网络首发时间：2018-08-09 13:22:35
网络首发地址：http://kns.cnki.net/kcms/detail/11.2127.tp.20180806.1701.004.html

基于流式计算的遥感卫星数据快视处理方法

宋　峣[1,2,3]，孙小涓[1,2,3]，胡玉新[1,2,3]，雷　斌[1,2,3]，卢晓军[4]

SONG Yao[1,2,3], SUN Xiaojuan[1,2,3], HU Yuxin[1,2,3], LEI Bin[1,2,3], LU Xiaojun[4]

1. 中国科学院大学, 北京 100049
2. 中国科学院 电子学研究所, 北京 100190
3. 中国科学院 空间信息与应用系统重点实验室, 北京 100190
4. 中国国际工程咨询公司, 北京 100048
1.University of Chinese Academy of Science, Beijing 100049, China
2. Institute of Electronics, Chinese Academy of Science, Beijing 100190, China
3. Key Laboratory of Technology in Geo-spatial Information Processing and Application System, Beijing 100190, China
4. China International Engineering Consulting Corporation, Beijing 100048, China

SONG Yao, SUN Xiaojuan, HU Yuxin, et al. A quick-view processing methodfor remote sensing data based on stream computing. Computer Engineering and Applications

Abstract: With the improvement ofdata acquiring ability of high resolution remote sensing satellites and data receiving ability of the ground receiving stations, the processing load of existed system grows increasingly heavier and the real-time processing demand becomes more difficult to meet. Focusing on these issues, a new system design method for quick-view processing of remote sensing satellite data is proposed by using the thought of stream computing. After analyzing the characteristics of quick-view processing data streams, we apply the Storm framework to the parallel optimization of the existed system, design the topology of stream computing tasks for the remote sensing satellite data processing, and use Kafka message oriented middleware to improve the mechanism of data exchanging and data buffering in processing units.In experiments the improved system shows good results in throughput and reliability.

Key words: stream computing; data stream; storm;quick-viewprocessing;remote sensing data processing

摘　要：随着高分辨率遥感卫星数据获取能力和地面数传接收能力的提高，现有遥感卫星快视处理系统的处理负载增大，实时性要求越来越难以满足。针对这些问题，采用流式计算思想提出了一种新的遥感卫星数据快视处理系统设计方法，在分析遥感卫星数据快视处理数据流特点的基础上，应用 Storm 框架对现有系统进行并行优化，设计遥感数据流处理任务拓扑结构，同时利用消息队列中间件 Kafka 改进处理单元间数据交换和数据缓存方式。实验表明该系统在数据吞吐率和可靠性方面测试效果良好。

关键词：流式计算；数据流；storm；快视处理；遥感数据处理
文献标志码：A　**中图分类号：**TP75　doi：10.3778/j.issn.1002-8331.1801-0342

作者简介：宋峣(1994-)，男，硕士，研究领域为空间信息处理，E-mail:sy94922@163.com；孙小涓(1980-)，女，博士，副研究员，研究领域为空间信息处理、高性能计算；胡玉新(1981-)，男，博士，研究员，研究领域为空间信息处理、信号处理；雷斌(1978-)，男，博士，研究员，研究领域为空间信息处理、信号处理；卢晓军(1977-)，男，博士，高级工程师，研究领域为智能控制、信号处理。

1 引言

随着高分辨率遥感卫星载荷数据获取能力的提高和卫星地面站数传接收能力的增强，遥感卫星数传数据量不断增大。在一次卫星过境中，地面接收站持续接收卫星下传数据，经过一系列的数据格式转换生成快视图像，数据处理过程不间断、数据吞吐率高并且数据处理延迟低，因此遥感卫星数据快视处理具有流式计算的特征。它为后续的图像快视显示提供实时平滑的数据处理结果，对快速分析数据接收质量、获取图像信息发挥了重要作用。

