文章编号:1671-4598(2018)12-0260-05 DOI:10.16526/j.cnki.11—4762/tp.2018.12.057 中图分类号:P474.2 文献标识码:A

基于 Spark 的飞机试验数据预处理技术研究

李利荣[1]，孙立伟[1]，杨 浩[2,3,4]，王晓栋[2,3,4]，房红征[2,3,4]
(1. 上海民用飞机健康监控工程技术研究中心，上海 200241；2. 北京航天测控技术有限公司，北京 100041；3. 北京市高速交通工具智能诊断与健康管理重点实验室，北京 100041；4. 轨道交通装备全寿命周期状态监测与智能管理技术与应用北京市工程实验室，北京 100041)

摘要：飞机飞行试验是飞机交付运营前的必要环节，在飞机飞行试验过程中会产生大量的试验数据，这些数据对飞机的研制、定型、改进具有重要意义；传统的试验数据预处理技术多采用单机工作模式，无法快速处理海量试验数据，鉴于此，对基于 spark 分布式内存计算技术进行了研究；通过预先剔除参数组中不存在于当前数据文件中的参数群，减少分布式集群间的数据通讯与航空领域 429 协议、232 协议、664 协议的数据预处理时间，提高飞行试验的效率；最后，选择 30 GB 飞行试验数据作为实验对象进行验证，结果表明，该方法有效地提高了数据解析效率，克服了传统数据处理方式效率低下，单个机器内存和 CPU 等硬件条件不足的问题。

关键词：飞行试验；数据预处理；分布式技术；内存计算

Research on Preprocessing Technology of Aircraft Experimental Data Based on Spark

Li Lirong[1], Sun Liwei[1], Yang Hao[2,3,4], Wang Xiaodong[2,3,4], Fang Hongzheng[2,3,4]
(1. Shanghai Engineering Research Center of Civil Aircraft Health Monitoring, Shanghai 200241, China; 2. Beijing Aerospace Measure & Control Corp. Ltd, Beijing 100041, China; 3. Beijing Key Laboratory of High-speed Transport Intelligent Diagnostic and Health Management, Beijing 100041, China; 4. Beijing Engineering Laboratory of Rail Transportation Equipment Life Cycle Condition Monitoring and Intelligent Management Technology and Application)

Abstract: Flight test is an essential part of the aircraft before delivery. It also plays an important role in the development of China's aviation industry. During the flight test, a large amount of test data is generated, which is important for the development, stereotypes and improvements of the aircraft. The traditional experimental data preprocessing technology mostly adopts the single machine working mode. This paper studies a new experimental data preprocessing technology, based on spark distributed memory computing technology by removing the parameter group that does not exist in current data file and reducing the data communication between distributed clusters to solve the high-speed processing of massive test data transforming by ARINC429、ARINC232 and ARINC 664. Finally, 30 GB flight test data is selected as the experimental object for verification. Compared with the traditional data processing method, the proposed method effectively improves the data analysis efficiency and overcomes the problems of traditional data processing methods, such as low efficiency and hardware problems.

Keywords: flight test; data preprocessing; distributed technology; memory computing

0 引言

随着计算机和测控技术的发展，试飞试验数据呈现出参数多（上万个）、数据量大（上百 TB）、参数类型多样化等特点[1]。飞行试验数据是完成新机定型、鉴定的主要依据，同时也是支撑航空科技发展的宝贵资料。能否将大量飞行试验数据有效管理并使用起来，这对我国航空事业的发展具有重要的现实意义。

飞行试验是在真实飞行条件下进行的科学研究和产品试验。它是航空科技发展的重要手段，是航空产品研制和鉴定的必须环节，是为用户摸索和积累经验的有效途径。试飞任务中的数据信息管理是一个庞大而复杂的系统工程，目前国内传统的试飞数据处理模式是：一型飞机、一个团队、一套数据格式及处理程序，即由专业人员使用自行开发的软件，对测试数据进行处理，中间结果及最终报告数据以操作系统文件的形式保存；有关数据处理的文档人工处理。这种处理模式的缺点是：1）数据和程序的继承性差，共享困难；2）低水平重复，不利于积累以往的经验；3）处理效率低，容易出错；4）处理周期长，不利于快速做出下一步的试飞决策。

