倒排文件页式存储方法①

时亚南, 束文杰, 于国欣

(新疆维吾尔自治区 特种设备检验研究院, 乌鲁木齐 830011)

摘 要: 页是磁盘与内存进行数据交换的基本单位, 它在操作系统、数据库管理系统以及倒排文件的数据组织中占据十分重要的地位. 为减少倒排索引的磁盘 I/O 读写开销, 提出了一种倒排文件按页存储的构建方法, 实现了按页读写文件. 该方法主要包括磁盘 I/O 层设计、页管理器设计以及堆文件管理器设计三个部分, 实现了页大小可变的分块式数据文件管理, 支持页内定长记录、变长记录的组装以及超长数据记录的跨页存储. 经实验测试, 结果表明该方法是行之有效的, 可以将其应用到实际的垂直搜索引擎中.

关键词: 倒排文件; 按页存储; 磁盘 I/O 层; 堆文件管理器; 记录

Page Storage Method of Inverted File

SHI Ya-Nan, SHU Wen-Jie, YU Guo-Xin

(Inspection Institute of Special Equipment, Xinjiang Uygur Autonomous Region, Urumqi 830011, China)

Abstract: Page is the basic unit of data exchange between disk and memory, in operating systems, database management systems, and inverted file's data organization, it occupies a very important position. To reduce the inverted index's disk I/O read and write overhead, proposing a method that the inverted file storages by pages, and achieving to read and write files by page. This method mainly contains three parts, including disk I/O layer design, page manager design, and heap file manager design, achieving variable page size's data file management using block, supporting for the fixed-length records, variable-length records storage assembly in the page and super long data record's cross-page storage. The experimental test results show that the method is effective, and it can be applied to actual vertical search engine.

Key words: inverted file; storage by pages; disk I/O layer; heap file manager; records

随着互联网技术的持续快速发展, 人类社会进入了一个前所未有的信息化时代, 那就是大数据时代. 在大数据时代, 人们掌握的数据在以爆炸性的速度增长, 并且数据的形态也在发生着根本性的变化. 据统计, 目前互联网上 80%以上的数据都是非结构化数据. 因此, 研究如何处理大规模非结构化数据, 成为解决人们在大数据时代如何快速获取有效信息的必由之路.

一方面, 倒排索引在非结构化数据处理方面具有无可比拟的优势;另一方面, 操作系统为了能够管理大容量的二级存储设备, 通常把若干个扇区组织为一个块(BLOCK), 以块(页)为单位编址和访问外存, 因此倒排文件的 I/O 设计很显然必须考虑二级存储设备的这一访问特征, 以便提高磁盘的读写效率. 综合以上两点考虑, 研究倒排索引的页式存储方法具有十分重要的理论意义和实践价值.

页主要用来存放记录, 它是记录在磁盘上存储的单位, 同时也是磁盘与缓存池交换数据的单位. 文献[1-5]详细讨论了辅助存储设备、文件组织、存储结构及存储系统的设计细节, 文献[6]介绍了记录组装格式及页的结构, 文献[7-14]分析了倒排索引结构的组织技术、存储模式及实现方法, 认为对倒排表的组织方式进行一定的优化, 按块存储倒排索引能显著提升检索的整体性能. 本文在充分借鉴前人的研究基础上, 提出并实现了一种新的倒排索引存储方法—页式存储

① 基金项目:新疆维吾尔自治区科技攻关项目(200931103)

收稿时间:2014-09-01;收到修改稿时间:2014-09-29

方法, 并将其应用到实际搜索引擎中. 该页式存储方法实现主要有三部分构成, 即磁盘 I/O 管理器模块、页管理器模块以及堆文件管理器模块. 下面对页式存储方法的实现进行详细的阐述.

1 磁盘I/O层设计

1.1 二级存储的 I/O 特征

扇区是磁盘 I/O 的最小单位, 根本原因是磁盘的数据校验以扇区为单位. 早期的磁盘一个扇区(Sector)通常存储 512 字节数据, 随着磁盘容量的增大, 很多厂家开始生产支持 Advanced Format 的 4K 扇区磁盘. 随着闪存(Flash Memory)技术的发展, 采用 NAND Flash 结构的存储随处可见, 如 Flash 存储卡、Flash USB 盘、Flash 移动盘及固态盘(SSD), NAND Flash 是嵌入式系统和移动系统的首选存储, 尽管它没有磁盘特有的物理结构(扇区、磁道、柱面、马达、磁头), 但 NAND Flash 通常内含控制器逻辑, 以便为上层应用提供基于块的I/O接口, 因此, 操作系统仍然把NAND Flash 存储当作和磁盘相同的块设备对待.