由于遥感卫星数传接收任务和数传数据量的增大，卫星数据快视处理的实时性要求越来越难以满足。并且卫星数据价值昂贵不可丢失，在快速处理的同时，还需兼顾海量数据的安全可靠性，因此，遥感卫星数据快视处理系统设计面临双重挑战。

随着遥感技术与计算技术的快速发展，遥感卫星地面数据处理系统也经历了十多年的发展，从为单颗特定卫星设计，到为特定多颗卫星设计[1]，业务流程的调度与管理作为遥感卫星处理系统关注的问题已有较多研究。2009 年开始我们结合工作流技术探索通用遥感卫星地面运行控制模型[2]，并针对遥感数据处理特点研究工作流任务规划方法[3,4]，已应用在实际系统中。采用传输帧和源包联合索引结构，以数据流驱动方法进行卫星数据处理系统框架研究[5]，对卫星数传数据流特点进行了分析并加以利用。在已有研究的基础上，本文选取典型的遥感卫星数据快视处理为研究对象，采用流式计算思想对系统设计方法开展进一步研究。

遥感卫星数据快视处理系统是对实时接收到的遥感卫星原始数据流[6]，进行帧同步、解压缩、分景编目与快视处理的数据处理系统。现有的遥感卫星数据快视处理系统将数据处理过程分为两个处理步骤，由高速网络连接的帧同步处理节点和快视处理节点完成。从解调器输出的卫星原始数据以二进制码流的形式由帧同步处理节点接收，软件通过查找同步码确定帧头位置，然后进行解扰和译码处理，一边进行数据处理一边将处理结果发送到快视处理节点；接着软件根据虚拟信道标识提取不同源包数据，针对传感器特点进行图像拼接等数据处理，处理结果用于支持快视图像的实时移动窗显示。

现有遥感卫星数据快视处理系统采用两个处理软件流水并行处理的方式，当接收卫星图像数据流量增大时，数据处理延迟较大；而且帧同步处理节点和快视处理节点负载增大，有时甚至发生数据无法接收处理的情况，造成图像缺失。针对这个问题，我们基于流计算框架 Storm 提出了一种新的遥感卫星数据快视处理系统设计方法，对现有工作流系统进行并行优化，对遥感卫星数据快视处理过程中数据流特点进行分析，设计任务拓扑结构并开发软件实现，最后对数据吞吐率和可靠性指标进行分析评价。

2 流式计算技术

流式计算[7,8]是相对于批量计算的一个概念，指将到达的数据流在内存中实时计算，而批量计算指对存储的静态数据进行集中计算，因此流式计算具有低时延、高吞吐且持续运行的特点。常见的流式计算框架有 Twitter Storm[9]、Spark Streaming[10]、Yahoo S4[11]等。近年来，流式计算技术因具有分布式计算的高效性与数据处理的实时性的特点，成为一个研究热点，广泛应用于金融、互联网、物联网等诸多领域，如股市实时分析、实时视频分析[12]、交通流量实时预警[13]等。

Storm[14,15]是一个分布式开源实时计算系统，它采用主从式结构（图 1），包括一个主节点 Nimbus 进程和多个从节点 Supervisor 进程，通过 ZooKeeper

分布式应用程序协调服务[16]同步节点状态信息，每个节点按需创建运行 Java 虚拟机的 Worker 进程，用于计算任务的执行。

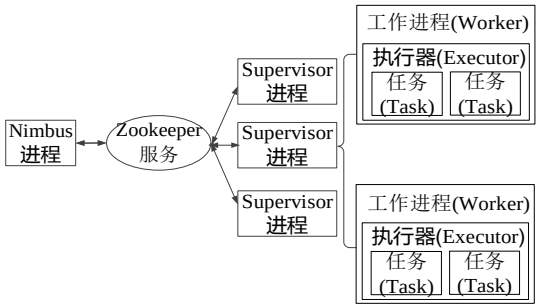

图 1　Storm 体系架构

Storm 的计算模型[17,18]（图 2）由拓扑（Topology）、元组（Tuple）、流（Stream）、喷口（Spout）、螺栓（Bolt）和任务（Task）构成，任务拓扑是由一系列 Spout 和 Bolt 组成的有向无环图，元组定义了 Spout 和 Bolt 之间传递消息的数据单元，而流是无界的元组序列，源源不断的传递元组就构成了流。通常 Spout 获取数据源并不停地发送数据给 Bolt，Bolt 接收数据进行相应的处理。Spout 和 Bolt 上执行的具体操作为 Task，可以灵活设置每个 Spout 或 Bolt 上并行执行的任务数。

