Journal of System Simulation

Volume 29 | Issue 1 Article 17

6-1-2020

Research of Parallel Process Method of Ground Penetrating Radar (GPR) Data Based on Hadoop

Yincheng Liang
School of Mechanical Electronic&Information Engineering, China University of Mining&Technology, Beijing 100083, China;

Yuan Yuan
School of Mechanical Electronic&Information Engineering, China University of Mining&Technology, Beijing 100083, China;

Yang Feng
School of Mechanical Electronic&Information Engineering, China University of Mining&Technology, Beijing 100083, China;

Research of Parallel Process Method of Ground Penetrating Radar (GPR) Data Based on Hadoop

Abstract

Abstract: The change of research way brought by big data and cloud computing has an impact on the research of GPR data processing. Filtering and analysis research for large scale GPR data was conducted by cloud computing platforms. *Data preprocessing to GPR data was realized through flow computing framework Storm so as to solve the mismatch of GPR data format and input data format of Hadoop and realize parallel convolution and gain process based on Hadoop. The filtering result and performance index was analyzed.* Experimental results show that the proposed method is accurate and effective on GPR data process, and it has significantly improved operating speed and parallel speed-up in comparison with previous method.

Keywords

Hadoop, GPR data, Storm, convolution, gain process, parallel

第 29 卷第 1 期 系统仿真学报© Vol. 29 No. 1
2017 年 1 月 Journal of System Simulation Jan., 2017

基于 Hadoop 的探地雷达数据并行处理方法研究

梁胤程，袁媛，杨峰
(中国矿业大学(北京) 机电与信息工程学院，北京 100083)

摘要：大数据及云计算等技术带来的科研方式的转变影响着探地雷达数据处理领域的研究工作，利用云计算平台对大规模探地雷达数据进行了数据解析的研究工作，*通过流式处理框架 Storm 平台对产生的探地雷达数据进行了数据预处理，解决了探地雷达数据格式与 Hadoop 平台数据处理输入格式不匹配的问题。并在 Hadoop 平台下实现了并行化的卷积、反卷积与增益相结合的数据滤波处理，对滤波效果和运行性能指标进行了分析。*仿真实验结果表明，该方法能准确有效地对探地雷达数据进行滤波解析，在运算速度、并行化加速度性能上都较传统方法有明显的改善。

关键词：Hadoop; 探地雷达数据; Storm; 卷积; 增益; 并行化
中图分类号：TP391 文献标识码：A 文章编号：1004-731X (2017) 01-0120-09
DOI: 10.16182/j.issn1004731x.joss.201701017

Research of Parallel Process Method of Ground Penetrating Radar (GPR) Data Based on Hadoop

Liang Yincheng, Yuan Yuan, Yang Feng
(School of Mechanical Electronic&Information Engineering, China University of Mining&Technology, Beijing 100083, China)

Abstract: The change of research way brought by big data and cloud computing has an impact on the research of GPR data processing. Filtering and analysis research for large scale GPR data was conducted by cloud computing platforms. *Data preprocessing to GPR data was realized through flow computing framework Storm so as to solve the mismatch of GPR data format and input data format of Hadoop and realize parallel convolution and gain process based on Hadoop. The filtering result and performance index was analyzed.* Experimental results show that the proposed method is accurate and effective on GPR data process, and it has significantly improved operating speed and parallel speed-up in comparison with previous method.

Keywords: Hadoop; GPR data; Storm; convolution; gain process; parallel

引言

探地雷达是利用高频电磁脉冲的反射来探测目的体的一种简便、快捷的电磁探测方法[1]。在采矿工程、道路病害检测、管线勘测、地质工程、岩土工程勘察、建筑工程、桥梁道路、隧道工程等领域有巨大的应用价值。当前的雷达信息采集系统采用多种信号采集方式获取现场的实时数据，通过不同来源数据的融合，辅助施工方做出合理有效的实施方案。探地雷达检测数据量不断增大，雷达数据的处理解析效率成为应用中亟待解决的问题。大数据和云计算技术的发展影响着数据挖掘分析领域的研究，也影响着探地雷达数据的并行分析处理，很多学者开展了用大数据和云计算技术对各自领域的海量数据分析处理研究工作，如文献[2]中用

收稿日期：2015-04-27 修回日期：2015-07-10；
基金项目：国家重大科学仪器设备开发专项(2012YQ 030126)，核三废专项科研课题(环 FZ1402-3)；
作者简介：梁胤程(1983-)，男，山东，博士生，研究方向为大数据处理；袁媛(1990-)，女，河北，硕士，研究方向为并行计算；杨峰(1968-)，男，河南，教授，研究方向为探地雷达仪器开发。

云计算技术对海量文本进行了并行化分类研究，文献[3]中用 map-reduce 计算框架对医学康复数据进行多模式数据挖掘研究，文献[4]进行了云计算技术对图像处理的应用研究，文献[5]进行了云计算技术对高铁监测数据的应用研究，如今在电商行业、生物信息和金融领域都可以看到大数据及云计算技术的存在。但当前国内外学者利用大数据及云计算技术进行探地雷达数据处理分析的研究工作比较少，也不是很成熟。探地雷达检测时，探测数据的不断累积，需要对数据进行快速准确有效地处理，而大数据云计算技术以其海量高速并行化的处理能力使其可以有效胜任这样的工作，本文正是基于这样的背景，进行了 hadoop 云计算环境[6]下对大规模探地雷达数据进行滤波分析及图像解析的应用研究。

