要包括：网页结构、网页标签树、文档对象模型DOM、网页结构分块技术。

2.1 网页结构模型

为了方便网页净化系统的预处理，我们用一个统一的结构化的模型表示预处理的结果。该网页净化系统结构化模型包括：网页标识、网页类型、内容类别、标题、关键词、摘要、正文、相关链接等要素。其中正文和相关链接要素属于网页的内容数据，而其他几项则属于网页的元数据。下面将对模型中的各个要素作详细描述。

网页标识是对Web上网页的唯一性标识，在网页净化系统结构化模型中使用网页的URL作为网页标识。

网页类型是根据网页内容的表现形式进行划分的，在本文中将网页分为三类：有主题网页、目录网页、图片网页。

有主题网页：网页中通过文字描述了一件或多件事物，是有一定主题的。一张具体的新闻网页就是典型的有主题网页。

目录网页：专门用来提供网页导向的网页，因而是超链接聚集的网页。一般来说，新闻网站的首页就是典型的目录网页。

图片网页：网页的内容是通过图片的形式体现的，其中文字很少，仅仅是对图片的一个说明。计算机学院网站对导师的介绍网页就是典型的图片网页。

将网页分为上述三个类型是因为三类网页在用途和处理方法上存在较大的差别。其中目录网页与其它两类网页的区别在于网页在Web上发挥的作用不同，目录网页通常不会具体的讲述一件事物，而是提供关于相关信息的链接集。而图片网页与其它两类网页的区别在于处理的方法不同，由于图片网页的内容是通过图片表达的而不是通过文字，因而，传统信息处理领域的方法对图片网页是不够有效的。三类网页间的区别导致很多应用领域都会对它们

作适当的区别。

内容类别是对网页的内容进行分类的结果，它是计算机获取网页语义信息的一个直接手段，在Web上的研究领域中有着广泛的应用。它是通过特定的分类器对网页内容分类得到的，依赖于一定的分类体系。Dublin Core中推荐用内容类别作为其中subject元素的值[11]。

标题、关键词和摘要是概括描述Web文档内容的重要的元数据，对于Web信息检等领域的工作有非常重要的作用；

正文是原始网页中真正描述主题的部分，可以看作是净化后的网页，因此，在某些具体应用中用正文代替原始网页更为合理。

相关链接是指在本网页中指向与正文内容相关的网页的链接，而非广告等噪音链接。可以看出，将正文和相关超链重新组合可以得到另外一个净化尺度的净化后的网页。

2.2 网页表示

网页的表示是网页内容分析的基础，在网页内容分析过程中通常需要对网页内容进行抽象表示。抽象表示是以网页制作规范（HTML规范）为依据和出发点，构造出能体现网页内容结构和内容重要性等信息的表示模型，其目的是充分利用网页制作规范，利用网页中的一些重要的标签，挖掘出网页中隐含的信息，最常用的方法是构造网页的标签树。

网页标签树表示：

今天，Web上大多数的文本信息都是以HTML网页的形式存在的。HTML是一个标识语言（Markup Language），网页中的内容都存在于标签之中。为了更清楚的描述网页内容的组织结构，通常将网页中的标签按照出现顺序，依次整理出来并用适当的结构记录下来。由于标签之间的嵌套关系，标签的整理结果自然是一棵树状结构。我们把整理一篇网页中的标签得到的树状结构称为该网页的标签树。为了获取所需的信息，Web上很多领域需要对网页内容进行分析，而随着研究和应用的深入，以整张网页为单位的分析粒度已经不能满足需要，这要求我们必须深入到网页内部，将分析对象的粒度缩小，以提高分析的准确性[12]。由于网页中的标签结构是对页面布局的描述，因而依据标签树对网页进行细化是合理的。因此，标签树在网页内容分析工作中经常

会用到。

目前，有很多构造标签树的工具，他们各有特点。下面我们主要介绍W3C Document Object Model (DOM [DOM])和HTML Tidy[13]。 DOM可以为每篇HTML构造一个树状结构，其中网页内的标签作为树的内部节点，而文字和图像作为树的叶子节点。 HTML Tidy也是一个被广泛使用的标签分析工具，它的特点是有很强的容错能力，可以发现网页中的标签错误（例如：结束标签丢失、结束标 签匹配错误等等）并进行较为合理的修正。 本文提出的标签树构造方法则是面向内容分析。该方法首先从内容分析的角度将标签分类，并以一种适合内容分析工作的方式组织标签信息。另外，在标签树中包含一定的统计信息，因此通过标签树中的信息，可以对网页有一个大致的了解。

适合内容分析的标签树与通用标签树相比有这样几个特点：

1) 在标签树的框架上，更强调对网页内容组织结构的刻划。换言之，内容分析中强调内容块的概念，而不是任意的标签都构成标签树中的一个结点。

2) 在标签树中信息的组织上，对内容分析经常用到的几类信息按内容块组织，并且提供可以快速且方便操作的存储方式。

3) 需要有适当的描述性信息。在做内容分析的时候，除了用标签树来刻划网页的结构，我们通常还希望得到这样的一些信息：标签树的规模（内容块的个数）、每个内容块的信息量（可以通过内容块中的字数体现）、哪些内容块中有超链、哪些内容块中有描述性标签、及相应的数量； 而这些信息在现有工具构造的标签树中是很难直接得到的。鉴于此，本文提出一套适合内容分析的标签树组织方式及其构造方法。

2.3 与网页结构对应的文档对象模型

随着 Internet 的发展，Web 正在不断演变成下一代应用平台，为了获得真正的交互式体验，在客户机上动态处理内容是最重要的。W3C 的文档对象模型(DOM)是迈向这一目标的重要一步[14]。

2.3.1 编写网页常用语言

1)可标记超文本语言 HTML

HTML ，HyperText Markup Language，中文翻译为“可标记超文本语言”。官方的定义描述为“为了发布全球化的信息，人们需要一种通用的理解性语

言，一种所有计算机本质上可以理解的发布‘母语’，WWW 使用 HTML 作为这种发布语言”[15]。

正是由于有了 HTML 这种通用语言，人们才可以在因特网上发布多种多样的资源，有了 HTML 语言意味着我们可以：

发布带有标题 title、文本 text、表格 table、列表 list、照片等资源的网络文档。

● 通过点击超文本链接来浏览网络文档；

设计通过远程服务管理事务，比方说搜索信息、房间预定、产品订货等等；

● 设计通过远程服务管理事务，比如说搜索信息，房间预定，产品订货等等；

● 把分析表格、视频片断、声音片断和其它应用程序都直接包含在它们所在的文档中。

HTML 文档具有结构化格式，这种格式通过 I-ITML 的元素(Element)来实现，这里列出一些实现的系统中用到的 HTML 元素以及它们的含义：

● A 和 LINK：到另一个文档或资源的链接；

● LINK 和 SCRIPT：链接到外部的样式(style)或脚本(script)；

● IMG，OBJECT，APPLET，INPUT 包含一个图片、对象或页面中的 applet；

● MAP 和 AREA 创建一个图片映象；

● FORM 提交表单；

● FRAME 和 IFRAME 创建一个框架文档；

● Q，BLOCKQUOTE，INS 和 DEL 指向外部的引用；

2)可扩展标记语言 XML

XMLTM (Extensible Markup Language 一可扩展标记语言)的产生是为了恢复 (SGML C Standard Generalized Markup Language 一标准通用标记语言)的强大功能和灵活性，而又不带有 SGML 的复杂性[16]。虽然只是 SGML 的一个“受限制”形式，XML 却保留了 SGML 大部分的功能和丰富内容，而且仍然具有 SGML 常用的那些特性。

简单地说，XML 就是一种文本。一个 XML 文件就是以特定格式安排的文本文件。文件可以在任何计算机系统上使用文本编辑程序来建立。由于它只

是一个文本，因此能够方便地在各个计算机系统间甚至在各个计算平台间传输。例如，在苹果公司 Macintosh 机上建立的 XML 文件可发送到 Window。或 Linux 的 PC，或一个主机或一个 Unix 服务器。这种可传输性使人们很容易理解为什么 IT 部门和软件销售商如此热烈的欢迎 XML。

XML 使用标记(包含在尖括号中的字，例如“<root>”)来识别信息元素。

初看起来，尖括号使 XML 文档很像 HTML (Hyper Text Markup Language，超文本标记语言)文档。但是 HTML 文档和 XML 文档大相径庭，并且应用目的完全不同。

HTML 能够告诉 Web 浏览器怎样绘制和显示一个文档。XML 则说明包含在一个文档中的数据。

3)可扩展超文本标记语言 XHTML

可扩展超文本标记语言(Extensible Hypertext Markup Language，简称 XHTML)是 HTML 和 XML 的混合物，它是为网络设备显示(包括 Web 浏览器、PDA 设备和移动电话）而特别设计的。2002 年 1 月 26 日标志了 XHTML 1.0 作为 Web 标记的正式 W3C 推荐的第二个生日[17]。

W3C 主管 Tim Berners-Lee 这样评价 XHTML：“XHTML 1.0 连接了现在的 Web 和将来的 Web……它为页面和网站作者提供了进入结构化数据 XML 世界的桥梁，同时仍然能够保持与支持 HTML 4 的用户代理的可操作性。”

