第 3 卷 第 4 期
2014 年 7 月
集 成 技 术
JOURNAL OF INTEGRATION TECHNOLOGY
Vol. 3 No. 4
Jul. 2014

天玑大数据引擎及其应用

查 礼 程学旗

（中国科学院计算技术研究所网络数据科学与工程重点实验室 北京 100190）

摘 要 大数据计算面对的是传统 IT 技术无法处理的数据量超大规模、服务请求高吞吐量和数据类型异质多样的挑战。得益于国内外各大互联网公司的实际应用和开源代码贡献，Apache Hadoop 软件已成为 PB 量级大数据处理的成熟技术和事实标准，并且围绕不同类型大数据处理需求的软件生态环境已经建立起来。文章介绍了大数据计算系统中存储、索引和压缩解压缩的硬件加速三项研究工作，即 RCFile、CCIndex 和 SwiftFS，有效解决了大数据计算系统的存储空间问题和查询性能等问题。这些研究成果已形成关键技术并集成在天玑大数据引擎软件栈中，直接支持了淘宝和腾讯公司的多个生产性应用。

关键词 大数据引擎；数据存储；行列混合；聚簇索引
中图分类号 TP 316.4 TP 319 **文献标志码** A

Golaxy Big Data Engine and Its Applications

ZHA Li CHENG Xueqi

(*Key Laboratory of Network Data Science and Technology*, *Institute of Computing Technology*, *Chinese Academy of Sciences*, *Beijing* 100190, *China*)

Abstract Volume, variety and velocity are the three challenges that the big data computing must be faced with, which cannot be dealt with by traditional IT technologies. Benefiting from numerous domestic and overseas Internet companies' practical applications and continuous code contributions, the Apache Hadoop has become a mature software stack and the de facto standard of the PetaByte scale data processing. Furthermore, around different types of data processing requirements, different software ecosystems have been established. In the big data system field, three research works of data placement, index construction and compression and decompression hardware acceleration, i.e. RCFile, CCIndex and SwiftFS respectively, effectively solving the storage space and query performance issues, were introduced in this paper. The above research achievements have been already integrated into the Golaxy big data engine software stack in the form of key technologies, and directly supported multiple practical applications of Taobao Inc. and Tencent Inc.

Keywords big data engine; data storage; row columnar; clustered index

收稿日期：2014-4-18
基金项目：国家高技术研究发展计划（863 计划）（2013AA01A213）。
作者简介：查礼（通讯作者），副研究员，博士，研究方向为分布式计算系统，E-mail：char@ict.ac.cn；程学旗，副总工程师、研究员，博士生导师，研究方向为网络科学与社会计算、互联网搜索与挖掘、网络信息安全、分布式系统与大型仿真平台等。

1 引　言

近年来，越来越多的国内外互联网公司和传统企业都已意识到数据资产规模化带来的潜在价值。这些呈爆炸性增长的数据资产的类型以非结构化和半结构化为主，如何低成本且高效率地存储和处理 PB[1] 至 EB[2] 量级的数据成为业界面临的极大挑战。谷歌(Google)公司陆续提出了 MapReduce[1]编程框架，GFS[3] 文件系统[2]以及 BigTable[3]存储系统，从而成为大数据处理技术的开拓者和领导者。而源于这三项技术的 Apache Hadoop[4]等开源项目则成为大数据处理技术的事实标准，迅速推广应用于国内外各大互联网企业，成为 PB 量级大数据处理的成熟技术和系统。天玑大数据引擎是构建在 Hadoop 之上的面向大数据计算(Big Data Computing)的工具集，其中包含了很多天玑团队既有的研究成果。这些研究成果已在互联网公司实际生产系统上经受住考验，如 RCFile 已应用到 Facebook(脸书)公司、CCIndex 已应用于淘宝网的“数据魔方”、ICTBase 已应用到腾讯“广点通”等。这些关键技术构成了天玑大数据引擎的核心竞争力，软件的生态环境也借由开源社区得到良性发展。

2 大数据技术现状与发展趋势

什么是大数据？麦肯锡公司的报告《大数据：创新、竞争和生产力的下一个前沿领域》中给出的大数据定义是：大数据指的是规模超过现有数据库工具获取、存储、管理和分析能力的数据集，并强调并不是超过某个特定数量级的数据集才是大数据。国际数据公司(IDC)用四个维度的特征来定义大数据，即数据集的规模(Volume)、数据流动的速度(Velocity)、数据类型的多少(Variety)和数据价值的大小(Value)。亚马逊的大数据科学家 John Rauser 的定义比较直接：“超过单台计算机处理能力的数据量则为大数据”。最后我们来看看维基百科上的大数据定义：大数据指的是数据规模庞大和复杂到难以通过现有的数据库管理工具或者传统的数据处理应用程序进行处理的数据集合。

上述大数据的概念中无一例外地都突出了“大”字。从表面上看，数据规模的增长的确为处理数据带来了很大的问题。具体来说，在同样时间内获取与以前相同价值的数据变得不可为了。换言之，本质问题是数据的价值密度变低了，数据交换速率变慢了，所以催生了很多新型数据处理技术和工具，如 Google 的谷歌文件系统(GFS)和 MapReduce、Apache Hadoop 生态系统、美国伯克利大学 AMPLab 的 Spark 等。同时出现了对时间敏感程度不同的计算模式，如批式计算模式、交互式计算模式、流计算模式和实时计算模式等。计算模式的差异决定了获取价值的技术不同，其选用取决于上层业务需求。实际上，所谓大数据问题的本质应是数据的资产化和服务化，而挖掘数据的内在价值是研究大数据的最终目标。如何解决数据资产化和价值挖掘问题，以及如何保证需求挑战和技术选型之间的平衡已经成为业界关注的焦点。

2.1 谷　歌

谷歌在搜索引擎上所获得的巨大成功，很大程度上是由于采用了先进的大数据管理和处理技术。这些技术是针对搜索引擎所面临的日益膨胀的海量数据存储问题以及在此之上的海量数据处理问题而设计的。

