文章编号:1671-4598(2016)06-0251-04 DOI:10.16526/j.cnki.11—4762/tp.2016.06.069 中图分类号:TP311 文献标识码:A

海量可视化数据的存储技术研究

张双双,周丽娟
(首都师范大学 信息工程学院,北京 100048)

摘要:可视化是一种将数据转换成图像且进行分析的技术,被多个领域广泛应用;但是当前测控试验平台海量数据可视化研究领域中,存在数据存取耗时大、展示图形速度慢等一系列问题;通常测控试验平台通入液氮来模拟太空中的低温环境,利用电流产生热量来模拟高温环境;然后周期性的采集温度、电流、电压、功率的信息,并将这些存入到本地数据库当中用于绘制曲线并进行分析;随着时间的推移,存入到本地数据库的温度等数据越来越大,关系型数据库在应付如此巨大的数据时需要耗费大量时间;针对上述问题本文提出一种方法在非关系型数据库 Raptor DB 的基础上改善优化其存储结构,从而加快信息的加载速度,从而减少绘制曲线需要等待的时间。

关键词:信息可视化;曲线展示;优化存储结构;Raptor DB

Curve Shows Technology Based on Massive Storage Research

Zhang Shuangshuang, Zhou Lijuan
(Information Engineering College, Capital Normal University, Beijing 100048, China)

Abstract: Visualization is a kind of technology that data can be converted into images and analysis, used in all kinds of fields. However, in the current research field of spacecraft thermal vacuum test, reading data speed and display image speed are very slow in the process of visualization. Usually the current research field of spacecraft thermal vacuum test through the liquid nitrogen temperature simulated space environment, the use of current generated heat to simulate high temperature environments. And periodically collect temperature, current, voltage, power information, and these were stored in a local database and analyzed for drawing curves. With the passage of time, stored in the local database, increasing the temperature and other data, the curve showing the conventional technique for loading large amounts of data plotted as slow, time-consuming. For this paper, a method of the above problems by optimizing data storage structure to improve the speed of loading information, thereby reducing the need to draw a curve to wait.

Keywords: information visualization; curve shows; optimize storage structure; eliminate duplication

0 引言

近几十年来,计算机运算存储能力不断提高,因而数据量越来越大;与此同时,当人们针对巨量数据所能直接获得的信息量愈加有限。为了更好地解决这种数据与信息之间地矛盾之处,信息可视化随即产生[1]。

设备在使用之前,需要在地面的模拟环境中进行模拟测试,以检验设备的各器件在高低温环境中的稳定性,利用容器来模拟设备使用环境,向真空容器中通入液氮来模拟低温环境,利用电流产生热量来模拟高温环境。然后周期性的采集温度、电流、电压、功率的信息,并将这些数据存入到本地数据库当中用于绘制曲线并进行分析。通常绘制曲线所需要的时间由以下几个因素决定:

曲线绘制时间消耗=加载绘图数据时间消耗执行绘图时间消耗;

可视化是一种将数据转换成图像且进行分析的技术,被多个领域广泛应用。但是当前测控试验平台海量数据可视化研究领域中,存在数据存取耗时大、展示图形速度慢等一系列问题。目前航空热试验领域里可视化数据很大程度上存储于关系型数据库中,关系型数据库将一行中的数据值串在一起存储起来然后存储第二行数据,以后数据均为此方式存储[2]。因此每一列数据在物理磁盘中所存放位置是间断不连续的。而在测控试验平台中可视化数据以列为单位进行读取的,因此造成读取可视化数据时需要进行比较多次的磁盘 I/O,将需要很长一段时间进行数据读取。因而,如何建立高效的数据缓存机制提高在测控平台试验领域中可视化数据的读取速度,优化海量数据存储结构并高效的展现可视化数据,已经变为航空测控平台数据可视化研究领域中十分重要的课题。伴随信息技术的快速发展,采用大容量计算机内存进行数据缓存,进行多通道数据采集、数据可视化分析处理和智能预警,将会为航空测控平台领域中数据可视化问题提供解决方案成为研究方向的重中之重。

