面向大数据场景的系统性能优化实践

王冀彬[1]，杨海龙[2]，冯凯[3]，孙欣[2]，张敏达[1]，雷克伦[2]，肖智文[1]，张逸飞[1]，吴佳熙[1]
1. 中国移动信息技术中心，北京 100033；
2. 北京航空航天大学，北京 100191；
3. 中移信息技术有限公司，广东 深圳 518048

摘要
在现有大规模分布式环境中，大数据应用的性能与计算效率仍有较大的提升空间。然而，在大规模环境中进行性能分析与优化需要大量领域专家。针对大数据应用中的性能优化问题，提出了一个通用的低效查询语句检测与优化流程，总结了4类显著影响大数据应用性能的低效行为，并针对每一类低效行为，提出了具体的优化策略。最后，通过实验评估验证了提出的优化方案在实际大规模集群中的有效性。

关键词
Hadoop；大数据系统；性能优化；调优工具

中图分类号：TP391 文献标志码：A doi: 10.11959/j.issn.2096-0271.2024049

System performance optimization practice for big data scenarios

WANG Jibin[1], YANG Hailong[2], FENG Kai[3], SUN Xin[2], ZHANG Minda[1], LEI Kelun[2], XIAO Zhiwen[1], ZHANG Yifei[1], WU Jiaxi[1]
1. China Mobile Information Technology Center, Beijing 100033, China
2. Beihang University, Beijing 100191, China
3. China Mobile Information Technology Co., Ltd., Shenzhen 518048, China

Abstract
In the existing large-scale distributed environments, there is still much room for improvement in the performance and computational efficiency of big data applications. However, performance analysis and optimization in large-scale environments requires a large number of human resources from domain experts. This paper proposes a general low-performance query statement detection and optimization process for performance optimization in big data applications, summarizes four types of low-performance behaviors that significantly affect the performance of big data applications, and proposes specific optimization strategies for each type of low-performance behavior. Finally, through experimental evaluation, the effectiveness of the optimization scheme in actual large-scale cluster is verified.

Key words
Hadoop, big data system, performance optimization, tuning tool

0 引言

随着科学应用和在线服务对大数据分析需求的快速增长，MapReduce[1]及其开源实现Hadoop被学术界和工业界广泛采用。MapReduce将计算分解为在多台机器上并行运行的小任务，并轻松扩展到由商用计算机组成的大型集群。Hadoop运行的作业可在内核上生成数百TB的数据。Hadoop提供了一个分布式文件系统（Hadoop distributed file system，HDFS）[2]和一个使用MapReduce分析和转换超大型数据集且内置YARN[3]资源管理器的框架。Hadoop的出现为构建大数据分析平台提供了高度可行的解决方案，促进了许多领域大数据应用的发展，如气候模型、生物信息学、天文学、工业系统和高能物理等。

Hadoop为分布式计算提供了高度可行的解决方案，但同时也暴露了性能和效率方面的缺点，包括低效查询语句导致的数据传输效率低下、不合理的任务配置导致的资源分配不均、数据倾斜导致的节点负载不均等，这些低效行为都有可能导致大数据系统出现性能问题。

大数据应用的性能对于服务提供商的用户体验和维护成本至关重要，进一步决定了相关产品的核心竞争力。在大规模分布式环境中进行调试和性能分析是一项具有挑战性的工作，在工业界需要专门的系统知识和大量实践经验才能获得适当的优化方案，开发人员需要性能分析工具来定位问题并优化性能。

为了缓解性能问题，研究人员将重点放在性能分析上，并根据分析结果采用各种方法来提高资源利用率。例如，文献[4-6]通过为Hadoop及其相关应用程序配置适当的参数，提高了集群资源利用率和应用程序性能。此外，文献[7-8]研究了不同的节能策略来提高目标应用的能源效率。然而，之前的研究没有针对Hadoop相关应用程序中的低效行为提出具体的分析和优化方案。本文的主要贡献如下。

- 提出了一个通用的大数据低效查询语句检测与优化流程，可针对不同平台上的SQL语句执行进行分析与优化；
- 总结了4类对大数据应用性能影响较大的低效行为，包括应用以及系统调优；
- 以数据倾斜与Tez任务配置不当两种低效行为为例，在大规模系统上进行案例研究，进一步讨论了针对同种低效行为的不同优化方案的效果，其中部分优化方案的性能提升显著。

1 研究背景

MapReduce[1]是Google提出的一种并行计算范式，由于其容错性、高可扩展性和编程简单等特点而被广泛应用。随着大规模数据集的分析需求的增长，MapReduce已成为大数据分析应用的主流编程模型。Hadoop是一个开源的分布式计算框架，是MapReduce最流行的开源实现，目前已被广泛部署在生产集群上。Hadoop对底层硬件系统进行了包装，并提供了高级编程接口，旨在按照MapReduce范式处理大规模数据集[6]。此外，Hadoop的设计具有良好的集群可扩展性，为应用程序提供了可靠的数据存储和移动服务。