W3C 声称，XHTML 的主要优点是可扩展性和可移植性：

1)可扩展性：XML 文档要求格式良好(元素嵌套正确)。使用 HTML，添加新的元素组需要更改整个 DTD。在基于 XML 的 DTD 中，新的元素组只需要内部一致并且格式良好，就可以添加到现有的 DTD 中。这极大地简化了新元素集合的开发和集成。

2)可移植性：越来越频繁地使用非台式设备来访问因特网文档。在大多数情况下，这些设备不具备台式计算机的计算能力，并且不像标准桌面浏览器那样可适用于格式差的 HTML。实际上，如果这些非桌面浏览器没有接收到格式良好的标记(HTML 或 XHTML)，它们可能根本无法显示文档。

一个 XHTML 的经典例子[18]如代码 2.1

```
<?xmlversion=" 1.0"  encoding="UTF-8" >
```

```
<!DOCTYPE html PUBLIC " -//W3C//DTD XHTML 1.0
Strict//EN" DTD/xhtml1-strict.dtd">
<html xmlns="http: //www.w3.org/1999/xhtml" xml: fang=" en"
fang="en" >
<head>
<title>Virtual Library</title>
</head>
<body>
<p>Moved to
<a href=" http: //vlib.org/" >vlib.org</a>
</p>
</body>
</html>
```

在这个经典的例子中：

1)由于 XHTML 是以 XML 文档表示的 HTML，所以它必须在文档的顶部包括初始 XML 声明<?xml version=" 1.0"?>

2)XHTML 文档必须由三组标准规则的其中一组来标识。这些规则存储在一个称为“文档类型声明(Document Type Declaration (DTD))”的单独文档中，并且使用这些规则验证 XHTML 文档结构的准确性。

3)XHTML 文档必须包括完整的头部区域。这个区域包含开始<head>标记和标题标记(<title></title>)，然后以结尾</head>标记结束。

4)XHTML 文档必须包含开始和结尾<body></body>标记。在这些标记中，您可以放置传统的 HTML 编码标记。要与 XHTML 符合，这些标记的编码必须是格式良好的。

5)最后，使用结尾</html>标记结束 XHTML 文档。

2.3.2 DOM 的定义及其特点

按照 W3C 的定义,DOM 是一个允许程序或者脚本能够动态地存取和更新 HTML/XML 文件内容、结构以及风格的接口和平台。DOM 目前主要由两部分组

成：DOM Core 和 DOM extension。DOM Core 主要定义了处理 XML 文件所需的功能；DOM HTML 定义了处理 HTML 文件所需的功能。

DOM 是语言独立的。DOM 的接口都是符合工业标准的界面定义语言 IDL(Interface Definition Language)描述的，不限制用何种语言具体实现这些接口。DOM 的核心是将面向对象(Object-Oriented)的概念引入 HTML/XML 文件的处理中。在 DOM 以前，无论是 HTML 还是 XML，均被看作是包含各种组件的数据集合，以面向数据的方式管理文件。引入对象后，在 DOM 看来，HTML/XML 的组件不只包含数据本身，每一个 HTML/XML 中的元素(Element)还包含有方法(Method)和属性(Attribute)。DOM 使用这些方法和属性的 API，通过方法和属性来存取和管理组件[19]。

文档对象模型(DOM)是一种用于 HTML 和 XML 文档的应用程序编程接口(API)。使用文档对象模型，程序员可以构造文档，增加、修改、或删除元素和内容，HTML 中的任何内容都可以使用文档对象模型进行存取、修改、删除或增加。DOM 是由一组对象和存取、处理文档对象的接口组成。下面介绍常用的几种对象，它们包括文档、节点、元素、文本节点、属性、N 维树。

1)文档(Document)

DOM 的文档是由分层的节点对象构成，这些节点对象构成一个 HTML 页面：文档是一个节点，该节点只有一个元素，这个元素就是它自己。文档接口表示整个 HTML 文档，从概念上讲，它是文档树的根，提供对文档数据的存取。

2)节点(Node)

节点是一般类型，它涉及一个文档中存在的所有对象。

3)元素(Element)

在细读一个文档时，最常碰到的东西就是元素，元素是除文本之外的几乎每一个对象。元素是从节点类型推导出来的。元素包含属性，而且可以是另一个元素的父类型。

4)文本节点(Text Node)

文本节点处理文档中的文本

5)属性(Attribute)

属性是元素的基本属性，因此它们不是元素的子节点。即使它们是从一

般节点类型推导出来，它们的行为也与其它节点的行为不同。例如，对属性调用 parentNode， previousSibling 和 nextSibling，它们将返回 Null。也就是说，它们不是文档树的一部分。

6)N 维树(N-ary Tree)

N 维树以像树一样的结构表示数据。N 维树具有一个根，这棵树有子节点。如果文档是根，则它的子节点是由它下一层的元素和文本节点构成。

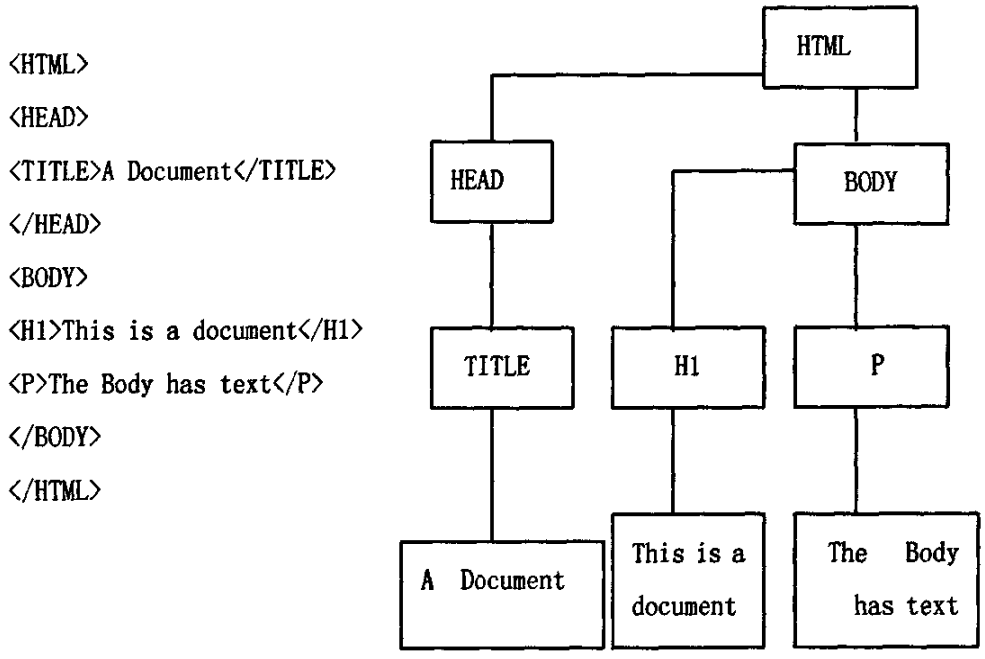
2.3.3 DOM 和 HTML 树型逻辑结构

一般来说，HTML 文件由标题(Title)、头(Head)，段落(Paragraph)，超链(Hyperlink)以及其它各种组件组成，并且组件在文件中的顺序与显示顺序相同。DOM 通过对 HTML 文件的解析，生成一个文件的树型内部结构，称为文件的树型逻辑结构或逻辑结构。树型结构可以准确地描述元素的相对位置关系，很适合描述 Web 的半结构化数据。从 HTML 文档到标记树的转化可以通过 HTML 的语法分析器来完成。文件的树型逻辑结构与 Web 文档一一对应，可以相互转化，文件的树型逻辑结构是便于计算机处理，用来表示 HTML/XML 文档的一种数据结构。DOM 在进行文件解析时，将 HTML 文件看成一棵树。<HTML>作为树的根，而 HTML 文件的其它组件被看作树中的节点(Node)；节点可以作为父节点包含节点，也可以作为其它节点的子节点；同一层的节点成为兄弟节点。

DOM 定义了 API 允许其它程序浏览树型逻辑结构，并且提供存取、添加、修改和删除节点的功能。图 2.1 是一个简单的例子说明了 DOM 是如何建立文件的逻辑结构，其中(a)是一个简单的 HTML 文档，(b)是它的树型逻辑结构。

从下面的例子可以看出，原来的 HTML 文件被转化为一个树型结构。其中<HTML>是树型结构的根节点；<TITLE>，<H1>，<P>都是<HTML>的子节点，<HTML>是它们的父节点：<TITLE>，<H1>，<P>互为兄弟节点。可以看出使用树结构可以达到以下的好处。

节点操作，添加、删除节点。在特定的节点中增加新的属性或节点，以及修改节点的内容。在网页视图重构和转化中可以通过这样一些操作，不改变网页内容，而改变内容的表现形式和视图的大小。在标记树的结构上根据不同的需要导出或生成一种新的代表 HTML 文档某方面特征的新的结构。



HTML 文档(a)　　树型逻辑结构(b)

