Computer Science and Application 计算机科学与应用, 2018, 8(12), 1895-1905
Published Online December 2018 in Hans. http://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2018.812211

Hans汉斯

Research and Implementation of Digital Ancient Book Library Based on Solr/Lucene

Xiaotao Chen, Yao Song, Hua Ye, Yanlan Yang

School of Automation, Southeast University, Nanjing Jiangsu
Email: 1785832339@qq.com

Received: Dec. 8th, 2018; accepted: Dec. 21st, 2018; published: Dec. 28th, 2018

Abstract

In order to meet the needs of readers for the retrieval and reading of ancient books, improve the value of ancient books, and protect the integrity of ancient books, under the large data volume of ancient books, this paper designs and constructs a Solr/Lucene full-text search ancient books library system for the low efficiency of ancient books search services, combined with SpringBoot, Nginx and other digital technologies. This paper describes the architectural design of the system. Firstly, the metadata of ancient books is extracted to establish Solr index, and then SpringBoot is used to build an application server to manage the access and storage between the client and Solr data. Nginx is used as a static resource server to provide storage access for ancient books and pictures. The test and application results show that the system has complete search results, including metadata information and picture path information of ancient books. The search methods are diverse, the response time is short, and the system retrieval time is much better than the traditional database retrieval.

Keywords

Ancient Books, Full-Text Search Engine, Solr, Nginx, SpringBoot

基于Solr/Lucene的数字化古籍书库的研究与实现

陈晓涛，宋　尧，叶　桦，仰燕兰

东南大学自动化学院，江苏　南京
Email: 1785832339@qq.com

收稿日期：2018年12月8日；录用日期：2018年12月21日；发布日期：2018年12月28日

摘　要

为满足读者对古籍的检索阅读的需求，提高古籍的利用价值，保护古籍的完整性，并且针对大数据量下

文章引用: 陈晓涛, 宋尧, 叶桦, 仰燕兰. 基于 Solr/Lucene 的数字化古籍书库的研究与实现[J]. 计算机科学与应用, 2018, 8(12): 1895-1905. DOI: 10.12677/csa.2018.812211

古籍搜索服务的效率低的问题，本文设计并构建了一种以Solr/Lucene全文搜索技术为核心，结合SpringBoot、Nginx等多门技术的数字化古籍书库系统。本文介绍了该系统的架构设计，首先提取古籍图书的元数据建立Solr索引，再利用SpringBoot搭建应用服务器，管理客户端与Solr数据中间的访问和存储，采用Nginx作为静态资源服务器，提供古籍图书图片的存储访问。经过测试和应用结果表明，该系统搜索结果齐全，包含了古籍的元数据信息和图片路径信息；搜索方式多样，响应时间短，系统检索时间大大优于传统的数据库检索。

关键词

古籍，全文搜索引擎，Solr，Nginx，SpringBoot

1. 引言

古籍作为中华历史文化的产物和依托，具有重要的学术意义和历史文物价值。古籍大多以纸质为载体保存，而且大多都很脆弱，大量的古籍深藏于仓库之中，不方便学者借阅，更加难以得到学者的合理利用[1]。近些年来，信息技术的快速发展推动了各行各业信息化管理的转型过程，在信息化逐渐成熟的背景下图书信息化建设水平也越来越高[2]，数字化书库以其图书的快速检索、在线阅读等功能受到读者的广泛欢迎。

当前，图书系统中主要采用两种检索模式：1) 以传统的数据库为基础，利用成熟的商业数据库使用 SQL 语句检索图书信息；2) 利用全文检索技术原理，开发搜索引擎，如百度、Google 等。随着数据量的不断增大，数据库的检索效率大幅度降低，并且数据库检索不支持复杂的查询方式。本文采用全文检索技术，提出了一种以传统数据库为存储，基于 Solr/Lucene 的分布式检索的方案，为古籍用户读者提供更加快捷多样化的检索服务，并采用 Nginx 作为静态服务器，提供古籍图书在线阅读的方案，为读者提供良好的在线阅读体验。