Hadoop主要由两个核心组件组成，如**图1**所示。

- HDFS：HDFS是Hadoop的分布式文件存储系统。它将大规模数据集分成小块，并将这些块存储在集群中的多个节点上。HDFS的冗余数据副本策略，保证了数

据的高可用性和容错性。HDFS旨在支持大文件的高吞吐量和数据的快速读写。

- MapReduce：MapReduce是Hadoop的分布式数据处理框架。它允许用户编写map和reduce任务，用于分布式处理大规模数据。map任务将数据分解成键值对，reduce任务则执行聚合操作。MapReduce框架自动进行任务调度、容错处理和数据分发，使开发人员能够专注于编写数据处理逻辑。

与任何大型软件系统一样，这些数据平台也需要进行调整，从而充分发挥其功能。例如，Babu[9]和Jiang等人[10]的研究发现，Hadoop参数的调整可以显著减少Hadoop的执行时间。这些平台的调整还会直接影响其他质量属性，如可扩展性、可靠性和资源利用率。然而，每个平台都有数百个配置参数调节旋钮，这使得平台的调整烦琐且耗时。

作业在Hadoop上运行之前，有很大的参数配置空间，这些参数的设置会影响作业的执行，如内存的分配和使用、并发性、I/O的优化和网络带宽的使用。Hadoop中作业配置参数的子集见**表1**。Hadoop用户可以选择使用程序级界面或通过XML配置文件来设置这些参数。MapReduce框架的高级语言（如HiveQL和Pig）已经开发了自己的参数规范提示语法。超过190个参数控制着Hadoop中MapReduce作业的行

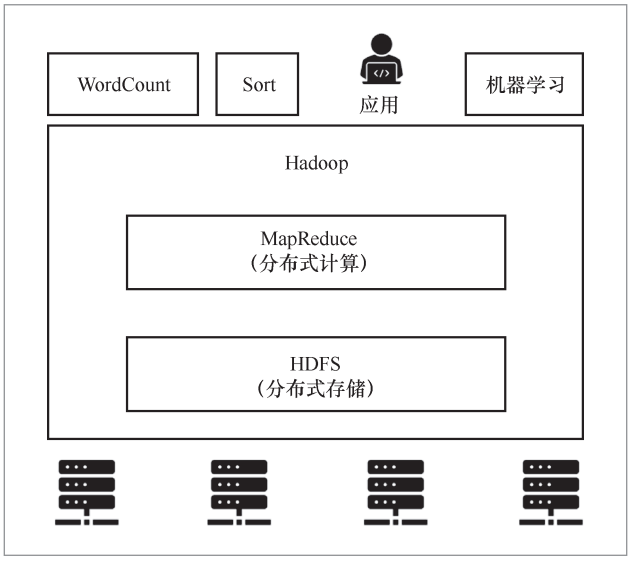


图1 Hadoop 生态系统

表1 Hadoop 中作业配置参数的子集

参数名称	参数说明	默认值	建议值
mapred.reduce.tasks	reducer作业数量	1	[5, 300]
io.sort.mb	用于排序键/值对的map端缓冲区的大小	100 MB	[100 MB, 200 MB]
io.sort.record.percent	io.sort.mb中专用于元数据存储的部分	0.05	[0.05, 0.15]
io.sort.factor	排序期间要合并的排序流的数量	10	[10, 500]
io.file.buffer.size	用于读/写（中间）序列文件的缓冲区大小	4 KB	32 KB
mapred.child.java.opts	所有mapper和reducer任务的Javacontrol选项	−Xmx200 m	−Xmx[200 m, 300 m]
mapred.job.shuffle.input.buffer.percent	用于在shuffle过程中缓存map输出的reducer任务的heap	0.7	0.7, 0.8
mapred.job.shuffle.merge.percent	mapred.job.shuffle.input.buffer.percent的使用阈值，用于在复制map输出的同时触发reduce侧合并	0.66	0.66, 0.8
mapred.inmem.merge.threshold	用于内存合并的另一个reduce侧触发器，为0时关闭	1 000	0
mapred.job.reduce.input.buffer.percent	用于在执行reduce时缓存map输出的reducer任务的heap	0	0, 0.8
dfs.replication	Hadoop的HDFS文件系统中的块复制参数	3	2
dfs.block.size	HDFS块大小（每个mapper任务处理的数据大小）	64 MB	128 MB

为，超过25个参数的设置可能会对工作性能产生显著影响。不少参数表现出与其他参数的强烈性能交互作用。

尽管Hadoop在大数据处理方面具有巨大的潜力，但它也面临一些挑战，如性能调优、集群管理、数据安全性和复杂性。为了更好地发挥Hadoop的潜力，研究人员和工程师不断开发和改进相关工具和技术。

