基于写数据页聚簇的固态硬盘缓冲算法①

李顺芬, 陈小刚, 周 密, 李鸽子, 王玉婵, 宋志棠

(中国科学院 上海微系统与信息技术研究所, 上海 200050)

摘 要: 针对 Flash 写前需擦除, 读写 I/O 开销不均衡等固有缺陷, 研究面向闪存缓冲区管理, 对提高基于 Flash 的固态硬盘(Solid State Disk, SSD)访问性能具有重要理论意义和应用价值. 通过分析 SSD 关键技术及现有缓冲区管理算法, 实现了一种适用于 SSD 的基于写数据页聚簇缓冲算法. 文章中详细介绍了该算法关键技术及原理, 并通过 FlashSim 仿真平台实现 SSD 写缓冲. 基于仿真结果与传统缓冲算法性能比对, 分析得出该缓冲算法可降低 SSD 随机写次数和 SSD 数据存储分散性, 并提升 SSD 响应速度.

关键词: 固态硬盘; 缓冲管理; 聚簇; 置换策略; 随机写

Buffer Algorithm for Solid State Disk Based on the Cluster of Write Pages

LI Shun-Fen, CHEN Xiao-Gang, ZHOU Mi, LI Ge-Zi, WANG Yu-Chan, SONG Zhi-Tang

(Shanghai Institute of Micro-system and Information Technology, Chinese Academy of Sciences, 200050, China)

Abstract: For the inherent characteristics of flash memory such as erase before write, the I/O overhead of reading and writing unbalance, and so on, studying the buffer management of flash memory has important theoretical significance and application value to improve access performance of Flash-based SSD (Solid State Disk, SSD). Analyzing of the key technologies of SSD and the existing buffer management algorithm, it has implemented a buffer algorithm used for SSD, which is based on the cluster of writing pages. The key technology and the principle of the algorithm has been described in detail, and the writing of SSD buffer management is implemented on basis of Flashsim platform. Compared the performances of traditional buffer algorithm and the new algorithm, it indicates that the buffer algorithm based on the cluster of write page can reduce the random write times and the data dispersion of SSD and improve the average response rate of SSD.

Key words: solid state disk; buffer managemen; cluster; replacement strategy; random write

随闪存作为一种新型固态存储介质, 因其具有体积小、重量轻、非易失、高速、高抗震、低功耗等优良特性, 近年来已经被广泛应用于固态存储, 然而由于 Flash 写操作前需要擦出操作、读写以及删除操作 I/O 开销不对称、随机写操作带来寿命有限等不足, 导致基于闪存的固态硬盘同样具有以上不足; 同时随机数据的写性能成为固态硬盘性能瓶颈, 对基于 SSD 的数据存储管理技术提出了新挑战, 其中, 缓冲区管理作为提高固态硬盘存取性能的一种重要的有效手段, 近年来已成为闪存数据管理领域研究热点. 目前, 已有的操作系统多采用针对磁盘存储系统的传统缓冲区页面置换算法, 没有考虑闪存读写代价不对称的特性以及闪存写代价显著高于读代价的问题, 直接用于闪存时使得闪存的访问性能较差[1].

因此, 针对闪存特性, 研究面向闪存缓冲区管理方面的问题, 对提高闪存访问性能具有重要的理论意义和应用价值.

1 固态硬盘及现有缓冲算法关键技术

1.1 固态硬盘的存储管理架构

基于闪存自身硬件特性, 传统磁盘操作系统和数据库系统均无法直接支持闪存设备. 为解决上述难题,

① 收稿时间:2013-08-15;收到修改稿时间:2013-10-04

固态硬盘采用独特存储管理架构，如图 1 所示，在充分利用闪存存储特性的基础之上，同时对上层文件系统屏蔽闪存存储特性，以实现地址映射、坏块管理、损耗均衡、读写调度等技术，从而为上层应用提供标准块设备调用接口并通过内部的缓存机制、磨损均衡机制、可靠的坏块管理、错误校验机制以及快速启动与掉电恢复机制等相关技术来提高固态硬盘的使用寿命以及数据安全性，降低读写请求平均响应时间.

