PostgreSQL 数据库预取算法研究*)

胡巧巧　王建民　叶晓俊
(清华大学软件学院　北京 100084)

摘　要　减少磁盘的存取时间是提高数据库性能的关键。本文讨论了 PostgreSQL 数据库顺序存取的特性，提出了一种 PostgreSQL 中顺序预取数据块的算法，预取的数据块数目可根据当前存取块之前的顺序存取情况作自适应调整。实验结果表明，该算法能有效地提高磁盘块的平均存取速度。

关键词　预取，顺序存取，PostgreSQL，数据块

Research on Prefetching Algrothim in PostgreSQL

HU Qiao-Qiao　WANG Jian-Min　YE Xiao-Jun
(School of Software, Tsinghua University, Beijing 100084)

Abstract　Access speed of disk is the key to database performance. In this paper, after discussing a characteristic of sequential access in PostgreSQL, an algorithm is proposed which can selectively prefetch data blocks ahead of the point of reference. The number of blocks prefetched is chosen based on the observed number of sequential block references immediately preceding reference to the current block. Experimental results show the proposed algorithm can improve access speed effectively.

Keywords　Prefetch, Sequential access, PostgreSQL, Data block

1　引言

在当今的数据库和超文本系统中，影响响应时间的主要问题在于访问对象从磁盘到主存的 I/O 操作。然而，在用户花费大量的时间处理一个数据页时，计算机和 I/O 系统却处于空闲。如果在这段时间内，计算机能够预测从磁盘请求数据块的顺序，并能在需要这些块之前将它们装入主存，就能够减少等待 I/O 完成所花费的时间[1,2]，当用户需要这一块时，可以直接从主存获得，数据库系统就可以获得更快的响应时间。这种将预期要使用的块预先取到主存储器中的方法叫做预取[3~5]。

在数据库系统中，许多查询需要顺序扫描整个表或至少是索引(如果索引存在)来查找记录[2,4]。针对这一特性，本文分析了 PostgreSQL7.4.3 的数据存储访问策略，并提出了 PostgreSQL7.4.3 上顺序预取相邻数据块的算法，最后给出实验结果，说明算法的有效性。

2　问题描述

为了提高效率，数据库逻辑上将磁盘数据划分为块进行存储和维护[6]。数据库块是数据库使用和分配的最小存储单位，数据库在进行输入输出时，都是以块为单位进行逻辑读写操作的。访问关系(表、索引等)中的某个元组时，需要把包含这个元组的整个数据块从磁盘读入主存中。

在 PostgreSQL7.4.3 中，每个关系对应一个单独的磁盘文件，它的元组排放在块中，文件中记录默认是以堆存储方式组织的。在这种组织方式中，一条记录可以放在文件中的任何地方，只要那个地方有空间存放这条记录，并且记录是没有顺序的。PostgreSQL 中关系表的访问方法有两种：表扫描和索引扫描，系统通过这两种方法来定位关系中的元组。在表扫描方法中，关系中记录的查找是通过顺序的表扫描进行的，系统知道包含关系的元组的块，并且可以一个接一个地得到这些块，其操作为 heap_beginscan/heap_getnext：函数 heap_beginscan 根据键值初始化扫描符，为扫描关系表做准备；heap_getnext 才真正开始扫描元组，从关系的第一块的第一个元组或最后一块的最后一个元组开始扫描(由传入的参数指定的扫描方向决定)，依次移向下一个元组，扫描完当前块的所有元组后，再移向文件的下一页，一直扫描直到找到复合条件的元组或到达文件尾结束；为了能快速随机地访问文件中的记录，提高查询性能，会对一个关系建立索引，当有索引存在时，会通过扫描索引得到在构成索引的属性上具有特定值的那些元组，索引扫描的操作为：index_beginscan/index_getnext。“getnext”表明了对索引项的搜索是顺序的。