2 相关工作

一些研究通过为Hadoop及其相关应用程序选择适当的参数配置来提高集群资源利用率和应用程序性能。文献[4]通过自适应模型和自适应候选生成方法，快速、准确地自动调整大数据Spark[11]应用程序的配置。文献[12-14]利用机器学习的方法来调整Hadoop和Spark。文献[15]使用基于高斯帕累托的多目标优化方法来调整Spark的参数，由3个适应度函数提供一组最优解决方案。文献[16]通过Adaboost算法比较各种性能模型，从而找到Spark的最佳设置。文献[17]设计并实现了一个基于强化学习的Spark配置参数优化器。文献[18]提出一种基于微操作重构的Hadoop参数自动调优方法，该方法通过建立比阶段更细粒度的微操作模型来刻画参数和单次微操作执行时间的关系，再基于运行原理对微操作模型进行组合得到阶段执行时间和参数的关系,在此基础上应用不同算法找出优化参数。文献[19]提出了一个云边端协同的大数据管理框架，该框架通过边缘计算来优化数据的存储、处理和传输，以适应元宇宙的分布式和动态特性。

此外，一些研究探索了不同的节能策略来提高目标应用的能源效率。文献[7]重点关注MapReduce（Hadoop）集群的关键功率峰值问题，提出了一种自适应方法，即在给定的功率预算内调节功率峰值以增加机器数量或降低功率预算。文献[8]采用节能方案，包括能源感知集群节点管理、能源感知数据管理、能源感知资源分配和能源感知任务调度，从而提高能源效率。这些研究工作通过大数据平台的宏观调控进行性能调优。文献[20]提出了一种针对混合工作负载场景的云服务器节能优化方法，根据实时工作负载的运行状态，动态地对CPU频率和系统内核参数进行联合调优。然而，很少研究针对Hadoop相关应用程序中的低效行为提出具体分析和优化方案。

针对数据查询效率问题，文献[21]提出一种优化的Hadoop数据放置策略，在满足HDFS默认数据放置算法的基本规则的前提下，通过分析数据块的需求量重新计算数据块的放置位置，将需求量最多的数据转移到能够最快处理它们的节点上，从而提升集群性能。文献[22]针对长时间序列、多站点和多气象要素的大量查询需求，重新设计了数据ETL流程，构建Parquet格式数据集并完成HDFS转换存储，嵌入Spark的广播变量，优化Spark集群的执行参数，提高了集群的处理并行度和SparkSQL的关联查询效率。文献[23]提出一种基于HiveSQL的增加任务并行度与建立中间表组合的优化查询方法，针对在某一个计算中处理多个指标的SQL，通过聚合SQL中相同的代码块提取出中间指标、建立中间表，这种方法减少了代码冗余，缩短了任务执行时长。

然而，本文的研究侧重于结合实际的Hadoop调试经验，即针对Hadoop相关应用程序中参数配置或SQL查询过程中的低效行为，提出了一种SQL语句瓶颈检测与优化流程来定位问题，并提供避免低效行为的快速优化方案。

3 查询语句与大数据系统的低效行为

大数据系统的低效行为是指，在处理大规模数据时，系统出现的一系列不利于系统性能和资源利用的情况，包括低效查询语句导致的数据传输效率低下、细碎文件查询导致的数据管理开销上升、中间结果文件过大导致的存储开销过大、数据倾斜严重导致的节点负载不均、不合理的任务配置导致的资源分配不均等。低效行为会导致任务执行速度下降，资源利用率低，系统响应时间延迟长，从而影响系统整体效率和性能。本节将从上述5个方面介绍大数据系统低效行为以及优化方法。

3.1 SQL语句瓶颈检测与优化流程

SQL语句瓶颈检测与优化流程涵盖了对用户反馈的SQL提交或执行缓慢问题的细致分析。这些问题的根源多种多样，比如底层主机的CPU、I/O资源信息，MySQL元数据服务的负载，HDFS、YARN以及Hive的负载情况，每一个环节都可能导致Hive的SQL任务执行缓慢。典型的排查思路包括对Beeline、HS2和Metastore进程的jstack分析，通过确认是否存在等待或阻塞的线程来确定问题的根源。例如HDFS的RPC延迟过高，可以进一步查看HDFS监控指标，确认近期RPC统计是否波动较大以及小文件数量是否激增。这一细致的排查流程有利于SQL语句性能问题的快速定位和解决。

3.2 细碎文件查询

大数据系统中的细碎文件查询是指处理大量小文件时产生的低效行为。大数据系统中出现细碎文件查询行为的原因包括数据生成方式导致产生大量小文件、任务调度策略导致的任务分配不合理、数据写入方式带来的频繁文件操作、存储策略和数据处理方式引发的分片和存储不合理以及系统配置管理方面的不足。这些因素共同导致了大量小文件存储在系统中，增加了元数据操作、数据定位开销和网络开销，影响了系统的整体性能和效率。细碎文件查询会增加大量元数据操作，因为每个文件都需要一定量的元数据来描述其位置、大小和权限等信息。此外，由于细碎文件分布在不同的位置，查询时需要花费更多时间来定位和读取这些文件，这增加了I/O开销和查询时间。大量小文件占用了系统的存储空间和内存资源，导致资源浪费、任务的并行度受限，因为每个文件都需要由一个任务来处理，这可能导致作业执行时间变长。在分布式环境中，查询大量小文件还会增加网络开销，因为文件分布在不同节点上，需要通过网络传输数据，这会增加网络延迟和带宽占用。