针对内部网络数据规模超大的特点，谷歌提出了一整套基于分布式并行集群方式的基础架构技术，利用软件处理集群中经常发生的节点失效问题。谷歌使用的大数据平台包括四个相互独立又紧密结合在一起的系统：GFS、针对谷歌应用程序的特点提出的 MapReduce 编程模式、分布式的锁机制 Chubby 以及大规模分布式数据库

[1]1015(千万亿)字节
[2]1018(百亿亿)字节
[3]Google File System，谷歌公司为了存储海量搜索数据而设计的专用文件系统

BigTable。

GFS 是一个大型的分布式文件系统，它为谷歌云计算提供海量存储，并且与 Chubby、MapReduce 和 BigTable 等技术结合得十分紧密，处于系统的底层。它与传统的分布式文件系统有许多相同的目标，例如性能、可伸缩性、可靠性以及可用性。除此之外，它的设计还受到谷歌应用负载和技术环境的影响。相对于传统的分布式文件系统，为了达到成本、可靠性和性能的最佳平衡，GFS 从多个方面进行了简化：(1)采用集中式元数据管理；(2)不缓存数据；(3)在用户态下实现；(4)只提供专用接口。另外，GFS 还将节点失效视为系统的常态，提供了极强的系统容错功能；设置三个数据块副本，以增强数据可靠性；使用了链式写和版本控制的双重保证，以确保数据一致性，即数据块的所有在线副本组成一条写更新链，用户进行写操作时，数据链式写入所有副本，当链上的所有副本都完成更新后，写操作才会成功，并更新对应数据块的版本号。

MapReduce 是处理海量数据的并行编程模式，用于大规模数据集的并行运算。MapReduce 通过“Map(映射)”和“Reduce(化简)”这两个简单的概念来参加运算。用户只需要提供自己的 Map 函数以及 Reduce 函数就可以在集群上进行大规模的分布式数据处理。这一编程环境能够使程序设计人员在编写大规模的并行应用程序时不用考虑集群的可靠性和可扩展性等问题。应用程序编写人员只需要将精力放在应用程序本身，关于集群的处理问题则交由平台来完成。与传统的分布式程序设计相比，MapReduce 封装了并行处理、容错处理、本地化计算和负载均衡等细节，具有简单而强大的接口。正是由于 MapReduce 具有函数式编程语言和矢量编程语言的共性，使得这种编程模式特别适合于非结构化和结构化的海量数据的搜索、挖掘和分析等应用。

Chubby 是提供粗粒度锁服务的一个文件系统，它基于松耦合分布式文件系统设计可靠的存储，解决了分布的一致性问题。这种锁只是一个建议性的锁而不是强制性的锁。通过使用 Chubby 的锁服务，用户可以确保数据操作过程中的一致性。GFS 使用 Chubby 来选取一个 GFS 主服务器，BigTable 使用 Chubby 指定一个主服务器并发现和控制与其相关的子表服务器。

大规模分布式数据库 BigTable 是基于 GFS 和 Chubby 开发的分布式存储系统。很多应用程序对于数据的组织是非常有规则的。一般来说，数据库对于处理格式化的数据是非常方便的，但是由于关系数据库要求很强的一致性，很难将其扩展到很大的规模。为了处理谷歌内部大量的格式化以及半格式化数据，谷歌构建了弱一致性要求的大规模数据库系统 BigTable。BigTable 在很多方面和数据库类似，但它并不是真正意义上的数据库。而谷歌包括 Web 索引和卫星图像数据等在内的很多海量结构化和半结构化数据都是存储在 BigTable 中的。BigTable 的内容按照行来划分，将多个行组成一个小表(Tablet)，保存到某一个服务器节点中。

2.2 Hadoop

Apache Nutch 是 Hadoop 的源头。该项目始于 2002 年，是 Apache Lucene 的子项目之一。当时的系统架构尚无法扩展到存储并处理拥有数十亿网页的网络化数据。谷歌于 2003 年在 SOSP[4] 上公开了描述其分布式文件系统的论文 The Google File System(《谷歌文件系统》)，为 Nutch 提供了及时的帮助。2004 年，Nutch 的分布式文件系统(NDFS[5])开始研发。同年，谷歌在 OSDI[6] 上发表了题为 MapReduce: Simplified Data Processing onLarge Clusters(《MapReduce—简化的大规模集群数据处理》)的论文，受到启发的道·卡廷(Doug Cutting)等人开始实现 MapReduce 计算框架，并与 NDFS(Nutch Distributed File System)结合起来，共同支持 Nutch 的主要算

[4] Symposium on Operating Systems Principles，操作系统原理会议
[5] Nutch Distributed File System
[6] Operating Systems Design and Implementation，操作系统设计与实现国际会议

法。至 2006 年，这个框架逐渐成为一套完整而独立的软件，命名为 Hadoop。2008 年初，Hadoop 成为 Apache 的顶级项目，不仅用于雅虎(Yahoo!)，还在众多互联网企业得以应用。

Hadoop 的核心由两部分组成：HDFS 和 MapReduce，其中 HDFS 是 Google GFS 的开源版本，是一个高可靠的分布式文件系统。它能够提供高吞吐率的数据访问能力，适合存储海量(PB 级)数据，其实现原理如图 1 所示。

HDFS 全部在用户态使用 Java 语言编写。目录节点(NameNode，也是主节点)在系统中只有一个实例，采用键值(Key-Value)全内存式管理模式，用于管理文件系统的元数据。元数据包括名字空间、副本数量及位置与文件到块的映射关系。数据节点(DataNode)负责固定大小数据块的存储(通常为 64 MB)。一个文件(home/foo/data)由存储在多个数据节点上的数据块构成，客户端访问数据时经由目录节点获得数据块的存储位置，再与数据块所在的数据节点交互，写入或读出数据。