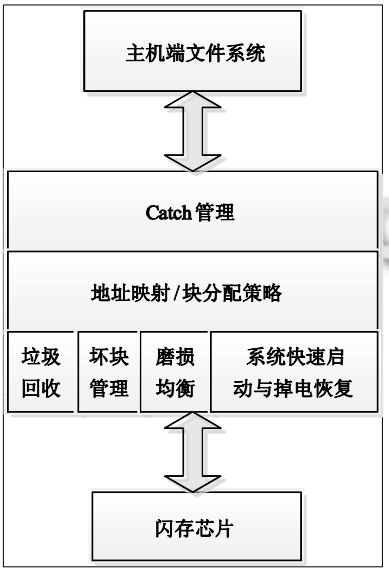

图 1　固态硬盘存储管理架构

基于固态硬盘存储架构，文件系统把基于闪存的固态硬盘看作一个块设备，上层应用程序以访问磁盘模式访问固态硬盘，而不需要更改访问接口. 当上层应用程序执行数据重写或者原位更新操作时，FTL(Flash Translation Layer) 会使得新数据写入固态硬盘中不同的物理页，甚至是不同的物理块. 因此，通过减少数据页的重写操作可以减少闪存的写操作和擦除操作，以提高闪存访问性能，延长闪存使用寿命.

1.2 现有缓冲算法进展

缓冲区管理作为固态硬盘存储体系的重要组成部分是提高闪存访问性能的有效手段之一. 现有的缓冲区置换策略如 LRU、LRU-K、LIRS、FBR、ARC 等已被广泛地用于操作系统、数据库系统以及存储系统中. 但现有缓冲区置换策略大多是针对磁盘，主要着眼于提高缓冲区命中率，然而闪存读写代价不对称特性决定了缓冲区管理策略不同于一般磁盘存储的缓冲区置换策略，需要对读写操作加以区分，在提高缓冲区命中率的同时，尽可能地减少写操作[2]. 此外，基于闪存随机写代价要远高于闪存连续写代价，因此增加连续写的概率也成为提高闪存访问性能的另一种手段. 目前，一些科研机构对缓冲管理进行了相关研究，其中，比较典型的缓冲算法主要包括以下几种:

CFLRU(Clean-First LRU)算法，在基于 LRU 算法基础上考虑闪存写代价显著高于读代价的特性，通过优先置换出 LRU 链表尾部的一定区间内干净页，减少闪存写操作，降低闪存访问开销[3].

LRU-WSR(LRU-Write Sequence Reordering)算法，针对闪存读写代价不对称特性，对冷热数据加以区分，优先换出干净页和冷脏页，延迟换出缓存中访问较为频繁的脏页，从而减少写操作，一定程度上可降低闪存访问开销[4].

FAB(Flash-Aware Buffer Management Policy)算法是针对便携式媒体播放器设计的一种块级缓冲区置换策略. FAB 算法针对顺序访问数据特性，在进行数据页置换时，优先置换出缓冲区中包含数据页目最多的数据块，将其位于缓冲区中的页面换出[5]. 基于特定应用环境下，FAB 可以显著减少闪存访问开销，但 FAB 针对特定应用环境，管理粒度较粗，通用性差.

2 基于写操作数据页聚簇缓冲算法

2.1 基于写操作数据页聚簇缓冲算法概述

针对 Flash 写操作前需要擦除操作、读写以及删除操作的 I/O 开销不对称、随机写代价远高于连续写代价以及随机写操作带来寿命有限等不足特性对固态硬盘储存访问性能的限制，本文提出一种适用于固态硬盘的基于写操作数据页聚簇缓冲算法，该缓冲算法根据上层文件系统应用程序写操作请求，基于逻辑页号将属于同一逻辑块号的数据页按写请求顺序以单向链表形式进行聚簇，以单向链表形式管理同一逻辑块内的写操作数据页，并以聚簇所对应逻辑块号为节点形成单向缓冲区链表，以缓冲区链表形式管理各个聚簇节点，从而实现对所有写操作数据页基于聚簇的分组管理，当聚簇内数据页更新或插入新数据页时将该聚簇节点移动到缓冲区链表队首，聚簇在缓冲区链表的顺序反应该聚簇访问频率，当聚簇节点达到阈值时，优先将包含数据页最多的聚簇节点回写到闪存并以页地址映射方式建立 FTL 表，基于写操作数据页聚簇缓冲算法原理如图 2 所示.