针对小文件问题，细碎文件查询优化分为查询时小文件的虚拟合并和结果数据写入HDFS的物理合并两个方面。

针对小文件查询问题，用户通过其他组件将数据流写入Hive表的分区，造成分区中有很多KB或MB级别的小文件。如果不开启参数优化，每个文件都需要单独启动一个任务进行处理。这种情况不仅会导致YARN资源浪费，任务的启动和关闭还会引入额外的开销，导致任务执行速度极其缓慢。为了解决这些问题，用户需要设置合并文件参数，使多个小文件共用一个任务进行处理，减少资源消耗。

针对结果小文件写入问题，用户使用动态分区INSERT对Hive表进行数据写入，默认情况下分区数据会产生很多小文件，小文件过多会给底层HDFS存储带来负载，对查询也不友好。因此，用户需要设置小文件合

并参数，在SQL任务的最后阶段启动一个合并任务，对分区的数据进行合并。

3.3 庞大的中间结果文件

大数据处理过程通常包含多个阶段的数据处理操作，每个阶段都会生成一些中间结果。如果每个阶段都将中间结果写入磁盘，会产生大量的中间结果文件，尤其是在数据转换、过滤、聚合等操作中。在MapReduce任务的执行过程中，reduce任务负责处理来自多个map任务的输出并生成中间结果。reduce任务的输出数据量过大或数据分区不均匀，可能会导致生成的中间结果文件也非常庞大，引起存储开销的增加、系统性能的下降。因此，需要采取合适的策略来减少中间结果文件的产生和存储，提高系统的效率和性能。

为了解决庞大的中间结果文件的问题，本文采用了一项有效的优化策略。传统的FileOutputCommitter算法在处理结果文件时采用串行的move操作，耗时且易受namenode性能和连接数的影响。为了优化这一过程，本文强烈推荐将mapreduce.fileoutputcommitter.algorithm.version参数设置为2（默认为1），将中间结果文件以目录形式存储。这样的优化策略仅需一次move操作，极大地提升了commit操作的性能。

此优化参数为应用端可配置的参数，用户只需在mapred-site配置文件中对该参数进行修改。测试发现，原本需要执行40 min左右的流程，经过该优化策略后，其执行时间减少至20~30 min，大幅提升了整体流程的执行效率。这一优化方案在实践中表现出色，支持大规模任务的处理。

3.4 数据倾斜

数据倾斜主要源于数据分布不均匀、数据处理操作不平衡、哈希函数冲突、数据倾斜的累积效应以及数据特征变化等。数据倾斜问题通常在MapReduce任务的reduce阶段显现，在进行join和group by等聚合操作时尤为明显。在这些场景中，某些键（key）对应的记录数远远超过其他键，导致部分reduce任务需要处理大量数据，导致负载不均衡，某些reduce任务的处理速度明显变慢，甚至可能比其他任务慢几个数量级。这种情况会造成系统资源的浪费，因为大量资源被用于处理少数几个任务，而其他任务可能处于空闲状态。此外，由于数据倾斜导致的处理不均衡，整个任务的完成时间会延长，影响系统的整体性能。

为解决数据倾斜问题，可从两个方面入手：一是在业务侧进行数据预处理，如在SQL中过滤掉可能导致数据倾斜的字段的空值；二是在Hive组件侧进行SQL参数优化，特别是在涉及group by和join的SQL中，使用专门的数据倾斜优化参数。

针对group by倾斜优化参数，可通过开启map阶段预聚合和启动额外任务将大key随机分配到多个reduce进行优化。针对join倾斜优化参数，可开启map join和启动map join任务对大key进行计算。同时，建议对join操作开启map join优化参数以加速SQL执行，减少I/O和网络资源的消耗。在设置map join优化参数时，需考虑小数据量表的阈值。这些简洁的优化方法有助于提升SQL执行效率，减少资源消耗，从而优化整体的计算性能。

3.5 不合理的任务配置

大数据系统中不合理的任务配置通常在任务调度和资源分配阶段出现。这种情况可能是由任务调度策略不合理、资源分配不均衡或者任务设计不合理等

原因导致的。任务调度策略不合理可能导致某些节点负载过重，而其他节点处于空闲状态，导致资源利用不均衡。资源分配不均衡也可能导致部分节点资源过多，而其他节点资源不足，影响系统整体性能。

