软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software,2012,23(1):32−45 [doi: 10.3724/SP.J.1001.2012.04091]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

大数据分析——RDBMS 与 MapReduce 的竞争与共生*

覃雄派[1,2+], 王会举[1,2], 杜小勇[1,2], 王 珊[1,2]

[1](教育部数据工程与知识工程重点实验室(中国人民大学),北京 100872)

[2](中国人民大学 信息学院,北京 100872)

Big Data Analysis—Competition and Symbiosis of RDBMS and MapReduce

QIN Xiong-Pai[1,2+], WANG Hui-Ju[1,2], DU Xiao-Yong[1,2], WANG Shan[1,2]

[1](MOE Key Laboratory of Data Engineering and Knowledge Engineering (Renmin University of China), Beijing 100872, China)

[2](School of Information, Renmin University of China, Beijing 100872, China)

+ Corresponding author: E-mail: qxp199@sina.com

Qin XP, Wang HJ, Du XY, Wang S. Big data analysis—Competition and symbiosis of RDBMS and MapReduce. *Journal of Software*, 2012,23(1):32−45. http://www.jos.org.cn/1000-9825/4091.htm

Abstract: In many areas such as science, simulation, Internet, and e-commerce, the volume of data to be analyzed grows rapidly. Parallel techniques which could be expanded cost-effectively should be invented to deal with the big data. Relational data management technique has gone through a history of nearly 40 years. Now it encounters the tough obstacle of scalability, which relational techniques can not handle large data easily. In the mean time, none relational techniques, such as MapReduce as a typical representation, emerge as a new force, and expand their application from Web search to territories that used to be occupied by relational database systems. They confront relational technique with high availability, high scalability and massive parallel processing capability. Relational technique community, after losing the big deal of Web search, begins to learn from MapReduce. MapReduce also borrows valuable ideas from relational technique community to improve performance. Relational technique and MapReduce compete with each other, and learn from each other; new data analysis platform and new data analysis eco-system are emerging. Finally the two camps of techniques will find their right places in the new eco-system of big data analysis.

Key words: big data; deep analysis; relational data management technique; MapReduce

摘 要: 在科学研究、计算机仿真、互联网应用、电子商务等诸多应用领域,数据量正在以极快的速度增长,为了分析和利用这些庞大的数据资源,必须依赖有效的数据分析技术.传统的关系数据管理技术(并行数据库)经过了将近 40 年的发展,在扩展性方面遇到了巨大的障碍,无法胜任大数据分析的任务;而以 MapReduce 为代表的非关系数据管理和分析技术异军突起,以其良好的扩展性、容错性和大规模并行处理的优势,从互联网信息搜索领域开始,进而在数据分析的诸多领域和关系数据管理技术展开了竞争.关系数据管理技术阵营在丧失搜索这个阵地之后,开始

* 基金项目: 国家自然科学基金(61070054, 60873017, 61170013); 核高基重大科技专项(2010ZX01042-001-002, 2010ZX 01042-002-002-03); 中央高校基本科研业务费专项资金(10XNI018)

收稿时间: 2011-04-04; 定稿时间: 2011-07-21; jos 在线出版时间: 2011-09-09

CNKI 网络优先出版: 2011-09-09 13:54, http://www.cnki.net/kcms/detail/11.2560.TP.20110909.1354.002.html

考虑自身的局限性,不断借鉴 MapReduce 的优秀思想改造自身,而以 MapReduce 为代表的非关系数据管理技术阵营,从关系数据管理技术所积累的宝贵财富中挖掘可以借鉴的技术和方法,不断解决其性能问题.面向大数据的深度分析需求,新的架构模式正在涌现.关系数据管理技术和非关系数据管理技术在不断的竞争中互相取长补短,在新的大数据分析生态系统内找到自己的位置.

关键词: 大数据;深度分析;关系数据管理技术;MapReduce

中图法分类号: TP311 **文献标识码**: A

1 大数据时代的来临

1.1 数据量的增长

在科学研究(天文学、生物学、高能物理等)[1]、计算机仿真、互联网应用、电子商务等领域,数据量呈现快速增长的趋势.比如:在科学研究方面,大型强子对撞机每年积累的新数据量为 15PB 左右(http://www-conf.slac.stanford.edu/xldb07/xldb_lhc.pdf);在电子商务领域,沃尔玛公司(Wal-Mart)每天通过 6 000 多个商店,向全球客户销售超过 2.67 亿(267Million)件商品(Data-Intensive Supercomputing: The Case for DISC. CMU Tech Report 2007),为了对这些数据进行分析,HP 公司为沃尔玛公司建造了大型数据仓库系统,数据规模达到 4PB,并且仍在不断扩大.

除了上述典型例子,我们还可以列举出大规模数据的几个主要来源:(1) 传感器数据(sensor data):分布在不同地理位置上的传感器,对所处环境进行感知,不断生成数据.即便对这些数据进行过滤,仅保留部分有效数据,长时间累积的数据量也是非常惊人的;(2) 网站点击流数据(click stream data):为了进行有效的市场营销和推广,用户在网上的每个点击及其时间都被记录下来;利用这些数据,服务提供商可以对用户存取模式进行仔细的分析,从而提供更加具有针对性的服务;(3) 移动设备数据(mobile device data):通过移动电子设备包括移动电话和 PDA、导航设备等,我们可以获得设备和人员的位置、移动、用户行为等信息,对这些信息进行及时的分析,可以帮助我们进行有效的决策,比如交通监控和疏导系统;(4) 射频 ID 数据(RFID data):RFID 可以嵌入到产品中,实现物体的跟踪.一旦 RFID 得到广泛的应用,将是大量数据的主要来源之一.

随着数据生成的自动化以及数据生成速度的加快,需要处理的数据量急剧膨胀.

1.2 数据分析的新趋势:超越常规报表的深度分析需求的增长

为了从数据中发现知识并加以利用,指导人们的决策,必须对数据进行深入的分析,而不是仅仅生成简单的报表.这些复杂的分析必须依赖于复杂的分析模型,很难用 SQL 来进行表达,统称为深度分析(deep analysis).

如图 1 所示,人们不仅需要通过数据了解现在发生了什么,更需要利用数据对将要发生什么进行预测,以便在行动上做出一些主动的准备[2].比如通过预测客户的流失预先采取行动,对客户进行挽留.

这里,典型的 OLAP 数据分析操作(对数据进行聚集、汇总、切片和旋转等)已经不够用,还需要路径分析、时间序列分析、图分析、What-if 分析以及由于硬件/软件限制而未曾尝试过的复杂统计分析模型[2]等,典型的例子包括时间序列分析和图分析等:(1) 时间序列分析(time series analysis):商业组织积累了大量的交易历史信息,企业的各级管理人员希望从这些数据中分析出一些模式,以便从中发现商业机会,通过趋势分析,甚至预先发现一些正在涌现出来的机会.比如在金融服务行业,分析人员可以开发针对性的分析软件,对时间序列数据进行分析,寻找有利可图的交易模式(profitable trading pattern),经过进一步验证之后,操作人员可以使用这些交易模式进行实际的交易,获得利润;(2) 大规模图分析和网络分析(large-scale graph and network analysis):社会网络(social network)虚拟环境本质上是对实体连接性的描述.在社会网络中,每个独立的实体表示为图中的一个节点,实体之间的联系表示为一条边.通过社会网络分析,可以从中发现一些有用的知识,比如发现某种类型的实体(有一种类型的实体把各个小组连接在一起,称为网络中的关键实体).这些信息可以用于产品直销、组织和个体行为分析、潜在安全威胁分析等领域.随着社会网络规模的增长,从几何角度看,图的节点和边都不断增长.使

用传统的方法处理大规模的图数据显得力不从心,急需有效的手段对这类数据进行分析.

一种处理大数据的方法是使用采样技术,通过采样,可以把数据规模变小,以便利用现有的技术手段(关系数据库系统)进行数据管理和分析.然而在某些应用领域,采样将导致信息的丢失,比如 DNA 分析等.在明细数据上进行分析,意味着需要分析的数据量将急剧膨胀和增长.