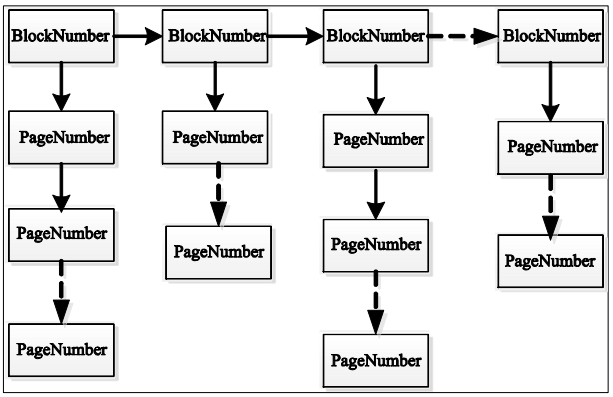

图 2 基于写操作数据页聚簇缓冲算法原理

基于写操作数据页聚簇缓冲算法主要涉及聚簇节点对应数据结构: 物理块号、聚簇节点所包含数据页数、聚簇节点内数据页逻辑页号、读写请求标识以及指向下一聚簇节点的结构体指针.

```
struct  CLUSTER_NODE _INF {
int  blk_num;
int  count;
int  mapdir_flag;//请求标示
int  page_index[64];//写入 page 逻辑页号
struct  CLUSTER_NODE_INF *node_next; };
```

2.2 基于写操作数据页聚簇缓冲算法读写流程

基于写操作数据页聚簇缓冲算法的写流程, 如图 3 所示. 当接收到上层文件系统应用程序发出的写数据请求时, 系统首先判定缓冲区链表是否为空, 若是则以请求信息建立聚簇节点并插入缓冲区链表并更新相应计数器, 若否则判断缓冲区链表中是否存在相应聚簇节点, 若存在将聚簇节点移动到缓冲区链表队首, 插入或更新数据页节点并更新聚簇相应节点计数器, 若否则判定聚簇节点是否达到阈值, 若否则则以请求信息建立聚簇节点并插入缓冲区链表并更新相应计数器, 若是则查找包含最多数据页的聚簇节点并查找闪存内是否存在空闲物理块, 若存在则直接将上述聚簇节点回写到闪存, 若不存在则启动均衡机制获得空闲可用物理块后将相应聚簇节点回写到闪存, 然后删除相应聚簇节点并更新聚簇节点计数器, 最后以请求信息建立聚簇节点并插入缓冲区链表并更新相应计数器, 从而完成写操作请求.

基于写操作数据页聚簇缓冲算法的读流程, 如图 4 所示, 当接收到上层文件系统应用程序发出的读数据请求时, 系统首先判定缓冲区链表中是否存在读操作请求对应的聚簇, 若否则直接从闪存中读取相应数据, 若是则判定聚簇节点中是否存在相应数据页, 若存在则说明数据页在聚簇缓冲区链表中, 则从缓冲区链表节点相对应的聚簇节点中读取数据, 若否, 则从闪存中读取相应数据页, 从而完成读操作请求. 读操作数据页时不改变数据页聚簇在缓冲区链表顺序.

