Intel IT Center

2012 年 10 月

Apache Hadoop* 社区聚焦

Apache* HDFS*

Apache Hadoop* 框架项目管理委员会成员兼 AltoScale 创始人 Konstantin Shvachko 详细介绍了 Apache* Hadoop* 分布式文件系统（简称 Apache HDFS* ），以及该软件今后的开发方向。

Apache HDFS：面向海量数据的分布式存储

从最基本的层面来说，Apache Hadoop 分布式文件系统（简称 Apache HDFS）是 Apache 开源项目 Hadoop 下应用程序所使用的主要分布式存储组件。同时，HDFS 还能作为独立的分布式文件系统使用。

Apache Hadoop 框架包括大量支持分布式计算的组件，用以解决大数据问题。位于堆栈最底层的是 Hadoop Common 组件，该组件中包含的实用程序负责为其他 Hadoop 模块提供支持。下一层是 HDFS，为其他层处理提供基础，其中包括 Apache MapReduce、Apache HBase* 数据库、Apache Hive* 数据仓库基础设施以及 Apache Pig* 数据流语言等。

此外，HDFS 还提供了一个抽象系统，一个 API 系统可支持用户部署其他分布式文件系统或存储系统，如 Amazon* 简单存储服务（S3）、开源并行虚拟文件系统（PVFS）、IBM* 通用并行文件系统*（IBM GPFS*）、Lustre* 文件系统，以及最近推出的 MapR* Map-Reduce Ready Distributed File System 等。

工作模型

从用户的角度而言，HDFS 是一种包含文件和目录层次结构的传统文件系统。它通过在商用服务器集群上运行的两项服务实现：

- **单一的命名节点（NameNode）**负责保持目录树结构，并管理命名空间和客户端对文件的访问。
- **数据节点（DataNode）**负责存储并管理数据块，将其作为集群其余服务器上的本地文件。

“Hadoop 2.0 对 HDFS* 进行了全新优化，并增添了许多重要的新功能，这是我们近期可实现的一项最具价值的改进举措。”*

— Konstantin Shvachko

HDFS 将文件拆分成大型数据块，并在多个数据节点中进行复制。此举旨在为文件数据提供冗余支持和可用性保证。元数据存储在命名节点中，负责执行命名空间操作（例如打开、关闭和重命名文件等），以及将数据块映射至数据节点。在当前的设计中，单一的命名节点将整个命名空间存储在内存（RAM）中，而数据在数据节点中复制和存储。数据节点负责支持来自客户端的请求，并管理命名节点所指定的数据块操作，例如删除和复制等。

在这一模型中，命名空间与数据分离，以支持实现对于元数据的最佳访问。尽管对元数据的请求通常简短，且频率不高，但典型的 Hadoop 查询会生成大量请求。这一分离设计可在不影响元数据操作的情况下，从数据节点传输数据流。

用户应用程序可使用 HDFS 客户端来访问文件系统，其中 HDFS 客户端是一个能够导出 HDFS 界面的库。HDFS 客户端可访问命名节点以获取有关某个数据文件的元数据，包括文件块的位置等，随后根据元数据所指定的要求将数据传输至数据节点，或从数据节点传输至客户端。

可靠、可扩展的处理和存储：四大设计原则

HDFS 开源社区的开发工作专注于四项关键的设计原则：

线性可扩展性：HDFS 旨在处理海量非结构化和结构松散的数据。由于数据量不断攀升，计算和存储的可扩展性变得至关重要。HDFS 可支持线性扩展，只需简单地向集群添加更多节点（或服务器），便可以加快数据的处理速度，同时提高存储容量。

系统可靠性和可用性。由于大型 Hadoop 集群中部署了大量硬件组件，因此驱动器发生故障的情况屡见不鲜。例如，平均情况下，一个驱动器预计每三年就会发生一次故障。当您在一个集群中部署了1,000 个甚至更多驱动器时，每天都将会有驱动器出现故障。HDFS 具有内置的容错能力和自动恢复功能。通过将数据分解为多个数据块，并在集群中的其他服务器中存储副本，HDFS 可建立冗余文件系统。当驱动器发生故障时，这一冗余意味着数据依然可用，而且系统仍可继续处理请求。

在数据所在的位置进行计算。过去，存储和计算在不同的位置进行。用户从存储系统中获取数据，进行处理，然后再将其返回到存储系统。Hadoop 将计算工作移至数据所在的位置进行。MapReduce 能够将工作负载分配至存储数据的服务器，从而可减少大型数据传输的时间，提高吞吐率。这一设计在处理海量数据时的优势最为明显。