当 PostgreSQL7.4.3 读取磁盘块时，如果系统能检测到某种顺序访问的模式(如正在读取顺序页)，并确定可能性来预取逻辑上连续的块，即使用单一 I/O 操作将几个连续的页读到缓冲池中，就可以提高 I/O 性能，减少应用程序开销。

3　基于 PostgreSQL 的顺序预取算法

3.1　定义

定义 1　BUF_READ_AHEAD_AREA 表示预读区域大小，以块为单位。

定义 2　BUF_READ_AHEAD_THRESHOLD 表示预读的阈值。

3.2　算法

*)本课题得到了国家 973 计划项目(2004CB719400)、国家自然基金项目(60473077)和国家 863 计划项目(2003AA413230)资助。胡巧巧　硕士研究生，主要研究方向为数据库。王建民　教授，主要研究数据库与工作流技术。叶晓俊　副教授，主要研究数据库。

当系统调用函数 ReadBuffer 存取数据库中的一个数据页时，如果数据页不是已在数据库缓冲池中，就会从文件中读取这个数据页到缓冲区，在读入这一页后，但在返回该页前，我们可以统计文件内以当前页为边界的某个区域内顺序存取情况，决定是否可以顺序预取相邻的连续的页。

算法思想：在以当前存取页为边界的区域内，检查区域内最近被存取的数据页是否按块号顺序存取（升序或降序），并统计发生的次数，如达到一个阈值，就按照这个顺序预取当前页前/后一些连续的页，使得系统存取这些页时，它们已在缓冲区中；否则，就不进行这种顺序预取。

基于 PostgreSQL 的顺序预取算法

输入：关系的定义信息，关系内的数据块号

输出：无

```
方法：the algorithm prefetches the 'natural' adjacent successor and
      predecessor of the page when a page in the buf. pool is accessed
      the first time
Procedure ReadBufferAhead(Relation rel,BlockNum){
  asc_or_desc=1;
  /*根据当前存取页计算物理块区域*/
   low =(BlockNum / BUF_READ_AHEAD_AREA) * BUF_
   READ_AHEAD_AREA;
   high =(BlockNum / BUF_READ_AHEAD_AREA + 1) *
   BUF_READ_AHEAD_AREA;
   if((blockNum ! = low)&&(blockNum ! = high-1))
        return;
   if(high>关系总的块数)
        high =关系总块数;
   if(等待读入数据页到 buffer 的操作>缓冲池大小/2)
        return;
   if(blockNum= = low)
     asc_or_desc=-1 ;
   /*统计区域内最近存取情况*/
   for i=low to high -1do {
       if(不在 buffer 中)
          fail_count++;
          /*非顺序存取*/
            else if(相邻两块的存取次序! = asc_or_desc)
              fail_count++;
   }
  /*如果区域中大多数块最近未被顺序访问过，则不做顺序预取*/
     if(fail_count > BUF_READ_AHEAD_AREA -
                     BUF_READ_AHEAD_THRESHOLD)
       return;
     if(当前存取的块不在 buffer 中)
       return;
     /*计算预读区域*/
     if(blockNum= = low)
        new_blockNum=blockNum-1
     else if(blockNum= =high-1)
        new_blockNum=blockNum+1
     low =(new_blockNum / BUF_READ_AHEAD_AREA) *
     BUF_READ_AHEAD_AREA;
     high=(new_blockNum/BUF_READ_AHEAD_AREA+1)
      * BUF_READ_AHEAD_AREA;
  /*按照块号降序读入的情况下，new_blockNum = high - 1；按照
   块号升序读入的情况下，new_blockNum =low*/
     if((new_offset ! = low)&&(new_offset ! = high-1)){
        return;
     }
     /* new_blockNum ==low 或 new_blockNum ==high-1
   时继续*/
     if(high>关系总的块数)
        high =关系总块数;
  /*按照顺序预取 BUF_READ_AHEAD_AREA 块*/
  for i=low to high -1do
     读入相应的块到缓冲池
}
```