本文研究基于 Spark 的飞机试验数据处理技术，将改变原有的“特定飞机，专用软件”的研制模式，将复杂、繁琐的数据管理工作抽象出来，使工程师能够将更多的精力

收稿日期:2018-08-23； **修回日期**:2018-09-21。

作者简介:李利荣(1989-)，女，工程师，主要从事飞机健康管理等工作方向的研究。

集中到数据的处理和分析工作中去。一方面，提炼共同要求，统一飞行试验信息化标准，完成系统的统一规划与部署，避免试飞信息化工作出现效率低下和资源浪费等问题。另一方面，通过建立健全和完善型号信息系统、数据与信息以及数据处理软件的标准和规范，提高信息系统和软件的可重复性，避免低水平的重复开发。

1　SPARK 内存计算框架

目前，应用最为广泛的大数据技术是 hadoop 以及其分布式架构 map－reduce。Hadoop MapReduce 采用 Master/slave 结构。只要按照其编程规范，只需要编写少量的业务逻辑代码即可实现一个强大的海量数据并发处理程序。其核心思想是分而治之。Mapper 负责把一个复杂的业务，任务分成若干个简单的任务分发到网络上的每个节点并行执行，产生的结果由 Reduce 进行汇总，输出到 HDFS 中，大大缩短了数据处理的时间开销。MapReduce 以一种可靠且容错的方式对大规模集群海量数据进行数据处理，数据挖掘，机器学习等方面的操作[2-7]。

尽管该架构具备开源分布式的特点且应用范围广泛，但在大量数据离线计算过程中，map－reduce 存在着大量的硬盘读写，这造成计算效率很低。Spark 是用于大数据处理快速而通用的引擎，其采用分布式内存计算方式，将过程数据保存在内存中，减少了由于磁盘交互产生的 I/O，从而提高数据计算效率[8-10]。

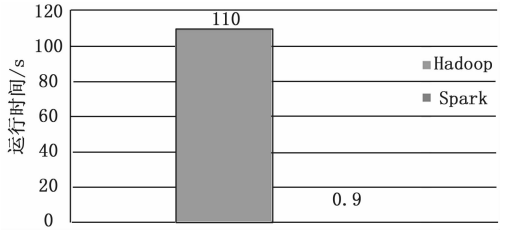

图 1　Hadoop 和 Spark 的逻辑回归

Spark 拥有 Hadoop MapReduce 所具有的优点，但和 MapReduce 的最大不同之处在于 Spark 是基于内存的迭代式计算[11]。Spark 的 Job 处理的中间输出结果可以保存在内存中，从而不再需要读写 HDFS，除此之外，Map－Reduce 在计算中只有两个阶段，即 map 和 reduce。而在 Spark 的计算模型中，可以分为 n 阶段，因为它内存迭代式的，我们在处理完一个阶段以后，可以继续往下处理很多个阶段，而不只是两个阶段。因此，Spark 能更好地适用于数据挖掘与机器学习等需要迭代的 MapReduce 的算法。不仅实现了 MapReduce 的算子 map 函数和 reduce 函数及计算模型，还提供更为丰富的算子，如 filter、join、groupByKey 等，是一个用来实现快速而同用的集群计算的平台。

此外，Spark 应用程序还离不开 SparkContext 和 Executor。Executor 负责执行任务，运行 Executor 的机器称为 Worker 节点，SparkContext 由用户程序启动，通过资源调度模块实现与 Executor 通信。图 2 为 Spark 的操作机制。

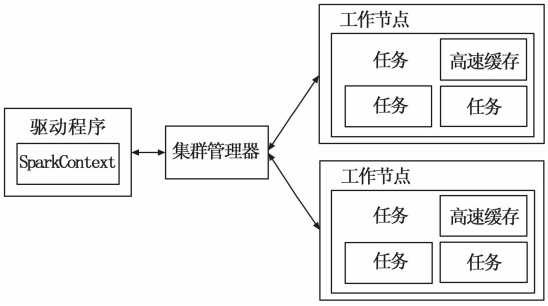

图 2　Spark 操作机制

集群管理器主要控制整个集群，监控各工作节点。工作节点负责控制计算节点，启动执行器和驱动程序。执行器是应用程序运行在工作节点上的一个进程。

RDD（Resilient Distributed Datasets）[1]，弹性分布式数据集，是分布式内存的一个抽象概念，RDD 提供了一种高度受限的共享内存模型，即 RDD 是只读的记录分区的集合，只能通过在其他 RDD 执行确定的转换操作（如 map、join 和 group by）而创建，然而这些限制使得实现容错的开销很低。对开发者而言，RDD 可以看作是 Spark 的一个对象，它本身运行于内存中，如读文件是一个 RDD，对文件计算是一个 RDD，结果集也是一个 RDD，不同的分片、数据之间的依赖、key－value 类型的 map 数据都可以看做 RDD。图 3 显示 Spark 计算的 RDD 模型。

