计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2018,27(9):74−80 [doi: 10.15888/j.cnki.csa.006551]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

基于 Spark 的油田应用日志行为分析系统①

陈雷鸣, 张伟光, 李翛然, 李宁宁

(中国石油大学 (华东) 计算机与通信工程学院, 青岛 266580)
通讯作者: 陈雷鸣, E-mail: chenleiming1930@sina.com

摘　要: 随着油田信息化建设的不断发展, 越来越多的 IT 业务系统在油田各级单位普及应用. 由于油田应用数量庞大、种类复杂, 如何快速评估各类系统的运行情况和安全状况成为油田关注的重要问题. 在使用这些应用系统的同时, 一些访问信息会以日志的形式储存下来, 因此通过分析日志数据可以挖掘出用户访问喜好, 发觉业务系统潜在的安全问题, 进而为油田应用评估提供决策依据. 然而随着 IT 业务访问量剧增, 应用日志的数量、容量也随之增加, 仅依靠单机环境对海量数据进行分析已经无法满足油田业务需求. 针对这个问题本文提出了基于 Spark 计算框架的应用日志行为分析方法, 同时设计了可视化平台完成对整个分析系统的管理.

关键词: 应用日志; 行为分析; Spark; 可视化平台; 分析系统管理

引用格式: 陈雷鸣,张伟光,李翛然,李宁宁.基于 Spark 的油田应用日志行为分析系统.计算机系统应用,2018,27(9):74–80. http://www.c-s-a.org.cn/1003-3254/6551.html

Oil Fild Application Log Behavior Analysis System Based on Spark

CHEN Lei-Ming, ZHANG Wei-Guang, LI Xiao-Ran, LI Ning-Ning

(College of Computer & Communication Engineering, China University of Pertroleum, Qingdao 266580, China)

Abstract: With the rapid development of Internet information construction, more and more IT systems are widely used. Due to the huge amount and complexity of oilfield applications, how to quickly evaluate the operation and safety status of various systems has become an important issue in oilfield. When using the business system, some access information was recorded in the form of logs at the same time. By analyzing the log data, the user's access preferences can be excavated and the potential network security problems of the business system can be found, thus providing a decision basis for the evaluation of oil field applications. However, with the rapid increase of business access, the amount and storage capacity of logs also increase. Relying on single computer environment, analyzing massive log data has been unable to meet the needs of applications. In view of this problem, this study proposes a log behavior analysis method based on Spark calculation framework and designs a service platform for visual management based on Web.

Key words: application log; behavior analysis; Spark; visual platform; management of analysis system

随着油田信息化、智能化建设的不断加快, 各类 IT 系统的在企业中广泛应用. 某油田现有超过一千套业务系统分别由各级单位独立运维管理. 在这些业务系统给企业提供便捷服务的同时, 如何对这些业务进行监控分析和安全评估上却面临难题. 由于油田现有的应用系统数量庞大、类型繁杂、开发技术多样、部署分散, 如何以最小的切入方式完成对这些应用的运行状况和安全状况的评估成为企业关注的重要问题. 由于这些应用系统和网站每天都会产生大量的日志数据, 这些日志中包括用户的访问信息和应用安全状况

① 收稿时间: 2018-01-06; 修改时间: 2018-01-23, 2018-02-06, 2018-03-13; 采用时间: 2018-03-26; csa 在线出版时间: 2018-08-16

等信息. 通过分析应用日志数据可以评估应用的使用情况、应用运行的安全状况, 进而为各企业信息决策提供重要依据. 随着各类应用系统的规模迅速扩大导致应用所产生的日志数据呈爆炸式增长, 若继续采用传统的数据储存和处理方式将无法及时评估出各类业务运行情况和安全状况.