在解决YARN上的任务偶发缓慢的问题时，笔者发现Tez任务占用了大量的网络流量，导致数据节点的读写性能受限。通过进一步的分析发现，问题的本质在于HQL中使用了9个CTE表达式，其中包含了大量的join和全表扫描操作。在Tez任务的shuffle过程中，这些操作导致大量的数据需要通过网络传输，使主机网络资源的利用出现瓶颈。

为了解决这一问题，本文提出了一系列优化方案。首先，建议对SQL进行拆分，将大量的子查询拆分成独立的步骤执行，以解决Tez任务瞬时占满主机网络资源的问题。其次，减少map和reduce的数量，即减少任务的并发性，可以缓解网络满载的情况。如果对SQL的延时不是特别敏感，还可以考虑切换到MapReduce任务进行慢跑，以减轻Tez任务的压力。

在作业提交时，本文建议根据实际内存消耗配置任务使用内存数，从而优化作业占用的容器资源数，以节约集群资源。这些详细而有针对性的优化策略，可以提高作业的执行效率，降低对集群资源的消耗。

4 评估实验

4.1 实验配置

本文在中国移动一个包含1 124节点的真实集群上进行实验，集群总内存资源共337.81 TB，集群中单节点的软硬件配置见表2。具体而言，每个节点CPU为Kunpeng 920，操作系统为基于openEuler的中国移动自研操作系统bigcloud linux 20.12，内存为384 GB。本文选用的工作负载数据库为中国移动的基站拉链表，表数据为各省上传的数据，记录了用户在进入某个基站后产生的信息，包括手机号、国家编码、事件的类型、2G/3G/4G类型、进入基站的时间和基站的位置等。本节的案例研究通过Hadoop和Hive引擎运行SQL，在此基站拉链表中进行查询。选取的SQL语句属于业务逻辑的关键部分，被频繁执行，执行频率约为每天每小时一次，统计信息用于提升中国移动为全国各地生活或出行的用户服务的质量。具体而言，数据倾斜实验在全集群规模上进行，而由于Tez实验任务队列占满集群资源对集群的影响较大，选择了在规模为全集群10%的小集群（112节点，总内存资源为33.7 TB）上进行。由于集群规模较大，不可避免会存在性能波动，为了消除性能波动对实验的影响，案例研究的每个实验配置在集群中运行了5次，以5次实验结果的平均值为最终结果。

4.2 低效行为验证性实验

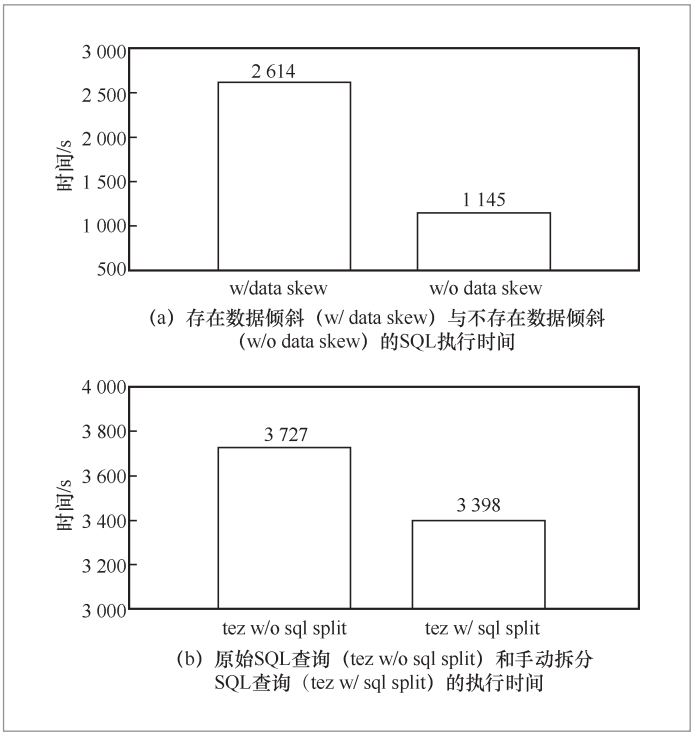
大数据场景下的数据倾斜问题通常由输入数据的分布不均匀引起。输入数据的不当分布可能会导致某些节点或任务分配

表2 单节点硬件与软件配置

配置项	配置详情
CPU	Kunpeng 920
内存	384 GB
操作系统	bclinux20.12
JDK	1.8.0_242
Hive	3.3.0
Hadoop	3.3.0

到的数据量明显多于其他节点或任务，导致计算资源利用不均衡、任务执行时间延长，甚至系统性能严重下降。在SQL语句的执行过程中，某些键值聚合的数据量可能远大于其他键值，使得某些节点可能处于空闲状态，而另一些节点超负荷工作，进一步导致关键路径变长，整体执行时间增加。为了展示数据倾斜对性能的影响，本文在存在数据倾斜和均匀分布的输入数据上执行同一查询，对比二者的性能差异。如**图2（a）**所示，在不存在数据倾斜的输入数据上执行SQL时间仅为存在数据倾斜的输入数据的43.8%。值得注意的是，在该实验中，不存在数据倾斜的数据量为3.9 TB，远大于存在数据倾斜的数据量（801.5 GB）。