MapReduce 计算框架实现了由谷歌工程师提出的 MapReduce 编程模型，其原理如图 2 所示。

当一个 MapReduce 作业提交给 Hadoop 集

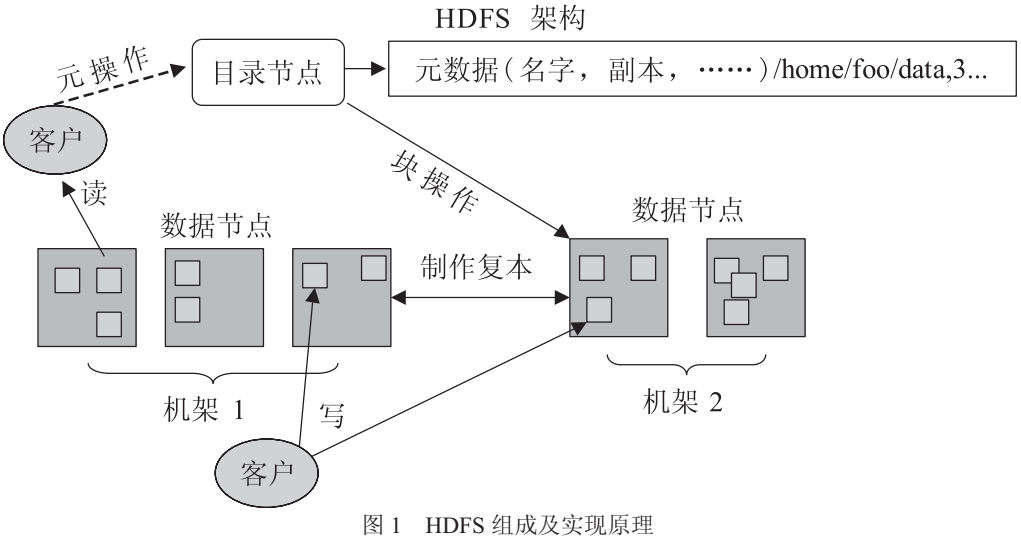

图 1　HDFS 组成及实现原理

Fig. 1.　Implementation principle and components of HDFS

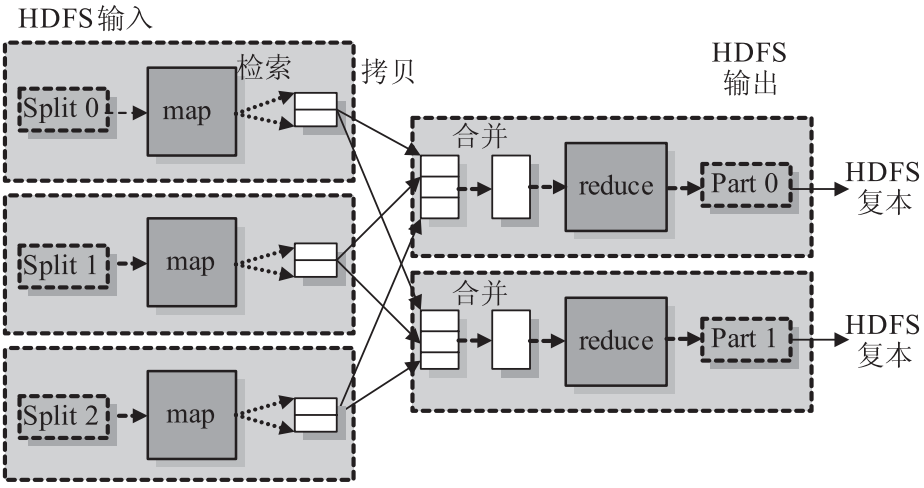

图 2　MapReduce 编程模型示意图

Fig. 2.　Programming model of MapReduce

群时，相关的输入数据将首先被划分为多个片断，然后由作业跟踪程序(Job Tracker)挑选空闲的任务跟踪器(Task Tracker)对数据片断并行地执行 Map 任务。接着这些由 Map 任务产生的中间记录会被再次划分并由作业跟踪程序挑选空闲的任务跟踪器对它们并行地执行 Reduce 任务，从而获得和每个键值相对应的数据集合作为运算结果。这样的过程将被反复执行，直到 MapReduce 作业中所有的 Map 任务和 Reduce 任务执行完毕。

虽然在 Hadoop 中有名的是 MapReduce 及其分布式文件系统 HDFS，但还有其他关联项目支持开发的工具提供配套和补充性服务。这些关联项目之间的关系如图 3 所示。各关联项目的特征如下：

(1) HDFS：以块数据为单位存储并具有副本机制的分布式文件系统；

(2) MapReduce：分布式数据处理模式和执行环境；

(3) Hive：分布式数据仓库，用于管理 HDFS 中存储的数据，并提供基于 SQL 的查询语言(由运行时解释引擎转换为 MapReduce 作业)用以查询数据；

(4) Mahout：其主要目标是构建可扩展的机器学习算法库，目前主要支持三种类型用例，推荐、聚类和分类；

(5) Pig：一种运行在 MapReduce 和 HDFS 的集群上的高层(High Level)数据流语言和运行环境，用以检索海量数据集；

(6) Oozie：一个对 Hadoop 上运行作业进行管理的工作流调度器系统。Oozie 工作流是放置在 DAG(有向无环图 Direct Acyclic Graph)中的一组动作(例如，Hadoop 的 Map/Reduce 作业、Pig 作业等)，其中指定了动作执行的顺序。使用 hPDL(一种 XML 流程定义语言)来描述 DAG；