针对这一难题, 主流的海量日志处理方案是借助于大数据计算框架提供的分布式处理技术. 大数据技术的发展大致可分为以下阶段: 第一阶段是基于 Hadoop 提供的 MapReduce 计算框架做分析. 由于 MapReduce 的编程机制需要严格按照 Map 和 Reduce 两个阶段, 因此缺少了程序设计的灵活性[1,2]. 然后是 Pig[3]数据分析程序以及 Hive[4]数据仓库等工具的出现. 这类工具简化了 MapReduce 的编程过程, 然而在任务执行时依然需要先转换为 MapReduce 作业任务然后再交给 Hadoop 执行[5]. 由于 Hadoop 在处理大批量数据时, 需要把中间结果缓存到磁盘上, 这一过程受限于磁盘 IO 速率, 因此严重影响分析效率. 针对这一问题, 基于内存计算的批处理框架 Spark 应运而生, 由于 Spark 将数据直接保存在内存中进行多次迭代操作[6], 从而不再从磁盘中重复的读写数据源, 因此具有更快的处理速度. 本文基于 Spark 内存计算框架来代替 MapReduce 计算框架来提高计算速率, 并基于 Spark 提供的各类功能模块设计数据分析算法来完成应用日志数据的预处理和行为分析.

1 Spark 数据分析平台

Apache Spark 是由加州大学伯克利分校 AMP 实验室开发的分布式并行计算框架. Spark 支持复杂的机器学习、图计算和实时流处理等功能模块[7]. 如图 1 所示为 Spark 生态圈, 从下至上依次为: 数据持久层、资源调度层、Spark 核心计算层、Spark 主要功能组件. 其中数据持久层: 包括分布式文件系统 HDFS 和分布式数据库 HBase、Cassandra. 资源调度层: 为 Spark 提供统一的资源调度和管理, 目前主流的资源调度组件为 Yarn 和 Mesos.

Spark 核心层: 包含 Spark 的基本功能, 定义了 RDD 的 API 和基本操作, Spark 其它的功能模块都是构建在 RDD 和 Spark Core 之上的. 最后一层为 Spark 主要的功能组件包括: 用于对流数据实时处理的 Spark Streaming、用于机器学习的算法库 MLlib、用于图操作的算法工具集合 GraphX 和用于在内存数据集上提供查询功能的 Spark SQL. 本文基于 Spark 提供的基本算子函数操先对日志数据进行预处理并利用 Spark SQL 模块对预处理后的数据进行指标分析.

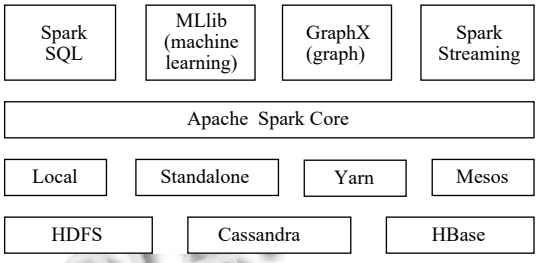

图 1 Spark 生态圈

2 系统架构设计

根据油田应用系统部署分散的特点, 设计应用行为分析平台架构如图 2 所示. 应用行为分析平台主要由数据收集层、数据存储层、数据处理层、数据可视化层、可视化管理调度平台 5 部分组成.

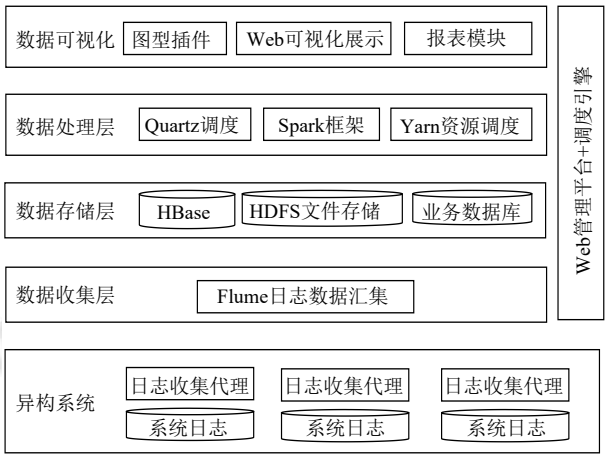

图 2 应用日志行为分析平台架构图

数据收集层: 用于将分散在各主机应用日志数据集中收集, 该模块基于 Flume 日志收集框架设计完成.

