广西科学院学报
Journal of Guangxi Academy of Sciences

2010,26(4):486～489
Vol. 26, No. 4 November 2010

# MS Windows 环境的桌面搜索系统设计与实现
# Design and Implementation of Desktop Search System under MS Windows

黄家裕,温家凯,刘连芳
HUANG Jia-yu, WEN Jia-kai, LIU Lian-fang

(南宁市平方软件新技术有限责任公司,广西南宁 530007)
(Pingsoft New Technology Co. Ltd. of Nanning, Nanning, Guangxi, 530007, China)

**摘要**:提出一个基于 Lucene 的 MS Windows 环境桌面搜索系统架构,探讨文件监视、基于磁盘文件的外存队列、文件文本提取以及索引和检索等各关键模块的实现方法,并测试验证该系统具有较高的实用价值。

**关键词**:桌面搜索 文件监视 外存队列 文件文本提取 Lucene
**中图法分类号**:TP391.3 **文献标识码**:A **文章编号**:1002-7378(2010)04-0486-04

**Abstract**: An architechture of a Lucene-based MS Windows Desktop Searching System is presented in this paper. The implemental methods on the key modules of the system such as file-monitering module, external storage queue module based on disk files, text-extracting module and indexing and Retrieving module are discussed and the high practical values of this system are praved.

**Key words**: desktop search, file-monitoring, external storage queues, file text extraction, Lucene

桌面搜索工具出现之前,人们用于查询个人计算机硬盘数据的主要工具是操作系统自带的查询工具,其特点是:(1)只能查找文件名和文本文件中的文本;(2)在用户需要查找数据时才逐一打开文件进行匹配,速度缓慢。随着计算机软硬件以及互联网的飞速发展,个人计算机硬盘中存储的数据越来越庞大复杂,数据量达几十 GB 甚至上百 GB,常用文档类型达数十种。面对这种状况,操作系统自带的工具已经很难满足用户的快速定位信息的要求。

2004 年起 Google、雅虎、微软、百度、中搜等搜索企业相继推出了自己的桌面搜索工具,对个人计算机硬盘数据进行索引,提供类似于互联网搜索引擎的检索功能,切实提高了对个人计算机数据查找的能力,逐渐获得用户的青睐。这些工具特点是:1)采用全文检索技术,预先建立索引,检索速度快;2)支持检索多种常用文档的文本。桌面搜索已成为搜索技术应用研究的新热点,并逐渐成为各搜索企业新的竞争领域。由于商业机密等原因,商业搜索企业均未公开发表相关技术文章对桌面搜索进行探讨。目前公开的研究有 Novell 的开源项目 Beagle[1],该系统运行于 Linux GNOME 桌面,其核心采用开源全文检索软件 Lucene[2] 实现。另外德国 L3S 研究中心研发了 Beagle++[3],该系统在 Beagle 基础上扩展了元数据搜索,是 NEPOMUK[4] (Networked Environment for Personalized, Ontology-based Management of Unified Knowledge)项目的子课题,为 Social Semantic Desktop 提供桌面搜索工具。可见,桌面搜索不仅是现有桌面的主要解决方案,也是未来新型桌面的重要组成部分。国内于江德[4]等基于 Oracle Ultra Search 组件研究了 MS Windows 的桌面搜索系统的实现,系统有较好的性能,但是系统只是 Oracle Ultra Search 一种应用,完全依赖于商业软件 Oracle,不具备独立性,应

**收稿日期**:2010-09-28
**修回日期**:2010-10-16

**作者简介**:黄家裕(1975- ),男,软件工程师,主要从事中文信息处理、机器翻译、计算机通信研究工作。

用成本高;另外该系统采用定期激活爬虫的方式收集数据,不具有实时性,与主流的桌面搜索系统有较大的差别。本文提出一个基于 Lucene 的 MS Windows 环境桌面搜索系统架构,并探讨各关键模块的实现方法,为国内桌面搜索领域相关研究提供有益参考。

## 1 系统概述

本系统运行于 MS Windows 环境,以全文检索为核心,提供对文件名、文件元数据(包括创建时间、最近更改时间等)以及文件中的文本等进行查找的功能,支持的文件类型包括 html、pdf、office、rtf、txt、图片、影音、邮件等等。系统允许用户指定需要索引的位置和数据类型范围,因为主要数据往往集中在某些硬盘分区或某些目录中,而且不同用户会偏向不同数据类型,即用户并不需要搜索硬盘上所有的数据,因此这项功能既可减少不必要的数据干扰,又可以避免不必要的资源浪费。本系统对各种数据源进行监视,及时发现文件的变动,及时更新索引,确保索引数据的实时性。

