基于服务器集群的云检索系统的研究与示范

安俊秀
（成都信息工程学院软件工程学院 成都 610225）

摘 要 在研究云计算及移动搜索引擎的基础上，依据当前技术发展，提出了基于服务器集群的云检索系统模型，该模型由云信息层、云检索集群系统和用户查询框组成。对云检索集群系统中数据存储技术进行了深入研究，提出了分布式云检索数据存储方案。为了提高云检索执行效率，提出了以程序流为核心的云检索软件执行模式。该模型的测试结果表明，系统功能能正确实现，性能表现较好且稳定。通过该模型的示范，给海量信息检索技术提供了拓展思维的方案。

关键词 云计算，云检索，程序流，分布式文件系统，集群
中图法分类号 TP391 文献标识码 A

Research and Demonstration of Cloud Retrieval System Based on Server Clusters

AN Jun-xiu
(School of Software Engineering, Chengdu University of Information Technology, Chengdu 610225, China)

Abstract Based on the study of cloud computing and mobile search engine, according to the development of current technology, we put forward the cloud retrieval system model based on clusters. The model consists of the cloud information layer, cloud retrieval cluster system, the user query box. We deeply studied distributed data storage technology in cloud retrieval system, and proposed program of cloud storage and retrieval of data structure. In order to improve the efficiency of parallel execution in cloud retrieval, put forward a program flow at the core of cloud retrieval software implementation of the model. The testing results of the model are shown that the system function can realize correctly, and its performance is good and steady. The demonstration of this model gives the retrieval technology of massive information a scheme to develop thinking.

Keywords Cloud computing, Cloud retrieval, Program flow, Distributed file system, Cluster

1 引言

Google 于 2006 年秋提出了“云计算(cloud computing)”概念。从表象上看，云计算的特点有 3 个：一是后台对用户透明，云计算系统为用户提供的是服务，服务的实现机制对用户透明，用户无需了解云计算的具体实现机制，就可以获得需要的服务；第二是接口简单，用户可以通过简单的接口体验云计算服务，云计算提供的统一简单接口可使程序开发变得更加容易；第三是客户端只需安装浏览器即可，客户端可以通过在浏览器中直接编辑存储在“云”的另一端的文档，随时与朋友分享信息。因此，从表象上看，任何一种基于互联网的服务都可以称之为云。但从本质上看，Google 提出的云计算关心的是后台服务的架构，即对用户透明的部分。经过多年研究发现，从本质上来说，云计算应有两个本质的突出特点。一是更高效的海量信息存储方式，云计算采用分布式存储的方式来存储数据，采用冗余存储的方式来保证存储数据的可靠性，即为同一份数据存储多个副本；二是软件执行模式采用并行执行的方式，云计算系统需要同时满足大量用户的需求，并行地为大量用户提供服务。从这两方面来说，很多称为云计算的公司只能是一种占位思想。

2 基于服务器集群的云检索系统模式

在深入研究对云计算、移动搜索引擎的基础上，提出了基于服务器集群的云检索系统架构模型，如图 1 所示。该模型由云信息层、云检索集群系统和用户查询框 3 部分组成。云检索集群系统由云采集层、云加工层、云索引层、云查询层、云接口层、数据存储云层、云检索监控系统、云管理和调度系统组成。基于服务器集群的云检索系统专注于海量数据的分布式存储和软件执行模式技术的突破。

基于服务器集群的云检索系统根据需要存储信息的类型采用了分布式文件系统及分布式数据库存储技术，将分布式应用部署到大型廉价集群上，从而实现海量信息的存储。为了便于管理及实现，分布式文件系统及分布式数据库均采用主/从架构。它实现了动态扩容、高效率的并行执行及数据的高可靠性，为用户的准确查找提供了强大的后台支持。