数据储存层: 日志文件储存在基于 HDFS 的文件储存系统上, 基于 HBase 储存经预处理分析后的结构化数据, 使用 MariaDB 作为业务数据库, 用于储存分析的最终结果和系统业务数据.

数据处理层: 基于 Quartz 任务调度框架[8]来完成各类任务的定时执行; 基于 Yarn 来完成集群环境下计算资源的分配和 Spark 任务调度. 基于 Spark 计算框架设计数据的预处理和数据分析算法.

数据可视化层: 用于分析结果的图表的可视化展示. 其中数据的图形化展示基于 Echarts 可视化插件来完成, 图表的数据通过报表程序模块来完成.

3 系统设计与实现

应用日志行为分析平台需要完成以下功能: 系统的可视化管理、各类计算框架的集成管理、分析算法的调度管理, 因此需要设计以下三个主要的模块: 基于 Web 管理平台、调度引擎和算法数据库.

Web 管理平台: 向用户提供交互功能和分析结果的可视化展示, 该模块基于 SSM 框架完成, 用于分析任务的管理、分析错误告警信息的管理、算法库管理、各类应用信息的管理以及与平台业务相关功能.

调度引擎: 该模块基于 Apache Felix[9]的 OSGI 框架开发完成, 主要完成不同数据源的数据信息拉取储存、数据处理分析模块的调度、分析任务定时执行, 该模块主要利用各类大数据框架提供的 API 封装成相应的功能模块集成开发完成.

算法信息库: 用于储存与日志行为分析的算法, 算法主要基于 Java 编程语言开发, 每个算法为单独 jar 包, 由调度引擎选择并提交到 Spark 集群执行. 行为分析系统各类组件的调度流程为:

第一步: 通过在应用服务器上安装日志收集代理 (Flume Agent), 将分散在各应用服务器的日志文件定时汇集到日志储存服务器, 然后经 Flume 框架上传到 HDFS 文件储存系统中规划的文件夹.

第二步: 由调度引擎执行定时任务、调度各类框架. 并由调度引擎选取各类算法提交给 Spark 集群.

第三步: Spark 集群从 HDFS 拉取数据, 首先对日志数据预处理, 并将结果反馈给调度引擎. 若处理过程无异常, 则将分析结果储存到 HBase 数据库.

第四步: 由调度引擎依次进行各类行为分析算法的调度, 并将分析结果储存到相关数据库中.

3.1 日志数据的预处理

油田应用日志的特点为: 业务量较小的应用每天生成一个日志文件, 大业务量的应用日志可能会被切分成多个日志文件 (在分析处理时若应用每天产生多个日志文件则逻辑上当作一个文件处理). 分析系统需要处理前一天所有应用产生的日志文件. 因此调度引擎模块会在每天 0 点开始执行总的分析任务. 每分析一个日志文件就执行一次预处理算法任务. 在日志的预处理分析顺序上, 调度引擎会根据传输到 HDFS 的日志文件顺序, 按照先来先服务的原则生成任务执行列表, 然后依次对各日志进行预处理分析.

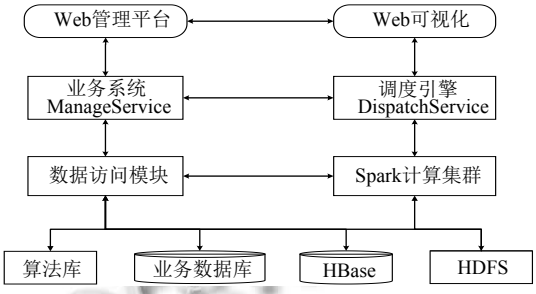

图 3 系统模块调度流程图

由于油田在部署各类应用系统时使用的服务软件种类繁多, 主流的服务软件包括: iis、tomcat、apache、nginx 等. 不同类型的服务软件产生的日志类型不一样; 同类型的服务软件可能有多个版本. 因此需要设计多种分析规则来处理不同类型的日志. 设计的原则为: 面向同类型日志分别设计相应的处理规则. 其中应用日志预处理分析流程如图 4 所示.

