|  |  |
| --- | --- |
| 学号 | 2017282110256 |
| 密级 |  |

**F2FS A New File System for Flash Storage**

**读书报告**

|  |
| --- |
|  |

|  |  |
| --- | --- |
| 院（系）名 称 ： | 计算机学院 |
| 专 业 名 称 ： | 计算机技术 |
| 学 生 姓 名 ： | 白春飞 |
|  |  |

二〇一七年十二月

摘 要

作者在论文中介绍了其对于 Flash SSD 的特点与传统文件系统实现的差异，然后横向比较了与 Ext4(Update in Place)、Btrfs(Copy on Write) 和 NILFS2(LFS) 的性能差异。介绍了主要的设计思路，数据结构，算法所得F2FS的性能。并且对该文件系统进行了实验并横向与其它文件系统做了对比。

作者从其对特性的论述中如 Flash SSD 友好的 on-disk 格式，在更新数据时减小对于元数据更新的损耗(减小随机小 IO)，用多个 logging 流来提高并发能力，同时支持两种写日志方式(Append-only 和 Write in Hole)来解决高利用率时前者性能雪崩问题，最后还有面向大量使用 fsync 程序如 SQLite 的 fsync 调用优化。

总的而言，F2FS 采用了空间换时间的策略，在实现上对于潜在的大量随机 IO 等待大量使用 Lazy 的策略处理，同时因为采用 Lazy 的原因对于在高利用率下的压测可能造成 Cleaning 延时的问题上，保留了 5% 的预留空间来避免潜在的雪崩情况。

关键词： F2FS；文件系统; 闪存

目录

[1 论文背景和作者所做的工作 4](#_Toc500432745)

[1.1 论文背景 4](#_Toc500432746)

[1.2 作者所做的工作 4](#_Toc500432747)

[1.3 F2FS文件系统简介 4](#_Toc500432748)

[2 F2FS设计和F2FS的实现 5](#_Toc500432749)

[2.1 磁盘布局 5](#_Toc500432750)

[2.2 文件结构 6](#_Toc500432751)

[2.3 目录结构 7](#_Toc500432752)

[2.4 多记录头 7](#_Toc500432753)

[2.5 擦除 8](#_Toc500432754)

[2.6 自适应记录 8](#_Toc500432755)

[2.7 检查点和恢复 9](#_Toc500432756)

[3 实验测试设置、结果与相关讨论 11](#_Toc500432757)

[3.1 实验设置 11](#_Toc500432758)

[3.2 文件系统性能测试结果 11](#_Toc500432759)

[3.3 相关讨论 12](#_Toc500432760)

[4 综述与收获 14](#_Toc500432761)

# 论文背景和作者所做的工作

该篇论文发表在15年的fast会议上，作者是走在存储领域前列的三星公司的研究员Changman Lee, Dongho Sim, Joo-Young Hwang, and Sangyeun Cho。

## 论文背景

NAND闪存设备广泛应用于移动设备。服务器也系统开始使用闪存设备作为其主要存储。尽管它的广泛使用，闪存具有若干限制，像擦除beforewrite要求，需要写上顺序地擦除的块和每个擦除块有限写入周期。NAND闪存放在一个平台上。随着存储需求的生长，但是，它是日益普遍使用的“解决方案”，其具有通过专用控制器连接的多个闪存芯片。控制器，通常称为FTL（快闪转换层）上运行的固件，解决了NAND​​快闪存储器的限制，并提供了一个通用块设备抽象。这样的闪存存储解决方案的例子包括的eMMC（嵌入式多媒体卡），UFS（通用闪存）和SSD（固态驱动器）。通常情况下，这些现代闪存存储设备会显示出比硬盘驱动器（HDD），其机械同行低得多的存取延迟。当涉及到随机I / O，固态硬盘进行数量级比HDD更好。他们不考虑闪存存储设备的特点，在性能和设备寿命方面难免不理想。传统的文件系统设计策略硬盘，虽然有益，达不到充分利用和优化的NAND闪存介质的使用。