综上所述,数据分析的两大趋势和挑战是:(1) 数据量的膨胀;(2) 数据深度分析需求的增长(Beyond Reporting:Requirements for Large-Scale Analytics.TDWI Research Whitepaper 2008).如图 2 所示.此外,数据类型不断多样化,包括各种非结构化、半结构化数据,对这些类型多样的数据进行管理和分析也是数据处理技术所面临的挑战.

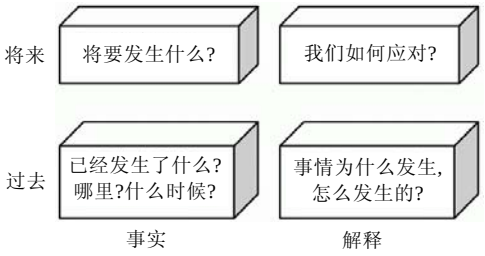


Fig.1 Dimensions of data analysis[2]

图 1 数据分析的维度[2]

Fig.2 Trends of data analysis[1]

图 2 数据分析的趋势[1]

2 以 MapReduce 为代表的非关系数据管理技术的兴起

关系数据库技术经过了将近 40 年的发展,成为一门成熟的、同时仍在不断演进的主流数据管理和分析技术.关系数据管理技术的主流应用包括 OLTP 应用、OLAP 应用以及数据仓库等.SQL 语言作为存取关系数据库系统的语言得到了标准化,经过不断扩充,其功能和表达能力不断增强.

但是,关系数据管理技术在大数据时代丧失了互联网搜索这个机会,其主要原因是关系数据管理系统(并行数据库)的扩展性遇到了前所未有的障碍,不能胜任大数据分析的要求.关系数据管理模型追求的是高度的一致性和正确性.面向超大数据的分析需求,纵向扩展(scale up)系统,即通过增加或者更换 CPU、内存、硬盘以扩展单个节点的能力,终将遇到瓶颈;横向扩展(scale out)系统,即通过增加计算节点连接成集群,并且改写软件,使之在集群上并行执行,才是经济的解决办法.使用大规模集群实现大数据的管理和分析,需要应对的挑战很多,其中,系统的可用性摆到了重要的位置[3].根据 CAP(consistency, availability, tolerance to network partitions)理论(Towards Robust Distributed Systems. PODC2004 Keynote)(对该理论尚存争议),在分布式系统中,一致性、可用性、容错性三者不可兼得,追求其中两个目标必将损害另外一个目标.并行数据库系统追求高度的一致性和容错性(通过分布式事务、分布式锁等机制),无法获得良好的扩展性和系统可用性,而系统的扩展性是大数据分析的重要前提.

2004 年,Google 公司最先提出 MapReduce[4]技术,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.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,它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:(1) 分布式文件系统;(2) 并行编程模型;(3) 并行执行引擎.

分布式文件系统(Google file system)运行于大规模集群之上,集群使用廉价的机器构建.数据采用键/值对(key/value)模式进行存储.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 3 个备份)实现高度容错.数据采用大块存储(64MB 或者 128MB 为 1 块)的办法,可方便地对数据进行压缩,节省存储空间和传输带宽.

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 Reduce 阶段.Map 函数处理

Key/Value 对,产生一系列的中间 Key/Value 对,Reduce 函数用来合并所有具有相同 Key 值的中间键值对,计算最终结果.

MapReduce 程序的具体执行过程如图 3 所示:首先对数据源进行分块,然后交给多个 Map 任务去执行,Map 任务执行 Map 函数,根据某种规则对数据分类,写入本地硬盘;Map 阶段完成后,进入 Reduce 阶段,Reduce 任务执行 Reduce 函数,具有同样 Key 值的中间结果,从多个 Map 任务所在的节点,被收集到一起(shuffle)进行合并处理,输出结果写入本地硬盘(分布式文件系统).程序的最终结果可以通过合并所有 Reduce 任务的输出得到.

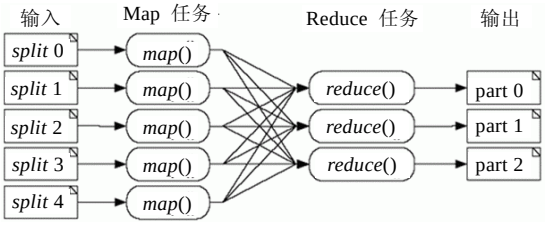


Fig.3 Parallel computing process of MapReduce

图 3 MapReduce 并行计算过程

MapReduce 技术是一种简洁的并行计算模型,它在系统层面解决了扩展性、容错性等问题,通过接受用户编写的 Map 函数和 Reduce 函数,自动地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. MapReduce 技术是非关系数据管理和分析技术的典型代表.在 Google 公司内部,通过大规模集群和 MapReduce 软件,每天有超过 20PB 的数据得到处理[4],每个月处理的数据量超过 400PB.在数据分析的基础上,Google 提供了围绕互联网搜索的一系列服务(包括地图服务、定向广告服务等).如此大规模的数据管理和分析,是传统的关系数据管理技术所无法完成的(见表 1).

Table 1 Google's applications based on MapReduce

表 1 Google 基于 MapReduce 的应用

--分布式 Grep	--建立倒排索引
--分布式排序	--文档聚类
--Web 连接图的反转	--机器学习
--基于主机的名词向量分析	--基于统计方法的机器翻译
--Web 存取日志分析	…

MapReduce 技术一经推出,立即遭到关系数据管理技术阵营(以著名的数据库技术专家 Stonebraker 为代表)的猛烈抨击.Stonebraker 认为,MapReduce 技术是一个巨大的倒退,并指出了 MapReduce 技术的众多缺点,包括不支持 Schema、没有存取优化、依靠蛮力(brute force)进行数据处理等.Stonebraker 等人在 100 个节点的集群上对 Hadoop 技术(MapReduce 的开源实现)、Vertica 数据库(一种基于列存储的关系数据库管理系统)和 DBMS-X 数据库(某厂商提供的商用数据库)进行了数据装载和数据分析的性能比较(包括 Grep,Selection,Join,Aggregation 等),MapReduce 的性能远远低于 Vertica 和 DBMS-X[5].但 Stonebraker 的批判并没有阻挡住以 MapReduce 技术为代表的大数据分析新技术的发展洪流.

近几年来,MapReduce 技术获得了广泛的关注,研究人员围绕 MapReduce 开展了深入的研究,包括 MapReduce 应用领域的扩展、MapReduce 性能的提升、MapReduce 易用性的改进等.同时,MapReduce 技术和 RDBMS 也出现了相互借鉴相互渗透的趋势.

2.1 MapReduce应用领域的扩展

在应用领域方面,MapReduce 技术已经从围绕搜索的数据分析扩展到数据挖掘、机器学习[6−10]、信息检索[11,12]、计算机仿真[13]、科学实验数据处理(生物、物理…)[14,15]等众多的领域.

针对传统分析软件扩展性差以及 Hadoop 分析功能薄弱的特点,IBM 公司的研究人员致力于对 R 和 Hadoop

的集成.R 是开源的统计分析软件,通过 R 和 Hadoop 的深度集成,把计算推向数据并且并行处理,使 Hadoop 获得了强大的深度分析能力[16].Purdue 大学的 RHIPE 项目(http://ml.stat.purdue.edu/rhipe/index.html)也致力于 R 和 Hadoop 的集成,为大数据分析提供开发环境的支持.Wegener 等人[17]则实现了 Weka(类似于 R 的开源的机器学习和数据挖掘工具软件)和 MapReduce 的集成.标准版 Weka 工具只能在单机上运行,并且不能超越 1GB 内存的限制.经过算法的并行化,在 MapReduce 集群上,Weka 不仅突破了原有的可处理数据量的限制,轻松地对超过 100GB 的数据进行分析,同时利用并行计算提高了性能.经过改造的 Weka,赋予 MapReduce 技术深度分析的能力.基于文献[6]以及相关文献,若干开发者发起了 Apache Mahout 项目的研究,该项目是基于 Hadoop 平台的大规模数据集上的机器学习和数据挖掘开源程序库,为应用开发者提供了丰富的数据分析功能.