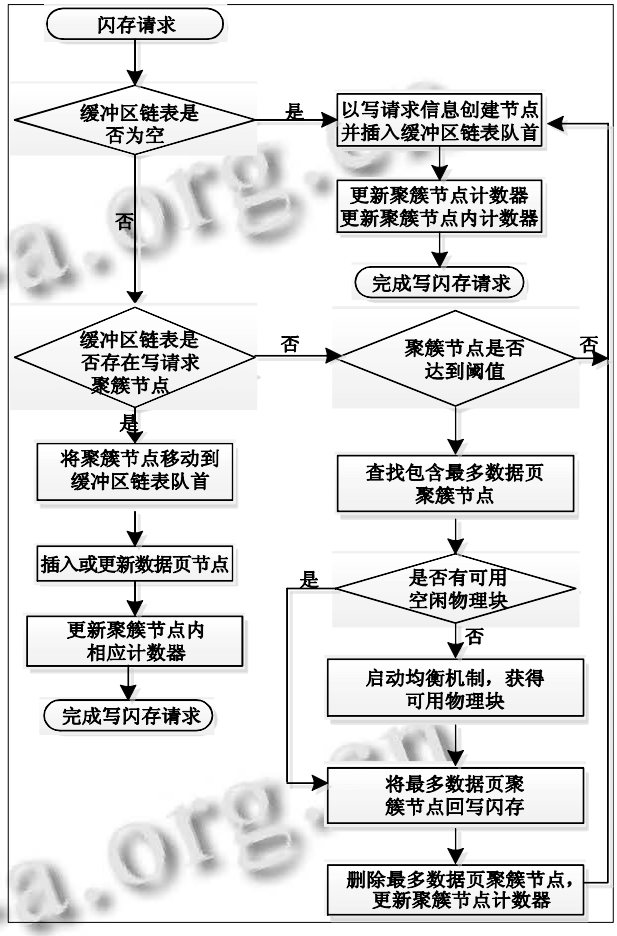

图 3 基于写操作数据页聚簇缓冲算法写操作流程

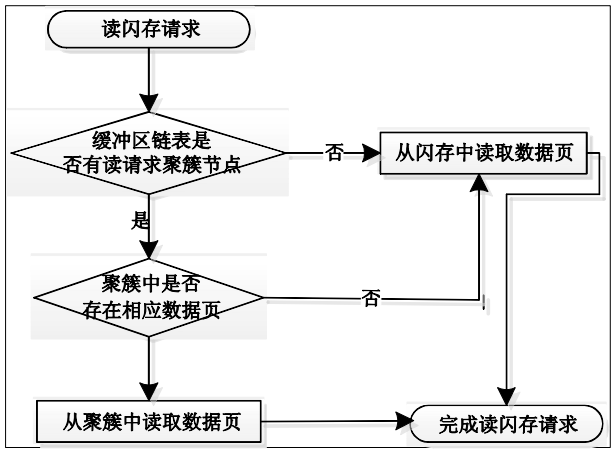

图 4 基于写操作数据页聚簇缓冲算法读操作流程

基于数据页聚簇算法可实现写数据缓冲，降低闪存随机写以及擦除次数，并提升系统平均响应速度.

3 实验与结果分析

3.1 实验平台与环境

本文实验数据基于服务器仿真而来，其中，仿真所使用服务器为英特尔八核中央处理器 E5504，主频为 2GHz，内存容量为 2GB，所用 Linux 内核为 Linux 2.6.32-5-686，硬盘格式 ext3，容量为 500GB；仿真软件采用由宾夕法尼亚大学研发的 SSD 仿真器 FlashSim，基于 FlashSim 实验平台软硬件模架构如图 5 所示. FlashSim 是一个由事件驱动的仿真平台，实现对 SSD 各个模块功能模拟，FlashSim 硬件组成包括：Page、Block、Plane、Die、Package、SSD、RAM、Bus、Channel、Controller[6]. 软件上包括: Event、Address、FTL、Garbage Collection、Wear Leveler. 其中 Event 用于触发 SSD 各个命令，比如读、写、擦除、合并等, Address 是物理地址描述，包含了 Page、Block、Plane、Die、Package 的序号值.