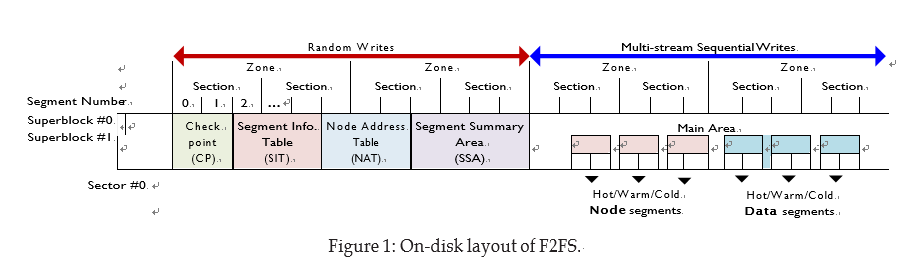
## 作者所做的工作

作者设计和实现了F2FS，现代闪存存储设备而优化的新的文件系统。F2FS是从头设计一个通用的块接口以优化性能和闪存设备的寿命的第一个公开和广泛使用的文件系统。论文介绍了它的设计和实施。与此同时进行了各种性能测试和性能结果和讨论

## F2FS文件系统简介

# F2FS设计和F2FS的实现

## **磁盘布局**

作者们精心布置了匹配NAND闪存是如何底层的组织和管理F2FS的磁盘上的数据结构如图Figure1：On-disk layout of F2FS所示，F2FS将整个体积成固定大小的分段。该段是管理在F2FS的基本单元，并用于确定初始文件系统元数据的布局。甲部分由连续的段，和一个区由一系列节组成。这些单位记录和清洁.

检查点（CP）保持文件系统状态，有效的NAT位图/ SIT集（见下文），孤儿的inode列表和摘要条目当前有效的区隔。一个成功的“关卡包”应该在突然断电事件（第2.7节）后存储在时间恢复点的给定点一致F2FS状态。一个用于最后稳定版本和其他用于中间（作废）版本

段信息表（SIT）包含persegment信息，诸如有效数量blocksand在“主”区域位图的所有块的有效性（参见下文）。的SIT信息被检索以选择受害者段和在擦除过程中（第2.5节）确定它们有效块。

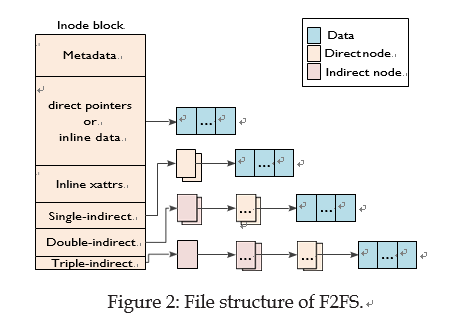
节点地址表（NAT）是一个块地址表来定位存储在主区域中的所有“节点阻塞”。

段汇总区（SSA）存储表示所有块的所有者信息的主要区域摘要enintries，如父节点号码与它的节点/数据偏移量。在SSA项清洁过程中迁移的有效块之前确定父节点块。

主要领域是充满了4KB块。每个块被分配并输入到benode或数据。节点块包含索引节点或数据块的索引，而一个数据块包含任一目录或用户文件数据。请注意，部分不同时存储数据和节点块。

基于其上述结构，文件的查找操作实现如下：（1）它通过读取块，其位置是从NAT获得获得根inode; （2）在根inode块时，它搜索从它的数据块命名DIR和获得其inode编号的目录条目; （3）平移获取的信息节点号码到通过NAT的物理位置; （4）获取通过读取对应的块命名DIR索引节点; 和（5）在dir索引节点，其标识所述目录项命名的文件，最后，获得通过重复步骤（3）和（4），用于文件的文件索引节点。实际的数据可以从主区域被检索，经由相应的文件结构而得到的索引。

## 文件结构

利用F2FS延伸的inode映射以找到更多个索引块“节点”结构。每个节点块具有一个唯一的标识号，“节点ID”。示意图如下图所示通过使用节点ID作为索引，NAT服务于所有节点的块的物理位置。节点块表示三种类型中的一种：i节点，直接和间接的节点。一个inode块包含文件的元数据，如文件名，inode编号，文件大小的atime和DTIME。直接节点块包含数据的块地址和一个间接节点块具有定位另一个节点的块的节点ID。

