File Defragmentation Scheme for a Log-Structured File System

日志结构文件系统的文件碎片整理方案

ABSTRACT

LFS=Log-Structured File System→提高随机写性能

/有效解决一致性问题

**但** LFS写机制→导致文件碎片整理问题→损害文件系统顺序读性能

结合底层存储设备特性 本文探讨：文件碎片整理

与 顺序读性能 的关系

提出：LFS新型文件碎片整理方案→有效解决文件碎片整理问题

碎片整理方案：LFS清理操作过程中→根据节点序号→重新整理属于受害段的有效数据块

INTRODUCTION

闪存优点：耗能低/存储密度高→闪存应用广泛

闪存缺点：寿命有限/异地更新

克服缺点：FTL层

①扩展写操作→延长闪存寿命

②GC:回收异地更新产生的失效页面

**但** GC需移动受害块的有效页到新块→产生额外I/O→GC损害I/O性能和闪存寿命

**且** 随机写将无效块分散→GC需拷贝的块数量增加→加剧上述开销

LFS：仅追加写机制→使大部分写入为顺序写入→避免随机写→解决闪存缺点

LFS缺点：文件碎片管理问题→降低文件系统的各方面性能

LFS上产生文件碎片管理问题的原因：

①多个进程同时在LFS上创建具有同步写入的文件→LFS从同一顺序池中取闪存块并将块交替分配给文件→各文件碎片化

②文件部分更新/文件大小增加→LFS上产生文件碎片管理问题

**文件碎片管理问题降低读性能**的原因：

①块请求表示为连续块的列表→文件碎片化拆分块请求→干扰有效块I/O→降低I/O性能

②闪存预取功能：FTL预先加载后续数据→提高读性能

预取数据取决于LBA→文件碎片化导致FTL预取无效数据→降低读性能

LFS仅追加写机制→不允许就地更新→LFS清理过程需回收无效块中的空闲空间←与FTL中GC类似

LFS现存清理工作不考虑受害块的顺序→无法消除文件碎片化问题

本文针对文件碎片化提出：LFS的新型清理机制

新型清理机制：

①清理过程对文件进行碎片整理：根据节点序号重新排列有效块

②提出碎片病毒受害者选择机制→提高清理效果

碎片病毒受害者选择机制：碎片化程度=选择过程中有效块的数量

2. RELATED WORK AND MOTIVATION  
2.1 File Fragmentation

利用SSD并行性→FTL预先从后续LBA中抓取数据→提高顺序读性能

但 碎片化→文件被分离为多个碎片(图1)→阻碍抓取

碎片化→增加内核开销/降低块I/O的顺序性

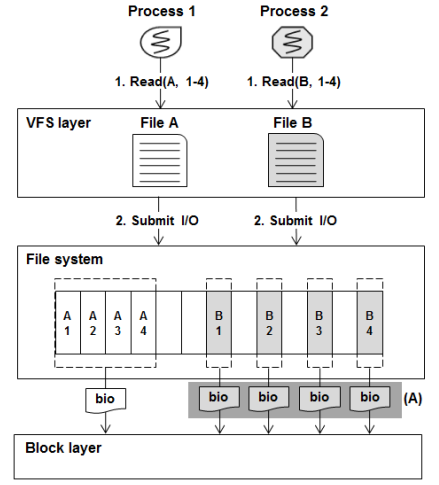


图1 文件碎片和拆分BIOS(bio是linux内核中通用块层的一个核心数据结构，它描述了块设备的I/O操作，联系了内存缓冲区与块设备)。

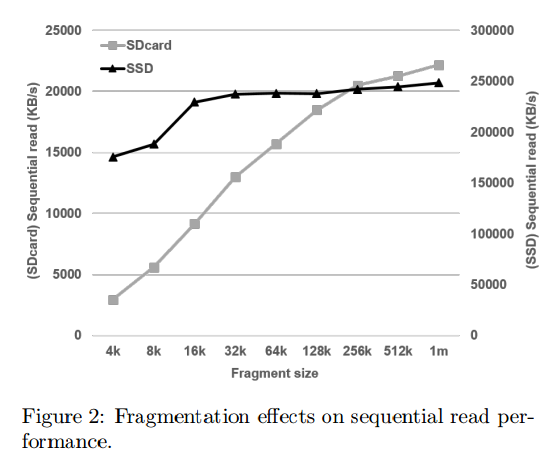
在文件系统层上，每个字母代表一个文件名，每个数字代表一个文件偏移量。

内核使用bio(bio具有连续块信息以向块层提交块I/O请求)

某操作提交碎片化文件的读请求：

其bio无法如图1(A)合并→内核需要为读请求准备更多bio

分离的I/O在底层设备一个个处理→文件碎片化降低整体读性能

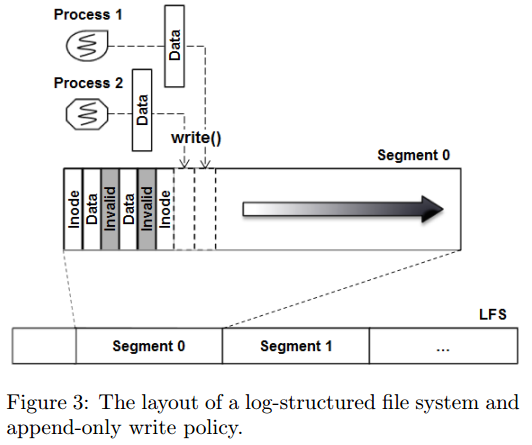


碎片化程度越高→顺序读性能越差

2.2 A Log-structured File System

LFS的布局：根据块类型将存储空间分为段(如节点块/数据块/直接块)