收稿日期:2016-03-16; **修回日期**:2016-04-06。

基金项目:中国国家科技支撑计划重点项目、"计算机应用技术"北京市重点建设学科(2013BAH19F01)。

作者简介:张双双(1990-),女,北京市人,硕士研究生,主要从事数据挖掘方向的研究。

周丽娟(1964-),女,辽宁省辽阳市人,教授,硕士研究生导师,主要从事数据挖掘方向的研究。

1 原理与设计

1.1 关系型数据库针对海量数据的分析研究

关系数据库,是以关系模型为基础创立的一种数据库,它通过使用集合代数等与数学相关的知识来对针对数据库里存储的数据进行操作[3]。关系型数据库即为二维表与其之间的一部分相关联系组成的一个完善的数据组织[4]。关系型数据库具有数据结构化、低冗余度、较高的程序和数据独立性、易于扩充、易于编制应用程序等优点[5]。关系型数据库是基于行的存储,基于行的存储就是将数据组织成多个行,举一张真空热试验中一张记录温度数据的二维表如图 1 所示。

这一张二维表存储在计算机硬盘中,数据库需要自动使这张二维表存储在一维的字节数组中,关系型数据库需要把任意

时间	Series1	Series2	Series3	···	Series1000
2013年12月17日 12时20分00秒	22.365	21.369	20.354		10.235
2013年12月17日 12时20分06秒	22.364	21.305	20.356		10.236
2013年12月17日 12时20分12秒	22.363	21.304	20.357		10.237
⋮					
2014年1月17日 12时20分00秒	-23.556	10.236	20.897		50.687

图 1 温度数据的二维表

一行中的数据值串联起来进行存储，之后再进行下一行数据串联存储，以上述方法继续进行存储[1]。所以每一列的所有数据存放到物理硬盘中地位置是间断不连续的[6]。例如利用 SQL 语句“Select Series1 from TemperatureTable”读取第一个系列的电流数据时，则硬盘需要寻址多次才能把这一系列的所有电流读取到内存当中。因为硬盘寻址时间相较于计算机上其他部件的运行速度来说慢的不是一般，通常顺序读取数据要比随机访问更快[1]。而且，硬盘寻址时间的提升比起 CPU 速度的进步要慢得多[2]，所以关系型数据库难以应付海量的真空热试验的可视化数据。因此非关系型数据库是应对海量数据的第一选择也是最优的选择方案[7]。

1.2 数据库索引的研究分析

目前大部分数据库系统及文件系统都采用 B－Tree 或其变种的其他变形树作为索引结构，在 B－ 树里从事数据查找包括 2 种基本操作[8]：

(1) 在 B－树中查找结点。

(2) 在结点中查找关键字。

鉴于 B－树的存储通常情况下是在物理磁盘上，因此前一查找也必须是在物理磁盘上进行的操作[9]，然而后一查找则需要在内存中进行的操作，也就是说需要在物理磁盘上查找到指针 p 所指结点[10]，然后先将结点中的信息读入内存，之后再使用顺序查找或折半查找来查询等于 K 的关键字[8]。显而易见的是，内存中查找一次比通过磁盘进行一次查找速度快的多[11]。因此，在磁盘上进行查找的次数以及查找关键字所在结点在 B－ 树上的层次树，是决定 B－树查找效率的首要因素。

在含有 n 个关键码的 m 阶 B－树中查找[9]，从根节点到关键字所在的节点所涉及的节点数 h 不超过 $\log_{\lceil m/2 \rceil}\left(\frac{n+1}{2}\right)+1$ 通常 m 的值都超过 100，因此 h 值会很小。同时这也就决定了以 B－树作为索引是相当高效的。

虽然 B－树的性能已经非常高效了，但是仍然有两个因素制约着它的性能：

(1) 在实际运用 B－树作为索引时，其叶面通常实现为数组或者指针列表，当插入一个关键字时程序需要移动子节点在数组或者指针列表中移动，这将会耗费时间。

(2) 在 B－树的分裂一个页面需要有效的配合父节点和子节点，因此这个时候，整个 B－树会被封锁一定的时间，所以并行的更新是非常的困难。

为了解决上述问题非关系型数据库 Raptor DB 提出数据库的检索应该能满足以下四个方面的要求：

(1) 页面的数据结构需要满足以下三个要求：它们分别为 a. 能够简单的写入磁盘和从磁盘上加载 b. 能够释放内存防止超过内存的限制 c. 按需加载以优化内存使用

