大数据与 OLAP系统

杜小勇，陈跃国，覃雄派
中国人民大学信息学院数据工程与知识工程教育部重点实验室 北京 100872

摘要

OLAP（online analytical processing，在线联机分析处理）是关系数据基础上实现商业智能的核心技术。在大数据时代，人们迫切希望在由普通机器组成的大规模集群上能实现高性能的OLAP，然而系统性能的挑战巨大。可喜的是，近年来进展迅速，涌现了很多以Hadoop上的数据进行OLAP的所谓SQL on Hadoop系统，并不断提升。在综述OLAP技术发展的基础上，重点对几个有代表性的SQL on Hadoop系统进行了测试分析，并展示了这类系统的性能特点。可以预见，未来在低成本的大数据OLAP市场，这类系统会占有重要位置。

关键词

大数据；OLAP；SQL分析；SQL on Hadoop

Big Data and OLAP Systems

Du Xiaoyong, Chen Yueguo, Qin Xiongpai
Key Laboratory of Data Engineering and Knowledge Engineering,
School of Information, Renmin University of China, Beijing 100872, China

Abstract

OLAP (online analytical processing) is a key technology of business intelligence based on relational data. In big data era, people want to achieve high performance OLAP using a large cluster of ordinary nodes. However, the performance of such systems is a big challenge. Recently, many SQL on Hadoop systems have been proposed to address this challenge. We have seen a significant performance improvement of such systems. A survey of technology development of OLAP technologies was first provided. Then, a study of the performance of three representatives SQL on Hadoop systems was focused on. Based on the results, it is expected that such systems will play an very important role in the market of low cost OLAP analysis.

Key words

big data, online analytical processing, SQL analysis, SQL on Hadoop

1 OLAP 的前世今生

数据分析一般指为了从数据中获得有价值的信息，而采用的诸如数据清理、建模、查询、统计、挖掘、展示等操作过程，其产生的结果往往用于决策支持，是传统商业智能所依赖的重要技术。数据分析的类型多种多样，根据被分析的数据类型和所采用的核心分析方法，常见的数据分析有流式分析、SQL分析、深度分析等，复杂度依次递增。流式分析主要是对数据流（动态连续产生的数据序列）在一定时间窗口内进行简单的统计分析或者实时监控，处理速度快，面向快速洞察新数据的实时应用；SQL分析是以SQL语句形式对关系模型下的数据进行的查询分析，习惯被称作OLAP（online analytical processing，在线联机分析处理），通常是在关系查询基础上进行的一些统计分析操作；深度分析则采用了复杂度较高的数据挖掘和机器学习中的方法，可以处理结构化或非结构化数据，通常采用多轮迭代的计算模型，计算代价更高。除此之外，还有更为复杂的分析，光靠机器很难得到令人满意的数据分析结果，往往需要人的参与，形成了人机结合的数据分析方法，如众包技术。

数据分析为决策服务，其性能（一个分析任务的平均执行时间）对很多应用来说非常重要。一般来讲，5 s以内（有些应用可以增加到10 s左右）执行完的分析常被称作实时分析，一两分钟以内执行完的分析任务则可以称作交互性分析，半小时以上完成的分析常被称作离线分析，而介于交互性分析和批处理之间的分析任务，可以称作非交互性数据分析。流式分析一般要求具备实时分析的性能，OLAP则希望具备交互性分析的性能，深度分析在数据量大、任务复杂的情况下往往是离线分析任务。

本文重点围绕OLAP，探讨大数据对于OLAP系统的深刻影响。截至目前，它还是商业数据分析（数据仓库技术）所依赖的最为重要也最为核心的数据分析技术。

使用SQL技术对数据进行分析在数据库发展的早期阶段就被人们研究，但OLAP这个词最早是关系数据库之父Edgar F Codd在1993年给出的[1]，作为与面向联机事务处理负载所不同的另一类重要的数据库负载。这个概念主要针对多维数据分析（MDA）领域，是商业智能的核心部分。20世纪90年代，人们在以OLAP为主的数据仓库技术方面做了很多研究，为了提升OLAP的处理性能，最为常见的方法就是基于多维数据构建Cube模型，通过大量的预聚集计算，实现生成支持多维分析的Cube，并在此基础上支持以下钻、上卷、切片、切块、旋转等操作为代表的高性能的OLAP[2]。这种OLAP处理方法也被称作MOLAP，是最为典型的OLAP处理方法。其优点就是通过数据的深加工（预聚集计算、索引、压缩、缓存等），换取高的OLAP性能；缺点是数据预处理环节多，处理周期长，数据维度不宜过高，数据冗余多，适合处理几十GB规模的数据。

OLAP的另一类技术是ROLAP，不做Cube计算，直接使用关系数据库，采用基于事实表和维表的星型模型或者更为复杂的雪花型模型，组织数据。在ROLAP过程中，能够将Cube上的各种操作转换为数据库上SQL查询分析语句执行。ROLAP的优缺点与MOLAP的优缺点几乎是相反的，性能上没有MOLAP快，但因为没有数据模型转换和深加工过程，数据导入快，而且其扩展性更好（支持分库、分表等数据库集群技术），查询也不受Cube模型的限制，广泛支持各类数据库上的SQL查询。

当然，也有人将 MOLAP 与 ROLAP 融合到一起，形成一种兼具数据库和 Cube 模型的 HOLAP 解决方案，以更好地发挥二者的优点，避免二者的缺陷。

进入新世纪后，ROLAP 技术随着 MPP 并行数据库技术的发展，尤其是在 Stonebraker [1] 等数据库专家所倡导的列存储技术的支持下 [3]，面向 OLAP 应用的 MPP 数据库集群技术得到了迅速发展，出现了以 HP Vertica 为代表的列存储并行数据库。列存储数据库技术针对数据分析的特点，能够对数据进行高性能的压缩，查询也只需访问必要的列，节省了很多 I/O，分析性能比传统行存储数据库有了很大的提升（可以多达两个数据量级）。正如 Stonebraker 所说，列存储数据库已经开始成为 OLAP 的主角，并会在未来一段时间长期保持作为 OLAP 的主流技术。同时，随着内存成本的降低、单机内存的增大，以 SAP HANA 为代表的内存数据库 [4] 也采用了列存储技术，支持更高性能的数据分析。这些技术的发展，使得它们成为 TB 级别数据仓库进行 OLAP 数据分析的最先进技术，已经涵盖了绝大多数 OLAP 市场。