1 基于 Hadoop 的雷达数据系统框架

传统的探地雷达数据都是以单机处理为主，探地雷达应用中，建立大数据平台是大势所趋。本文结合探地雷达数据特点设计并实现了一种基于 Hadoop 的雷达数据管理平台 GRBase。

GRBase 管理系统是一种基于 Hadoop 的分布式集群，其核心是 Master-Slave 架构的分布式集群来管理数据，采用分布式文件系统 HDFS[7]和 MySQL 的关系型数据库集群来解决结构化数据的海量存储和高效访问。系统采用 MVC 的 3 层功能结构表示，具体的系统功能分层如图 1 所示。

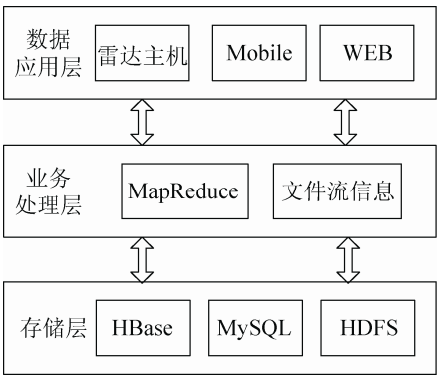


图 1 GRBase 系统功能分层
Fig.1 Function layers of GRBase system

第 1 层是数据应用层。用户可以操作雷达主机与数据管理平台进行系统交互，将采集的探地雷达数据进行上传，也可以通过 WEB 页面或者智能手机客户端对探地雷达数据进行查看。

第 2 层为业务处理层。该层主要运行 MapReduce 程序，实现探地雷达数据存储、雷达数据解析处理、病害特征提取以及平台维护相关的操作：数据上传/下载、日志收集、Hive sql 语句提交运行、集群间数据同步。

第 3 层为数据平台存储层。大数据平台的特点是多种存储方式并存实现良好的可扩展性和数据备份。所以在本层采用传统的关系型数据库和 Hbase，HDFS 等多种方式存储方式，采用 MySQL 数据库较好的解决了传统程序数据的移植问题，而 HDFS 文件和 HBase 数据库解决了多样化的数据存储格式问题。

2 探地雷达数据预处理方法

2.1 探地雷达数据格式

探地雷达数据的格式为：

数据文件包含采集参数和数据体，以“.dat”为文件名后缀，数据文件具体格式如下：

{
　　文件头；
　　数据体；
}

数据体说明如下：

{
　　第 1 道数据；
　　第 2 道数据；
　　……
}

每一道数据的每个样点采用短整型保存，每一道的前 4 个字节同时也是标记控制信息，如果为十六进制的 55aa，就是打标记录道。

2.2 MapReduce 输入格式

Hadoop 自带几个输入格式[8]。其中有一个抽

象类叫 FileInputFormat，所有操作文件的 InputFormat 类都是从它那里继承功能和属性。默认的输入格式是 TextInputFormat，它把输入文件每一行作为单独的一个记录。另一种输入格式是 KeyValueInputFormat，这个格式也是把输入文件每一行作为单独的一个记录，然而不同的是 TextInputFormat 把整个文件行当做值数据。KeyValueInputFormat 则是通过搜寻 tab 字符来把行拆分为键值对。还有一种输入格式是 Sequence FileInputFormat，它会读取特殊的特定于 Hadoop 的二进制文件，这些文件包含了很多能让 Hadoop 的 mapper 快速读取数据的特性。Sequence 文件是块压缩的并提供了对几种数据类型直接的序列化与反序列化操作。Squence 文件可以作为 MapReduce 任务的输出数据，并且用它做一个 MapReduce 作业到另一个作业的中间数据是很高效的。

2.3 探地雷达数据预处理方法

综合 2.1 探地雷达的数据格式及 2.2MapReduce 的输入格式不难发现，探地雷达数据无法直接应用 MapReduce 程序来处理。MapReduce 提供的默认输入接口由 FileInputFormat 来完成输入文件的切分，然后分发给各个 Map 任务，实现大量数据的并行操作。FileInputFormat 以”\n”为依据来切分输入文件，在雷达数据格式中不存在”\n”分隔符，所以无法采用默认的 TextInputFormat 输入格式来读取探地雷达数据。如何有效的切分雷达数据，既能符合 MapReduce 程序的处理，又不破坏雷达数据的整体性，便于雷达数据病害信息发现是用 Hadoop 平台处理探地雷达大数据的关键。针对此问题，本文提出了一种基于 storm 流处理的探地雷达数据预处理方法[9]。通过预处理将探地雷达数据进行预处理使之符合 MapReduce 程序的输入格式，而且还不破坏探地雷达数据的整体性，便于数据的并行化滤波处理及病害发现。