2. 技术综述

2.1. SpringBoot 框架

Spring 是一款为了解决企业应用程序开发复杂性而创建的开源的、轻量级的框架。Spring 最重要的核心功能是 IoC (Invertion of Controll，控制反转)和 AOP (Aspect Oriented Programming，面向切面编程)。其中 IoC 用于管理 Java 对象和 Java 对象之间的依赖关系，AOP 用于抽象和解耦业务代码和公共服务代码(如日志、安全、事务等)。IoC 和 AOP 两大核心功能可以简化开发，使得代码具有良好的松耦合性和可测试性，为开发人员提供了很大的便利。

SpringBoot 是由 Pivotal 团队提供的一个全新的基于 Spring 的框架，但大大简化了 Spring 应用程序的各个方面的配置，包括搭建程序框架、配置、开发和部署等。SpringBoot 基于“约定优先配置”的原则，相比于 Spring 框架，开发人员不再需要大量繁琐的模板化配置[3]。

2.2. Nginx 技术

Nginx [4]是一个开源的、轻量级的、高性能的 HTTP 和反向代理服务器，是俄罗斯人 Igor Sysoev 为

俄罗斯网站 Rambler.ru 开发的，具有高并发量、负载均衡、配置简单、内存消耗少、简单稳定、成本低廉、支持多系统等优点。

Nginx 采用多进程的运行方式[5]，它采用异步非阻塞与 IO 多路复用的方式来处理请求。区别于传统 Web 服务器为每个连接创建一个进程或线程，Nginx 中使用 worker 进程来处理不同的请求。Nginx 正常启动后，会首先创建一个 master 进程，之后会从 master 进程中 fork 出几个 worker 进程，master 进程作为 worker 进程“父亲”的角色。当 master 进程接收请求连接后，会将实际的处理工作交给下面的 worker 进程，master 进程只负责监督和管理工作。基本的网络事件是通过 worker 进程进行处理的。Worker 进程的数量一般要求与服务器的 CPU 核心个数保持一致，多个 worker 进程之间通过平等的竞争客户请求来获取处理连接的机会，进程之间彼此保持独立，不会出现此进程处理另一进程中请求的情况[6]。Nginx 多进程模型如图 1 所示。

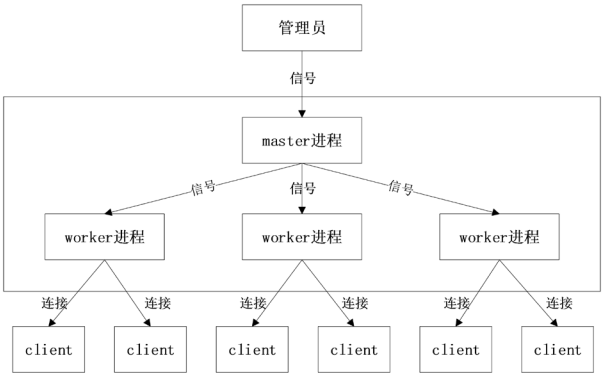

Figure 1. Nginx internal working principle
图 1. Nginx 内部工作原理

2.3. Solr/Lucene 全文检索技术

Solr 是 Apache 基金会开源搜索引擎 Lucene 下的子项目[7]，它是一个高性能、基于 Lucene (开源搜索引擎框架)的全文搜索服务器，它基于 Java 开发，具备良好的易用性、跨平台性和移植性[8]。Solr 的内部结构如图 2 所示。

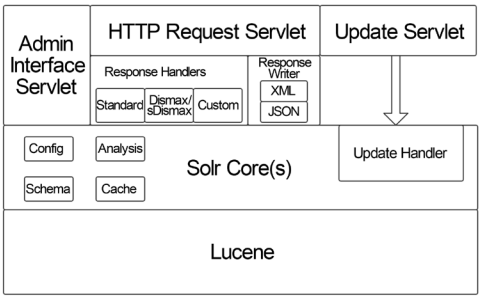

Figure 2. Solr internal structure diagram
图 2. Solr 内部结构图

Solr 底层的核心技术是通过 Lucene 实现的。Solr 专注于企业级应用，Lucene 更加专注于搜索底层的设计。Lucene 仅提供搜索服务，不提供支撑搜索服务的管理。Lucene 提供高度优化的倒排索引搜索功能[9]，它将文档按照约定的结构构建倒排表，并将索引和数据存储在文件中，文件格式被高度的优化以确保能被 Searcher 快速加载并且响应搜索[10]。Solr 与 Lucene 的关系如图 3 所示。