而倒排索引最终必须保存在二级存储中, 通过比较各类二级存储设备的访问模式、性能以及读写特点, 最终选择磁盘作为倒排索引底层文件的存储结构, 选择跨平台编程语言Java设计并实现了磁盘I/O管理器模块. 为了提高辅存 I/O 操作的效率, 结合磁盘、FLASH 存储的读写特点及 Linux 操作系统的底层 I/O 特性, 本研究设计的磁盘 I/O 管理器以页为单位进行读写操作, 页大小可以为操作系统页的 2n 倍(n 为正整数).

1.2 磁盘 I/O 层的功能

磁盘 I/O 层处于存储系统的最底层(即文件管理器模块), 它定义了倒排文件在磁盘上的组织方式、存储结构和访问模式, 同时封装了多个操作系统中对文件的各种操作, 并支持用户自定义页大小(从文件系统页大小到 64KB). 它为上层提供按页读写文件操作, 实现数据在磁盘和内存之间永久存储和瞬时状态切换, 该层实现的核心是如何保证数据的读写操作快.

本文在实现过程中使用了 Java 底层实现的文件内存映射 File Channel 读写数据, 经测试, 它读写数据的能力比使用顺序读写流或随机读写流方式读写数据快 16 倍左右, 快的主要原因在于它不需要对文件执行 I/O 操作, 也不用为文件申请并分配缓存, 取消了将文件数据加载到内存、数据从内存到文件的回写以及释放内存块等步骤, 其所有的文件缓存操作均通过 Java 虚拟机直接管理.

2 页管理器设计

处理 I/O 操作需要了解页的格式问题, 倒排文件的上层把数据看成记录的集合. 页是由一组插槽组成, 一个插槽对应一条记录, 这样记录就通过插槽在页中有效的组织在一起. 在页管理器中, 我们将页号和插槽号作为记录索引(即记录 id), 唯一标识一条记录.

该模块的目的是实现倒排索引的各种数据页描述和其函数. 在倒排文件中, 一共有四种页:基本页、文件头页、文件目录页和数据页. 其中, 基本页存储本页的页号和下一页的页号. 文件头页用来存储文件验证信息(头字符串)和文件中每个数据页的页号和其对应的剩余空间, 它的下一页页号指向的是该文件中的目录页. 文件目录页用来存储该文件中所有数据页的页号和其对应的剩余空间, 并用下一页的页号将所有目录页链起来. 数据页用来存储数据记录, 每个记录占用一个槽, 槽中存储该记录在该页中的偏移量和长度, 若一个数据页存放不下一条记录, 则通过下一页页号形成一个数据页链.

2.1 对定长记录的操作

在数据页中, 如果页上所有记录长度相同, 那么记录槽可以是统一的, 并且在页内连续存放. 记录的操作包括插入、删除、检索等. 当一个记录被插入一页, 必须找到一个空插槽并把记录存入, 难点在于如何跟踪空插槽以及如何定位页上所有的记录. 下面重点介绍一下插入操作的两种实现方法:

第一种方法是选择在前面的 N 个插槽中存放记录, 无论什么时候记录被删除, 都把页上最后的记录移到删除后空下的插槽. 这种格式可以通过简单的偏移量计算来定位页上的第 I 个记录, 并且所有的空插槽都出现在页的尾端.

第二种方法是通过使用一个比特位的数组来处理记录的删除, 每个槽对应一比特位, 用于跟踪空闲插槽信息, 在页上定位记录需要扫描比特位数组, 找到比特位为 1 的插槽, 当记录被删除时, 它的比特位设置为 0.

2.2 对变长记录的操作

在数据页中, 还需要实现可变长记录的插入、删除和更新操作. 针对变长记录的插槽式页组织也可以

用于定长记录. 将每条记录的实际字节序列长度单独存储在记录插槽数组中, 而记录本身的字节序列存储在磁盘块指定的位置.

(1)在插入一条可变长记录时, 首先检查可变长记录的合法性, 然后查看是否有可用的记录槽, 若有可用的记录槽则直接将记录中的数据拷贝到数据页中再设置记录槽中的偏移量 offset 和长度 length, 若没有可用的记录槽则先在数据页中创建一个新的记录槽然后再将记录中的数据拷贝到数据页中并设置记录槽中的偏移量和长度. 在记录槽中, offset 代表该条数据记录在数据页中的起始位置, 而 length 表示该条数据记录的长度.