Storm 可以处理源源不断流进来的消息，处理之后将结果写入到某个存储中去，是一种适合实时处理的流式计算框架。所以 storm 系统非常适合进行探地雷达数据的预处理。针对 2.1 节中的探地雷达数据。为探地雷达数据增加”\n”分隔符，使之便于 MapReduce 程序对输入数据进行切割，实现 Map 程序的并行处理。同时在切割位置通过数据重叠保证数据整体性，防止切割位置出现病害信息，影响病害的发现。预处理的 topology 过程如图 2 所示。

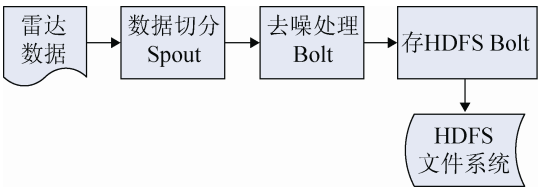


图 2 探地雷达数据预处理流程图
Fig.2 Flow chart of data preprocessing for GPR

Spout 首先读取雷达数据的文件头，从文件头中提取当前雷达数据每道数据的大小，舍弃该文件头。根据一道数据的大小，读取 500 道雷达数据，增加”\n”分隔符封装成一个 tuple，由于将文件切分，所以在封装 tuple 过程中增加标号，来标示该 tuple 在原文件中的道位置。由于标号的存在，通过在 MapReduce 处理过程的 sort 操作，可以得到排序的 tuple 组，根据该标号和道数可以精确定位潜伏病害位置。在封装 tuple 的时候，为了保持雷达数据完整性，每个 tuple 和前一个 tuple 重叠 20%的数据，防止病害信息正好在切分的 tuple 处的情况影响病害检测算法的效果。在去噪 Bolt 中会对病害数据进行平滑处理，去除噪声减小 MapReduce 作业的负担。最后通过一个存储 Bolt 将预处理完成的数据存储到 HDFS 中，便于 MapReduce 作业抽取数据。

3 探地雷达数据处理算法及实现

探地雷达仪器数据采集时，由于内外部各种因素的影响，会带有噪声信号，需要对探地雷达数据进行滤波处理。当前的雷达数据处理耗时比较长，处理效率比较低。所以通过并行化来提高雷达数据处理算法的效率具有深远的意义。探地雷达数据的

常规处理流程包括：1、一维滤波处理；2、二维滤波；3、小波变换；4、增益控制；5、反卷积运算。其中一维滤波、二维滤波和小波变换算法可变换为以卷积为基础的运算，本文将重点介绍卷积、反卷积和自动增益控制的算法实现。

3.1 探地雷达数据处理算法

如果雷达子波是未知的，则反卷积问题的解是统计性的。维纳预测理论提供了统计性反卷积的一种方法。对于没有噪声的卷积模型为：

$$x(t) = w(t) \times e(t) \tag{1}$$

如果一个卷积算子 $a(t)$ 这样定义，$a(t)$ 与已知的雷达记录卷积产生一个对地层脉冲响应 $e(t)$ 的一个估计，则：

$$e(t) = a(t) \times x(t) \tag{2}$$

将(2)式带入(1)式得到：

$$x(t) = w(t) \times a(t) \times x(t) \tag{3}$$

将 $x(t)$ 从两边消去，得到下列表达式：

$$\delta(t) = w(t) \times a(t) \tag{4}$$

式中：$$\delta(t) = \begin{cases} 1 & t = 0 \\ 0 & \text{其他} \end{cases} \tag{5}$$

对于算子 $a(t)$ 求解，根据(4)式可以得到：

$$a(t) = w'(t) \times \delta(t) \tag{6}$$

$w'(t)$ 是雷达子波 $w(t)$ 的逆。由此可以知道，反卷积运算的关键是要计算出雷达子波及其的逆。雷达子波提取算法很多，最简单的可以采用自相关方法提取。

自动增益的目的是使雷达剖面上各有效波的能量均衡，这种处理便于有效波的追踪，也利于弱信号的对比。

自动增益依靠雷达记录乘以随时间变化的增益权函数来实现，即：

$$\tilde{y}(t) = P(t)y(t) \tag{7}$$

式中：$y(t)$为自动增益前的雷达记录；$P(t)$为自动增益权函数；$\tilde{y}(t)$ 为自动增益后的雷达记录。

对于能量大的反射信号，所乘的权因子应该小；对于能量小的反射信号，所乘的权因子应该大。为了使反射波的记录不会发生失真，权函数因子随时间的变化应该比较缓慢。

为了计算增益权函数 $P(t)$，首先将整个时间窗分割成若干个等时间的小窗口，利用小窗口能量大小来确定控制点的增益大小。

由参数控制点数确定时间窗的数量，在计算平均振幅时，每两个相邻时间窗之间要重叠半个时间窗。因此，根据控制点数，需要计算以下参数：时间窗长、时间窗的起始时间、时间窗的终止时间。