## 2 系统架构

本系统由数据监视、数据处理控制中心、数据扫描、信息提取、索引、检索等模块组成(图 1)。"数据监视"模块负责监视索引范围以及数据本身的变更,将变更消息提交"数据处理控制中心"。"数据处理控制中心"负责将通知消息加入处理队列,控制中心的另外一个线程则不断从队列中取出通知并处理:如果是增加了索引范围或硬盘上增加了目录,则交由"数据扫描"模块进行处理;如果是增加了文件或文件发生了更改,则直接交由"信息提取"模块处理,然后调用"索引"模块进行索引;如果是删除了索引范围、文件或文件夹则直接调用"索引"模块删除相应数据的索引。"数据扫描"模块负责扫描指定目录,获取所有文件的文件名,为这些文件构造"添加文件"的通知消息,提交"数据处理控制中心"加入队列。"信息提取"模块负责判断要处理的数据的类型,寻找合适的提取算法,从文件中提取元数据及内容文本。"索引"模块负责添加删除数据的索引。"检索"模块接收用户查询请求,从索引库中查询符合条件的数据,抽取动态摘要,将结果信息返回给用户。

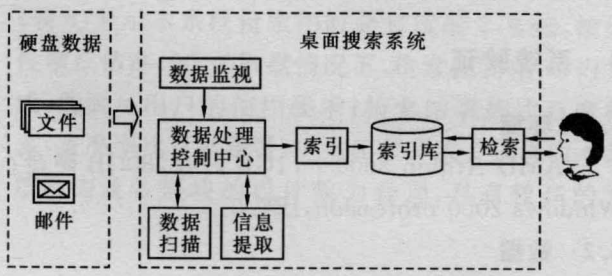


图 1 系统架构

## 3 系统模块设计

### 3.1 数据监视

系统采用 MS Windows 下发现文件变更的 IOCP 方式实现对文件系统的监视。首先通过 CreateIoCompletionPort 创建一个完成端口绑定要监视的一个或多个文件(或文件夹);然后另外启动一个侦听线程,通过 GetQueuedCompletionStatus 绑定上述完成端口,等待操作系统的通知;当该文件完成一次 I/O 操作时,操作系统将已经完成的 I/O 操作信息放入完成端口中,并通知侦听线程,让侦听线程从完成端口获取该 I/O 操作信息,从而达到监视目的。

### 3.2 数据处理控制中心

设置"数据处理控制中心"的一个原因是文本抽取及索引的用时远远比从完成端口读取信息的用时要长,如果让监视过程等待文本抽取和索引过程,可能会因为环境异常、系统异常或通知队列溢出而导致部分数据的丢失,所以监视到的信息需要及时存储到外存中,然后再逐一处理。

系统设计一个基于磁盘文件的外存队列统一存放扫描和监视到的消息。对于基于磁盘文件的队列,如果不能重复利用文件空间,将会导致文件不断增长,最终耗尽磁盘空间。我们借助链表的概念,在文件中存放 2 个链表:队列链表,可重用节点链表。在文件头存放基础数据:队列头指针(即队列头节点开始地址,地址值为 -1 则表示指针指向 null)、队列尾指针、可重用节点链表头指针、可重用节点链表尾指针以及节点空间大小(节点空间大小固定)等数据。每个节点除了记录通知消息外,还记录后续节点的指针。元素的入列算法是读入基础数据,查看可重用节点链表是否为空(即判断头指针和尾指针是否指向 null),如果非空则将该链表的头节点加入队列链表末尾(即将队列链表尾节点的后续节点指针指向该节点,将队列链表尾指针也指向该节点),并从可重用节点链表中删除(即将可重用节点链表头指针指向下一个节点),更新该节点信息,保存基础数据。元素的出列的算法是读入基础数据,将节

点加入可重用节点链表末尾，并从队列链表中删除，保存基础数据。

### 3.3 信息提取

“信息提取”模块需要通过文件的后缀名或MIME type判断文件类型，并交由合适的算法抽取其文本。

系统解析MS Office文档结构，寻找文本存放的地方，直接读取文本，而不是通过MS Office提供的COM组件直接操纵MS Office进程，在进程中打开相应的文档，使用COM提供的命令将Office文档保存成txt格式，从而获得文档中的文本。因为后一种方法完全依赖MS Office软件，对还没有安装MS Office的计算机无效，同时该方法直接操纵MS Office进程，容易与人工操作及其它进程发生冲突。