图 2.1 HTML 文档及与其对应的树型逻辑结构

2.4 中文网页分块模型

实际当中，人们在设计网页时，常常将网页分成多个区域，把不同主题、不同作用的文字安排在不同的区域里，类似于报纸、书刊、杂志中的排版。连贯的文字通常放在一起组成段落，并采用一致的版式表达，而不相关联的内容则用不同的版式加以区分[20]。我们是自顶向下来得到我们想要得到的内容块的。也就是说，我们从文档树的根节点开始向下进行块的提取。我们首先假定我们的文档是只有一个大块的，然后我们通过分析来确定是不是这样的，如果分析的结果是这个大块的主题并不是一致的，我们就会将它按照子节点分成几个小的块，然后对于分成的小块递归的使用这个方法。

在看到微软亚洲研究院的工作之后，我们想到可以使用视觉的因素来为网页进行分块。之所以使用视觉因素而不是文本特征的方法，是因为我们考虑到除了首页之外的网页多数都不是足够大，无法有效的利用文本特征来区分它们，我们还借鉴了瞿有利，于浩，徐国伟的工作[21]，于满泉，陈铁睿，许洪波的工作[22]和邹涛，王继成的工作[23]。

我们发现，通常一个网页含有很多的表格，而这些表格就自然的将网页分成了几个矩形的区域。而且，这些矩形区域的视觉特征通常是非常一致的。同时这些一致的视觉特征的块的主题也是一致的。这都是因为网页的制作者希望读者可以容易的分辨网页内容才会将相似的内容以相似的视觉特征加以表现。所以我们就可以利用网页上面的视觉和结构的要素来分割一个网页。当然这其中我们也需要借助 DOM 结构，所以我们的工作是加入了视觉要素的DOM内容抽取。

下面我们来讨论比较重要的几种用来分块的标签：

1)Table，Table标签是表格标签。Table本身的内容可以说明很多问题，如果一个Table里面又嵌套了其他的Table，几乎可以肯定的是，它们的主题一定是不相关的。就像是我们经常看到的网页的例子，左面是一个导航栏，而右面是一个文本。相反如果一个Table里面没有嵌套其他的Table，那么它们很有可能是同一主题。这时通常需要考虑Table的视觉特征，如果这个Table的内容是单一的，那么可以肯定的是它的视觉观感也是统一的，比如它的字体，颜色，还有字体的大小。相反的如果一个网页的Table的视觉特征是不同的，比如说，左侧的表格是黄颜色，而右侧的表格是绿颜色，那么它们的内容就有可能是不同的了。不过在这里我们也要看到另外一种情况，通常为了表示内容的主题，如果一个Table只有两个格的话，它很有可能会在上面的一个格里面加入一个小的标题，在下面的一个格子里面加入的是一个文本，如果是这个样子的话，它们的观感虽然不是很统一但是它们的内容确实是非常相关的。对于 table 的处理，我们通常会首先考虑Table里面的是不是多个Table，如果是的话，我们就认为它们是不相关的，如果都是文本的话，我们就要分析一下，以前面为例子，如果两个格里面的内容是明显的不对称。也就是说一个表格里面的内容明显很多，而另外一个表格的内容是明显很少的，我们就假设它们是相关的。进一步的处理，如果Table里面的内容是多个表格，而且它们刚好也是一个表格的内容很多，一个表格的内容很少，我们就将它们两两的分成一组。如果表格的内容是对称的，也就是说它们的内容都很多，而且它们的观感不尽相同，比如说颜色。那我们就将他们分成多个块。

2)dl与ol，它们都是列表的标签。列表可以看作是一种简化的Table它们通常都是比较的简单的。而且虽然它们的内部与会有一点分别，比如说也会

分成几个块的感觉，如代码2.1和2.2。

代码2.1 dl与ol标签html代码

```
<html>
<head>
<title>一个普通列表</title>
</head>
<body text="blue">
<dl>
<dt>中国城市</dt>
<dd>北京 </dd>
<dd>上海 </dd>
<dd>广州 </dd>
<dt>美国城市</dt>
<dd>华盛顿 </dd>
<dd>芝加哥 </dd>
<dd>纽约 </dd>
</dl>
</body>
</html>
```

代码2.2 dl与ol显示页面

中国城市

北京

上海

广州

美国城市

华盛顿

芝加哥

纽约

但是，我们可以很容易的发现它们的内容多半都是相关的，没有必要对于它们进行更进一步的划分。同样的，加入了数字标号的ol标签也不需要进

行更多的划分。

3）hr，hr标签是下划线标签。这个标签好像很多的网页在主页上面都没有使用，主要是因为主页上面通常都是大量的链接，但是如果我们到了具体的内容页的时候，这个标签的作用就是很大的了。通常如果一个网页制作者在制作网页的时候，也会考虑到标题内容的细分。比如说，介绍了现在最流行的几种打领带的方法的时候，通常会详细的分成，亚波特王子节，温沙节，简式节，浪漫节，等等。这么多的标题，如果只是通过文本来区分的时候，那么就需要让读者比较容易的找到自己想要看的样式，所以，通常会在文本的前面加一个标题，或者加一个下划线，几乎可以肯定的是，如果一个文本被加了一个下划线，通常都是为了和其他的文本区分开，也就是说它们的主题是有区别的，对于这样的标签，我们在处理的时候，认为完全可以通过它们来区分两个主题。与之相对的是 p 标签，也就是段落标签，最初我们曾经考虑使用段落标签，但是我们却发现段落之间的联系是非常紧密的，所以我们忽略了它。

4）Frame，Frame标签是帧标签，可以将一个网页链接过来，在处理帧的时候，我们可以采用刚才的方法，来分析一个帧的内部的情况。但是对于帧和帧之间的联系，我们所需要考虑的就比较不同于Table了，毕竟之所以会将网页分成为两个帧，那通常都是认为它们之间的关系不是很紧密的，因为对于关系紧密的内容，网页的制作者会尽量将他们放在一个网页内，这样便于读者的阅读，所以就算是两个帧之间的结构和布局上面有着类似的地方，我们也打算将它们当作是两个网页内容来处理。这一点表面看起来和我们原来的认为只要是两个内容块的视觉观感一致就认为是相同主题的原则不一致，我们会将指向新Frame的文本加入对新Frame的分析。实际上，我们需要考虑的是，作者之所以将网页的内容做成是观感一致的是为了读者的阅读，也就是希望读者在阅读这些内容的时候，可以把它们和网页的其他部分轻易的区分开，但是对于不同的网页而言，就很明显没有这个必要了。我们并没有讨论全部HTML标签的必要，因为对于我们的分块任务来说，很多标签是可以忽略不计的。而且，我们通过讨论也明白只是考虑 DOM 树当前Node节点的属性是远远不够的，所以我们对于一个节点是否应该被分割，是基于两个方面的考虑：

（1）节点本身的属性。例如，字体，颜色等等。

（2）节点的子节点集合的属性。例如，字体，颜色，大小。

文本分块的大小。就像是我们前面提到的那样，现在的文本分类技术是首先对于文本进行特征向量的提取，如果文本的内容比较少的话，在进行分类的时候，很有可能会分类不准确。所以对于网页的内容块的划分不应该太小。从另外一个角度来看的话，将网页的内容块分割的过小也是没有道理的，毫无疑问，如果我们将内容块选取的过小的话，那么就很有可能会出现，我们的网页之中相关的内容却被分割成为了不同的块。这非常容易理解，就好像是一篇文章，本来它是有着一个唯一的主题的，就像文章题目所描述的一样，但是如果我们以段为单位进行文本特征向量的提取，那么每一段都会因为夹角过大而被认为是不相关的，结果就是我们得到了一个有着很多很多主题的文章，这很明显是错误的。

在我们制定分块原则之前，我们有必要进行一个定义工作来方便我们的内容描述。

有效节点：一个节点可以通过浏览器观察的到，也就是说，这个节点不是一个空的节点，也不是那些注释类的 comment 节点，定义这个节点是因为如果一个节点无法通过浏览器观察的话，它必然不会传递网页制作者的任何信息。

非结构的 Element节点：就是说这些DOM树的Element节点只是对文本起修饰的作用，比如，加粗，斜体，字体，颜色等等不会使文本的结构发生诸如换行，列表之类变化的Element节点。

结构Element节点：除了非结构 Element节点之外的Element节点。

纯文本块：完全是由文本组成就是说是 DOM 树的 Text 节点。

准文本块：它的子节点是由文本纯文本块或者是准文本块或者是纯文本块和准文本块构成的；它的子节点之中包含有非结构的Element节点，但是这些非结构的 Element节点的子节点是纯文本块。

基于上面的讨论和定义，我们已经可以对我们的分块提出一些分块的原则了。

1)如果一个节点的子节点是一些含有 hr 元素的节点那么就意味着它们的主题是不同的，我们应该将它们分割成为不同的内容块。

2)如果一个Table元素的子节点之中含有Table那么它们的主题是不同的，应该分割成为不同的内容块。

3)如果一个Table节点的子节点的颜色，文字大小，字体是不同的那么它们的主题是不同的，我们应该将他们分块。

4)如果一个节点的子节点都是文本节点或者是准文本节点，那么我们认为它们的主题是一致的，不应该分割。

5)如果一个节点的大小，也就是所含有的字符数已经低于我们事先设定好的数值，那么我们不再分割它们，之所以这么决定，完全是为了主题爬行的下一步处理在抽取特征向量的时候，不会因为它们太小而忽略它们。