因此，第 i 个时间窗内的平均振幅计算公式：

$$A_i = \frac{\sum_{t=T_{i-1}}^{t=T_{i+1}} |y(t)|}{N} \tag{8}$$

式中：$y(t)$为要处理的雷达采集的单道信号；T_{i-1} 为第 i 个时窗的起始时间；T_{i+1} 为第 i 个时窗的终止时间；N 为第 i 个时窗的样点数；A_i 为第 i 个时窗的平均振幅。把每个时窗的平均振幅对应于各自的时窗中心作为控制点的增益参数存放起来。

加权函数为：

$$P_i = \frac{M}{A_i} \tag{9}$$

式中：A_i 为第 i 个时窗的平均振幅；p_i 为第 i 个时窗中心对应的加权因子；M 为用于调整处理后有效振幅大小的平衡系数。对于那些非时窗中心各点的加权因子，可以利用相邻两个时窗中心的加权因子线性内插得到。

3.2 探地雷达数据处理算法实现

3.2.1 卷积实现

Map 阶段

Map 阶段的任务是将预处理后的数据逐行读取，接着以“0X55AA”为分隔符将数据按道次拆分，接着对每道数据 data[]利用用户设定的卷积因子 value_data[]进行卷积操作，最后将每一行卷积结果拼接成雷达文件预处理之后的格式，并将行号和拼接的数据以(<key, resultLine >)键值对的形式存入 context 集合。Map 阶段伪码如下：

输入：< Object key, Text value>

输出：<IntWritable, Text> Context

Begin

//拆分 value 得到待转换的每道采样点数据 data 数组

//对每一道数据进行卷积、反卷积操作

return_data = conv_code(value_data, value_data.length, data, data.length, value_data.length + data.length - 1);

//将数据的卷积结果构建一个字符串

for all k in return_data.length do

resultLine += second[k] + ",";

Endfor

//增加道间标识

resultLine += "0X55AA";

context.write<key , new Text(resultLine));

End

Ruduce 阶段

Ruduce 阶段的任务是对 Map 函数输出的集合中的键值对重新洗牌(shuffle)和排序(sort)，从上一阶段获得(<key, resultLine >)键值对，将属于同一个 key 值的 value 值重新按照雷达数据文件的格式进行拼接，同时通过对行号的排序将各行排列组合，形成最终输出的雷达文件。Reduce 阶段的伪代码如下：

输入：< IntWritable key, Iterable<Text> values >

输出：< NullWritable, Text > Context

Begin

for (Text val : values) do

resultLines += val.toString();

Endfor

context.write(NullWritable.get(),new Text(resultLines));

End

3.2.2 自动增益实现

Map 阶段

Map 阶段的任务是将预处理后的数据逐行读取，以“0X55AA”为分隔符将数据拆分并存入存储了该行所有采样点数据的数组 point[]，接着对每道数据利用用户设定的增益因子进行增益操作，对最后一道数据进行延拓操作，最后将增益结果拼接成雷达文件预处理之后的格式，并将行号和拼接的数据以(<key, resultLine >)键值对的形式存入 context 集合。Map 阶段伪码如下：

输入：< Object key, Text value>

输出：<IntWritable, Text> Context

Begin

//拆分 value 得到待转换的 point 数组

// 获取每行包含的道数 sn

//循环对每一道数据进行增益操作

for (int j = 0; j < 1023; j++) do

for (int k = 0; k < sn; k++) do

//自动增益

point[j * sample_num / 1024 + k] = (10 * ((sn - (float) k)/ sn * value_data[j] + k / sn * value_data[j + 1]) *point[j* sample_num / 1024 + k]);

Endfor

Endfor

//对最后一道数据进行延拓操作

for all k in sn do

point[2 * sample_num / 1024 + k] = (10 * ((sn - (float) k)/ sn * value_data[1023] + k / sn * value_data[1022]) * point[1023* sample_num / 1024 + k]);

Endfor

//将数据的增益结果构建一个字符串 resultLine，并赋值给 context

context.write<key, new Text(resultLine));

End

Ruduce 阶段

Ruduce 阶段的任务是对 Map 函数输出的集合中的键值对重新洗牌(shuffle)和排序(sort)，从上一阶段获得(<key, resultLine >)键值对，将属于同一个 key 值的 value 值重新按照雷达数据文件的格式进

行拼接，同时通过对行号的排序将各行排列组合，形成最终输出的雷达文件。Reduce 阶段的伪代码如下：

输入：< IntWritable key, Iterable<Text> values >

输出：< NullWritable, Text > Context

Begin

for (Text val : values) do

resultLines += val.toString();

endfor

resultLines=key+":"+resultLines.substring(4);

context.write(new Text(resultLines));

