具有磨损均衡意识的混合固态硬盘 FTL 算法

姚英彪[1),2)] 王发宽[1)]

[1)](杭州电子科技大学通信工程学院 杭州 310018)

[2)](南京邮电大学通信与信息工程学院 南京 210003)

摘 要 在采用 SLC(Single-Level-Cell)和 MLC(Multi-Level-Cell)闪存的混合固态硬盘设计中,SLC 和 MLC 之间的写数据分配和磨损均衡是混合固态硬盘闪存转换层设计的关键问题之一.针对此问题,提出一种具有磨损均衡意识的混合固态硬盘闪存转换层算法——WLAFTL(Wear Leveling Aware Flash Translation Layer).首先,它提出了一种动态的基于磨损均衡思想和请求大小融合的数据分配机制,即根据 SLC 和 MLC 的磨损速率来动态调整热数据识别阈值的大小,然后将小的写请求分配到 SLC、大的写请求分配到 MLC.其次,它提出了一种基于磨损均衡思想与数据先进先出(FIFO)调度策略融合的 SLC 冷数据回收/迁移机制,减少由 SLC 向 MLC 迁移的数据量.实验结果显示,与 ComboFTL 和 CFTL 算法相比,在使用相同地址映射机制的条件下,WLAFTL 算法的平均响应时间分别平均有 13.6%和 12.7%的改善,总的擦除次数分别平均减少 9.2%和 20.4%,同时能够更好地实现 SLC 和 MLC 之间的磨损均衡.

关键词 混合固态硬盘;闪存转换层;磨损均衡;数据分配;冷/热数据识别

中图法分类号 TP311 **DOI 号** 10.11897/SP.J.1016.2018.02379

Wear Leveling Aware FTL for Hybrid Solid State Disks

YAO Ying-Biao[1),2)] WANG Fa-Kuan[1)]

[1)](*School of Communication Engineering, Hangzhou Dianzi University, Hangzhou* 310018)

[2)](*School of Telecommunication and Information Engineering, Nanjing University of Posts & Telecommunication, Nanjing* 210003)

Abstract With the rapid development of NAND flash memory technologies, replacing hard disks with NAND flash based solid state disks (SSDs) has become a recently emerging trend in computer storage domain. Nowadays, SSDs are mainly based on homogeneous NAND flash chips such as single-level cell (SLC) flash chips and multi-level cell (MLC) flash chips. SLC and MLC chips are distinctive in terms of capacity, performance, and endurance. Specifically, SLC has the advantage of fast read/write, low power consumption and long life, and MLC has the advantage of capacity units with low cost. Therefore, it is a promising method to design hybrid SSDs, consisting of both SLC and MLC flash chips, to achieve a response time and lifetime as same as a SLC-based SSD while maintaining the price of a MLC-based SSD. In order to realize these design goals of hybrid SSDs, the key issues are: (1) how to allocate data into SLC and MLC region; and (2) how to make wear leveling between SLC region and MLC region. In other words, SLC region should be used to store hot data (frequently accessed or updated data) to benefit from its fast read and write performance and long durability, and MLC region should be used to store cold data to benefit from its low cost and large capacity. Meanwhile, the SLC and MLC region should be

收稿日期:2016-05-02;在线出版日期:2017-04-14.本课题得到国家自然科学基金项目(61100044,61671192)、浙江省重点科技创新团队项目(2013TD03)和"电子科学与技术"浙江省一流学科 A 类资助.**姚英彪**,男,1976 年生,博士,教授,中国计算机学会(CCF)会员,主要研究领域为存储系统设计、无线传感器网络、媒体信号处理等.E-mail:yaoyb@hdu.edu.cn.**王发宽**,男,1991 年生,硕士,工程师,主要研究方向为固态硬盘存储技术.

synchronous wear as far as possible to realize the same lifespan of heterogeneous SLC/MLC chips. In order to solve these problems, this paper proposes a wear leveling aware flash translation layer (WLAFTL) for hybrid SSDs. Firstly, WLAFTL proposes a dynamic data allocation scheme, which is based on the fusion of wear leveling and write-request size. Specifically, it employs a traditional write-request size based hot or cold data identification and allocation scheme which identifies write requests whose size is less than the threshold value as hot data and sends them to SLC region, and identifies the rest of data as cold data and sends them to MLC region. However, the threshold value of hot or cold data identification is not fixed and can be adaptively adjusted according to the wear speed of SLC and MLC region. Secondly, WLAFTL proposes a garbage collection or migration scheme of cold data for SLC region, which is based on the fusion of wear leveling and first in first out (FIFO) strategy. Specifically, it employs FIFO strategy to manage the data block in SLC region, and when the space of SLC region is not enough, it employs the following scheme to make room for newly arrived hot data: if SLC region wears faster than MLC region, the valid data block of the first in SLC region is immediately migrated into MLC region; otherwise, the valid data block of the first in SLC region is collected into SLC region no more than N times. Our experiment results demonstrate that under the condition of using the same address mapping scheme, compared to ComboFTL and CFTL, WLAFTL can achieve up to 13.6% and 12.7% average response time improvement, as well as 9.2% and 20.4% total block erasure reduction, respectively. Moreover, it can achieve better wear leveling between SLC and MLC than ComboFTL and CFTL.

Keywords hybrid solid-state drive; FTL; wear-leveling; data allocation; cold/hot data identification

1 引 言

近年来,随着固态硬盘 SSD(Solid State Disk)设计技术的不断进步,相比传统的机械硬盘,SSD 显示出具有读写速度快、功耗低、体积小、防震抗摔、便于携带等方面的优势,它已经在许多领域开始替代传统机械硬盘.因此,SSD 的设计研究逐渐受到学术界和工业界的关注,目前成为存储领域的研究热点之一.

常见的 SSD 主要基于 NAND 闪存,而 NAND 闪存的结构与传统的磁存储介质不同,其主要物理特性有:(1)按页(page)、块(block)、平面(plane)的结构进行组织;提供读、写和擦除 3 种操作;页是读/写的最小单位,一般为 2 KB/4 KB/8 KB;块是擦除的最小单位,一个块一般包含 64 个页/128 个页;(2)闪存擦除后只能写一次,即所谓的写前擦除,这造成闪存不能原地更新,否则会带来巨大的开销;(3)闪存每个存储单元的编程/擦除(P/E)次数有限,超过擦除次数后该存储单元存储数据不再可靠[1-2].

由于闪存的上述特点,在 SSD 的设计中,一般要提供一个中间软件转换层,称为闪存转换层 FTL (Flash Translation Layer),负责隐藏闪存上述特性,将 SSD 模拟成只有读写操作的传统硬盘的形式,以适应当前的文件系统[3].FTL 一般由地址映射、垃圾回收和磨损均衡三个模块组成[4].地址映射负责将来自文件系统的逻辑地址转换为闪存中的物理地址;垃圾回收[5]负责擦除块来回收无效页,令无效页可以被重复利用;磨损均衡负责保证每个块的磨损速率尽量一致,防止部分块因磨损过快而提前损坏.

目前,基于 NAND 闪存的 SSD 大多采用同质结构,即采用单一 SLC(Single-Level-Cell)或 MLC (Multi-Level-Cell)芯片构成底层的物理存储.SLC 结构简单,每个 SLC 单元存储 1 bit 数据;MLC 结构复杂,每个 MLC 单元可以存储 2 bit 数据.SLC 的读写速度、功耗、P/E 次数都明显优于 MLC;但是,SLC 的单位存储容量的成本也明显高于 MLC①.由

① SLC vs. MLC: An Analysis of Flash Memory. http://www.supertalent.com/datasheets/SLC_vs_MLC%20whitepaper.pdf

于 SLC 和 MLC 各有优缺点，学术界开始探讨利用 SLC+MLC 构建混合 SSD，其目标是利用 SLC 的快速读写性能和耐久性存储热数据(频繁访问/更新的数据)，利用 MLC 的低成本及大容量存储冷数据，从而实现混合 SSD 在性能、寿命、成本之间的折衷.

本文主要研究混合 SSD 的 FTL 的 SLC/MLC 写数据分配和磨损均衡问题.具体来说，写数据分配是指 FTL 在处理写请求时，将哪些数据写到 SLC 区域，哪些数据写到 MLC 区域；磨损均衡是指 FTL 尽量保证 SLC 区域和 MLC 区域的磨损速率一致，实现 SLC/MLC 同步磨损的目标.针对此问题，本文提出具有磨损均衡意识的混合 SSD WLAFTL 算法，其创新主要体现在两个方面：(1)提出一种基于磨损均衡思想和请求大小融合动态写数据分配机制，即根据 SLC 和 MLC 的磨损速率来动态调整热数据识别阈值的大小，最后将小于阈值的写请求分配到 SLC、大于阈值的写请求分配到 MLC；(2)提出一种基于磨损均衡思想与数据先进先出(FIFO)调度策略融合的 SLC 冷数据回收/迁移机制，减少由 SLC 向 MLC 迁移的数据量.实验结果显示，与 ComboFTL 和 CFTL 相比，在使用相同地址映射机制的条件下，平均而言，WLAFTL 的平均响应时间有 13.6% 和 12.7% 的改善，总的擦除次数减少 9.2%和 20.4%，同时能够更好地实现 SLC 和 MLC 之间的磨损均衡.

2　相关工作

2.1　同质 SSD 设计

根据地址映射粒度，同质 SSD 的 FTL 可以分为页映射[6-11]、块映射[12]以及混合映射[13-16].页映射优点在于映射灵活、性能高，缺点在于映射表需要占用大量 RAM 空间.块级映射表优点在于 RAM 开销少，缺点在于地址映射的灵活性低、性能差.混合映射机制可以克服以上两种映射方式的缺陷，但它存在垃圾回收效率低和处理随机访问请求性能差的缺点.