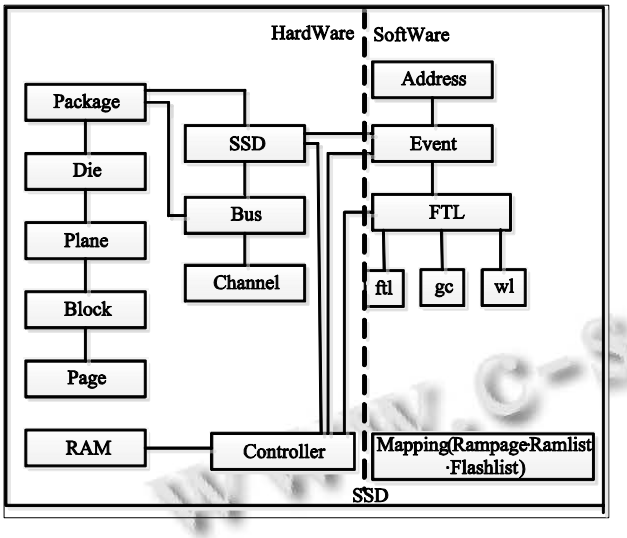

图 5　基于 FlashSim 的实验平台软硬件模块

仿真测试数据是基于 FlashSim 仿真 SSD 工作重要组成部分，本文所采用仿真测试数据是基于 DiskMon 采集 NTFS 文件系统下硬盘真实读写操作请求，本文基于仿真测试数据表征上层应用程序通过文件系统下对 SSD 存储数据读写请求状况，其中，仿真测试数据记录共包含 5 个域：请求时刻(Time，单位是毫秒)、设备号(Device number)、请求类型(Request Type，0 表示 Write，1 表示 Read)、扇区号(Sector)、长度(Length)，部分仿真测试数据如图 6 所示.

2333.081000	0	3	2	0
2346.188000	0	7	3	0
2398.617000	0	11	1	0
2898.617000	0	12	5	0
6383.206000	0	18	5	1

图 6　DiskMon 收集的硬盘的数据访问记录

3.2 实验结果分析

为表征基于写操作数据页聚簇缓冲算法优越性，本文基于仿真平台同时仿真无缓冲算法及基于传统 RUL 置换缓冲算法下固态硬盘的读写操作状况，并基于同一套仿真测试数据在无缓冲算法(NO-CATCH)、传统 RUL 置换缓冲算法(BUF-RUL)以及基于写操作数据页聚簇缓冲算法(BUL-CLUSTER)下固态硬盘读写性能比对.

图 7 为基于 DiskMon 收集的同一套仿真测试数据请求下基于 Flashsim 仿真对固态硬盘实际写次数，可以明显看出无缓冲算法(NO-CATCH)情况下对固态硬盘的实际写次数最大，在基于传统 RUL 置换缓冲算法(BUF-RUL)下，对固态硬盘实际写次数降低了接近 30%，而基于写操作数据页聚簇缓冲算法(BUL-CLUSTER)可对固态硬盘实际写次数降低 64%，由此可见基于写操作数据页聚簇缓冲算法(BUL-CLUSTER)可显著降低对固态硬盘随机写次数，进而提高了固态硬盘使用寿命；同时，基于数据页聚簇缓冲算法在聚簇节点回写闪存时以数据块形式回写，相对以数据页回写闪存，可明显降低固态硬盘存储数据的分散性.

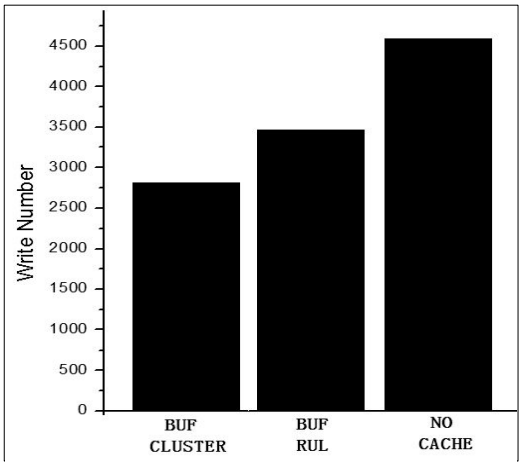

图 7　基于 Flashsim 仿真固态硬盘的实际写次数

图 8 为基于 DiskMon 收集的同一套仿真测试数据请求下基于 Flashsim 仿真固态硬盘平均响应速度, 图 9 为在 DiskMon 收集的同一套仿真测试数据请求下基于 Flashsim 仿真固态硬盘完成写操作累积概率-时间分布图. 由图 8、图 9 可见, 基于写操作数据页聚簇缓冲算法可显著提速固态硬盘性能, 极大提升固态硬盘平均响应速度, 无缓冲算法(NO-CATCH)情况下固态硬盘平均响应速度最慢, 在基于传统 RUL 置换缓冲算法(BUF-RUL)下, 固态硬盘平均响应速度提高了 15%, 而基于写操作数据页聚簇缓冲算法(BUL-CLUSTER)固态硬盘平均响应速度提高了 30%; 同时由图 9 可见, 在无缓冲算法(NO-CATCH)情况下及基于传统 RUL 置换缓冲算法(BUF-RUL)完成写请求 80%工作量的时间远落后于写操作数据页聚簇缓冲算法(BUL-CLUSTER), 可著提升固态硬盘响应速度.