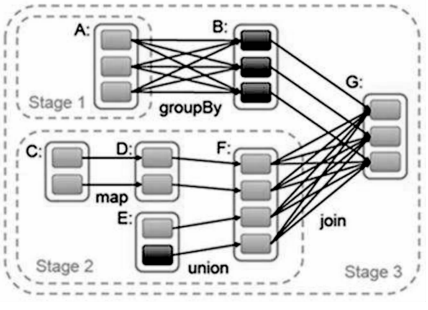

图 3　Spark 计算过程不同阶段的 RDD 模型

2　海量飞行数据预处理方法

飞机试验数据解析效率是影响大数据分析，挖掘额外信息资源的重要影响因素。随着飞机集成度的提高，传感器数据的增多，飞机试验数据的参数种类与数据量剧增。传统的飞机数据解析方法主要是针对源码数据量小的情况，无法满足当前大型飞机试验任务的需求。而针对大规模源码数据，目前一般都是工程师对任务进行分工，每个人负责任务的一部分，在各自的机器上运行，最后将结果进行汇总，这不仅会增加人力成本，也会增加解析的复杂度，降低解析效率，影响下一步试飞决策。因此，需要建立一种针对大规模飞行试验数据的预处理方法。

本文一方面采用 Spark 处理框架，通过分布式计算方

式，减少磁盘交互产生的 I/O，提高计算效率；另一方面，利用飞机数据采集特点，减少不必要的解析，节省集群内存计算空间，提高解析效率。从数据解析过程优化与并行处理的角度，提出了基于 Spark 的海量飞行试验数据的预处理方法。

2.1 飞行数据解析优化

飞机运行过程中，机载采集器根据规定的采样率对机载数据进行采集，当次试飞结束后下载原始数据。由于数据参数格式固定，在数据采集过程，根据实际情况解析前段数据，直到获取所有需要的参数初始信息位置，包括参数的采样初始时间，整个数据文件采样的初始时间以及数据文件出现的参数名全集等。

确定数据文件中存在的参数集合，预先剔除参数组中不存在于当前数据文件中的参数群，只提取存在于数据文件中的参数属性，有效地跳过不必要的解析过程。同时减少广播变量（Broadcast）占用的内存，为集群内存计算节省空间，显著提高解析的效率。

进而，切割并行数据，生成若干个原始数据包。切包的规则是只能在两个单包之间切，而不能跨包，否则会丢失数据，通过不断计算单包的长度来用指针偏移的方法来切包。

在此基础上，对飞行测试数据进行预处理。预处理过程分为两个阶段，配置阶段和应用阶段。图 4 显示了两个阶段的过程，包括控制流程和数据流程。

在配置阶段，主要是在试飞数据处理设备正式投入运行之前和运行过程中，进行系统基础信息的配置与扩展开发，并且进行系统数据库的管理，为系统提供原始数据的解码、分析逻辑、算法功能等方面的配置和扩展开发能力，使系统具备持续的完善能力。

1）首先进行的是用户配置、试验信息配置与基础信息配置的工作。

2）其次是完成对配置文件的解析和试验数据的预处理。

3）再次是存储原始二进制数据和解析后工程值数据。

4）最后针对解析后数据为用户提供数据服务，包括数据的查询、下载、导出、分析、二次计算等。

在准备阶段，从配置文件中读取帧格式的配置信息，包括帧参数的位置，采样率，通道数等。然后读取参数数据表中所有帧参数的数据类型、分辨率、数据长度等。最后，根据参数的位置信息设置参数索引表。

在解码阶段，当接收到源代码时，根据 429、232、664 协议规定的帧格式对帧头进行解析，获取当前帧的索引表唯一标识，通过标识确定当前帧采集的参数索引表，进而根据参数信息将源码数据解析为工程值数据。