(2) 关键字能够快速地插入和检索[12]。

(3) 支持多线程和并发操作。

(4) 页面应该能够联系在一块，所以能够方便的进入下一页面进行范围查询

Raptor DB 设计的索引如图 2 所示。

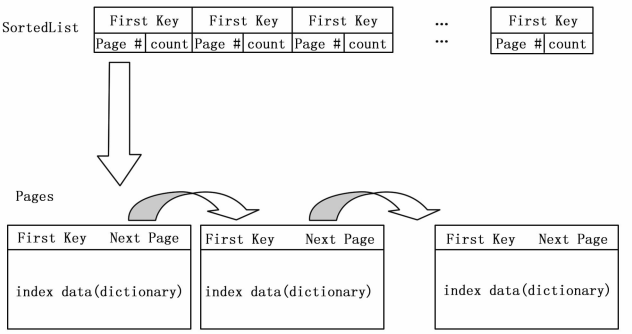

图 2 Raptor DB 设计的索引

每一个 Page 以它的 FirstKey 字段进行升序排序存储在 SortedList 当中，FirstKey 字段和 Page 中存储的关键字相关联，Page 中存储地是关键字和字典结构的关键字相互对应的记录数，记录数也可以说是指向数据的指针[13]，一个 Page 最多能够存储 PageItemCount（默认值是 10000）个关键字，当 Page 存储的关键字的数量超过 PageItemCount 则当前 Page 将会分裂成两个 Page。输入一个关键字检索其所对应的数据，Raptor DB 将会利用 Page 的 FirstKey 进行二分查找，找到关键字所在的 Page 之后，将该 Page 加载到内存中继续查找关键字所对应地记录数，然后利用记录数读取出数据[14]。该索引寻找关键字对应的 Page 的时间复杂度是 O（log$M$）（$M=N/PageItemCount$，N 是所有关键字的数量），在内存中进行查找关键字对应的记录数时间复杂度是 O（1），所以该索引的时间复杂度是 O（log$M$）＋ O（1）。

Raptor DB 的索引有如下优势：

(1) 每个 Page 被 Page List 结构相互隔离开的，所以当前 Page 上锁很容易，且只会锁住当前分裂的 Page，其他的 Page 仍然可以进行其它操作，而不像 B 树在树的节点分裂的时候需要将整棵树锁住。

(2) 分裂 Page 的操作很简单，不需要像 B 树在分裂节点的时候需要来回遍历检查节点的关键字数量是否超出限制。

(3) 存储 Page 的序列结构更新频率很低，所以对 Page 序列上锁几乎不影响其他操作。

在 Raptor DB 中利用 Set（T Key，byte [] data）函数进行存储数据，其中 Key 是二进制数据 data 的唯一索引值。每次调用 Set（T Key，byte [] data）函数 Raptor DB 将会生成一个 doc record 记录输入的 Key 和 data 并存储在磁盘上。假设某次真空热试验中有 n 个温度传感器它们的序号分别是 K_1，K_2，…，K_n，每周期需要将它们采集到的温度数据存储入数据库，一共有 m 个周期。则对于 RaptorDB 而言所有温度传感器第一周期的数据到来时需要调用 n 次 Set（T Key，byte [] data）函数进行存储数据，分别为 Set（K_1，data 1）、Set（K_2，data 2）…Set（K_n，data n），第二周期的数据到来时同样需要调用 n 次 Set（T Key，byte [] data）函数，分别为 Set（K_1，data $1+n$）、Set（K_2，data $2+n$）… Set（K_n，data $2n$），以此类推，当实验结束时 Raptor DB 产生的 doc record