为了提高检索效率，各层之间实行分布式执行，每个层内部的功能模块实行并行执行，这个工作调度由云管理和调度系统来完成。云管理和调度系统就像管家程序，协调处理云

到稿日期：2009-08-21 返修日期：2009-10-21 本文受四川省教育厅旅游专项课题(LY09-01)，四川省青年科学基金(09ZQ026-068)资助。
安俊秀(1970—)，女，硕士，副教授，主要研究方向为海量信息检索、移动计算及并行计算，E-mail：anjunxiu@cuit.edu.cn。

检索的各项工作。云监控软件监控整个云检索系统，保证系统中的不同工作能够“和睦相处”，如监视是否有来源于单个IP地址过分密集的访问等工作。只有在这样一个云检索的框架下，才能实现对海量信息搜索的规模、速度、准确性、全面性和其他方面的改进。

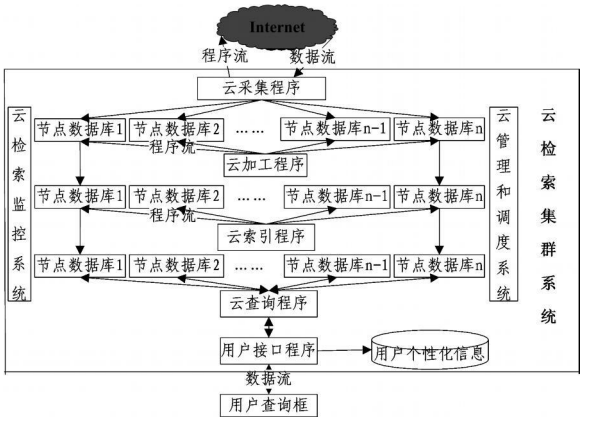

图1　云检索系统架构

基于服务器集群的云检索系统采取了以程序流为核心的基本理念。云采集层把Internet中的海量数据经过网络机器人转化成数据流存储到多个节点数据文件中；接着云加工和云索引程序转化为程序流，流入到各个节点数据库中，并行处理多个节点文件中的数据，生成数据库文件存储在相应的节点上；用户输入查询信息，经过用户接口程序及云查询程序的处理，找到相关信息后转化成数据流反馈给用户。

由于在集群系统中采用程序流思想降低了由海量数据流动对检索系统执行效率产生的负面影响，使得操作处理速度大大提升；减少了数据在处理时的流动，从而增加了数据的安全性与完整性。

3　分布式数据存储设计

在云检索集群系统中，海量数据的组织特点是不一样的。在云采集层通过网络机器人从Internet上爬取下来的信息具有文件系统的格式特点(按页存储，数据间没有格式)，因此对本层的信息采用了分布式文件系统存储方案。在云加工层及云索引层对文件内容进行了处理，文件内容有了一定的格式，具有了分布式数据库的特点，因此在这两层中采用了分布式数据库系统存储方案。

3.1　分布式文件系统存储设计

在云采集层中采用了分布式文件存储信息方案。分布式文件系统的典型代表有HDFS(Hadoop Distributed File System)和GFS(Google File System)两类。分布式文件系统可由若干个集群组成，集群与集群之间保持独立性。每个集群均采用主/从架构，包含若干个服务器(又称为节点)，即在每个集群内都由一个主服务器节点和多个数据服务器节点组成。分布式文件系统均采用3层架构，由主服务器、数据服务器和客户端组成，如图2所示。客户端通过主服务器节点向数据服务器发送数据请求，而数据服务器则将取得的数据直接返回给客户端。

主服务器节点需要选择本集群内性能好、存储容量大、带宽高的服务器节点作为主服务器节点，用它来管理该集群内的其它数据服务器节点的信息存储、调度和客户端对文件的访问，实现元数据及文件到数据之间的映射关系及存储位置。主服务器节点映射一个文件到一批的页，映射页到数据服务器节点上。因此在主服务器节点内存中需要保存文件列表、每一个文件的页列表、每一个数据服务器节点中页的列表和文件属性等数据。

