高能物理计算环境中存储系统的设计与优化

程耀东 汪 璐 黄秋兰 陈 刚
(中国科学院高能物理研究所计算中心 北京 100049)

摘 要 高能物理是典型的数据密集型计算,数据访问性能对整个系统至关重要并与应用的计算模式密切相关。从剖析高能物理的典型计算模式入手,总结出其数据访问的特点,提出针对操作系统I/O调度、分布式文件系统缓存等多个因素的优化措施,优化后数据访问性能和CPU利用率明显提高。大规模存储系统对于元数据管理、数据可靠性、扩容等可管理性等功能也有较高要求,结合现有Lustre并行文件系统的不足,提出了Gluster的高能物理存储系统设计,在进行数据管理以及扩容等方面的优化后,系统已经正式投入使用,数据访问性能能够满足高能物理计算的需求,同时具有更好的可扩展性和可靠性。

关键词 高性能计算,海量存储,Lustre,Gluster
中图法分类号 TP301 **文献标识码** A **DOI** 10.11896/j.issn.1002-137X.2015.1.012

Design and Optimization of Storage System in HEP Computing Environment

CHENG Yao-dong WANG Lu HUANG Qiu-lan CHEN Gang
(Computing Center, Institute of High Energy Physics Chinese Academy of Sciences, Beijing 100049, China)

Abstract High energy physics computing is a typical data-intensive application, and the performance of data access throughput is critical to the computing system. The performance of data access is closely related to the computing model of application. This paper firstly analyzed the typical high energy physics computing models, and then summarized the characteristics of data access. Based on these characteristics, some optimization measures were proposed, including operating system I/O scheduling policy, distributed file system cache configuration and so on. Data access performance and CPU utilization are improved significantly after optimization. Metadata management, data reliability, scalability and other manageability functions are also important for large-scale storage system. Considering some shortcomings of existing Lustre parallel file system, this paper finally proposed Gluster storage system as a new solution for high energy physics. After tuning of some key factors, such as data management and scalability, the system has been put into use. It is demonstrated that the data access performance with better scalability and reliability can meet the needs of high-energy physics computing.

Keywords HPC, Mass storage system, Lustre, Gluster

人类探索基本物质组成以及宇宙起源的脚步不断前行,高能物理实验的规模也在不断扩大,欧洲大型强子对撞机LHC、北京正负电子对撞机BEPCII、大亚湾中微子实验、羊八井国际宇宙线实验等大型高能物理实验已经建成并投入运行,累积了越来越多的数据。LHC每年会产生25PB的数据,采用网格计算模型进行全球分布式的处理与分析[1]。BEPCII累积超过5PB的数据,主要采用大规模计算集群进行分析。此外,大亚湾中微子探测器二期、大型高海拔空气簇射实验等新的实验正在建设,将产生更多的数据。

越来越多的高能物理实验数据对存储系统提出了越来越高的要求,包括容量、性能、可扩展性、可靠性、长期保存与性价比等。目前,在高能物理领域,海量存储系统主要包括dCache[2]、CASTOR[3]、DPM、GPFS[4]、Lustre[5]等。dCache主要应用于全球高能物理网格WLCG中的Tier1站点,DPM主要应用于WLCG的Tier2站点,CASTOR主要在欧洲核子中心CERN,GPFS主要在法国和意大利的一些高能物理实验室采用,德国的GSI和中科院高能所建设的Lustre系统具有典型代表性。

无论采用哪种系统,数据访问性能都是其中关键的指标,而数据访问性能与计算模式是密切相关的。因此,本文首先着重剖析高能物理的典型计算模式及文件访问模式,并给出一些统计数据实例。接着,根据高能物理应用的访问模式,提出了存储系统优化措施。除了性能之外,可管理性也越来越重要,结合现有系统的不足,本文最后介绍了相关的功能设计与优化,包括元数据管理、容错与可靠性、扩容等。