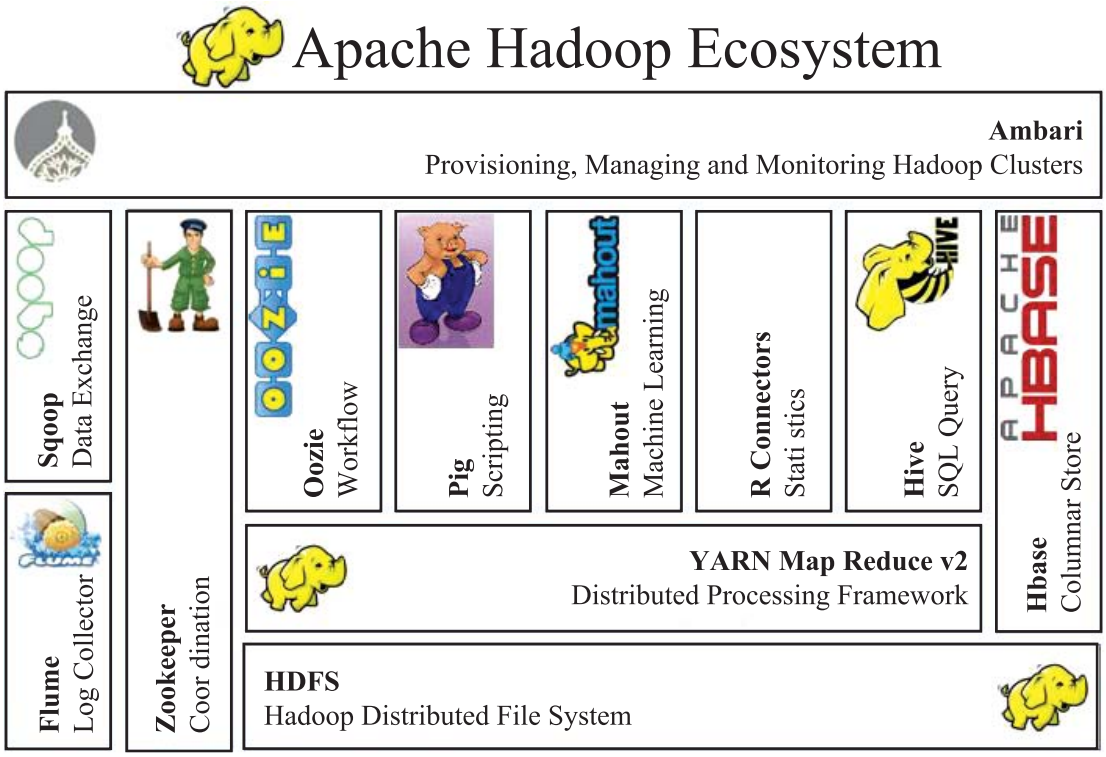

图 3 Apache Hadoop 生态系统构成示意图

Fig. 3. Apache Hadoop ecosystem

(7) HBase：一个分布式列存储数据库，使用 HDFS 作为底层存储，同时支持 MapReduce 的批式计算和点查询(随机读取)；

(8) ZooKeeper：一个分布式高可用的协同服务，提供分布式锁相关的基本服务，用于支持分布式应用的构建；

(9) Flume：一个高性能的日志收集系统，具有分布式、高可靠、高可用等特点，支持对海量日志采集、聚合和传输。Flume 支持在日志系统中定制各类数据发送端，同时，Flume 提供对数据的简单处理，并具有分发到各数据接收端的能力；

(10) Sqoop：一个可将 Hadoop 和关系型数据库中的数据相互迁移的工具。可将关系数据库(例如 MySQL、Oracle、Postgres 等)中的数据导入到 Hadoop 的 HDFS 中，也可以将 HDFS 的数据导入到指定的关系数据库中；

(11) Ambari：一个基于 Web 的 Hadoop 机群管理和监控工具。Ambari 目前已支持大多数 Hadoop 组件，包括 HDFS、MapReduce、Hive、Pig、HBase、Zookeeper、Sqoop 和 Hcatalog 等。

2.3　Spark

Apache Spark 是由加州大学伯克利分校 AMP 实验室(Algorithms, Machines and People Lab)于 2011 年启动研发的分布式大数据计算软件栈。如图 4 所示，Spark 与 Hadoop 具有很多相似和相融之处，比如两者均可定位为面向大数据的分布式计算系统，Spark 目前还依赖 Hadoop 中的 HDFS 作为数据存储等。Spark 比 Hadoop 优越的地方在于可以支持多种类型负载。换句话说，Spark 可以同时支持批式、流式、迭代和实时这四种大数据计算模式。Spark 的核心技术和创新在于引入名为弹性分布式数据集(Resilient Distributed Dataset，RDD)的系统抽象。RDD 是分布在一组节点中的只读对象集合。Spark 对 RDD 的操作有两种类型，即转换操作(Transformations)和行动操作(Actions)。不像 Hadoop 只提供了 Map 和 Reduce 两种计算方式，转换操作包括 map、filter、flatMap、sample、groupByKey、reduceByKey、union、join、cogroup、mapValues、sort 和 partionBy 等；行动操作包括 count、collect、reduce、lookup 和 save 等。因为 RDD 为只读，所以转换操作会生成一个新的 RDD，但并不是立即执行这个计算，只有遇到行动操作的时候才会真正进行计算。这种设计使得 Spark 可以更加高效地运行。