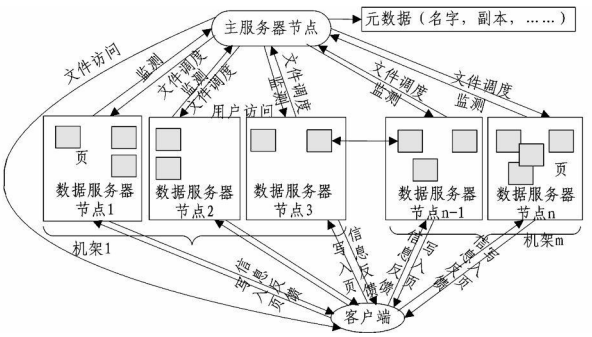

图2　分布式文件系统架构

数据服务器节点在集群中一般是一个节点一个数据服务器，负责管理节点上的数据存储。在数据服务器节点内部，采用了操作系统中文件管理的概念，一个文件被分成多个大小相等的页，典型的页的大小为64MB，即将每个文件存储成页序列。数据服务器节点在主服务器节点的指挥下进行页的创建、删除和复制。为了保证数据的可靠性，所有页都需建立两个以上的副本。主服务器节点周期性地监测集群中的每个数据服务器节点是否正常工作并监测该数据服务器节点上的所有页组成的列表。在集群中数据服务器节点失效是正常的，因此在分布式文件系统中要保存两个副本，检测失败和快速恢复数据是主服务器节点的核心任务，数据服务器节点的死亡可能引起一些页的副本数量低于预定值，主服务器节点会不断跟踪需要复制的页，在需要时启动复制。

客户端要存储数据时，从主服务器节点获取存储页的数据服务器节点位置列表，客户端发送页到第一个数据服务器节点上，第一个数据服务器节点收到数据后通过管道流的方式把页发送到另外的数据服务器节点上。当页被所有节点写入后，客户端继续发送下一个页。对于页的放置位置采用的是一个放在当前的数据服务器节点上，一个放在远程的机架上的一个数据服务器节点上，一个放在相同机架上的一个数据服务器节点上。客户端选择最近的一个节点读取数据。

3.2　分布式数据库系统存储设计

通过云加工层程序和云索引程序处理后的文件内容具有一定的格式和属性，因此对信息的存储采用的是分布式数据库存储。分布式数据库系统的典型代表有HBase，Hypertable，Amoeba和Bigtable。Hypertable与HBase的工作原理一样，均采用了Google BigTable的稀疏的、面向列的数据库实现方式理论，只是Hypertable用C++写成，HBase用Java写成。Java项目在设计模式和文档上一般都比C++项目好，非常适合开源项目。C++的优势在于性能和内存的使用上。Hbase和Hypertable提供了类似于BigTable的可伸缩数据库实现。HBase和Hypertable是两个开源项目，Google的BigTable不开源。它们主要解决的都是数据的组织和存储策略问题。

云加工层程序处理后的文件内容是正排索引文件，云索引程序处理后的文件内容是倒排索引文件，两者都将文件内容切分成以词为单元的格式。本文在深入研究分布式数据库

的基础上，采用了列方式存储数据，不需要给出像关系数据库中关键字那样丰富的关系属性。在分布式数据库中，数据在存储前要经过排序和压缩，按流字符串来存储数据。分布式数据库是以分布式文件为基础的，因此分布式数据库的架构类似于分布式文件，采用的是主/从结构。分布式数据库中主服务器负责管理所有的区域数据服务器，但主服务器本身不存储任何数据。分布式数据库逻辑上的 Table 被定义为一个区域，存储在某一台区域数据服务器上，区域数据服务器与区域的对应关系是一对多的关系，如图 3 所示。

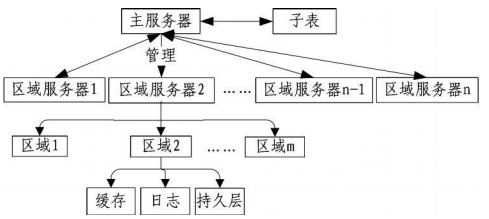

图 3　分布式数据库系统架构

客户端通过主服务器获得区域所在的区域数据服务器，然后就直接与区域数据服务器进行交互。区域数据服务器之间不通信，只和主服务器进行通信，区域数据服务器受主服务器的监控和管理。