某操作提交写操作→LFS在当前段末尾分配新块以避免就地更新(即仅追加写策略，如图3)



LFS仅追加写策略→将大部分写入操作顺序处理→**仍导致文件碎片化**

如：

①某操作覆盖写文件，覆盖写数据与该文件数据分离→文件碎片化

②多个操作同时创建同步写文件→破坏缓冲与分组写入→因LFS从同一顺序池中向文件交替分配块→文件碎片化

可用块→LFS清洁操作是必需的→清洁操作有开销→降低性能

清洁线程的详细步骤：

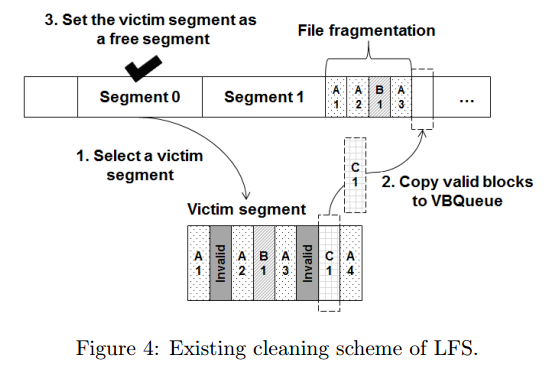
①根据受害者选择机制(如贪心或成本效益策略)选择受害段

②查看受害段中的各块是否有效

③探测到有效块后将有效块通过主存拷贝至新段

④对受害段中所有块重复2-4步

⑤若已拷贝全部有效块，该受害段可被标记为可用段



减少清洁开销→LFS的受害者选择机制:贪心策略/成本效益策略

①贪心策略：选择有效块数量最少的段→保证最低的拷贝开销/最高的可用有效空间创造率

②成本效益策略：综合考虑1.各段的最后修改时间及2.有效块的数量来选择受害段→受害段能持续很长时间

现存清洁操作不按原始顺序拷贝有效块→文件碎片化

1. LFS CLEANING SCHEME FOR FILE DEFRAGMENTATION
   1. A Cleaning Scheme for Defragmentation

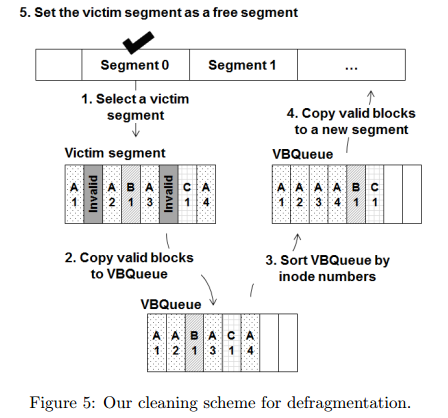
现存清洁操作未考虑受害块的顺序→无法消除文件碎片化问题

新型清洁机制：将段中受害块重新排列

1. 通过受害者选择机制选择受害者并查看段中块是否有效
2. 探测到有效块则将其加载到主存的VBQueue中，即将其延迟到新段
3. 已将全部有效块加载到VBQueue中后，通过节点数字**整理**有效块并拷贝到新段→整理后有效块与同一文件的其他块是连续的(因各文件具有独一无二的节点数字)

→消除文件碎片化

→整理过程无写入操作→开销极小

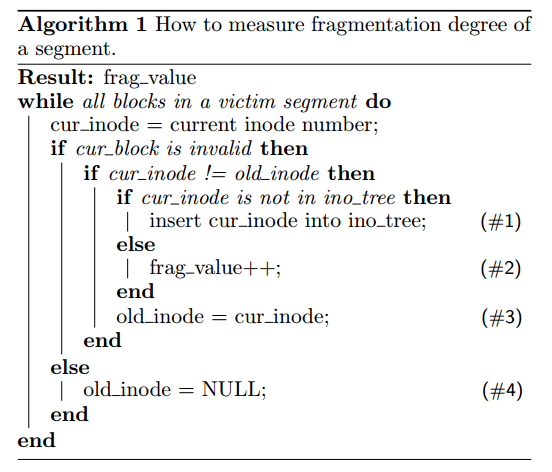


3.2 Fragmentation-aware Victim Selection Policy

现存受害者选择机制不能实现新型清洁操作→提出碎片病毒受害者选择机制实现新型清洁操作

碎片病毒受害者选择机制：利用各段的碎片化程度来选择受害段

(碎片化程度=选择过程中有效块的数量)



当前块有效→当前块节点与前一块节点对比

→若相同→两块属于同一文件且LBA连续→未碎片化

→若不同→检测当前块节点是否包含于节点树中

→若包含→检查过的块已有该文件的另一块→文件碎片→碎片化程度+1

→若不包含→该块是该文件中出现的第一块→未碎片化