1 高能物理计算访问模式

1.1 高能物理计算系统典型部署

高能物理计算是典型的数据密集型应用,其计算特点是从海量的数据中挖掘出稀有事例。由于事例之间的无关性和

到稿日期:2013-12-26 返修日期:2014-03-15 本文受国家自然科学基金(11205179)资助。
程耀东(1977—),男,博士,副研究员,主要研究领域为海量存储、网格计算与云计算;汪 璐(1983—),女,博士,助理研究员,主要研究领域为海量存储;黄秋兰(1982—),女,硕士,助理研究员,主要研究领域为海量存储;陈 刚(1961—),男,研究员,博士生导师,主要研究领域为高能物理计算。

独立性，高能物理往往把一系列的事例组成一个文件，多个文件可以在多个机器上同时处理，而不需要相互通信。因此，高能物理计算的特点是高吞吐率的数据并发，不需要采用 MPI 等并行计算技术。基于这些特点，目前高能物理领域普遍采用集群计算系统以及计算和存储分离的模式，典型的系统结构如图 1 所示。

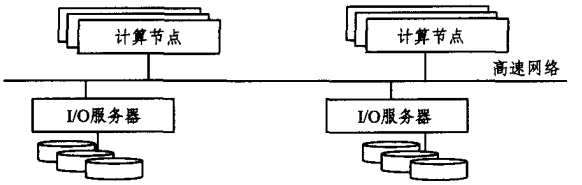

图 1　高能物理计算系统典型结构

如图 1 所示，海量的实验数据存储在 I/O 服务器中，计算节点通过高速网络从 I/O 服务器中获取数据。多个 I/O 服务器通过分布式存储系统来管理，计算节点通过批处理作业系统来管理。分布式存储系统通常有两种模式，一种是分布式文件系统，提供文件系统接口，直接挂载到计算节点上，计算任务像访问本地文件系统一样来存取数据，每个计算节点上的环境都是一致的，包括统一文件系统视图。分布式文件系统采用得比较多的是 GPFS、Lustre、Gluster 等。另外一类仅提供统一的文件系统视图和文件拷贝/传输命令，计算节点安装相应的存储系统客户端，每个计算任务通过拷贝/传输命令把需要处理的文件一次性拷贝到计算节点的本地磁盘上，后续的操作仅访问本地磁盘，不再与 I/O 服务器通信。这类的存储系统使用得比较多的是 dCache、CASTOR、DPM 等。中国科学院高能物理研究所主要采用分布式文件系统来构建高性能存储系统，其优点是很好地兼容了用户在早期 NFS 时代的使用习惯，应用程序和作业提交程序无需任何修改，但是在大规模分布式文件系统调优以及数据可靠性等方面其面临较大的挑战。高能物理研究所从 2006 年开始跟踪 Lustre 系统，2008 年开始部署，目前拥有 3PB 以上的实际使用空间，其部署如图 2 所示。

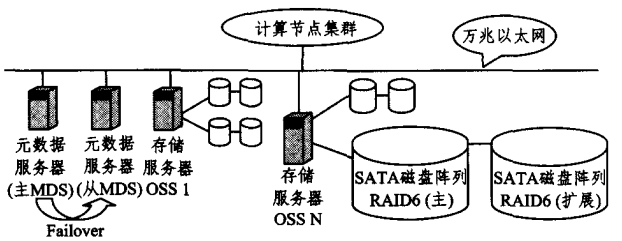

图 2　Lustre 部署图

高能所的 Lustre 部署有 4 个特点：a)元数据服务器 MDS 分为主从，但是无法在线备份和容错；b)采用万兆以太网；c)没有采用分片存储，一个文件仅存储在一个 OST 上；d)存储设备采用廉价的 SATA 盘，通过 RAID6 实现数据可靠性。

