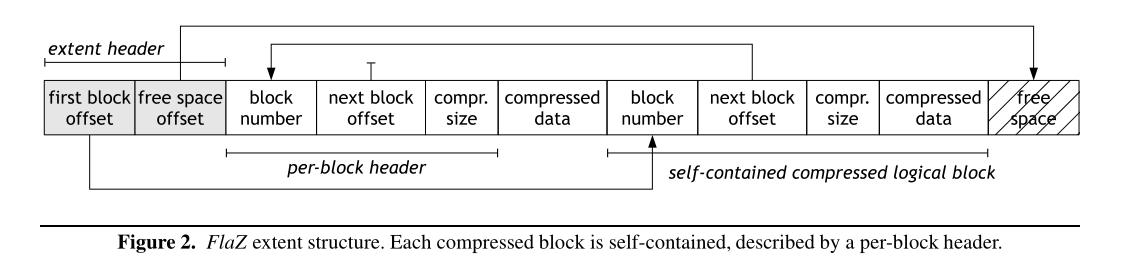
# Using Transparent Compression to Improve SSD-based I/O Caches

问题：存储I/O性能是现代系统中的一个重要问题。基于闪存的固态硬盘的出现有可能减轻I/O开销:SSD具有较低的读响应时间，并且不受seek的影响。此外，最近的SSD提供的峰值吞吐量明显优于磁性硬盘驱动器(HDD)。虽然其容量继续增加，但没有迹象表明SSD目前每GB的高成本将很快接近磁盘的价格。因此，I/O层次结构将包括HDD和SSD，因此，研究可以提高SSD成本效率的技术是很重要的。

分析：一种方法是使用多级单元(MLC) ssd存储。另一种与单元密度正交的方法是在I/O路径中使用透明的在线数据压缩。此前，压缩大多应用于文件级别。尽管这减少了存储数据所需的空间，但也带来了一些限制。通常，文件级压缩技术是特定于文件系统和操作系统的，这限制了在I/O路径中应用压缩的位置。此外，文件系统通常不知道物理存储设备的属性。最后，在文件级处理压缩增加了管理的复杂性。这些问题可以通过将压缩功能转移到较低的系统层(如块设备层)来解决。因此文章提出设计了FlaZ，一个在I/O路径中使用ssd作为压缩缓存的系统。ssd的地址空间对应用程序不可见。FlaZ缓存数据块在专用的SSD分区上，当数据在主内存和SSD之间的I/O路径中流动时，透明地压缩和解压数据。数据只以压缩格式存储在ssd上，以非压缩格式提供给文件系统。

解决办法：FlaZ使用两个映射来定位存储压缩块的区和区偏移。第一个映射使用逻辑到物理的转换表，而第二个映射使用嵌入在范围中的链表，如图2所示。



本设计中物理块是不可变的，因为对逻辑块的修改总是写入底层设备上的一个新的物理区段。当第一次写入逻辑块时，FlaZ压缩块数据并选择适当的范围。逻辑块的就地更新通常是复杂的，因为块的大小可能会因为更新而发生变化。因此，设计中使用分组为更大区段的不可变物理块。

创新点：该设计使用ssd作为缓存，并在数据在主存和缓存之间流动时压缩数据。压缩在公共路径上在线执行，并增加有效缓存大小。

效果：使用TPC-H、PostMark和SPECsfs在一个具有多核cpu的商用服务器上对它进行了评估。结果表明，压缩缓存为I/O性能牺牲了CPU周期，但将SSD作为缓存的效率分别提高了99%、25%和11%。

思考：想学习一下文章中对于压缩之后块的分布和管理的处理，文章中采用数倍块大小的一个Extent的粒度来存储数据，感觉这篇论文时间有点久远，直观上觉得这种方式对于现在的SSD存储方式有点格格不入，不过其对于压缩数据的管理方式是值得学习的，他的反式可以极大利用存储空间而基本不会浪费压缩数据之后的引起的空缺问题。进一步思考可不可以跟其他的管理数据块的方式结合一下，有新的更好的管理和分布方式。