针对混合映射机制出现的问题，Gupta 等人[6]重新设计页级映射机制，提出一种按需的页级 FTL 算法(DFTL).在 DFTL 思想的启发下，国内外的研究人员提出各种新型页级闪存转换层算法[6-11].S-FTL[7]提出通过识别 3 种空间本地性(顺序写、簇写和稀疏写)访问模式来减少按页映射的映射表大小.SDFTL[8]提出在 RAM 中增设连续映射表缓存和二级映射表缓存.CDFTL[9]提出以“簇”的方式来装载/剔除映射项.WAPFTL[10]提出一种能够预测负载读写特性并自适应地调整地址映射信息缓存的页级地址映射算法.TPFTL[11]提出用两级 LRU 表来组织映射表信息，并提出基于请求级和选择性两种映射项自适应的预取策略.

缓冲区管理算法[17-20]及磨损均衡算法[21-24]同样是同质 SSD 的研究热点.FClock[17]将数据页组织为两个环形数据结构，分别用于存储缓冲区中的干净页和脏页.APB-LRU[18]提出一种缓冲区脏页概率替换机制.CBM[19]提出一种页/块自适应的合作缓冲区管理机制.CARF[20]提出一种具有代价意识的缓冲区管理机制.Jimenez 等人[21]提出一种通过均衡页的擦除次数来延长 SSD 的使用寿命的新型的磨损均衡算法；Pan 等人[22]提出一种基于原始比特误码率的贪婪磨损均衡算法.Woo 等人[23]提出一种具有多样化磨损指数的新型磨损均衡算法.Liao 等人[24]提出一种不同阶段采用不同触发条件进行块间数据交换的自适应的磨损均衡算法.

2.2　混合 SSD 设计

目前，存在两种类型的 SLC+MLC 混合 SSD：一种是利用闪存芯片厂家提供的同时包含 SLC 块和 MLC 块的闪存芯片来构建混合 SSD[25-26]；另外一种是利用分开的 SLC 芯片和 MLC 芯片来构建混合 SSD[27-29].本文针对的混合 SSD 是第 2 种方案，这方面代表性的工作如下：

较早的混合 SSD 的 FTL 算法是 Park 等人[30]提出的 MixedFTL. MixedFTL 将所有的写数据先写到 SLC 区域，当 SLC 区域中较长时间没有发生更新，则将 SLC 区域中的冷数据迁移到 MLC 区域.这种方法的弊端是会导致 SLC 区域中大量不必要的写操作以及造成大量的 SLC 到 MLC 的数据迁移.Lee 等人[31]提出的 FTL 算法叫 FlexFs，该算法根据 SLC 区域和 MLC 区域的存储空间大小来决定数据的分配，而并非取决于数据的热度，最后根据数据的热度来决定是否迁移到 MLC 区域，这同样会导致大量的冷数据写入到 SLC 区域，占据 SLC 区域的存储空间，引发频繁的垃圾回收和 SLC 向 MLC 的数据迁移.上述两种方式均会导致 SLC 区域的磨损速度过快，SLC 区域存储块损坏的时间比 MLC 区域早，这样缩短了混合固态硬盘的整体使用寿命.

为更好地利用 SLC 闪存特性,进一步改善混合 SSD 的整体性能和使用寿命,研究人员提出只将热数据(频繁更新的数据)存储到 SLC 区域,即 SLC 区域负责接收热数据,MLC 区域负责接收冷数据. Kwon 等人[28]提出 DPFTL 算法,DPFTL 根据负载的访问模式区分数据冷热,将连续请求分配到 MLC 区域,非连续请求(热数据)分配到 SLC 区域,根据 SLC 和 MLC 区域的磨损速率来决定是否将 MLC 区域的热数据迁移到 SLC 区域. 然而该文未考虑将 SLC 中的冷数据迁移到 MLC 区域,这样可能导致冷数据长时间停留在 SLC 区域,占据 SLC 区域的存储空间,影响 SLC 闪存的处理数据的性能. Im 等人[25]提出 ComboFTL 算法,该算法中采用基于负载大小的热数据识别机制,通过统计 SLC 往 MLC 区域的数据迁移量来调整阈值大小,将热数据和冷数据分别分配到 SLC 和 MLC 闪存中. 此外,ComboFTL 将 SLC 区域分为 hot 区和 warm 区进行数据回收/迁移管理,当 hot 区启动垃圾回收机制时,将有效数据复制到 warm 区,warm 区垃圾回收时根据 N 次机会原则将循环周期次数达到 N 次的有效数据迁移到 MLC 区域,没有达到 N 次的依然停留在 warm 区. Chang 等人[27]提出的 CFTL 算法也是采用基于负载大小的热数据识别机制,它通过二均值聚类算法来调整阈值大小,将热数据和冷数据分别分配到 SLC 和 MLC 闪存中. 此外,CFTL 将 SLC 区域作为循环队列使用,启动垃圾回收时选择循环队列中最老的块的作为目标块,然后将其中的有效数据直接迁移到 MLC 区域.

2.3 本文动机

本文通过研究发现,CFTL 和 ComboFTL 各有优缺点. CFTL 将 SLC 作为循环队列使用,可以有效实现 SLC 内部的磨损均衡;将 SLC 区域中的冷数据直接迁移到 MLC,可以有效减少 SLC 的擦除次数. 但是,当 MLC 区域磨损速度过快时,CFTL 没有有效机制降低 MLC 区域的磨损速率;当 SLC 区域的容量很小时,可能存在热数据还没来得及更新就被直接迁移到 MLC 区域,无法利用 SLC 提高混合 SSD 的性能. ComboFTL 将 SLC 区域分为 hot 和 warm 区,warm 区垃圾回收时采用 N 次机会原则. 一方面,这种机制能够延长热数据在 SLC 停留的时间,提高其被 SLC 服务的概率,从而提高混合 SSD 的性能. 另一方面,这种机制也额外增加 SLC 内部的数据迁移次数,增加了 SLC 的垃圾回收开销和磨损程度,有时可能得不偿失. 此外,这两种算法的热数据识别阈值的自适应调整机制都不与 SLC 和 MLC 之间的磨损均衡直接相关,造成这两种算法控制 SLC 和 MLC 的磨损均衡效果都不是很优秀.

本文提出的 WLAFTL 是在 CFTL 和 ComboFTL 的基础上进行改进. 具体来说,本文在 SLC 的使用上,也是将 SLC 作为循环队列使用,与 CFTL 一样;在 SLC 向 MLC 的数据迁移方面,本文既采用了 CFTL 的直接迁移策略,也采用了 ComboFTL 的 N 次机会原则;同时,这 3 种算法都是采用基于请求大小的热数据识别机制. WLAFTL 与 ComboFTL 和 CFTL 不同之处在于:

(1)数据分配机制不同. 这 3 种算法都采用了基于请求大小的热数据识别方法,但是它们的热数据识别的阈值调整机制不同. 具体来说,WLAFTL、CFTL 和 ComboFTL 分别采用基于 SLC 和 MLC 的磨损速率、二均值聚类算法、SLC 往 MLC 的数据迁移量的阈值自适应调整机制.

(2)数据迁移机制不同. WLAFTL 的数据迁移机制是 CFTL 和 ComboFTL 数据迁移机制的综合,并根据 SLC 和 MLC 区域的磨损速率来选择适合的迁移机制. 具体来说,当 SLC 磨损慢于 MLC 时,采用 ComboFTL 的 N 次机会策略,本文称之为延迟迁移;反之,采用 CFTL 的直接迁移策略,本文称之为正常迁移.

(3)地址映射机制不同. 具体来说,对于 SLC 区域,WLAFTL 和 ComboFTL 均采用采用纯页级地址映射机制,CFTL 采用扇区级地址映射机制,而扇区级地址映射机制需要较大的地址映射表空间;对于 MLC 区域,WLAFTL、CFTL 和 ComboFTL 分别采用 SDFTL[8]、BAST、基于聚簇思想的页级地址映射机制,BAST 的垃圾回收开销大,基于聚簇思想的页级地址映射机制需要维护较大的地址映射表空间,而我们前期研究的 SDFTL 地址映射机制,不仅能消除垃圾回收时的合并操作,且只需要维护较小的地址映射表空间.

3 WLAFTL 算法设计

3.1 总体概述

WLAFTL 总体架构如图 1 所示,其写数据分配机制由磨损均衡控制(wear leveling control)和热数据识别(hot data filter)两部分组成. 地址映射为

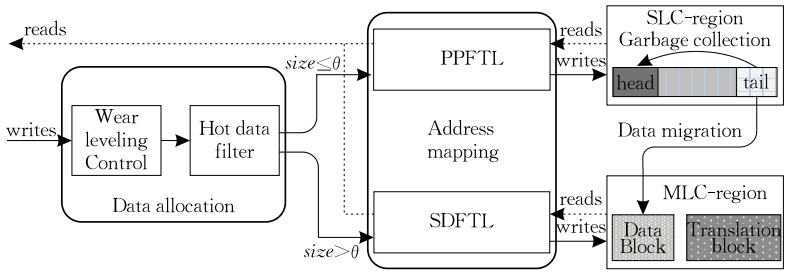


图 1　WLAFTL 总体架构图

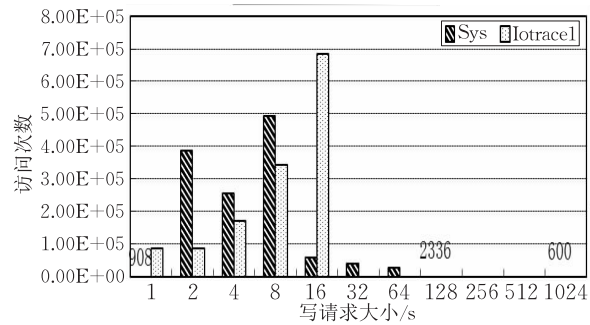
PPFTL 和 SDFTL，其中 PPFTL 表示纯页级地址映射机制，用于 SLC 闪存的地址映射；SDFTL 表示基于连续缓存和二级缓存的 DFTL 改进型地址映射机制，用于 MLC 闪存的地址映射. SLC 闪存作为循环队列使用，head 和 tail 表示循环队列的头指针和尾指针；MLC 闪存被分为数据块区域和翻译块区域，数据块区域用于实际的数据存储，翻译块区域用于页级映射表的存储.