4 实验结果与分析

我们在改造后的 PostgreSQL 上实现了该算法，并对实现预取前后的数据库分别进行了 TPC-C 性能测试。

实验中，我们对 BUF_READ_AHEAD_LINEAR_AREA 和 BUF_READ_AHEAD_LINEAR_THRESHOLD 分别取了不同的值，以下通过一个实例来验证前面所提出的算法。

```
# define BUF_READ_AHEAD_LINEAR_AREA 16
# define BUF_READ_AHEAD_LINEAR_THRESHOLD (3 * BUF_READ_AHEAD_LINEAR_AREA / 8)
```

测试平台：

硬件：内存 1.0G，CPU Pentium 4 2.6G Hz；操作系统：RedHat Linux 9.0；测试时间：1 小时；测试工具：多机版 TPC-C 测试软件。实现预取前 TPC-C 测试结果如图 1，2 所示（每分钟 New-Order 事务个数统计图）。

实现预取后 TPC-C 测试结果如图 3，4 所示（每分钟 New-Order 事务个数统计图）。

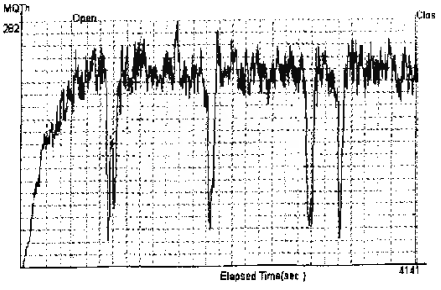

图 1 预取实现前 TPC-C 测试结果(19warehouse)

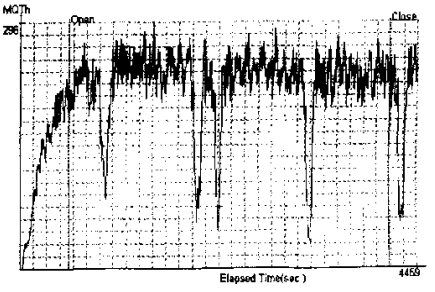

图 2 预取实现前 TPC-C 测试结果(20warehouse)

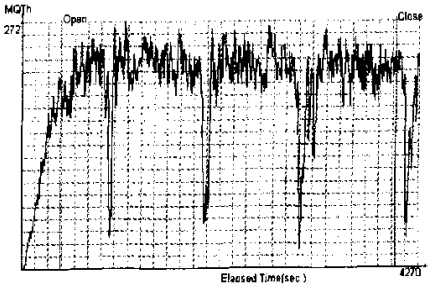

图 3 预取实现后 TPC-C 测试结果(19warehouse)

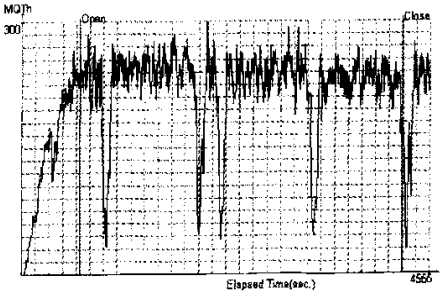

图 4 预取实现后 TPC-C 测试结果(20warehouse)

