基于 Hadoop 的云计算平台研究与实现

范素娟[1],田军锋[2]
(1. 河南大学 信息化管理办公室,河南 开封 475004;
2. 河南大学 计算机与信息工程学院,河南 开封 475004)

摘 要:随着网络技术的发展,网络数据量正以指数级增长且规模日渐庞大。面对正在增长的海量数据,传统的数据处理方法存在效率低下等诸多缺点。人们需要一种新的技术思想来解决这些问题。因此,云计算的思想被提出。云计算是一种新兴的计算模型,是分布式计算技术的一种。而 Hadoop 作为一个开源的分布式平台是当前最为流行的云计算平台实现之一,被用于高效地处理海量数据。为了提高对海量数据处理的效率,文中首先简要分析了云计算的概念和 Hadoop 主要组件的工作流程,然后详细介绍了基于 Hadoop 的云计算平台配置方法和实现过程,并对云平台的搭建过程中遇到的典型问题进行了总结阐述。最后通过实验证明,该平台可以有效地完成分布式数据处理任务。

关键词:Hadoop;HDFS;MapReduce;云计算
中图分类号:TP39 **文献标识码**:A **文章编号**:1673-629X(2016)07-0127-04
doi:10.3969/j.issn.1673-629X.2016.07.027

Research and Implementation of Cloud Computing Platform Based on Hadoop

FAN Su-juan[1], TIAN Jun-feng[2]
(1. Information Management Office, Henan University, Kaifeng 475004, China;
2. School of Computer and Information Engineering, Henan University, Kaifeng 475004, China)

Abstract: With the development of network technology, the number of online information is increasing in exponential and becoming larger and larger. With the growing amount of data, the traditional methods for processing massive data have many shortcomings like low efficiency. A novel technology is needed to solve these problems, so the cloud computing has been brought. It is an emerging computational model, as a kind of distributed computing technology. Hadoop is one of the most popular cloud computing platforms as a kind of open sources distributed platform, which is always applied on the area that needs to handle massive data efficiently. In order to improve the efficiency of processing massive data, it briefly analyzes the concept of cloud computing and the work flow of the main components of Hadoop in this paper, then introduction of the implementation method of the cloud computing platform based on Hadoop in detail, discussion of the typical problems encountered in the process of building cloud computing platform. Finally, the experiments show that the platform can effectively complete the processing tasks of distributed data.

Key words: Hadoop; HDFS; MapReduce; cloud computing

0 引 言

云计算[1]是一种新兴的计算模型,它是分布式计算、并行计算、虚拟化等传统计算机和网络技术发展融合的产物。其最基本的概念,是通过网络将庞大的计算处理程序自动分拆成无数个较小的子程序,再交由多部服务器所组成的庞大系统,经搜寻、计算分析之后将处理结果回传给用户。通过这项技术,网络服务提供者可以在数秒之内处理数以千万计甚至亿计的信息,达到和“超级计算机”同样强大效能的网络服务。目前有众多的云计算模型[2-3],但大部分属于商业模型,而 Hadoop 作为 Apache 基金会下的开源云计算模型,实现了包括分布式文件系统(HDFS)和 MapReduce 框架在内的云计算软件平台的基础架构,并且使用 Java 语言编写,可移植性强,在很多大型网站上都

收稿日期:2015-10-20 **修回日期**:2016-01-27 **网络出版时间**:2016-06-22
基金项目:国家自然科学基金资助项目(61202098)
作者简介:范素娟(1983-),女,助理工程师,硕士,研究方向为云计算技术。
网络出版地址:http://www.cnki.net/kcms/detail/61.1450.TP.20160622.0842.014.html

得到了应用。其已成为企业和个人进行云计算应用和研究[4]的标准平台。

1 Hadoop 简介

Hadoop[5]是 Apache 开源组织的一个分布式计算开源框架,其最核心的设计是 MapReduce 和 HDFS(Hadoop Distributed File System)。简言之 MapReduce 就是任务的分解与结果的汇总,HDFS 为分布式计算存储提供了底层支持。

Hadoop 的优势不但在于它的开源,而且还具有如下一些特点:

(1)高可靠性:HDFS 的备份恢复机制及 MapReduce 的任务监控保证了分布式处理的可靠性。

(2)高扩展性:Hadoop 是在可用的计算机集簇间分配数据并完成计算任务的,这些集簇可以方便地扩展到数以千计的节点中。

(3)高效性:通过分发数据,Hadoop 可以在数据所在的节点上并行处理,因此处理速度非常快。

(4)高容错性:Hadoop 能够自动维护数据的多个副本,并且可自动将失败的任务重新分配。

(5)低成本:Hadoop 依赖于社区服务器,在任何普通的 PC 上安装配置 Hadoop 集群,都可以对海量的数据进行高效处理[6]。

1.1 分布式文件系统

HDFS[7]是 Hadoop 中数据存储管理的基础。它具有高吞吐率、高容错性、高扩展性和高可靠性等特点,为海量数据存储提供了良好的保障和便利,非常适合大规模数据集上的应用[8]。其体系结构如图 1 所示。

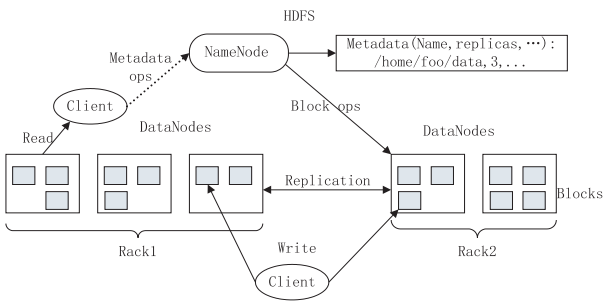

图 1 HDFS 体系结构

HDFS 采用 master/slave 架构。一个 HDFS 集群由一个 NameNode 和多个 DataNode 组成。其中 NameNode 作为主服务器,负责管理文件系统的命名空间和客户端对文件的访问。集群中的 DataNode 管理它所在节点上的数据存储。HDFS 对外公开文件系统的命名空间,用户能够以文件的形式在上面存储数据。从内部看,一个文件其实被分成若干个数据块(Block),这些 Block 存储在一组 DataNode 上。NameNode 执行文件系统的命名空间操作并负责确定 Block 到具体 DataNode 节点的映射。DataNode 负责处理文件系统的读写请求,在 NameNode 的统一调度下进行 Block 的创建、复制和删除。

1.2 MapReduce 编程模型

MapReduce[9]是一种用于处理大规模数据集并行运算的编程模型[10-11]。其处理数据的过程主要分成两个阶段:Map 阶段和 Reduce 阶段。先执行 Map 阶段,再执行 Reduce 阶段。用户在使用该模型时,可根据一定的编程规则来分别实现 Map 和 Reduce 函数,之后 MapReduce 会自动地对任务进行分割从而实现并行执行。MapReduce 模型如图 2 所示。

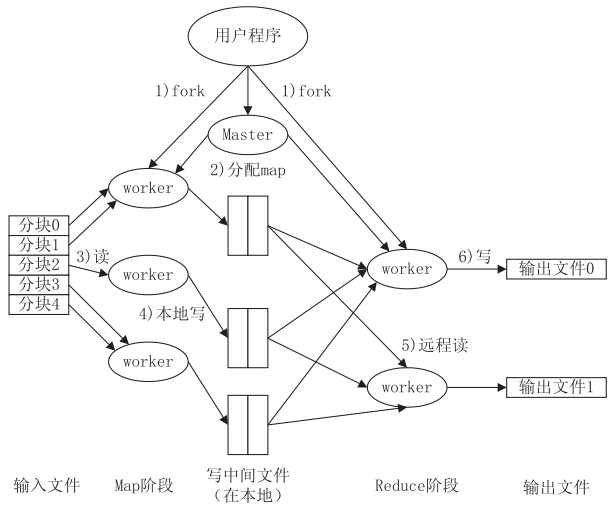

图 2 MapReduce 模型

MapReduce 过程在正式执行 Map 函数前,首先把输入文件数据分成 M 份数据块文件;然后 Master 服务器调度不同的 Worker 服务器进行 Map 过程,不同的 Map 之间是互相独立且高度并行的,它以<Key,Value>的形式读入数据,把处理后的中间结果也以<Key,Value>的形式保存在本地存储;最后 Master 服务器调度不同的 Worker 服务器执行 Reduce 过程,把<Key,Value>形式的中间结果合并输出保存到 HDFS。

2 Hadoop 云平台实现方法

文中云平台搭建所用的服务器均为生产环境里的虚拟机,基本配置:CPU(Intel(R) Xeon(R) CPU E5-2660 v2 @ 2.20 GHz 2.20 GHz)、内存(4 GB)、硬盘(30 GB,容量可根据需要增加)。这里主要介绍在 Linux 操作系统下安装配置 Hadoop 的方法,所用操作系统版本为 Red Hat Enterprise Linux Server release 6.1(32 位),Java 版本为 1.7.0_79,Hadoop[12]版本为 2.4.0。安装的主要步骤如下所述。