对于Tez任务不合理分配的问题，在同一输入数据上分别进行原始SQL查询以及手动拆分SQL查询（保证每个语句的任务量都不会造成严重的Tez任务队列拥塞）。如**图2（b）**所示，手动拆分SQL的查询时间仅为原始SQL的91.2%。



（a）存在数据倾斜（w/ data skew）与不存在数据倾斜（w/o data skew）的SQL执行时间

（b）原始SQL查询（tez w/o sql split）和手动拆分SQL查询（tez w/ sql split）的执行时间

图 2 SQL 执行时间

综上所述，数据倾斜问题以及Tez任务的不合理分配会显著影响大数据任务的性能。4.3节将针对这两种低效行为进行案例分析，给出调优方案并进行效果验证。

4.3 案例研究：数据倾斜

数据倾斜情况可能发生在使用group by和join操作的SQL语句中。因此，对涉及这两种操作的SQL均进行了案例研究。

4.3.1 group by

数据倾斜实验group by的SQL查询语句为“select province, count(1) from ods.to_d_evnt_procedure_cnt group by province”。在集群中针对无参数调优、设置hive.groupby.skewindata=true、设置hive.map.aggr=true以及同时设置hive.groupby.skewindata=true和hive.map.aggr=true这4种情况分别进行实验，记录SQL查询的端到端执行时间。实验结果如**图3**所示，hive.groupby.skewindata和hive.map.aggr参数对缓解数据倾斜非常有效，其中hive.map.aggr参数的效果更好，而hive.groupby.skewindata和hive.map.aggr参数结合使用带来的优化效果更好，其执行时间仅为不进行参数调优的原始版本的7.9%。

4.3.2 join

数据倾斜实验join的SQL查询语句为“select imsi from (select imsi from ods.to_d_evnt_procedure_cnt where

day = 20231008 limit 1000000) as a join (select imsi from ods.to_d_evnt_procedure_cnt where day = 20231009 limit 10000) as b on a.imsi = b.imsi”。在集群中针对无参数调优、设置hive.auto.convert.join=true、设置hive.optimize.skewjoin=true以及同时设置hive.auto.convert.join=true和hive.optimize.skewjoin=true这4种情况分别进行实验，记录SQL查询的端到端执行时间。实验结果如**图4**所示，hive.auto.convert.join和hive.optimize.skewjoin参数对缓解数据倾斜的效果并不好，其执行时间甚至比不进行参数调优的原始版本的还长。

由上述实验可知，在Hive引擎上运行SQL查询语句，当查询涉及group by操作时，设置hive.groupby.skewindata和hive.map.aggr参数有助于缓解数据倾斜，但当查询涉及join操作时，设置hive.auto.convert.join和hive.optimize.skewjoin参数并不能缓解数据倾斜。

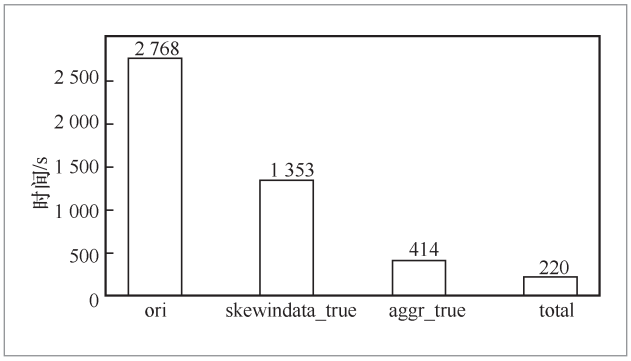


图 3　group by 的查询时间

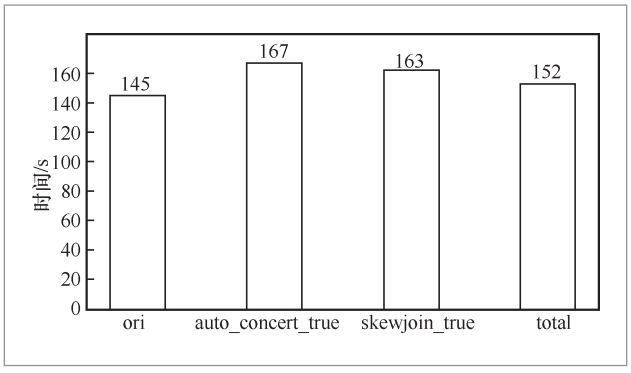


图 4　join 的查询时间

4.4 案例研究：Tez

Tez任务队列满载实验的SQL查询语句为“select province, count(1) from ods.to_d_evnt_procedure_cnt group by province”。在集群中针对无参数调优、设置tez.grouping.max-size=52 428 800（集群默认设置的最小值，用grouping_max_size1表示）、设置tez.grouping.max-size=52 428 800×4=209 715 200（用grouping_max_size2表示）以及设置hive.exec.reducers.max=500（集群默认设置为1 009，当前测试设置的500，用reducers.max表示）这4种情况分别进行实验，记录SQL查询的端到端执行时间。实验结果如**图5**所示，调节tez.grouping.max-size参数可以缓解Tez任务队列满载的问题，调大tez.grouping.max-size的查询时间仅为不进行参数调优的原始版本的6.4%。此外，调小hive.exec.reducers.max也可以缓解Tez任务队列满载的问题，其查询时间仅为不进行参数调优的原始版本的执行时间19.9%。