2.2 MapReduce性能提升的研究

2.2.1 多核硬件与 GPU 上的性能改进

在 MapReduce 的性能提升方面,研究人员做了大量的工作.MIT(Optimizing MapReduce for Multicore Architectures.MIT Tech. Report 2010)和 Manchester 大学的研究人员研究了多核硬件上的 MapReduce 性能改进[18].文献[19,20]讨论了 Cell Broadband Engine 上的 MapReduce 性能优化技术,其中,Wisconsin 大学的研究人员利用 Cell Sort 算法,充分发挥硬件能力,极大地提高了排序的性能.MapReduce 在多核硬件上的性能改进工作还包括文献[21]等.此外,GPU 的核心数量和工作频率不断提高,Texas 大学 Austin 分校(DisMaRC: A Distributed MapReduce Framework on CUDA.University of Texas at Austin Tech. Report 2009)等科研机构的研究人员,就如何利用 GPU 提高 MapReduce 的执行性能展开了研究[22−24],并且扩展了 MapReduce 的应用领域.清华大学和 IBM 实验室的研究人员提出了 Map CG[25],在源代码级提供 CPU 编程和 GPU 编程的可移植性,大大提高了 MapReduce 程序编写的容易程度.Ohio 州立大学的研究人员面向多核环境,提出 MATE 编程接口与环境,不仅减小了内存占用,同时,性能也大大超越 Hadoop 和 Phoenix[26](Phoenix 是一种 MapReduce 的 C++实现,http://MapReduce.stanford.edu/).

2.2.2 索引技术与连接技术的优化

中国科学院计算技术研究所围绕 MapReduce 开展了索引优化[27,28]、利用分布式内存 Cache 提高性能[29]等研究.文献[30]研究了非侵入式的 MapReduce 性能提升技术,包括特洛伊索引(Trojan index)和分区数据并置(co-partition,即把需要连接的数据分区保存到同一个节点或者在网络拓扑上接近的节点,以加快数据分区之间的 Join 操作)技术等.而文献[31]则提出事实表上的虚拟视图(virtual view over fact table)、事实表和维表连接的优化、基于列存储的压缩(columnar compression)等技术,提高了 MapReduce 环境下星型模型上的 OLAP 类应用的执行性能.Iu 等人[32]通过对 MapReduce 执行函数的分析,对 MapReduce 查询进行改写,充分利用 SQL 数据库的索引、聚集函数等功能,提高 MapReduce 函数的执行效率.文献[33]研究了 MapReduce 架构下面向日志处理的连接操作的性能,在 100 个节点组成的 Hadoop 集群上进行若干连接技术的性能研究,包括标准的重新分区连接方法(standard repartition join)、改进的重新分区连接方法(improved repartition join)、直接连接方法(directed join)、广播连接方法(broadcast join)、半连接(semi-join)、基于分片的半连接(per-split semi-join)等,为不同应用场景下使用不同的连接技术提供了参考.周傲英等人提出基于 MapReduce 的列存储数据的连接优化方法,极大地加快了连接的速度[34].文献[35]研究了星型模型上特大事实表和特小维表之间的连接优化方法和图数据上的路径连接(chain join)优化方法.

2.2.3 调度技术优化

文献[36]试图利用基于优先级的调度策略提高 MapReduce 的运行效率.文献[37]提出了基于 MPI 的 MapReduce 优化实现,利用 MPI-3 的新特性,比如 MPI Reduce Local 等,在 127 个节点的集群上获得 25%的性能提升.Toronto 大学和 Boston 大学的研究人员[38]尝试在多个 MapReduce Job 之间进行查询处理工作的共享,以此提高系统的总体吞吐能力.Purdue 大学(Relaxed Synchronization and Eager Scheduling in MapReduce.Purdue University Tech Report 2009)的研究人员通过放松同步要求和饥渴式调度(eager scheduling)方法,提高 MapReduce 任务的执行效率[39].Barcelona 超级计算中心和 IBM Watson 实验室的研究人员研究了任务联合调度

策略[40],以期提高性能.文献[41–44]研究了异构处理器和异构集群环境下新的任务调度算法,保证并行任务执行的性能不受异构环境的负面影响.

2.2.4 其他优化技术

新加坡国立大学的研究人员[45]提出了 5 种有效的优化方法,包括基于指纹的分组方法(fingerprint based grouping)、直接存取文件系统(direct I/O in HDFS)、在数据解析中使用可变的 Java 对象(mutable Java object for data parsing)、使用索引(using indexing)以及数据块感知的调度方法(block-aware scheduling)等,一举提高 Hadoop 系统的数据分析性能,大幅度逼近传统关系数据库的性能.文献[45]指出,在大规模数据分析领域,基于 Hadoop 的数据分析系统具有比传统数据库更好的扩展性,足以使得 Hadoop 系统成为和并行数据库正面竞争的一支力量.如图 4 所示.Berthold 等人[46]基于 Eden 平台,使用延迟数据流处理(lazy stream processing)、动态应答通道(dynamic reply channel)、数据流合并(stream merge)等技术优化 MapReduce 的实现.文献[47]提出利用生产者和消费者的共享缓冲区(shared buffer between producer and consumer),消除 MapReduce 两个计算阶段(Map 阶段和 Reduce 阶段)的传输瓶颈.文献[48]提出在 MapReduce 两个计算阶段的基础上增加一个 Merge 阶段,以更好地支持集合合并(set union)、Join 等操作,同时提出了合并 Reduce 和 Merge 操作以改进性能的办法.韩国科学技术院以及三星公司、Yahoo 公司的研究人员,利用预取技术和预 Shuffle 技术提高 MapReduce 的执行性能[49].Duke 大学的研究人员进行了 MapReduce 执行系统的自调优研究[50],以减轻运行时系统的手工配置要求.

HadoopOpt 和并行数据库的非直接性能比较

	DBMS-X (倍)	Vertica (倍)	HadoopOpt (倍)
Grep 文本搜索	1.5	2.6	1.47
聚集(大规模)	1.6	4.3	1.54
连接	36.3	21.0	14.68

Fig.4 Performance improvement of HadoopOpt[45]

图 4 HadoopOpt 的性能改进[45]

2.3 MapReduce易用性的改进

针对 MapReduce 技术缺乏类似 SQL 的标准存取语言、依靠底层语言编程的弱点,研究人员研究更为高层的、表达能力更强的语言和系统,包括 Yahoo 的 Pig[51]、Microsoft 的 LINQ[52−54]、Hive 等.Pig 是编写 MapReduce 程序的脚本语言,Yahoo 不仅致力于提高 MapReduce 的易用性,同时还不断提高 Pig 的性能,采用包括操作符间的 Pipeline 等技术避免物化中间结果,从而提高性能[55],并且支持数据流的处理.此外,值得指出的是,Microsoft 的 Dryad 系统通过有向无环图表达基于串行程序的并行计算,进而在大规模集群上并行执行.虽然与 MapReduce 技术在概念上有些区别,但从渊源来看,可以把它看成是 MapReduce 技术的变种,同属非关系数据管理和分析技术阵营.Hive[56]是基于 Hadoop 的大型数据仓库系统,实现了 Schema,SQL 查询等类关系数据库的功能. Facebook 在 Hive 上实现了例行性报表、即席(ad hoc)查询、机器学习以及其他复杂的数据分析;通过 SQL 接口,改善了 MapReduce 技术的易用性和接受度.文献[57]提出 Hadoop-ML,利用该语言环境,开发人员可以很方便地在程序块的基础上构建任务并行或数据并行的机器学习和数据挖掘算法.开源软件 Cascading 是基于 Hadoop 的一个 Java 库,包含查询 API(query API)、查询计划器(query planner)和进程调度器(process scheduler),是 Hadoop 上的工作流软件,开发者可以在 Cascading 的基础上快速地组装并行数据处理程序.

3 RDBMS 和 MapReduce 技术的竞争与相互渗透

MapReduce 技术在广泛用于搜索相关的数据分析工作之后,随着其性能的不断提升和应用领域的扩展,迅速成为 RDBMS 的年轻的竞争者,两者的竞争也促进了其相互学习和渗透.表 2 对比了 MapReduce 技术和关系数据库技术的特点.