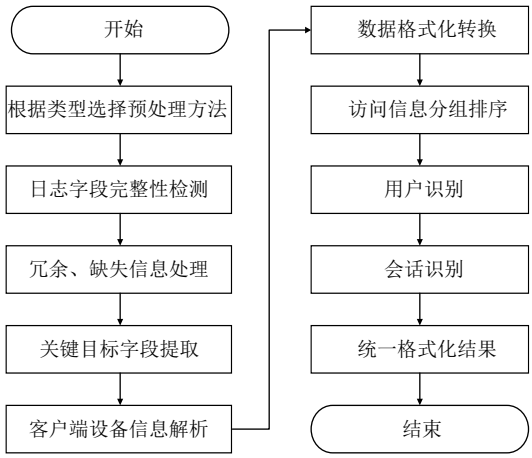

图 4 应用日志预处理分析流程

结合各类应用特点和部署环境等因素, 数据预处理过程可分为以下阶段: 数据清洗、用户识别、会话识别、数据格式化[10]. 数据清洗阶段主要完成对残缺信息的过滤 (字段缺失、信息缺失)、冗余信息的过滤 (主要过滤掉与请求无关的静态数据文件如 JS 文件、CSS 文件、图片数据等)、核心字段的提取. 然后根据客户端 IP 地址将访问信息按照时间先后顺序分组排序. 在用户识别阶段, 采用的是 IP 和客户端组合方式

来识别. 分析规则为: 不同的 IP 为不同用户, 同一 IP、不同客户端为不同用户. 在会话识别阶段, 采用的是基于固定阀值会话识别算法 (固定阈值为 30 min)[11]. 为了便于下一阶段进行应用行为分析, 需要对多种日志类型预处理后的结果各字段进行数据格式统一, 最后将处理后的数据储存到 HBase 数据库中.

预处理算法的设计主要基于 Spark Core 模块提供的操作 RDD 的算子实现. RDD 是 Spark 计算框架提供的分布式数据架构及弹性数据集, 它会在集群环境中的多个节点进行数据分区, 但是在逻辑上可看成一个分布式数组[12]. 预处理算法的设计原理: 主要利用 Spark 提供的各类算子设计相应函数, 从而实现对各类 RDD 的操作; Spark 最终会将者一系列对 RDD 的操作翻译成有向无环图 (DAG) 的形式进行调度和分布式任务分发[13]; 最终整个执行过程会形成预处理分析算法. 根据分析流程设计分析预处理算法: 首先日志文件数据会由 spark 读取加载到内存, 并将源数据转变成分布式数据集; 然后按照各阶段目标, 设计并实现相应算法模块或者基于各类算子设计相应的函数实现对已有的 RDD 进行转变操作. 应用日志预处理算法主要的分析步骤如算法 1.

算法 1. 数据预处理算法

1) 根据日志类型选取处理方法[A|B|C…].
2) 利用 textfile() 函数将日志文件加载到内存, 并转换为可操作的 RDD 数据集.
3) 调用字段检查函数对数据字段完整性检查,对字段完整的数据利用 map() 算子实现数据类型转换.
4) 使用 filter() 算子对 url 字段数据过滤, 去除与访问请求无关的数据以及自动加载的静态资源数据等.
5) 利用 map() 算子提取与分析目标相关的核心字段.
6) 调用设备解析算法模块对 agent 字段进行解析, 解析出客户端的设备、操作系统、浏览类型版本等信息.
7) 使用 sortByKey() 算子按照 IP、时间将访问记录排序.
8) 调用用户识别函数对数据处理.
9) 基于固定时间间隔会话识别算法, 划分用户会话{cuserID|(pid,time,url1,url2…)}.
11) 调用 map() 算子对数据格式进行归一转换.
12) 调用数据储存模块将数据储存到 HBase 数据库.