一旦文件系统有新的写请求到来，首先经过磨损均衡控制机制判断 SLC 与 MLC 区域之间的磨损快慢，根据两个区的磨损速率来调整阈值 θ 大小，由热数据识别机制将负载小于阈值 θ 的写请求分配到由 SLC 区域中，负载大于阈值 θ 的写请求分配到 MLC 区域中. 当 SLC 区域中空闲块不足时，垃圾回收机制被触发，选择最先写入数据的块作为垃圾回收的目标块，然后分别根据延迟迁移或正常迁移策略将目标块中的有效数据进行内部迁移或者向 MLC 区域迁移，再擦除该块进行回收. 如果是更新的写请求，则意味着此数据为热数据，直接分配写到 SLC 区域.

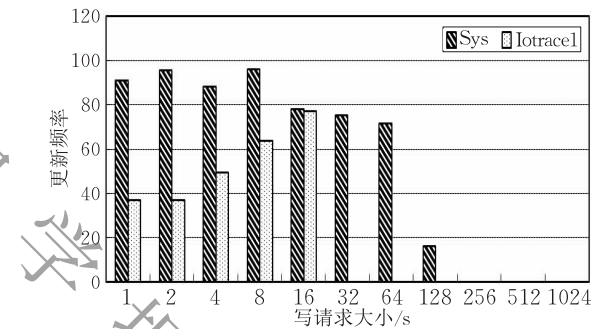
WLAFTL 算法中的磨损均衡主要包括区域内磨损均衡和区域间磨损均衡. 区域内磨损均衡指的是 SLC 和 MLC 区域内部各块间的磨损均衡；区域间磨损均衡指的是 SLC 区域和 MLC 区域之间的磨损均衡. 将 SLC 区域作为循环队列使用，可以保证 SLC 内部块的磨损均衡一致；MLC 区域内部通过监测各个块的擦除次数，优先选择擦除次数最少的块来存储数据，通过这种方式来实现 MLC 内部块间的磨损均衡. 区域域间磨损均衡主要通过两种方式实现：基于磨损均衡的热数据识别阈值调整和冷数据从 SLC 到 MLC 的迁移.

3.2　数据分配机制

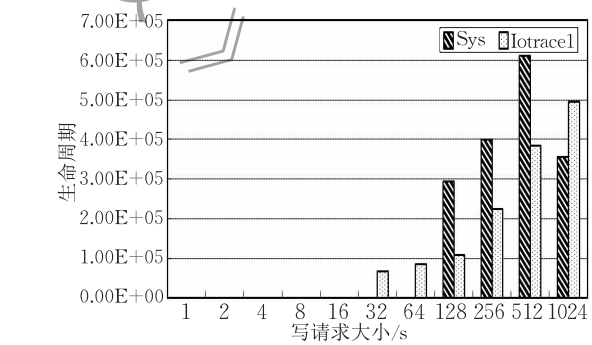
Sys 和 Iotrace1 都是由 DiskMon[①] 收集的个人 PC 的负载，图 2 为本文对两组负载进行离线分析后得到的统计图.



(a) 不同负载下写请求大小分布图



(b) 相同大小的写请求的更新频率图



(c) 相同大小的写请求的平均生命周期

图 2　不同 trace 的写请求特点

图 2(a)是不同负载下写请求大小分布图，该图表明负载中大部分写请求为小请求；图 2(b)为写请求的更新频率，该图表明更新频率高的写请求是小的写请求；图 2(c)为相同大小的写请求的平均生

① Russinovich M. DiskMon for Windows v2.01. [2006]. http://technet.microsoft.com/enus/sysinternals/bb896646.aspx

命周期,生命周期指的是相同请求再次出现时的访问次数间隔,图 2(c)表明小请求的生命周期较短,即更新快.图 2 数据表明小的写请求具有较高的时间局部性,其更新频率或成为热数据的概率大.文献[16,32]在对其它负载进行离线分析时也得出同样结论.因此,WLAFTL 采用基于负载大小的热数据识别机制,将小于阈值 θ 的数据分配到 SLC 区域,大于阈值 θ 的数据分配到 MLC 区域.

WLAFTL 数据分配机制另外一个关键问题是阈值 θ 如何设置.如果 θ 值过大,则过多数据会被写入到 SLC 区域,导致 SLC 闪存磨损过快,同时也会增加 SLC 区域往 MLC 区域的数据迁移量;如果 θ 值过小,虽然从 SLC 区域往 MLC 区域的数据迁移量减少了,但会有更多的写请求数据被写入到 MLC 区域,从而导致 MLC 区域磨损速度过快.为延长混合 SSD 的寿命,这需要尽可能使得 SLC 区域和 MLC 区域的相对磨损速率趋于一致,即使得 SLC 和 MLC 区域的使用寿命保持一致,同时达到擦除次数极限.考虑到阈值 θ 与 SLC 和 MLC 的磨损速率直接相关,本文提出直接根据 SLC 和 MLC 的磨损速率来动态调整 θ 的大小,具体如下:

令 N_s 和 N_m 分别表示 SLC 闪存块和 MLC 闪存块的总数量,l_s 和 l_m 分别表示 SLC 闪存块和 MLC 闪存块的擦除周期,E_i^s 和 E_j^m 分别表示 SLC 闪存块和 MLC 闪存块中每个块的擦除次数.SLC 和 MLC 区域的相对磨损速率 RW_s、RW_m 分别定义为

$$RW_s=\frac{\sum_{i=0}^{N_s}E_i^s}{N_s}\Big/\frac{l_s}{l_m},\ RW_m=\frac{\sum_{j=0}^{N_m}E_j^m}{N_m}.$$

混合 SSD 的等效总擦除次数定义为

$$E_{\text{total}}=\frac{\sum_{j=0}^{N_m}E_j^m}{N_m}\times\frac{l_s}{l_m}+\frac{\sum_{i=0}^{N_s}E_i^s}{N_s}.$$

当$[RW_s]=[RW_m]$($[\cdot]$代表取整)时,可以认为此时 SLC 区域和 MLC 区域的磨损速度一致,即此时处于磨损均衡状态.当$[RW_s]>[RW_m]$时,此时 SLC 区域磨损比 MLC 区域快,因此需要减小阈值 θ,使得更多的新的写请求被分配到 MLC 区域,直到磨损速率回到$[RW_s]=[RW_m]$状态;反之当$[RW_s]<[RW_m]$时,则需要增大阈值 θ,将更多的写请求分配到 SLC 区域,直到磨损速率回到$[RW_s]=[RW_m]$状态.

综上所述,WLAFTL 提出的基于磨损均衡和请求大小融合的数据分配机制如算法 1 所示,其中 $\Delta\theta$ 表示 θ 的增减幅度,其值可根据不同的需求自行设定.

算法 1. WLAFTL 算法的数据分配机制.

输入:请求项 R,请求类型 R_{type},请求大小 $Size=R_{size}$,逻辑请求页号 $LPN=R_{lpn}$,判决阈值:θ 初始值为 4KB

输出:依据 MLC 和 SLC 的磨损速率自适应调整阈值 θ

```
1.  WHILE Size≠0 DO
2.   IF R_type is read THEN
3.    Search PPN mapped to LPN in the page mapping table;
4.     IF the PPN exist in SLC region THEN
5.        read from SLC region;
6.     ELSE
7.        read from MLC region;
8.     END
9.   ELSE  /* R_type is write */
10.    IF it is an update request THEN
11.      update R in SLC region;
12.    ELSE
13.      IF [RW_s]==[RW_m] THEN
                                  /* 磨损速率相等 */
14.        θ←θ;                   /* θ不变 */
15.      ELSE IF [RW_s]>[RW_m] THEN
                                  /* SLC 磨损快 */
16.        θ←max{0,θ−Δθ};
         /* 减小θ,减少 SLC 区域的写数据接收量 */
17.      ELSE     /* MLC 磨损快 */
18.        θ←θ+Δθ;
         /* 增大θ,增加 SLC 区域的写数据接收量 */
19.      END
20.      IF Size≤θ THEN
21.        write R to SLC region;
22.      ELSE
23.        write R to MLC region;
24.      END
25.    END
26.  END
27. END
```

3.3 SLC 区域数据管理与迁移机制

为较好实现 SLC 区域内各闪存块之间的磨损均衡,我们借鉴 CFTL[27] 的做法,将 SLC 区域当作一个循环队列使用,新来的数据写入到 head 指针指向的块,当 SLC 空余块少于一定量后,启动垃圾回收机制,回收 tail 指针指向的块,然后将 tail 指针前

移.可以看到,这种机制就是一种简单的基于 FIFO 的数据迁移机制.这种机制的缺点在于如果 SLC 区域的容量很小,则热数据可能还没来得及更新就被迁移到 MLC 区域.为此,WLAFTL 的 SLC 区域数据回收/迁移机制里面融合 ComboFTL 的 N 次机会策略和 CFTL 的直接迁移策略,在此基础上提出一种基于磨损均衡思想和 FIFO 调度策略融合的 SLC 冷数据回收/迁移机制.具体方法是对 SLC 的地址映射表的每个映射项中增加一个 *Cycle_Time* 变量来记录数据停留在 SLC 区域的循环周期数,当垃圾回收目标块中所对应的页为有效页且循环一周,则 *Cycle_Time* 值加 1;最后根据 *Cycle_Time* 的大小和 SLC、MLC 的磨损均衡情况决定是采用何种迁移策略.该机制框图如图 3 所示.