MIT 的研究人员[58]借鉴 MapReduce 的容错思想,试图在 Shared Nothing 架构的并行数据库系统上实现更高的容错性能,取得良好的容错和负载均衡效果.

Table 2 Characteristics of RDBMS and MapReduce

表 2 RDBMS 与 MapReduce 的特点比较

	RDBMS	MapReduce
模式	内部支持	外部附加
索引	内部支持	编程实现
数据类型	结构化数据	非结构化、半结构化、结构化数据
编程模型	声明性语言 SQL	过程性语言
灵活性	有限	大
扩展性	上百节点	上千节点
容错性	低,查询重启	高,子任务重新执行
性能	高	比 RDBMS 低[1]
应用范围	在线事务处理 在线分析处理	批量处理[2] 深度分析

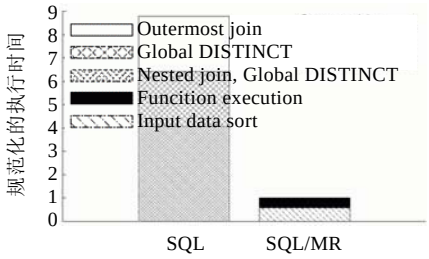
1. 目前,已有大量的研究致力于提高 MapReduce 的性能(参考第 2.2 节);
2. 持续分析与增量分析可以减小分析延迟.

HadoopDB[59]是试图混合 MapReduce 和 RDBMS 技术的一项重要工作.在 HadoopDB 中,系统清晰地分成两层,上层使用 Hadoop 进行任务的分解和调度,下层用 RDBMS(Postgresql)进行数据的查询和处理.该工作的创新之处是:试图利用 Hadoop 的任务调度机制提高系统的扩展性和容错性,以解决大数据分析的横向扩展问题;利用 RDBMS 实现数据存储和查询处理,以解决性能问题.在其性能实验中,HadoopDB 的性能仍然落后于关系数据库系统.如何提升 MapReduce 的性能,已引起研究人员的高度重视(见第 2.2 节),研究人员提出了 MapReduce 的各种优化技术,获得了重要的性能改进.Yale 大学 Abadi 领导的小组正在使用包括列存储、持续装载和分析(continuous loading and analysis)等技术,以改进 HadoopDB 的性能[60].

Greenplum[2]和 Aster Data[61]是两家新兴的面向大数据分析的公司,他们采用的策略是在 MPP 架构的并行数据库里内置地支持 MapReduce,其核心引擎同时作为 MapReduce 作业的执行引擎.两家公司正在进行一项重要的工作,即对分析函数进行 MapReduce 风格的并行化(MapReduce style parallelization).通过并行化,数据分析函数的执行性能大幅提升.通过引进 MapReduce 计算模型的思想,对传统的并行数据库进行改造,两家公司的 MPP 架构的并行数据库系统可以轻松扩展到几百个节点的规模.Aster Data 更是在 2010 年中发布了超过 30 个的分析软件包,提供上千个可以定制的分析函数,这些函数都将以并行的方式运行在 MPP 平台上,从而在性能上大大超越传统的 RDBMS 用户自定义函数(UDF).图 5 所展示的是经过 MapReduce 并行化改写后的分析函数的执行性能和 SQL(包含子查询)查询性能的比较,可见,MapReduce 技术大幅度提升了分析函数的执行性能.

随着 MapReduce 技术性能的提升、应用领域的扩展,关系数据管理技术和 MapReduce 技术的争论一直持续着.2010 年初,ACM 通讯杂志同时向 Stonebraker[62]以及 Google 的 Dean 进行约稿[63].Dean 指出,MapReduce 是进行大规模数据分析处理的灵活而有效的工具;而 Stonebraker 则从最初的对 MapReduce 技术的彻底否定,转为肯定 MapReduce 的良好扩展性,并且指出,MapReduce 非常适合做 ETL 这样的工作.目前,越来越多的数据库研究人员(包括 Stonebraker 在内)逐渐意识到,MapReduce 和关系数据库可以互相学习,并且走向集成(Andrew Pavlo. MapReduce and Parallel DBMSs: Together at Last. New England Database Summit 2010). MapReduce 可以从 RDBMS 学习查询优化、Schema 支持、外围工具(ETL 工具、可视化工具等)支持等,而 RDBMS 可以从 MapReduce 学习得到高度的扩展性和容错性、快速装载、易于使用等特点.

除了 Greenplum,Aster Data 等新兴公司以外,Oracle,Teradata,IBM,Vertica 等传统数据库厂商也致力于 MapReduce 和 RDBMS 的集成.它们所采用的策略基本类似,即在 RDBMS 引擎内支持 MapReduce 作业的运行.与 Greenplum 和 Aster Data 的分析函数并行化改写技术方案相比,Teradata[64]的工作相对简单,仅仅实现了数据装载的加速、数据库表和 HDFS(Hadoop file system)的互相转换等功能.Vertica 数据库系统在 2009 年底开始了 MapReduce 技术的集成,通过集成,使得 Vertica 数据库不仅能够处理结构化数据,而且能够处理非结构化数据和半结构化数据.Vertica 数据库的前身是 C-Store 数据库原型,C-Store 数据库是在 Stonebraker 的领导下开发的基于列存储、大内存、压缩等技术,面向数据分析应用的数据库系统,Stonebraker 本人为 Vertica 的 CTO.



SQL 和 SQL/MR 的点击流数据分析的时间分解

Fig.5 Performance of analytic function in aster data[61]

图 5 Aster Data 分析函数的性能[61]

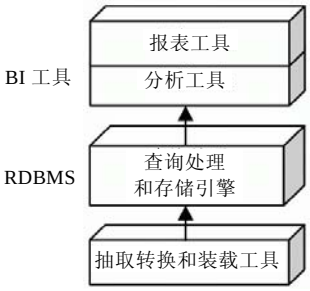


Fig.6 Data analysis eco-system of RDBMS

图 6 围绕 RDBMS 的数据分析生态系

4 把分析推向数据以及数据分析新生态系统的浮现

4.1 把分析推向数据

随着数据量的增长,对大数据进行分析的基本策略是把计算推向数据,而不是移动大量的数据(In-Database Analytics: The Heart Of The Predictive Enterprise. Forrester Whitepaper 2009).

围绕关系数据库管理系统,衍生出了传统的数据分析生态系统(eco-system,生态系统是指多种生物共存共生的自然系统,在这里用来表达围绕数据分析的共存的各类系统和工具).关系数据库作为核心的数据引擎,各种来源的数据通过 ETL 工具导入关系数据库系统,客户端工具通过 SQL 语言实现例行性的报表生成.针对复杂的分析,SQL 的表达能力就暴露出了其局限性,无法胜任.这时,必须把数据从数据库中提取出来,导入前端分析工具(SAS,SPSS)以进行后续分析.

这种模式的主要缺点是,由于 SQL 分析能力的局限,需要借助于统计分析软件进行数据的深度建模和分析,导致了大量数据的移动.需要指出的是,当分析人员从关系数据库中利用 SQL 查询把数据提取到分析软件中(比如 SAS)进行后续分析时,SQL 退化成为数据提取的接口.最为致命的是,大量数据的移动导致性能下降,这是大规模数据分析所应该极力避免的.值得指出的是,SAS 等数据分析厂商正在致力于把分析能力下压到数据库系统执行,但是进行得不是很彻底,分析函数的并行化以及系统的扩展性仍然是有待解决的问题.

相对于 RDBMS,MapReduce 技术从存储模型和计算模型上支持更高的容错性、更强的扩展性,为大数据分析提供了很好的运行平台保障.同时,难以用 SQL 进行表达的分析任务更容易用 MapReduce 计算函数表达(如图分析、各种数据挖掘算法等,参考第 2.1 节).可见,MapReduce 技术在数据的深度分析上比 RDBMS 更胜一筹.

4.2 新生态系统的浮现

随着 MapReduce 技术的兴起,我们看到,数据分析的生态系统正在发生变化,Facebook 的系统就是一个典型的范例[65].Facebook 系统的数据量是 15PB(压缩以后为 2.5PB),每天增加的数据量是 60TB(压缩以后是 10TB).如此庞大的数据量迫使 Facebook 采用新的数据处理架构.