1.2　文件访问模式分析

高能物理数据处理的主要过程如图 3 所示。探测器产生的原始数据保存到分布式存储系统中。原始数据非常宝贵，通常要做多份备份保存，并且一般不对最终用户开放。蒙特卡罗模拟程序根据理论物理模型以及探测器的特性，模拟产生数据。事例重建程序读取原始数据或者模拟数据，进行与探测器相关的重建，产生刻度常数与重建数据。重建数据对最终用户开放，用户使用物理分析程序读取重建数据，进行相应的数据分析与挖掘，产生物理成果，包括图表、论文等。

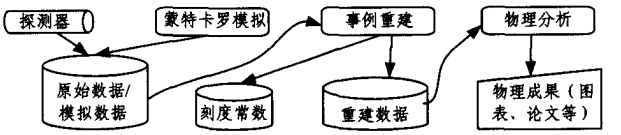

图 3　高能物理数据处理过程

从图 3 中可以看出，高能物理数据处理主要包括模拟计算、重建计算以及物理分析 3 种类型。每种计算类型各有其特点，模拟计算需要很少的输入，主要消耗大量的 CPU，产生大量数据。从探测器建设到运行，一直到产生物理成果，都离不开模拟计算，因此需求量很大。事例重建用来处理探测器产生的原始数据与模拟计算产生的模拟数据，需要读入和写出大量数据，对 I/O 和 CPU 需求都比较大，实时性要求比较高，因为该步骤是物理分析的基础。物理分析由最终用户也就是物理学家发起，读取重建数据，进行数据处理和分析，这一步骤主要从海量数据筛选出稀有事例，对系统的 I/O 要求很高。

下面以高能物理所 BES 集群计算环境中 Lustre 文件访问为例来分析文件访问模式。模拟计算把产生的模拟数据写入 Lustre，模拟计算主要消耗 CPU，对于 I/O 要求不高。重建程序从 Lustre 上读取原始数据，然后把重建后的数据写入 Lustre；分析作业需要读取重建后的物理数据，做分析和处理，将结果写回用户目录。通过实际统计发现，系统主要的负载包括：

1)重建计算中多个进程顺序读/写操作，一般输入文件大小为 500MB 左右，输出文件为 2GB 到 4GB 左右；

2)物理分析计算中多个进程读操作，每个文件大小为 2GB 左右，并发进程数量即集群的最大分析作业数量，目前是 6000。

3)从分级海量存储系统 CASTOR 中读取或者写出大量的原始数据或者重建数据。CASTOR 系统用来管理底层磁带库，本负载实现数据从磁带库离线存储到磁盘在线存储的转换。

高能物理计算中数据是一次写入、多次读取，负载 1，3 占总的系统负载比例较小，性能优化的关键在于负载 2，即通过对分析作业访问的性能的优化来提高文件系统的性能，进而减少计算节点的 I/O 等待时间，提高 CPU 利用率。

通过 Lustre 的/proc 选项 extent_stats、offset_stats 监控计算节点，当节点上运行物理分析任务时得到文件系统的访问模式，即大部分文件的连续读请求大小分布在 256k 到 4M 之间，每两个连续读请求之间都有 offset，65% 的 offset 绝对值分布在 1M～4M 之间，这说明文件的读访问方式为大记录块的跳读。

2　存储系统性能优化

2.1　文件访问性能影响因素分析

分布式存储系统是由多台服务器以及多个存储设备通过网络连接的复杂存储系统，影响性能的因素众多，为了方便讨论，本节从硬件级、操作系统级和文件级分层讨论 I/O 性能的影响因素。

2.1.1　硬件级影响因素

万兆以太网连接

目前 Lustre 的客户端节点全部为刀片服务器，每个刀片

通过千兆网卡或者万兆网卡连接到刀片箱的万兆以太网交换机，每个刀片箱有一个万兆以太网的出口。

存储连接通道

目前每个 OSS 都通过 Express X8 的 HBA 卡连接两个磁盘阵列，阵列的 SCSI 通道带宽为 4Gb/s。因此，每个阵列存储通道的带宽限制为 500MB/s，每个 OSS 的存储连接通道上限为 1000MB/s。