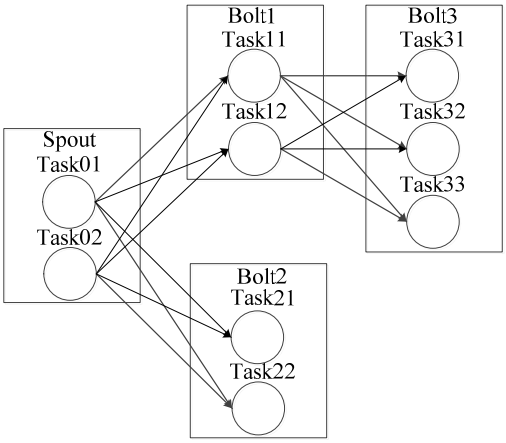

图 2　Storm 计算模型图

3　遥感卫星数据快视处理的数据流分析

3.1　遥感卫星数据快视处理数据流

在卫星过境时实时接收卫星数传通道下传的遥感数据，并经过解调处理的卫星原始码流数据，是遥感卫星数据快视处理系统的数据源。实时快视处理的数据流具有单次线性扫描处理的特点，无法重复对数据流进行处理。数据以二进制码流形式进行处理，通过帧同步、解扰、译码、虚拟信道分离、IQ 拼接和图像拼接等步骤，生成用于快视显示的图像。遥感卫星数据快视处理的数据流图如图 3。

遥感卫星数据快视处理系统的数据流处理步骤包括：

（1）帧同步：从数传原始码流中按位查询同步码，确定每个传输帧起始位置。

（2）解扰：以传输帧为处理单位，对除同步码以外的数据进行位运算。

（3）虚拟信道分离：以传输帧为处理单位，解析传输帧头信息，按照虚拟信道标识进行源包数据提取。

（4）IQ 拼接：以 B-PDU（Bitstream Protocol Data Unit）位流数据为处理单位，将连续多帧 I 路和 Q 路的位流数据拼接，获取 VCDU（VirtualCode DataUnit）源包数据。

（5）RS（Reed-solomon）译码：以 VCDU 源包数据帧为处理单位，对传输中的误码进行纠正。

（6）图像拼接：以行数据为处理单位，提取图像数据，对不同图像传感器信号进行拼接，得到完整图像。

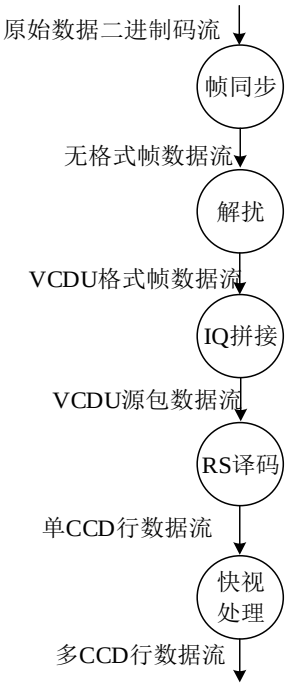

图 3　遥感卫星数据快视处理的数据流图

3.2　数据流定义

卫星原始数据具有传输帧和源包两层数据结构，对卫星数据采集和传输过程中不同数据源设备进行区分，构成可变长度的“源包”，过长的包分成段，加上帧头和帧尾构成“传输帧”进行传输。数据传输帧具有虚拟信道标识符、VCDU 计数器等字段，源包具有数据指针等字段，可用于数据重组和数据提取。