6)如果一个节点不是文本节点，它的子节点里面也没有有效节点，那么这个节点会被删除。

7)如果一个节点的子节点里面含有有效节点，但是这个节点不是文本节点或者准文本节点的话，我们应该分割它。

8)如果一个节点的子节点是 DOM 树的话，比如通过Frame引出的，那么我们要分割它。

9)如果Table的子节点都是文本节点，并且文本是对称的，视觉特征不相同，那么应该分割它。

文档的最后输出依然可以是一个 DOM 树结构的文档，但是为了下面的处理的方便，我们将之输出为几个块的分别的文本文件，同时保留了上面的链接在另外一个同名加 a 的文本之中，例如一个块文本的名字是block1，那么它的链接文本的名字就是 block1a。

2.5 本章小结

本章详细阐述了网页结构和网页分块的理论。通过对网页结构的分析，得出网页可以用标签树表示和文档对象模型 DOM 树表示。文档对象模型 DOM 树把 HTML 文件转换成一棵 DOM 树。<HTML>作为树的根，而 HTML 文件的其它组件被看作树中的节点(Node)，节点可以作为父节点包含节点，也可以作为其它节点的子节点，同一层的节点成为兄弟节点。网页分块是网页净化的前提，本章提出了九条网页分块的规则。

第3章 信息网行业搜索引擎构架

3.1 系统概述

信息网行业搜索引擎包含一下几个主要部分：WebServer&SO，CACHE 管理系统，网页净化系统，最新库，历史库。如图 3.1

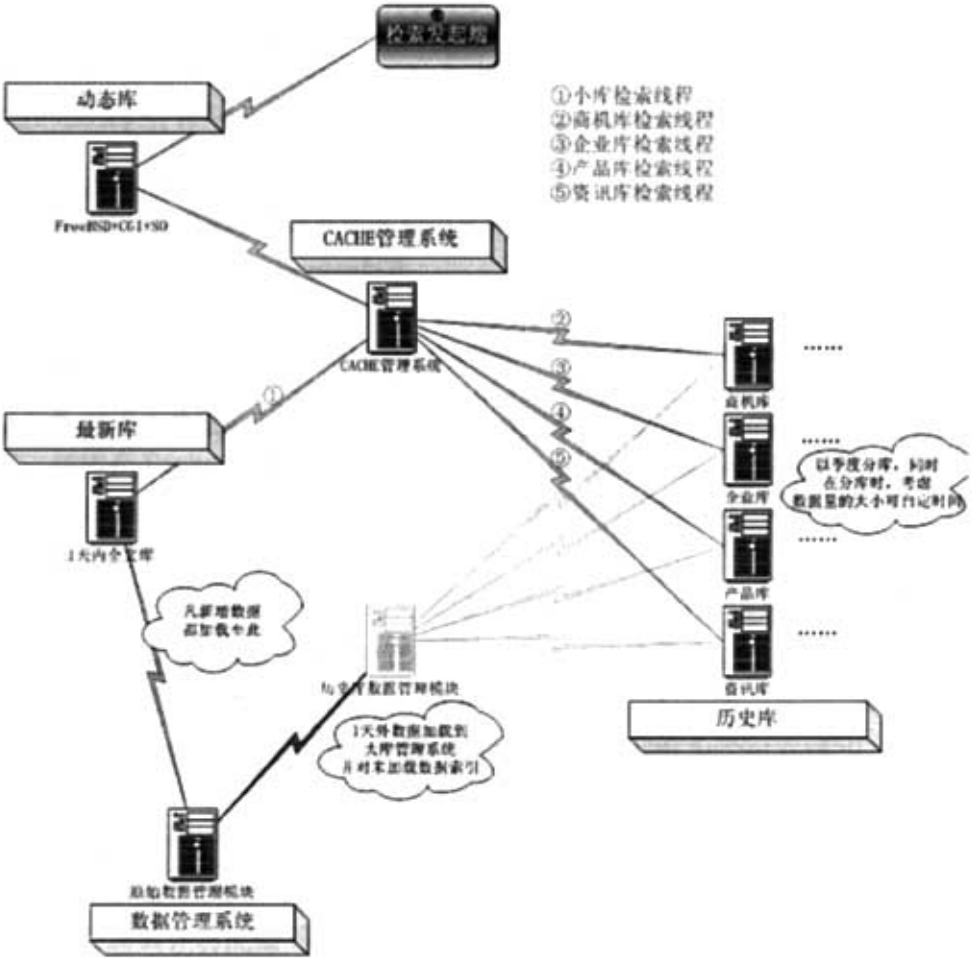


图 3.1 行业搜索引擎架构

3.1.1 业务范围及需求

行业搜索引擎是信息网 B2B(Bussiness To Bussiness)网络营销平台的一部分，主要满足 B2B 客户搜索自己想要的产品，信息网行业搜索引擎和信息网买卖通结合在一起，为客户提供优质的网络资讯信息。行业搜索引擎从买卖通数据库加载数据，并对其进行净化处理，使用户在搜索关键词的时候，展现给用户的是净化后的，噪音干扰小的，排序后的网页。

3.1.2 系统的组成

1)WebServer&SO

WebServer 采用 FreeBsd 服务器， WebServer 端程序采用 C/C++编写的 CGI 程序。该子系统负责接收检索请求及检索结果。网页净化系统与 WebServer&SO 相关联，WebServer&SO 端显示净化后的网页。

2) CACHE 管理系统

接收检索请求，查询最新库和历史库。对检索回来的结果进行排序，缩进，保存，返回给 WebServer&SO。

Cache 工作流程概述：

- 接收 WebServer&SO 检索请求
- 将请求发给最新库进行检索 接收最新库检索结果，结果已计算好排序值，同时在 Cache 中进行检索。如果找到，直接取结果。如果没找到，丢到 CacheList，由检索服务器进行检索等待检索结果，结果已计算好排序值。
- 等待所有检索结果返回
- 如果 Cache 未命中，由历史库进行的检索结果，合并历史库检索结果，进行缩进。将缩进后的结果存储或更新到 Cache，如果 Cache 命中，则该结果就是缩进后的结果。将此结果与最新库结果合并进行排序、缩进，根据请求得到相应页号，取相对应的内容，组织结果，返回给 WebServer&SO

3) 最新库

当天的数据，以及漏加和修改的数据（非当天），为保证快速增删改，最新库数据暂定为最新一天，在具体实施过程中可根据情况再定。

最新库具体功能如下：

- 检索

对从 CACHE 传入的检索串进行检索，得到 DOCID，根据排序算法进行排序。返回到 CACHE。

● 数据加载

对逻辑删除的数据进行物理删除，因物理删除后需要释放空间，所以每次加载前，先释放空间，再进行批量加载。

● 数据删除

直接对库进行逻辑删除。

● 取简介

根据 DOCID 取得该数据所对应的所有内容。

● 统计功能

库容量，检索次数等。

4)历史库

除最新库以外所有的数据，历史库按分类分成四个分类大库，每个分类大库按时间分为不同的小库。每个小库数据的时间范围可以自定，默认为三个月数据为一个小库。

历史库的具体功能如下：

● 检索

对从 CACHE 传入的检索串进行多库检索，得到 DOCID，根据排序算法进行排序。返回到 CACHE。

● 数据加载

对逻辑删除的数据进行物理删除，因物理删除后需要释放空间，所以每次加载前，先释放空间，再进行批量加载。

● 数据删除

直接对库进行逻辑删除。

● 取简介

根据 DOCID 取得该数据所对应的所有内容。

● 统计功能

库容量，检索次数等。

● 分库功能

系统默认情况下为三个月一个子库。也可根据实际数据情况进行自定义

子库。

● 数据管理系统

数据管理系统负责处理所有的数据加载准备工作。数据管理系统分为二个大模块，一是原始数据管理模块，二是历史库数据管理模块;原始数据管理模块负责所有的增删改数据，为最新库和历史库管理模块提供数据，历史库管理模块负责为历史库提供加载列表及对加载列表的同步。

3.2 信息网行业搜索引擎检索过程

整个检索过程分为初始化检索，打开库，检索，释放存放检索结果的缓冲区，关闭库，退出系统六个部分。如图 3.2。

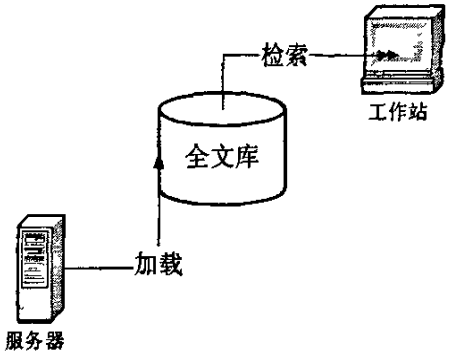


图 3.2 检索过程

3.2.1 各个子系统之间接口关系

1) WebServer&SO 只负责与 CACHE 接口

数据查询通过 WebServer&SO 传送给 CACHE 管理系统，由 CACHE 将正确的结果返回给 Web，完成此次查询。

2) CACHE 负责与最新库和大库接口

CACHE 在接收到检索请求后，对于不在 CACHE 中命中的查询，负责对大库和最新库进行检索，组合为最终结果，返回给 WebServer&SO 并对此结果保存。