每个区域在物理上可分为 3 部分：缓存、日志和持久层。缓存是为了提高效率而在内存中建立的，保证了部分最近操作过的数据能够快速的被读取和修改，日志是作为同步缓存和持久层的事务日志，在区域数据服务器周期性地发起清洗高速缓存命令的时候，将缓存中的数据持久化到持久层中，同时清空缓存中的数据。在读取区域信息的时候，优先读取缓存中的内容，如果未取到再去读持久层中的数据。

云检索系统采用分布式数据存储机制，可以保证数据的有效存储和组织，为应用提供高效和可靠的访问接口，保持良好的伸缩性和可扩展性，并且可以解决云检索系统需要处理大量信息的问题。

4　以程序流为核心的云检索软件执行模式设计

随着海量数据的不断增加，云检索要处理的数据迅猛膨胀，采用数据流运行模式让庞大的数据在网络间流动已造成诸多瓶颈及系统运行效率低下等问题，也就是说采用数据流的工作方式只适用于小规模数据。为了突破这种现状，在云检索系统中提出了采用以程序流为核心的软件执行模式，即让较小的程序代码段在网络间流动来处理各个节点的庞大数据块，从而解决了系统运行过程中数据传输瓶颈，最终实现处理数据本地化，大大提高了用户检索效率。

4.1　程序流的概念引入

云检索系统关心的是后台庞大的数据处理问题，为了提高程序执行效率与系统的性能，仅仅采用并行执行方式是不能满足海量数据处理的。以前由于程序要处理的数据量少，采用的是数据流方式，即数据处理采用的是网络间数据流动的方式，但随着数据的迅猛增加，数据流方式具有耗时长、数据易流失的缺点，并且庞大的数据流动造成了严重的 I/O 瓶颈。如果将庞大的数据存储在节点数据库，让较小的程序流在网络间流动执行，那么各个节点就能够快速地为用户提供数据。

在大型和复杂的软件系统开发过程中，数据流图（Data Flow Graph，简称 DFD）用来描绘数据流在系统中的流动和处理情况，是系统的逻辑模型。它以图形的方式刻画数据流从输入到输出的移动变换过程，是结构化系统分析方法的主要表达工具及用于表示软件模型的一种图示方法。而程序流用来描述程序在系统中的流动及对数据处理的情况，它的基本思想是通过让较小的程序代码段在网络间流动来处理海量数据信息，从而实现高速处理海量信息的目的。不管是数据流还是程序流，都运用了流的最基本概念，即站点能够为用户提供速度足够快的数据和程序，以致用户或网络节点感觉到程序或数据似乎在本地机器上运行一样，图 4 反映了采用数据流及程序流结合来完成云检索的基本思想。

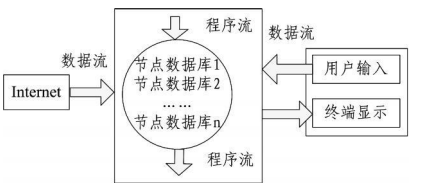

图 4　网络间数据流及程序流的基本流程

从图 4 中可以看出，除了从 Internet 上爬取信息存储到节点服务器上和与用户间的交互使用的是数据流外，信息在节点数据文件中的处理运用的是程序流，多股程序流同时流向节点数据库中的海量信息并进行并行处理。