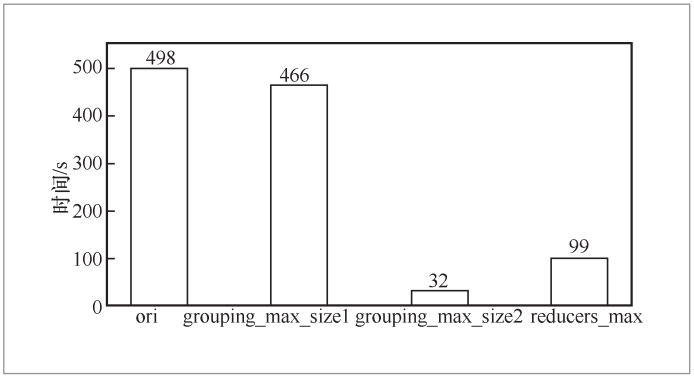


图 5　Tez 任务队列满载的查询时间

5 结束语

本文系统地探讨了大数据系统性能优化实践，重点关注SQL语句低效行为的检测与优化、细碎文件查询优化、庞大的中间结果文件优化、数据倾斜和不合理的任务配置等方面。首先，本文提出了一个通用的大数据低效查询语句检测与优化流程，该流程能够有效地分析和优化不同平台上的SQL语句执行，为性能调优提供了有力的工具和方法。其次，总结了4类对大数据应用性能影响显著的低效行为，并针对每种行为提出了具体的解决思路和方法论。对数据倾斜和Tez等案例的研究，验证了本文提出的优化方案的实用性和可行性。

参考文献:

[1] DEAN J, GHEMAWAT S. MapReduce: simplified data processing on large clusters[J]. Communications of the ACM, 2008, 53: 107–113.

[2] SHVACHKO K, KUANG H R, RADIA S, et al. The hadoop distributed file system[C]//Proceedings of the 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST). Piscataway: IEEE Press, 2010: 1–10.

[3] VAVILAPALLI V K, MURTHY A C, DOUGLAS C, et al. Apache Hadoop YARN: yet another resource negotiator[C]//Proceedings of the 4th annual Symposium on Cloud Computing. New York: ACM, 2013: 1–16.

[4] LIN C, ZHUANG J Q, FENG J D, et al. Adaptive code learning for spark configuration tuning[C]//Proceedings of the 2022 IEEE 38th International Conference on Data Engineering (ICDE). Piscataway: IEEE Press, 2022: 1995–2007.

[5] LU J H, CHEN Y X, HERODOTOU H, et al. Speedup your analytics: automatic parameter tuning for databases and big data systems[J]//Proceedings of the VLDB Endowment, 2019, 12(12): 1970–1973.

[6] WU D L, GOKHALE A. A self-tuning system based on application profiling and performance analysis for optimizing Hadoop MapReduce cluster configuration[C]//Proceedings of the 20th Annual International Conference on High Performance Computing. Piscataway: IEEE Press, 2013: 89–98.

[7] ZHU N, RAO L, LIU X, et al. Taming power peaks in mapreduce clusters[J]. ACM SIGCOMM Computer Communication Review, 2011, 41(4): 416–417.

[8] WU W T, LIN W W, HSU C H, et al. Energy-efficient hadoop for big data analytics and computing: a systematic review and research insights[J]. Future Generation Computer Systems, 2018, 86: 1351–1367.

[9] BABU S. Towards automatic optimization of MapReduce programs[C]//Proceedings of the 1st ACM symposium on Cloud computing. New York: ACM, 2010: 137–142.

[10] JIANG D W, OOI B C, SHI L, et al. The performance of MapReduce[J]. Proceedings of the VLDB Endowment, 2010, 3(1/2): 472–483.

[11] ZAHARIA M, CHOWDHURY M, FRANKLIN M J, et al. Spark: cluster computing with working sets[C]//Proceedings of the 2nd USENIX

Conference on Hot Topics in Cloud Computing. Berkeley: USENIX Association, 2010.

[12] YIGITBASI N, WILLKE T L, LIAO G D, et al. Towards machine learning-based auto-tuning of MapReduce[C]// Proceedings of the 2013 IEEE 21st International Symposium on Modelling, Analysis and Simulation of Computer and Telecommunication Systems. Piscataway: IEEE Press, 2013: 11-20.