3) 数据管理系统与历史库接口

数据管理系统与历史库的接口仅为删除数据。

4) 数据管理系统与最新库、历史库管理模块接口

对于新增、删除的数据对最新库和历史库管理模块进行加载和删除。历

史库管理模块与历史库接口，此模块提供历史库加载列表。

3.2.2 数据加载流程

一批数据在经过净化后，分为删除数据和加载数据。对于一条新增数据，直接增加到最新库里，并通知历史库管理模块提供服务；对于一条修改的数据，会生成一个删除文本和一个加载文本。系统判断信息本身日期，定位到所在库中，先进行删除，然后进行增加到最新库提供服务；同时，通知历史库数据管理模块定位到历史库的加载列表中；对于一条重发的信息处理同修改；对于一条删除的数据，系统需要知道此条删除数据存在于哪个系统中。如果是当天的数据删除，则从最新库和历史库数据管理模块中删除（称为同步）。如果数据不在最新库，那么需要判断是否在历史库数据管理模块中准备进行加载，如果存在则删除，那么历史库在加载时，不周对此条数据进行加载。否则从历史库中直接删除。

3.3 WebServer 端 CGI 程序

CGI（公用网关接口）是Web服务器和运行在服务器端的外部程序之间的一个接口规范，按照这个规范编写并运行的外部程序就叫做CGI程序，其目的是实现Web服务器与客户端浏览器之间的动态交互。CGI程序通过由Web服务器来调用，实现和浏览器的交互。

CGI 程序运行在 WebServer 端，他负责接收用户的查询串，然后把查询串进行合理的分词，传到搜索系统中进行查询，把搜索到关键词相关的网页通过网页净化系统净化，最后由 CGI 程序把净化后的网页显示在浏览器中。

3.3.1 CGI 环境变量

环境变量是文本串(名字/值对)，它们定义了程序的执行环境，可以被操作系统或其他程序设置，也可以被其他程序访问。环境变量是Web服务器传递相关信息给CGI程序的主要手段。Web服务器为CGI程序设置了一些环境变量，用来向CGI程序传递一些重要的参数。以下是CGI程序设计中常常要用到的一些环境变量：

REQUEST_METHOD：指的是当Web服务器传递数据给CGI程序时所采用的方法，分为GET和POST两种方法。

GET方法仅通过环境变量QUERY_STRING来传递数据给CGI程序，而POST方法通过标准输入传递数据给CGI程序,并使用环境变量CONTENT_LENGTH来指定传递的数据长度。因此POST方法可较方便地传递较多的数据给CGI程序。

QUERY_STRING：当使用GET方法时，Form中的数据放在QUERY_STRING中，传递给CGI程序。

CONTENT_LENGTH:当使用POST方法时指定通过标准输入传递给CGI程序的数据字节数。

CONTENT_TYPE:传递给CGI程序数据的MIME类型,通常为“application x-www-form-url，encoded”,它是从HTML Form中以POST方法传递数据给CGI程序的数据编码类型，称为URL编码类型。

REMOTE_ADDR，REMOTE_HOST：访问该CGI程序的Web浏览器的机器的IP地址及域名。

信息网搜索引擎环境变量的设置如代码3.1

```
char   *Buffer_Env=NULL;
set_env();
char *pRequestMethod;
if(argc<2)
 {
  pRequestMethod = getenv("REQUEST_METHOD");
  if (pRequestMethod==NULL)
{   printf("</head>");
    printf("<BODY>Hello");
    set_tail();
    return 0;
}
 int  i_content_length=0;
 if ( strcmp( pRequestMethod ,  "POST" ) == 0 )
{
  char  *p_connect_length;
  p_connect_length = (char *)getenv("CONTENT_LENGTH");
```

```
    if ( p_connect_length !=NULL)
    i_content_length = atoi( (char*)p_connect_length );
  if ( ( Buffer_Env = new char[i_content_length+11] ) == NULL)
     {
     set_head("error");
     back_script("402, 系统忙, 请稍候再试。。。");
     if (Buffer_Env)
     delete [] Buffer_Env;
     return -1;
     }
     memset(Buffer_Env, 0, i_content_length+11);
   if(p_connect_length)
     {
     if(fgets(Buffer_Env, i_content_length+1, stdin)<0)
       {
         set_head("error");
         back_script("403, 标准输入错误!");
         if (Buffer_Env)
           delete [] Buffer_Env;
         return -1 ;
       }
     }
     else
       Buffer_Env[0]=0;
  }
  if ( strcmp( pRequestMethod , "GET" ) == 0 )
  {
    char  *p_query_string;
    p_query_string = (char *)getenv("QUERY_STRING");
    if ( p_query_string !=NULL)
```

```
        i_content_length = strlen(p_query_string);
if( ( Buffer_Env = new char[i_content_length+11] ) == NULL)
        {
          set_head("error");
          back_script("405, 系统忙, 请稍候再试...");
          if (Buffer_Env)
            delete [] Buffer_Env;
          return -1;
        }
        if(p_query_string)
          strcpy( Buffer_Env ,  p_query_string );
        else
          Buffer_Env[0]=0;
        }
      }
    else
    {
      Buffer_Env = new char [500];
      sprintf(Buffer_Env, "word=mp4&mem=99");
    }
/*********************************
    int set_env()
  {
     setvbuf(stdin, NULL, _IONBF, 0);
  setvbuf(stdout, NULL, _IOFBF, 1024*16);
  printf("Content-type:  text/html\n");
  printf("\n");
  return 0;
}
```

```
    void set_head( char *title)
  {
    printf( "<HTML>\n" );
    printf( "<HEAD>\n" );
    printf( "<TITLE>%s</TITLE>\n" ,  title );
    printf("</HEAD>\n");

 }
 void set_tail()
{
 printf("</BODY>\n</HTML>\n");
}
```

3.3.2 CGI 标准输入输出

1)标准输入

Web服务器传递数据给CGI程序采用POST的方法时,CGI程序像其他可执行程序一样，可通过标准输入(stdin)从Web服务器得到输入信息，如Form中的数据。

2)标准输出

CGI程序通过标准输出(stdout)将输出信息传送给Web服务器。传送给Web服务器的信息可以用各种格式，通常是以纯文本或者HTML文本的形式。Web服务器对这些结果进行必要的检查,如果CGI程序产生的结果格式有问题,Web服务器会给出一些错误信息，必要的时候还会给出错误日志信息。如果程序运行结果格式正确，Web服务器就会把结果发给客户，这中间Web 服务器还要加上一些HTTP协议所需要的HTTP头信息。CGI的输出必须是包含一个格式如HTTP的头、一个空行、以及正文信息，且只能包含Content-type，Location，Status三种头。

客户端、服务器、CGI接口与外部程序间的关系是相互连通的，服务器是客户端(如浏览器)与扩展程序之间的通道。当客户端的用户完成了一定查询关键字的输入工作之后向服务器发出HTTP请求(CGI请求)，服务器守护进程

接收到该请求后，就创建一个子进程(CGI进程)[25]。该CGI子进程将CGI请求的有关数据设置成环境变量，在外部CGI程序与服务器间建立两条数据通道，然后启动URL指定的CGI程序，并与该子进程保持同步，以监测CGI程序的执行状态。子进程通过输出流将处理结果传递给服务器守护进程，守护进程再将净化后的处理结果作为应答消息回送到客户端。

外部CGI程序通过环境变量、命令行参数、输入/输出与WWW服务器进行通信，传递有关参数和处理结果[26]。

1)环境变量：当服务器守护进程创建子进程运行CGI程序时，设置相应的环境变量和命令行参数，以传递客户端和服务器端的有关信息给该子进程。

2)命令行参数：命令行参数仅在有HTML 文档中有ISINDEX查询的情况下使用。

3)输入/输出：当HTTP请求模式采用POST方式时，CGI程序通过标准输入流和有关环境变量来获取客户端传输数据，如采用GET方式，CGI程序直接通过环境变量获取客户端传输数据。当CGI程序要返回处理结果(一般为HTML文档)给客户端时，它通过输出流将该结果数据传递给服务器守护进程。

3.4 本章小结

本章详细介绍了慧聪网行业搜索引擎的架构，网页净化系统所处的位置，各个子系统之前的关系。这个系统的检索过程：整个检索过程分为初始化检索，打开库，检索，释放存放检索结果的缓冲区，关闭库，退出系统六个部分。数据库分为历史库和最新库，最新库是当天的数据，以及漏加和修改的数据（非当天），为保证快速增删改，最新库数据暂定为最新一天。历史库除最新库以外所有的数据，历史库按分类分成四个分类大库，每个分类大库按时间分为不同的小库。每个小库数据的时间范围可以自定，默认为三个月数据为一个小库。网页净化系统对历史库中的网页进行净化，净化后的网页仍存储在历史库中，把当天的数据存放在最新库中。用户从 Web 端的 CGI 程序输入查询串，CGI 程序对它进行合理的分割并把查询串传入搜索系统，把查询到相关词的页面传入网页净化系统，最后把净化后的页面通过 Web Server 端的 CGI 程序显示在浏览器中。