Word文本提取方法是使用OLE API打开Word文档，构造出所有的Stream，寻找名字为WordDocument的Stream。从WordDocument Stream中读取FIB，如果fib.fComplex为false，则从fib.fcMin和fib.fcMac指定的范围中读取文本；如果fib.fComplex为true，则读取Piece Table，根据Piece Table读取文本。

Excel文本提取方法是使用OLE API打开Excel文档，构造各个Stream，寻找WorkBook Stream，从中定位各个Sheet SubStream，通过Sheet SubStream中的Index Record寻找各个Cell，从中读取文本。

PowerPoint文本提取方法是使用OLE API打开PowerPoint文档，获取CurrentUserStream，从CurrentUserStream的找到CurrentUserAtom，获取DocumentStream。在DocumentStream中逐个定位一系列的UserEditAtom，构造PersistPtrIncrementalBlock，根据获得的PersistPtrIncrementalBlock找到Document的位置，在Document中顺序寻找所有的SlidPersistAtom；根据SlidPersistAtoms定位每个Slide Contain的位置，在Slide Contain中顺序查找TextCharsAtom和TextBytesAtom，从中读取文本。

Pdf文件结构由文件头(Head)、文件体(Body)、交叉引用表(Cross－reference Table)和文件尾(Trailer)顺序排列而成。而Pdf的文档结构是一个树形结构，根节点是Catalog，根节点下包含以下各个子树：Pages Tree、Outline Tree、Article Threads、Named Destination、AcroForm等，而Pdf文档的文本都存放在Page之中。Pdf文档的文本抽取流程是首先从文件末尾读取最后一个文件尾，找到startxref的值(最后交叉引用表地址)；根据startxref的定位最后交叉引用表，从最后交叉引用表开始构造整个交叉引用表；交叉引用表允许随机访问文件体中的间接对象，因此通过交叉引用表直接构建Page Tree；遍历Page Tree，读取每个Page中的文本。

### 3.4 索引及检索

系统采用Lucene[2]全文检索架构开发索引和检索模块。使用IndexWriter创建索引，IndexReader删除索引，IndexSearcher(针对单个库)和MultiSearcher(针对多个库)进行数据检索。Lucene的索引库的每个字段有4种格式[5]：(1)Keyword(不切分，索引，保存)；(2)UnStored(切分，索引，不保存)；(3)Text(切分，索引，保存)；(4)UnIndexed(不切分，不索引，仅保存)。切分意味着该字段的数据将被切分成一个个Term，索引表示字段可以检索，保存意味着可以获得该字段的原始内容。由于不同数据类型需要索引的信息不同，所以系统建立4个索引库：文档库、图片库、影音库、邮件库。每个库都有不同的字段，文档库和邮件库的字段如表1和表2所示。当用户指定检索某种数据类型，则使用IndexSearcher在相应的库中检索，而如果用户不指定数据类型，则使用MultiSearcher在所有库中检索。

**表1 文档库字段**

| 字段名 | 格式 | 字段名 | 格式 | 字段名 | 格式 |
|---|---|---|---|---|---|
| 文件名 | Keyword | 作者 | Keyword | 文本内容 | Text |
| 文件类型 | Keyword | 创建时间 | Keyword | | |
| 标题 | Text | 最近修改时间 | Keyword | | |

**表2 邮件库字段**

| 字段名 | 格式 | 字段名 | 格式 | 字段名 | 格式 |
|---|---|---|---|---|---|
| 邮件地址 | Keyword | 接收者 | Keyword | 文本内容 | Text |
| 标题 | Text | 发送时间 | Keyword | 附件内容 | UnStored |
| 发送者 | Keyword | 接收时间 | Keyword | | |

## 4 系统验证

### 4.1 环境

AMD Athlon 3600＋，1GB内存，80GB硬盘；Windows 2000 profession，IE6.0。

### 4.2 数据

3组数据，每组数据包含Word、Excel、Power-

Point、PDF、Html、Txt、邮件各 100 个文件,即,每组数据 700 个文件。数据 1 大小:260.874MB,数据 2 大小:139.811MB,数据 3 大小:232.078MB。