如图 7 所示,在 Facebook 数据分析系统中,关系数据库系统处在系统的边缘(挂接在 Web server farm 上),负责进行 OLTP 类的事务处理.交易数据通过定时的装载,导入核心生产用 Hive 系统(production Hive-Hadoop cluster),重要的分析功能在 Hive 系统里面完成.经过分析和聚集的结果(summary data),可以重新注入关系数据库系统(包括 Oracle RAC,federated MySQL 等),接受用户的查询.为了减轻即席查询对核心 Hive 系统的压力,数据被复制到一个备份的 Hive 系统(ad hoc Hive-Hadoop cluster),进行用户即席查询的处理,隔离未经优化的查询有可能给核心 Hive 系统造成的性能冲击,保证核心数据分析系统的性能.

从这个体系结构和生态环境中,我们看到浮现的大数据分析生态系统的几个特点:首先,系统具有高度的扩展性,支持 PB 级甚至更大规模数据的分析和处理;靠近数据进行数据的深度分析,无需大量数据的移动;得益于 Hadoop 的并行能力,经过改写的分析函数获得良好的性能;整个系统既可以处理结构化数据,也可以处理非结

构化和半结构化数据.

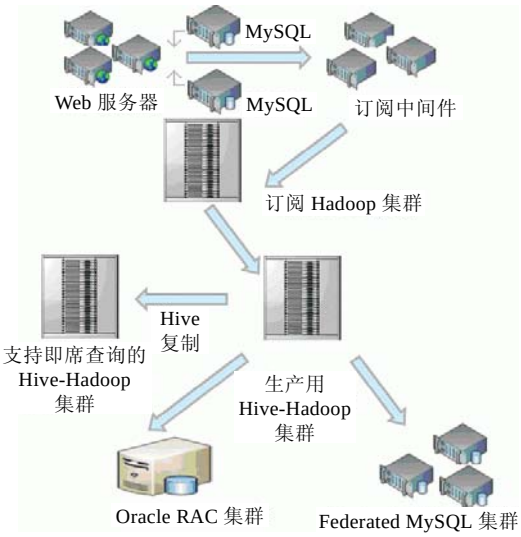


Fig.7 Architecture of the analysis platform in Facebook[65]

图 7 Facebook 数据分析平台架构[65]

在这个生态系统里,RDBMS 负责其擅长的 OLTP 类应用,为大数据分析平台提供数据源;数据深度分析之后的汇总数据和分析结果重新导入 RDBMS,供用户观察(包括可视化)和使用;前端工具不再承担分析功能,仅仅实现数据的可视化;RDBMS 担任数据集市(data mart)的角色;真正的复杂深度的分析,依靠高度可扩展的 Hadoop 系统来完成.由于 MapReduce 技术所具有的良好的扩展性,可以实现大量历史数据的在线,历史久远的数据也可以唾手可得地进行分析,结合新数据和新算法,有利于新知识的发现.

5 当前研究热点和我们的研究

相对于关系数据库技术,MapReduce 是一项崭新的并行计算技术,仍然有若干重要问题有待研究:

(1) 更加复杂的分析、更大规模的分析:在 MapReduce 模型上实现更加复杂和更大规模的分析,比如更细粒度的仿真[66]、时间序列分析、大规模图(Christos Faloutsos. Mining Billion-node Graphs: Patterns, Generators and Tools. Hadoop Submit 2010)分析[67]和大规模社会计算[68]等;

(2) 继续改进 MapReduce 的性能,提供数据分析的实时性: MapReduce 是面向批处理的并行计算模型,其性能与关系数据库相比仍然有一定的差距.人们迫切希望尽快地从数据中发现知识,如何提高 MapReduce 的性能、增强大规模数据处理的实时性[69,70]是研究的热点之一.比如文献[71]提出在 MapReduce 上的增量式数据挖掘方法,能够极大地缩短数据挖掘的时间.此外,基于数据流的数据分析和挖掘也是加快知识获取速度的可行办法[72],Brown 大学已经开始这方面的研究(C-MR 系统)(ftp://ftp.cs.brown.edu/pub/techreports/10/cs10-01.pdf);

(3) 开发、调试与管理工具:在大数据上进行复杂的并行分析,需要开发、调试、管理等一整套支撑环境的支持[73−75];

(4) 云平台上 MapReduce 计算的节能问题与调度优化:MapReduce 作为云平台上进行大规模数据处理的重要技术,其节能问题引起了研究人员的兴趣,已有研究人员开始了这方面的研究[76].此外,云平台上的 MapReduce 计算的调度优化也是必须解决的问题[77];

(5) 突破 MapReduce 计算模型的局限性:深入分析 MapReduce 计算模型内在的局限性,考虑如何改进或扩展 MapReduce.比如提高 MapReduce 系统的容错性[78],改善 MapReduce 系统任务调度的方法[79],超越 MapReduce 的局限性,实现更为有效的[80]并行计算模型.Washington 大学的研究人员对 MapReduce 框架进行了

扩展,使之能够有效地支持迭代式并行程序的执行[81];

(6) 关系数据库和 MapReduce 混合技术研究:如上文所述,关系数据库和 MapReduce 技术各有优缺点,如何融合关系数据库和 MapReduce 技术,设计同时具备两者优点的技术架构(既有 MapReduce 的高度扩展性和容错性,又有 RDBMS 的高性能),也是大数据分析技术的研究趋势.

中国人民大学高性能数据库研究小组针对面向 OLAP 应用的数据仓库数据膨胀问题,采用层次编码方法,把星型模型中各个维表的层次信息编码到事实表中,然后把事实表横向分割,分布到大规模集群上以便并行处理.同时,通过改写 SQL 查询,将谓词演算转变为层次编码的操作.由于事实表数据已经包含聚集查询涉及的层次信息,子查询可以在集群上并行执行,节点之间无需交换数据.局部聚集结果由主节点进行合并,生成最终的结果集.我们已基于 Postgresql 进行了技术原型(LinearDB)的实现,取得了较大的性能提升[82].目前,我们正在 Hadoop 框架下移植层次编码方法及其查询处理方法,在充分利用 Hadoop 的节点管理能力和扩展性的同时,提高数据仓库星型查询的性能.初步实验结果显示(实验基于 14 台普通 PC,其中 1 台作为 Name Node),我们的 Dumbo 原型系统在 500GB SSB(star schema benchmark)数据集上获得了比 HadoopDB 平均高 6~7 倍的性能.

6 总 结

面对大数据深度分析的挑战,关系数据库技术的扩展性遇到了前所未有的困难.同时,SQL 的表达能力不足以进行复杂深入的数据分析.MapReduce 技术具有简洁的模型、良好的扩展性、容错性和并行性,随着其性能的不断改进和分析能力的不断增强(与 R,Weka 的结合等),在大数据分析的技术竞争中异军突起.关系数据库技术和 MapReduce 技术相互竞争、相互学习和相互渗透,促进了数据分析新生态系统的浮现.在新生态系统中,关系数据库技术和 MapReduce 技术找到了自己的位置,发挥出各自的优势,从大数据中分析和发现有用的知识.

References:

[1] Zhou AY. Data intensive computing-challenges of data management techniques. Communications of CCF, 2009,5(7):50−53 (in Chinese with English abstract).

[2] Cohen J, Dolan B, Dunlap M, Hellerstein JM, Welton C. MAD skills: New analysis practices for big data. PVLDB, 2009,2(2): 1481−1492.