顺序数据处理。HDFS 针对批处理进行了优化，专注于提高整体吞吐率，而非个别操作的延迟。顺序操作能够避免随机 I/O 或可显著减缓 I/O 操作的寻道问题。

Apache* HDFS* 如何处理大型、非结构化的数据集？

Apache* HDFS* 分布式文件系统（简称 HDFS*）并未规定数据集结构，而是从文件集具备的特征进行处理。对于 HDFS 而言，每个文件只是一组字节，而文件结构是在应用层予以实现的。因此，Apache Hadoop* 框架能够处理众多来源中任何非结构化的或结构松散的数据，如博客、推特、商业文档、DNA 数据，以及高能量物理数据（大型强子对撞机）等。由于 HDFS 顺序处理数据，因此当用户将大型数据集作为一个整体分析，而不是试图定位数据集中某个特定的元素时，其优势将最为明显。

HDFS 的局限性

HDFS 能够可靠地存储超大型数据集，并可将这些数据传输至用户应用程序。但是，随着数据量的不断激增，HDFS 以及其他 Hadoop 组件将逐渐受限于其单主机设计。由于元数据存储于单一命名节点上的内存中，因此在该内存中可存储的对象数量将受到限制。[1] 此外，系统中的大型负载有可能超过单一命名节点的处理能力，从而导致性能瓶颈。

Apache Hadoop 社区正致力于突破这些可扩展性的局限。其中一种可能的解决办法是采用“联合”的方式。这种联合功能已在代码中准备就绪，只是尚未包含于稳定版本中。它能够在一个集群中使用多至 10 个命名节点，并共享同一个数据节点池。这一功能将在下一个稳定版本 Hadoop 2.0 中提供。[2]

然而，即便如此，联合功能最终也会导致局限性。这是因为当一个集群中包含 10 个命名节点时，用户会看到 10 个不同的文件系统或卷。由于无法轻易地将元数据卸载到其他命名节点，如果这些卷中有一个的增长速度超过其他卷的增长速度，那么该命名节点将会超载。命名空间动态分区可解决该问题，而这也是 Giraffa 文件系统 [3]（一个新的 Apache Extras* 项目）的目标。Giraffa 采用 HBase 数据库来存储元数据（HBase 数据库是 Hadoop 框架的另一个存储组件，正在快速发展成熟）。传统 HDFS 数据节点将继续提供数据存储服务。这样一来，文件系统中的元数据和数据部分都将成为分布式系统。这与 Google 当前处理其 Colossus 文件系统的方式有异曲同工之妙。

[1] 本文章包含与内存限制和可存储的对象数量相关的特定数字。Shvachko, Konstantin V. “HDFS 可扩展性：增长的限制。”Usenix；Login：35，No. 2（2010 年 4 月）http://c59951.r51.cf2.rackcdn.com/5424-1908-shvachko.pdf

[2] 当前稳定版本为 Hadoop 1.0。

[3] 如欲了解有关 Giraffa 文件系统的更多信息，请参阅 2012 年 Hadoop 峰会中 Shvachko, Konstantin V. 和 Plamen Jeliazkov 的演示视频。“利用 Giraffa 文件系统进行动态命名空间分区”2012 年 Hadoop 峰会（2012 年 6 月 14 日）youtube.com/watch?v=tRVLNm_HM3I&feature=youtu.be

HDFS 的后续发展

HDFS 社区正积极增强软件的稳定性，以推出 Hadoop 2.0 版本。Hadoop 2.0 将包含至关重要的全新优化和功能，使其成为近期优先级最高的开发任务。从长期来看，下一代 Hadoop 以及 HDFS 的核心仍将是四大设计原则。例如，改善可扩展性始终非常重要，因为数据将会不断增长，且计算能力也将会继续提高。此外，社区将专注于不断提高可用性，并扩展功能，以支持实时处理。

Apache Hadoop 框架今后的一个重要发展前景为将 Hadoop 与云计算和虚拟化相结合。将这些技术汇聚一堂将实现 1+1+1 > 3 的效果，进而解决两大问题：提高 CPU 的利用率，并分离集群中的资源。

Apache* HDFS* 是否符合 POSIX 的要求?

Apache* Hadoop* 分布式文件系统（简称 Apache HDFS*）并不完全符合便携式操作系统界面（POSIX）的要求。POSIX 是一系列的标准，用于保证操作系统间的兼容性。由于 HDFS 并未完全实施所有标准，因而提高了处理超大型数据集的性能。

但是，HDFS 符合 POXIS 绝大多数的要求。它不支持对目录、随机写入和硬链接进行完全访问。但在硬链接中，符号链接（symlink）适用于大多数 Hadoop 应用程序。

本文摘自 2012 年 8 月 24 日对 Konstantin Shvachko 的采访。如欲查看完整采访内容，请收听播客。

如欲了解有关 Apache HDFS 项目的更多信息，请访问 http://hadoop.apache.org

与同事分享

1012/RF/ME/PDF-CN 328129-001

英特尔®与你共创明天™