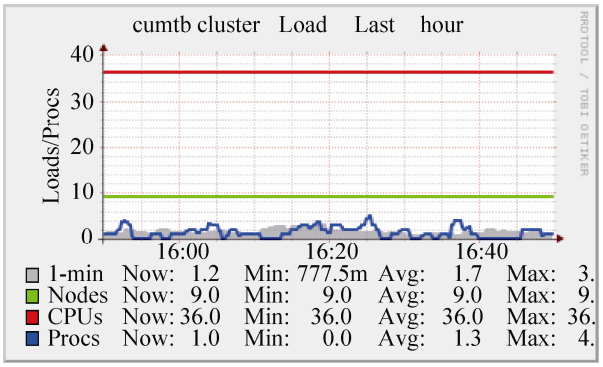
End

4 实验结果与分析

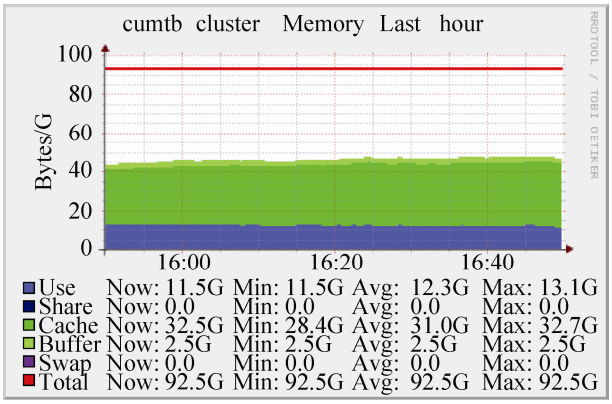
实验环境为机房 9 台 PowerEdge R720 计算机。主机的配置：CPU 为 Xeon E5-2620，内存为 16 GB DDR3，硬盘大小为 1 TB。其中 4 台服务器组成 Hadoop 集群，5 台组成 Storm 集群。全部的服务器均有 ganglia 来监控，Hadoop 集群中 1 台服务器为 NameNode 结点，其余 3 台为 DataNode 结点。Storm 集群 1 台服务器为 Nimbus 结点，其余 4 台服务器为 supervisor 结点。Hadoop 版本为 hadoop-2.2.0。实验数据为 100 g 的原始探地雷达数据。

4.1 算法性能比较

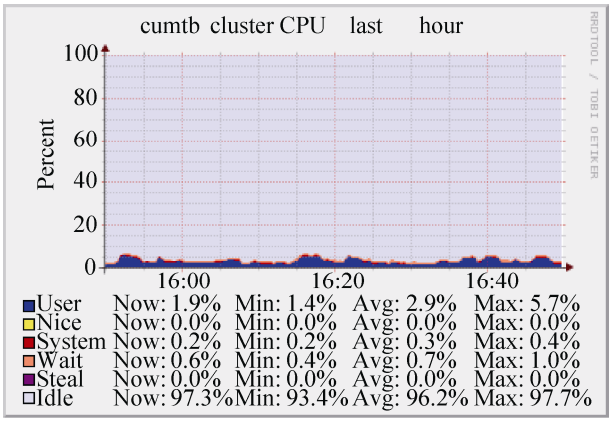
通过 Ganglia 监控软件对集群运行时的状态进行监控[10]，当 1 g 探地雷达数据进行处理时，整个集群的运行状态如图 3 所示。实验集群只进行探地雷达数据处理相关的运算，从图中可以看出程序运行时服务器的性能指标良好。当处理程序运行时，集群在负载、内存利用和 CPU 利用率的性能指标保持一个较稳定的状态，图中曲线略有起伏，但部署的集群远远可以满足程序的运行需求。当预处理程序运行时，集群的网络吞吐量会有明显增加，运行 1 g 雷达数据的处理程序，网络吞吐量的峰值达到 7.3 M/S。



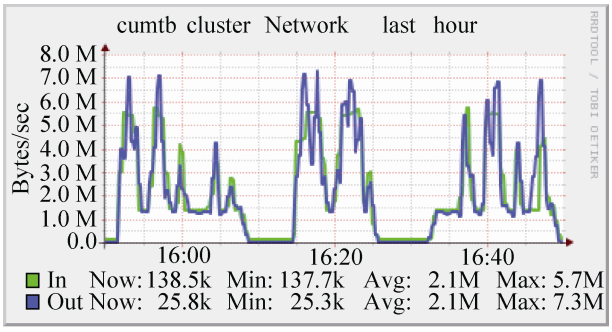
(a) 负载性能图



(b) 内存性能图



(c) CPU 性能图



(d) 网络吞吐性能图

图 3 雷达数据处理时集群性能参数图

Fig.3 Cluster performance diagram of GPR data process

https://dc-china-simulation.researchcommons.org/journal/vol29/iss1/17
DOI: 10.16182/j.issn1004731x.joss.201701017

第 29 卷第 1 期 系统仿真学报 Vol. 29 No. 1
2017 年 1 月 Journal of System Simulation Jan, 2017

为了比较本文雷达数据处理系统与单机雷达解析软件在效率上的差别，实验测试了集群和单机对雷达数据运行先卷积再增益的运行时间和 Speedup 指标。Speedup 表示在固定任务处理量不变的情况下，依次增加计算节点数量后系统运行时间的对比度变化。实验中当 1 g 探地雷达数据进行处理时，单机和集群环境下的 5 次实验结果均值效果进行了比较，时间对比和 Speedup 对比分别如表 1 和图 4 所示。当节点数目较少时，计算时间体现不出集群的优势，但当节点数目超过 4 个甚至更多时候，执行效率也越来越高，理论上集群可以不断扩展以提高程序执行效率，直到节点间数据通信成为瓶颈，由于试验环境的限制，本文无法给出进一步证实。根据已有实验结果，基于 Hadoop 的雷达数据管理系统一定程度上解决了单机数据随着数据量的增加执行效率呈指数衰减的问题，相比传统的探地雷达数据处理方法提高了数据处理的效率。