第4章 信息网网页净化系统详细设计与实现

Web 页面去噪主要包括以下 3 个层次的工作：

1)去除页面中包含的注释、脚本、样式表等信息，这些信息对后续检索、分类等应用的作用小大。这项工作比较简单直接本文中小再赘述。

2)根据页面的组织结构将页面划分为若干几个信息块，这些信息块可以分为文本块、链接块、图像等。这是 Web 页面去噪中的基础而重要的环节。

3)在第一层次工作的基础上，根据信息块的语义对信息块作进一步的区分。这样，以从文本块中区分出版权信息块、广告信息块；从链接块中区分出相关链接块、导航链接块。

4.1 页面净化构架

HTMLParser 是一个对现有的 HTML 进行分析的快速实时的解析器。它是一个开源项目，通过它，可以准确高效地对 HTML 文本中的格式、数据进行处理。利用它可以很容易地对网页的内容进行分析、过滤和抓取[27][28]。它的主要功能有：文本信息抽取、链接提取、资源提取、链接检查和内容检验等。

HTMLParser 类虽然并没有对以上提到的一些功能进行专门的处理，但是它完全可以胜任上面提及的功能，在实际应用中如果遇见上面提及的问题可以使用它内部的一些方法来处理。本文应用 HTMLParser 类来分割网页中的各个 Table，得到各个 Table 的 width 和 height 属性值，并对去噪后的网页进行去标签，也就是将其转化成为文本文件。

HTML 解析器能够读取用户指定的 Web 页面（存储在硬盘上）并依据 HTML 规范对其进行解析，得到一棵 DOM 树（建立在内存中）。该 DOM 树反映了 Web 页面的内容和结构，即组成的 Web 页面的各个元素（元素名称、元素内容、元素所包含的各个属性），以及各个元素之间的层次构成关系。对于页面净化模块而言，结构化的 Web 页面 DOM 树比流式的 Web 页面数据更易于访问和操作[29]。

如图 4.1 所示，Web 页面净化系统 PageClean 由两个部分构成：HTML Parser 函数库和页面净化模块。HTML Parser 函数库将输入的流式 Web 页面解析为 HTML 文档树，页面净化模块在此基础上依据特定的规则对 Web 页面进行净化。把净化后的网页通过 CGI 程序显示给用户的同时，把它们存在网页库中。

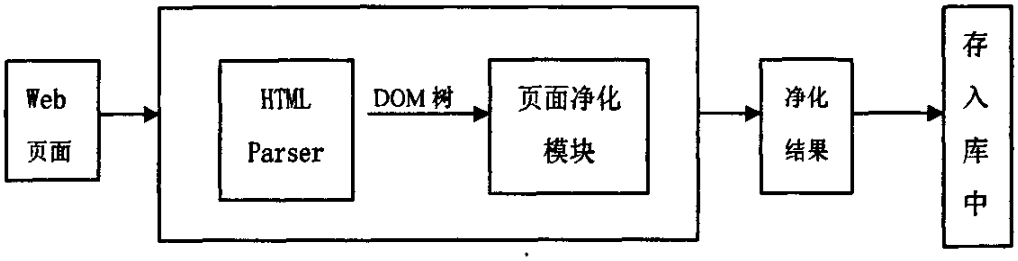


图 4.1 Web 页面净化系统 PageClean

4.2 页面净化模块设计

算法实现依据的规则

根据一般网页的 HTML 文档以及观察其它网页的格式，可以得到一些启发式规则，如下：

1) 标签<table>和</table>之间如果有标签<p>或
，可以看作是正文内容。也就是说认为网页的主题通常是用成段的文字来描述。

2) 若标签<table>的 width 或 height 属性为其占页面的百分比，则需要根据这个百分比的值来确定其是否为主题内容。若 width 或 height 属性的百分比数值较大，则认为有可能是主题内容。

3) 对于多层嵌套的标签<table>，认为只在其中某一层 table 中存在主题内容。

4) 对于没有标签<table>的网页，即不是由表格分割的网页，如果存在段落文字，则认为是主题内容。

4.3 页面净化算法思想

对于有标签<table>的网页，认为重要的信息都放在网页的中间区域，而且该区域长度和宽度都比较大。而网页边缘区域的重要性相对于中间区域都很弱，而且该区域比较狭长；对于没有标签<table>的网页，只是根据其是否存在段落文字来判断是否为主题内容，并没有考虑更多，这是因为对于大多

数网页设计者来说，通常他们会先进行网页布局的设计，即先用表格等标记在页面上描绘出页面内容分布的区域，然后再在每个区域内部进行详细的内容设计，把需要的元素加进去。所以没有表格(table)的网页现在越来越少，这必然是以后网页制作发展的主流方向。本文就是根据这样一种思想来进行网页去噪，然后将提取出的主题内容变为文本文件，为以后对网页的一些处理，如分类、检索等提供了很大的方便。

4.4 页面净化算法流程

1) 由于本文使用的是慧聪网网页库中的网页，而需要事先加载网页库，把网页库中的数据加载到行业搜索引擎的最新库中。

2) 加载完毕后，开始进行去噪工作。对于没有 table 的网页，处理方法如代码 4.1 所示。

代码 4.1 无 table 网页的去噪算法

```
If (无标签<table>){
   If (存在段落文字){
     认为是主题内容，保留
   }
}
```

3) 对于有 table 的网页，处理方法如代码 4.2 所示。

代码 4.2 有 table 网页的去噪算法

```
For (对于每一个 table 表格){
   得到 table 的 width 和 height 属性
   If (width 和 height 属性有一个不存在且 width 不以百分
     比的形式出现){
      If (width 属性值> δ1)  //δ1 为 width 属性阈值
         If (有段落文字)
            认为是主题内容，保留
   }
   Else if (width 和 height 属性都存在且 width 属性以百分
         比的形式出现){
```

If (width 百分比数值>δ2) //δ2 为 width 百分比数值的阈值

If (有段落文字)

认为是主题内容，保留

}

Else if (width 和 height 属性都存在且 height 属性以百分比的形式出现){

If (height 百分比数值>δ3) //δ3 为 height 百分比数性阈值

If (有段落文字)

认为是主题内容，保留

}

Else if (width 和 height 属性都存在且 width 和 height 属性值以数值的方式出现){

计算 width 与 height 的长宽比

If (长宽比很小)

If (有段落文字) //该层判断是为了防止出现长宽比很小的图片等

//非主题内容

认为是主题内容，保留

}

Else{ //对于没有段落文字，即对于目录型网页或图片型网页

认为 table 长宽比很小，但长和宽的值占页面很大的部分为主题内容，保留

}

}//endfor

4) 将去噪后的网页去标签，并转化为文本文件。

4.5 联机现实净化页面

如图 4.2，当客户输入关键词进行查询时，CGI 对该请求建立一个进程，创建共享内存，把查询串处理后，传入搜索系统中，如果关键词匹配，把相关的网页根据固定算法排序，把排序后的网页送到网页净化系统进行净化处理，最后，把处理好的网页显示在浏览器中。