### 4.3 验证内容及结果

#### 4.3.1 功能验证

将数据 1、数据 2、数据 3 所在的文件夹添加到索引范围,索引完成后提示共索引 2100 个文件,分别从数据 1、数据 2、数据 3 中每种文件类型各随机选择 5 篇,取文件中的一小段文本进行检索,均能检索到相应的文件;在数据 1、数据 2、数据 3 所在的文件夹中进行添加、删除、修改、拷贝、剪切、粘贴文件及文件夹的操作,检索结果均能正确反映这些变化,证明系统功能运行良好。

#### 4.3.2 性能验证

为了使评价结果相对客观,将本系统与百度硬盘搜索 2.7 版进行对比测试。

##### 4.3.2.1 索引效率

由于百度硬盘搜索只在操作系统相对空闲时进行索引,而且索引时间只显示到分钟,所以,本测试在操作系统无其他操作的情况下,以分钟为单位,分别对上述三组数据进行索引,记录索引用时。结果(表 3)显示本系统的索引速度与百度相当。

**表 3 索引用时**

| 系统 | 用时(min) | | |
|---|---|---|---|
| | 数据 1 | 数据 2 | 数据 3 |
| 百度 | 2 | 2 | 3 |
| 本文 | 3 | 2 | 3 |

##### 4.3.2.2 检索效率和效果

在大数据量情况下,检索的查全率和查准率较难统计,因此本文使用前 $x$ 命中记录准确率[6] $P(x)$ 来评价检索的准确程度($x$ 取值 50),并且不做查全率统计,只提供返回的结果数作为参考。在完成上述三组数据索引后进行检索测试。测试分三趟,每趟从词典中随机抽取若干个词,逐一检索,记录前 10 个有效检索(有检索结果)的检索用时,并记录每个有效检索的前 50 条结果的准确率。结果(表 4)显示本系统检索用时是百度的 2.9 倍,按线性增长估算,9G 的数据情况下,检索仍能在 1s 内完成,能满足用户的使用要求;检索结果数比百度稍多,查准率比百度稍低,但是相差不大。这说明本系统架构及各模块的设计较为合理,具有较高的实用性。

**表 4 检索效率及效果***

| 检索 | 百度 | | | 本文 | | |
|---|---|---|---|---|---|---|
| | 速度(秒/词) | 结果数(条/词) | $P(x)$(%) | 速度(秒/词) | 结果数(条/词) | $P(x)$(%) |
| 第一趟 | 0.023 | 223.7 | 100 | 0.074 | 227.8 | 100 |
| 第二趟 | 0.016 | 23.8 | 100 | 0.031 | 27.7 | 97.62 |
| 第三趟 | 0.031 | 237 | 100 | 0.099 | 273.5 | 97.4 |
| 平均 | 0.023 | 161.5 | 100 | 0.068 | 176.3 | 98.34 |

*:表中速度、结果数和 $P(x)$ 的数值都是前 10 个有效检索的平均值。

## 5 结束语

桌面搜索是搜索技术应用研究的新热点,也是未来新型桌面的重要组成部分。本文提出一个 MS Windows 桌面搜索的系统架构,探讨各关键模块的实现方法。通过测试验证该系统各项功能运行良好,索引速度与百度相当,检索用时是百度的 2.9 倍,能满足用户的使用要求,检索结果数比百度稍多,查准率和百度相差不大,具有较高的实用价值。桌面数据是用户的工作数据,也是用户赖以形成工作知识的基础,我们下一步的工作重点是研究通过文本自动分类、聚类等技术为桌面搜索系统增加知识挖掘功能,促进用户的知识转化。

**参考文献:**

[1] Beagle. Development-Beagle[EB/OL]. (2010-09-10). http://beagle-project.org/Development HYPERLINK "http://beagle-project.org/Development".

[2] The Apache Software Foundation. Welcome to lucene![EB/OL]. (2010-09-09). http://lucene.apache.org/ HYPERLINK"http://lucene.apache.org/".

[3] Par Lannerc. NEPOMUK-The social semantic desktop [EB/OL]. (2010-09-10). http://nepomuk.semantic-desktop.org/xwiki/bin/view/Main1/.

[4] 于江德,樊孝忠,尹继豪. 基于 Ultra Search 的桌面搜索设计与实现[J]. 广西师范大学学报:自然科学版,2007,25(2):218-221.

[5] Hatcher E, Gospod nefic O. Lucene in action[M]. Greenwich: Manning Press, 2005.

[6] Leighton Vemon H, Srivastava Jaideep. First 20 precision among World Wide Web search services (search engines)[J]. Journal of the American Society for Information Science, 1999, 50(10): 870-881.

(责任编辑:邓大玉)