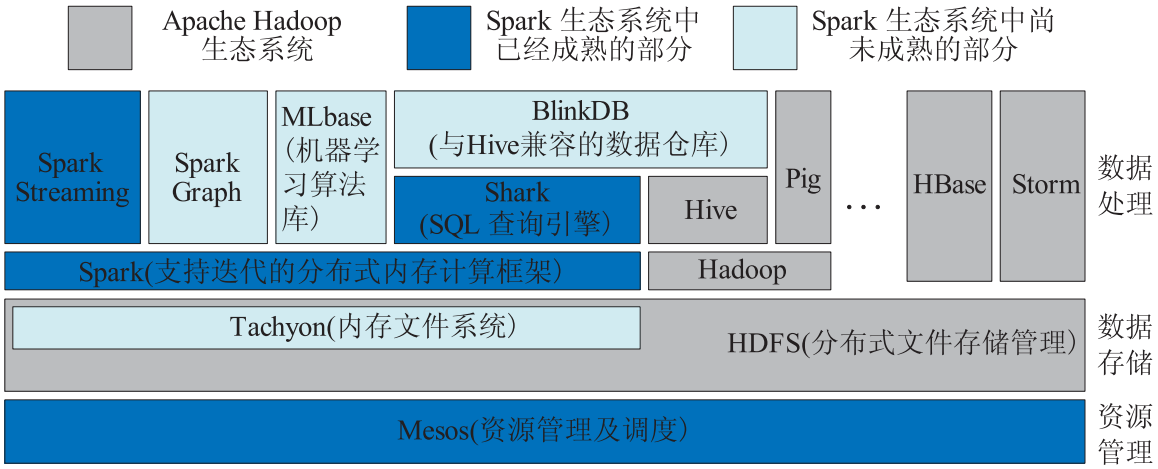

图 4　Spark 与 Hadoop 生态系统间的关联关系

Fig. 4.　Apache Stack and Hadoop software stack comparison

为了充分利用机群的聚合 I/O 能力，需要对 RDD 进行分区，以便数据分散存储到机群的各个服务器上，这时会根据记录的主键进行分区(如 Hash 分区)。RDD 根据 useDisk、useMemory、deserialized 和 replication 四个参数的组合可以提供 11 种存储级别。RDD 可以全部装载到内存中，每次 RDD 的转换操作结果都可以存放到内存中，而下一个转换操作可以直接从内存中输入，省去了 MapReduce 大量的磁盘 I/O 操作。这对于迭代运算比较常见的机器学习和数据挖掘算法来说，系统性能提升很大。RDD 都是可序列化的，当前 RDD 默认存储于内存，但当内存不足时，RDD 会自动溢出到磁盘，这一过程对编程人员以及最终用户来说是透明的。为了保证 RDD 中数据的可靠性，RDD 数据集通过所谓的血统关系(Lineage)记住了它是如何从其他 RDD 中演变过来的。相比其他系统的细颗粒度内存数据更新级别的备份或者 LOG 机制，RDD 的 Lineage 记录的是粗颗粒度的特定数据转换操作。当这个 RDD 的部分分区数据丢失时，它可以通过 Lineage 获取足够的信息来重新运算和恢复丢失的数据分区。这种粗颗粒的数据模型，会限制 Spark 的适用场景，但同时相比细颗粒度的数据模型，也带来了性能的提升。

Spark 使用 Scala 开发，默认使用 Scala 作为编程语言。编写 Spark 程序比编写 Hadoop MapReduce 程序要简单得多，写 Spark 程序的一般步骤就是创建或使用(SparkContext)实例，使用 SparkContext 创建 RDD，然后就是对 RDD 进行转换操作或行动操作。Spark 中的应用程序称为驱动程序，可以在单节点上或机群上执行。对于机群执行，Spark 依赖于 Mesos 资源管理器。Mesos 为上层应用提供了必要的分布式资源管理与调度功能。

2.4 发展趋势

从当今 IT 技术的发展角度看，提出系统结构上的解决方案是“应用驱动的大数据架构与技术”。也就是说根据具体类型应用的需求，在系统架构和关键技术上进行创新。为了降低成本并获得更好的能效，大数据应用系统越来越趋向扁平化和专用化的系统架构和数据处理技术，逐渐摆脱了传统的通用技术体系。这让传统的应用服务器、数据库服务器和存储服务器这样的典型三层架构受到极大的冲击。应用开发人员更深入地理解计算机系统结构，“程序”＝“算法”＋“数据结构”将逐渐演变成“程序”＝“算法”＋“数据结构”＋“系统结构”。

2.4.1 大数据技术生态环境范围扩大

克隆了 Google 的 GFS 和 MapReduce 的 Apache Hadoop 自 2008 年以来逐渐被互联网企业接纳，并成为大数据处理领域的事实标准。但 2013 年出现的 Spark 作为一匹黑马可以说终结了这一神话，大数据技术不再一家独大。由于应用不同导致 Hadoop 一套软件系统不可能满足所有需求，在全面兼容 Hadoop 的基础上，Spark 通过更多的利用内存处理大幅提高系统性能。此外，Scribe、Flume、Kafka、Storm、Drill、Impala、TEZ/Stinger、Presto 和 Spark/Shark 等的出现并不是取代 Hadoop，而是扩大了大数据技术生态环境，促使生态环境向良性和完整发展。今后在非易失存储层次、网络通信层次、易失存储层次和计算框架层次还会出现更多、更好和更专用化的软件系统[11]。

随着“数据意识”逐渐增强，尤其是传统企业将采纳 Hadoop 这样的技术，将企业生产运营相关的数据资产保存起来。这一进步将扩大 Hadoop 技术的应用场景，产生相当规模的企业大数据计算市场。企业大数据计算市场将基于开源大数据技术生态环境，由商业化 IT 技术公司主导，让开源大数据技术进入企业计算市场，将企业计算的特征融入进来，形成有依赖又相对独立发展的商业大数据技术生态环境。数据成为资

产以后，还将产生数据交易市场。尽管数据交易的形式和内容还需要探讨，但可以预见将产生以数据共享与交易为中心的大数据生态环境，将产生在政策法规允许范围内出现明码实价的数据交易。

2.4.2 实时和精准要素更为应用重视

在全球互联网企业的努力下，Hadoop 已经可以处理百 PB 级的数据，在不考虑时间维度的前提下，可以处理价值密度低的数据了。在解决了传统关系型数据库技术无法处理如此量级的数据之后，业界正在向速度和准确度两个方向要价值。换句话说，如何从大数据中获取更大回报成为业界新的挑战。互联网服务强调用户体验，原本做不到实时的应用在向实时化靠拢，比如前端系统及业务日志从产生到收集入库的延迟从 1 到 2 天时间进化到 10 秒以内。传统企业无法忍受关系数据库动辄几十分钟的查询分析性能，纷纷求助于性价比更好的技术和产品。这些需求使大数据交互式查询分析、流式计算和内存计算成为业界研发和应用的新方向。

作为批式计算的补充，交互式分析计算的目标是将 PB 级数据的处理时间缩短到秒级。Apache Drill 是开源的 Dremel 实现，虽已有应用但尚不成熟。由 Cloudera 公司主导的 Impala 也是参照 Dremel 实现，同时也参考了 MPP（Massively Parallel Processing）的设计思想，目前已经接近实用阶段。Hortonworks 公司主导的 TEZ/Stinger。TEZ 是运行在 YARN（Hadoop 2.0 的资源管理框架）上的 DAG 计算框架，而 Stinger 是下一代的 Hive（SQL on Hadoop 的事实标准），不同的是 Hive 将 SQL 解析为 MapReduce 命令执行，而 Stinger 将 SQL 解析为能够在 TEZ 上执行的 DAG，从而解决计算实时性问题。2013 年底，由 Facebook 开源的 Presto 分布式 SQL 查询引擎可对 250 PB 以上的数据进行交互式分析，比 Hive 的性能高出 10 倍。类似的 Shark 是 Spark 上的 SQL 执行引擎，得益于 Shark 的列存储和 Spark 的内存处理等特性，Shark 号称可以比 Hive 的性能提高 100 倍。