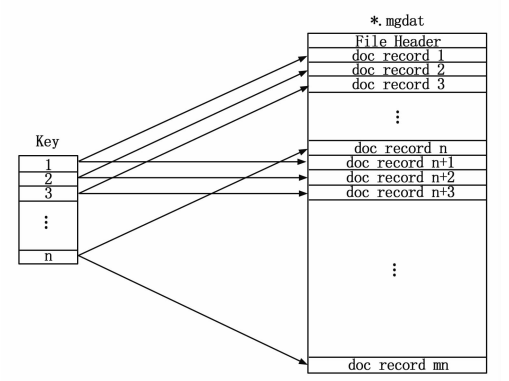

图 3　Raptor DB 产生的 doc record

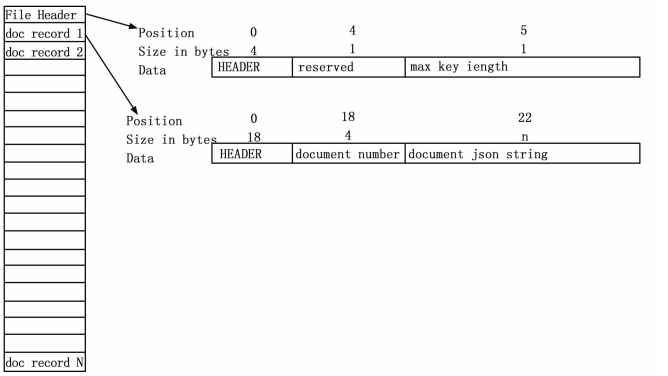

图 4　mgdat 文件的格式

如图 3 所示。在 Raptor DB 中利用 Set（T Key，byte [] data）函数进行存储数据，如图所示当加载某个 Key 对应的所有数据时需要将 *.mgdat 文件中的 Key 所对应的所有 doc record 读取到内存当中，因为同一个 Key 所对应的所有 doc record 存放在 m 个不连续的磁盘块当中所以磁盘需要寻址 m 次，这会导致加载数据的时间很长从而增加绘制曲线的时间。

2　对 Raptor DB 的改进

考虑到从磁盘中加载一个 Key 对应的所有数据需要寻址多次花费过多时间可以考虑改进 RaptorDB 的存储结构使得加载数据的时候磁盘的寻址次数尽可能的少。预分配空间的机制能够有效的减少寻址次数即在真空热试验准备阶段为每个传感器在磁盘上预分配一块较大的空间。新开辟的空间大小需要通过计算得到，例如某个温度传感器在一次实验当中需要收集 m 个周期的温度数据，每个温度的数据类型是 float（占四个字节）类型，则需要为该传感器预分配 $m\times4$ 个字节的空间。当调用 Set（T Key，byte [] data）函数存储数据 data 时，将数据存储在相对应的且已经开辟好的磁盘空间上，然后将 data 的尾部所在磁盘空间的位置记录到 *.mgidx 文件的 Page 上。

其中 Key 是二进制数据 data 的唯一索引值。当调用 Set（T Key，byte [] data）函数时，RaptorDB 会为该 Key 生成唯一的 Duplicate record number，该 Duplicate record number 从 0 开始，每输入一个新的 Key 则 Duplicate record number 和新输入的 Key 相关联后自增 1，自增 1 后的 Duplicate record number 后下一个输入的新 Key 相关联。对于每次输入的 Data 无论它的 Key 是新的还是之前已经出现过的 RaptorDB 将会为该数据生成唯一的 RecordNumber，该 RecordNumber 从 0 开始，每存储一份数据则 RecordNumber 自增 1，自增后的 RecordNumber 和下一个输入的 data 相关联。以同样的 Key 存储若干份数据则会产生一个 Duplicate record number 和若干个 RecordNumber。Key 和 data 存储在. mgdat 文件当中，.mgdat 文件的格式如图 4 所示。

前六个字节是头文件，每调用一次 Set（T Key，byte [] data）函数存储数据将会创建一个新的 doc record，doc record 的头 18 个字节主要用于存储 Key 的字节长度和 data 的字节长度，同时可以存储写入数据时的时间。document number 字段一共四个字节用于存 Key，document json string 字段用于存储 data。同时新创建的 doc record 在 .mgdat 文件上的位置被记录在 .mgrec 文件中，.mgrec 文件的格式如图 5 所示。