SATA 磁盘阵列的 I/O 能力

磁盘 I/O 速度取决于磁头的移动速度和磁盘的转速，廉价的 SATA 磁盘在多进程访问的时候，磁盘移动频繁，会导致性能明显下降。

2.1.2 操作系统级性能影响因素

由于 OST 的本地文件读写最终表现为 Linux 对本地磁盘设备的块操作，针对前文讨论的特定 I/O 模式，对 Linux block 层不同的 I/O scheduler，nr_request，read_ahead_size 和 max_sector_size 做了测试。测试采用的软件 IOzone[6] 的块大小为 1MB，读模式为跳读，跳块大小为 2MB，并发进程数为 40。

I/O Scheduler

Linux 内核中每一个设备的 I/O 块请求都由专门的 Scheduler 来排队。对于磁盘设备来说，请求排队非常重要，因为它可以节省磁头移动的时间。在 Linux 内核中，有 4 种 I/O 调度的模式[7]。

- CFQ 可以对通用服务器的大多数负载保证调度的公平性，但是这种调度引入了 overhead 和 I/O 延时。
- AS 适合连接单个慢速磁盘存储的服务器，它总是试图聚合或者批量化 I/O 请求来提高慢速磁盘的性能。
- DEADLINE 是一种相对简单的调度器，它试图通过重新排序 I/O 请求、最小化 I/O 延时来提高性能。
- NOOP 是最简单的调度器，它只有一个 FIFO 队列，适用于连接了高性能存储设备的存储服务器。

通过测试发现，deadline 和 noop 对目标负载的调度性能相当，cfq 和 as 较差，如图 4 所示。

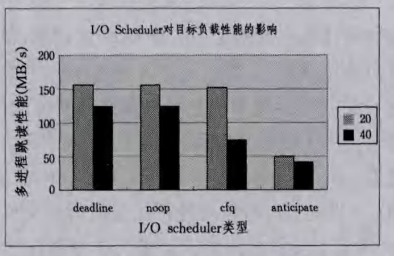

图 4 I/O Scheduler 对目标负载性能的影响

nr_request 代表每个块设备可以接收请求的队列长度，经过测试这个值对 20，40 个进程的跳读性能影响不大。

read_ahead_size 指每次 I/O 请求时磁盘向前读的数据大小，对于多进程跳读模式，readahead 的值对性能基本没有影响。

max_sector_size 指每个磁盘读写的数据大小，这个值小于 bulk I/O 的大小，会导致连续的 I/O 数据被分散到磁盘阵列的不同磁盘上。这个值是由阵列控制器的 stripe size 限制的，只能小于阵列的 stripe size，图 5 示出将这个值设成 32kB，64kB，128kB 对目标负载性能的影响。对于目标负载 I/O，max_sector_size 的值越大，每次 I/O 请求被分成的块数就越少。根据前文的讨论，目标负载对 OST 连续的 I/O 请求为 1MB，因此 max_sector_size 越大 I/O 性能越高，如图 5 所示。

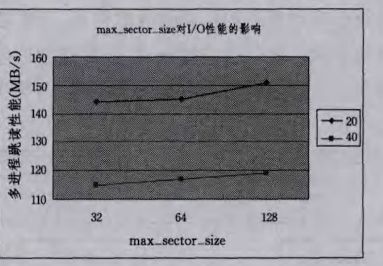

图 5 max_sector_size 对 I/O 性能的影响

2.1.3 Lustre 文件系统设置影响因素

Lustre 文件系统提高读写性能的主要方式是文件条带化存储和读预取。文件条带化存储时，系统将单个 I/O 请求分散到多个 OST 上，从客户端的角度可以提高单次文件访问的性能。不过，高能物理计算的设计目标是多个客户进程访问多个文件的聚合性能。在高吞吐率访问模式下，文件条带化的意义不大。