在实时计算、机器学习和深度学习等技术的支撑下，个性化推荐已经开始从简单的商品推荐走向复杂的内容推荐。根据用户的特性与偏好，推荐内容的特征以及当时的上下文数据（客户端设备类型、用户所处时空数据等），向特定用户提供个性化的内容推荐服务，内容包括商品（包括电商和零售）、广告、新闻和资讯等。在移动设备和移动互联网飞速发展的时代，个性化推荐将成为用户获取信息最直接的渠道之一。

2.4.3 面向系统能效潜力挖掘的差异化技术发展成为重点

2014 年的大数据技术走向将是更高效的系统和更差异化的技术。系统能效将会是业界关注的重点。比如百度云存储万台定制 ARM 服务器就是典型案例，节电约 25%，存储密度提升 70%，每瓦特计算能力提升 34 倍（用 GPU 取代 CPU 计算），近 10 个月以来每 GB 存储成本降低 50%。差异化的技术指的是更加专用的技术，一个系统可能只针对问题的某一个方面，一个问题的解决可能会依赖若干个系统和软件。比如 Hadoop 将逐渐成为取代磁带库的成熟技术，而直接对接应用的可能会是并行数据库和内存数据库。又如并行数据库更鲜明的分化为面向事务处理的 Transaction 类数据库和面向分析的 Analysis 类数据库等。

3　天玑大数据引擎

Hadoop 作为 Google 系统的开源实现已经在互联网领域得以广泛的应用。国外企业，如雅虎、Facebook、亚马逊（Amazon）和 IBM 等和国内企业，如百度、中国移动、阿里巴巴、腾讯、网易和人人网等都在使用 Hadoop 软件。Hadoop 核心以及外围工具和服务为快速构建互联网量

级的数据处理提供了可直接使用的工具集。开源软件的众包特点和草根特性在 Hadoop 软件上得以充分体现。开源软件应用最广泛的是互联网公司，尤其是那些开始创业的小企业(start-ups)，在技术选型方面 LAMP[7] 、memcache[8] 和 Hadoop 是他们的软件构件首选。这里，成本是一方面的原因，另一方面，选用开源软件可以很容易地根据自身业务特点进行定制开发，形成企业的核心竞争力。

互联网企业在使用 Hadoop 的同时也根据自身业务需求，开发出相关的软件和工具，不断增强 Hadoop 软件功能和壮大 Hadoop 的开发队伍。比如 Facebook 公司因为其数据分析工程师只熟悉 SQL 语言而不熟悉 MapReduce 编程框架，由此催生 Hive 这样的项目。其初衷就是实现 SQL 到 MapReduce 的解释执行。Hive 现在已经演化为数据仓库的实用解决方案。这从一个侧面反映了软件开放源代码对信息技术的巨大推动作用。国内的大数据计算技术和产业发展应该从开源文化中汲取经验，重视开源软件，以开源软件为基础形成核心竞争力。天玑大数据引擎的研发就是遵循了这一原则，发挥中国科学院计算技术研究所科研能力强的优势，面向大数据计算的技术需求，解决关键问题，形成关键技术。利用开源 Hadoop 作为平台，集成整合并回馈开源社区，从而达到天玑大数据引擎软件生态环境的良性循环和良性发展。

如图 5 所示，天玑大数据引擎的特点是：针对企业计算领域的大数据生产需求，兼容传统关系数据库操作接口，支持流式计算、图计算等模式。支持 EB 级数据分布式存储及离线式非线性处理能力，PB 级数据在线式处理能力，达到每秒千万记录级流式处理能力。达到这样的目标需要攻克统一存储、查询引擎、隔离机制、自动化运维和软硬件一体等技术难点和难题，最终逐步建立起包含模型、算法、接口和开发库等在内的天玑大数据引擎软件栈和生态环境。

4 大数据关键技术研发

国内的高校和科研院所基于 Hadoop 在数据存储、资源管理、作业调度、性能分析优化、系统高

图 5 天玑大数据引擎软件栈构成

Fig. 5. Golaxy big data engine software stack

[7] 指一组通常一起使用来运行动态网站或者服务器的自由软件：Linux 操作系统 Apache（阿帕奇）网页服务器、MySQL 数据管理系统、PHP 脚本语言
[8] 一个高性能的分布式的内存对象缓存系统

可用性和安全性等方面开展了研究工作，相关研究成果多以开源形式贡献给 Hadoop 社区。近两年，我们主要在数据的存储和索引等技术上开展研究工作，形成了天玑大数据引擎的核心竞争力。

4.1　行列混合式数据存储技术

Apache Hive 是基于 Hadoop 的一个数据仓库工具，可将 SQL 语句转换成 MapReduce 命令执行。在 Hive 中，二进制关系数据采用 SequenceFile 文件格式按行序存储，即只能按照行存和行取的方式来访问数据。当要读取某一列时需要先取出所有数据，然后再从中提取出该列的数据，效率很低。

我们观察到行存储技术的优势在于写入性能高，数据一致性好；列存储技术的优势在于压缩率高，数据加载性能好。RCFile 的研究动机在于，将行存储和列存储的优点集于一身，保持高压缩率和加载性能，同时提高数据写入性能和一致性，后者恰好是列存储的短板。RCFile 结构的设计原则是“宏观行存储结构，微观列存储结构”，即先对关系表水平划分为“行组”，“行组”间遵循行序存储；然后在“行组”内采用列序存储方式。这样一来，结合 HDFS 文件系统的块式存储机制，可以保证同一行的数据存储在同一个节点，继承了行存储的优势；同时“行组”内采用了逐列压缩技术，可实现列维度的高压缩比，节省存储空间，并可实现只对查询所需要的列进行解压处理。