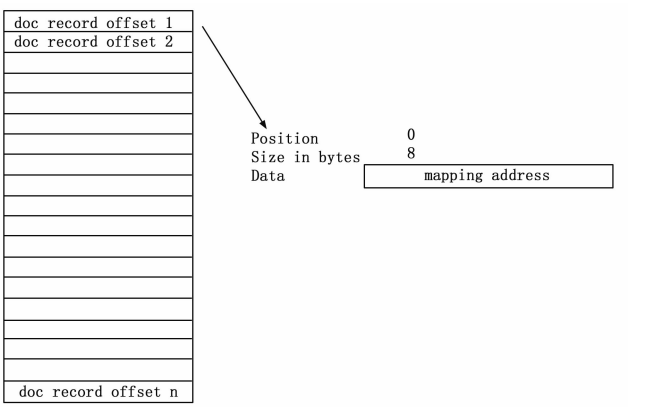

图 5　.mgrec 文件的格式

.mgrec 文件中每个 doc record offset 的长度是 8 个字节，按顺序存储每个 doc record 在 .mgdat 文件中的位置。每个输入的数据对应的 RecordNumber 被存储在 .mgbmp 文件中。.mgbmp 文件的格式如图 6 所示。

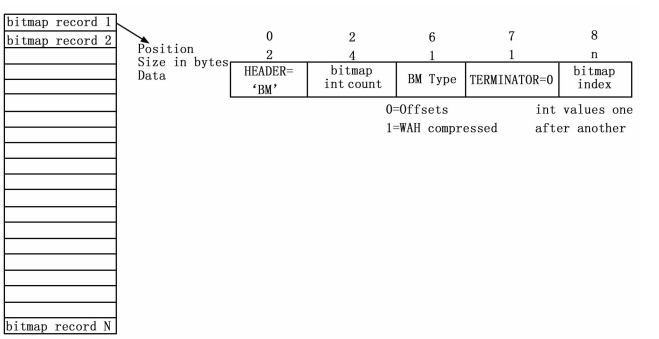

图 6　.mgbmp 文件的格式

每个 bitmap record 记录一个 key 所对应数据的所有 RecordNumber，其中 bitmap int count 字段存储当前 Key 对应数据的份量。Bitmap index 存储的是当前 Key 对应数据的所有 RecordNumber 的二进制表示。每写入一个 bitmap record 则该 bitmap record 在. mgbmp 文件中的位置会被记录在 .mgbmr 文件当中，.mgbmr 文件的格式如图 7 所示。

.mgbmr 文件中每个 doc record offset 的长度是 8 个字节，记录每个写入的 doc record 在 .mgdat 文件中的位置并且按升序的方式一一对应每个 Duplicate record number。即 Duplicate

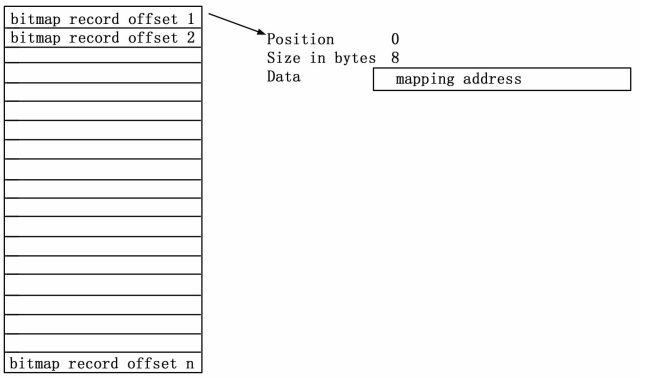

图 7 .mgbmr 文件的格式

record number 乘以 8 得到的结果就是该 Duplicate record number 所对应的 bitmap record offset 在 .mgbmr 文件中的位置。输入的 Key 和 RaptorDB 为其产生的 Duplicate record number 同时被记录在 Page 中，Page 存储在 .mgidx 文件当中，.mgidx 文件的格式如图 8 所示。