对遥感卫星数据快视处理的数据流进行分析，定义 Storm 流式处理的数据流格式。根据不同处理步骤，以虚拟信道传输帧、源包数据和行图像数据为依据，来划分数据单元或元组进行并发处理，由此可定义各处理步骤的数据流（表 1），包括无格式帧数据流、VCDU 格式帧数据流，单 CCD（Charge Coupled Device）图像行数据流和多 CCD 图像行数据流。

4　基于流式计算的处理方法

4.1　数据流处理拓扑结构

在分析数据流的基础上，采用 Storm 流式计算框架对遥感卫星数据快视处理系统进行改造。用任务拓扑 Topology 来描述完整的遥感数据快视处理过程，数据接入 Spout 组件负责原始数据流的实时接收；帧同步、解扰、IQ 拼接、RS 译码、快视处理等处理环节分别由相应的数据处理 Bolt 组件完成，用于对遥感图像数据流进行计算处理并形成相应结果；RS 译码后，对图像数据按行排序。任务拓扑结构如图 4 所示。

表 1　遥感卫星数据快视处理的数据流定义

(a) 无格式帧数据流（大小为 1058 字节）

帧同步头	加扰的数据
4bytes	1054bytes

(b) VCDU 格式帧数据流（大小为 1058 字节）

	VCDU 主导头							VCDU 插入区	VCDU 数据单元		
帧同步头	版本号	VCDU 标识符		VCDU 计数器	信号域		备用域	字段差错控制域	B-PDU 导头		B-PDU 位流数据区
		航天器标识符	虚拟信道标识符		I/Q 标志	备用域			备用域	位流数据指针	
32bits	2bits	8bits	6bits	24 bits	2bits	6bits	144bits	32bits	2bits	14bits	8192bits

(c) VCDU 源包数据流（大小为 9690 字节，第 1-8 帧的数据区为 1024 字节，第 9、10 帧的数据区为 749 字节）

第 1 帧的 B-PDU 位流数据区	……	第 8 帧的 B-PDU 位流数据区	第 9 帧的 B-PDU 位流数据区	第 10 帧的 B-PDU 位流数据区
1024bytes	……	1024bytes	749bytes	749bytes

(d) 单 CCD 图像行数据流（大小为 9234 字节）

行同步字	行计数	卫星辅助数据	辅助数据计数	相机图像数据		填充数据
				暗象元	图像数据	
6bytes	3bytes	64bytes	2bytes	39bytes	9000bytes	120bytes