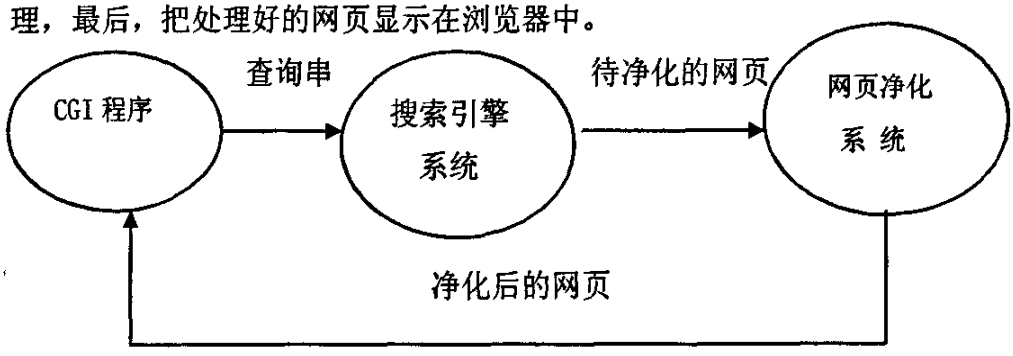


图 4.2 联机实现净化页面

CGI 显示净化后页面的程序代码片断，如代码 4.3。

代码 4.3 CGI 显示净化页面程序代码

```
Void SetSearchResult(long num , CIndSearch *p , NewsRes_Str *sRes ,
SearchPara *parameter )
{
  char *pVal;
  char strTmp[256]=  ;
  char rclass_cc[512]="";
  char rReplacestr[400]= "";
  char *pID;
  struct tm *Nt;
  long  lt;
  char *pPic;
    //docid
  p->FieldsDocid(pVal);//取 docid
  strcpy(sRes[i].docid, pVal);
  p->FreeBuf(pVal);
    //标题标红
```

```
p->FieldsString("BT", pVal);
ToCapital(pVal);
if (parameter->shopWin[0] == '0')
  szReplace(pVal, sRes[i].title, "", "");
else
  szReplace(pVal,sRes[i].title,"<FONT COLOR=ff0000>","</FONT>");
  p->FreeBuf(pVal);
//关键字
  p->FieldsString("TP", pVal);
  strcpy(sRes[i].tp, pVal);
  p->FreeBuf(pVal);
  //正文
  p->FieldsString("JJ", pVal);
  KillWrongKeyForText(pVal);
  char *regex="<[^>]*.";
  char tempo_text[1024]="";
  char temp1[1024] = "";
  reg_htm2txt(regex, pVal, tempo_text);
  if(strstr(tempo_text, "<?xml"))
    strcat(tempo_text, ">");
  reg_htm2txt(regex, tempo_text, temp1);
  szReplace(sRes[i].text, " ", ""); //去掉 
  if(parameter->nSelect != 6 && parameter->nSelect !=7)
    cutBrief(temp1, sRes[i].text, 25);
  else
    cutBrief(temp1, sRes[i].text, 100);
  dealHTML(sRes[i].text, 0);
  ToCapital(sRes[i].text);
  strcpy(tempo_text, sRes[i].text);
  szReplace(tempo_text, sRes[i].text, "", "");
```

```
    p->FreeBuf(pVal);
    //图片
    p->FieldsString("IM", pVal);
    strcpy(sRes[i]。image, pVal);
    p->FreeBuf(pVal);
    strcpy(sRes[i]。corname, pVal);
    p->FreeBuf(pVal);
      p->FieldsNumeric("JY", pVal);
    //放置完毕，指针下移
    p->MoveNext();
}
```

4.6 本章小结

本章是全文的核心部分，详细阐述了慧聪网网页净化系统 PageClean 的架构：HTML Parser 函数库和页面净化模块。HTMLParese 类，一个对现有的 HTML 进行分析的快速实时的解析器，它能读取指定的 HTML 文件，并按照 HTML 规范转换成 DOM 树。HTML Parser 函数库将输入的流式 Web 页面解析为 HTML 文档树，页面净化模块在此基础上依据特定的规则对 Web 页面进行净化。把净化后的网页通过 CGI 程序显示给用户的同时，把它们存在网页库中。净化算法实现依据的规则，分别对有<table>的网页和没有<table>的网页分别阐述其算法依据规则及算法的实现。对于有标签<table>的网页，我们认为重要的信息都放在网页的中间区域，而且该区域长度和宽度都比较大。而网页边缘区域的重要性相对于中间区域都很弱，而且该区域比较狭长，很可能是噪声区；对于没有标签<table>的网页，只是根据其是否存在段落文字来判断是否为主题内容。最后，本章还介绍了联机实现网页净化的具体流程：当客户输入关键词进行查询时，CGI 对该请求建立一个进程，把查询串处理后，传入搜索系统中，如果关键词匹配，把相关的网页根据固定算法排序，把排序后的网页送到网页净化系统进行净化处理，最后，把处理好的网页显示在浏览器中。

第5章 网页净化系统的测试及分析

对于网页净化系统PageClean而言,它的功能主要是对输入的每一个Web页面进行联机净化。从净化速度上来看，由于它需要在前台以联机的方式对海量的Web页面进行净化，因此要求尽可能快速运行；从净化效果上来看，由于它是检索、分类等后续应用的预处理，因此要求尽可能高的正确性。

净化效果的评估主要有两种方法：一种是直接评估，其目的是考察PageClean 本身的性能，即净化的正确性；另一种是间接评估，其目的是考察 PageClean 对后续应用的影响，即在检索、分类等应用中，Web 页面净化对检索精度、召回率、分类准确度等指标的影响。在此次实验中，仅包含直接评估结果，评估过程主要由人工对净化结果的正确性进行评价。

5.1 网页净化系统的实验数据集

本文对来自慧聪网网页库中的1142589个网页，平均每个网页约20k进行测试。经过测试，网页大小由去噪前的22.85G减少到了去噪后的17.14G，网页噪音部分竟然占了网页约75%的大小！可想而知，对Web网页进行分类或检索等其它应用时，如果不事先进行消除噪音，那么对于分类或检索的速度和精度将会有多么大的影响。

对于待净化的Web页面没有特别的要求，也可以随机地选择一些除慧聪网网页库以外其他Web站点上的页面。由于PageClean具有批处理功能，因此执行一次可以完成对一批Web页面的净化工作。

进行净化速度评估时使用的Web页面数据集如表5.1所示。

表5.1 Web页面数据集

页面数目	页面大小总和	平均文件大小
1142589个	22.85G	20kB/文件

5.2 测试网络环境

WebServer端共六台Unix服务器，其中4台在北京，另外两台在上海。

搜索引擎系统服务器 Windows2003 在北京，网页净化系统 PageClean 系统 Windows2003。

5.3 测试软硬件环境

5.3.1 测试软件环境

操作系统：Free BSD 5.5，Windows 2003

Apache 2.0 Unix 版本，GCC 编译器，makefile，SSH Secure Shell，Visual C++6.0，Oracle9i

5.3.2 测试硬件环境

WebServer 端服务器 IBM R6000，（4G/15*1 46G）（双机备份）

为了验证本课题研究的网页净化系统设计和实现的有效性和合理性，本章将描述该系统的测试的环境、方法和标准，并根据测试数据，得出结论：系统能够满足用户功能需求。

5.4 网页净化系统的速度评估

就速度而言，评估标准非常简单，即计量页面净化模块。在 1142589 个网页中随即抽取 700 个页面进行特定的净化执行时间计算，结果如表 5.2。

表 5.2 速度评估

执行时间（s）	速度（文件/s）	速度（kB/s）
10	70	898

可以看出，页面净化模块执行的速度比较快，完全能够满足对大批量 Web 页面进行快速联机净化的要求。

5.5 网页净化系统的效果评估

实验只对1142589个网页中的2588个(供应信息部分) 网页进行人工检验。实验结果分为3种情况：

1)正确去噪，表示达到网页去噪的效果。

2)错误去噪，表示去噪效果不好， 仍存在较多网页噪音。

3)无法去噪，表示去噪算法对该网页没有作用。实验结果为：正确去噪率99.5%；错误去噪率0%；无法去噪率0.5%。可见，本文算法能够较好地对大

多数网页进行去噪。

对文字块和链接块区分的判断标准很难加以形式化，同时存在主观性，因此采用总体准确度（accuracy）作为评估标准。具体评估时可以采用人工评价打分的方法。即人工检查净化系统的提取结果，并按照"好、一般、差"三档进行打分。如果页面提取正确则打分为"好"、如果有少量一般性的错误但结果整体上可以接受则打分为"一般"、如果有严重错误或错误较多则打分为"差"。分别对两批网页进行净化效果评估，每批网页有 100 个，净化效果评估结果如表 5.3 和 5.4 所示。

表 5.3 第一批网页净化效果评估结果

效果	好	一般	差
页面数目	31	50	19
百分比	31%	50%	19%

表 5.4 第二批网页净化效果评估结果

效果	好	一般	差
页面数目	62	24	14
百分比	62%	24%	14%

可以看出，前台页面净化模块的效果还是可以的，净化结果中可以接受的占 80%以上。

综合上述页面净化的实验结果，从直接评估的角度来看，PageClean 无论是在净化速度还是在净化效果上都具有较好的性能，达到了预期的目标。但是，对 Web 页面的净化是否能够提高后续应用的效果，还有待于对其进行间接评估实验。

5.6 网页净化系统的分析

在对去噪效果较差的网页的观察中，可以发现以下一些原因：首先，网页的主题内容不明显，即主题内容只有一句话，或者是网页中的主题是图片

信息等等。其次，一少部分网页的噪音中也存在一些段落文字的内容，也就是说存在<p>或
等认为是段落文字的标签，这样会误把这部分内容看作主题内容保留下来而没有被清除掉。算法已经对类似这样的错误进行了一定的处理，因为噪音中的段落文字通常比主题内容中的段落文字少，即噪音中的标签<p>或
的个数会比主题内容中的少，所以，针对这种情况，可以根据标签<p>或
的个数来判断该部分是噪音还是主题内容。

我们对页面净化模块在区分文本块和链接块时的错误进行了分析，发现主要原因在于区分"块"时所用的规则还不完善。事实上，对于一个 Web 页面而言，在语法上没有关于"块"的定义。页面净化模块在界定"块"时主要依据启发规则。但是，应该说，绝对正确的界定规则是不存在的，即使是人工在界定"块"时也存在主观性和模糊性。

PageClean 系统的输出端是由 CGI 编写的程序，之所以采用 CGI 而不采用 ASP，JSP 等是由于 CGI 执行速度快，速度是衡量一个搜索引擎好坏的一个重要指标。Google，百度也是采用的 CGI 程序。但是，CGI 程序有一个非常致命的弱点：CGI 程序每启动一次就要生成一个进程，慧聪网日访问量为平均 1300 万/天，日访问人数为平均 260 万/天，这样就需要非常强大的处理器和非常大的内存，服务器的负担非常重。如果采用 FastCGI 就可以解决上述的问题。FastCGI 像是一个常驻 (long-live)型的 CGI，它可以一直执行着，只要激活后，不会每次都要花费时间去 fork 一次 (这是 CGI 最为人诟病的 fork-and-execute 模式)。FastCGI 的应用程序亦兼容于 CGI，即 FastCGI 的应用程序也可以当成 CGI 来执行。把现有的 CGI 程序改成 FastCGI 是本系统下一个版本的主要任务。