2.2 基于 SPARK 的数据处理技术

在使用 SPARK 技术来解决大规模数据预处理问题的过程中，如何在每个预处理步骤中构建测试数据 RDD 模型以及如何使用 SPARK 算子至关重要，直接影响数据预处理的效率。图 6 为基于 SPARK 的数据处理流程图。

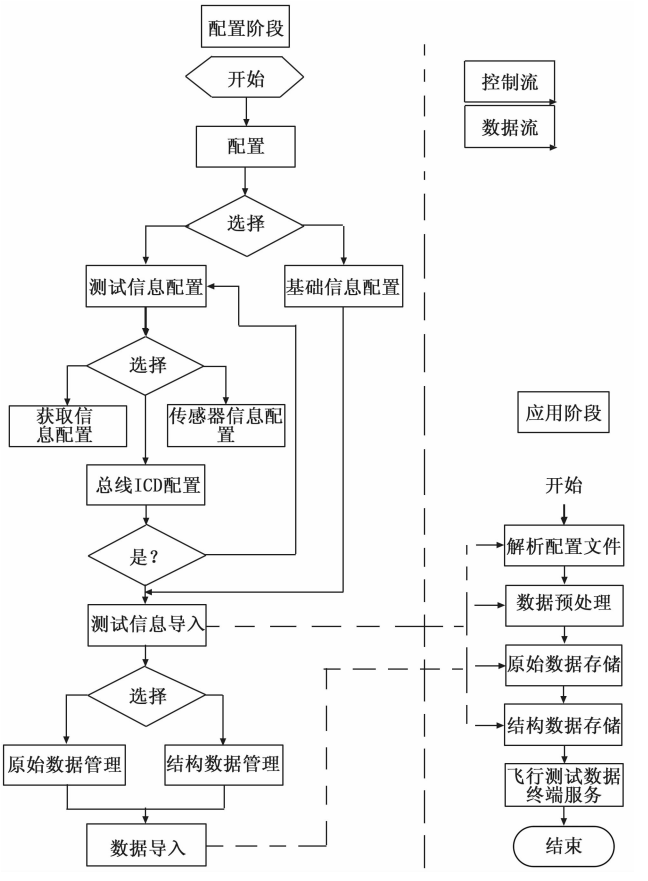

图 4 图解码阶段的数据预处理步骤

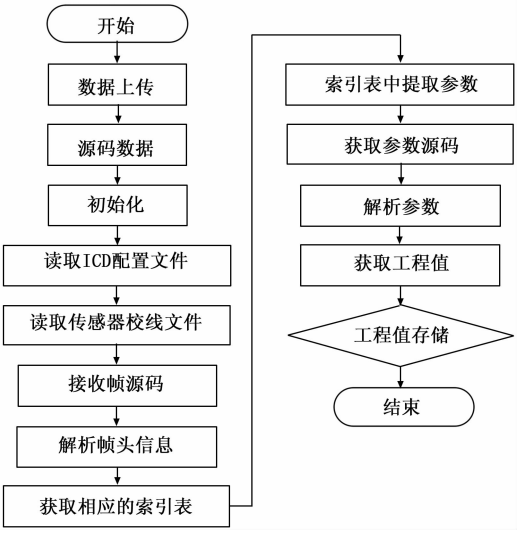

图 5 解码阶段的数据预处理步骤

基于 SPARK 的数据处理具体步骤如下：

步骤 1：初始化。包括共享变量初始化，通过解析配置文件，获取参数信息和协议解析规则，并打包成广播变量

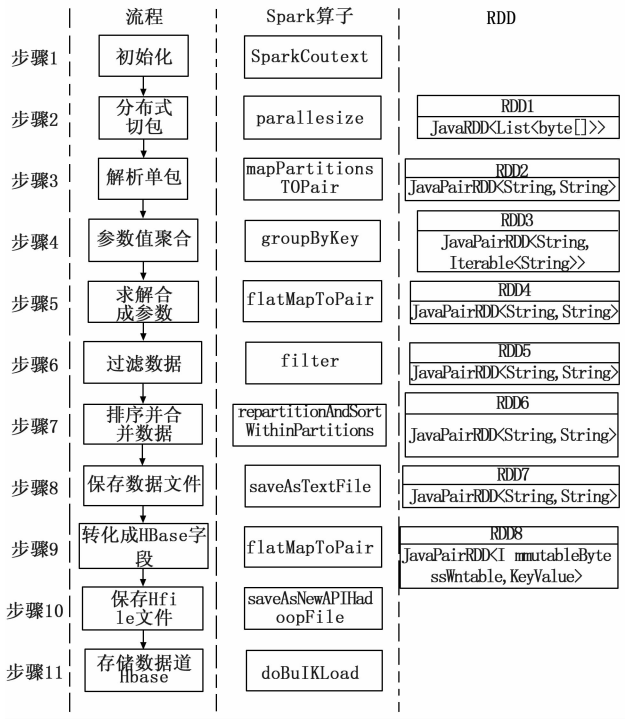

图 6　基于 SPARK 的数据处理流程图

作为集群共享信息，同时初始化累加器，作为分布式运算过程中不同线程之间数据通信的媒介；以及待解析数据初始化，将源码数据文件的上传到 HDFS，并按照 1 GB 一个单元进行预先切包。最后还对 spark 环境进行初始化，指定基本的参数配置，保证 spark 的高效运行。

步骤 2：分布式切包。将上传到 HDFS 上的数据加载到分布式内存中，在加载的过程中，同时将数据每隔 N（一般可以设定为 1000）个帧切一次包，并将所有切后的包重组为 K（一般可以设定为 2000）份，设定为每份启动一个线程，进行分布式处理。将加载到分布式算子中。一部分数据将打开一个线程。创建数据结构为 JavaRDD＜List＜byte []＞＞的 RDD 数据集，该数据集作为分布式处理的数据源，其中每个 byte 数组代表的是帧二进制数据。

步骤 3：解析单包。根据步骤一中初始化的广播变量信息，解析帧头，获取帧头的标识信息，根据标识以及解析的配置文件获取当前帧采集的参数以及参数属性，进而将源码帧数据解析成工程值数据。期间，待解析的参数组范围之外的参数直接跳过，保证解析速度的最大化；此外，整个解析过程通过 mapPartitionsToPair 算子执行并行处理，各个线程共享广播变量，尽可能的避免不同线程在运算过程中的 shuffle 操作，减少线程之间由于通信导致的延迟。该步骤获取的数据格式为 JavaPairRDD＜String，String＞，存储的内容为＜参数组＋时间，参数位置＋参数值＞信息，保证每个基本参数都从二进制解析成工程值数据。