(e) 多 CCD 图像行数据流（大小为 18000 字节）

图像数据 CCD1	图像数据 CCD2	图像数据 CCD3
6000bytes	6000bytes	6000bytes

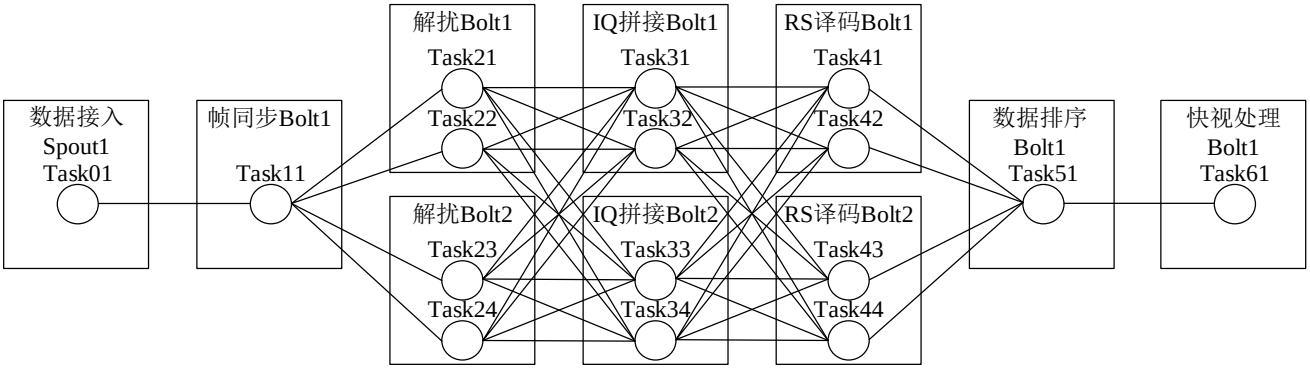

图 4　遥感卫星数据快视处理的任务拓扑图实例

```
<topologyconfig>
<spout name='DataSpout' parallelism='1'/>
<bolt name='FrameSyncBolt' parallelism='1' datainput='DataSpout'/>
<bolt name='DescramBolt' parallelism='2' datain-
put='FrameSyncBolt'/>
<bolt name='StitchBolt' parallelism='2' datainput='DescramBolt'/>
<bolt name='DecodeBolt' parallelism='2' datainput='StitchBolt'/>
<bolt name='SortBolt' parallelism='1' datainput='DecodeBolt'/>
<bolt name='QuickViewBolt' parallelism='1' datainput='SortBolt'/>
</topologyconfig>
```

图 5　遥感卫星数据快视处理系统任务拓扑定义示例

图 5 为遥感数据快视处理系统任务拓扑的 xml 文件实例。改造的系统将工作负载分解为数据接入 Spout 组件和帧同步 Bolt、解扰 Bolt、IQ 拼接 Bolt、RS 译码 Bolt、数据排序 Bolt、快视处理 Bolt 等 6 个 Bolt 组件。与现有系统相比，改造系统中数据处理单元从原来的 2 个增加到 6 个，对于耗时较长、影响整体处理进度的 Bolt 组件可增加并行处理单元，增加的 Bolt 组件可以运行在不同的处理节点，用来分散较大的任务负载。以图 5 任务拓扑图为例，解扰 Bolt、IQ 拼接 Bolt、RS 译码 Bolt 的并行 Bolt 数为 2，并行 Task 数为 4，此时改造系统不会出现处理节点过载的情况。同时，除了帧同步 Bolt、数据排序 Bolt 和快视处理 Bolt，改造系统的其余 Bolt 都可以多任务并行处理，系统结构从原来的“1+1”模式转变为“1+n+1”的并行模式，使得系统结构动态可扩展。

4.2　数据交换优化

由于遥感数据处理环节多而复杂，数据规模庞大，因此数据交换速率很大程度上影响系统的整体性能。使用 Storm 内置的数据通信方式不能较好的

满足高速数据交换需求，需要设计高吞吐率、安全可靠的数据交换系统，用于各组件间的数据交换。采用 Kafka[19,20]消息队列系统，对 Storm 的数据交换方式进行改进。发送与接收数据示例代码如图 6 所示：

```
帧同步组件 FrameSyncBolt 数据发送线程：
Properties props = new Properties();//配置文件
props.put("metadata.broker.list", "10.0.0.2:9092");//连接 kafka 服务器
props.put("serializer.class", "kafka.serializer.DefaultEncoder");//设置
数据格式
producer = new Producer<String, byte[]>(new ProducerConfig(props));
producer.send(new KeyedMessage<String, byte[]>,"DATAFRAME",
key ,sendbytes));//数据发送

解扰组件 DescramBolt 数据接收线程：
Properties props = new Properties();//配置文件
props.put("zookeeper.connect","192.168.11.1:7181");//连接 zookeeper
props.put("zookeeper.session.timeout.ms", "4000");//设置超时时间
props.put("zookeeper.sync.time.ms", "200");//zookeeper 同步时间
ConsumerConfig config = new ConsumerConfig(props);
consumer = kafka.consumer.Consumer.createJavaConsumerConnector
(config);//创建数据接收线程
KafkaStream<String, byte[]> stream = consumerMap.get("DATAFR
AME").get(0);//订阅数据并开始接收
it = stream.iterator();//使用迭代器提取数据
if (it.hasNext()){
  MessageAndMetadata<String, byte[]> strr = it.next();
  String key = strr.key();
  byte[] data = strr.message();
}
```