预取前后每分钟平均 New-Order 事务个数（吞吐量）及 New-Order 响应时间如表 1 所示。

（下转第 159 页）

2 Wilks S S. Mathematical Statistics. New York : Wiley Press, 1962
3 Duda R, Hart P. Pattern Classification and Scene Analysis. New York : Wiley Press, 1973
4 Swets D L, Weng J. Using discriminant eigenfeatures for image retrieval. IEEE Transactions on Pattern Analysis and Machi ne Intelligence, 1996, 18(8): 831～836
5 Belhumeur P N, Hespanha J P, Kriengman D J. Eigenfaces vs Fisherfaces : Recognition using class specific linearprojection. IEEE Trans on Pattern Anal Machine Intell, 1997, 19(7): 711～720
6 Liu Cheng-Jun, Wechsler H. A shape and texture-based enhanced Fisher classifier for face recognition. IEEE Transactions on Image Processing, 2001, 10(4): 598～608
7 Foley D H, Sammon J W Jr. An optimal set of discriminant vectors. IEEE Transactions on Computer, 1975, 24(3): 281～289
8 Duchene J, Leclercq S. An optimal Transformation for discriminant and principal component analysis. IEEE Transactions on Pattern A nalysis and Machi ne Intelligence, 1988, 10(6): 978～983
9 Tian Q. Image classification by the Foley-Sammon transform. Optical Engineering, 1986, 25(7): 834～839
10 杨键,杨静宇,叶晖,等. Fisher 线性鉴别分析的理论研究及其应用[J]. 自动化学报,2003,29(4): 482～493
11 Yang Jian, Yang Jing-Yu . Why can LDA be performed in PCA transformed space? [J]. Pattern Recognition, 2003, 36: 563～566
12 边肇祺, 张学工. 模式识别(第二版)[M]. 北京: 清华大学出版社,1999
13 Sirovich L, Kirby M. Low-Dimensional Procedure for Characterization of Human Faces. J Optical Soc Am, 1987, 4: 519～524
14 Kirby M, Sirovich L. Application of the KL Procedure for the Characterization of Human Faces. IEEE Trans Pattern Analysis and Machine Intelligence, 1990, 12(1): 103～108
15 Turk M, Pentland A. Eigenfaces for Recognition. J Cognitive Neuroscience, 1991, 3(1): 71～86
16 金忠. 人脸图像特征抽取与维数研究:[博士论文]. 南京: 南京理工大学,1999
17 Hong Z Q, Yang J Y, et al. Optimal discriminant plane for a small number of samples and design method of classifier on the plane [J]. Pattern Recognition, 1991, 24(4): 317～324
18 Liu K, Yang J-Y, et al. An efficient algorithm for Foley-Sammon optimal set of discriminant vectors by algebraic method [J]. International Journal of Pattern Recognition and Artificial Intelligence, 1992, 6(5): 817～829
19 Liu K, Cheng Y-Q, Yang J-Y, et al. Algebraic feature extraction for image recognition based on an optimal discriminant criterion [J]. Pattern Recognition, 1993, 26(6): 903～911
20 Chen Li-Fen, Mark Liao H-Y, Ko M-T, et al. A new LDA-based face recognition system which can solve the small sample size problem [J]. Pattern Recognition, 2000, 33(10): 1713～1726
21 Yu Hua, Yang Jie. A direct LDA algorithm for high-dimensional data—with application to face recognition [J]. Pattern Recognition, 2001, 34(10): 2067～2070
22 Gottumukkal R, Asari V K. An improved face recognition technique based on modular PCA approach [J]. Pattern Recognition Letters, 2004, 25: 429～436
23 王松桂. 线性模型的理论及其应用. 合肥: 安徽教育出版社,1987

(上接第 139 页)

表 1 预取算法应用前后 TPC-C 测试数据比较

	数据量 (warehouse)	吞吐量 (tmpC)	New-Order 事务的响应时间(s)		
			90%	平均	最大
预取前	19	212.7	2.93	2.6	48.04
	20	224.07	5.34	3.63	90.23
预取后	19	215.5	2.7	2.16	75.89
	20	225.5	4.17	3.21	82.5

从实验结果可以看出,实现预取后数据库吞吐量有一定的提高,但不是很明显,响应时间有比较大的降低。根据事务处理性能委员会(TPC, Transaction Processing Performance Council)2001 年制定的 TPC-C 测试标准[7],符合:

(1)9 ≤(tpmC 值 / warehouse 数)≤ 12.86

(2)90% New-Order 事务的响应时间≤ 5 sec.