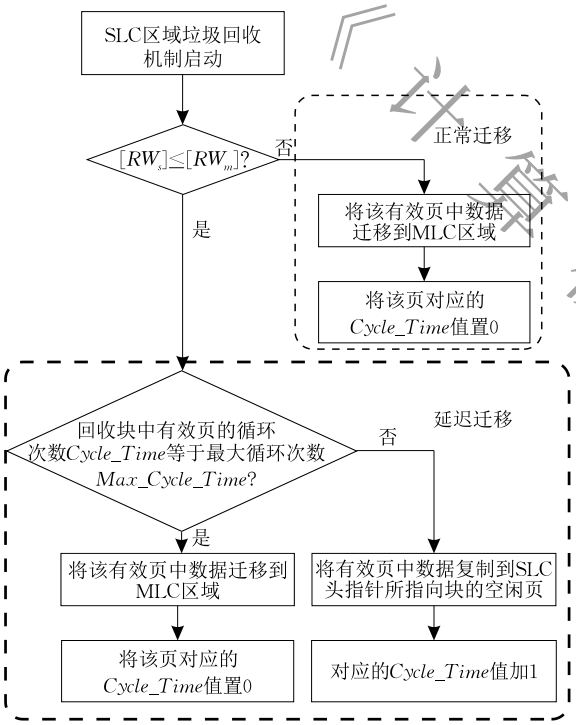


图 3　SLC 区域数据迁移策略

延迟迁移策略.在 SLC 磨损速率低于 MLC 磨损速率时,只有当垃圾回收目标块中的有效页的 $Cycle\_Time = Max\_Cycle\_Time$(最大循环次数)时,该页才作为更新不频繁的冷数据迁移到 MLC;当有效页的 $Cycle\_Time < Max\_Cycle\_Time$ 时,该有效页被复制到 SLC 区域 head 指针指向的块中,并将其对应的 *Cycle_Time* 值加 1,开始新的一轮循环.通过这种方式,使得 SLC 中的数据能够在 SLC 区域停留 Max_Cycle_Time 个循环周期,从而增加其在 SLC 被服务的机会.

正常迁移策略.在 SLC 磨损速率高于 MLC 磨损中的有效页的数据直接迁移到 MLC 区域,减少 SLC 的写入次数,从而降低其磨损速率.

具体的数据迁移算法伪代码如算法 2 所示.

算法 2.　SLC 区域数据迁移算法.

```
/* SLC 中所有有效数据页都有一个 Cycle_Time 标识表示该页在 SLC 的循环次数,Source block 是位于 SLC 尾部的数据块,Target block 是位于 SLC 头部的数据块 */
输入:SLC_Free_Block:SLC 中的空闲数据块数,
      Max_Cycle_Time:SLC 中数据页的最大循环次数
输出:将 SLC 中满足条件的有效数据页迁移到 MLC
1.  WHILE SLC_Free_Block<4 DO
    /* 启动垃圾回收机制 */
2.    FOR each valid page in the Source block DO
3.      IF [RW_s]≤[RW_m] THEN
4.      /* 启动延迟迁移策略 */
5.        IF Cycle_Time==Max_Cycle_Time THEN
6.          move the valid pages to MLC region;
7.          Cycle_Time←0;
8.        ELSE
9.          copy the valid pages to the Target block of
            SLC region;
10.         Cycle_Time←Cycle_Time+1;
11.       END
12.     ELSE
          /* 当 [RW_s]>[RW_m],启动正常迁移策略 */
13.       move the valid pages to MLC region;
14.       Cycle_Time←0;
15.     END
16.   END
17. END
```

由算法 1 和算法 2 可以看到,当 MLC 区域磨损速率大于 SLC 区域磨损速率时,WLAFTL 有两种机制降低 MLC 写入次数,增加 SLC 的写入次数.第一,增大冷热数据识别的阈值 θ,此时能够增大往 SLC 区域的数据写入量,减少 MLC 区域的数据写入量;第二,采用延迟迁移策略,延长热数据停留在 SLC 区域的时间,使数据在 SLC 区域中更新的可能性增大,同时也减少了往 MLC 区域的数据迁移量.同理,当 MLC 区域磨损速率小于 SLC 区域磨损速率时,WLAFTL 只需要采取降低阈值 θ 和正常迁移策略就能实现增加 MLC 写入次数,减少 SLC 的写入次数的目标,从而能够更好地实现 SLC 和 MLC 的磨损均衡.

4 实验结果与分析

4.1 实验设置

在实验中，本文采用 FlashSim[33] 进行算法性能评估. FlashSim 是宾夕法尼亚州立大学开发的固态硬盘仿真器，它是对 DiskSim① 的扩展，即在其基础上添加闪存仿真模块和闪存与上层的接口模块. 我们在 FlashSim 的基础上，通过修改底层闪存模块和开发针对混合 SSD 的 FTL 算法，实现混合 SSD 架构. 实验中，混合 SSD 的最大总容量为 2.5 GB，SLC 区域设置为 32 MB～512 MB，MLC 区域设为 2 GB，闪存的关键参数设置如表 1 所示. 数据分配机制中阈值 θ 的初始值设为 4 KB，$\Delta\theta$ 值设为 4 KB.

表 1 SLC 闪存与 MLC 闪存的配置参数

Flash type	SLC	MLC
Page size	2 KB	4 KB
Block size	128 KB	512 KB
Page read	25 μs	60 μs
Page write	200 μs	800 μs
Block erase	1.5 ms	1.5 ms
Block Endurance	100 k cycles	10 k cycles

本文主要选取 3 个性能指标——混合 SSD 的平均响应时间、等效总擦除次数、磨损均衡程度来评价 FTL 算法的性能，具体解释如下：

(1) 平均响应时间是指请求完成时间与到达时间的时间差，它是垃圾回收、地址翻译的开销及数据访问时间的综合，是衡量混合 SSD 读写性能的关键指标.

(2) 等效总擦除次数指的是将 MLC 区域的擦除次数等效成的 SLC 擦除次数，再加上 SLC 区域的擦除次数，它直接反映混合 SSD 的使用寿命.

(3) 磨损均衡程度用来衡量 SLC 与 MLC 的磨损均衡效果，它定义为 $\Phi=\max(RW_s,RW_m)/\min(RW_s,RW_m)$. $\Phi=1$ 代表 SLC 和 MLC 的磨损速率一致，此时为理想的磨损均衡. 实际中，Φ 越接近 1 则磨损均衡效果越好.

实验中总共使用了 6 个负载，它们可以分为 3 类：企业级负载，DiskSim 产生的合成负载，以及用磁盘驱动数据跟踪器 DiskMon 收集的个人 PC 的负载. 我们主要集中于写请求，这是因为由表 1 可知，SLC 闪存和 MLC 闪存的读延迟性能差异不大. Fin1② 来自 OLTP 应用，是金融机构处理在线联机事务的负载. Sys、Iotrace1 和 Iotrace2 是由 DiskMon 收集的负载，这些负载也是真实负载. Test1 和 Test2 是由 DiskSim 合成的负载. 表 2 给出了实验中使用的负载的特性.

表 2 实验中 trace 的特性

负载类型	请求数量	写请求比例/%	平均请求大小/KB
Fin1	5344983	76.84	3.38
Sys	3698864	50.1	4.78
Test1	3181345	50.0	2.67
Test2	1644566	90.1	10.4984
Iotrace1	1846732	83.2	8.4967
Iotrace2	90333809	53.4	5.9505

4.2 WLAFTL 性能分析

(1) 延迟迁移策略的 *Max_Cycle_Time* 设置

为研究 *Max_Cycle_Time* 值大小的配置问题，我们设计了一组仿真实验，取 *Max_Cycle_Time* 值为 1～5. 仿真 trace 选择具有代表性的 Fin1，仿真结果如图 4 所示. 结果表明，随着 *Max_Cycle_Time* 值的增大，Fin1 的平均响应时间先减少后增大，当 *Max_Cycle_Time* 值取为 3 时，整体性能达到最优；另外几组 trace 也有相似的结果. 因此，后续实验中我们取 *Max_Cycle_Time* 值为 3.