2.1 网络配置

集群中包括 4 个节点:1 个 master 和 3 个 slave。所有节点处于同一 VLAN 内,可以相互 ping 通,并且

四个节点有一个相同的用户 Hadoop。master 节点主要配置 NameNode 和 JobTracker 的角色,负责总管分布式数据和分解任务的执行;3 个 salve 节点配置 DataNode 和 TaskTracker 的角色,负责分布式数据存储以及任务的执行。

在 Linux 系统的安装过程中配置好各节点的主机名、IP 及 Hadoop 用户的创建。然后是配置各节点的 hosts 文件。"/etc/hosts"文件是用来配置主机将用的 DNS 服务器信息,为了方便节点间通过节点名称来访问,分别在每个节点的"/etc/hosts"文件中添加各节点的 IP 地址和主机名的对应信息,格式如下所示:

172.31.0.250 master

172.31.0.251 slave1

172.31.0.252 slave2

172.31.0.253 slave3

2.2　SSH 无密码验证配置

Hadoop 集群的各个节点之间需要进行数据的访问,如果 Hadoop 对每个节点的访问均需要进行验证,其效率将会大大降低,所以需要配置 SSH 免密码的方法直接远程连入被访问节点,这样将大大提高访问效率。所有节点都用 Hadoop 用户登陆进行配置。

(1)公私密钥的生成。

每个节点分别产生公私密钥[13],并将公钥文件复制成 authorized_keys 文件:

[hadoop@ master ~] $ ssh-keygen -t dsa -P '' -f ~/.ssh/id_dsa

[hadoop@ master .ssh] $ cat id_dsa.pub >> authorized_keys

(2)单机回环 SSH 免密码登录测试。

首先将"~/.ssh"目录下的所有文件权限设置为 600,然后在单机节点上用 ssh 进行登录。如当前目录变为 ~,则表示操作成功。输入命令 exit 注销退出,当前目录重新回到"~/.ssh"。单机回环 SSH 登录及注销成功,将为后续跨子节点 SSH 远程免密码登录作好准备。

[hadoop@ master .ssh] $ ssh localhost

(3)让主节点能通过 SSH 免密码登录三个子节点。

为实现这个功能,三个 slave 节点的公钥文件中必须要包含主节点的公钥信息:

[hadoop@ slave1 .ssh] $ scp hadoop@ master:~/.ssh/id_dsa.pub ./master_dsa.pub

[hadoop@ slave1 .ssh] $ cat master_dsa.pub >> authorized_keys

首先 slave1 节点通过 scp 命令远程登录 master 节点,并复制 master 的公钥文件到当前的目录下,这一过程需要密码验证。接着,将 master 节点的公钥文件追加至 authorized_keys 文件中,通过这步操作,master 节点就可以通过 ssh 远程免密码连接 slave1 节点了。在 master 节点中操作如下:

[hadoop@ master ~] $ ssh slave1

子节点首次连接时需要输入 yes 确认连接,连接后注销退出至 master 节点。

下次再执行连接命令时,master 就可以免密码直接登录至子节点了。

(4)让子节点能通过 SSH 免密码登录主节点。

[hadoop@ master .ssh] $ scp hadoop@ slave1:~/.ssh/id_dsa.pub ./slave1_dsa.pub

[hadoop@ master .ssh] $ cat slave1_dsa.pub >> authorized_keys

[hadoop@ master .ssh] $ rm ./slave1_dsa.pub

用同样的方法配置另外两个子节点。至此,master 和每个 slave 能够相互无密码验证登录。

2.3　Java 环境配置

所有节点上都要安装 JDK,现在就先在 master 节点安装,然后其他节点按照步骤重复进行即可。安装 JDK 以及配置环境变量,需要以 root 的身份进行。

(1)先卸载服务器自带的 JDK 软件包。

[root@ master ~] # java -version

[root@ master ~] # rpm -qa | grep jdk

[root@ master ~] # rpm -e --nodeps java-1.6.0-openjdk-1.6.0.0-1.39.1.9.7.e16.i686

首先查看服务器当前的 JDK 软件版本,然后执行卸载命令卸载服务器自带的版本。

(2)上传 JDK 安装文件并执行安装。

创建 JDK 安装目录"/usr/java",将 JDK 安装文件上传到该目录并安装。如果"/usr/java"下面出现一个名为"jdk1.7.0_79"的文件夹,说明 JDK 安装结束,删除 JDK 安装文件。