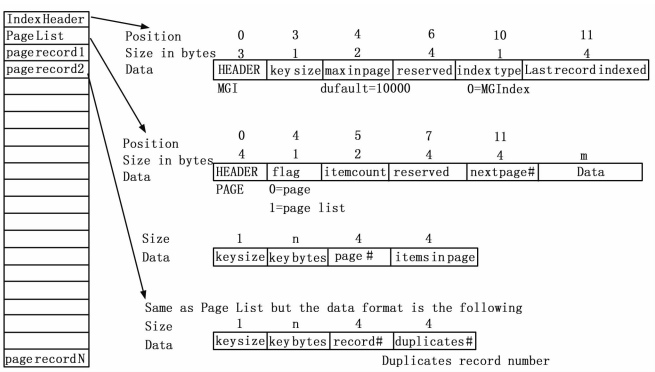

图 8 .mgidx 文件的格式

.mgidx 文件的头部 18 个字节用于存储指示作用的字符串、索引 key 的字节长度、最大页数、指示当前运用哪种类型的索引的数字、最后一个数据存储后产生的 RecordNumber。Page List 头部存储了指示作用的字符串、指示当前为 PageList 类型的数字、当前的 PageList 中包含 Page 的数量，下一个 PageList 的序号。Page List 中的每个 Data 指示的是与当前 Page List 相关联的 Page 的信息，Page 头部的结构和 PageList 的相似，Page 的 Data 存储的是用户输入的 Key 信息包括 Key 的二进制长度、Key 的二进制数据、该 Key 最后一次数据存储的 RecordNumber、该 Key 对应的 Duplicate record number，一个 Page 能存储 PageItemCount（默认值是 10000）个 Key 的信息，当一个 Page 中存储 Key 的数量超过 PageItemCount 时则对 Page 中的所有 Key 进行升序排序一个 Page 分裂成两个 Page，原始的 Page 拥有前［PageItemCount/2］个 Key，并把原始 Page 的 FirstKey 字段赋值为前［PageItemCount/2］个 Key 中最小的 Key，令新生的 Page 的 NextPage 指向原始 Page 的 NextPage，令原始 Page 的 NextPage 指向新生成的 Page，新生成的 Page 拥有后 PageItemCount－［PageItemCount/2］个 Key，并把新生的 Page 的 FirstKey 字段赋值为后 PageItemCount－［PageItemCount/2］个 Key 当中最小的 Key。然后每个 Page 按照 FirstKey 字段进行升序排序存储在 SortedList 当中。

在 RaptorDB 中利用 Get（T Key，out byte［］data）函数来读取数据或者通过 GetDuplicates（T Key）函数和 FetchRecord（intRecordNumber）函数组合来获取数据。在读取数据的时候通过输入的 Key 结合 Page 的 FirstKey 字段进行二分查找找到该 Key 所在的 Page，然后在 Page 的 Data 里找到该 Key 并获取相对应的 Duplicate record number，然后定位到 *.mgbmr 文件的 Duplicate record number * 8 的位置读取 8 个字节长度的数据得到结果 bitmapMappingAddress，然后定位到 .mgbmp 文件的 bitmapMappingAddress 的位置，读取 bitmap record 中记录的所有 Record number，对于每个 RecordNumber 将 .mgrec 文件定位到 RecordNumber 乘以 8 的位置并读取八个字节的数据，得到结果 datMappingAddress，将 .mgdat 文件定位到 datMappingAddress 的位置，根据 doc record 的头文件格式规定来读取数据，并依次返回每个 Record Number 对应的数据。从而获取输入的 Key 对应的所有数据。

3 实验结果对比分析

首先对数据集和试验环境进行描述和说明；然后验证数据索引的有效性，结合可各存储方式在数据存储和读取的对比，海量数据的存储技术的特点进行说明。

3.1 数据集描述和试验环境说明

本文以某测控试验平台为例对提出的可视化方法进行案例研究，该数据集以航天设备在地面的模拟环境中进行模拟测试，以检验设备的各器件在高低温环境中的稳定性，利用容器来模拟环境，向容器中通入液氮来模拟太空中的低温环境，利用电流产生热量来模拟高温环境。然后周期性的采集温度、电流、电压、功率的信息，并将这些数据存入到本地数据库当中用于绘制曲线，该数据集约有 50 万数据。