步骤 4：参数值聚合。对步骤三得到的结果进行聚合操作，执行 groupBykey 算子，将 key（参数组＋时间）相同的 value（参数位置＋参数值）聚合在一起，得到当前参数组当前帧时间的所有参数数据集，并将数据集根据参数组规定的顺序组成长字符串，以便获取同一采样时间点的参数值，便于进行事后试验数据的关联分析。最终获取到的数据结构为 JavaPairRDD＜String，Iterable＜String＞＞，存储的结果为＜参数组＋时间，参数值字符串集合＞。

步骤 5：求解合成参数。有一些参数需要一些基本参数经过一定的运算法则转换才能获取，如两个参数的加减乘除运算、位权重运算、EU 转换、多项式计算等。所以在将基础参数工程值解析出来后，需要找到合成参数需要的基

	Host Name	OM IP	Network Speed	Health Status	Disk Usage	Memory Usage	CPU Usage
	hadoop001	10.137.241.131	R:81.90KB s,W:1…	Good			
	hadoop002	10.137.241.132	R:10.20KB s,W:5…	Good			
	hadoop003	10.137.241.133	R:10.20KB s,W:4…	Good			

图 7　主机配置

本参数相同时间点的数据集，然后经过算法运行，得到合成参数的值。其结果形式与步骤 5 产生的结果一致。

步骤 6：过滤数据。针对不同的参数组，需要分别导出到不同的文件中，以保证不同试验科目的独立性，便于不同专业的专家对数据进行分析。然后以参数组＋时间字符串中的参数组为过滤条件，使用 filter 算子，筛选出所需的参数组数据集，由于执行的是 filter 算子，得到的 RDD 数据集数据结构没有变化，结果形式与步骤 5 一致。

步骤 7：首先需要对整体数据进行全局排序，以保证所有数据能够以时间顺序进行排列。此时数据是分布式的散播在内存中，为了将参数组数据以规定的数据格式导出，需要将内存中所有的相关数据合并在一起，使用 repartitionAndSortWithinPartitions 算子，将 partitioner 参数设置为 1，即得到合并后的数据。由于只是数据的合并，所以 RDD 结构仍然不变，为 JavaPairRDD＜String，String＞。