[root@ master ~] # mkdir /usr/java

[root@ master ~] # rpm -ivh /usr/java/jdk-7u79-linux-i586.rpm

[root@ master ~] # rm -f /usr/java/jdk-7u79-linux-i586.rpm

(3)配置环境变量。

编辑"/etc/profile"文件,在末尾添加如下内容:

export JAVA_HOME=/usr/java/jdk1.7.0_79

export CLASSPATH=.:$CLASSPATH:$JAVA_HOME/lib:$JAVA_HOME/jre/lib

export PATH=$PATH:$JAVA_HOME/bin:$JAVA_HOME/jre/bin

然后执行 source 命令使配置生效并查看当前的

Java 版本,确认是所安装的版本。

[root@ master ~] # source /etc/profile

[root@ master ~] # java -version

2.4 Hadoop 集群配置

所有节点都要安装 Hadoop,先在 master 节点上安装,然后其他节点按照相同步骤进行安装配置。下面以 root 的身份进行安装和配置 Hadoop。

(1)上传并解压 Hadoop 安装包。

上传并解压 Hadoop 安装包到"/usr"目录下,安装完成后删除,然后把"/usr/hadoop-2.4.0"的读权限分配给 Hadoop 用户:

[root@ master usr] # tar - zxvf hadoop-2.4.0. tar. gz

[root@ master usr] # rm - rf hadoop-2.4.0. tar. gz

[root@ master usr] # chown - R hadoop:hadoop hadoop-2.4.0/

在"/usr/hadoop-2.4.0"目录下面分别创建 tmp,dfs/name,dfs/data 文件夹,用于存放命名空间以及数据信息,并将所建文件夹权限分配给 Hadoop 用户。

[root@ master hadoop-2.4.0] # mkdir /usr/hadoop-2.4.0/tmp

[root@ master ~] # mkdir - p /usr/hadoop-2.4.0/dfs/name

[root@ master ~] # mkdir - p /usr/hadoop-2.4.0/dfs/data

[root@ master ~] # chown - R hadoop:hadoop /usr/hadoop-2.4.0/tmp

[root@ master ~] # chown - R hadoop:hadoop /usr/hadoop-2.4.0/dfs

(2)配置文件"/etc/profile"。

在文件末尾添加如下信息,并执行 source 命令使配置生效。

export HADOOP_HOME=/usr/hadoop-2.4.0

export PATH= $ PATH: $ HADOOP_HOME/bin

(3)配置文件 hadoop-env. sh 和 yarn-env. sh。

分别添加 Java 路径信息。注意:步骤(3) ~ (8)所配置文件均在"/usr/hadoop-2.4.0/etc/hadoop/"目录下。

export JAVA_HOME=/usr/java/jdk1.7.0_79

(4)配置文件 slaves,添加三个子节点的主机名:slave1、slave2、slave3。

(5)配置文件 core-site. xml。

在文件末尾的 configuration 节点添加如下属性:

```
<property>
<name>hadoop. tmp. dir</name>
<value>file:/usr/hadoop-2.4.0/tmp</value>
<description>A base for other temporary directories.</description>
</property>
<property>
<name>io. file. buffer. size</name>
<value>131072</value>
</property>
<property>
<name>fs. default. name</name>
<value>hdfs://master:9000</value>
</property>
<property>
<name>hadoop. proxyuser. root. hosts</name>
<value> * </value>
</property>
<property>
<name>hadoop. proxyuser. root. groups</name>
<value> * </value>
</property>
```

(6)配置文件 hdfs-site. xml,格式同步骤(5)。注意:步骤(6) ~ (8)省略了<property></property>这一组标签的书写,实际配置时需要加上。

```
<name>dfs. replication</name><value>3</value>
<name>dfs. namenode. name. dir</name>
<value>file:/usr/hadoop-2.4.0/dfs/name</value>
<final>true</final>
<name>dfs. dataname. data. dir</name>
<value>file:/usr/hadoop-2.4.0/dfs/data</value>
<final>true</final>
<name>dfs. namenode. secondary. http - address</name>
<value>master:9001</value>
<name>dfs. webhdfs. enabled</name><value>true</value>
```

(7)配置文件 yarn-site. xml,格式同步骤(5)。

```
<name>yarn. resourcemanager. address</name>
<value>master:8032</value>
<name>yarn. resourcemanager. scheduler. address</name>
<value>master:8030</value>
<name>yarn. resourcemanager. webapp. address</name>
<value>master:8088</value>
<name>yarn. resourcemanager. resource-tracker. address</name>
<value>master:8031</value>
```