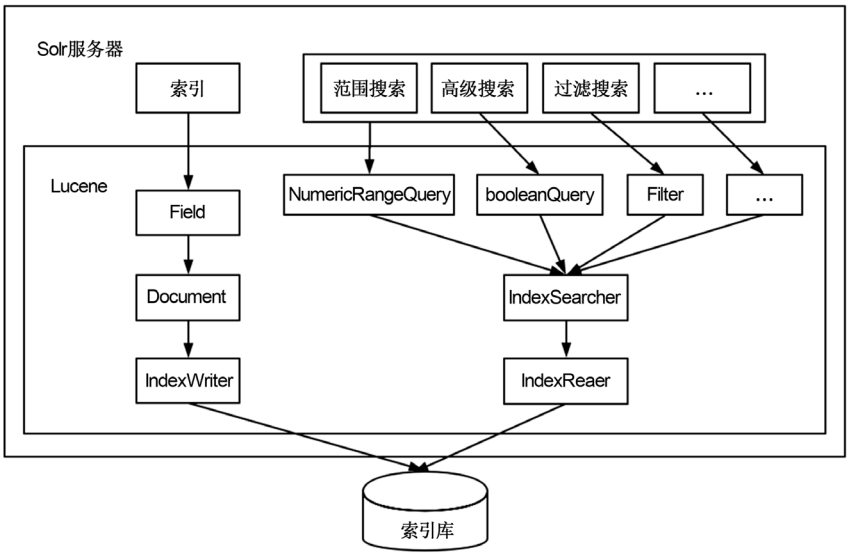

Figure 3. Relationship between Solr and Lucene
图 3. Solr 和 Lucene 的关系图

2.3.1. 全文搜索引擎工作原理

全文检索是计算机程序通过扫描文章中的每一个词，对必要的词建立一个索引，指明该词在文章中出现的次数和位置。当用户查询时根据建立的索引查找，类似于通过字典的检索字表查字的过程。

全文搜索引擎通常由五个部分组成：数据采集、数据处理、数据索引、数据搜索和用户接口[11]。数据采集指的是将要检索的内容(一般是元数据信息)保存到本地数据库中；数据处理是对数据进行预处理，生成符合存储和检索的格式；数据索引负责对上一步处理后的数据建立索引；数据搜索负责根据用户的检索关键词，查询索引库，然后将匹配的结果按照设定的排序规则进行排序反馈给用户；用户接口是指用户与搜索引擎进行交互的页面或者 API，用户可以通过接口来进行数据搜索操作，其工作原理如图 4 所示。

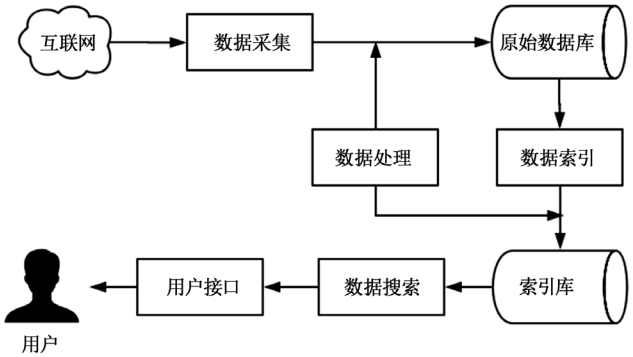

Figure 4. Full-text search engine working principle diagram
图 4. 全文搜索引擎工作原理

2.3.2. 倒排索引结构

倒排索引[12] (Inverted index)是一种索引方法，被用来存储在全文搜索下某个关键词在一个文档或者一组文档存储位置的映射。而倒排索引结构是一种以时间换空间的索引方式，是通过记录关键词和其对应的文档编号来存储数据的，即先记录关键词，然后记录包含该关键词文档的编号。其主要原理如图 5 所示。