[3] Schroeder B, Gibson GA. Understanding failures in petascale computers. Journal of Physics: Conf. Series, 2007,78(1):1−11. [doi: 10.1088/1742-6596/78/1/012022]

[4] Dean J, Ghemawat S. MapReduce: Simplified data processing on large clusters. In: Brewer E, Chen P, eds. Proc. of the OSDI. California: USENIX Association, 2004. 137−150. [doi: 10.1145/1327452.1327492]

[5] Pavlo A, Paulson E, Rasin A, Abadi DJ, Dewitt DJ, Madden S, Stonebraker M. A comparison of approaches to large-scale data analysis. In: Cetintemel U, Zdonik SB, Kossmann D, Tatbul N, eds. Proc. of the SIGMOD. Rhode Island: ACM Press, 2009. 165−178. [doi: 10.1145/1559845.1559865]

[6] Chu CT, Kim SK, Lin YA, Yu YY, Bradski G, Ng AY, Olukotun K. Map-Reduce for machine learning on multicore. In: Scholkopf B, Platt JC, Hoffman T, eds. Proc. of the NIPS. Vancouver: MIT Press, 2006. 281−288. [doi: 10.1234/12345678]

[7] Wang CK, Wang JM, Lin XM, Wang W, Wang HX, Li HS, Tian WP, Xu J, Li R. MapDupReducer: Detecting near duplicates over massive datasets. In: Elmagarmid AK, Agrawal D, eds. Proc. of the SIGMOD. Indiana: ACM Press, 2010. 1119−1122. [doi: 10.1145/1807167.1807296]

[8] Liu C, Guo F, Faloutsos C. BBM: Bayesian browsing model from petabyte-scale data. In: Elder JF IV, Fogelman-Soulié F, Flach PA, Zaki MJ, eds. Proc. of the KDD. Paris: ACM Press, 2009. 537−546. [doi: 10.1145/1557019.1557081]

[9] Panda B, Herbach JS, Basu S, Bayardo RJ. PLANET: Massively parallel learning of tree ensembles with MapReduce. PVLDB, 2009,2(2):1426−1437.

[10] Lin J, Schatz M. Design patterns for efficient graph algorithms in MapReduce. In: Rao B, Krishnapuram B, Tomkins A, Yang Q, eds. Proc. of the KDD. Washington: ACM Press, 2010. 78−85. [doi: 10.1145/1830252.1830263]

[11] Zhang CJ, Ma Q, Wang XL, Zhou AY. Distributed SLCA-based XML keyword search by Map-Reduce. In: Yoshikawa M, Meng XF, Yumoto T, Ma Q, Sun LF, Watanabe C, eds. Proc. of the DASFAA. Tsukuba: Springer-Verlag, 2010. 386−397. [doi: 10.1007/978-3-642-14589-6_40]

[12] Stupar A, Michel S, Schenkel R. RankReduce—Processing *K*-nearest neighbor queries on top of MapReduce. In: Crestani F, Marchand-Maillet S, Chen HH, Efthimiadis EN, Savoy J, eds. Proc. of the SIGIR. Geneva: ACM Press, 2010. 13−18.

[13] Wang GZ, Salles MV, Sowell B, Wang X, Cao T, Demers A, Gehrke J, White W. Behavioral simulations in MapReduce. PVLDB, 2010,3(1-2):952−963.

[14] Gunarathne T, Wu TL, Qiu J, Fox G. Cloud computing paradigms for pleasingly parallel biomedical applications. In: Hariri S, Keahey K, eds. Proc. of the HPDC. Chicago: ACM Press, 2010. 460−469. [doi: 10.1145/1851476.1851544]

[15] Delmerico JA, Byrnesy NA, Brunoz AE, Jonesz MD, Galloz SM, Chaudhary V. Comparing the performance of clusters, Hadoop, and active disks on microarray correlation computations. In: Yang YY, Parashar M, Muralidhar R, Prasanna VK, eds. Proc. of the HiPC. Kochi: IEEE Press, 2009. 378−387. [doi: 10.1109/HIPC.2009.5433190]

[16] Das S, Sismanis Y, Beyer KS, Gemulla R, Haas PJ, McPherson J. Ricardo: Integrating R and Hadoop. In: Elmagarmid AK, Agrawal D, eds. Proc. of the SIGMOD. Indiana: ACM Press, 2010. 987−998. [doi: 10.1145/1807167.1807275]

[17] Wegener D, Mock M, Adranale D, Wrobel S. Toolkit-Based high-performance data mining of large data on MapReduce clusters. In: Saygin Y, Yu JX, Kargupta H, Wang W, Ranka S, Yu PS, Wu XD, eds. Proc. of the ICDM Workshop. Washington: IEEE Computer Society, 2009. 296−301. [doi: 10.1109/ICDMW.2009.34]

[18] Kovoor G, Singer J, Luján M. Building a Java Map-Reduce framework for multi-core architectures. In: Ayguade E, Gioiosa R, Stenstrom P, Unsal O, eds. Proc. of the HiPEAC. Pisa: HiPEAC Endowment, 2010. 87−98.

[19] De Kruijf M, Sankaralingam K. MapReduce for the cell broadband engine architecture. IBM Journal of Research and Development, 2009,53(5):1−12. [doi: 10.1147/JRD.2009.5429076]

[20] Becerra Y, Beltran V, Carrera D, Gonzalez M, Torres J, Ayguade E. Speeding up distributed MapReduce applications using hardware accelerators. In: Barolli L, Feng WC, eds. Proc. of the ICPP. Vienna: IEEE Computer Society, 2009. 42−49. [doi: 10.1109/ICPP.2009.59]

[21] Ranger C, Raghuraman R, Penmetsa A, Bradski G, Kozyrakis C. Evaluating MapReduce for multi-core and multiprocessor systems. In: Dally WJ, ed. Proc. of the HPCA. Phoenix: IEEE Computer Society, 2007. 13−24. [doi: 10.1109/HPCA.2007.346181]

[22] Ma WJ, Agrawal G. A translation system for enabling data mining applications on GPUs. In: Zhou P, ed. Proc. of the Supercomputing (SC). New York: ACM Press, 2009. 400−409. [doi: 10.1145/1542275.1542331]

[23] He BS, Fang WB, Govindaraju NK, Luo Q, Wang TY. Mars: A MapReduce framework on graphics processors. In: Moshovos A, Tarditi D, Olukotun K, eds. Proc. of the PACT. Ontario: ACM Press, 2008. 260−269.

[24] Stuart JA, Chen CK, Ma KL, Owens JD. Multi-GPU volume rendering using MapReduce. In: Hariri S, Keahey K, eds. Proc. of the MapReduce Workshop (HPDC 2010). New York: ACM Press, 2010. 841−848. [doi: 10.1145/1851476.1851597]

[25] Hong CT, Chen DH, Chen WG, Zheng WM, Lin HB. MapCG: Writing parallel program portable between CPU and GPU. In: Salapura V, Gschwind M, Knoop J, eds. Proc. of the PACT. Vienna: ACM Press, 2010. 217−226. [doi: 10.1145/1854273.1854303]

[26] Jiang W, Ravi VT, Agrawal G. A Map-Reduce system with an alternate API for multi-core environments. In: Chiba T, ed. Proc. of the CCGRID. Melbourne: IEEE Press, 2010. 84−93. [doi: 10.1109/CCGRID.2010.10]

[27] Liao HJ, Han JZ, Fang JY. Multi-Dimensional index on Hadoop distributed file system. In: Xu ZW, ed. Proc. of the Networking, Architecture, and Storage (NAS). Macau: IEEE Computer Society, 2010. 240−249. [doi: 10.1109/NAS.2010.44]

[28] Zou YQ, Liu J, Wang SC, Zha L, Xu ZW. CCIndex: A complemental clustering index on distributed ordered tables for multi-dimensional range queries. In: Ding C, Shao ZY, Zheng R, eds. Proc. of the NPC. Zhengzhou: Springer-Verlag, 2010. 247−261. [doi: 10.1007/978-3-642-15672-4_22]

[29] Zhang SB, Han JZ, Liu ZY, Wang K, Feng SZ. Accelerating MapReduce with distributed memory cache. In: Huang XX, ed. Proc. of the ICPADS. Shenzhen: IEEE Press, 2009. 472−478. [doi: 10.1109/ICPADS.2009.88]

[30] Dittrich J, Quiané-Ruiz JA, Jindal A, Kargin Y, Setty V, Schad J. Hadoop++: Making a yellow elephant run like a cheetah (without it even noticing). PVLDB, 2010,3(1-2):518−529.