测试环境为处理器 Intel（R）Core（TM）i5－3337U CPU @ 1.80 GHz；内存类型为 DDR3 SDRAM，其大小是4 096 MB；磁盘型号为 WDC WD5000LPVT－22 G 33T0，容量 476 940 MB（500 GB），转速 5 400 RPM 缓存 8 192 KB。

该数据集完全符合本文所需要的数据集的定义。数据集有一段时间内的温度、电流、电压、功率等信息，可以根据时间进行分析。

3.2 试验结果对比

表 1 各存储方式在数据存储和读取的对比

存储方式	写入总时间/ms	每次写入时间/ms	读取总时间/ms	每次读取时间/ms
页面索引	1542629	1.543	1359923	1.360
SQLite	1204682187	1204.682	97963818	97.964
SQL Server 2008	775853205	775.853	7472808	7.473
MySQL	335464054	335.464	79633331	79.633
内存 B－树	2921019	2.921	1674745	1.675

试验采用生成随机数的类生成随机数模拟传感器采集的数据。传感器数量为 10 000 个，写入行数为 1 000 000 行，写满 1 000 000 行数据后，从第 1 行到最后一行对进行读取，各存储方式的写入和读取的测试结果记录在表 5.1 中。通过表 1 可以看出本文提出的页面索引方法平均每次写入速度是 SQLite780.740 倍，是 SQL Server 2008 的 502.821 倍，是 MySQL217.410 倍，是内存 B－树的 1.893 倍。它的平均每次检

（下转第 259 页）

参考文献：

[1] Vapnik V. The nature of statistical learning theory [M]. Springer Science & Business Media, 2013.

[2] Vapnik V N, Vapnik V. Statistical learning theory [M]. New York: Wiley, 1998.

[3] Chapelle O, Vapnik V, Bousquet O, et al. Choosing multiple parameters for support vector machines [J]. Machine learning, 2002, 46 (1－3): 131－159.

[4] University of California－Irvine. UCI Machine Learning Repository. http://archive. ics. uci. edu/ml/datasets. html. 2015－04－11.

[5] Eitrich T, Lang B. Parallel tuning of support vector machine learning parameters for large and unbalanced data sets [M]. Springer Berlin Heidelberg, 2005: 253－264.

[6] He H, Garcia E A. Learning from imbalanced data [J]. Knowledge and Data Engineering, IEEE Transactions on, 2009, 21 (9): 1263－1284.

[7] Zhang Z, Guo H. Research on Fault Diagnosis of Diesel Engine Based on PSO－SVM [A]. Proceedings of the 6th International Asia Conference on Industrial Engineering and Management Innovation [C]. Atlantis Press, 2016: 509－517.

[8] Smits G F, Jordaan E M. Improved SVM regression using mixtures of kernels [A]. 2002. IJCNN'02. Proceedings of the 2002 International Joint Conference on IEEE [C]. 2002, 3: 2785－2790.

[9] Keerthi S S, Lin C J. Asymptotic behaviors of support vector machines with Gaussian kernel [J]. Neural computation, 2003, 15 (7): 1667－1689.

[10] Duan S M, Mao J L, Li J L, et al. Design Implementation and Application of Swarm Intelligence Algorithm Optimization Function Simulation Platform [A]. Software Engineering and Information Technology: Proceedings of the 2015 International Conference on Software Engineering and Information Technology (SEIT2015) [C]. 2016: 196－203.

[11] Yu S, Zhang J, Zheng S, et al. Provincial carbon intensity abatement potential estimation in China: a PSO－GA－optimized multi－factor environmental learning curve method [J]. Energy Policy, 2015, 77: 46－55.

[12] Shi Y, Eberhart R C. Empirical study of particle swarm optimization [A]. CEC 99. Proceedings of the 1999 Congress on Evolutionary Computation [C]. IEEE, 1999, 3.

[13] Rao B J, Babu M S P. Ongole Breed Cattle Health Expert Advisory System Using Parallel Particle Swarm Optimization Algorithm [J]. 2014.

[14] Yang C, Xu X, Han J, et al. Energy efficiency－based device－to－device uplink resource allocation with multiple resource reusing [J]. Electronics Letters, 2015, 51 (3): 293－294.