4.2　程序流执行模式及测试结果

以程序流为核心的云检索系统在理论上是能够提高处理速度的，但是在实际操作中是否如此有待测试验证。系统测试主要用对比的测试方法，对比数据流与程序流的执行效率，因此需要同时测试数据流与程序流。在测试工作中，采用大量的测试数据，对其进行了严格的测试。数据流的测试是通过云检索调度系统把待处理的数据传递给节点服务器，节点服务器上的程序对数据进行处理，处理完成后向调度系统发送结束标识，云检索调度系统通过调度开始与结束的时间计算出数据流执行完成的总时间。程序流的测试是把待处理的数据存放到节点服务器中，接着云检索调度系统把要进行处理的程序传递给节点服务器，节点服务器接收到程序后马上对数据进行处理，处理完成后同样也向调度系统发送结束标识，这时计算出程序流执行完成的总时间，与数据流执行完成时间进行对比。

测试用例是在 Eclipse 集成开发环境下采用 Java 语言编写的，它在 4 核心处理器 2G 内存的服务器上采用多线程方式提供服务的部署下完成。为了保证可比性，测试中使用了一样的数据与处理程序，以下是部分程序代码。

调度系统中的核心代码：

```
Socket cs=s.accept();
OutputStream out=cs.getOutputStream();
DataOutputStream dos=new DataOutputStream(out);
FileReader is=new FileReader("word.txt");
//word.txt:存放要处理的数据
BufferedReader bs=new BufferedReader(is);
//建立数据文件的文件缓冲流
String str;
while((str=bs.readLine())!=null)
//向节点服务器发送数据信息
{
dos.writeUTF(str);
```

```
}
dos.writeUTF("0000"); //发送数据传递结束标识
bs.close();
InputStream in = cs.getInputStream();
//接收节点服务器传递的结束标识
DataInputStream din= new DataInputStream(in);
//如果节点服务器传来结束标识"stop",则计算出程序的执行时间
if((str=(din.readUTF())).equals("stop"))
{
t= System.currentTimeMillis()-t;
System.out.println("程序所用时间为:"+t);
}
```

节点服务器中的核心代码:

```
//clientword.txt:节点服务器用来存储调度系统传递来的要处理的数据
BufferedWriter bs= new BufferedWriter(new FileWriter("clientword.txt"));
InputStream is= s.getInputStream();
//接收调度系统传来的待处理数据
DataInputStream ds=new DataInputStream(is);
OutputStream os= s.getOutputStream();
DataOutputStream din= new DataOutputStream(os);
//把调度系统传递来的数据保存在节点服务器中
while(true)
{
str= ds.readUTF();
if(str.equals("0000"))
//接收到数据传递结束标识就退出读取数据操作
break;
System.out.println(str);
bs.write(str);
}
```

测试结果如图 5 和图 6 所示。

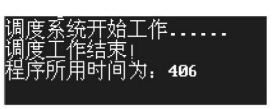

图 5 数据流测试结果

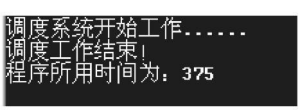

图 6 程序流测试结果

测试结果表明,系统可以正确实现,性能表现较好且稳定,并且程序流执行效率高于数据流程序的执行效率。系统运行一段时间,通过日志监测表明,系统没有发现异常错误,也没有收到任何错误报告。

结束语 本文提出了云检索在程序并行化执行模式上的改进方案——采用程序流并行执行模式,并给出云检索核心技术分布式文件系统及分布式数据库系统的架构模式。建立云检索系统理论基础及架构方案,为实现集群系统分布式数据及海量数据的检索处理提供了新的思路。

参 考 文 献

[1] 初晓博, 秦宇. 一种基于可信计算的分布式使用控制系统[J]. 计算机学报, 2010(1): 93-102

[2] 李卫疆, 赵铁军, 王宪刚. 基于上下文的查询扩展[J]. 计算机研究与发展, 2010, 47(2): 300-304

[3] 付雄, 王汝传, 邓松. 无线传感器网络中一种能量有效的数据存储方法[J]. 计算机研究与发展, 2009, 46(12): 2111-2116

[4] Hsieh J W, Kuo T W, Chang L P. Efficient Identification of Hot Data for Flash Memory Storage Systems[J]. ACM Transactions on Storage, 2006, 2(1): 22-40