Lustre 客户端在两次连续读请求在内存中没有命中后，会触发预读。每个客户端对文件的预读上限为 40MB。预读机制可以增大 I/O 服务器上每次 I/O 请求的连续访问区域，减小磁头移动的频率，对于多进程的顺序访问可以提高 I/O 性能。但是，当内存不足时，预读的内容会被换出，下次需要时还要再读一次，这样就发生“预读抛弃”的现象，造成网络带宽浪费。由于网络带宽几乎全部被预读内容占用，正常请求得不到响应，使得计算任务出现 I/O 性能瓶颈，CPU 利用率大幅下降。曾经在计算节点上挂载 5 个 Lustre 文件系统，由于每个 Lustre 挂载点缺省可以使用 3/4 的本机内存，这样 5 个挂载点上的预读策略都起作用时，内存就不足了，发生了“预读抛弃”的现象，所有的 OSS 的带宽都全部跑满，达到 1GB/sec，但是计算节点的 IOWait 大幅增加，CPU 利用率降低到 10%以下，计算任务几乎无法正常完成。通过调整每个挂载点的内存限制，规避了“预读抛弃”现象，CPU 利用率恢复正常，如图 6 所示。

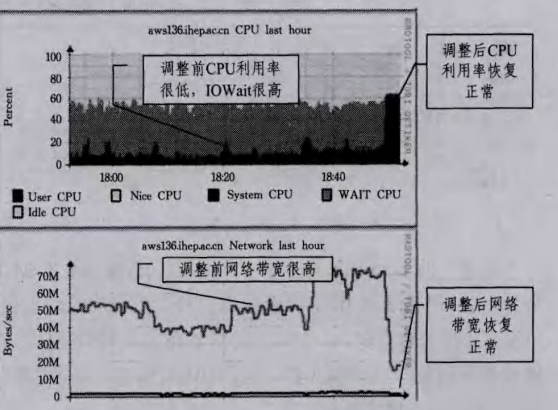

图 6 “预读抛弃”问题调整

2.2 分布式文件系统性能优化

在详细分析应用的文件访问模式以及影响存储系统的因素的基础上，对高能物理计算环境中的 Lustre 进行了如下优化：

1）SATA 磁盘的硬件特性是制约多进程跳读访问的性能的主要原因，因此在单个存储服务器上尽可能挂载更多的硬

盘。目前环境中典型的配置是一台存储服务器挂载 4 个阵列，每个阵列 16 块 SATA 硬盘，这样每台存储服务器在非常大的压力下仍能轻松达到 1GB/s 的读写性能，跑满了万兆网络。

2)对于大文件、大记录块的并发访问，使用合理的 I/O Scheduler，增大块设备的 max_sector_size，可以提高聚合性能。目前环境中，每个阵列的块设备都使用了 DEADLINE 的调度模式。

3)为了体现 Lustre 预读功能的作用，应用应该在不影响逻辑的前提下，增大读 buffer，减少跳读的频率。但是，为了规避“预读抛弃”现象，应该根据客户端机器内存的实际情况进行限制。

在进行如上的调优后，同时结合自动化监控运维等系统，高能物理研究所的 Lustre 并行文件系统提供了 3PB 以上的存储空间、20GB/s 以上的聚合带宽，能够支持 6000 以上的并发计算任务访问。

3 存储系统功能设计与优化

3.1 可管理性问题与分析

Lustre 文件系统在 HPC 领域有着非常广泛的应用，但是高能物理领域实验众多，需求各不相同。在实际应用过程中，Lustre 在可管理性等功能上存在较多不足。