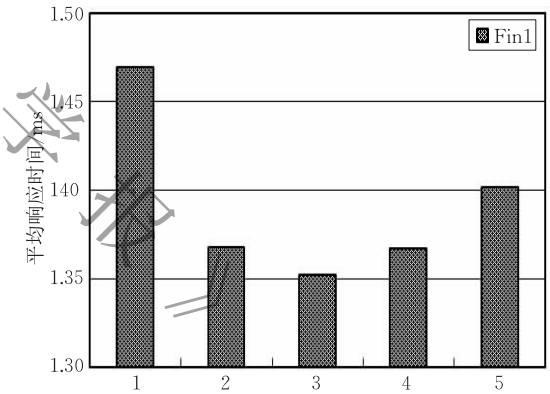


图 4 不同 *Max_Cycle_Time* 值时的平均响应时间

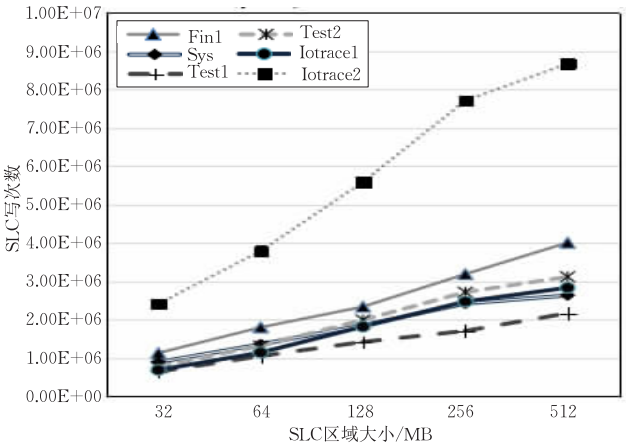
(2) SLC 区域大小对整体性能的影响

为研究 SLC 区域大小对混合固态硬盘整体性能的影响，本实验分别设置 SLC 区域大小为 32 MB、64 MB、128 MB、256 MB、512 MB，并且对 6 组 trace 进行实验，仿真结果图 5 所示.

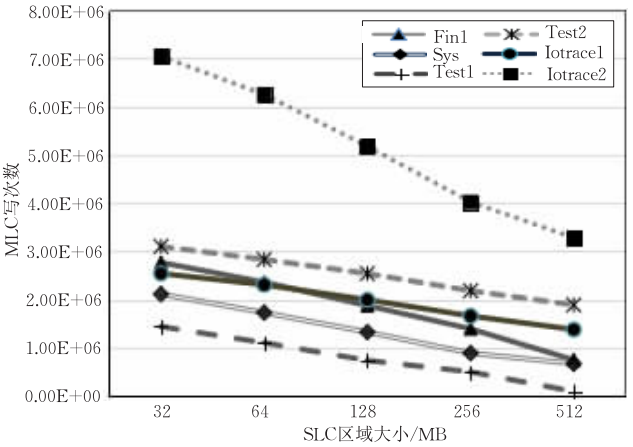
从图 5(a)、(b)可以看到，随着 SLC 区域容量不断增大，SLC 区域的写次数不断增加，而 MLC 区域

① Bucy J S, Ganger G R, et al. The DiskSim Simulation Environment Version 3.0 Reference Manual. [2017-7-14]. http://studylib.net/doc/10590244/the-disksim-simulation-environment-version-3.0-reference-…

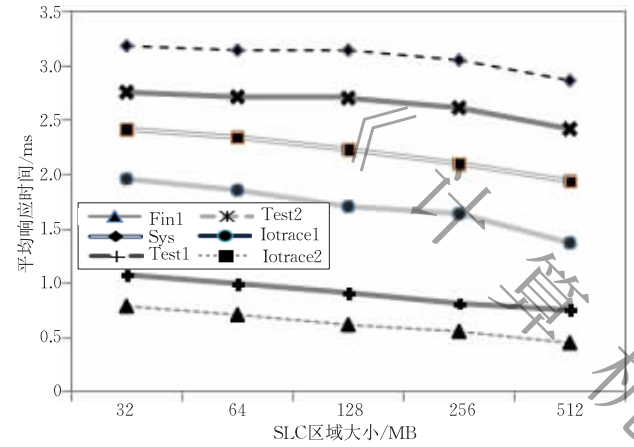
② Liberate M, Shenoy P. OLTP Trace. UMass Trace Repository. [2015-05-09]. http://traces.cs.umass.edu/index.php/storage/storage



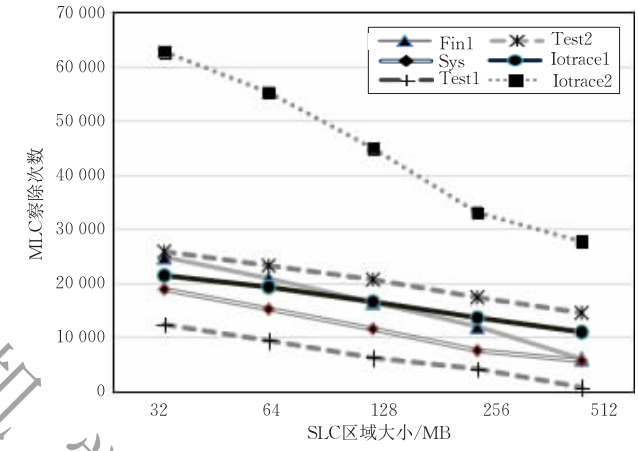
(a) SLC区域不同大小情况下SLC的写次数



(b) SLC区域不同大小情况下MLC的写次数



(c) SLC区域不同大小情况下的平均响应时间



(d) SLC区域不同大小情况下的MLC擦除次数

图 5　不同 SLC 区域大小下的性能比较

的写次数不断减少，即热数据更多地被读写性能更好的 SLC 区域处理，减少了 MLC 区域的写操作和擦除次数，这使得混合 SSD 的平均响应时间和 MLC 擦除次数都随着 SLC 区域容量不断增大而减小，如图 5(c)、(d)所示，从而改善混合 SSD 的读写性能和使用寿命.

(3) 动态阈值调整机制对整体性能的影响

为比较静态阈值与基于损耗均衡的动态阈值调整性能的差异，我们进行了 24 组实验. 根据表 2 中负载的特点，在实验中静态阈值依次设为 4 KB、8 KB、16 KB，性能指标选择归一化平均响应时间和 SLC 与 MLC 区域的磨损速率比 Φ，仿真结果如图 6 和表 3 所示.

由图 6 可以看出，相比于固定的 4 KB、8 KB、16 KB 阈值大小，WLAFTL 提出的动态阈值调整机制的平均响应时间总体性能较为稳定，对各种负载都具有较大适应性. 图 6 与表 3 中比较特殊的是负载 Test1，在本次实验中它没有受到阈值变化的影响；原因在于 Test1 95%以上的写请求小于 4 KB，

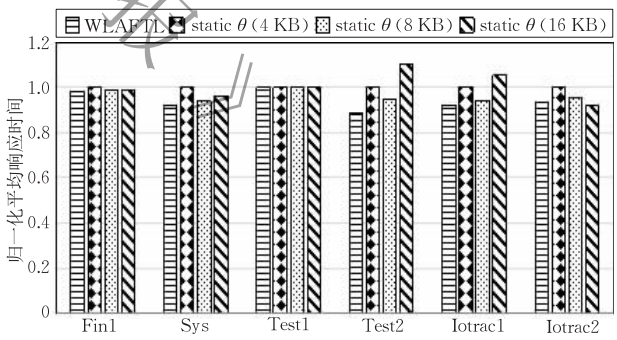


图 6　固定阈值和动态阈值在各 trace 下的平均响应时间

造成阈值以 4 KB 变化对数据分配没影响. 此外，根据表 3 数据，在阈值分别为 4 KB、8 KB、16 KB 和动态调整的情况下，磨损均衡程度 Φ 的均值分别为 11.52、2.69、1.74、1.59；WLAFTL 的动态阈值调整机制实现的 SLC 和 MLC 之间的磨损均衡效果最好. 需要指出的是，虽然阈值为 16 KB 时其磨损均衡程度与自适应动态调整时接近，但是此时的平均响应时间指标是明显差于 WLAFTL.

表 3 动态数据机制磨损均衡比较

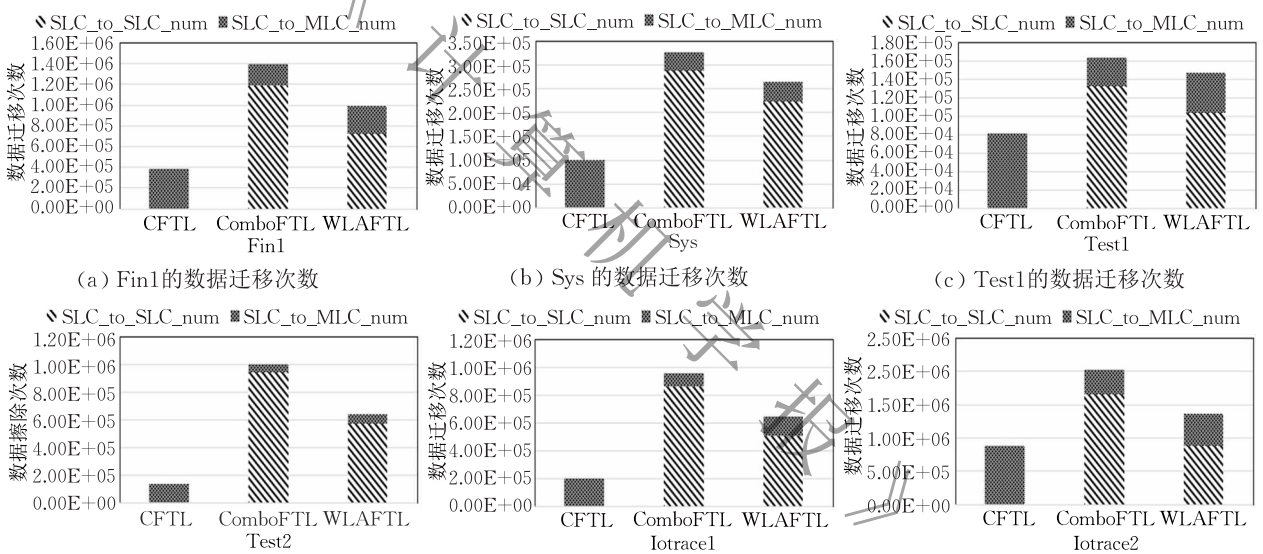
	Static threshold(4 KB)			Static threshold(8 KB)			Static threshold(16 KB)			WLAFTL		
	SLC擦除次数	MLC擦除次数	磨损均衡程度 Φ	SLC擦除次数	MLC擦除次数	磨损均衡程度 Φ	SLC擦除次数	MLC擦除次数	磨损均衡程度 Φ	SLC擦除次数	MLC擦除次数	磨损均衡程度 Φ
Fin1	82298	4602	**1.79**	82376	4561	**1.81**	82376	4561	**1.81**	78139	5154	**1.51**
Sys	30652	9452	**3.08**	39838	6181	**1.55**	44316	5083	**1.14**	46348	4763	**1.02**
Test1	32431	938	**3.45**	32431	938	**3.45**	32431	938	**3.45**	32431	938	**3.45**
Test2	8369	28227	**33.73**	43478	19094	**4.39**	94706	8353	**1.13**	96095	8636	**1.11**
Iotrace1	10237	21175	**20.68**	51160	11642	**2.27**	82076	7501	**1.09**	66706	7501	**1.12**
Iotrace2	77801	49985	**6.42**	132222	35171	**2.66**	154795	28776	**1.86**	190450	24788	**1.30**