[5] 杨代庆, 张智雄. 基于 Hadoop 的海量共现矩阵生成方法[J]. 现代图书情报技术, 2009(4): 23-26

[6] 刘立坤, 武永卫, 徐鹏志, 等. CorsairFS: 一种面向校园网的分布式文件系统[J]. 西安交通大学学报, 2009, 43(8): 43-47

[7] 张刚, 谭建龙. 分布式信息检索中文档集合划分问题的评价[J]. 软件学报, 2008, 19(1): 136-143

[8] 张海军, 史树敏, 朱朝勇, 等. 中文新词识别技术综述[J]. 计算机科学, 2010, 37(3): 6-10

[9] 李小龙, 林亚平, 胡玉鹏, 等. 基于分组的分布式节点调度覆盖算法[J]. 计算机研究与发展, 2008, 45(1): 180-188

[10] 张刚, 刘悦, 郭嘉丰, 等. 一种层次化的检索结果聚类方法[J]. 计算机研究与发展, 2008, 45(3): 542-547

[11] 侯东风, 刘青宝, 张维明, 等. 一种适应性的流式数据聚集计算方法[J]. 计算机科学, 2010, 37(3): 152-155

[12] 马亮, 陈群秀, 蔡莲红. 一种改进的自适应文本信息过滤模型[J]. 计算机研究与发展, 2005, 42(1): 79-84

[13] 王鹏, 陈高云, 安俊秀, 等. 移动搜索引擎原理与实践[M]. 北京: 机械工业出版社, 2009

[14] 郎皓, 王斌, 李锦涛 等. 文本检索的查询性能预测[J]. 软件学报, 2008, 19(2): 291-300

[15] 黄瑞, 史忠植. 一种新的 Web 异构语义信息搜索方法[J]. 计算机研究与发展, 2008, 45(8): 1338-1345

[16] 陈全, 邓倩妮. 云计算及其关键技术[J]. 计算机应用, 2009, 29(9): 2562-2567

[17] 叶育鑫, 欧阳丹彤. 语义 Web 搜索技术研究进展[J]. 计算机科学, 2010, 37(1): 1-5

[18] 洪亮, 卢炎生, 陈锦富, 等. 一种基于位置数据库聚类的动态适应缓存位置信息策略[J]. 计算机研究与发展, 2008, 45(7): 1203-1210

[19] 田晓珍, 尚冬娟. Web 个性化服务[J]. 重庆工学院学报: 自然科学版, 2008, 22(7): 76-80

(上接第 140 页)

[2] 张世琨, 张文娟, 常欣. 基于软件体系结构的可复用构件制作和组装[J]. 软件学报, 2001, 12(9): 1351-1359

[3] 任洪敏, 张敬周, 钱乐秋. 面向体系结构的构件接口模型及其形式化规约[J]. 计算机工程, 2005, 12: 67-69

[4] Reussner R H. Enhanced Component Interfaces to Support Dynamic Adaption and Extension[C] //Proceedings of the 34th Hawaii International Conference on System Sciences. 2001

[5] 卢炎生, 查虎平, 徐丽萍. PCCM: 具有性能约束的构件模型[J]. 计算机科学, 2004, 3: 89-92

[6] Whittle B, Ratcliffe M. Software component interface description for reuse[J]. Software Engineering Journal, 1993, 1(8): 307-318

[7] Han Jun. Temporal logic based specification of component interaction protocols[C] //Proceedings of the 2nd Workshop of Object Interoperability at ECOOP 2000. Cannes, France, June 2000

[8] Han Jun. A Comprehensive Interface Definition Framework for Software Components[C] //Proceedings of the 1998 Asia-Pacific Software Engineering Conference. IEEE Computer Society, Taipei, Taiwan, December 1998: 110-117

[9] Berger M. Towards Abstractions for Distributed Systems[D]. Imperial College, Department of computing, 2002