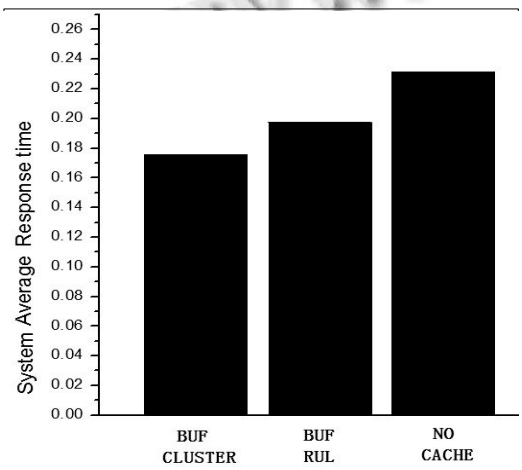

图 8 基于 Flashsim 仿真固态硬盘的平均响应速度

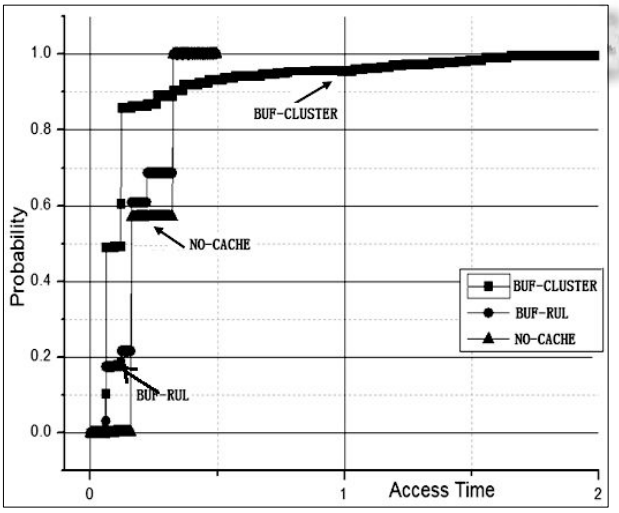

图 9 基于 Flashsim 仿真固态硬盘写操作累积概率-时间分布(单位为毫秒)

综上所述, 基于写操作数据页聚簇缓冲算法(BUL-CLUSTER)能够显著降低对固态硬盘随机写次数和固态硬盘数据存储的分散性, 提升了固态硬盘响应速度.

4 结语

作为一种新型存储介质, 闪存已被广泛应用于嵌入式系统和航天航空等重要领域. 随着闪存容量快速增长和价格不断下降, 基于闪存的 SSD 已成为重要存储设备; 同时, 缓冲区管理作为提高固态硬盘存取性能的重要手段也已成为数据存储又一热点研究方向. 本文所提出的适用于固态硬盘基于写操作数据页聚簇缓冲算法, 实现了 SSD 写缓冲, 可大副降低固态硬盘随机写次数和存储数据分散性, 并提升其响应速度, 该缓冲算法对基于固态存储缓冲研究具有一定借鉴及参考意义.

参考文献

1 Yoo Y, Lee H, Ryu Y, Bahn H. Page replacement algorithms for NAND flash memory storages. Computational Science and Its Applications, 2007: 201–212.

2 Bouganim L, Jónsson B, Bonnet P. Understanding flash IO patterns. Proc of 4th Biennial Conference on Innovative Data Systems Research. http://www-db.cs.wisc.edu/cidr/cidr2009/Paper_102. pdf.

3 Park SY, Jung D. CFLRU: A replacement algorithm for flash memory. Proc. of the 2006 International Conference on Compilers, Architecture and Synthesis for Embedded Systems. 2006. 234–241.

4 Jung H, Shim H, Park S, et al. LRU-WSR: Integration of LRU and writes sequence reordering for flash memory. IEEE Trans on Consumer Electronics, 2008, 54(3): 1215–1223.

5 Jo H, Kang JU, Park SY, et al. FAB: Flash-aware buffer management policy for portable media players. IEEE Trans on Consumer Electronics, 2006, 52(2): 485–493.

6 Kim Y, Taurus B, Gupta A, Urgaonkar B. Flashsim: A simulator for NAND flash-based solid-state drives. Proc. SIMUL. 2009. 9.