< name > yarn. resourcemanager. admin. address </ name>

<value>master:8033</value>

<name>yarn. nodemanager. aux-services</name>

<value>mapreduce_shuffle</value>

< name > yarn. nodemanager. aux - services. mapreduce. shuffle. class</name>

<value>org. apache. hadoop. mapred. ShuffleHandler </value>

(8)配置文件 mapred-site. xml。

文件 mapred-site. xml 从 mapred-site. xml. template 复制而来,格式同步骤(5)。

<name>mapreduce. framework. name</name>

<value>yarn</value>

<name>mapreduce. jobhistory. address</name>

<value>master:10020</value>

< name > mapreduce. jobhistory. webapp. address </ name>

<value>master:19888</value>

(9)在剩余节点上安装配置 Hadoop。

拷贝 Hadoop 安装目录到其他节点,并分别配置各节点的"/etc/profile"文件。注意:"/usr/hadoop-2. 4. 0/"目录下所有文件权限均为 hadoop:hadoop。

[root@ master ~] # scp - r /usr/hadoop-2. 4. 0 slave1 :/usr/

[root@ slave1 ~] $ chown - R hadoop:hadoop /usr/hadoop-2. 4. 0/

2.5　启动及验证

(1)格式化 HDFS 文件系统。

在 master 上使用用户 Hadoop 进行操作。只要出现类似"Storage directory /usr/hadoop-2. 4. 0/dfs/name has been successfully formatted"的信息,就表示格式化成功。注意:只需格式化一次,下次启动时不需要,仅执行启动命令 start-all. sh 即可。

[hadoop@ master ~] $ hdfs namenode - format

(2)启动 Hadoop。

[hadoop@ master hadoop-2. 4. 0] $ ./sbin/start-all. sh

(3)验证 Hadoop。

用 jps 命令查看进程,如 master 上出现 NameNode、SecondaryNameNode、ResourceManager 和 Jps,slave 上出现 DataNode、NodeManager 和 Jps 等进程,则表示集群配置成功。经验证,文中集群配置成功。

3　Hadoop 云平台的测试

HDFS 中的文件与目录用 Linux 命令是无法直接访问,必须使用 Hadoop 自身提供的命令且所有操作都需在 NameNode 上进行。下面通过运行 WordCount 程序来对所建的云计算平台进行测试。WordCount 是最简单也是最能体现 MapReduce 思想程序之一,其主要功能是统计一系列文本文件中每个单词出现的次数。

3.1　在本地创建示例文件

首先在"/home/Hadoop"目录下创建文件夹 file。接着在 file 文件夹下创建两个文本文件 file1. txt 和 file2. txt,并输入相应内容。

[hadoop@ master ~] $ mkdir ./file

[hadoop@ master file] $ echo "Hello World Hello everyone" > file1. txt

[hadoop@ master file] $ echo "Hello Hadoop and everyone" > file2. txt

3.2　将本地 file 中的文件上传到 Hadoop 文件系统

[hadoop@ master ~] $ hadoop fs - mkdir /input

[hadoop@ master ~] $ hadoop fs - put ./file/file * . txt /input

3.3　在集群上运行 WordCount 程序

图 3 是 Hadoop Job 运行记录的部分截图。

```
[hadoop@master ~]$ hadoop jar /usr/hadoop-2.4.0/share/hadoop/mapreduce/sources/h
adoop-mapreduce-examples-2.4.0-sources.jar org.apache.hadoop.examples.WordCount
/input /output
15/09/22 16:53:57 INFO client.RMProxy: Connecting to ResourceManager at master/1
72.31.0.250:8032
15/09/22 16:53:58 INFO input.FileInputFormat: Total input paths to process : 2
15/09/22 16:53:58 INFO mapreduce.JobSubmitter: number of splits:2
15/09/22 16:53:58 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_14
41765482318_0004
15/09/22 16:53:58 INFO impl.YarnClientImpl: Submitted application application_14
41765482318_0004
15/09/22 16:53:58 INFO mapreduce.Job: The url to track the job: http://master:80
88/proxy/application_1441765482318_0004/
15/09/22 16:53:58 INFO mapreduce.Job: Running job: job_1441765482318_0004
15/09/22 16:54:05 INFO mapreduce.Job: Job job_1441765482318_0004 running in uber
 mode : false
15/09/22 16:54:05 INFO mapreduce.Job:  map 0% reduce 0%
15/09/22 16:54:16 INFO mapreduce.Job:  map 100% reduce 0%
15/09/22 16:54:23 INFO mapreduce.Job:  map 100% reduce 100%
15/09/22 16:54:24 INFO mapreduce.Job: Job job_1441765482318_0004 completed succe
ssfully
```

