**original paper/**

Barrier-Enabled IO Stack for Flash Storage (FAST'18)

•问题：为保证写请求的存储顺序，块设备采用了写请求与Transfer-and-Flush交替进行的方法，但这带来了较大的开销。  
•方案：提出了一种屏障使能的IO栈，利用缓存屏障命令，调整了IO调度器、调度模块和文件系统，使其能够保证数据块在存储空间中的持久顺序性，且没有Transfer-and-Flush所带来的开销。  
•平台：服务器和移动平台；MySQL和SQLite。  
•效果：在服务器中，SQLite性能提高了270%；在移动平台上，SQLite性能提  
高了75%。在服务器中，通过放宽事务的持久性，BarrierFS相对于EXT4，在MySQL和SQLite中分别实现了43倍和73倍的性能提升。

Enlightening the IO Path (FAST'17)

•问题：多数据应用需减少请求处理延迟（多由IO栈和存储的竞争引起），现有方法是适当规定IO的优先级，但考虑到静态规定IO优先级存在优先级反转的可能且规定IO优先级时并未全面考虑存储栈中各层，此法效果不佳。

•方案：RCP，以请求为中心，在整个IO路径中动态检测IO请求处理延迟并规定IO优先级。

•平台：Linux；3个应用：PostgreSQL，MongoDB和Redis。

•效果：与Linux默认设置相比，请求吞吐量提高了53%；请求延迟提高了42倍。

FastTrack in Android Smartphones (ATC'18)

•问题：前台应用对用户输入的反应速度决定了智能手机的用户体验，对此，需解决IO优先级的倒置问题。

•方案：前台应用感知的IO管理方案FastTrack，通过1）抢占后台应用2）防止前台应用数据被页缓存刷新，加速前台应用的请求。

•平台：4部智能手机。

•效果：无论后台应用多少，前台应用均可达用户感知。同时运行6个多IO后台应用时，FastTrack能减少用户反应时间94%。

Unblinding the OS to Optimize User-Perceived Flash SSD Latency (HotStorage'2016)

•问题：减少IO完成调度的延迟。

•方案：将SSD内部行为暴露给OS，使其应用IO延迟预测以精准调度。

•平台： Xilinx Zynq-7000 FPGA和MLC flash （在内部资源层面（DRAM/NAND）跟踪IO以实现IO延迟预测）。

•效果：能够在不严重影响系统并行性的情况下减少调度延迟。

**基于Linux内核的IO优化/**

ATC/

Lightweight Application-Level Crash Consistency on Transactional Flash Storage (ATC'15)

•问题：为确保FS一致性的更新协议效果不佳。

•方案：CFS，通过声明原子操作的码区实现一致性。

•平台：事物型闪存（SSD/X-FTL）；5种真实应用。

•效果：速度提高2-5倍；减少1.9-4.1倍的磁盘写；减少1.1-17.6倍的磁盘缓存刷新；代码改动少（317/(3.5\*10^6)）。

Request-Oriented Durable Write Caching for Application Performance (ATC'15)

•问题：盲目缓存全部写操作的接纳策略无法充分发挥NVWC的优势（提高多IO应用性能）。

•方案：面向请求的接纳策略，仅在请求执行的上下文中缓存等待的写入。通过应用程序级提示标识处理请求所涉及的关键过程以准确地检测此类写入。

•平台：Linux内核；PostgreSQL数据库；Redis NoSQL存储。

•效果：性能比盲目缓存全部写入提高了2.2倍；减少了87%的NVWC写入量。

SpanFS: A Scalable File System on Fast Storage Devices (ATC'15)

•问题：针对于硬盘设计的传统FS，难以发挥多核结构的快速存储设备（SSD）的优点（低延迟/高并行性），即难应用于快速存储设备，缺乏扩展性。

•方案：SpanFS（由一组称为域的微文件系统服务组成）在域之间分发文件和目录，在域顶部提供全局文件系统视图，并在系统崩溃时保持一致性。

•平台：Ext4（对比实验）；PCI-E SSD

•效果：在32核机器上扩展性更好；在微基准测试中，性能高1226%；在应用程序级基准测试中，性能高73%。

Understanding Manycore Scalability of File Systems (ATC'16)

•问题：对于多I/O应用，FS是隐藏的可扩展性瓶颈，即使在应用程序级别没有明显争用时也是如此。

•方案：分析ext4，XFS，btrfs，F2FS和tmpfs五种FS的多核可扩展性。

•平台：基准套件FXMARK。

•效果：找到25个瓶颈；提出FS需解决的问题。

WALDIO: Eliminating the Filesystem Journaling in Resolving the Journaling of Journal Anomaly (ATC'15)

•问题：对于日志异常的记录问题。

•方案：编排sqite和ext 4文件系统，这样sqite的文件支持日志活动就可以避免昂贵的文件系统干预，即日志记录，而不会在意外的文件系统故障时损害文件的完整性。在存储日志时，利用直接IO来抑制文件系统的干扰。WALDIO具体包括：1）显式日志预分配（消除了文件系统日志，正确地保护文件元数据免受意外的系统崩溃的影响）2）头嵌入（重新设计sqliteb-树结构，兼容直接IO，与块IO友好。）3）组同步（最大限度地减少了直接IO的同步开销，使SQLITE操作NAND闪存变得友好）。

•平台：商用智能手机。

•效果：与WAL相比，性能提高5.1倍；与LS-MVBT相比，性能提高2.7倍；WALDIO模式将IO体积减少到1/6。