使用F2FS基于指针的文件索引与直接和间接的节点块以消除更新传播。在传统的LFS设计，如果叶数据进行了更新，其直接和间接指针块递归更新。F2FS，但是，只更新一个直接的节点块和它的NAT表项，有效地解决了徘徊树问题。例如，当一个4KB数据被附加到8MB的到4GB的文件时，更新LFS 2分指针块递归而F2FS仅更新一个直接节点块（不考虑高速缓冲存储器影响）。对于大于4GB的文件，在LFS更新多了一个指针块（总共三个），而F2FS仍然只有一个更新。

一个inode块包含直接指向该文件的数据块，两个单间接指针，双doubleindirect指针和一个三重间接指针。F2FS支持嵌入式数据和内嵌扩展属性，其嵌入在inode块本身小型数据或扩展属性。内联减少了空间的需求，提高I / O性能。请注意，许多系统都有小文件和少量的扩展属性。默认情况下，F2FS激活数据的内联如果文件大小小于3692个字节。F2FS保留在一个inode块200个字节用于存储扩展属性。

## 目录结构

在F2FS，一个4KB目录条目块由一个位图，并在双时隙和名称的两个阵列的。位图告诉每个插槽是否有效或无效。甲时隙承载的散列值，i节点号，文件名称和文件类型（例如，正常的文件，目录和符号链接）的长度。目录文件构建多层次的哈希表来有效地管理大量目录项的。当F2FS查找特定文件名的目录时，它首先计算文件名的哈希值。然后，它遍历增量从0级到记录在索引节点的最大分配水平构建的哈希表。在每一个级别，它扫描两个或四个目录项块中的一个挖斗，导致O（日志目录项的（＃））的复杂性。为了更快地找到一个目录项，它比较的位图，散列值，并以文件名。当大型目录是优选的，用户可以配置F2FS最初许多目录项分配空间。在低级别较大的哈希表，F2FS更迅速地达到目标目录项。

## 多记录头

F2FS维持六大日志区域最大化冷热数据分离的效果。默认情况下，F2FS激活6个日志对写开放。如果这样做被认为是产生一个给定的存储设备和平台上更好的结果，用户可以在安装时调整写入流的数量，以两个或四个。如果使用六个日志，每个日志段直接对应于在表1中列出。在四个日志的情况下的温度水平，F2FS结合冷暖日志中的每个节点和数据类型的。仅具有两个日志，F2FS分配一个节点，另一个用于数据类型。F2FS介绍配置区域是用FTL兼容，以减轻垃圾收集（GC）开销。FTL算法主要分为根据数据和“日志的闪存块”之间的关联性三组（块相联，组相联和全相联）。一旦一个数据闪存块被分配来存储初始数据，日志的闪存块同化数据更新尽可能，像轴颈EXT4。日志闪存块可以专门用于单个数据闪存块（块相联），对于所有的数据闪存块（全相联），或用于一组邻接数据闪存块（组关联的）。现代FTLs采用全相联或组关联方法，能够妥善处理随机写入。需要注意的是F2FS使用多记录头和缔FTL将混合分离的块（在文件系统级别）到相同的闪存块中并行写入节点和数据块。为了避免这种不对准，F2FS活动日志映射到不同的区域将它们在FTL分离。这种策略有望成为有效的组关联FTLs。多头记录也是天然的匹配与最近提出的“多流传输”界面。

## 擦除

擦除是回收散射和无效块的处理，并且确保自由段用于进一步记录。由于清洁不断发生，一旦底层的存储容量已装满时，限制与清理相关的费用是F2FS的持续性能（和一般的任何LFS）非常重要。在F2FS，擦除是在部分单位进行。F2FS执行擦除以两种不同的方式中，前景和背景。当没有足够的自由部分前景擦除时才会触发，而内核线程醒来定期在后台进行擦除。清洁过程需要三个步骤：

（1）**牺牲选择**。擦除过程开始首先确定非空部分中的受害者部分。有两个受害者选择LFS擦除贪婪和成本效益在著名的政策。贪婪的政策选择与有效块数最少的部分。直观地看，这一政策控制迁移有效块的开销。F2FS采用其前景擦除，以尽量减少等待时间的应用程序可见贪婪的政策。此外，F2FS保留一个小的未使用的容量，使得在清洁过程有余地在高存储利用率水平足够操作。