然而，市场的发展并不总以技术论成败，系统的建造成本和开放性也是重要因素。在建造成本方面，MPP 并行数据库和内存数据库依赖昂贵的硬件配置，其中的很多商业软件还有价格高昂的使用许可证，这些成本并不是每个公司都能够承担或者愿意承担的；而开源大数据系统采用通用、廉价的硬件设施，使得人们更容易尝试和使用这些系统，数据和业务迁移的成本也更低。在开放性方面，以 Hadoop 为代表的开源大数据系统形成较大的社区之后，就会有各种相关系统补充进来，构成生态圈，满足人们不同的需求；而商业大数据系统则相对封闭，依赖少数的厂商发掘用户需求，再加上历史系统的遗留问题，系统的迭代发展效率不高。此外，以 Hadoop 为代表的大数据系统并不一定是免费和廉价的代名词，参与这些系统研发的个人或者组织也可以从中获得巨大的利益。在这样的驱动力下，很多用户也希望参与其中，积极地贡献和完善相关功能，也给了高校和科研院所更多参与大数据研究的机会。这些原因使得开源大数据系统逐渐得到认可，并成为大数据分析的新宠儿。

2 SQL on Hadoop 系统

互联网公司最先遇到大数据难题，需要为海量互联网网页构建倒排列表。2004 年，Google 公司提出 MapReduce 技术 [5]，作为面向大数据分析和处理的并行计算模型，引起了工业界和学术界的广泛关注，并很快形成了 MapReduce 的开源实现 Hadoop 计算和存储的框架。MapReduce 在设计之初，致力于通过大规模廉价服务器集群实现大数据的并行处理，把扩展性和系统可用性放在了优先考虑的位置。MapReduce 技术框架包含 3 个层面的内容：分布式文件系统（GFS 或者 HDFS）、并行编程模型、并行执行引擎，具体介绍如下。

分布式文件系统运行于大规模集群之上，集群使用廉价的服务器构建。数据采用键值对模式进行组织和分发。整个文件系统采用元数据集中管理、数据块分散存储的模式，通过数据的复制（每份数据至少有 3 个备份）实现高度容错。数据采用大块存储（64~256 MB 为 1 块）的办法，可方便地对数据进行压缩，实现块粒度的数据复制与查询处理的容错。

MapReduce 并行编程模型把计算过程分解为两个主要阶段，即 map 阶段和

1 Michael Stonebraker 由于对现代数据库系统底层的概念与实践所做出的基础性贡献，获得了 2014 年度 ACM 图灵奖

reduce 阶段。 map阶段 多个任务在多节点上并发读取和处理数据，中间结果根据键值对的形式写入本地磁盘，并经过shuffle 过程发到执行 reduce 任务的节点，再由 reduce节点归并 来自不同 节点的 数据，将结果输出或 写入 HDFS，完成一轮MapReduce任务。 复杂的任务可以转换为多轮 MapReduce任务执行。

MapReduce技 术是一种简洁的并行计算 模型， 在系统 层面解决了扩展性、 容错性等问题， 通过接收用户编写的 map 函数和 reduce 函数， 自动在可伸缩的大规模集 群上并行执行， 从而可以处理和分析大规模的数据。 如今随着 Hadoop技术的不断成熟（如 YARN等新一代 Hadoop技术） ，MapReduce任务可以运行在成千上万个节点组成的集群上， 处理 PB级别的数据。

Hadoop技 术很快也影响了数 据库研究领域， 有面向简单的键值对读写事务型负载的 NoSQL系统 （如 HBase等） ，也有面向数据分析任务的 Hive 系统。 Hive 系统的出现， 一改传统的 OLAP只能在关 系数据仓库中运行的局面， 从而 可以对 HDFS中存储的结构化数据， 基于一种类似 SQL的HiveQL语言， 进行 ROLAP方式的数据分析。Hive 系统将用 HiveQL描述的查询语句，转换成 MapReduce任务来执行 ，并且具备了一定的查询优化能力， 这样就可以在大规模集群环 境下对 TB 级别甚至 PB 级别的大数据进行 OLAP。 这显然对传统并行数据库和数据仓库技术构成了挑战， 也很快得 到了数 据分析 领域一些 著名的学 者（如 Dewitt 和Stonebraker 等） 的回应，这就有了 2009年SIGMOD会议上发表的在大数据分析领域 MPP数据库与 Hadoop技术的对比 [6]。其结论是： 结构化大数据分析（OLAP ）方面 MPP数据库的性能要远好于以Hive 为代表的 Hadoop上的数据分析技术，而Hadoop技术也有其优势， 比如高度的扩展能力和容错性能、 对非结构化数据的支持、 用户自定义函数的使用等方面。

然而， 来自互联网领域或者其他领域很多大 数据创新公司， 并没有止 步于Hive 。最近五六年间做出了很多努力， 开发了多个 SQL on Hadoop系统， 以提升这些系统的性能。 这些系统借鉴了 20世纪 90年代以来在并行数据库方面所积累的一些先进技术， 大幅度提升了 SQL on Hadoop系统的性能， 所以目前这类 SQL on Hadoop系统是非常有吸引力的。

接下来， 举出其中几个典 型的系统 加以介绍。

2.1 Hive 系统

自从 Facebook 在2007年推出 Apache Hive系 统及其Hive QL语 言以来，已经成为 Hadoop平台标准的 SQL实现。 Hive把HiveQL查 询首先 转换成MapReduce作业，然后在Hadoop集群上执行。某些操作（如连接操作）被翻译成若干个MapReduce作业， 依次执行。 早期版本的Hive ，性能与 Impala 、 Presto 等系统有很大差距 [7]。近年来， 开源社区对 Hive 进行持续改进， 主要包括以下几个方面。

- 在SQL接口方面， 增加了新的数据类型、子查询支持、 更加完备的 Join 语法等。
- 在文本类型、RCFile 列存 储格 式之外， 增加了具有更高效率的列存储格式ORCFile。
- 和Tez紧密集成， 以便执行更通用的任 务，获得更 高的性 能。Apache Tez是一种新的计算 模型， 扩展了 Hadoop的MapReduce计算模型， 能够执行复杂的以DAG （directed acyclic graph ， 有向无环图）表达的计算任务。 Tez的DAG顶点管理模块， 在运行时从任务收集相关 信息， 从而动态改变数据流图的一些参数， 以便优

化资源消耗，获得更高的性能。

- 增加初步的查询优化能力，能根据数据特点，包括表格的基数、各个数据列的统计信息（最大值、最小值、平均值、不同值的个数）等，进行表连接顺序调整和连接算法选择。目前 Hive 仅支持等值连接，可选的算法包括 Multi Way Join、Common Join、Map Join、Bucket Map Join、SMB Join、Skew Join 等，连接算法的详细信息，可以参考参考文献 [8]。
- 新的向量化的查询执行引擎，通过更好地利用现代 CPU 的特点，提高查询性能。