3.2 应用系统的安全分析与行为分析

油田日志的分析包括应用系统的安全性分析和行为分析. 在安全分析方面, 由于被攻击的应用日志记录中会包含了两类请求: 正常访问请求和恶意攻击请求, 本文主要通过匹配记录中的恶意访问信息的特征来判断应用系统是否被攻击. 在安全检测方法上采用基于特征方式的检查方法, 该方法的实现主要借助于预先设计攻击特征库和基于 RDD 算子设计的函数模块. 其中攻击特征库是依据各类攻击特征设计正则表达式, 从而匹配出可能存在的攻击类型[14]. 基于 RDD 算子设计的函数模块主要在集群环境下通过 Spark 并发处理机制来提高日志安全分析检索速率.

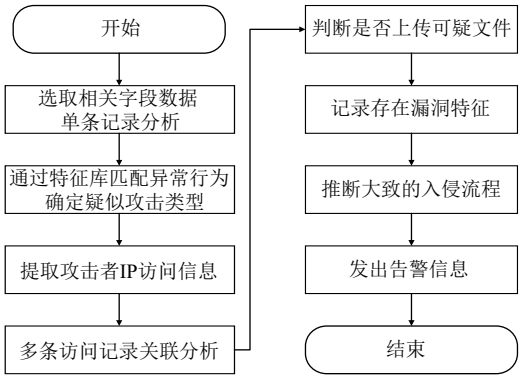

图 5 日志安全分析流程

基于 RDD 算子的安全分析算法主要步骤如算法 2.

算法 2. 应用日志安全检查算法

1) 利用 map() 算子提取相关字段进行单条数据分析.
2) 调用攻击特征库, 通过正则表达式完成攻击行为匹配, 并确定疑似攻击类型.
3) 利用 sortByKey() 算子重现攻击者访问行为轨迹.
4) 利用 union() 算子进行多条记录关联分析.
5) 提取 post 字段, 利用 filter() 算子判断可疑文件.
6) 记录漏洞特征, 推断大致入侵流程并发出告警信息.

在油田应用行为分析方面, 主要是在基于 HBase 的键值存储模型上运行各类分析算法. 由于油田在规划内部网络时会预留一部分 IP 地址作为应用服务地址, 因此可以根据 IP+端口 (appID) 来作为应用的唯一标识, 根据分析指标需求设计 HBase 表存储结构包括: 一个唯一标识的行健 (RowKey) 和两个信息列簇. 其中行健由: 应用 ID、访问时间、用户 IP 三者的组合来标识; 两个列簇分别为用于描述用户设备信息和请求访问结果信息, 其中每个列簇又包括多个列. 应用行为分析存储结构详细描述如表 1.

表 1 HBase 键值表结构

参数名称	参数值
行健 (RowKey)	appID_date_userIP
列簇 1(用户信息)	userIP device os browser
列簇 2(访问信息)	time method url query traffic taken-time status

在应用日志的行为分析算法方面, 主要基于 Spark 计算框架中的 Spark SQL 模块设计完成, Spark SQL 向用户提供了在大数据集上的类 SQL 查询功能, 同时还支持将原有持久化储存数据迁移到 Spark 环境下进行分析[15]. Spark SQL 的分析的核心模块是 DataFrame. DataFrame 是一个以命名列方式组织的分布式数据集. 它类似于关系型数据库中的一张表. DataFrame 可以由结构化数据、现存在的 RDD 或者从外部的关系数据库导入并转换而来[16]. 其中 DataFrame 包括: 用于描述列字段的集合 Schema 和行数据集 DataSet<Row>, 其中列描述信息用于方便下一步运行 Spark SQL 时查询列的标识, 行信息主要由分析数据信息组成.

根据油田管理评估要求需要统计的应用行为指标包括: 应用每小时的访问量、应用运行安全状况、各模块的使用量、应用模块异常信息、使用次数用户排名等 27 个行为指标. 由于 HBase 根据 rowkey 来检索数据并且支持以字符串匹配方式的扫描方法. 因此将时间和 IP 作为查询条件, 可以在各类应用间进行用户访问行为的关联分析, 进而描绘出用户每天在各类应用的停留时间和访问轨迹并推断出用户访问喜好.