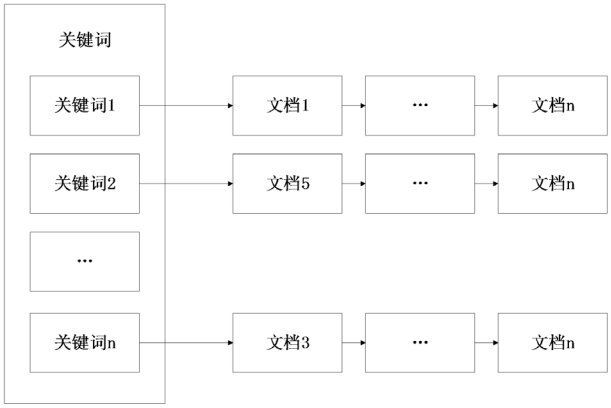

Figure 5. Inverted index structure
图 5. 倒排索引结构图

根据倒排索引的结构可以看出，全文搜索引擎采用该结构，用户通过关键词来搜索文档，只要找到关键词，然后就可以找到包含该关键词的所有文档编号。这种结构很大程度上简化了搜索文档的过程，提高查询的效率。

2.3.3. 分词技术

中文分词[13]是自然语言处理的基础和关键[14]。分词技术就是搜索引擎针对用户提交查询的关键词串进行的查询处理后根据用户的关键词串用各种匹配方法进行分词的一种技术。主要分为三类：

1) 机械式分词法：也称词典分词法，它是按照一定的策略将待分析的汉字串与一个“充分大的”机器词典中的词条进行匹配。该方法有三个要素，即分词词典、文本扫描顺序和匹配原则。

2) 基于语义的分词方法：语义分词法引入了语义分析，对自然语言自身的语言信息进行更多的处理，如扩充转移网络法、知识分词语义分析法、邻接约束法、综合匹配法、后缀分词法、特征词库法、矩阵约束法、语法分析法等。

3) 基于统计的分词法：该方法的主要思想：词是稳定的组合，因此在上下文中，相邻的字同时出现的次数越多，就越有可能构成一个词。因此字与字相邻出现的概率或频率能较好地反映成词的可信度。可以对训练文本中相邻出现的各个字的组合的频度进行统计，计算它们之间的互现信息。互现信息体现了汉字之间结合关系的紧密程度。当紧密程度高于某一个阈值时，便可以认为此字组可能构成了一个词。该方法又称为无字典分词。

目前主要的分词器[15]如表 1 所示：

Table 1. Various word breakers
表 1. 常用分词器介绍比较

分词器	内部操作步骤
Standard-Analyzer	基于复杂的语法实现词汇单元化， 这种语法规则可以识别 Email 地址、首字母缩写、字母数字等
Simple-Analyzer	在分字母字符切分文本，并将其转成为小写形式
IK-Analyzer	实现了以词典作为基础信息的正方向切分， 以及正反双向最大匹配切分，是第三方实现的分词器
ICTCLAS4J	中科院的分词器，是基于语义分词的，简化了传统分词程度的复杂度

标准的中文分词器会把中文句子的每个字都拆开，IK 分词器会先加载两个词典：扩展词典和扩展停止词典，然后对中文的分词比较准确，因此本系统采用 IK 中文分词器。

3. 设计与实现

3.1. 系统体系结构

本文设计的数字化古籍书库，支持在线快速检索和实时阅读功能。数字化古籍书库面临的主要问题是海量数据的存储和古籍图书的快速检索问题，海量数据包括古籍元数据信息和图片信息。针对这两个难点，本系统采用 Mysql 主从备份存储古籍的数据元信息，Nginx 作为静态图片服务器，采用 Solr 实现分布式索引和检索，采用 SpringBoot 搭建应用服务器，为客户端提供检索服务 API。系统整体主要分为 4 个模块：元数据存储、图片服务器、创建索引以及索引检索模块。系统总体架构如图 6 所示。