图 6　发送与接收数据示例

Kafka 采用零拷贝机制[21]，大幅提高数据收发性能，具有较高的吞吐率，并提供数据持久化能力。Kafka 采用集群架构使多个代理 Broker 协同完成消息交换。生产者发布选定主题的消息，消费者通过向主题注册接收生产者发布的该主题消息。

引入 Kafka 数据交换系统后，卫星数据快视处理时间总耗时从原来的 104.73s 缩短到 71.08s，采用 Kafka 进行数据交换优化后获得了约 32%的提升。

4.3 数据流备份机制

遥感卫星数据较为昂贵，因此遥感数据的可靠性格外重要，一旦出现故障容易导致数据丢失。使用 Kafka 通过配置备份机制提高数据可靠性，保证系统能自动容错，如果有服务器宕机，数据可以从其他运行的服务器中读取，不影响集群系统稳定性。具体方法是：

（1）根据数据流备份机制，对于各类数据均有一个主节点和多个从节点，从节点个数为数据备份数。为了负载均衡，各类数据的主节点在集群中均匀分布。

（2）主节点负责处理数据的所有读写请求，从节点从主节点中复制数据用于备份并保存，并与主节点保持同步；当所有从节点都将数据保存成功，数据才被认为获取成功。

（3）主节点负责跟踪并保存从节点的所有状态，维持各个备份间的状态同步。

（4）当主节点故障时，从备份从节点中选举出新的主节点，此时新的主节点获取各从节点状态信息，继续负责处理读写请求，对外正常提供服务；当备份从节点故障时，主节点将删除该节点信息，不再用于备份。

应用这种数据流备份机制能提高系统可靠性，当具有 m 个数据节点时，在 m-1 个数据节点同时故障的情况下数据不丢失，系统仍提供正常服务。即使只有一个数据，仍可以保证数据的正常发送和接收。

5　性能分析

5.1　实验方法

我们搭建了测试验证环境，对资源一号 02C 卫星快视处理流程进行测试验证。实验采用 10GInfiniband 网络连接的 4 台高性能计算服务器节点（2 个 12 核 Intel Xeon X5670 CPU，主频 2.93 GHz，36 GB 内存，2 块 60G SATA 磁盘）对数据吞吐率和数据完整度进行测试。服务器使用 Red Hat Enterprise Linux 7.2 操作系统，将数据接入 Spout 组件

和帧同步处理 Bolt 组件部署到一个节点上，其余节点部署解扰处理 Bolt 组件、IQ 拼接处理 Bolt 组件、RS 译码处理 Bolt 组件、数据排序 Bolt 和快视处理 Bolt 组件。实验数据使用资源一号 02C 卫星原始数据，采用文件回放方式模拟卫星数据接收过程。

数据吞吐率指标为数据处理组件接收到的遥感数据量除以接收时间，单位为 MB/s。数据完整度指标为节点关机后未丢失的数据占总数据量的比例，单位为%。

5.2 数据吞吐率

（1）不同数据流元组大小

实验使用总大小 5GB 的数据文件，分别按每块大小 1KB、10KB、50KB、100KB、200KB、500KB、1MB 和 5MB 对组件之间的数据交换速率进行测试，记录数据发送速率，实验结果如图 7 所示。