表 1 不同数量节点集群处理雷达数据时间对比
Tab.1 Time consumption of GPR data process /s

机器节点数	1g 雷达数据	10g 雷达数据
单机	2 529	18 462
2 节点集群	2 116	15 385
4 节点集群	1 410	10 256
6 节点集群	1 058	8 391

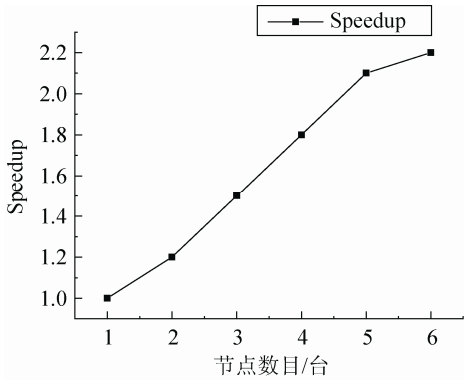


图 4 Speedup 对比
Fig.4 Comparison of Speedup

4.2 探地雷达数据并行处理结果分析

雷达数据经过预处理后的结果如图 5 所示。通过增加”\n”分隔符将数据切分成 500 道数据一条的记录，在后续的 MapReduce 程序中，每个 Map 的输入分片即为预处理切分的 500 道雷达数据。同时为了标识隐藏病害信息的位置，对切分的雷达数据进行增加标号处理。

为了测试基于 Hadoop 的雷达数据处理的准确性，分别对原始数据、传统雷达数据处理软件处理的数据和经过本文系统处理后的雷达数据经过 matlab 仿真软件进行成图显示，如图 6 所示，其中(a)为原始雷达数据，(b)为传统单机处理软件经过反卷积和增益处理后的成像结果，(c)为本文系统处理后的数据显示。经图 6(b)和(c)的对比分析，基于 Hadoop 的雷达数据处理程序和传统单机软件处理程序效果基本一致，经过并行卷积后的雷达波曲线比较平滑，达到单机处理软件相同的效果。

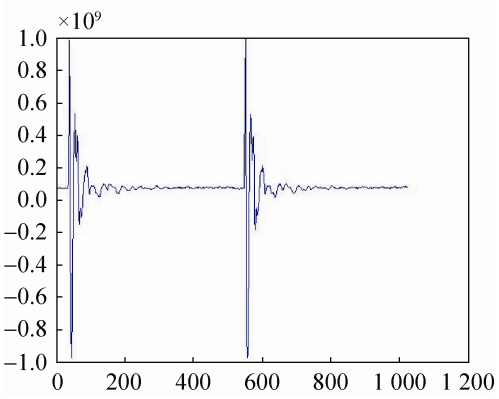
图 7 为探地雷达波数据先经过并行卷积处理再经过并行增益处理的实验效果图。其中卷积因子从离散高斯函数获得，增益因子为线性增益 $at+be^{ct}$，参数 a 为 6.1，b 为 0.006，c 为 0.055。雷达波每道数据采样时间窗为 200 ns，实验中增益控制点数选为 10。图 7 中(b)为通过卷积处理对雷达波干扰信号进行去除同时对低频漂移进行压制的处理效果。图 7 中(c)为平衡雷达有效波的能量均衡的增益效果。对能量大的反射信号乘以较小的权因子，对能量弱的反射信号乘以较大的权因子。雷达解析平台对处理后的雷达数据进行成像显示结果如图 8 所示。图中数据为管径 200 mm，顶深 1.1 m 上水管上探测处理结果。钢筋属良导体，对垂直入射的电磁波的能量消耗很少，几乎都被导体表面反射回来。所以金属目标的反射波幅度一般都大于空洞或其他目标物。从图中可以看出，基于 Hadoop 的并行处理后的图像较好的表现了不同介质的回波反射数据，成像效果边界清晰，便于隐藏病害信息发现和标识。