要求的最大 warehouse 数即为 TPC-C 测试结果。从表 1 可以看出,预取前数据库 TPC-C 测试的结果为 19warehouse,预取后数据库 TPC-C 测试结果可达到 20 warehouse 或以上。以上结果表明,实现预取后 PostgreSQL 的性能有了提高。在实验的过程中,通过取不同的 BUF_READ_AHEAD_LINEAR_AREA 值,该算法对性能都有一定程度的提高,能有效地提高磁盘块的平均存取速度。

结论 为了提高数据库性能,减少存取磁盘块所需要的平均时间,本文给出了 PostgreSQL 上顺序预取的算法,实验结果表明了算法的有效性。在以后的工作中,我们将进一步研究其它预取技术[8～10]在数据库上的应用,并与顺序预取技术作比较。

参 考 文 献

1 Palmer M, Zdonik S. Fide: A Cache that Learns to Fetch. In: Proc. of the 1991 Int Conf. on Very Large Databases, Barcelona, Catalonia, Spain, Sep. 1991. 255～264
2 Shah P, Paris J-F, Amer A, Long D D E. Identifying Stable File Access Patterns. In: Proc. of the 21st IEEE Symposium on Mass Storage Systems and Technologies (MSST 2004), College Park, MD, April 2004. 159～163
3 DB2 Administration Guide: Prefetching Data into the Buffer Pool. https://aurora.vcu.edu/db2help/db2d0/fra me3.htm#preftch
4 Smith A J. Sequentiality and Prefetching in Database Systems. ACM Transactions on Database Systems(TODS), 1978, 3(3): 223～247
5 Kroeger T M, Long D D E. Design and Implementation of a Predictive File Prefetching Algorithm. In: Proc. of the 2001 USENIX Annual Technical Conf. Boston, Massachusetts, USA, June 2001. 105～118
6 Silberschatz A, Korth H F, Sudarshan S 著, 阳冬青, 唐世渭, 等译. 数据库系统概念(原书第 4 版). 北京: 机械工业出版社, 2003. 275～289
7 Transaction Processing Performance Council. http://www.tpc.org. TPC Benchmark™C - Standard Specification, Revision 5.0. February 26, 2001. 60～70
8 Vitter J S, Krishnan P. Optimal Prefetching via Data Compression. Journal of the ACM, 1996, 43: 771～793
9 Curewitz K, Krishnan P, Vitter J S. Practical Prefetching via Data Compression. In: Proc. of the 1993 ACM SIGMOD Int. Conf. on Management of Data, Washington, D. C., USA, May 1993. 771～793
10 曹新平, 刘美华, 等. 预取技术研究进展. 计算机科学, 2003, 30(8)

PostgreSQL数据库预取算法研究

作者： 胡巧巧， 王建民， 叶晓俊， HU Qiao-Qiao， WANG Jian-Min， YE Xiao-Jun
作者单位： 清华大学软件学院,北京,100084
刊名： 计算机科学 ISTIC PKU
英文刊名： COMPUTER SCIENCE
年，卷(期)： 2006，33(3)
引用次数： 0次

参考文献(10条)

1. Palmer M. Zdonik S Fide:A Cache that Learns to Fetch 1991

2. Shah P. Paris J-F. Amer A. Long D D E Identifying Stable File Access Patterns 2004

3. DB2 Administration Guide:Prefetching Data into the Buffer Pool

4. Smith A J Sequentiality and Prefetching in Database Systems 1978(3)

5. Kroeger T M. Long D D E Design and Implementation of a Predictive File Prefetching Algorithr 2001

6. Silberschatz A. Korth H F. Sudarshan S. 阳冬青. 唐世渭 数据库系统概念 2003

7. Transaction Processing Performance Council TPC BenchrnarkTMC - Standard Specification,Revision 5.0 2001

8. Vitter J S. Krishnan P Optimal Prefetching via Data Compressior 1996

9. Curewitlz K. Krishnan P. Vitter J S Practical Prefetching via Data Compression 1993

10. 曹新平. 刘美华. 韩真. 古志民. 张建鑫 预取技术研究进展[期刊论文]-计算机科学 2003(8)

相似文献(0条)