5.7 本章小结

本章首先描述了网页净化系统 PageClean 测试的数据集，实验的网络环境，软硬件环境，测试的数据集是慧聪网行业搜索引擎网页库提供的。然后本章又详细的介绍了网页净化系统在速度和效率上的数据，并对这些数据进行了定量的分析。从直接评估的角度来看，PageClean 无论是在净化速度还是在净化效果上都具有较好的性能，达到了预期的目标。最后，介绍了网页净化系统上的不足，分别从网页分块规则的不完善和 Web 端 CGI 程序的瓶颈

上进行分析，得出结果，及其改进方案。即对于一个 Web 页面而言，区分"块"时所用的规则还不完善，在语法上没有关于"块"的定义，页面净化模块在界定"块"时主要依据启发规则。CGI 程序有一个非常致命的弱点：CGI 程序每启动一次就要生成一个进程，这样对服务器的负担是非常重的，采用 FastCGI 可以解决这个问题。把 WebServer 端的 CGI 程序修改成 FastCGI 也是本课题以后将要做的工作。

结 论

本论文主要研究和实现了慧聪网行业搜索引擎网页净化系统，为慧聪网行业搜索引擎系统的预处理阶段中增加检索命中率，提高服务器性能提供了解决办法。在论文工作过程中，查阅和借鉴了当前国内外关于网页净化和搜索引擎的先进技术和方法，针对慧聪网行业搜索引擎系统架构的具体特点，结合 b2b 网络营销模式及有关需求，提出了慧聪网网页净化系统的详细设计方案，并实现了网页联机净化的方法。具体工作可归纳为以下几个方面：

1)利用 HTMLParser 实现 DOM 树

HTMLParser 类 是一个对现有的 HTML 进行分析的快速实时的解析器。利用这个类可以把 HTML 文件转换成 DOM 树结构。

2)网页净化系统 Web 端 CGI 程序

网页净化系统 PageClean 的输出端是用 CGI 程序完成，CGI 净化后的网页显示在浏览器中。

3)网页净化系统算法的依据

根据 HTML 网页标签的特点，提出网页净化系统算法依据的一些规则。

4)网页净化系统算法的思想及流程

对有标签<table>的网页，认为重要的信息都放在网页的中间区域，而且该区域长度和宽度都比较大。而网页边缘区域的重要性相对于中间区域都很弱，而且该区域比较狭长；这些重要性相对弱的，狭长的区域就被认为是网页噪声区。对于没有标签<table>的网页，由于网页制作人员一般都使用网页制作工作，象 Dreamweaver 等，在制作网页过程中绝大多数是建立<table>后，再进行其他操作的，目前在万维网上越来越少，可以忽略。

最后，该系统通过大量反复测试，并在速度和效果上做了定性的评估，现正在慧聪网行业搜索引擎上试运行，其效果良好，这也说明该研究成果的可行性。

互联网的飞速发展给搜索引擎技术的研究带来新的机遇和挑战。随着数据信息库积累的数据越来越多，怎样快速、有效、经济地检索到某个主题的

所有相关的信息就成了一个当前十分热门的课题之一[30]。慧聪网行业搜索引擎的网页净化系统极大地解决了搜索预处理阶段的网页命中率和服务器负载的问题，在每天1300万的流量中，表现良好。

参考文献

[1] Yiming Yang. Noise Reduction in a Statistical Approach to Text Categorization. In Proceedings of SIGIR-95, 18th ACM International Conference on Research and Development in Information Retrieval. 1995: 256-263P

[2] LI Xiaoli and Shi Zhongzhi. Innovating Web Page Classification Through Reducing Noise. In Journal of Computer Science & Technology. 2002, 17(1): 9-17P

[3] 常育红，姜哲，朱小燕. 基于标记树表示方法的页面结构分. 计算机工程与应用. 2004（16）：129-132页

[4] 李效东，顾毓清. 基于DOM的Web信息提取. 计算机学报. 2002, 25（5）：526-533页

[5] 王琦，唐世渭，杨冬青，王腾蛟. 基于DOM的网页主题信息自动提取. 计算机研究与发展. 2004, 41（10）：1786-1792页

[6] Shian-Hua Lin and Jan-Ming Ho. Discovering Informative Content Blocks from Web Documents. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2002

[7] Lan Yi, Bing Liu, Xiaoli Li. Eliminating noisy information in Web pages for data mining. KDD, 2003: 296-305P

[8] 封化民，刘飚，刘艳敏. 含有位置坐标树的Web页面分析和内容提取框架. 清华大学学报. 2005, 45（S1）：1767-1771页

[9] Suhit Gupta, Gail E. Kaiser, David Neistadt Peter Grimm: DOM-based content extraction of HTML documents In Proceedings International WWW Conference, Budapest, Hungary. 2003: 207-214P

[10] Deng Cai, Shipeng Yu, Ji-Rong Wen, Wei-Ying Ma: Extracting Content Structure for Web Pages Based on Visual Representation. APWeb: 406-417P

[11] Oren Eizioni. The World Wide Web Quagmire or Gold Mine Communication of CAM.1996, 39(11): 65-68P

[12] 欧健文，董守斌，蔡斌. 模板化网页主题信息的提取方法. 清华大学学报. 2005, 45 (S1): 1743-1747页

[13] Hors A L, Wilson C.Document Object Model(DOM) Level 2 HTML Specification(Version 1.0).W3C Working Draft, 2000-11-13

[14] LI Xiaodong, GU Yuqing. DOM 2based information extraction for the web sources[J].Ch in J Computers, 2002, 25(5): 526-533P

[15] Raggett D, Hors A L.HTML 4.01 Specification.W3C Recommendation, 1999: 12P

[16] 李晶，陈恩红. Web信息抽取. 计算机科学. 2003, 30 (6): 78-81页

[17] 王继成，张福炎. Web文本挖掘技术研究. 计算机研究与发展. 2000, 37 (5): 513-520页

[18] 刘杰. 一种高效的HTML/XHTML至WML的转换方法. 北京工商大学学报. 2006, 24 (6): 45-47页

[19] 韩家炜，孟小峰，王静，李盛恩. Web挖掘研究. 计算机研究与发展. 2001 (4): 405-414页

[20] 刘开瑛，薛翠芳，郑家恒，周晓强. 中文文本中抽取特征信息的区域与技术. 中文信息学报. 1998, 12 (2): 1-2页

[21] 瞿有利，于浩，徐国伟等. Web页面信息块的自动分割. 中文信息学报. 2004, 18 (1): 6-13页

[22] 于满泉，陈铁睿，许洪波. 基于分块的网页信息分析解析器的研究与设计. 计算机应用. 2005, 25 (4): 974-976页

[23] 邹涛，王继成等. WWW上的信息挖掘技术及其实现. 计算机研究与发展. 1999, 36 (8): 1019-1024页

[24] LI Xiaoli and Shi Zhongzhi. Innovating Web Page Classification Through Reducing Noise. In Journal of Computer Science & Technology. 2002, 17(1): 9-17P

[25] 关胜晓. 公共网关接口CGI机制浅析. 微电子学与计算机. 1998 (4): 15-17页

[26] 赵翔，朱志明.一种提高CGI性能的方法.小型微型计算机系统. 1997，18（4）：74-75页

[27] 陈剑化，包煊. Web挖掘系统的设计与实现. 计算机工程. 2002，28（8）：141-143页

[28] 李芳，盛焕烨，姚天防. 信息检索与信息抽取技术的研究. 计算机应用研究. 2002（1）：16-18页

[29] 王实，高文，李锦涛. Web数据挖掘. 计算机科学. 2000，27（4）：28-31页

[30] 陈莉，焦李成. Internet Web 数据挖掘研究及最新进展. 西安电子科技大学（自然科学版）. 2001，28（1）：114-119页

攻读硕士学位期间发表的论文和取得的科研成果

[1] 徐冉，杨永田．网页净化方法的研究．计算机工程．2006（3）已投稿

致 谢

本文至此已经到了尾声，我在哈尔滨工程大学两年多的学习生活也将告一段落。在此，我要衷心感谢我的导师杨永田教授。学习期间，杨老师以他敏锐的思维、渊博的知识给我留下了深刻的印象。在学习和研究上，杨老师对我严格要求、耐心指导，为我提供了良好的学习和研究环境，丰富了我的知识，所有这些都对我受益匪浅。

在此，谨向尊敬的导师杨永田教授表示我最衷心的感谢和崇高的敬意！在哈尔滨工程大学的学习和生活对我一生的影响都是深远的，也留下了许多美好的记忆。在这里我认识了许多学识渊博的老师，正是由于他们的辛勤施教，使我从他们那里获得了很多教益和帮助。

另外，我与北京慧聪网的合作非常愉快，是北京慧聪网给了我这个课题进行研究。此外，计算机网络实验室的讨论和交流给予我难得的学习机会，在这里对实验室的各位表示真挚的感谢。

感谢我的父母，感谢他们一直以来给我在生活和学业上无私的关心、资助和支持。

最后，感谢 2004 级的全体研究生同学在两年多里给予我的帮助和支持。