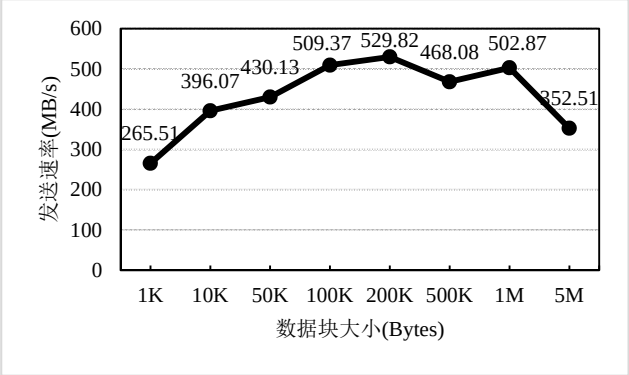

图 7 发送速率随数据块大小变化图

实验表明，数据块大小在 100KB-1MB 之间能保持较高的数据吞吐率。

（2）不同数据流容量

将数据切分为 100KB 的数据流，对总量为 1GB、5GB、10GB、20GB 和 30GB 的数据文件进行测试，记录数据发送速率，实验结果如图 8 所示。

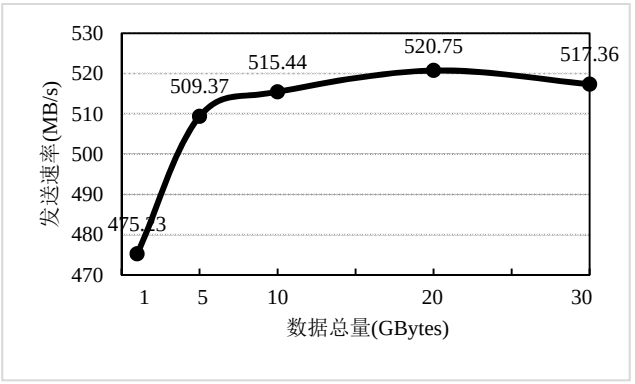

图 8 发送速率随数据总量大小变化图

实验表明，随着发送数据总量的增多，发送速率较稳定，保持在约 516MB/s 的较高水平。

（3）不同数据流并行数

分别发起 1、2、4 个数据接入 Spout 任务线程，数据流大小为 100KB，总量为 5GB，实验结果如表 2 所示。

表 2 线程数对发送速率的影响

线程数	发送速率
1	509.37 MB/s
2	819.87 MB/s
4	907.33 MB/s

实验表明，多任务线程能提高数据总吞吐率，但由于节点资源受限，多个遥感卫星原始数据码流的处理性能明显降低。

5.3 数据可靠性

对于无备份方案，即不使用数据流备份机制，单节点故障后无法恢复将导致数据丢失。

为了保证数据的可靠性，使用 Kafka 的数据流备份机制。对于单备份方案，即设置一个从节点用于数据备份，若一个节点故障，数据不丢失；两个及以上节点同时故障时，故障节点上数据丢失。

对于双备份方案，即设置两个从节点用于数据备份，若一个节点故障，数据不丢失；两个节点同

时故障时，有可能丢失；节点数越多，数据丢失概率越低。对数据完整度分析如表 3。

表 3　数据完整度随节点数变化

节点数	4	8	16	32
数据完整度	83.3%	96.4%	99.2%	99.8%

从上表可知，节点数越多，数据完整度越大，节点故障导致的数据丢失概率越小，当系统具有 8 个节点，两个节点同时故障也可保证 95%以上数据不丢失。通过设置数据备份个数来验证数据备份对数据交换速率的影响，实验数据表明，设置单备份和双备份与无备份相比数据交换速率分别降低 20%和 29%。因此，系统设计需合理选取数据备份数量，在数据吞吐率和可靠性指标之间折中。

6　结束语

根据遥感卫星数据处理系统实时性、数据可靠性的特点与需求，本文提出基于流式计算的遥感卫星数据快视处理系统设计方法，利用已有流式计算框架设计实现了通用的遥感卫星数据快视处理系统。与已有系统相比，本文的系统通过数据流拓扑结构研究与设计实现了细粒度的数据并行处理，采用数据备份机制为数据的可靠性提供了保障机制，同时提升了处理单元间的数据交换效率。通过实验分析，验证了改进系统在数据吞吐率和数据可靠性方面具有较好的效果。

目前的研究针对资源一号 02C 卫星开展，该卫星载荷类型多样，数据流特点具有代表性，后续仍需选取不同类型卫星开展进一步研究。本文将遥感卫星数据快视处理作为研究对象，作为一种系统设计方法也适用于实时性要求高的一般遥感卫星数传数据自动化处理过程，应用场景可扩展到 0-2 级数据产品生成、融合产品处理和深加工处理等方面。

参考文献：

[1]刘定生,陈元伟,李景山.遥感卫星地面预处理系统技术发展模式探讨[J].遥感信息:2008(5):87-91.