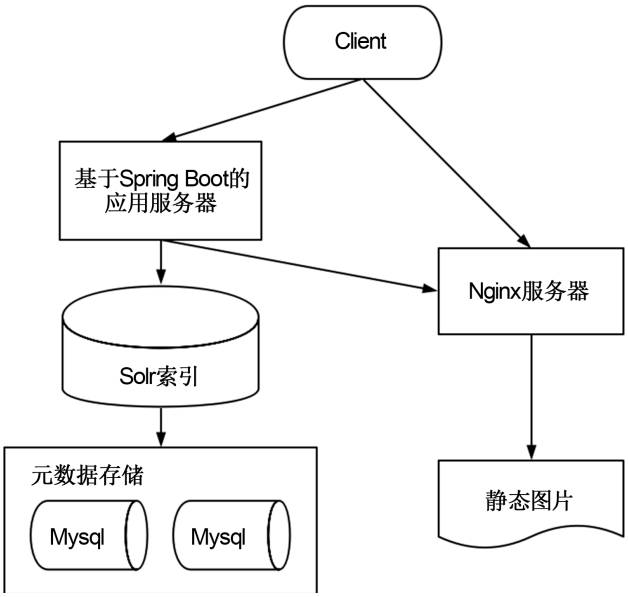

Figure 6. Digital ancient bookstore system structure
图 6. 数字化古籍书库系统结构

3.2. 数据存储

本系统中对数据库进行了详细的设计，对古籍图书进行了分类、编号，建立了完整的数据库。古籍图书资源的存储分为两种：古籍元数据信息和静态图片信息。元数据信息存储在 Mysql 数据库中，采用主从备份的方式进行存储，进行容灾备份。静态图片存储在 Nginx 服务器上，且在元数据中添加了 ImagePath 字段来保存在 Nginx 服务器中的图片路径。

3.3. 索引创建

Solr 是基于 Lucene 的全文检索引擎，其索引技术的底层实现原理和 Lucene 是一致的。当用户向 Solr 服务器提交索引数据时，Solr 服务器首先从请求参数中提取索引数据，然后进行去除停止词、分词和大小写转换等处理。Solr 服务器将处理后的索引数据封装到 Lucene 的若干个 Field 对象中，再将 Field 对象封装到 Document 中，最后调用 IndexWriter 对象的方法将索引数据存储到索引库中。流程如图 7 所示。

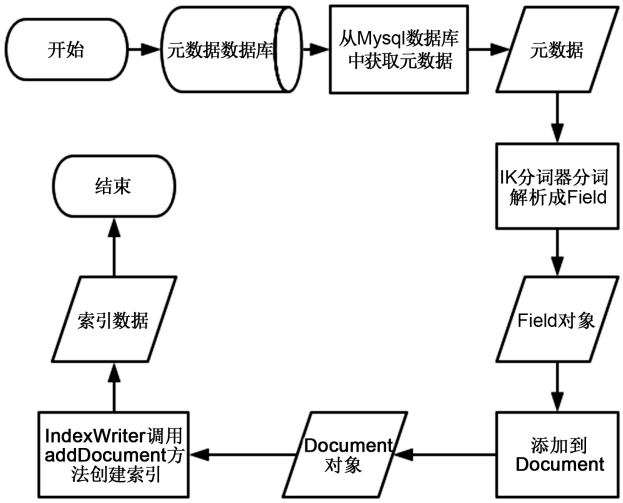

Figure 7. Solr index creation procedure
图 7. Solr 索引创建流程

IndexWriter 是 Lucene 索引过程的核心组件，通过 IndexWriter 可以创建新索引、更新索引、删除索引操作。IndexWriter 需要通过 Directory 对索引进行存储操作。IndexWriter 调用过程如图 8 所示。

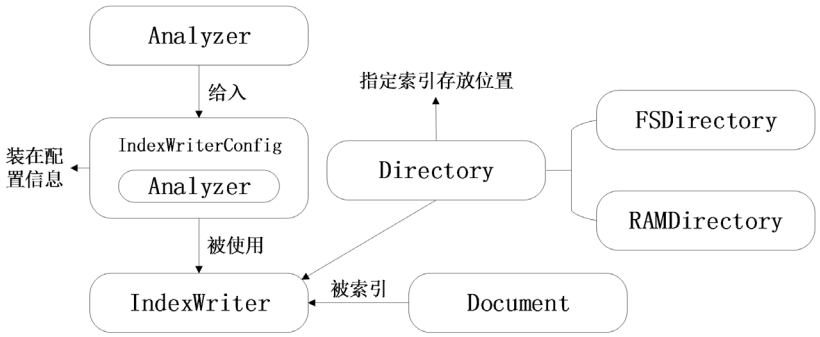

Figure 8. IndexWriter calling procedure
图 8. IndexWriter 调用过程

古籍图书的元数据信息全部储存在 Mysql 中，需要将元数据导入 Solr 创建索引，数据的导入是通过配置 Solr 的 DataImportHandler，它解决了原始数据处于不同机器以及不同数据库(Oracle 和 Mysql)的问题。具体步骤如下：