[31] Chen ST. Cheetah: A high performance, custom data warehouse on top of MapReduce. PVLDB, 2010,3(1-2):1459−1468.

[32] Iu MY, Zwaenepoel W. HadoopToSQL: A MapReduce query optimizer. In: Morin C, Muller G, eds. Proc. of the EuroSys. Paris: ACM Press, 2010. 251−264. [doi: 10.1145/1755913.1755939]

[33] Blanas S, Patel JM, Ercegovac V, Rao J, Shekita EJ, Tian YY. A comparison of join algorithms for log processing in MapReduce. In: Elmagarmid AK, Agrawal D, eds. Proc. of the SIGMOD. Indiana: ACM Press, 2010. 975−986. [doi: 10.1145/1807167.1807273]

[34] Zhou MQ, Zhang R, Zeng DD, Qian WN, Zhou AY. Join optimization in the MapReduce environment for column-wise data store. In: Fang YF, Huang ZX, eds. Proc. of the SKG. Ningbo: IEEE Computer Society, 2010. 97−104. [doi: 10.1109/SKG.2010.18]

[35] Afrati FN, Ullman JD. Optimizing joins in a Map-Reduce environment. In: Manolescu I, Spaccapietra S, Teubner J, Kitsuregawa M, Léger A, Naumann F, Ailamaki A, Ozcan F, eds. Proc. of the EDBT. Lausanne: ACM Press, 2010. 99−110. [doi: 10.1145/1739041.1739056]

[36] Sandholm T, Lai K. MapReduce optimization using regulated dynamic prioritization. In: Douceur JR, Greenberg AG, Bonald T, Nieh J, eds. Proc. of the SIGMETRICS. Seattle: ACM Press, 2009. 299−310. [doi: 10.1145/1555349.1555384]

[37] Hoefler T, Lumsdaine A, Dongarra J. Towards efficient MapReduce using MPI. In: Oster P, ed. Proc. of the EuroPVM/MPI. Berlin: Springer-Verlag, 2009. 240−249. [doi: 10.1007/978-3-642-03770-2_30]

[38] Nykiel T, Potamias M, Mishra C, Kollios G, Koudas N. MRShare: Sharing across multiple queries in MapReduce. PVLDB, 2010, 3(1-2):494−505.

[39] Kambatla K, Rapolu N, Jagannathan S, Grama A. Asynchronous algorithms in MapReduce. In: Moreira JE, Matsuoka S, Pakin S, Cortes T, eds. Proc. of the CLUSTER. Crete: IEEE Press, 2010. 245−254. [doi: 10.1109/CLUSTER.2010.30]

[40] Polo J, Carrera D, Becerra Y, Torres J, Ayguadé E, Steinder M, Whalley I. Performance-Driven task co-scheduling for MapReduce environments. In: Tonouchi T, Kim MS, eds. Proc. of the IEEE Network Operations and Management Symp. (NOMS). Osaka: IEEE Press, 2010. 373−380. [doi: 10.1109/NOMS.2010.5488494]

[41] Zaharia M, Konwinski A, Joseph AD, Katz R, Stoica I. Improving MapReduce performance in heterogeneous environments. In: Draves R, van Renesse R, eds. Proc. of the ODSI. Berkeley: USENIX Association, 2008. 29−42.

[42] Xie J, Yin S, Ruan XJ, Ding ZY, Tian Y, Majors J, Manzanares A, Qin X. Improving MapReduce performance through data placement in heterogeneous Hadoop clusters. In: Taufer M, Rünger G, Du ZH, eds. Proc. of the Workshop on Heterogeneity in Computing (IPDPS 2010). Atlanta: IEEE Press, 2010. 1−9. [doi: 10.1109/IPDPSW.2010.5470880]

[43] Polo J, Carrera D, Becerra Y, Beltran V, Torres J, Ayguadé E. Performance management of accelerated MapReduce workloads in heterogeneous clusters. In: Qin F, Barolli L, Cho SY, eds. Proc. of the ICPP. San Diego: IEEE Press, 2010. 653−662. [doi: 10.1109/ICPP.2010.73]

[44] Papagiannis A, Nikolopoulos DS. Rearchitecting MapReduce for heterogeneous multicore processors with explicitly managed memories. In: Qin F, Barolli L, Cho SY, eds. Proc. of the ICPP. San Diego: IEEE Press, 2010. 121−130. [doi: 10.1109/ICPP.2010.21]

[45] Jiang DW, Ooi BC, Shi L, Wu S. The performance of MapReduce: An in-depth study. PVLDB, 2010,3(1-2):472−483.

[46] Berthold J, Dieterle M, Loogen R. Implementing parallel Google Map-Reduce in Eden. In: Sips HJ, Epema DHJ, Lin HX, eds. Proc. of the Euro-Par. Delft: Springer-Verlag, 2009. 990−1002. [doi: 10.1007/978-3-642-03869-3_91]

[47] Verma A, Zea N, Cho B, Gupta I, Campbell RH. Breaking the MapReduce stage barrier. In: Moreira JE, Matsuoka S, Pakin S, Cortes T, eds. Proc. of the CLUSTER. Crete: IEEE Press, 2010. 235−244. [doi: 10.1109/CLUSTER.2010.29]

[48] Yang HC, Dasdan A, Hsiao RL, Parker DS. Map-Reduce-Merge simplified relational data processing on large clusters. In: Chan CY, Ooi BC, Zhou AY, eds. Proc. of the SIGMOD. Beijing: ACM Press, 2007. 1029−1040. [doi: 10.1145/1247480.1247602]

[49] Seo SW, Jang I, Woo KC, Kim I, Kim JS, Maeng S. HPMR: Prefetching and pre-shuffling in shared MapReduce computation environment. In: Rana O, Tang FL, Kosar T, eds. Proc. of the CLUSTER. New Orleans: IEEE Press, 2009. 1−8. [doi: 10.1109/CLUSTR.2009.5289171]

[50] Babu S. Towards automatic optimization of MapReduce programs. In: Kansal A, ed. Proc. of the ACM Symp. on Cloud Computing (SoCC). New York: ACM Press, 2010. 137−142. [doi: 10.1145/1807128.1807150]

[51] Olston C, Reed B, Srivastava U, Kumar R, Tomkins A. Pig Latin: A not-so-foreign language for data processing. In: Wang JTL, ed. Proc. of the SIGMOD. Vancouver: ACM Press, 2008. 1099−1110. [doi: 10.1145/1376616.1376726]

[52] Isard M, Budiu M, Yu Y, Birrell A, Fetterly D. Dryad: Distributed data-parallel programs from sequential building blocks. ACM SIGOPS Operating Systems Review, 2007,41(3):59−72. [doi: 10.1145/1272996.1273005]

[53] Isard M, Yu Y. Distributed data-parallel computing using a high-level programming language. In: Cetintemel U, Zdonik SB, Kossmann D, Tatbul N, eds. Proc. of the SIGMOD. Rhode Island: ACM Press, 2009. 987−994. [doi: 10.1145/1559845.1559962]

[54] Chaiken R, Jenkins B, Larson PÅ, Ramsey B, Shakib D, Weaver S, Zhou JR. SCOPE: Easy and efficient parallel processing of massive data sets. PVLDB, 2008,1(2):1265−1276. [doi: 10.1145/1454159.1454166]

[55] Condie T, Conway N, Alvaro P, Hellerstein JM, Gerth J, Talbot J, Elmeleegy K, Sears R. Online aggregation and continuous query support in MapReduce. In: Elmagarmid AK, Agrawal D, eds. Proc. of the SIGMOD. Indianapolis: ACM Press, 2010. 1115−1118. [doi: 10.1145/1807167.1807295]

[56] Thusoo A, Sarma JS, Jain N, Shao Z, Chakka P, Anthony S, Liu H, Wyckoff P, Murthy R. Hive a warehousing solution over a MapReduce framework. PVLDB, 2009,2(2):938−941.

[57] Ghoting A, Pednault E. Hadoop-ML: An infrastructure for the rapid implementation of parallel reusable analytics. In: Culotta A, ed. Proc. of the Large-Scale Machine Learning: Parallelism and Massive Datasets Workshop (NIPS 2009). Vancouver: MIT Press, 2009. 6.