2.2 Impala 系统

Impala是由Cloudera公司推出的一个支持交互式（实时）查询的 SQL on Hadoop系统。Impala 放弃使用效率不高的MapReduce计算模型，设计专有的查询处理框架，把执行计划分解以后，分配给相关节点运行，而不是把执行计划转换为一系列的 MapReduce作业。Impala 不把中间结果持久化到硬盘上，而是使用 MPP 数据库惯用的技术，即基于内存的数据传输，在各个操作之间传输数据。Impala 后台进程以服务的形式启动，避免了类似于 MapReduce 任务的启动时间。查询执行引擎针对新硬件做了相关优化，比如充分利用新的指令集（包括单指令多数据指令 SIMD）进行数据处理。同时 Impala 使用 LLVM技术，把查询编译成汇编指令，以加快其执行，无需不断进行 SQL查询的语法检查和翻译。

在磁盘 I/O 方面，Impala 维护每个数据块的磁盘位置信息，对磁盘块的操作顺序进行优化调度，保持各个磁盘忙闲均衡。此外，Impala 在实现细节上，进行了一系列优化。在存储格式方面，Impala 支持最新研发的列存储格式 Parquet，有利于提高数据仓库查询性能，这些查询一般只涉及少数属性列，列存储可以避免不必要的数据列的提取。

在连接操作的处理方面，Impala 根据表的绝对和相对大小，在不同的连接算法之间进行选择。广播连接是默认的方式，右侧的表默认比左侧的表小，小表内容被发送到查询涉及的各个节点上。另外一种连接算法称为分区连接，适用于大小相近的大型表之间的连接。使用分区连接，每个表的内容被散列分布到各个节点，各个节点并行地进行本地连接，连接结果再进行合并。Impala 使用 Compute Stats 语句，收集数据库表的统计信息，辅助进行连接算法的选择。根据 Cloudera 的评测结果，对于 I/O 限制的查询，相对于老版本的Hive，Impala 有3~4倍的性能提升。而对于需要多个 MapReduce 作业或者需要 reduce 阶段实现连接操作的查询，Impala 可以获得更大的性能提升。对于至少有一个连接操作的查询，性能提升达到 7~45倍。如果数据集可以完整地保存到缓存中，则性能提升达到 20~90倍之巨，包括简单的单表聚集查询。Impala 令人印象深刻的性能使人们相信，只要充分利用各种优化措施，包括存储优化、执行引擎优化、查询优化等技术，Hadoop平台上的 SQL查询也能达到交互式的性能要求。

2.3 Spark SQL

Spark SQL是美国加州大学伯克利分校提出的大数据处理框架BDAS（Berkeley data analytics stack）[10]的一个重要组成部分，包括资源管理层、存储层、核心处理引擎、存取接口、应用层等层次和部件。Spark SQL是实现大数据交互式SQL查询的处理系统，包括接口 Spark SQL和处理引擎 Spark Core。Spark 是一

个分布式容错内存集群，通过基于血统关系的数据集重建技术，实现内存计算的容错。当一个内存数据集损坏时，可以从上游数据集通过一系列的操作重建该数据集。Spark SQL使用内存列存储技术支持分析型应用。在复杂查询执行过程中，中间结果通过内存进行传输，无需持久化到硬盘上，极大地提高了查询的执行性能。Spark SQL在设计上实现了和Apache Hive在存储结构、序列化和反序列化方法、数据类型、元信息管理等方面的兼容。此外，BDAS还支持流数据处理和图数据的计算，并通过迭代计算支持各种机器学习算法；甚至可以运行在YARN上，和Hive使用的是同一个资源管理模型。

MapReduce系统性能不佳的原因有很多，包括中间结果持久化到硬盘、数据存储格式性能低劣、不能控制数据的并置（co-partitioning）、执行策略缺乏基于统计数据的优化、任务启动和调度的开销过大等，Spark SQL设计者据此实现了一系列优化措施，包括：采用基于内存的列存储结构；支持基于散列的shuffle和基于排序的shuffle操作；基于range统计信息进行分区裁剪，减少查询处理过程中需要扫描的数据量；下推查询限制条件；支持分布式排序；支持分布式并行装载；集成机器学习功能，方便在SQL语句中执行更加复杂的分析等。Spark SQL部分实现了基于成本的优化功能，根据表格和各列数据的统计信息，估算工作流上各个阶段数据集的基数，进而可以对多表连接查询的连接顺序进行调整。另外如果连接的中间结果或者最终结果集具有较高基数，系统可以根据启发式规则，调整reduce任务的数量，完成Join操作。此外，新版本的Spark SQL还计划支持数据并置以及部分DAG执行技术，允许系统根据运行时搜集的统计信息，动态改变执行计划，以获得更高的性能。

2.4 其他典型系统

由于大数据上交互式SQL查询应用方面的强劲需求以及现有系统在性能方面的不足，近几年涌现了一批新的大数据SQL查询分析系统。Ebay开源了其Hadoop上的MOLAP系统Kylin[2]。Kylin的工作原理是：从Hive获取数据，使用MapReduce预处理这些数据，生成数据立方体，保存到HBase中，以支持100亿行数据规模的低时延查询，部分查询可以达到亚秒级的响应时间。Kylin提供REST、SQL和OLAP接口，支持TB级别到PB级别的数据量；兼容ANSI SQL标准以及Tableau、Microstrategy和Excel等前端工具；支持数据的编码和压缩，支持Cube的增量更新功能，支持不同值个数的近似查询能力。

Tajo[3]是韩国初创公司Gruter研发的大数据分析软件，目前已经作为Apache一个孵化阶段的开源项目。Tajo本质上是Hadoop平台上基于关系模型的分布式数据仓库系统，其设计目标是：支持存储于HDFS的大数据集上的低时延的即席查询、在线聚集查询以及ETL操作等。Tajo使用了基于成本的查询优化技术以及渐进式查询优化技术（即在查询执行过程中，根据搜集的统计数据，动态改变执行计划）；可以支持内存不足情况下的查询执行；对超长时间运行的查询，提供容错保证和动态调度能力，这种针对性的优化是Hive、Impala系统所欠缺的。为了支持Tajo走向实际应用，其开发团队在Tajo中集成了不同文件格式的支持，包括CSV、JSON、RCFile、Sequence File以及列存储格式Parquet等；查询语言兼容ANSI SQL标准，通过JDBC接口，可以方便存取Tajo系统；可以直接存取Hive的元信息管理服务MetaStore。在最新的Tajo评测中