1) 创建 Solr core，core 是一个索引库，Solr 可以对多个 core 进行综合管理，每个 solr core 都可以提供索引和查询功能。

2) 修改 solrconfig.xml。该配置文件主要定义了 Solr 的一些处理规则，包括索引数据的存放位置，更新，删除，查询的一些规则配置。

3) 创建 data-config.xml。配置从指定数据中查询数据并导入索引。

4) 配置 schema.xml 文件。该文件里面主要定义了索引数据类型，索引字段等信息。

3.4. 索引检索

本系统分为全库检索、分库检索和高级检索 3 种检索方式。三种检索方式的运行流程本质上是一样的，区分在不同的过滤条件。具体流程如下：1) 客户端提交检索请求；2) Ik 分词器将搜索词进行分词解

析；3) 生成 Query 对象(采用 FacetAndHighlightQuery，实现分组统计和高亮统计功能)；4) 创建 Index-Searcher 实例，SolrCloud 的各个节点中的 Solr 节点进行分布式检索，并且将各个节点的检索结果进行合并排序，保存到结果集中；5) 将结果集的 JSON 格式数据返回给客户。检索的结果中包含古籍图书的图片路径 URL，用户可以根据图片 URL 获取图片进行展示阅读。

简单检索包括全库检索和分类别检索，如图 9 所示。

Figure 9. Simple search page
图 9. 简单检索页面

高级检索可以通过书名、作者、出版社、年代字段进行精确检索，如图 10 所示。

Figure 10. Advanced search page
图 10. 高级检索页面

3.5. 图片服务器

因为数字化古籍系统中的所有图书在检索之后都是通过图片的形式进行在线阅读，包括缩略图和原图模式，如果采用传统的搜索方式网络时延较高，并且当访问量很大时，服务器容易崩溃，响应时间长，影响用户体验。因为。采用静态资源和动态资源分离的策略，为图片建立一台单独的服务器，使用 Nginx 来管理静态资源，以此提高系统的搜索效率，提高系统响应时间，给用户带来更好地阅读体验。

Nginx 主要用来是 http 和反向代理服务器，同时 Nginx 具有良好的性能处理静态文件，并且耗费内存少。Nginx 在管理一台静态资源服务器的时候配置十分简单。只需要设置好 Nginx 监听的主机地址 IP、端口号以及对应的静态资源文件路径，再重新启动 Nginx 即可。

相关的配置如下：

```
server {
listen 80;
server_name 127.0.0.1;
location ~.* \.(gif|jpg|png|bmp)?$ {
root    [path\to\image];
autoindex on;
```

```
        }
    }
```

从 Solr 检索结果中获取到古籍图书图片的 URL 之后，可直接根据地址到 Nginx 服务器中访问到该图片，如图 11 所示。

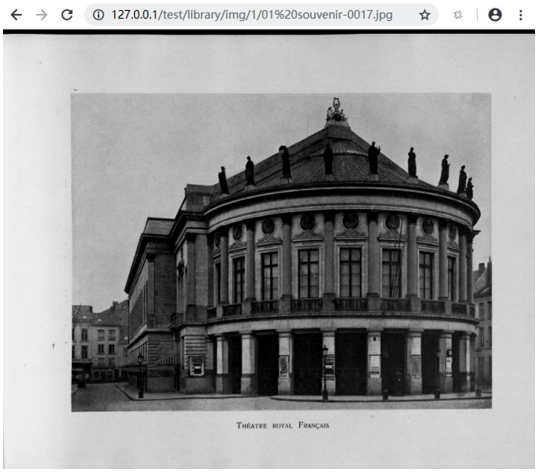

Figure 11. Access image resources via Nginx
图 11. 通过 Nginx 访问静态图片资源

4. 系统测试

4.1. 运行效果

在数字化古籍书库系统中，用户可以快速方便地根据输入的关键词以及古籍的类型进行检索古籍，并将结果根据相关度依次列出。

搜索结果页面如图 12 所示，在简单搜索中用户可以选择全库检索或者分库检索，通过 Type 字段作为过滤条件限制搜索的范围，通过模糊匹配来检索所要查阅的古籍。在高级搜索中，用户可以根据古籍的书名、作者、出版社、年代等字段来进行精确匹配检索，大大缩小检索的范围，提高检索的精确度和速度。用户在检索到了图书的基本信息之后，可以进行在线阅读。