本文在实现应用行为分析算法时, 将这些应用统计指标封装在一个算法内, 因此执行一次算法就可统计出所有应用指标. 在 Spark 执行行为分析时需要确定数据源和具体的分析算法: 其中算法选取由调度引擎来完成并提交给 Spark 集群来; 数据源来自于上一阶段的数据预处理算法处理后储存在 HBase 中的结构化数据, 需要调度引擎将要分析数据的起始行键提交给 Spark 集群. Spark 集群根据 HBase 起始行键拉取数据并执行指定算法程序, 完成处理后返回处理结果. 由于每个分析算法需要完成多个分析指标的统计, 因此需要根据分析指标制作多个 DataFrame 数据集. 因为在数据量过大时, 制作 DataFrame 数据集非常耗时. 因此制作数据集时要尽可能满足多个查询需求, 以减少重复制作数据集的处理时耗. 算法流程如下图 6 所示, 其中每一个分支流程对应一个分析指标.

每个行为指标具体的分析流程如下: 首先选取相应的字段并对字段数据进行格式转换、数据规约, 该过程主要借助于 Spark 提供的算子函数完成; 然后将 RDD 数据集转换成 DataSet<Row>数据集并加入列描述信息; 将数据集注册为临时表并运行查询语句; 最后格式化储存结果. 行为指标分析流程如图 7 所示.

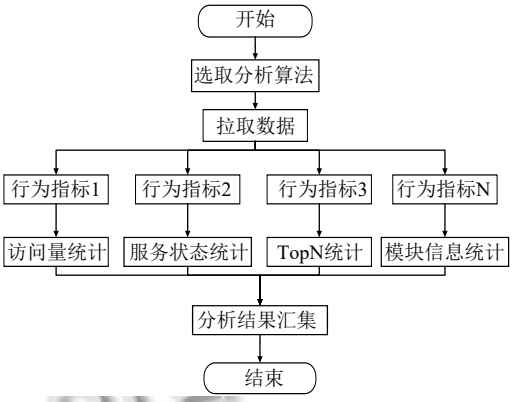

图 6 分析算法流程图

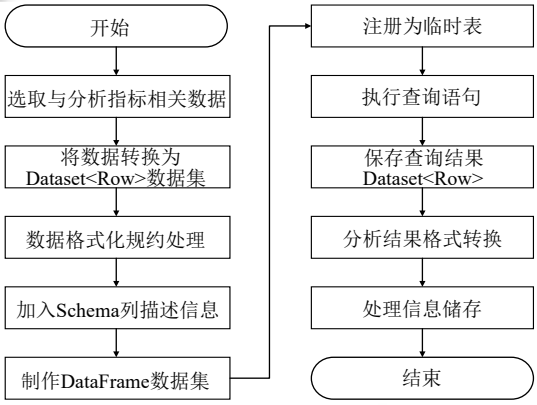

图 7 行为指标分析流程图

4 系统开发环境及实验分析

4.1 系统平台部署

系统平台的主要由三个部分组成: 数据收集层、数据分析层、Web 业务层. 依据实际生产场景, 系统开发环境部署规划如下, 数据收集层由 1 台日志储存服务器组成, 用于部署 Flume 日志收集框架. 数据分析平台是由 1 台主机点和 3 台计算节点组成计算集群, 各节点分别搭建 Hadoop 服务集群、Spark 服务集群、HBase 储存集群, 并在主节点搭建调度引擎程序. 业务层由一台 Web 服务器组成, 用于部署业务管理平台和业务数据库. 系统具体部署规划如图 8 所示.

4.2 实验结果分析

实验分析聚焦在数据分析层上, 主要统计各类算法的分析耗时, 本文的实验环境是由 4 台节点组成的集群环境, 日志文件储存在 HDFS 上, 基于 Spark 框架设计分析算法完成数据的分析, 基于 HBase 储存分析

数据, 并将 Spark 任务直接提交到 Yarn 上, 由 Yarn 完成资源分配和 Spark 任务调度. 其中各节点的环境信息和部署组件信息如表 2 所示.