2 http://www.kylin.io/

3 http://tajo.apache.org/

（使用 TPCH测试基准），大部分查询取得和 Impala 相当的性能，某些查询则优于 Impala，比如 Q3以及 Q5等查询。

星云科技是国内少数掌握Hadoop和 Spark 核心技术的高科技初创公司，其大数据平台 Transwarp Data Hub (TDH) [4]，基于 Hadoop和 Spark 技术，实现了流数据处理、大数据交互式查询、批处理和机器学习。其中，Transwarp Inceptor 是基于内存处理的交互式 SQL查询分析引擎（基于Spark），同时支持数据挖掘功能（通过R软件包）。星云公司在 Hadoop和 Spark 上进行了大量的二次开发，借鉴了 MPP数据库的查询优化技术，以提供更高的性能；在SQL接口方面，不但支持 HiveQL，还支持 Oracle PL/SQL存储过程语言，支持PL/SQL是Transwarp Inceptor 的一大特色；支持R软件包，实现数据挖掘功能，并且提供了若干并行化的算法；支持 Tableau、SAP BQ Oracle OBIEE等前端报表工具。

Actian公司收购了列存储数据库MonetDB的商业化版本 Vector 后，对其进行移植，以便可以原生地运行在 Hadoop 平台上。SQL分析是 Actian Hadoop大数据分析平台的一部分，整个项目称为 Vortex。Actian 招募了荷兰著名研究机构CWI的数据库专家 Peter Boncz，同时也是 Vector 的架构师（Vector 是Peter Boncz 领导研发的 MonetDB开源列存储数据库的商业化版本）。在Boncz的带领下，研发团队把 Vector X100 SQL（商业化版本称为Vectorvise）查询引擎进行并行化，运行在 Hadoop集群上。Vortex 除了完全兼容ANSI SQL 标准之外，还支持ACID 特性，即支持数据的更新。Vector 支持细粒度的更新，这些更新首先被保存到独立的数据结构里，称为 PDT（positional delta tree），然后和主数据结构进行合并。凭借MonetDB团队在列存储、数据压缩、查询优化、向量化查询执行、CPU cache存取优化等方面十几年的研究和开发积累，Vector 获得了极高的性能，根据Actian内部的评测，Vector目前是同类系统在TPC-H评测基准上最高性能纪录的保持者[5]，其查询处理性能是 Impala 的16~32倍。

Flink[6]和 Spark 都是通用的大数据处理系统，并且都是 Apache软件基金会的顶级开源项目。在处理 SQL查询方面，目前 Spark 中有 Spark SQL，Flink 中有 MRQL。在图计算、流数据处理和机器学习方面，两者也都有对应的组件予以支持。Spark 与Flink 都可以部署在 HDFS 之上，通过内存方面的优化来提高计算性能。但Spark 和Flink 也存在一些差别：Spark 通过基于内存的 RDD将大批量的数据缓存在内存中进行处理，这种方式在做批处理式的计算时比较适合，流计算和实时数据处理是通过 mini batch 的方式实现的，本质上并不是非常适合实时计算；而Flink 针对循环和迭代计算的优化更多，可支持的数据处理粒度更细，流数据处理方式类似 Storm 的元素粒度的流水线，而不是Spark 中的 mini batch。

3 大数据分析系统技术特点

为了更好地了解现有的 SQL on Hadoop 系统技术发展的状况和各系统的技术特点，选用了 3个典型的 SQL on Hadoop 系统：Hive-Tez、Impala、Spark SQL，在TPC-H 基准基础上进行性能对比和分析，TPC-H 基准由 22个OLAP查询组成。

3.1 实验设定

虚拟机和物理机使用的硬件环境见表 1。

4 http://transwarp.io/

5 http://www.datanami.com/2014/06/03/actian-aims-engulf-impala-vortex/

6 http://flink.apache.org

表 1 测试硬件环境

部件	物理机	虚拟机
CPU	Intel Xeon E5-2670 ，2.60 GHz， 12核24线程	基于 Intel Xeon E5645 ，通过 KVM虚拟出 2.4 GHz， 4核4线程
内存	24 GB	24 GB （物理内存充足）
硬盘	单块 Seagate SAS 900 GB, 10 000 RPM	基于 6块Seagate SAS 2 TB, 7200RPM组成的 RAID5磁盘阵列的 500 GB虚拟磁盘
网络	吉比特以太网	吉比特以太网

操作系统采用的是 CentOS 6.4 版本。Hive-Tez 使用的是 Hortonworks Hive 0.14 版本，用ORCFile存储格式； Impala 的版本为 Impala 2.1.3 ,使用 Parquet 存储格式；Spark SQL使用的是 Spark 1.2 , 采用 Parquet 存 储格式。 各个系统在测试之前进行了大量参数调优的工作, 以确保查询在优化的参数配置下执行。

3.2 性能对比测试

使用 TPC-H基准生成了 300 GB规模的数据， 在16个节点组成的虚拟机环境下 ，对Hive 、 Impala 和Spark 3个系统的性能进行对比。 各查询的执行时间见表 2。 相应的，Impala 和Spark SQL相对于 Hive-Tez 的加速比在图 1中给出。

综合表 2和图 1的结果，可以看出，Impala 仅在 Q1、 Q6、 Q12、 Q15上的性能明显优于 Hive， 这几个查询都是比较简单的查 询，有两个是直接在事实表 lineitem 上的单表统计 查询，而另 外两个是包含 lineitem 的两表 连接， 都可以使用广播连接的方式将小表数据复制到各个节点上与事实表 lineitem 进行连接操作。 这个测试结果一方面说明 Hive 在过去 数年中在性能上的提升非常显著， Impala 最初在性能上表现出的优势已经不明显； 另一方面也说明 Impala 在一些简单的查询上有好的优化效果， 对于一些复杂的查询优化效果不好，甚至无法成功执行。

表 2 16 节点 -300 GB-OpenStack 虚拟机环境下 3个系统的查询执行时间

查询	执行时间 /s		
	Hive-Tez	Impala	Spark SQL
Q1	168.8	30.2	115.9
Q2	99.4	140.3	110.5
Q3	354.1	time out	215.0
Q4	227.1	time out	121.0
Q5	time out	time out	time out
Q6	54.0	17.0	47.3
Q7	527.0	time out	669.0
Q8	784.1	3724.1	821.6
Q9	725.0	time out	842.1
Q10	299.1	642.2	195.4
Q11	87.8	failed	time out
Q12	156.5	56.3	193.1
Q13	170.4	210.8	127.3
Q14	81.4	59.0	75.1
Q15	98.9	18.3	101.1
Q16	136.8	260.2	82.1
Q17	712.1	421.2	842.8
Q18	582.0	time out	654.1
Q19	114.8	191.0	77.4
Q20	240.1	290.2	185.2
Q21	1 235.3	1 106.9	1 216.1
Q22	123.8	failed	113.4

