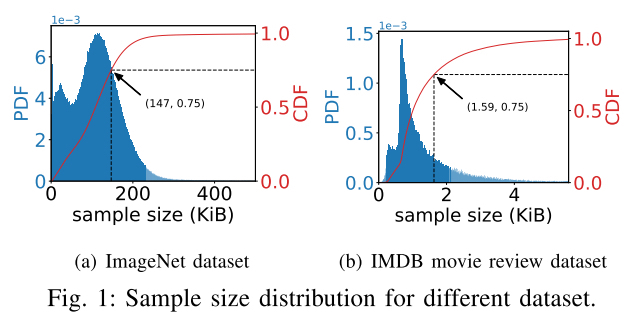
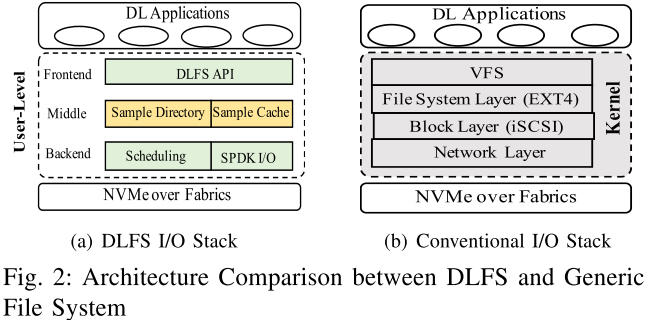
**Efficient User-Level Storage Disaggregation for Deep Learning**

摘要：本文首先揭示了深度学习应用中存在很多小的随机读的IO特征，然后在基于SPDK实现资源聚集的HPC系统中，提出了针对深度学习应用的用户级别的文件系统——DLFS。

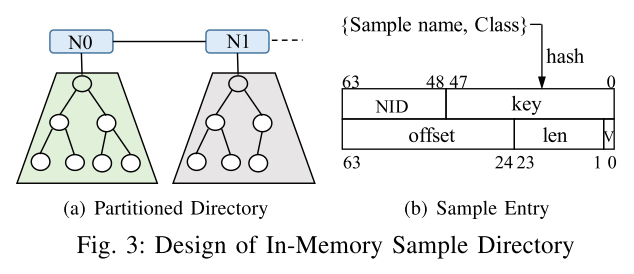
1. 动机
2. **充分利用HPC中的NVMe SSD资源**：为了让DL应用充分利用HPC中的NVMe SSD资源，NVMe over Fabrics的架构标准可以实现资源聚合，使得两个结点之间可以相互访问存储数据（访问远程的数据在10μs内）；为了让一个结点存储的数据可以被多个结点同时访问，Intel提出了SPDK协议，可以在用户层实现这一要求。本文中为了让DL应用充分应用HPC集群中的SSD资源，使用的就是SPDK协议。
3. **缓解大量小随机读问题**：DL应用中的ImageNet数据集和IMDB数据集的数据大小都很小，在应用中有很多的小的随机读。并且由于mini-batch的增加，需要小的随机读的文件数量也很大。



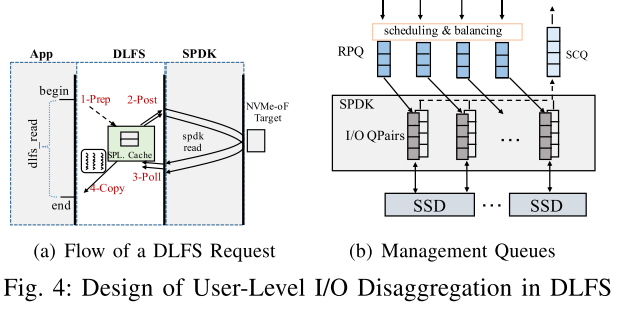
1. 设计
2. **整体设计**：由于很多小的随机读，所以传统文件系统在整个IO过程中经内核的时间开销占比变大（上下文切换和数据拷贝到内核区都要花时间）。因此，DLFS尽在用户层实现文件的读取，而没有经过内核层。



1. **元数据管理**：由于每个结点从PFS读取数据到NVMe SSD后，文件不再移动，所以DLFS中将让每个结点在内存中都存储了全局文件的元数据。为了加快查询，每个结点上的元数据都采用AVL平衡树进行存储。



1. **数据读取流程**：依次有prep/post/poll/copy四个环节实现从本地或者远程读取数据，并且每个结点有缓存空间。此外，通过给每个SSD都分配两个队列分别用于记录请求和标记请求完成，从而可以实现负载均衡。



1. **Batch过程优化：**针对DL中的batch过程，本文提供了两种方法进行优化。一种是由DLFS产生一个全局的随机序列，每个结点取其中一部分数据，因此不用等应用给出随机序列，从而实现不断得随机读取。另一种方法是一次读取一块数据到本地缓存起来，块是随机的，一个块包含很多的小文件，然后随机从这里面取数据组成新的batch，这样可以避免小图片的读取，而是读取一个块的大小。（虽然第二种方法随机度有所下降，但是经过实验表明不影响训练精度）