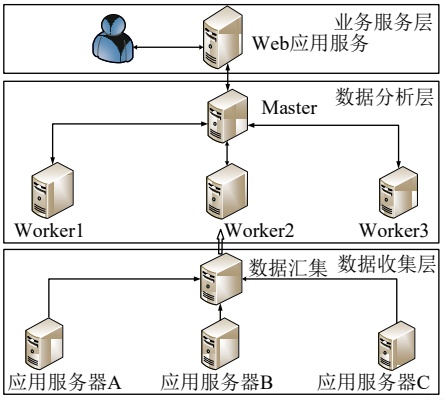

图 8 系统部署图

表 2 Spark 集群运行环境

主机名	操作系统	处理器	内存	硬盘	Hadoop	HBase	Spark
Master	CentOS7	4 核	8 GB	1 TB	2.7.3	1.2.0	2.1.0
Work1	CentOS7	4 核	8 GB	1 TB	2.7.3	1.2.0	2.1.0
Work2	CentOS7	4 核	8 GB	1 TB	2.7.3	1.2.0	2.1.0
Work3	CentOS7	4 核	8G	1T	2.7.3	1.2.0	2.1.0

实验分别在单节点环境和四节点组成的集群环境下测试了 2 个典型算法的耗时, 测试的算法为: 日志文本数据的预处理算法和应用行为指标分析算法 A(该算法主要用于统计 IIS 类型应用日志的行为指标, 包括统计每小时 IP 量、总 UV 量、每小时 PV、总 PV 量、各模块的访问量、TOPN 用户等 27 个行为指标). 日志预处理算法选取了某油田企业内部具有代表性的应用日志数据, 日志数据格式为 IIS W3C 格式. 选取并整理的单个日志数据大小依次为 106 MB、511 MB、1.1 GB、5.1 GB、9.8 GB、20 GB. 实验对比结果如图 9 所示.

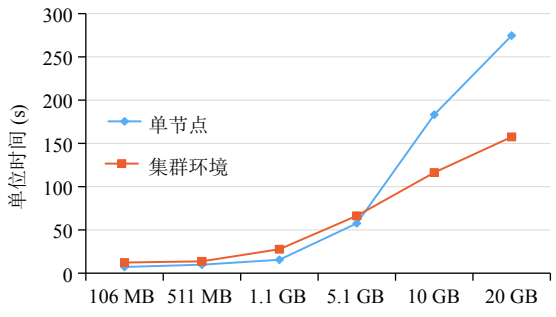

图 9 预处理算法时长对比图

由实验结果可以看出, 当数据量较小时, 单节点的处理时长较短; 当数据容量大于 5 GB 时, 集群环境下的处理时长远小于单节点的处理时长.

算法 A 的实验数据为储存在 HBase 中的结构化数据, 分别选取的数据集分别为: 956 887 条、1975 511 条、5911 511 条、29 479 329 条、63 906 591 条数据. 这里统计数据算法 A 的耗时为从数据加载到内存到预处理数据分析完成的时间 (不包括将数据写回数据库中的时间), 结果如表 3. 该算法的时间消耗主要在于: 制作 DataFrame 数据集的耗时和运行查询 SQL 的耗时, 算法 A 完成 27 个指标的统计, 需要制作 9 个 DataFrame 数据集, 运行了 35 次 SQL 查询.

表 3 算法 A 处理时长对比

数据量 (条)	单节点处理时长 (s)	集群处理时长 (s)
956 887	5.8	9.6
1975 511	13.3	15.8
5911 511	52.2	46.6
29 479 329	135.7	73.6
63 906 591	337.3	107.6

从实验结果可以看出, 当数据集增长到一定程度, 采用集群环境的处理耗时远低于单机处理耗时.

从两个分析算法的耗时统计可以得出: 当数据量大小在单节点处理能力范围内, 单节点处理时长要小于集群环境下处理时长; 若数据量过大, 采用集群环境的处理耗时要小. 这是由于集群环境下涉及到数据的分片, 任务间的通信, 代码序列化分发, 如果数据储存不在本地, 还会涉及到数据的移动问题, 此外处理时长还受主机磁盘 IO 传输速率、网络带宽的传输速率的影响, 这些多方面的因素都会影响处理时长. 因此集群环境在处理大批量数据时才会发挥优势.