output41.txt

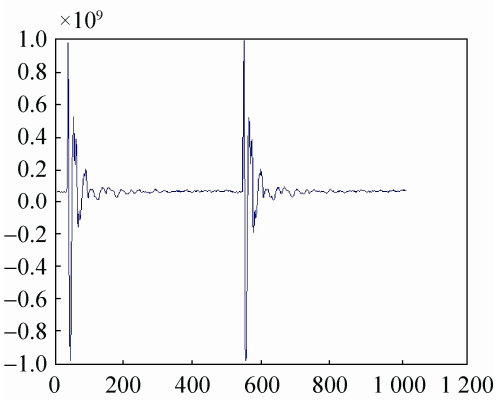
```
1:16711935,65274111,66716659,66061309,65864678,62260143,63570889,66716623,68748305,66454531,67044371,67306479,67699747,64750587,64095194,61408173,64422
2:73204825,66454582,66651122,66061281,66847735,63439828,67437536,65668085,66323435,63898593,68879354,67961877,69338142,65143795,68027378,64881651,67044
3:71107656,71042110,67306518,62981068,60949437,59704206,63177675,66323387,67699729,70190078,69993536,69600302,68224061,65668068,63308788,64357319,66585
4:71435309,70714444,68092958,65143802,61735889,62456725,64160712,66847714,68158467,67896348,67568636,68944910,67961885,66651122,63243206,64685011,66716
5:68879381,70059063,66454537,66913288,63308778,61473683,61342651,64160688,65602541,66520049,64947194,66192368,65733616,65274852,67634178,66978786,66323
6:71500847,71173215,68682786,65078275,64357341,63112138,61998039,65078200,65668076,69207025,68617234,68486163,65405954,65995756,62260191,62981055,64553
7:68617262,66323455,61670360,62260127,62981087,65668029,66454546,69731329,67961901,67437561,67699716,65012686,65602539,66257894,62587850,63701959,69469
8:68486148,68879394,65733665,60294052,60752817,61604780,63833029,67568623,69534736,68224027,65340408,64947181,64029665,64226285,61866967,64029642,66323
9:66716673,68748337,68682771,63177724,60294080,61866910,61604785,67109853,67699716,69600272,66978848,68682747,66520078,64881656,62194632,63439796,64226
10:66192342,66585607,67503104,64947167,64685020,62653382,65733579,63374304,67175395,63636472,64422861,65733614,68092909,66126866,68158454,66716681,66526
11:73270325,72025194,67961879,66257951,64422875,63308772,63046587,64685021,61998015,64226249,63112133,67568617,66323452,68355069,67765256,68289545,65012
12:69338214,70321185,65274899,66388982,65012698,65733585,65864689,69731323,67109902,64291805,64291804,65471465,63964152,67765202,67568674,70255635,67961
```

图 5 雷达数据预处理结果图

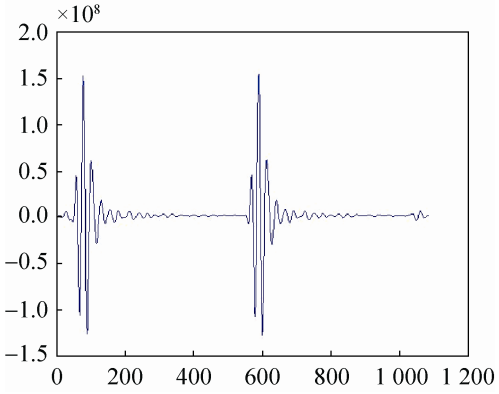
Fig.5 Result of GPR data preprocessing



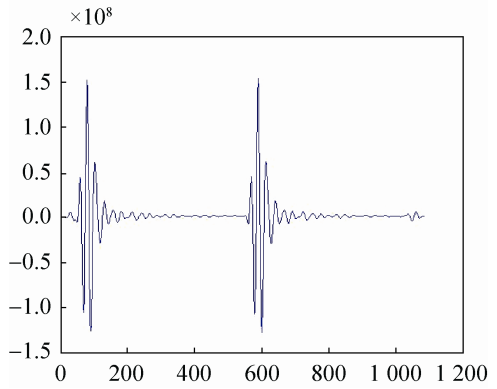
(a) 原始探地雷达数据



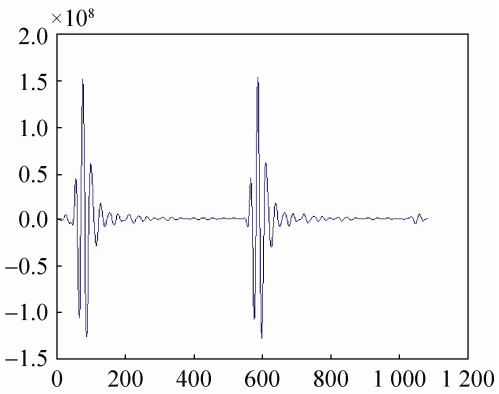
(a) 原始探地雷达波数据



(b) 单机软件处理后的雷达数据



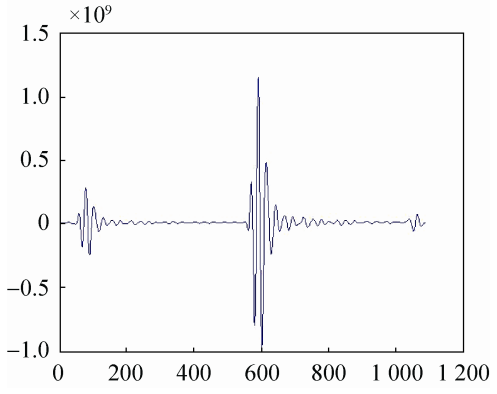
(b) 并行卷积处理后的雷达波



(c) 并行处理后的雷达数据

图 6 单机卷积结果和并行卷积结果仿真示意图

Fig.6 Simulation of convolution results



(c) 并行增益处理后的雷达波

图 7 雷达波先卷积后增益仿真示意图

Fig.7 Simulation of convolution and gain

https://dc-china-simulation.researchcommons.org/journal/vol29/iss1/17
DOI: 10.16182/j.issn1004731x.joss.201701017