对于 Spark SQL而言， 其查询性能与 Hive 在很多查询上相差无几， 各有千秋。Spark SQL仅在 Q3、 Q4、 Q10、 Q16上的表现好于Hive （1.5 ~1.9 倍的 加速比） 。这几个查询复杂度不一， 从两表 连接到 lineitem 基础上的 4表连接链式查询都有。Spark SQL在个别查询上多连接执行次序更优化一些， 也是其 在个别查 询上性能明显好于 Hive 的一个重要因素。

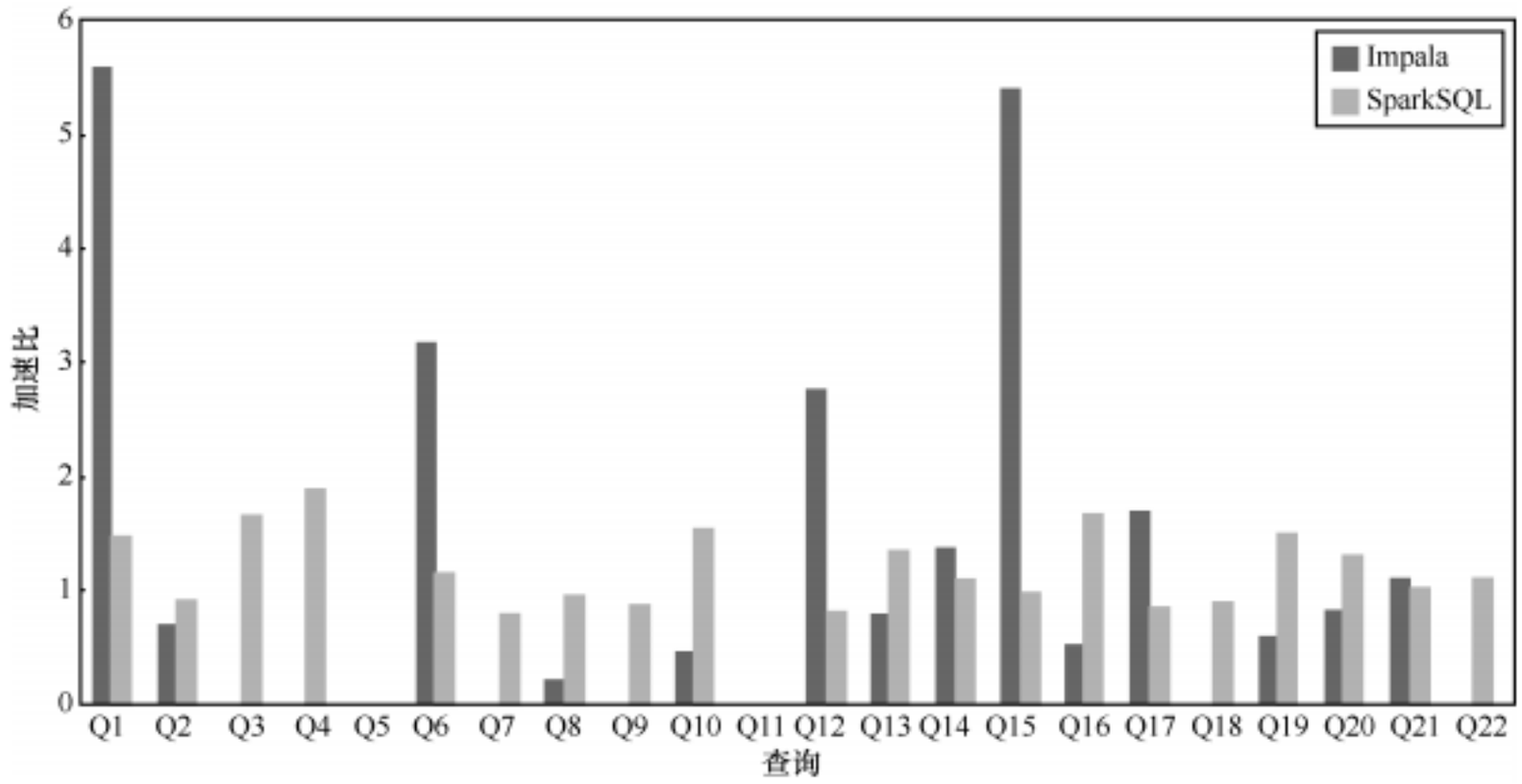

图1 16 节点 -300 GB-OpenStack 虚拟机环境,相对 Hive-Tez 的性能提升率

从表 2的数据能够看出, Impala 在很多查询上出错或超时, 暴露出其不稳定性, 尤其是在相对复杂的查询中。 具体分析发现, Impala 在一些查询上, 经常会因为内存不足而出现错误或者长时间跑不出结果。 Impala 深度借鉴 MPP数据库技术, 依赖大内存提升连接执行效率。 因为抛弃了 MapReduce, 系统不再具备查询间容错, 一个查询如果在一个节点上出现问题 (内存不足或其他节点故障等) , 整个查 询就不能够被成功完成。相比而言, Hive 和 Spark SQL 由于还保留有 MapReduce的中间结果 shuffle 机制, 在容错方面比 Impala 好很多。

从3个系统整体上的性能表现看, 相比于一年多以前的测试结果 [8], 以Hadoop为基础的 Hive 和 Spark SQL系统在性能上很多方面已经追赶上了以 MPP并行数据库技术为基础的 Impala 系统。 这一方面得益于 Hadoop生态圈的开放性和进取心, 同时也充分体 现了随着更多传 统MPP并行数 据库技术的引入, SQL on Hadoop系统的分析性能越 来越强大。 当然也要意识到, 因为Hadoop存储的透明性, 数据缺少好的划分和全局索引机制, 势必也会限制 SQL on Hadoop系统所能够达到的性能高度。 进一步的技术突破, 除了在查询优化方面深度借鉴 MPP数据库技术外, 在数据存储与划分方面 可能还需迈出坚实的一步, 需要 牺牲一些 Hadoop数据存储的透明性。