(上接第 254 页)

索速度是 SQLite72.032 倍，是 SQL Server 2008 的 5.495 倍，是 MySQL58.555 倍，是内存 B－树 1.231 倍。提出的方法和内存 B－树方法写入和读取速度远远快于其他方法，这是由于索引存储在内存中可以获得高速的存储速度，然而同样把索引维持在内存中，本文提出的页面索引方法的速度要快于 B－树的速度。原因是以 B－树作为索引时，其结点在程序中实现为数组或者指针列表，当插入一个关键字时程序需要移动子结点在数组或者指针列表中移动，这将会耗费时间；当 B－树分裂一个结点时需要有效的平衡该页面所在结点的父结点和子结点。因而在此时整棵 B－树会被封锁一定的时间，所以在这个时候不能同时进行读写操作，而在本文所设计的索引中每个页面被页面数组结构相互隔离开，所以当前页面进行分裂时给当前页面上锁很容易，且只有当前的页面被锁住而其他的页面不会被锁住因此其他的页面仍然可以进行读写操作，页面分裂的操作简单、限制少，因此减少了时间消耗，页面中包含 10 000 个索引项，因此更新的频率很低，因此对页面进行分裂耗费的时间很少。

4　结束语

在 C/S 模式下，分析客户端可视化数据缓存的存储结构和算法，深入分析缓存的存储结构和算法对读取可视化数据所需要时间以及缓存的稳定性的影响。通过在现有的主流数据库当中进行读写速度的比较以及稳定性的比较，选择最合适的数据库，减少存储的数据量，提高缓存的读写速度。详细介绍了一种新型的页面数据索引的设计，该页面索引应用在高实时性的读取某个历史时间戳的数据，并将当前采集的数据和读取的历史数据进行对比，给出了查找页面的具体算法并结合具体例子对页面查找算法进行了详细分析，介绍了存储历史数据的文件结构，并给出了结构设计的依据，详细介绍了数据写入和读取的功能实现，并对各存储方法进行了比较、分析得出了结论。可以较好地解决这一问题. 通过对应用实例进行评估，验证了该技术的有效性和易用性。

参考文献：

[1] 张　棋. 信息可视化在数字图书馆中的应用研究 [D]. 湖北：华中师范大学，2009.

[2] 傅雅玉，源艳芬，梁慎青，等. 提高气象数据查询速度的存储分析 [J]. 电脑知识与技术，2014，(6)：1170－1172.

[3] 虞海燕. 基于 Caché 数据库的电子病历系统性能评价 [D]. 上海：浙江大学，2013.

[4] 傅荣会. 三种关系型数据库管理系统的比较研究——Visual FoxPro，Access 和 SQL Server [J]. 重庆三峡学院学报，2011，(3)：58－59.

[5] 宋亚林，任冬英. 文档型数据库与关系型数据库中数据集成的研究 [J]. 福建电脑，2011，(2)：157－158.

[6] 闫常友. 广域测量系统若干技术及电压稳定应用研究 [D]. 北京：华北电力大学，2005.

[7] 邱胜海，高成冲，王云霞，等. 大数据时代非关系型数据库教学与实验改革探索 [J]. 电脑知识与技术，2013，(31)：7046－7048.

[8] 李东林. 基于 Linux 的嵌入式实时数据库访问算法研究与实现 [D]. 长春：长春理工大学，2008.

[9] 张　华，顾红飞，刘　涛. 基于 B+树的文本信息检索技术 [J]. 皖西学院学报，2010 (2)：31－35.

[10] 朱秋香. 流分类技术研究及其原型系统的实现 [D]. 福建：东南大学，2004.

[11] 田晓辉. B－tree 检索技术及其应用 [J]. 福建电脑，2013 (10)：22－23.

[12] 叶晓舟，王劲林. 混合 Chord 网络中的有效关键字检索 [J]. 计算机工程，2008 (11)：246－248.

[13] 汪　超，何丕廉，李志明. 基于 Hash 表的数据库索引结构设计与实现 [J]. 微处理机，2007 (4)：53－56.

[14] 雷红利. 查询执行算法的设计与优化 [D]. 长沙：华中科技大学，2004.