3.1.1 元数据管理问题

Lustre 采用元数据和存储数据分离的技术，元数据存放在 MDS 服务器中。Lustre 系统能设置两个 MDS 服务器，采用主备模式工作。两个 MDS 服务器通过共享存储的方式来存放数据。当主 MDS 出现故障后，备份服务器能接管其服务，确保系统正常运行。但是，这种模式仍然存在一些问题，包括：1)共享存储一旦出现问题，将会是灾难性的事件，比如，共享存储底层文件系统错误，可能导致整个 Lustre 文件系统不可写，甚至不可读，从而造成业务中断和数据丢失；2)文件数目比较多时，元数据服务器性能会下降，甚至会影响到用户操作，使得系统变得不可用。

3.1.2 容错与可靠性问题

Lustre 在容错和数据可靠性方面也存在一些问题。目前，Lustre 不支持副本机制或者纠删码技术，数据的可靠性完全依赖于底层存储设备和 OSS 存储服务器。存储设备采用 RAID 技术，无论采用哪种 RAID 级别，其冗余度是固定的，且一旦出现问题，重建(Rebuild)时间比较长，对于系统的性能有较大影响。此外，RAID 技术只能屏蔽单台存储服务器上的硬盘故障，如果 OSS 服务器本身或者网络出现故障，仍然会导致系统不可用或者数据丢失，也是一个比较大的问题。

3.1.3 扩容问题

按需扩展是现代存储系统设计的基本原则，因此良好的系统扩容能力是必需的功能之一。扩容有两点需求，其一是在线扩展，不影响业务的运行；其二是扩容完成后能够自动进行存储系统的平衡，实现文件的均衡分布和数据访问性能的平衡。Lustre 能够实现第一点的在线扩展，但是却没有自动平衡的功能，导致扩容后严重的数据不均衡。

3.2 基于 Gluster 的存储系统构建

随着存储系统规模越来越大，应用越来越复杂，可管理性变得越来越重要，有必要考虑采用新的文件系统进行高能物理计算的存储系统设计。Gluster 是一个开源的分布式文件系统[8]，具有强大的横向扩展能力，通过扩展能够支持数 PB 存储容量和处理数千客户端。由于 Gluster 分布式文件系统采用无元数据服务器设计，具有很好的可扩展性，本文提出基于 Gluster 构建大规模高能物理存储系统，并针对目前存在的问题进行针对性的优化。

3.2.1 元数据管理

Gluster 没有专门的元数据服务器，因此无法保存逻辑文件视图与物理文件存储路径的映射关系。Gluster 的解决方法是，目录在所有存储服务器上全部都存在，要保持完全一致，文件定位采用 Davies-Meyer 算法计算文件名 hash 值，获得一个 32 位整数。假设存储卷中有 N 个存储服务器，则 32 位整数空间被平均划分为 N 个连续子空间，每个空间分别映射到一个存储服务器。这样，计算得到的 32 位 hash 值就会被投射到一个存储服务器。Gluster 的元数据管理方法消除了单点故障和性能瓶颈，具有非常好的可扩展性和性能。但是，在实际使用过程中存在两个问题需要解决：

1)每个存储服务器上的目录结构要保持完全一致，但是在高能物理计算中，多个进程会操作同一个目录，比如创建和删除目录等，这样会导致大量的“脑裂”问题，即不同存储服务器上同一个目录 ID(GFID)不一致。为此通过修改代码，实现了一个“统一目录层”，所有的删除和创建目录都通过该层，这样就构建了一个集中目录管理服务。在系统发现“脑裂”问题时，自动与这个“统一目录层”中的基准 GFID 同步，从而修正该问题。显然，该方案在高并发的目录创建以及删除场景中会形成瓶颈，而高能物理计算中主要体现的是文件访问的高并发，因此该方案不会造成性能问题。