步骤 8：保存数据文件。使用 saveAsTextFile 算子，将步骤 7 得到的合并结果保存到 HDFS 的文件系统中，每个参数组都分别形成独立的文件系统，通过服务的形式分发给不同试验科目的专业人员，进行进一步的数据分析。

步骤 9：转化为 HBase 字段。预处理的数据需要进行卸载，一方面以系统文件的形式分发给专业人员，另一方面需要持久化存储到分布式数据库 HBase 中，以便时候的随时查询和进一步的分析，通过 flatMapToPair 算子将步骤 6 的 RDD 数据集转化为满足 HBase 特定形式的 JavaPairRDD＜ImmutableBytesWritable，KeyValue＞ 。

步骤 10：保存 HFile 文件。由于实际产生的数据量过大，直接存储到 HBase 容易出现内存溢出问题，并且消耗时间较长。故而采取先转存为 HFile 文件的形式进行 HBase 存储，使用 saveAsNewAPIHadoopFile 算子将数据集转化为 HFile 文件。

步骤 11：存储到 HBase。通过 doBulkLoad 方法将 HFile 文件转化为 HBase 数据，进而完成送源码数据到工

程值数据的处理工作。

3 实验验证

3.1 实验环境

实验环境部署在有 3 个物理服务器的集群中，其中一个服务器既作为主管理节点，又作为计算节点；另一个服务器作为备管理节点和计算节点；最后一个服务器作为计算节点，具体环境配置如表 1 所示。

表 1 实验环境配置

	服务器 1	服务器 2	服务器 3
节点	主管理/计算节点	备管理/计算节点	计算节点
运行系统	Centos 6.5	Centos 6.5	Centos 6.5
SPARK 版本	1.3	1.3	1.3
CPU/GHz	2.1	2.1	2.1
CPU 内核数量	12	12	12
内存/G	128	128	128
硬盘/TB	30	30	30

试验数据来源于某型号飞机试飞过程产生的数据，数据量为 30 GB。将试验数据分成三份，分别在三台主机上进行预处理。

3.2 实验结果分析

基于 Spark 的飞机试验数据预处理结果，如图 8 所示。数据切包、单包解析、参数聚合、参数合成、数据存储所需时间约为 30 分钟。传统数据处理平台数据预处理时间则需要 9 小时。相较于传统数据处理方法，本文提出的数据预处理方法大大提高了解析效率，克服了传统数据处理方式效率低下，单个机器内存和 CPU 等硬件条件不足的问题。

Stage ID	Description	Submitted	Duration	Tasks:Succeeded/Total	Input	Output	Shuffle Read	Shuffle Write
8	saveAsNewAPIHadoopFile at SparkHDFSDirSrveToFTPAndHDFSSync.java:950	2017/4/20 17:38:11	17min	1000/1000		28.7GB	6.8GB	
7	mapToPair at SparkHDFSDirSrveToFTPAndHDFSSync.java:905	2017/4/20 17:31:16	6.9min	1000/1000			719.0MB	6.8GB
5	RangePartitioner at SparkHDFSDirSrveToFTPAndHDFSSync.java:916	2017/4/20 17:26:25	4.6min	1000/1000			719.0MB	
3	count at SparkHDFSDirSrveToFTPAndHDFSSync.java:321	2017/4/20 17:21:52	4.5min	1000/1000			719.0MB	
1	count at SparkHDFSDirSrveToFTPAndHDFSSync.java:270	2017/4/20 17:21:49	2s	1000/1000			719.0MB	
0	repartition at SparkHDFSDirSrveToFTPAndHDFSSync.java:262	2017/4/20 17:21:29	20s	24/24				719.0MB

图 8 各阶段处理时间

主机一运行线程 333 个，各线程累积执行时间 2.3 h，解析工程数据文件 9.5 GB；主机二运行线程 331 个，各线程累积执行时间 2.3 h，解析工程数据文件 9.5 GB；主机三运行线程 336 个，各线程累积执行时间 2.3 h，解析工程数据文件 9.7 GB。各线程执行时间最小为 5 s，最大为 55 s。数据垃圾回收时间最小为 0 ms，最大为 32 s。各线程处理原始数据最小为 3.6 MB，最大为 11.3 MB，解析数据输出大小最小为 17.2 MB，最大为 42.2 MB，如图 9 所示。尽管本文提出的方法相较于传统方法，大大提高了数据解析效率。但各线程负载不均衡，未充分利用系统资源，影响系统运行性能，对数据解析时间产生一定的负面影响，这也是后续需要进一步改进的地方。