总体来看, SQL on Hadoop系统由于其开放性和在容错方面的优势, 性能提升明显, 在Hadoop上的很多 OLAP任务已经具备交互性能。

3.3 可扩展性能力测试

为了更好地了解各个系统的查询性能随着系统节点数和系统处理数据量的变化情况, 分 别在 这 3 个系统 上, 通 过 调整 节点 数 量 和系统 数 据量 进行可扩展性方面的测量。 在 Hive-Tez 系统上, 节点数量从8个增加到 16个和 32个的实验 结果如图 2所示, 而在该系统上数 据量 从100 GB变到300 GB的实验结果如图 3所示。 最后, 把3个系统在所有查 询上的平均性能变化列在表 3与表 4中。

通过表 3的数据能够看到, 随着节点数量的翻倍, 系统的查询性能提升很难达到翻倍的理想线性加速比。 相对而言, Spark SQL随着节点数量的增加, 查询提升效果

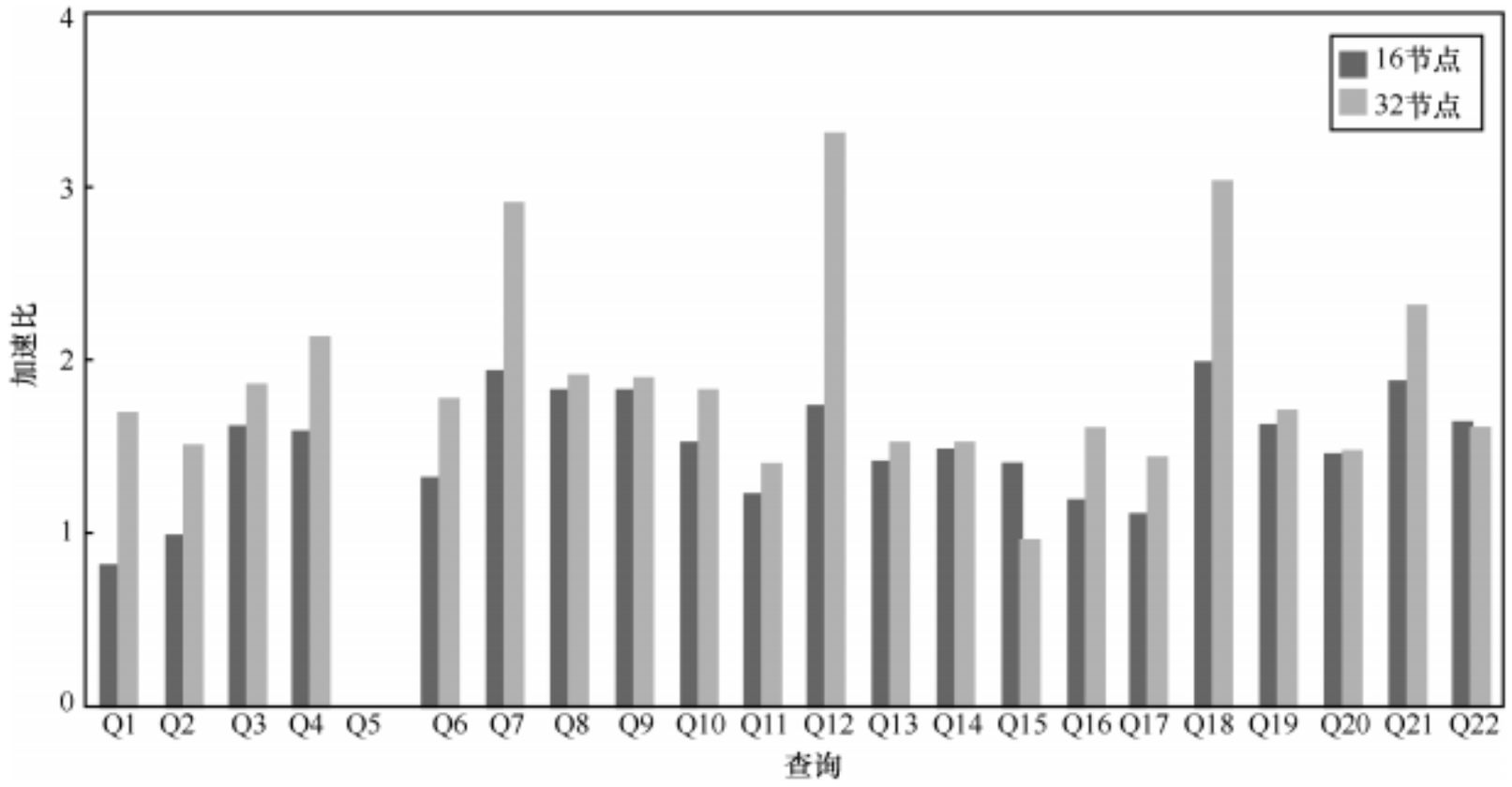

图2 Hive-Tez 系统 32节点与 16节点相对 8节点集群的加速比

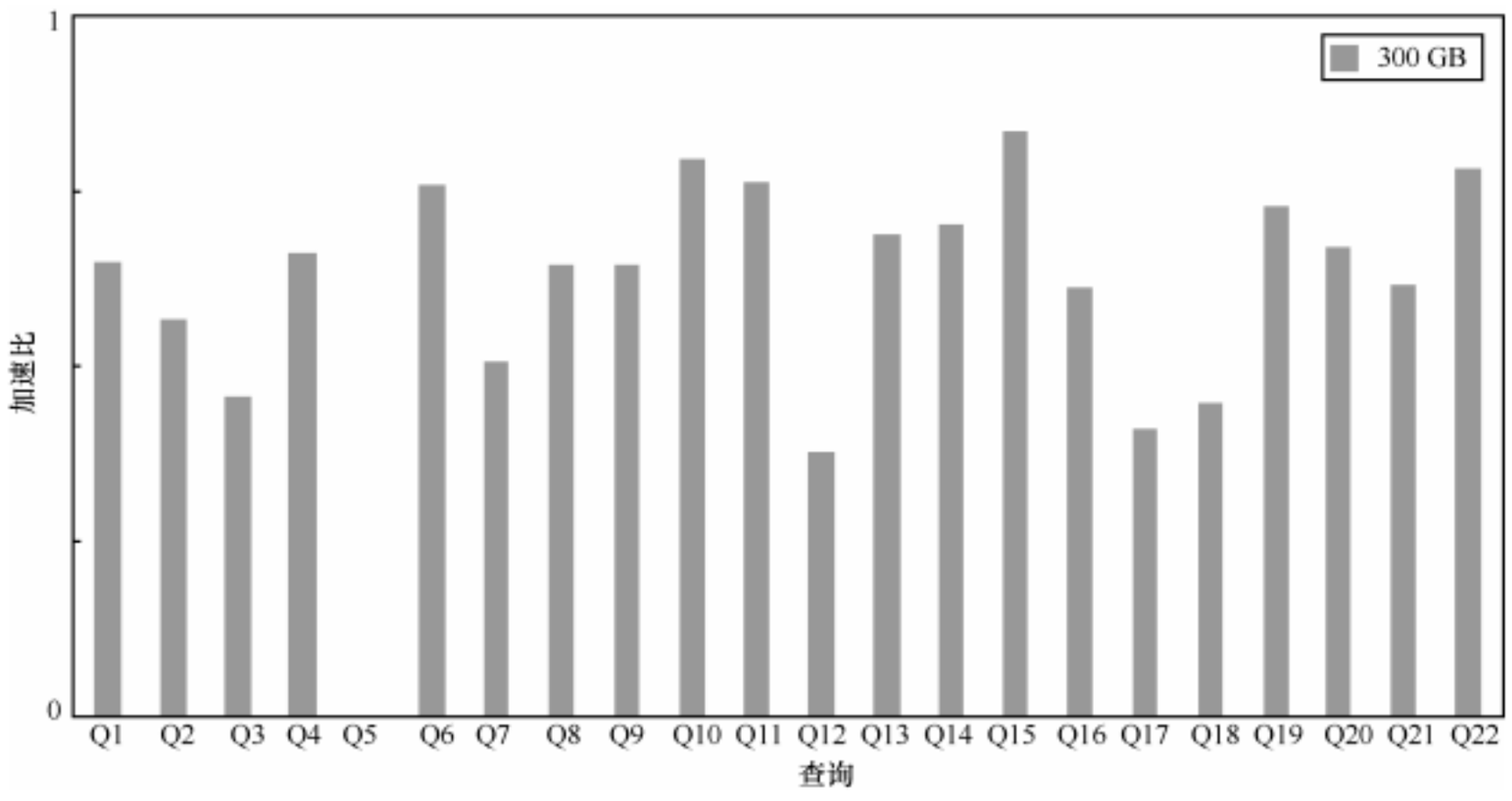

图3 Hive-Tez 系统 300 GB相对 100 GB的加速比

更为明显， 而Impala 则随着系统节点数量的增加， 性能提升比例最为微弱。 这也源于Impala 是一个偏 MPP数 据库型的解决方案， 系统的扩展性不好。 而Hive 和Spark SQL是从 Hadoop扩展而来的大数据 SQL 分析 解决方案， 其扩展性更有优势， 能够更好地发挥出大规模集群计算的优势。

从 表4的结果 来 看，各系统的查 询性能都随着数量的增加而降低， 但性能下降的比例并不如数量增加的比例高。 这是一个好现象， 说明系统在处理更大规模的数据量时， 性能牺牲的幅度没有想象的大。相比而言， Hive 随着数据量的增加， 系统性能下降的比例最低。 这也是有原因的，Impala 和Spark SQL对内存依赖大， 数据量增加会直接影响内存的使用， 容易造成这两个系统的性能瓶颈； 而Hive 对内存的依赖不强， 因此表现好于另外两个系统。

通过可扩展性实验， 发现现有 SQL on Hadoop系统的稳定性还存在很大的问题，当数据量变化或者节点数量增大时， 出现了更多的查询无法被正确执行。 Hive 相对

表 3 100 GB- 相对 8节点集群的平均加速比

系统	Hive-Tez	Impala	Spark SQL
16节点	1.50	1.27	1.75
32节点	1.87	1.69	2.82

表 4 300 GB 相对 100 GB 的平均加速比

系统	Hive-Tez	Impala	Spark SQL
300 GB	0.633	0.437	0.430