(2)在删除一条可变长记录时, 首先检查记录编号 RID 的合法性, 然后检查该数据页中是否有这条记录, 删除记录标识为 rid 的记录, 并且需要压缩删除该记录后留下的页内碎片, 在压缩页内碎片时, 页内碎片后面的所有记录的偏移量都要前移碎片大小的位置, 然而在删除数据槽中间的记录时也会留下一个较小的页内碎片, 只有在删除最后一个数据槽所对应的记录时, 数据槽数组方可被压缩. 如果没有发生任何错误则返回 OK, 如果 rid 无效(即磁盘文件中没有该物理页或者页中没有该条记录)则返回 INVALIDRID.

(3)在更新一条可变长记录时, 首先检查记录编号(RID)的合法性, 然后检查该数据页中是否有这条记录, 更新 RID 为 rid 的记录, 若新纪录的长度和老记录的长度相同则直接用新记录的数据替换老记录的数据, 若新纪录的长度大于老记录的长度则将后面的记录数据后移并修改后面记录的插槽数组, 若新纪录的长度小于老记录的长度则将后面的记录数据前移并修改后面记录的插槽数组.

3 堆文件管理器设计

倒排文件通常在外存中以堆文件方式持久化存储数据. 堆文件管理器的主要作用是封装数据记录和数据类型模块、数页管理器模块、磁盘 IO 模块. 它能提供针对倒排文件中的记录扫描、过滤、插入、修改和删除等功能.

为了实现可变长记录的查找、插入、删除和修改功能, 堆文件管理器通过文件头页和文件目录页来管理堆文件中的所有数据页, 在文件头页和文件目录页中构建一个记录该堆文件中所有数据页的页号和其对应的剩余空间的映射关系.

插入记录操作的实现过程是:首先扫描一下目录页, 如果目录页中存在页可以将记录存储下, 将记录插入到那一页. 如果目录页中没有能够将该记录存储的下的页, 那么在磁盘上新分配一个新页, 然后将记录插入到新页中, 并将该页的 pageNo 和 freespace 信息加入到目录页中. 特别的, 在新分配页之前, 要检查一下目录页是不是已经满了, 如果目录页已经满了, 新增目录页, 并将新增的目录页加入到目录页集合中. 另外, 如果一条记录超过一页的大小, 在一页中存储不下时, 要考虑跨页存储, 需将记录进行分割分别写入不同的新页中.

更新记录操作的实现过程是:首先对非跨页记录直接调用页管理器部分的 updateRecord 函数对记录进行更新, 而针对跨页存储的记录, 则需要先将原记录删除, 然后再将新记录插入到页空间中, 最后再修改目录页中页号和剩余空间的关系以及头页中的信息即可.

删除记录操作的实现过程是:首先调用页管理器部分的 deleteRecord 函数删除记录, 然后修改目录页中页号和剩余空间的关系以及头页中的信息即可.

查找记录操作的实现过程是:首先扫描文件的头页和目录页, 根据扫描结果得到要查找的数据页链. 针对得到的数据页链, 处理思路为:一次扫描一页, 对于每一页, 用页类的 firstRecord()和 nextRecord()方法来处理页里的所有记录, 根据 rid 取出对应的记录, 调用 matchRec()来判断记录是否满足扫描条件, 如果满足, 则将 rid 存入 curRec 并返回 curRec, 为了提高运行速度, 在处理完一页的所有记录之前, 不要释放该页的摁钉, 处理完一页的所有记录后再扫描下一页.

4 构建实际应用

农业垂直搜索引擎 AgriRoom(中文译为农家屋)是新疆维吾尔自治区科技攻关子项目《农村科技信息服务平台关键技术研究与应用示范》的研究成果, 该搜索引擎在单服务器下能够为用户提供上百万个 Web 页面的信息检索服务. 它的工作流程为:根据指定的种子站点, 进行多线程定向抓取, 并将抓取的网页源码以 zlib 的方式压缩后存储在数据库中, 然后经过去标签化、中文分词、消除停用词、敏感词过滤、对抓取的网页建立倒排索引等, 最后再通过一定的排序算法, 对搜集的网页按照相关性进行排序, 将搜索结果以一

个清晰明了的方式的展现给用户. 本文将该页式存储方法应用到上述农业垂直搜索引擎中, 将网页源码、倒排索引、停用词库、分词库等信息分离出来, 使用页式存储方法进行存储, 减少了用户访问数据库的时间开销, 因而极大地提高了检索的效率, 可以满足用户对海量数据的快速检索需求. 在搜索目标中输入关键词玉米, 搜索结果共检索出 7360290 张网页, 耗时仅 0.065 秒, 而使用数据库存储却耗时 0.371 秒. 由此可见, 该页式存储方法在垂直搜索引擎领域具有良好的应用效果.