4.3 WLAFTL 与 CFTL、ComboFTL 性能对比

(1)数据迁移机制的性能对比分析

在不同负载下,其它机制保持一致,仅仅比较 CFTL、ComboFTL 和 WLAFTL 提出的数据迁移机制的性能差异.图 7 比较了这 3 种 FTL 的迁移机制的数据迁移情况,实验数据表明:在 SLC 内部数据迁移开销方面,CFTL 无额外开销,ComboFTL 开销最多,WLAFTL 处于中间;在 SLC 向 MLC 数据迁移开销方面,CFTL 最多、ComboFTL 最少、WLAFTL 处于中间.造成这样结果的原因在于:

①CFTL 直接将 SLC 循环队列的 tail 块中的有效数据迁移到 MLC,因此无 SLC 内部数据迁移开销,但也带来数据从 SLC 向 MLC 迁移过多和降低数据在 SLC 中得到服务的概率的两个问题.



(a) Fin1的数据迁移次数 (b) Sys 的数据迁移次数 (c) Test1的数据迁移次数

(d) Test2的数据迁移次数 (e) Iotrace1的数据迁移次数 (f) Iotrace2的数据迁移次数

图 7 不同数据迁移机制的数据迁移情况

②WLAFTL 和 ComboFTL 都避免了 CFTL 这方面的不足,但是 ComboFTL 增加了 SLC 内部 hot 区向 warm 区的迁移开销.

③WLAFTL 因为有两种迁移策略存在,因此既能够在$[RW_s]>[RW_m]$的情况下,及时将 SLC 区域的冷数据迁移到 MLC 区域,降低 SLC 的磨损速率,又能在$[RW_s]\leq[RW_m]$的情况下,延迟将 SLC 区域的冷数据迁移到 MLC 区域,提高数据在 SLC 中得到服务的概率.

表 4 比较了这 3 种 FTL 的迁移机制的擦除次数和磨损均衡情况.实验数据表明,平均而言,相比 ComboFTL 和 CFTL,WLAFTL 的等效总擦除次数分别减少 8.6%和 12.9%;具体而言,ComboFTL 在 Sys 和 Test1 最优,CFTL 在 Fin1 最优,WLAFTL 在 Test2、Iotrace1 和 Iotrace2 最优;此外,WLAFTL 没有取得最差的情况出现.磨损均衡方面,平均而言,ComboFTL、CFTL 和 WLAFTL 的 Φ 的均值依次为 3.12、2.61、1.59,WLAFTL 的磨损均衡效果最好;具体而言,WLAFTL 在 Sys 和 Test2、CFTL 在 Fin1 和 Test1、ComboFTL 在 Iotrace1 和 Iotrace2 取得最优的磨损均衡;此外,WLAFTL 没有取得最差的情况出现.表 4 数据还表明,CFTL 的迁移机制会造成 MLC 磨损过快,因为处于这 6 个负载下,MLC 的等效擦除次数实质上都是大于 SLC 的擦除次数.ComboFTL 在 Test1 负载下出现特殊情况,其 N 次迁移策略造成大部分热数据一直在 SLC,使得 SLC

表 4　数据迁移机制擦除次数与磨损均衡比较

	ComboFTL				CFTL				WLAFTL			
	SLC 擦除次数	MLC 擦除次数	等效总擦除次数 E_{total}	磨损均衡程度 Φ	SLC 擦除次数	MLC 擦除次数	等效总擦除次数 E_{total}	磨损均衡程度 Φ	SLC 擦除次数	MLC 擦除次数	等效总擦除次数 E_{total}	磨损均衡程度 Φ
Fin1	94199	4570	139899	2.06	54276	7296	**127236**	**1.34**	78139	5154	129679	1.52
Sys	45343	4797	**93313**	1.05	31121	9141	122531	2.94	46348	4763	93978	**1.03**
Test1	34810	288	**37690**	12.09	21533	4489	66423	**2.08**	32431	938	40741	3.46
Test2	93422	12328	216702	1.31	49850	17486	178003	3.51	96095	8636	**182455**	**1.11**
Iotrace1	82762	8852	171282	**1.07**	40283	13772	178003	3.42	66706	7501	**141716**	1.11
Iotrace2	215104	25143	466534	**1.17**	137335	32496	462295	2.36	190450	24788	**438330**	1.30

的磨损速率远高于 MLC. WLAFTL 的迁移策略实质是根据 SLC 和 MLC 的磨损均衡情况来选择 CFTL 或 ComboFTL 的迁移策略，因而能避免极端情况的出现，实现较好的磨损均衡. 下面以 Fin1 负载为例，具体分析这 3 种算法的性能. Fin1 负载下，WLAFTL 和 CFTL 均最终处于$[RW_s]=[RW_m]$状态，与 CFTL 相比，WLAFTL 的 MLC 擦除次数大约减少 29.3%，SLC 擦除次数大约增加 43.9%，SLC 的擦除次数增长比例高于 MLC 的擦除次数减少的比例，因此 WLAFTL 磨损均衡程度比 CFTL 差；但是由于有更多的数据被分配到 SLC，造成 WLAFTL 的时间性能指标优于 CFTL. ComboFTL 最终处于$[RW_s]>[RW_m]$状态，其磨损均衡情况最差；此外，由于其 SLC 内部存在的大量数据迁移，导致其时间性能指标也差于 WLAFTL.

图 8 比较了这 3 种 FTL 的迁移机制的归一化平均响应时间情况. 实验结果表明，平均而言，相比 ComboFTL 和 CFTL，WLAFTL 的数据迁移机制使得平均响应时间分别有 10.6%和 10.4%的改善；具体而言，WLAFTL 在这 6 个负载中都取得最优结果，CFTL 在 Sys 和 Test1 负载得到的平均响应时间差于 ComboFTL，其它 4 个负载优于 ComboFTL. ComboFTL 在 Sys 和 Test1 负载下可以取得最优的等效擦除次数从一个方面解释了其平均响应时间优于 CFTL 的原因，另一个原因在于，Sys 和 Test1 负载写请求比例小，同样处于$[RW_s]=[RW_m]$情况下时，ComboFTL 迁移机制延长热数据在 SLC 停留时间，提高其被 SLC 服务的概率，从而提高整体性能.

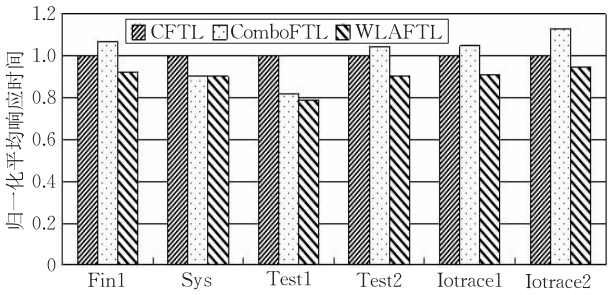


图 8　不同数据迁移机制的归一化平均响应时间

(2) 数据分配机制的性能对比分析

在不同负载下，其它机制保持一致，仅仅比较 CFTL、ComboFTL 和 WLAFTL 提出的数据分配机制的性能差异. 归一化平均响应时间如图 9 所示，磨损均衡程度比较如表 5 所示.