（2）**有效块识别和迁移**。选择一个受害者部分后，F2FS必须在部分快速识别有效块。为此，F2FS维持SIT每段有效性位图。一旦已经通过扫描位图标识的所有有效块，F2FS检索包含从SSA信息及其索引父节点块。如果块是有效的，F2FS他们迁移到其他自由日志。有关背景清洁，F2FS不发出实际的I / O的迁移有效块。相反，F2FS加载块到页面缓存然后将其标记为脏。然后，F2FS只是使他们在页面缓存为内核工作线程以后他们刷新到存储。这个懒惰的迁移，不仅减轻了前台I / O活动对性能的影响，而且还允许合并小型写入。

（3）**后擦除过程**。所有有效块迁移后，受害者部分登记为候选人，成为一个新的免费部分。检查点作出后，该段终于成为自由部分，被重新分配。我们这样做是因为如果无预部分检查点之前重用，文件系统可能会失去当意外停电时由以前的检查点所引用的数据。

## 自适应记录

原来LFS推出了两款日志政策，正常的日志记录和线记录。在正常的记录，块写入擦除段，得到严格顺序写入。F2FS同时实现的政策和动态根据文件系统的状态之间切换。具体地，如果存在大于k擦除段，其中k是一个预先定义的阈值比较，在正常记录被启动。否则，线日志被激活。k被默认设置为总截面的5％，并且可被配置。

还有就是当有线日志招致不良随机写入的机会。然而，这样的随机写入通常会比那些在更新就地文件系统空间局部性更好，因为在一个肮脏的段中的所有孔之前F2FS搜索第一填充更多的其它脏段。快闪存储装置上显示具有较强的空间局部性更好的随机写入性能。

## 检查点和恢复

F2FS实现了检查点，以提供从突然断电或系统崩溃一致的恢复点。每当它需要保持遇到像同步，使用umount和前景吸尘的事件一致的状态，F2FS触发检查点过程如下：（1）在页面缓存中的所有脏节点和目录项块刷新; （2）暂停普通书写活动，包括系统调用如创建，取消关联和MKDIR; （3）文件系统元数据，NAT，SIT和SSA，被写入到磁盘上的专用区域; 及（4）最后，F2FS写入检查点包，内容主要包含页眉和页脚、NAT和SIT位图、NAT和SIT日志、活跃的摘要的块和孤儿块。

F2FS实现的回滚主要包括2种，回滚恢复和前滚恢复。

F2FS执行如下滚恢复。突然断电后，F2FS回滚到最新的一致的检查点。为了保持至少一个稳定的关卡包，同时创造了新的包装，F2FS维护两个关卡包。如果检查点包在页眉和页脚内容完全相同，F2FS认为有效。否则，将被丢弃。

同样地，F2FS还管理两套NAT和在每个检查点包SIT块，由NAT区分和SIT位图。当写入更新NAT或检查点期间SIT块，F2FS它们写入两组交替地中的一个，然后标记位图以指向其新的一组。

如果NAT少数或坐项频繁更新，F2FS会写许多4KB大小的NAT或坐块。为了减轻这种开销，F2FS实现了关卡包内的NAT和SIT杂志。该技术减少的I / O的数目，并且因此，所述检查点设置等待时间为好。F2FS执行如下滚恢复，果我们用最后一个检查点稳定为N日志位置，（1）F2FS收集其位于N + N的特殊标志，而构建自己的节点信息的列表，直接节点块。n引用自上一个检查更新的块数。（2）通过使用列表中的节点信息，它加载最近写入节点块，命名为NN，进入页面缓存。（3）然后，在NN和N + N之间的数据索引进行比较。（4）如果检测到不同的数据索引，则刷新存储在N + n中的新索引缓存节点阻塞，最后将其标记为脏。一旦完成了前滚恢复，F2FS执行检查点存储在内存中的变化到所述磁盘的整体。

# 实验测试设置、结果与相关讨论

## 实验设置

作者的试验评估分别在移动系统和服务器系统上进行实验。比较了F2FS与EXT4 ，BTRFS 和NILFS2这几种不同的文件系统。EXT4是一种广泛使用的更新就地文件系统。BTRFS是写入时复制文件系统，并且NILFS2是LFS。对于服务器的工作负载，作者利用Filebench合成基准。模拟各种文件系统的工作负载，主要包括videoserver，文件服务器，varmail和OLTP。如下表所示