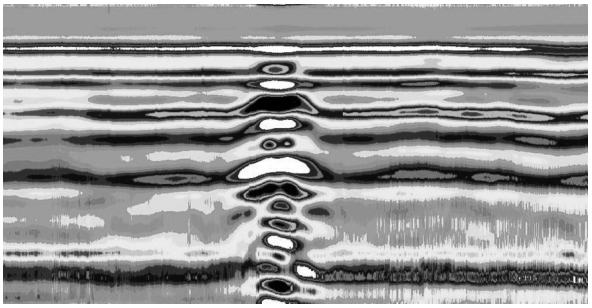


图 8 雷达数据解析成像图
Fig.8 Image of GPR data after processing

5 结论

本文对探地雷达数据处理进行了有别于传统单机模式的并行化处理，首次提出并实现了基于 Hadoop 的探地雷达数据处理，证实了该系统对大规模的探地雷达数据处理速度和性能有了很大的提升，另外对基于 Hadoop 的雷达数据处理的正确性做了分析，并与传统单机处理方式进行了对比研究。本文的工作开启了探地雷达数据处理新方法和新思路的积极探索，后续的工作将是进一步优化并行化算法，探索大数据框架下雷达病害信息发现算法的并行化处理。

参考文献:

[1] 杨峰, 彭苏萍. 地质雷达探测原理与方法研究 [M]. 北京: 科学出版社, 2010.(Yang Feng, Peng Suping. Research on principle and method of geological radar detection [M]. Beijing, China: Science Press, 2010.)

[2] Xiang X, Gao Y, Shang L, et al Parallel Text Categorization of Massive Text Based on Hadoop [J]. Computer Science(S1002-137X), 2011, 38(10): 184-188.

[3] Wang Fei, Vuk E, David B, et al. Large-scale multimodel mining for healthcare with MapReduce [C]// Proceedings of the 1st ACM International Health Informatics Symposium, USA: ACM, 2010.

[4] Malewicz Grzegorz, Austern Matthew H, Bik Aart J C, et al. Pregel: A system for large-scale graph processing [C]// Proceedings of the SIGMOD, USA: ACM, 2010: 135-146.

[5] 李贵兵, 金炜东, 蒋鹏, 等. 面向大规模监测数据的高铁故障诊断技术研究 [J]. 系统仿真学报, 2014, 26(10): 2458-2464. (Li Guibing, Jin Weidong, Jiang Peng, et al. Research on Fault Diagnosis Technology of High-speed Rail for Large Scale Monitoring Data [J]. Journal of System Simulation, 2014, 26(10): 2458-2464.)

[6] Gu T, Zuo C, Liao Q, et al. Improving MapReduce Performance by Data Prefetching in Heterogeneous or Shared Environments [J]. International Journal of Grid and Distributed Computing (S2005-4262), 2013, 6(5): 71-82.

[7] Shvachko K, Kuang H, Radia S, et al. The hadoop distributed file system [C]//Mass Storage Systems and Technologies (MSST), 2010 IEEE 26th Symposium on. USA: IEEE, 2010: 1-10.

[8] 周敏奇. Hadoop 权威指南 [M]. 北京: 清华大学出版社, 2011: 198-215.(Zhou Minqi. Hadoop the definitive guide [M]. Beijing: Tsinghua University Press, 2011: 198-215.)

[9] Jonathan Leibiusky, Gabriel Eisbruch, Dario Simonassi. Getting started with storm [M]. Sebastopol: O'reilly Press, 2012.

[10] Massie M L, Chun B N, Culler D E. The ganglia distributed monitoring system: design, implementation, and experience [J]. Parallel Computing (S0167-8191), 2004, 30(7):817-840.

(上接第 119 页)

[3] 左洪福. 航空维修工程学[M]. 北京: 科学出版社, 2011: 294-325. (Zuo Hongfu. Aviation Maintenance Engineering [M]. Beijing, China: Science Press, 2011: 294-325.)

[4] Guan Song, Hua Keqiang, Huang Yuming. Control modeling of an aircraft spare parts inventory and the optimal study [C]// Control and Decision Conference (978-1-4244-8737-0). China: IEEE, 2011: 3644-3646.

[5] 金元郁, 崔红. 改进的最小方差自校正控制算法 [J]. 青岛科技大学学报, 2005, 26(2): 45-48. (Jin Yuanyu, Cui Hong. Modified Minimum Variance Self-tuning Control Algorithm [J]. Journal of Qingdao University of Science and Technology, 2005, 26(2): 45-48.)

[6] MARODE OLIVIER, ARNAUD DENIS. Inventory control and optimization [P]: US8099294.

[7] 温创新. 基于大数据泊松分布的配件预测模型分析与建模 [J]. 计算机与数字工程, 2014, 42(8): 1412-1414. (Wen Chuangxin. Fitting Prediction Model Analysis and Modeling Based on Bulk Data and Possion Distrition [J]. Computer and Digital Engineering, 2014, 42(8): 1412-1414.)