[13] WANG G L, XU J G, HE B. A novel method for tuning configuration parameters of spark based on machine learning[C]//Proceedings of the 2016 IEEE 18th International Conference on High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS). Piscataway: IEEE Press, 2016: 586-593.

[14] DE OLIVEIRA D, PORTO F, BOERES C, et al. Towards optimizing the execution of spark scientific workflows using machine learning-based parameter tuning[J]. Concurrency and Computation: Practice and Experience, 2021, 33(5): e5972.

[15] ÖZTÜRK M M. Tuning parameters of Apache Spark with Gauss-Pareto-based multi-objective optimization[J]. Knowledge and Information Systems, 2024, 66(2): 1065-1090.

[16] CHENG G L, YING S, WANG B M, et al. Efficient performance prediction for apache spark[J]. Journal of Parallel and Distributed Computing, 2021, 149: 40-51.

[17] HUANG X, ZHANG H, ZHAI X M. A novel reinforcement learning approach for spark configuration parameter optimization[J]. Sensors, 2022, 22(15): 5930.

[18] 李耘书, 滕飞, 李天瑞. 基于微操作的Hadoop参数自动调优方法[J]. 计算机应用, 2019, 39(6): 1589-1594.
LI Y S, TENG F, LI T R. Microoperation-based parameter auto-optimization method of Hadoop[J]. Journal of Computer Applications, 2019, 39(6): 1589-1594.

[19] 朱锐, 王宏志, 崔双双, 等. 面向元宇宙的云边端协同大数据管理[J]. 大数据, 2023, 9(1): 63-77.
ZHU R, WANG H Z, CUI S S, et al. Cloud-edge-end collaborative big data management for metaverse[J]. Big Data Research, 2023, 9(1): 63-77.

[20] LIANG J C, LIN W W, XU Y G, et al. Energy-aware parameter tuning for mixed workloads in cloud server[J]. Cluster Computing, 2023: 1-17.

[21] 黄志, 苏传程, 苏晓红. 大数据环境下Spark性能优化分析研究与应用[J]. 气象科技, 2022, 50(1): 51-58.
HUANG Z, SU C C, SU X H. Research and application of spark performance optimization analysis in big data environment[J]. Meteorological Science and Technology, 2022, 50(1): 51-58.

[22] 吴岳. 一种优化的Hadoop数据放置策略[J]. 软件工程, 2023, 26(7): 44-47.
WU Y. An optimized hadoop data placement strategy[J]. Software Engineering, 2023, 26(7): 44-47.

[23] 郑灵逸, 李擎. 一种基于HiveSQL的增加任务并行度与建立中间表组合的优化查询方法[J]. 现代计算机, 2021, 27(36): 55-59.
ZHENG L Y, LI Q. An optimization query method based on HiveSQL to increase task parallelism and build intermediate table combination[J]. Modern Computer, 2021, 27(36): 55-59.

作者简介

王冀彬（1980– ），男，中国移动信息技术中心高级工程师、大数据事业群总经理，主要研究方向为大数据、数据分析。

杨海龙（1985– ）男，博士，北京航空航天大学教授，主要研究方向为高性能计算、分布式和并行计算、计算机系统结构、深度学习编译优化技术。

冯凯（1977– ），男，中移信息技术有限公司高级工程师、项目总监，主要研究方向为大数据运维、数据分析。

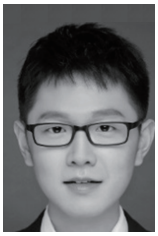
孙欣（2000– ），女，北京航空航天大学硕士生，主要研究方向为计算机系统结构、性能分析工具。

张敏达（1998– ），女，中移信息技术有限公司项目经理，主要研究方向为大数据运维、数据分析。

雷克伦（2000– ），男，北京航空航天大学博士生，主要研究方向为高性能计算、性能分析工具和编译优化。

肖智文（1993– ），男，博士，中国移动信息技术中心中级工程师、项目经理，主要研究方向为大数据、机器学习、云计算。

张逸飞（1996– ），男，博士，中国移动信息技术中心中级工程师、项目经理，主要研究方向为大数据、机器学习、云计算。



吴佳熙（1993– ），男，博士，中国移动信息技术中心中级工程师、项目经理，主要研究方向为大数据、云计算。

收稿日期： 2024-04-18

通信作者： 杨海龙，hailong.yang@buaa.edu.cn

基金项目： 国家重点研发计划项目（No.2023YFB4503100）；国家自然科学基金项目（No.62322201，No.62072018，No.U23B2020，No.U22A2028）；中央高校基本科研业务费专项资金资助（No.YWF-23-L-1121）

Foundation Items: The National Key Research and Development Program of China(No.2023YFB4503100), The National Natural Science Foundation of China(No.62322201, No.62072018, No.U23B2020, No.U22A2028), The Fundamental Research Funds for the Central Universities(No.YWF-23-L-1121)