2)在大压力场景下，元数据操作与数据操作混合会造成元数据操作“饥饿”。比如，读取一个 100MB 的文件，每次 I/O 块 128kB，需要 800 次 I/O 操作，而元数据操作请求，比如 stat 一个文件信息，一次 I/O 操作即可，但是在通信时与所有的 I/O 请求一起进行排队。这样在大量的数据操作过程中，元数据 I/O 请求基本被上被湮灭了，不能得到及时的响应，导致操作“饥饿”。为了解决这个问题，对 I/O 请求队列进行优化，分别建立不同优先级的 I/O 请求队列，把元数据操作队列的优先级设置为最高，且将其线程数量加大，以此缓解元数据操作“饥饿”的问题。

3.2.2 容错与可靠性

由于没有元数据服务器，目录结构存储在所有存储服务上，消除了元数据的单点故障。同时，Gluster 具有副本功能，能够屏蔽硬盘损坏、服务器故障、电源故障以及网络故障等，具有较好的数据可靠性。但是，仍然存在两个问题要解决：

1)Gluster 的副本机制是通过镜像来实现的，一旦某台存储服务器出现故障，其它存储服务器的压力过大，且数据恢复需要完全从镜像服务器上读取，除了增加额外的压力，数据恢复时间也过长。为此，提出多哈希的数据分布策略，在此基础上实现随机副本，该工作正在进行中。

2)数据容错仅支持副本模式，对于高能物理领域海量的数据来说，需要 2 倍甚至 3 倍的存储设备，成本过高。目前，基于柯西 RS 编码[9]实现 Gluster 的纠删码存储，可以实现任意磁盘和存储服务器的冗余，使其更好地满足高能物理的需求。实施时，常用的方式为 6+2 模式，需要 1.33～1.5 倍的存储设备。

3.2.3 系统扩容

系统扩容要求具有在线扩展与扩容后自动平衡的功能，

Gluster 基本上能够满足，但是有一个限制，即在平衡前，对于扩容前创建的老目录不能使用新增加的存储空间，只有新创建的目录才能使用所有存储设备。这一点导致 Gluster 扩容对于高能物理应用来说基本上是无用的，因为实验的目录基本上都是固定的。为此，提出基于版本的扩容方案，扩容后，系统存在新老等多个版本，文件定位在多个版本间进行，使得老目录也可以使用新增加的设备。当平衡完成后，所有版本合并为一个。通过扩容方案的修改，使得 Gluster 扩容能较好地满足高能物理的需求。

3.2.4 应用实例与性能

虽然 Gluster 具有很好的架构设计，但是针对高能物理计算来说，其元数据管理、容错与可靠性、扩容等方面还不能很好地满足需求。在对这些不足进行优化后，将 Gluster 应用到一个实际的高能物理实验（羊八井宇宙线实验）计算中，目前管理了 4 台存储服务器，186TB 的存储空间，近 1500 万的文件，支持 500 多个并发作业访问。实际运行显示，Gluster 能够满足计算的 I/O 需求，计算节点上基本不会出现 IO-Wait，CPU 利用率高。图 7 显示了其中一台存储服务器的数据访问性能，其优化措施与 2.2 节一致，最高时接近 1GB/sec，跑满了整个万兆网络带宽，说明 Gluster 具有很好的数据访问性能。

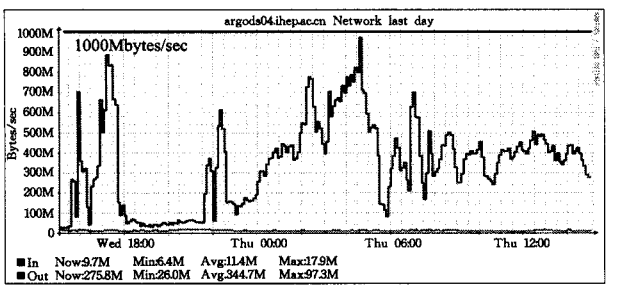

图 7 Gluster 存储服务器性能

结束语 海量的高能物理实验数据对于存储系统提出了很高的要求，既要求有极高的访问性能，还要有良好的可管理性。高能物理领域主要以自主开发、开源软件为主来构建海量存储系统。各种系统有各自的特点，每个应用也有自己的需求，因此希望一种存储系统满足各种应用的需求是不现实的。因此，必须要深入分析应用和存储系统的特点，并进行应用优化才能达到理想的效果。高能物理计算的需求一直在不断变化，存储系统的结构、技术和方法也会随着应用的需求而变化。