5 结论

面对油田应用部署分散、种类繁多、数量庞大的复杂场景. 本文借助于各类主流的大数据处理框架实现对海量数据收集和储存; 在数据处理分析方面, 本文基于 Spark 计算框架设计了应用日志行为分析系统, 并设计了应用的安全状况分析和行为指标分析的算法; 此外为了方便运维人员使用该系统, 又基于 Web 设计了可视化的管理平台实现了各类框架的集成与管理. 该系统解决了油田进行海量应用数据分析的滞后性难题; 为油田迅速评估各类应用系统的运行状况和安全

状况提供了决策依据; 并为油田快捷高效的管理各类业务系统带来了一系列巨大优势.

参考文献

1 Wang GY, Butt AR, Pandey P, *et al*. Using realistic simulation for performance analysis of MapReduce setups. Proceedings of the 1st ACM Workshop on Large-Scale System and Application Performance. New York, NY, USA. 2009. 19–26. [doi: 10.1145/1552272.1552278]

2 Yang HC, Dasdan A, Hsiao RL, *et al*. Map-Reduce-merge: Simplified relational data processing on large clusters. Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data. New York, NY, USA. 2007. 1029–1040. [doi: 10.1145/1247480.1247602]

3 Olston C, Reed B, Srivastava U, *et al*. Pig Latin: A not-so-foreign language for data processing. Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data. New York, NY, USA. 2008. 1099–1110. [doi: 10.1145/1376616.1376726]

4 Thusoo A, Sarma JS, Jain N, *et al*. Hive–a petabyte scale data warehouse using Hadoop. Proceedings of 2010 IEEE 26th International Conference on Data Engineering. Long Beach, CA, USA. 2010. 996–1005. [doi: 10.1109/ICDE.2010.5447738]

5 Thusoo A, Sarma JS, Jain N, *et al*. Hive: A warehousing solution over a Map-Reduce framework. Proceedings of the VLDB Endowment, 2009, 2(2): 1626–1629. [doi: 10.14778/1687553]

6 Koliopoulos AK, Yiapanis P, Tekiner F, *et al*. A parallel distributed WEKA framework for big data mining using Spark. Proceedings of 2015 IEEE International Congress on Big Data. New York, NY, USA. 2015. 9–16. [doi: 10.1109/BigDataCongress.2015.12]

7 Apache Spark is a fast and general-purpose cluster computing system. http://spark.apache.org/docs/latest/. [2017-12-30]

8 Quartz is a richly featured, open source job scheduling library. http://www.quartz-scheduler.org/documentation/. [2017-12-16].

9 Apache Felix is a OSGi framework and service platform. http://felix.apache.org/documentation.html. [2017-12-30].

10 顾兆军, 李晓红, 王伟, 等. Web 日志挖掘中的会话识别方法研究. 计算机技术与发展, 2012, 22(4): 45–49.

11 Facca FM, Lanzi PL. Mining interesting knowledge from weblogs: A survey. Data & Knowledge Engineering, 2005, 53(3): 225–241.

12 高彦杰, 倪亚宇. Spark 大数据分析实战. 北京: 机械工业出版社, 2016: 6–7.

13 Zhang F, Liu M, Gui F, *et al*. A distributed frequent itemset mining algorithm using Spark for big data analytics. Cluster Computing, 2015, 18(4): 1493–1501. [doi: 10.1007/s10586-015-0477-1]

14 张树壮, 罗浩, 方滨兴, 等. 一种面向网络安全检测的高性能正则表达式匹配算法. 计算机学报, 2010, 33(10): 1976–1986.

15 Armbrust M, Xin RS, Lian C, *et al*. Spark SQL: Relational data processing in spark. Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data. New York, NY, USA. 2015. 1383–1394.

16 Peng P, Zou L, Özsu MT, *et al*. Processing SPARQL queries over distributed RDF graphs. The VLDB Journal, 2016, 25(2): 243–268. [doi: 10.1007/s00778-015-0415-0]