稳定，但其采用的 Map-side-join 技术，在某些数据量和节点数配置下，会出现错误。说明这些新技术的引入，还存在很多细节问题没有得到很好的解决，需要进一步的研究和改进。从Hive 、Spark SQL和Impala 的对比来看，前两者由于拥有更好的开放性和更大范围 Hadoop生态圈的支持，稳定性进步比较明显；相比而言，在开放性方面相对保守的 Impala 系统存储的问题比较多，还需更长时间完善。

测试结果说明，SQL on Hadoop系统在查询优化和执行的一些细节方面还存在一些问题，需要时间进一步完善。在这一过程中 Hive 和 Spark SQL由于其有更好的开放性，所存在的细节问题更容易得到解决。

3.4 物理机与虚拟机性能对比

为了更好地了解物理机和虚拟机集群在查询处理性能方面的差别，在8个节点的虚拟机集群和物理机集群中对3个系统进行性能对比实验，所选用节点的内存数量一样，CPU个数和磁盘性能有所不同。其结果见表 5。

表 5 8 节点 -100GB- 物理机与虚拟机性能对比

	处理时间 /s					
	Hive-Tez		Impala		Spark SQL	
	虚拟机	物理机	虚拟机	物理机	虚拟机	物理机
Q1	70.1	34.0	21.3	10.0	126.7	25.0
Q2	79.8	47.2	39.9	30.8	81.2	54.0
Q3	202.2	100.0	time out	time out	203.2	92.1
Q4	174.1	71.1	time out	time out	141.7	38.7
Q5	time out	time out	time out	time out	time out	failed
Q6	41.1	22.6	11.2	5.9	44.6	13.0
Q7	380.8	138.3	542.6	280.3	479.1	401.5
Q8	515.9	200.6	678.5	312.3	659.1	706.1
Q9	501.0	240.2	1444.7	700.8	710.5	899.0
Q10	223.1	108.9	115.7	79.3	153.3	83.6
Q11	71.0	36.9	failed	failed	time out	failed
Q12	118.4	53.0	25.5	18.6	173.5	58.6
Q13	124.9	70.6	184.3	90.9	85.4	27.1
Q14	65.9	36.4	25.2	18.9	83.0	30.9
Q15	71.9	55.5	20.0	12.2	146.0	31.7
Q16	108.1	48.5	49.7	39.9	67.4	27.6
Q17	319.9	168.3	275.5	130.2	622.8	failed
Q18	483.3	155.0	698.5	330.9	699.7	failed
Q19	86.5	42.6	39.6	30.3	65.8	37.3
Q20	195.0	125.5	70.6	60.4	173.0	77.4
Q21	985.4	334.8	946.9	450.3	1 078.7	failed
Q22	123.8	60.6	failed	failed	86.7	30.8