参 考 文 献

[1] WLCG -Worldwide LHC Computing Grid [OL]. http://lcg.web.cern.ch/LCG,2013.7

[2] Fuhrmann P, Gülzow V. dCache, storage system for the future [C]// Euro-Par 2006 Parallel Processing. Springer Berlin Heidelberg, 2006: 1106-1113

[3] Peters A J, Janyst L. Exabyte Scale Storage at CERN[J]. Journal of Physics Conference Series, 2011, 331(5)

[4] Schmuck F, Haskin R. GPFS: A Shared-Disk File System for Large Computing Clusters[C]// Proceedings of the Conference on File and Storage Technologies (FAST'02). Monterey, CA, January 2002: 231-244

[5] Schwan P. Lustre: Building a file system for 1000-node clusters [C]// Proceedings of the 2003 Linux Symposium. 2003

[6] IOzone Filesystem Benchmark[OL]. http://www.iozone.org

[7] Shakshober D J. Choosing an I/O scheduler for Red Hat Enterprise Linux 4 and the 2.6 kernel [M]. Red Hat magazine, 2005

[8] Gluster web site[OL]. http://www.gluster.org

[9] 罗象宏，舒继武. 存储系统中的纠删码研究综述[J]. 计算机研究与发展，2012，49(1)：1-11

（上接第 53 页）

结束语 本文利用静态调优和动态调优相结合的方式创建了一个渐进式智能回溯向量化代码调优架构。该架构能够有效地加强用户和编译器之间的交互，极大地减轻程序员编写向量化代码的压力，能够产生高效、易于阅读的向量化代码。利用该系统进行交互式调优，可以便捷、直观地进行代码级调优工作。交互调优界面中的向量化报告和阻碍向量化原因报告对程序员进行代码调试有很大的帮助。

但是我们的工作更多地还要依赖于自动向量化关键技术方法的发展，这样才能有更为丰富的调优方法和手段。我们的架构中用到的静态调优和动态调优的方法，很多是依赖于当前自动化技术的发展水平来定的。我们期待自动向量化技术能有更大的发展和突破，这样我们的调优架构也将更为丰富。

参 考 文 献

[1] Stewart J. An investigation of SIMD instruction sets. University of Ballarat School of Information Technology and Mathematical Sciences, 2005. [OL]. http://noisymime.org/blogimages/SIMD.pdf

[2] Nuzman D, Rosen I, Zaks A. Auto-Vectorization of interleaved data for SIMD[C]// Proc. of the ACM SIGPLAN Conf. on Programming Language Design and Implementation. Ottawa: ACM Press, 2006: 132-143

[3] 魏帅，赵荣彩，姚远. 面向 SLP 的多重循环向量化[J]. 软件学报，2012(7)：1717-1728

[4] 李玉祥，施慧，陈莉. 面向非多媒体程序的 SIMD 向量化算法的研究及改进[J]. 小型微型计算机系统，2009(10)：1927-1935

[5] 白书敬，李中升，漆锋滨. 反馈式编译优化技术浅析[J]. 高性能计算技术，2005，10(5)：1-5

[6] 郝云龙，赵荣彩，侯永生，等. 反馈式编译在循环级性能分析中的应用[J]. 计算机工程，2011，5(5)：32-34

[7] 姚远，赵荣彩. 基于 Profile 信息的连续性分析算法及其优化[J]. 计算机工程，2012(9)：28-31