[58] Yang C, Yen C, Tan C, Madden SR. Osprey: Implementing MapReduce-style fault tolerance in a shared-nothing distributed database. In: Li FF, Moro MM, Ghandeharizadeh S, Haritsa JR, Weikum G, Carey MJ, Casati F, Chang EY, Manolescu I, Mehrotra S, Dayal U, Tsotras VJ, eds. Proc. of the ICDE. Long Beach: IEEE Press, 2010. 657−668. [doi: 10.1109/ICDE.2010.5447913]

[59] Abouzeid A, Bajda-Pawlikowski K, Abadi DJ, Silberschatz A, Rasin A. HadoopDB: An architectural hybrid of MapReduce and DBMS technologes for analytical workloads. PVLDB, 2009,2(1):922−933.

[60] Abouzied A, Bajda-Pawlikowski K, Huang JW, Abadi DJ, Silberschatz A. HadoopDB in action: Building real world applications. In: Elmagarmid AK, Agrawal D, eds. Proc. of the SIGMOD. Indiana: ACM Press, 2010. 1111−1114. [doi: 10.1145/1807167.1807294]

[61] Friedman E, Pawlowski P, Cieslewicz J. SQL/MapReduce: A practical approach to self describing, polymorphic, and parallelizable user defined functions. PVLDB, 2009,2(2):1402−1413.

[62] Stonebraker M, Abadi D, deWitt DJ, Maden S, Paulson E, Pavlo A, Rasin A. MapReduce and parallel DBMSs: Friends or foes? Communications of the ACM, 2010,53(1):64−71. [doi: 10.1145/1629175.1629197]

[63] Dean J, Ghemawat S. MapReduce: A flexible data processing tool. Communications of ACM, 2010,53(1):72−77. [doi: 10.1145/1629175.1629198]

[64] Xu Y, Kostamaa P, Gao LK. Integrating hadoop and parallel DBMS. In: Elmagarmid AK, Agrawal D, eds. Proc. of the SIGMOD. Indianapolis: ACM Press, 2010. 969−974. [doi: 10.1145/1807167.1807272]

[65] Thusoo A, Shao Z, Anthony S, Borthakur D, Jain N, Sarma JS, Murthy R, Liu H. Data warehousing and analytics infrastructure at facebook. In: Elmagarmid AK, Agrawal D, eds. Proc. of the SIGMOD. Indianapolis: ACM Press, 2010. 1013−1020.

[66] Mcnabb AW, Monson CK, Seppi KD. MRPSO: MapReduce particle swarm optimization. In: Ryan C, Keijzer M, eds. Proc. of the GECCO. Atlanta: ACM Press, 2007. 177−185. [doi: 10.1145/1276958.1276991]

[67] Kang U, Tsourakakis CE, Faloutsos C. PEGASUS: A peta-scale graph mining system—Implementation and observations. In: Wang W, Kargupta H, Ranka S, Yu PS, Wu XD, eds. Proc. of the ICDM. Miami: IEEE Computer Society, 2009. 229−238. [doi: 10.1109/ICDM.2009.14]

[68] Kang S, Bader DA. Large scale complex network analysis using the hybrid combination of a MapReduce cluster and a highly multithreaded system. In: Taufer M, Rünger G, Du ZH, eds. Proc. of the Workshops and Phd Forum (IPDPS 2010). Atlanta: IEEE Presss, 2010. 11−19. [doi: 10.1109/IPDPSW.2010.5470691]

[69] Logothetis D, Yocum K. AdHoc data processing in the cloud. PVLDB, 2008,1(1):1472−1475. [doi: 10.1145/1454159.1454204]

[70] Olston C, Bortnikov E, Elmeleegy K, Junqueira F, Reed B. Interactive analysis of WebScale data. In: DeWitt D, ed. Proc. of the CIDR. Asilomar: 2009. https://database.cs.wisc.edu/cidr/cidr2009/Paper_21.pdf

[71] Böse JH, Andrzejak A, Högqvist M. Beyond online aggregation: Parallel and incremental data mining with online Map-Reduce. In: Tanaka K, Zhou XF, Zhang M, Jatowt A, eds. Proc. of the Workshop on Massive Data Analytics on the Cloud (WWW 2010). Raleigh: ACM Press, 2010. 3. [doi: 10.1145/1779599.1779602]

[72] Kumar V, Andrade H, Gedik B, Wu KL. DEDUCE: At the intersection of MapReduce and stream processing. In: Manolescu I, Spaccapietra S, Teubner J, Kitsuregawa M, Léger A, Naumann F, Ailamaki A, Ozcan F, eds. Proc. of the EDBT. Lausanne: ACM Press, 2010. 657−662. [doi: 10.1145/1739041.1739120]

[73] Abramson D, Dinh MN, Kurniawan D, Moench B, de Rose L. Data centric highly parallel debugging. In: Hariri S, Keahey K, eds. Proc. of the HPDC. Chicago: ACM Press, 2010. 119−129. [doi: 10.1145/1851476.1851491]

[74] Morton K, Friesen A, Balazinska M, Grossman D. Estimating the progress of MapReduce pipelines. In: Li FF, Moro MM, Ghandeharizadeh S, *et al*., eds. Proc. of the ICDE. Long Beach: IEEE Press, 2010. 681−684. [doi: 10.1109/ICDE.2010.5447919]

[75] Morton K, Balazinska M, Grossman D. ParaTimer: A progress indicator for MapReduce DAGs. In: Elmagarmid AK, Agrawal D, eds. Proc. of the SIGMOD. Indianapolis: ACM Press, 2010. 507−518. [doi: 10.1145/1807167.1807223]

[76] Lang W, Patel JM. Energy management for MapReduce clusters. PVLDB, 2010,3(1-2):129−139.

[77] Wieder A, Bhatotia P, Post A, Rodrigues R. Brief announcement: Modelling MapReduce for optimal execution in the cloud. In: Richa AW, Guerraoui R, eds. Proc. of the PODC. Zurich: ACM Press, 2010. 408−409.

[78] Zheng Q. Improving MapReduce fault tolerance in the cloud. In: Taufer M, Rünger G, Du ZH, eds. Proc. of the Workshops and PhD Forum (IPDPS 2010). Atlanta: IEEE Presss, 2010. 1−6. [doi: 10.1109/IPDPSW.2010.5470865]

[79] Groot S. Jumbo: Beyond MapReduce for workload balancing. In: Mylopoulos J, Zhou LZ, Zhou XF, eds. Proc. of the PhD Workshop (VLDB 2010). Singapore: VLDB Endowment, 2010. 7−12.

[80] Chatziantoniou D, Tzortzakakis E. ASSET queries: A declarative alternative to MapReduce. SIGMOD Record, 2009,38(2):35−41. [doi: 10.1145/1815918.1815926]

[81] Bu YY, Howe B, Balazinska M, Ernst MD. HaLoop: Efficient iterative data processing on large clusters. PVLDB, 2010,3(1-2): 285−296.

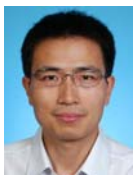
[82] Wang HJ, Qin XP, Zhang YS, Wang S, Wang ZW. LinearDB: A relational approach to make data warehouse scale like MapReduce. In: Yu JX, Kim MH, Unland R, eds. Proc. of the DASFAA. Hong Kong: Springer-Verlag, 2011. 306−320. [doi: 10.1007/978-3-642-20152-3_23]

附中文参考文献:

[1] 周傲英.数据密集型计算-数据管理技术面临的挑战.中国计算机学会通讯,2009,5(7):50−53.

覃雄派(1971－),男,广西百色人,博士,讲师,CCF 会员,主要研究领域为高性能数据库,大规模数据分析.

杜小勇(1963－),男,博士,教授,博士生导师,CCF 高级会员,主要研究领域为智能信息检索,高性能数据库,知识工程.



王会举(1979－),男,博士生,主要研究领域为云计算,高性能数据库.

王珊(1944－),女,教授,博士生导师,CCF 高级会员,主要研究领域为高性能数据库,知识工程,数据仓库,数据分析.