5 实验测试

一个好的设计必须经过严格的测试. 为此, 本文在一台曙光 64 位刀片服务器上针对该存储方法进行性能测试, 测试数据来源于垂直搜索引擎抓取的 100 万张网页. 为更好地说明使用块式存储的优越性, 本文在相同编程语言和相同的编程环境下, 将该存储方式与传统的按关系型数据库的存储方式进行了时空对比测试, 测试结果如下:

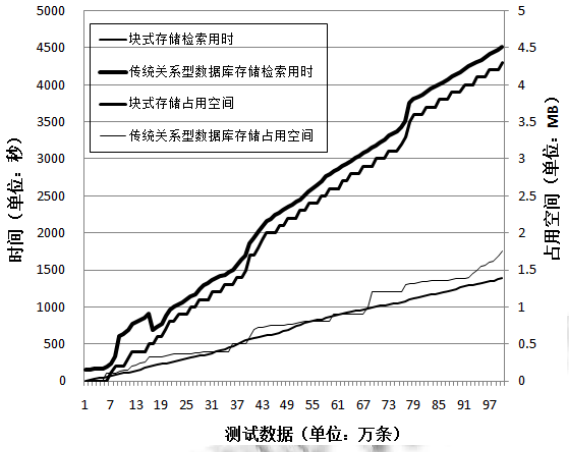

图 1 时空对比测试结果

测试结果表明, 随着并发检索数据量的上升, 两种存储方式检索用时与数据量大小大致呈现出线性关系, 另外, 随着倒排表的增大两种存储方式占用的存储体积与倒排表数据的插入时间也大致呈现出线性关系, 通过对实验结果进行线性拟合以后, 可以得出以下结论:

(1)在数据检索方面, 该方式检索用时明显低于传统关系型数据库存储方式, 尤其在数据量急剧增长时, 块式存储的检索优势更明显;

(2)在占用磁盘存储体积方面, 采用块式存储方式占用磁盘存储较大, 由此可见, 块式存储是以牺牲磁盘存储空间换取检索速度加快的做法.

6 结语

本文着重介绍了磁盘 I/O 层文件管理器、定/变长记录存储以及堆文件管理器的实现, 综合考虑倒排索引的查询性能要求, 设计并实现了合适的记录格式. 最后结合搜索引擎倒排索引构建的实际应用环境, 对该存储方法进行了测试, 测试结果表明该存储方法能有效地实现对倒排文件的数据管理, 具有很强的应用价值. 因此, 在下一步的研究中, 可考虑将该存储方法应用到其他领域, 满足其他领域对数据管理的需要.

参考文献

1 吕晖,丁亚军,郑方等.支持跨步访问的嵌入式存储系统.计算机工程与科学,2014,42(2):211–215.
2 刘锐,李盘林,李秉智.一种适用于大容量 Flash 存储系统的管理方案.计算机应用研究,2006,23(2): 87–88,95.
3 安洋,赵洪松.基于行列混合存储的大数据存储方法研究与实现.通信管理与技术,2014,46(1):24–27.
4 刘阳成,周俭,谢玉波.海量数据存储管理技术研究.微计算机应用,2011,32(10):33–36.
5 张孝,周宁南.非结构化数据存储管理研究.科研信息化技术与应用,2013,6(1):30–40.
6 陈燕红,张太红,冯向萍.小型数据库管理系统中页的设计与实现.电脑知识与技术,2010,17(19):5134–5136.
7 彭波,李晓明.搜索引擎倒排文件的一种分块组织技术.电子学报,2005,44(2):358–362.
8 杨晓波.分块组织技术的倒排索引方法研究.计算机工程与应用,2012,49(5):113–117.
9 郑榕增,林世平.基于 Lucene 的中文倒排索引技术的研究.计算机技术与发展,2010,20(3):80–83.
10 刘小珠,彭智勇,陈旭.高效的随机访问分块倒排文件自索引技术.计算机学报,2010,33(6):977–987.
11 马健,张太红,陈燕红.中文搜索引擎分块倒排索引存储模式,2013,23(7): 2031–2036.
12 邓攀,刘功申.一种高效的倒排索引存储结构.计算机工程与应用,2008,45(31):149–152.
13 王冬,左万利,赫枫龄等.一种增量倒排索引结构的设计与实现.吉林大学学报(理学版),2007,43(6):953–958.
14 黄少林,王华,张玉红等.基于 Lucene 的索引系统的设计与实现.现代情报,2009,30(7):169–171.