可以明显看出，在大多数查询上，物理机的性能要明显好于虚拟机。原因可能是多方面的，虚拟机环境存在同一物理节点上不同虚拟机争抢资源的情况（如 I/O 通道）。物理机与虚拟机的性能差别，一定程度上也体现了硬件差异对数据分析性能的影响，但scale-up 不是大数据技术合理的发展方向，依赖更多的普通机器的 scale-out 技术方案才有更大的发展前景。

SQL on Hadoop系统的潜力巨大，随着普通机器硬件性能的提升，同时借鉴MPP并行数据库技术，其终将在低成本的大数据 OLAP市场中占据重要的位置。

4 结束语

OLAP是传统商业智能中最重要的一类分析任务。尽管 MPP并行数据库提供了高性能的 OLAP技术，但从近些年大数据分析系统的发展趋势来看，SQL on Hadoop系统在未来的 OLAP市场中将占有重要的位置，是值得关注的方向。除了体系结构与生俱来的优势（如容错、可扩展、开源、低成本），制约其实用性的最大困难——性能在过去数年中已经有了长足的进步。技术的渗透和融合在这个过程中起到了主导的作用。Stonebraker 与 Hadoop 社区的争论似乎还在耳边，传统数据库技术与 Hadoop技术的融合早就悄悄地在进行，完善 SQL on Hadoop系统今后还要更多地从数据库技术中汲取养分。

此外，依赖 Hadoop的数据分析系统还要在数据存储的透明性上做出牺牲，以换取更高的 OLAP性能。从测试结果分析来看，目前 SQL on Hadoop系统还存在不少问题，如何根据资源情况和数据分布更好地进行查询优化，如何进一步提高系统的顽健性，如何增加对 SQL更全面的支持，都给有志于此的科技工作者提供了很大的探索空间。

致谢

中国人民大学信息学院数据库与智能信息检索实验室的研究生卞昊穹、陈峻、李帅、刘捷思、张慧杰参与了系统评测工作。此项研究在中国人民大学“人大行云”平台上完成。

参考文献

[1] Codd E F , Codd S B, Salley C T. Providing OLAP (online analytical processing) to user-analysts: an IT mandate. E f codd & Associates, 1998

[2] Thomsen E. OLAP Solutions: Building Multidimensional Information Systems, 2nd Edition. Hoboken: John Wiley & Sons, 2002

[3] Daniel M S, Abadi D J, Batkin A, et al. C-store: a column-oriented DBMS. Proceedings of the 31st Very Large Data Bases (VLDB) Conference, Trondheim, Norway, 2005: 553~564

[4] Kaufmann M, Manjili A A, Vagenas P , et al. Timeline index: a unified data structure for processing queries on temporal data in SAP HANA. Proceedings of Acm Special Interest Group on Management of Data (SIGMOD) International Conference on Management of Data, New York, USA, 2013: 1173~1184

[5] Dean J, Ghemawat S. MapReduce: simplified data processing on large clusters. Proceedings of Operating Systems Design and Impleme nt at io n (OSDI), San Francisco, CA, USA, 2004: 137~150

[6] Pavlo A, Paulson E, Rasin A, et al. A comparison of approaches to large-scale

data analysis. Proceedings of the ACM Special Interest Group on Management of Data (SIGMOD) International Conference on Management of Data, Providence, USA, 2009: 165~178

[7] Chen Y G, Qin X P, Bian H Q, et al. A study of SQL-on-hadoop systems. Lecture Notes in Computer Science, 2014(8807): 154~166

[8] Hive cost based optimization. https://cwiki.apache.org/confluence/display/Hive/Cost-based +optimization+in+Hive, 2015

[9] Ion Stoica. Berkeley data analytics stack (BDAS) overview. http://amp camp. berkeley. edu/wp-content/uploads/2013/02/Berkeley-Data-Analytics-Stack-BDAS-Overview-Ion-Stoica-Strata-2013.pdf, 2013

作者简介

杜小勇，男，博士，中国人民大学信息学院教授、 博士生导师， 中国计算机学会会士，数据库专家委员会委员、 大数据专家委员会委员， 人民邮电出版社《大数据》期刊编委会副主任， Springer 出版社《Communications of Computer and Information Systems》系列编委， 主要研究方向为智能信息检索、 高性能数据库、 知识工程。 主持和参与多项国家核高基 （核心电子器件、高端通用芯片及基础软件产品） 、“ 973 ”计划、“ 863 ”计划、国家自然科学基金项目， 近年来在 SIGMOD、VLDB、 AAAI、 IEEE TKDE 等国际重要期刊和会议上发表论文百余篇。

陈跃国，男，博士，中国人民大学信息学院副教授、 硕士生导师， 中国计算机学会高级会员， 大数据专家委员会通信委员， Frontiers of Computer Science 青年编委，主要研究方向为大数据分析系统和语义搜索。 主持国家自然科学基金项目 2项，参与多项国家核高基（核心电子器件、 高端通用芯片及基础软件产品） 、“ 973 ”计划、“ 863 ”计划项目， 近年来在 ICDE、AAAI、 IEEE TKDE等国际重要期刊和会议上发表论文 30余篇。

覃雄派，男，博士，中国人民大学信息学院讲师、 硕士生导师，目前主要从事高性能数据库、 大数据分析、 信息检索等方面的研究工作， 主持 1项国家自然科学基金面上项目，参与多项国家“ 863 ”计划、“ 973 ”计划及国家自然科学基金项目， 在国内外期刊和会议上发表论文 20余篇。

收稿日期 ：2015-04-30； 修回日期 ：2015-05-10

基金项目 ：国家自然科学基金面上项目“ 高度可扩展的数据仓库数据编码方法及查询处理新技术研究”（No.61170013 ） ，中国人民大学科学研究基金（ 中央高校基本科研业务费专项资金） 资助项目 （No.14XNLQ06） ，国家社会科学基金重大项目 “ 云计算环境下的信息资源集成与服务研究 ” （No.12&ZD220）

Found ation Items: The Nation al Scie nc e Fou nd at io n of Ch in a Under Gr ant (No. 61170 013), The Fundamental Research Funds for the Central Universities, the Research Funds of Renmin University of Ch ina(No.14 XNLQ06), The Nat ional Social Science Fo und at ion of Ch ina : Maj or Re sea rch Project(No.12&ZD220)

论文引用格式 ：杜小勇， 陈跃国， 覃雄派．大数据与 OLAP系统．大数据 , 2015005

Du X Y, Chen Y G, Qin X P. Big data and OLAP systems. Big Data Research, 2015005