RCFile 的实现代码现已贡献给 Hive 开源项目，并已经应用于 Facebook 公司的 Hadoop 生产系统。经 Facebook 公司实测可节约 25% 存储空间 。与 Apache Hive 数据仓库系统之前缺省使用的行存储技术(SequenceFile)相比，RCFile 在不影响查询性能的前提下节省高达 20% 的磁盘空间；与雅虎公司开发的数据分析系统(Apache Pig)中的列组存储技术相比，RCFile 在磁盘利用率相当的情况下可以将数据加载性能提高 23% 左右。自 0.4.0 版开始，RCFile 已经集成到 Apache Hive，用以替换 SequenceFile 成为缺省的二进制数据存储结构。据了解，从 2009 年起，国际和国内使用 Apache Hive 的很多互联网公司逐步转向使用 RCFile 存储数据。RCFile 已经成为诸如 Apache Hive 的分布式离线数据分析系统中数据存储结构的事实标准。在 RCFile 的基础上，俄亥俄州立大学的张晓东教授的研究团队提出了 ORCFile，通过优化行分组的大小并加入索引功能等，进一步提高了系统性能。

4.2　列存储数据库索引技术

Apache HBase 是 Hadoop 中 BigTable 的一个开源实现，是一种适用于海量数据(TB 到 PB 级)下单个维度(仅限主键)区间查询的数据库系统。其特点是逻辑上数据按主键顺序排列，而物理上数据按主键分片存储到多个数据节点。HBase 可以按主键迅速定位数据，同时还支持主键上高吞吐量的范围查询。但是在实际应用中往往要将数据按多个不同的属性进行排序以支持多个维度的区间查询。目前 HBase 中还没能提供一种查询速度快、存储开销低的索引方法来实现以上功能。

为了满足海量数据上的多维区间查询需求，我们基于 HBase 实现了互补聚簇索引方法(CCIndex)和片内索引方法(IRIndex)查询系统。这两种索引机制适合的应用场景完全不同。其中，CCIndex 是全局索引，适合数据读写分离模式的在线多维区间查询，查询延迟小，但数据一致性维护代价和索引膨胀率高；IRIndex 则是局部索引，是一种通用型多维区间查询索引机制，能够保证索引和数据原表的一致性，索引膨胀率非常低，理论上只到 9%。CCIndex 为每个索引列构建独立的全局索引表，并在索引表中存储直接数据而非偏移数据。这样在进行多维区间查询时，就可以直接从对应的索引表中取得完整记录，实现了一级索引，利用索引表上高效的连续扫描代替原表上的随机读取，从而大幅提高多维

区间查询性能。

IRIndex 将索引构建在 Region 内部，通过与原表 HFile 文件分离的独立 IndexFile 文件实现，数据写入和更新时先写入 IndexFile 再写入 HFile，保证了索引和原表数据的强一致性，实测写入和更新吞吐率均高于 CCIndex 和其他全局或局部索引实现技术。

结合淘宝公司“数据魔方”实时数据分析系统的实际需求，作为其全属性实时计算系统的核心，CCIndex 技术经适配和优化后，已集成到生产系统中投入实际运行。CCIndex 增强了全属性实时计算系统的扩展性和性能。目前，系统处理的数据条目超过 100 亿。采用 CCIndex 技术后，在硬件规模保持不变的前提下，系统处理的数据时效范围从原来的 7 天增大到 3 个月，处理容量增大了一个数量级，系统吞吐率增大了 7 倍，对原来延迟大于 1 s 的查询请求响应时间平均降低了 57.4%。IRIndex 技术也在积极与工业界合作，以期在大规模数据分析领域获得第一手实际应用的验证数据。

4.3 基于硬件加速的流式透明压缩技术

通过对分布式文件系统之上应用的类型进行分析可以知道，这些应用使用或产生的大部分数据是文本信息，特别是离线或在线分析系统中的数据基本都是文本。而文本本身是一种高度可压缩的数据，因此通过引入一种快速的数据压缩方法，可以有效降低数据的存储开销，提高磁盘和网络读写的有效带宽，从而提高应用的吞吐量[9]。

传统的压缩方法，如 GZip，在压缩或解压缩过程中会占用大量的 CPU 资源，使系统的处理能力受到较大的影响。虽然压缩能使系统的存储开销减小，但也有可能会使系统的处理能力下降。随着硬件技术的发展，可以使用硬件设备来压缩数据，达到分流 CPU 负载和提高压缩处理效率的目的。我们的解决方法提供一种分布式文件系统上的基于硬件加速卡的流式透明压缩技术，在占用少量系统资源的情况下，完成对用户透明的压缩和解压缩过程，能够有效降低系统的存储开销和提高系统的处理能力。

首先，我们采用硬件加速卡来对内存缓冲区进行压缩或解压缩。其次，数据的压缩或解压缩对于用户是完全透明的，无论是写入或读取数据，都可以提高磁盘和网络读写的有效带宽。此外，采用分片式压缩格式，将文件分成大小为 64～128 KB 的分片(chunk)。每一个分片，在实际的压缩数据前部是该分片的头部信息，包括：原始数据大小和压缩数据大小。最后，对上层系统提供一个基于硬件加速卡的流式压缩器，用以封装原有的输入流或输出流，创建压缩或解压后的输入流或输出流。如果硬件加速卡出现故障，采用软件压缩/解压缩，形成良好的容错机制。

我们的原型采用了基于 Apache HDFS 实现基于硬件加速卡的流式透明压缩。基于硬件加速卡的流式透明压缩原型代码基础采用了 hadoop-0.20.2，用 Java 语言实现。基于硬件加速卡的流式透明压缩器位于客户端和数据节点之间。客户写入数据时，数据先经过压缩再发送到数据节点；客户读取数据时，先将从数据节点读取的数据进行解压缩后再返回给客户端。

借助硬件加速卡，压缩过程只占用少量 CPU 资源，能够卸载高达 20%～30% 的 CPU 负载，压缩处理吞吐率高于磁盘读写带宽。从目前测试效果看来，数据压缩比大约为 25%，有效降低了存储开销，同时将磁盘的有效带宽提高了 4～5 倍。压缩过程对上层应用透明，因此基于 HDFS 的在线或离线数据分析系统(HBase 和 Hive 等)都可以方便地使用。该项技术已经随天玑大数据一体机广泛应用到政府、国防、安全和公安等部门，大大提高了海量数据处理的计算效率，同时节省了存储空间。