[2]孙小涓,雷斌,程兆运,等.遥感数据处理运行控制中的工作流应用[J]. 计算机工程:2012, 38(4):28-30.

[3]孙小涓,雷斌,胡玉新.科学工作流技术及在空间信息科学计算中的应用[C]//中国计算机学会.2013 全国高性能计算学会年会,无锡, 2013.无锡,2013: 601-609.

[4]徐业帷.科学工作流在空间信息处理领域的应用研究[D].北京:中国科学院大学, 2016.

[5]孙小涓,石涛,李冰,等.空间科学卫星数据快速处理方法[C]//中国计算机学会.2017 全国高性能学术年会,合肥,2017.合肥,2017:438-443.

[6]王峰.卫星混编数据地面接收快视系统的设计与实现[J].航天器工程, 2008, 17(6):44-48.

[7] 孙大为,张广艳,郑纬民.大数据流式计算:关键技术及系统实例[J].软件学报:2014, 25(4):839-862.

[8] 李圣,黄永忠,陈海勇.大数据流式计算系统研究综述[J].信息工程大学学报: 2016, 17(1):88-92.

[9]Simoncelli D, Dusi M, Gringoli F, et al. Scaling Out the Performance of Service Monitoring Applications with BlockMon[C]// Springer Berlin Heidelberg, International Conference on Passive and Active Network Measurement, Hong Kong, 2013: Hong Kong, 2013:253-255.

[10] 韩德志,陈旭光,雷雨馨,等.基于 Spark Streaming 的实时数据分析系统及其应用[J].计算机应用:2017,37(5):1263-1269.

[11] Neumeyer L, Robbins B, Nair A, et al. S4: Distributed Stream Computing Platform[C]//IEEE Computer Society, IEEE International Conference on Data Mining Workshops,Sydney.2010.Sydney.2010:170-177.

[12] 韩杰, 陈耀武. 基于 Storm 平台的实时视频分析系统[J].计算机工程, 2015, 41(12):26-29.

[13] 乔通.基于 Storm 的海量交通数据实时处理平台的研究[D]. 北京:北方工业大学, 2017.

[14]Fang L, Longlong D, Zhiying J, et al. Single-Pass Clustering Algorithm Based on Storm[C]//IUPAP Conference on Computational Physics. Journal of Physics Conference Series, 2017.IOP Publishing, UK, 2017:12-17.

[15]丁维龙,赵卓峰,韩燕波.Storm:大数据流式计算及应用实践[M].北京:电子工业出版社,2015:53-68.

[16] Chintapalli S, Dagit D, Evans R, et al. PaceMaker: When ZooKeeper Arteries Get Clogged in Storm Clusters[C]// IEEE, International Conference on Cloud Computing, Nicosia, 2017.Nicosia, 2017:448-455.

[17]Karunaratne P, Karunasekera S, Harwood A. Distributed stream clustering using micro-clusters on Apache Storm[J]. Journal of Parallel & Distributed Computing: 2017, 108:74–84.

[18]Apache Software Foundation. Apache Storm. [EB/OL]. USA:2018[2018].http://storm.apache.org/

[19]Esmaili K S, Esmaili K S. Kafka versus RabbitMQ: A comparative study of two industry reference publish/ subscribe implementations:Industry Paper [C] //ACM International Conference on Distributed and Event-Based Systems. ACM, 2017:227-238.

[20] IBM developer works. Efficient data transfer through zero copy, zero copy, zero overhead.[EB/OL].USA: 2018[2018].https://www.ibm.com/developerworks/library/ j-zerocopy

[21] Goodhope K, Koshy J, Kreps J, et al. Building LinkedIn's Real-time Activity Data Pipeline.[J]. IEEE Data(base) Engineering Bulletin: 2012: 33-45.