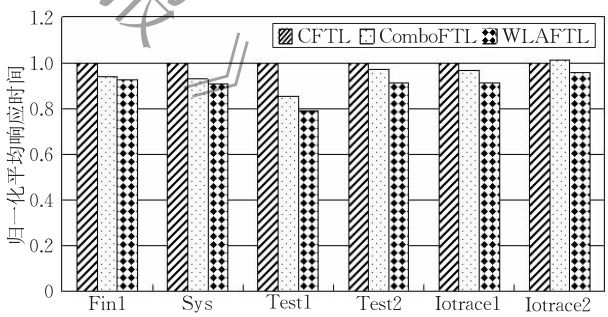


图 9　不同数据分配机制的归一化平均响应时间

表 5　数据分配机制磨损均衡比较

	ComboFTL				CFTL				WLAFTL			
	SLC 擦除次数	MLC 擦除次数	等效总擦除次数 E_{total}	磨损均衡程度 Φ	SLC 擦除次数	MLC 擦除次数	等效总擦除次数 E_{total}	磨损均衡程度 Φ	SLC 擦除次数	MLC 擦除次数	等效总擦除次数 E_{total}	磨损均衡程度 Φ
Fin1	66425	6405	130475	**1.04**	53440	9584	149280	1.79	78139	5154	**129679**	1.52
Sys	48531	4316	**91691**	1.12	30679	8661	117289	2.82	46348	4763	93978	**1.03**
Test1	33645	712	40765	4.73	21159	4407	65229	**2.08**	32431	938	**40741**	3.46
Test2	53173	16711	220283	3.14	45122	16602	211142	3.68	96095	8636	**182455**	**1.11**
Iotrace1	47545	12083	168375	2.54	39231	13354	172771	3.40	66706	7501	**141716**	**1.11**
Iotrace2	178212	25264	**430852**	1.42	136456	30819	444646	2.26	190450	24788	438330	**1.30**

由图 9 可以看出，相比于 ComboFTL 和 CFTL，WLAFTL 中的数据分配机制使得平均响应时间平均分别有 4.5%和 10.1%的改善. 由表 5 可以得出，相比于 ComboFTL 和 CFTL，WLAFTL 的总擦除次数分别减少 5.1%和 11.4%. 磨损均衡方面，ComboFTL、CFTL和WLAFTL 的 Φ 的均值分别为 2.33、2.67 和 1.59，这说明 WLAFTL 的磨损均衡效果更佳. 图 9 数据还表明，CFTL 的调整热数据识别阈值的方法是差于ComboFTL和WLAFTL. 这是因为实验中使用的 6 组 trace 中大部分负载的请求分布没有出现双峰值的现象，因此基于二均值聚类的数据分类机制不能很好地适应负载的变化. 由此可以看出，WLAFTL 的数据分配机制不仅能改善混合 SSD 的平均响应时间，还能延长使用寿命，SLC 和 MLC 之间的磨损均衡控制效果也更加理想.

(3) 整体性能比较

在整体性能对比时，为客观评价本文提出的数据分配和迁移机制性能，我们使 ComboFTL、CFTL 和 WLAFTL 的地址映射机制保持一致，即 SLC 区域采用纯页级映射，MLC 区域采用 SDFTL，仅比较三者的数据分配和迁移机制对混合 SSD 的性能影响，归一化平均响应时间如图 10 所示，磨损均衡效果的比较如表 6 所示.

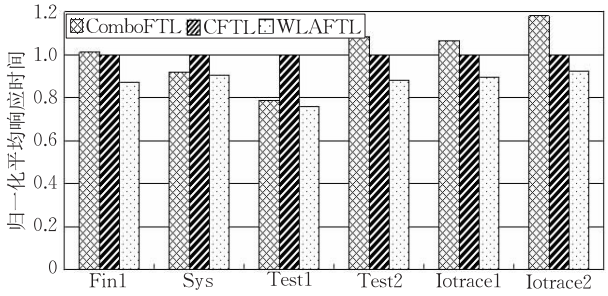


图 10 不同 trace 下的归一化平均响应时间

表 6 整体磨损均衡比较

	ComboFTL				CFTL				WLAFTL			
	SLC擦除次数	MLC擦除次数	等效总擦除次数 E_{total}	磨损均衡程度 Φ	SLC擦除次数	MLC擦除次数	等效总擦除次数 E_{total}	磨损均衡程度 Φ	SLC擦除次数	MLC擦除次数	等效总擦除次数 E_{total}	磨损均衡程度 Φ
Fin1	95745	4295	138695	**2.23**	58660	8382	142480	**1.42**	78139	5154	**129679**	**1.52**
Sys	57325	2813	**85455**	**2.03**	30679	8661	117289	**2.82**	46348	4763	93978	**1.03**
Test1	34810	288	**37690**	**12.09**	19360	5368	73040	**2.77**	32431	938	40741	**3.46**
Test2	81427	15245	233877	**1.87**	31457	21955	251007	**6.98**	96095	8636	**182455**	**1.11**
Iotrace1	74736	10568	180416	**1.41**	26698	16907	195768	**6.33**	66706	7501	**141716**	**1.11**
Iotrace2	258530	19799	456520	**1.31**	113586	39895	512536	**3.51**	190450	24788	**438330**	**1.30**

表 6 比较了这 3 种 FTL 的擦除次数和磨损均衡情况. 实验数据表明，平均而言，相比 ComboFTL 和 CFTL，WLAFTL 的等效总擦除次数分别减少 9.2%和 12.7%%；具体而言，ComboFTL 在 Sys 和 Test1 最优，WLAFTL 在 Fin1、Test2、Iotrace1 和 Iotrace2 最优；磨损均衡方面，平均而言，ComboFTL、CFTL 和 WLAFTL 的 Φ 的均值依次为 3.49、3.97 和 1.59，WLAFTL 的磨损均衡效果最好；具体而言，WLAFTL 在 Sys、Test2、Iotrace1 和 Iotrace2CFTL 在 Fin1 和 Test1、ComboFTL 在 Iotrace1 和 Iotrace2 取得最优的磨损均衡；此外，WLAFTL 没有取得最差的情况出现.

图 10 比较了这 3 种 FTL 的归一化平均响应时间情况. 实验结果表明，平均而言，相比 ComboFTL 和 CFTL，WLAFTL 的数据迁移机制使得平均响应时间分别有 13.6%和 12.7%的改善；具体而言，WLAFTL 在这 6 个负载中都取得最优结果，CFTL 在 Sys 和 Test1 负载得到的平均响应时间差于 ComboFTL，其它 4 个负载优于 ComboFTL. 由表 6 数据可以得出，在 Test2、Iotrace1 和 Iotrace2 负载情况下，与 CFTL 相比，ComboFTL 的 SLC 擦除次数分别增加了 158%、180%和 127%，而 MLC 擦除次数只是分别减少了 30.5%、36.9%和 50.4%，SLC 区域磨损程度的增加却对 MLC 区域的性能提升不大，数据表明，在这 3 种负载下，ComboFTL 中 SLC 区域的垃圾回收代价过大，严重影响了 ComboFTL 的整体性能；而在 Sys 和 Test1 负载情况下，与 CFTL 相比，ComboFTL 的 SLC 擦除次数分别增加了 86.8%和 79.8%，而 MLC 擦除次数却分别减少了 67.5%和 94.6%，合理地延长热数据在 SLC 停留的时间，提高其被 SLC 服务的概率，提高了性能；在 Fin1 负载情况下，与 CFTL 相比，ComboFTL 的 SLC 擦除次数增加了 63%，而 MLC 擦除次数减少了 48.7%，通过合理地延长热数据在 SLC 停留的时间，提高其被 SLC 服务的概率，从而削弱了 SLC 内部 hot 区向 warm 区的迁移开销对性能的影响，使得性能与 CFTL 接近.

5 结　论

本文针对 SLC＋MLC 类型的混合固态硬盘，提出一种具有磨损均衡意识的混合固态硬盘 FTL 算法——WLAFTL. WLAFTL 以磨损均衡为指导思想，解决了混合 SSD 的 SLC、MLC 数据分配问题和 SLC 冷数据回收/迁移问题. 一方面，提出一种动态的基于磨损均衡思想和请求大小融合的 SLC、MLC 数据分配机制，即根据 SLC 和 MLC 的磨损速率来动态调整热数据识别阈值，最后将小于阈值的写请求分配到 SLC 区域、大于阈值的写请求分配到 MLC 区域. 另一方面，提出一种基于磨损均衡思想和 FIFO 机制融合的 SLC 冷数据回收/迁移机制，根据 SLC、MLC 磨损速率选择延迟迁移策略或正常迁移策略，实现更有效的 SLC 到 MLC 的数据迁移. 实验结果表明，与 ComboFTL 和 CFTL 相比，在使用相同地址映射机制的条件下，WLAFTL 的平均响应时间分别平均有 13.6%和 12.7%的改善，总的擦除次数分别平均减少 9.2%和 20.4%，同时能够更好地平衡 SLC 和 MLC 之间的磨损均衡. 但 WLAFTL 仍有可改进之处，例如可以根据 SLC 和 MLC 的磨损均衡情况自适应设置 Max_Cycle_Time 和 $\Delta\theta$ 的值，或采用机器学习方法来设计更加精确地冷热数据区分机制，包括混合 SSD 中 SLC 和 MLC 的最佳构成比例等，这些问题都留待下一阶段的研究工作进行解决.