4 结论

本文针对海量飞行数据预处理方法处理效率低、处理

Metric	Min	25th percentile	Median	75th percentile	Max
Duration	5s	17min	24s	30s	55s
GC Time	0ms	6.9min	0.3s	0.4s	32s
Output Size/Records	17.2MB/226632	4.6min	28.9MB/378515	31.9MB/415703	42.2MB/555038
Shuffle Read Size/Records	3.6MB/226632	4.5min	7.0MB/378515	7.9MB/415703	11.3MB/555038

Aggregated Metrics by Executor

Executor ID	Address	Task Time	Total Tasks	Failed Tasks	Succeeded Tasks	Output Size/Records	Shuffle Read Size/Records
1	10.137.241.131:23941	2.3h	333	0	333	9.5GB 126724000	2.3GB 126724000
2	10.137.241.132:23707	2.3h	331	0	331	9.5GB 126443513	2.3GB 126443513
3	10.137.241.133:23256	2.3h	336	0	336	9.7GB 128928448	2.3GB 128928448

Tasks

Index	ID	Attempt	Status	Locality Level	Executor ID Host	Launch Time	Duration	GC Time	Output Size/Records	Shuffle Read Size Records	Errors
0	4024	0	SUCCESS	PROCESS_LOCAL	1 hadoop001	2017/4/20 17:38:11	17s	2s	27.4MB 368584	9.9MB 368584	
3	4027	0	SUCCESS	PROCESS_LOCAL	1 hadoop001	2017/4/20 17:38:11	15s	2s	31.3MB 431862	10.4MB 431862	

图 9 飞行测试数据预处理记录

周期长等问题，提出了基于 Spark 的飞机试验数据预处理方法，并以某型飞机 30 GB 的试飞数据为例进行实验验证，结果显示本文提出的数据预处理方法相较于传统的大规模数据预处理方法可有效地缩短数据预处理时间，提高处理效率，帮助用户快速做出下一步试飞决策。未来将重点研究数据倾斜调优与 Shuffle 调优的问题，优化 Spark 的性能，进一步提高数据解析效率。

参考文献：

[1] Li H，Ghodsi A，Zaharia M，et al. Reliable，memory speed storage for cluster computing frameworks [A]. Proc. SoCC [C]. 2014.

[2] Meng X，Bradley J，Yavuz B，et al. Mlib：Machine learning in apache spark. arXiv preprint，ar Xiv：1505. 06807 [Z]. 2015.

[3] Ansel J. Butter field data mining of NASA Boeing737 Data [R]. Flight Data Frequency Analysis of In－Flight Recorded. NASA/CR－2001－2106 41，2001.

[4] Dean J，Ghemawat S. MapReduce：simplified data processing on large cluster [A]. Proceedings of the 6th Symposium on Operating System Design and Implementation [C]. 2004.

[5] Liu Z Q，Gu R，Yuan C F，et al. The parallelization of classification algorithms based on SparkR [Z]. 2014.

[6] Liu H Y，Yuan Q Q，Wang B B. Survey of parallel algorithms for data mining [J]. Electronic Science and Technology. 2006，1：6573.

[7] Jiang J，Lu J，Zhang G，et al. Scaling－up itemBased collaborative filtering recommendation algorithm based on hadoop [A]. 2011 IEEE World Congress on Services（SERVICES） [C]. IEEE，2011：490－497.

[8] 黎文阳. 大数据处理模型 Apache Spark 研究 [J]. 现代计算机（专业版），2015（8）：55－60.

[9] 李 爽. 基于 Spark 的数据处理分析系统的设计与实现 [D]. 北京：北京交通大学，2015.

[10] 张 磊，朱 锋，钟 华. 基于 Spark 的交互式数据预处理系统 [J]. 计算机系统应用，2016，25（11）：84－89.

[11] 张 洪，赵 平，伍 玲，等. 基于 Spark 的分布式车流量检测方法设计与实现 [J]. 计算机测量与控制，2018，26（2）：199－202，206.