5　天玑大数据引擎典型应用

5.1　读写分离统计分析型应用

结合淘宝的数据魔方在线系统的实际需求，作为数据魔方全属性实时计算系统的核心，天玑大数据引擎的重要组件—ICTBase 已上线投入实际运行，使淘宝网原有的业务逻辑能够直接迁移到经改进的 HBase 上，同时增强了全属性实时计算系统的扩展性和性能。目前，该实时计算系统处理的数据记录超过 108 亿。ICTBase 的索引及分布式查询技术解决了诸如 HBase 等当前主流的列簇式 NoSQL 数据库系统在多列查询上的功能缺失和性能低下的问题，通过融合各种索引技术及联合优化，可以对 NoSQL 中数据非主键列进行定位和查询，从而弥补了 NoSQL 与传统关系数据库相比查询功能及能力的欠缺。同时利用服务端计算技术，可以对海量数据进行本地化聚合计算而无需进行大量数据拷贝传输。如此经过强化的查询统计能力配合 NoSQL 的高扩展性及大吞吐量的数据处理能力，使众多关系型数据库面对的数据处理瓶颈得以克服。

5.2　低延迟流式处理型应用

一直以来，对于用户流量来源以及用户点击行为的分析都是淘宝的“量子统计”提供的服务中最为重要的组成部分。以往采用传统技术只能为用户提供按小时统计的分析数据，即用户可查询店铺内某一天 24 小时分时段的数据报表。其内容包括各时段用户浏览量、访客数及来源和店内浏览路径。而采用 ICTBase 进行数据流式存储和统计之后，店主可以实时地看到当前正在浏览客户的实时点击行为。新系统实时地收集分析了淘宝全网用户点击日志，统计内容包括淘宝 300 万店铺的实时 UV 和 PV 值，并能绘制出淘宝网日均 1.2 亿用户的实时点击行为图示，最后将这些信息分类推送给相关店主。整个系统的数据处理延时仅为 2 至 3 秒。实际日志处理量为 3 万至 5 万记录每秒，每天 20 亿记录，数据写入操作为 15 至 25 万次每秒，单日原始数据量为 600 GB，存储一周用户数据则原始数据量为 4 TB 左右。

5.3　高并发访问型应用

腾讯网是目前中国最大的互联网综合服务提供商，也是中国服务用户最多的互联网企业之一。截至 2013 年 8 月 14 日，QQ 即时通信的活跃帐户数达到 8.185 亿，最高同时在线帐户数达到 1.732 亿。其数据平台一直致力于发掘用户数据的价值，为用户提供更为精准的个性化服务。广点通即数据平台核心产品之一，旨在根据用户访问数据提高平台广告推送效率。

面向海量用户访问数据的实时存储查询系统是广点通智能推荐系统的基础。全内存分布式的 ICTBase 优化了线上系统查询性能，提高了存储层数据访问效率，大幅减轻集群内部网络压力，提高了广点通整体性能，并成功支持了对存储性能要求更高的复杂用户推荐算法。新系统经过相应优化之后，实测单机查询性能提升 20 倍，占用服务器数量缩减为原系统的 1/5，日均处理日志数量 30 亿记录，处理用户请求数量达 25 亿次。

6　结　语

Hadoop 是大数据计算领域的一项具体技术，一套软件系统和工具。因其开源而对推动大数据计算技术发展起到了重要作用。面向不同的应用需求，基于 Hadoop 的数据处理工具也应运而生。天玑大数据引擎集成了 Hadoop 生态环境中成熟且社区活跃的组件，如 Hive 和 HBase 等，并整合了天玑团队的众多研究成果，如 RCFile、CCIndex/IRIndex、SwiftFS 等，可以满足舆情分析、社会计算、商业智能和数据挖掘等大数据处理的实际需求。可以预见，大数据计算的出现将催生更多、更好、更面向大众的

新应用，同时更能够加快大数据计算技术发展的步伐。

参 考 文 献

[1] Dean J, Ghemawat S. MapReduce: simplified data processing on large clusters [J]. Communications of the ACM, 2008, 51(1): 107-113.

[2] Ghemawat S, Gobioff H, Leung ST. The Google file system [C] // Proceedings of the 19th ACM Symposium on Operating Systems Principles, ACM, 2003: 29-43.

[3] Chang F, Dean J, Ghemawat S, et al. Bigtable: a distributed storage system for structured data [C] // Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation, 2006, 7: 205-218.

[4] The Apache Software Foundation. Apache Hadoop [EB/OL]. http://hadoop.apache.org/.

[5] The Apache Software Foundation. Apache Hive [EB/OL]. http://hive.apache.org/.

[6] The Apache Software Foundation. Apache HBase. [EB/OL] http://hbase.apache.org/.

[7] He YQ, Lee RB, Huai Y, et al. RCFile: a fast and space-efficient data placement structure in MapReduce based warehouse systems [C] // Proceedings of International Conference on Data Engineering, 2011: 1199-1208.

[8] Zou YQ, Liu J, Wang SC, et al. CCIndex a complemental clustering index on distributed ordered tables for multi-dimensional range queries [C] // Proceedings of the 2010 IFIP International Conference on Network and Parallel Computing, 2010: 247-261.

[9] Nicolae B, Moise D, Antoniu G, et al. BlobSeer: bringing high throughput under heavy concurrency to Hadoop Map-Reduce applications [C] // 2010 IEEE International Symposium on Parallel and Distributed Processing, 2010: 1-11.

[10] Huai Y, Ma SY, Lee RB, et al. Understanding insights into the basic structure and essential issues of table placement methods in clusters [C] // Proceedings of 39th International Conference on Very Large Data Bases, 2013.

[11] 程学旗. 关于中国大数据生态系统的基础问题思考 [EB/OL]. http://www.china-cloud.com/yunjishu/shujuzhongxin/20140213_23094.html.