Figure 12. Operation effect of the system
图 12. 系统运行效果图

4.2. 测试分析

本系统选取的测试数据为建筑类的古籍图书，总共约 3 万册，图片量达到约 250 万张。在这里主要对建筑类的一些常用的关键词进行检索测试，记录下每次检索的响应时间，并且与不用搜索引擎的传统 Mysql 数据库的检索时间进行对比。结果如表 2 所示。

Table 2. Comparison to searching time in different searching methods
表 2. 不同检索方式检索时间对比

关键词	Solr 检索		Mysql 检索	
	1	2	1	2
建筑	637 ms	661 ms	2382 ms	2271 ms
房屋	154 ms	147 ms	1134 ms	1125 ms
城市	485 ms	512 ms	1874 ms	1924 ms

从表中的测试结果分析可得，Solr 的检索的整体检索效率比 Mysql 数据库的传统检索效率明显高很多，尤其是在一些常用的词汇上。

5. 结束语

本文在深入研究分布式检索原理和全面剖析 Solr/Lucene 相关技术的基础上，实现了基于 Solr/Lucene 的数字化古籍书库系统，搭配 Nginx 实现静态图片服务器，实现古籍图书的快速检索和实时阅读功能。并且和传统的数据库 Mysql 数据库 SQL 的检索速度进行了对比，得到了比较好的测试效果。本文也有不足之处，在对检索结果进行排序的时候只采用了简单的相关度排序算法，在后序的开发中，将针对数字化古籍书库的检索结果的排序算法进一步的深入研究和改进。

基金项目

东南大学复杂工程系统测量与控制教育部重点实验室开放课题(MCCSE2016B01)。

参考文献

[1] 程启航. 中国民族古籍云平台及古籍 SNS 研发[D]: [硕士学位论文]. 银川: 北方民族大学, 2016.

[2] Vohra, D. (2016) Apache Solr. Practical Hadoop Ecosystem. Apress, Berkeley, CA, 349-376.

[3] 王永和, 张劲松, 邓安明, 等. Spring Boot 研究和应用[J]. 信息通信, 2016(10): 91-94.

[4] 张云, 许江淳, 李玉惠, 等. 基于 Nginx 服务器负载均衡技术的研究与改进[J]. 软件, 2017, 38(8): 6-12.

[5] Zhou, B., Xia, X., Lo, D., *et al.* (2014) Build Predictor: More Accurate Missed Dependency Prediction in Build Configuration Files. 2014 *IEEE* 38*th Annual Computer Software and Applications Conference* (*COMPSAC*), Vasteras, 21-25 July 2014, 53-58. https://doi.org/10.1109/COMPSAC.2014.12

[6] 杜星. 轻量级 Web 服务器 Nginx 的理论与技术研究[D]: [硕士学位论文]. 南京: 南京邮电大学, 2016.

[7] 孙毅芳. 基于数据挖掘的图书馆推荐系统的设计与实现[D]: [硕士学位论文]. 济南: 山东大学, 2017.

[8] 贾贺, 艾中良, 贾高峰, 等. 基于 Solr 的司法大数据检索模型研究与实现[J]. 计算机工程与应用, 2017, 53(20): 249-253.

[9] Sajja, K. (2007) Performance Study of Lucene in Parallel and Distributed Environments. Boise State University, Boise, Idaho, US.

[10] 邱宇芳. 基于 SolrCloud 大数据平台日志管理系统的设计与实现[D]: [硕士学位论文]. 北京: 中国科学院大学(中国科学院工程管理与信息技术学院), 2017.

[11] 赵亮. 基于 Solr 的企业搜索引擎研究与实现[D]: [硕士学位论文]. 北京: 中国地质大学(北京), 2017.

[12] Mogotsi, I.C., Manning, C.D., Raghavan, P. and Schütze, H. (2010) Introduction to Information Retrieval. *Information*

Retrieval, **13**, 192-195. https://doi.org/10.1007/s10791-009-9115-y

[13] 兰冲. 基于统计规则的中文分词研究[D]: [硕士学位论文]. 西安: 西安电子科技大学, 2011.

[14] 韩冬煦, 常宝宝. 中文分词模型的领域适应性方法[J]. 计算机学报, 2015, 38(2): 272-281.

[15] 韩云辉. 基于 Lucene 的数字版权资源库的构建与应用研究[D]: [硕士学位论文]. 北京: 北方工业大学, 2013.