参　考　文　献

[1] Zheng Wen-Jing, Li Ming-Qiang, Shu Ji-Wu. Storage memory technology of Flash. Journal of Computer Research and Development, 2010, 47(4): 716-726(in Chinese)
(郑文静，李明强，舒继武. Flash 存储技术. 计算机研究与发展，2010，47(4)：716-726)

[2] Pavan P, Bez R, Olivo P, et al. Flash memory cells—An overview. Proceedings of the IEEE, 1997, 85(8): 1248-1271

[3] Hui S, Rui Z, Jin C, et al. Analysis of file system and block I/O scheduler for SSD in performance and energy consumption //Proceedings of the 2011 IEEE Asia-Pacific Services Computing Conference. Jeju Island, Korea, 2011: 48-55

[4] D'Abreu M. NAND flash memory—Product trends, technology overview, and technical challenges//Proceedings of the 2011 Asian Test Symposium. New Delhi, India, 2011: 463-463

[5] Lin M, Yao Z. Dynamic garbage collection scheme based on past update times for NAND flash-based consumer electronics. IEEE Transactions on Consumer Electronics, 2015, 61(4): 478-483

[6] Gupta A, Kim Y, Urgaonkar B. DFTL: A flash translation layer employing demand-based selective caching of page-level address mappings//Proceedings of the 14th International Conference on Architectural Support for Programming Languages and Operating Systems, Washington, USA, 2009: 229-240

[7] Jiang S, Zhang L, Yuan X, et al. S-FTL: An efficient address translation for flash memory by exploiting spatial locality//Proceedings of the of the 2011 IEEE 27th Symposium on Mass Storage Systems and Technologies (MSST). Denver, USA, 2011: 1-12

[8] Yao Ying-Biao, Shen Zuo-Bing. An improved DFTL algorithm based on sequential cache and second level cache. Journal of Computer Research and Development, 2014, 51(9): 2012-2021(in Chinese)
(姚英彪，沈佐兵. 基于连续缓存和二级缓存的 DFTL 改进算法. 计算机研究与发展，2014，51(9)：2012-2021)

[9] Lee Y, Jung T, Shin I. Demand-based flash translation layer considering spatial locality//Proceedings of the 28th Annual ACM Symposium on Applied Computing. New York, USA, 2013: 1550-1551

[10] Xie Xu-Chao, Song Zhen-Long, Li Qiong, et al. WAPFTL: A workload adaptive page-level flash translation layer with prediction. Computer Engineering and Science, 2014, 36(7): 1238-1243(in Chinese)
(谢徐超，宋振龙，李琼等. WAPFTL：支持预测机制的负载自适应闪存转换层算法. 计算机工程与科学，2014，36(7)：1238-1243)

[11] Zhou Y, Wu F, Huang P, et al. An efficient page-level FTL to optimize address translation in flash memory//Proceedings of the 10th European Conference on Computer Systems. Bordeaux, France, 2015: 12

[12] Chen R, Qin Z, Wang Y, et al. On-demand block-level address mapping in large-scale NAND flash storage systems. IEEE Transactions on Computers, 2015, 64(6): 1729-1741

[13] Wang Y, Qin Z, Chen R, et al. A real-time flash translation layer for NAND flash memory storage systems. IEEE Transactions on Multi-Scale Computing Systems, 2016, 2(1): 17-29

[14] Lee S W, Park D J, Chung T S, et al. A log buffer-based flash translation layer using fully associative sector translation. ACM Transactions on Embedded Computing Systems, 2007, 6(3): 1-27

[15] Kim J, Kim J M, Noh S H, et al. A space-efficient flash translation layer for CompactFlash systems. IEEE Transactions on Consumer Electronics, 2002, 48(2): 366-375

[16] Lee S, Shin D, Kim Y. LAST: Locality-aware sector translation for NAND flash memory-based storage systems. ACM SIGOPS Operating Systems Review, 2008, 42(6): 36-42

[17] Tang Xian, Meng Xiao-Feng. FClock: An adaptive buffer replacement algorithm for SSD. Chinese Journal of Computers, 2010, 33(8): 1460-1471(in Chinese)

(汤显，孟小峰. FClock：一种面向 SSD 的自适应缓冲区管理算法. 计算机学报，2010，33(8)：1460-1471)

[18] Lin Zi-Yu, Lai Ming-Xing, Zou Quan, et al. Probability-based buffer replacement algorithm for flash-based databases. Chinese Journal of Computers, 2013, 36(8): 1568-1581(in Chinese)
(林子雨，赖明星，邹权等. 基于替换概率的闪存数据库缓冲区替换算法. 计算机学报，2013，36(8)：1568-1581)

[19] Wei Q, Chen C, Yang J. CBM: A cooperative buffer management for SSD//Proceedings of the 30th International Conference on Massive Storage Systems and Technology (MSST 2014). Santa Clara, USA, 2014: 1-12

[20] Jiang Z, Zhang Y, Wang J, et al. A cost-aware buffer management policy for flash-based storage devices//Proceedings of the 20th International Conference on Database Systems for Advanced Applications (DASFAA 2015). Hanoi, Vietnam, 2015: 175-190

[21] Jimenez X, Novo D, Ienne P. Wear unleveling: Improving NAND flash lifetime by balancing page endurance//Proceedings of the 12th USENIX Conference on File and Storage Technologies (FAST). Santa Clara, USA, 2014: 47-59

[22] Pan Y, Dong G, Zhang T. Error rate-based wear-leveling for NAND flash memory at highly scaled technology nodes. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2013, 21(7): 1350-1354

[23] Woo Y J, Kim J S. Diversifying wear index for MLC NAND flash memory to extend the lifetime of SSDs//Proceedings of the 11th ACM International Conference on Embedded Software. Montreal, Canada, 2013: 6

[24] Liao J, Zhang F, Li L, et al. Adaptive wear-leveling in flash-based memory. IEEE Computer Architecture Letters, 2015, 14(1): 1-4

[25] Im S, Shin D. ComboFTL: Improving performance and lifespan of MLC flash memory using SLC flash buffer. Journal of Systems Architecture, 2010, 56(12): 641-653

[26] Lee S, Kim J. Improving performance and capacity of flash storage devices by exploiting heterogeneity of MLC flash memory. IEEE Transactions on Computers, 2014, 63(10): 2445-2458

[27] Chang L P. A hybrid approach to NAND-flash-based solid-state disks. IEEE Transactions on Computers, 2010, 59(10): 1337-1349

[28] Kwon S J, Chung T S. Data pattern aware FTL for SLC+MLC hybrid SSD. Design Automation for Embedded Systems, 2015, 19(1): 101-127

[29] Park J W, Park S H, Weems C C, et al. A hybrid flash translation Layer design for SLC-MLC flash memory based multibank solid state disk. Microprocessors and Microsystems, 2011, 35(1): 48-59

[30] Park S H, Park J W, Jeong J M, et al. A mixed flash translation layer structure for SLC-MLC combined flash memory system//Proceedings of the 1st International Workshop on Storage and I/O Virtualization, Performance, Energy, Evaluation and Dependability (SPREED2008), Salt Lake City, USA, 2008: 1-12

[31] Lee S, Ha K, Zhang K, et al. FlexFS: A flexible flash file system for MLC NAND flash memory//Proceedings of the USENIX Annual Technical Conference. San Diego, USA, 2009: 1-14

[32] Chang L P. Hybrid solid-state disks: Combining heterogeneous NAND flash in large SSDs//Proceedings of the 13th Asia South Pacific Design Automation Conference. Seoul, Korea, 2008: 428-433

[33] Kim Y G, Tauras B D, Gupta A, et al. FlashSim: A simulator for NAND flash-based solid-state drives//Proceedings of the 1st International Conference on Advances in System Simulation. Porto, Portugal, 2009: 125-131

YAO Ying-Biao, born in 1976, Ph. D., professor. His current research interests include storage system design, wireless sensor networks and media signal processing.

WANG Fa-Kuan, born in 1991, M. S., engineer. His current research interest is SSD techniques.

Background

Due to nonvolatile, shock resisting and low power consuming characteristics, the NAND flash based SSD (Solid State Disk) begins to take place of traditional magnetic hard disk drive in many domains. NAND flash has three different operations: read, write, and erase; read/write operations are performed in page units, and erase operations are performed in block units. NAND flash has two types of physical restrictions: erase-before-write restrictions and a limited number of erasures. FTL (Flash Translation Layer) played an important role in hiding the physical nature of NAND flash, and it contains

address mapping, garbage collection and wear-leveling.

At present, the physical storage medium of SSD is mainly based on homogeneous SLC (Single-Level-Cell) or MLC (Multi-Level-Cell) flash chips. Although SLC chips have relatively superior performance, longer life time and durability, they are also more expensive than MLC chips. Taking the advantages of both SLC and MLC into consideration, academics have proposed the hybrid SSD, consisting of both SLC and MLC flash chips, to achieve the balance between performance, lifetime, and cost by allocating hot data and cold data to SLC and MLC, respectively. This paper mainly investigate the problem how to solve the data allocation and wear leveling between SLC region and MLC region, because they are important issues that affects the performance, lifetime, and cost of hybrid SSD.

In this paper, we proposes a wear leveling aware flash translation layer (WLAFTL) for hybrid SSDs. Firstly, a dynamic data allocation scheme which is based on the integration of wear leveling and write-request size is proposed. Specifically, it adaptively adjust the threshold value of cold/hot data identification according to the wear speed of SLC and MLC region, and then, the small write requests whose size is less than the threshold value are send to SLC region, while the large write requests are send to MLC region. Secondly, a cold data's garbage collection/migration scheme in SLC which is based on the integration of wear leveling and first in first out (FIFO) is proposed. This scheme can reduce the number of data migration from SLC to MLC region. Our benchmark results demonstrate that under the condition of using the same address mapping scheme, compared to CFTL, WLAFTL can achieve up to 9.09% average response time improvement and 35.23% block erasure reduction of MLC region. Moreover, it can achieve better wear leveling between SLC and MLC than CFTL.

This work was supported in part by the National Natural Science Foundation of China (No. 61100044 and No. 61671192), Zhejiang Province Science and Technology Innovation Focused Team Foundation (No. 2013TD03), and the Top-ranking Discipline A Class of Electronics Science and Technology in Zhejiang Province. Our research group has worked in the area of design and implementation of NAND flash based SSD for many years and published many papers and patents for invention in this area.