Table 1: Summary of benchmarks.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Target** | **Name** | **Workload** | **Files** | **File size** | **Threads** | **R/W** | **fsync** |
| Mobile | iozone | Sequential and random read/write | 1 | 1G | 1 | 50/50 | N |
| SQLite | Random writes with frequent fsync | 2 | 3.3MB | 1 | 0/100 | Y |
| Facebook-app | Random writes with frequent fsync  generated by the given system call traces | 579 | 852KB | 1 | 1/99 | Y |
| Twitter-app | 177 | 3.3MB | 1 | 1/99 | Y |
| Server | videoserver | Mostly sequential reads and writes | 64 | 1GB | 48 | 20/80 | N |
| fileserver | Many large files with random writes | 80,000 | 128KB | 50 | 70/30 | N |
| varmail | Many small files with frequent fsync | 8,000 | 16KB | 16 | 50/50 | Y |
| oltp | Large files with random writes and fsync | 10 | 800MB | 211 | 1/99 | Y |

## 文件系统性能测试结果

结果至今都清楚地表明了整体设计和实施F2FS的相对有效性。在移动端和服务器端的性能测试结果分别如Figure3和Figure4所示

## 屏幕剪辑相关讨论

接下来作者论述了多头记录的影响和清洗成本。

在评估多头记录的影响时，使用的度量是在擦除之前在给定的脏片段有效块的数目。如果冷热数据分离是完全完成，脏段将具有零个有效块或在一个段有效块的最大数目（512下的默认配置）。一个上了年纪脏段将进行零点有效块，如果存储在段中的所有（热）数据已经失效。相比之下，脏段完整且有效的块很可能保持冷数据。冷热数据分离的好处是具有零个有效块和全段的数量自由预段的数量增加显著。此外，存在具有相对较少的有效块（128或更少）中，用许多的有效块（384或更多个）链段多段。这种双峰分布的影响明显提高擦除效率（如擦除成本在受害者段取决于有效块的数目）。

在量化擦除成本时，为了专注于文件系统级别的擦除费用，作者保证不实验期间通过在SSD故意留出了充足的自由空间发生SSD级别GC。要做到这一点，我们格式化250GB SSD，并获得头（只）120GB的分区。F2FS不显着的高利用率水平提高WAF; 自适应记录起着保持WAF下来的重要作用。需要注意的是线的记录即被随机写入，而正常的日志记录问题的顺序写入。而随机写入相对昂贵和激励appendonly记录作为操作的许多文件系统的优选模式，我们的设计选择（切换到线日志）是合理的，因为：当文件系统的清洗可以使非常昂贵，由于高WAF零散的，固态硬盘具有较高的随机写入性能。结果在本节中显示，F2FS成功地控制在高利用水平清洗的成本。像F2FS日志结构文件系统可能遭受高的擦除成本。这个工作量在存储设备碎片数据，并随着工作负载的重复触发设备内部GC操作的存储性能会受到影响。

# 综述与收获

论文作者从开篇引题介绍了现存文件系统的弊端，采用了一种新的设计方式重新设计了FTFS文件系统。F2FS (Flash Friendly File System) 是专门为基于 NAND 的存储设备设计的新型开源 flash 文件系统。特别针对NAND 闪存存储介质做了友好设计。

接着作者论述了其设计理念与实现。事无巨细的介绍了F2FS的磁盘布局、文件结构、目录结构、多记录头、清洗、检查点和恢复等设计要点。F2FS 选择log-structured文件系统方案，并使之更加适应新的存储介质(NAND)。同时，修复了旧式日志结构文件系统的一些已知问题，如(1) wandering tree 的滚雪球效应和(2)高清理开销。根据内部几何结构和闪存管理机制(FTL)，闪存存储设备有很多不同的属性，所以F2FS的设计者增加了多种参数，不仅用于配置磁盘布局，还可以选择分配和清理算法，优化性能(并行IO提高性能)。

作者最后分别在移动系统和服务器系统针对不同的服务做了性能试验并和其他文件系统进行了比较。实验表明了F2FS文件系统的优良性能。

这篇文章对我们以后进行科研有很大帮助。作者思路清晰，针对问题分而治之又统筹大局，使得文件系统整体效果处在很好的一个水平。愿自己能够学习作者这种发现问题，分析问题，结局问题的思路和事必具细的科研精神，自己能够更上一层楼。