图 3　Hadoop 运行任务过程

从图中可以看到,这个 Job 的 ID 号为 job_1441765482318_0004,输入文件有两个,另外从 Job 运行记录中还可以了解到 map 和 reduce 的输入输出记录等信息。在本例中,map 的 task 数量是 2 个,reduce 的 task 数量是 1 个。map 的输入 record 数是 2 个,输出 record 数是 8 个。

3.4　查看结果输出文件内容

实验模拟结果如图 4 所示。

```
[hadoop@master ~]$ hadoop fs -cat /output/part-r-00000
Hadoop  1
Hello   3
World   1
and     1
everyone        2
[hadoop@master ~]$
```

图 4　实验模拟结果

实验结果证明,所建云平台可以有效地完成分布式数据处理任务。

4 问题总结

在云平台的搭建过程中也遇到了一些问题,下面列出比较典型的几个,以便在以后的平台搭建过程中引起注意和提供参考。

(1)权限问题。

ssh 免密码配置后各节点间数据访问还是需要输入密码,将防火墙关闭也不行。最后将. ssh 目录下的所有文件权限设置为 600(chmod 600 ~/. ssh/ *),问题就解决了。

(2)防火墙的问题。

Hadoop 配置完成后,用 jps 命令查看发现 slave 节点中没有 nodemanager 进程,禁用防火墙后 slave 节点中出现 nodemanager 进程。

(3)格式化 HDFS 的问题。

重新格式化 HDFS 文件系统之后,发现 DataNode 无法启动。原因是每次格式化 HDFS 文件系统会重新创建一个 namenodeId,而目录"tmp/dfs/data"下包含了上次格式化后的 id,格式化 HDFS 文件系统清空了 NameNode 下的数据,但没有清空 DataNode 下的数据,所以导致启动时失败。所要做的就是每次格式化 HDFS 文件系统前,先清空 tmp 文件夹下的所有目录。

值得注意的是,在处理问题的过程中要善于通过查看运行日志找到问题的产生原因,并通过借助互联网寻求问题的处理办法来解决问题。

5 结束语

综合采用并行计算、分布式计算和虚拟化等技术的云计算将海量数据处理推进到一个新时代。而 Hadoop 的开源、跨平台、高容错等特点使其成为构建云计算平台的首选技术。文中详细介绍了 Hadoop 集群的搭建方法,成功地搭建了 Hadoop 云计算平台并进行了测试,有助于以后对大数据[14]的研究。最后将云平台搭建过程中遇到的问题进行了总结,以便引起注意并提供参考。下一步的工作主要是在 Hadoop 云计算平台下进行相关算法的研究和应用。

参考文献:

[1] 柯栋梁,郑 啸,李 乔. 云计算:实例研究与关键技术[J]. 小型微型计算机系统,2012,33(11):2321-2329.

[2] 林 利,石文昌. 构建云计算平台的开源软件综述[J]. 计算机科学,2012,39(11):1-7.

[3] Armbrust M,Fox A,Griffith R,et al. A view of cloud computing[J]. Communication of the ACM,2010,53(4):50-58.

[4] 张良将. 基于 Hadoop 云平台的海量数字图像数据挖掘的研究[D]. 上海:上海交通大学,2013.

[5] 王彦明,奉国和,薛 云. 近年来 Hadoop 国外研究综述[J]. 计算机系统应用,2013,22(6):1-5.

[6] Chaudhary A,Singh P. Big data - importance of Hadoop distributed filesystem[J]. International Journal of Scientific & Engineering Research,2013,4(11):234-237.

[7] 童 明. 基于 HDFS 的分布式存储研究与应用[D]. 武汉:华中科技大学,2012.

[8] 郝增勇. 基于 Hadoop 用户行为分析系统设计与实现[D]. 北京:北京交通大学,2014.

[9] Dean J,Ghemawat S. MapReduce:simplifier date processing on large clusters[J]. Communications of the ACM,2008,51(1):107-113.

[10] Berlinska J,Drozdowskib M. Scheduling divisible MapReduce computations[J]. Parallel and Distributed Computing,2011,71(3):450-459.

[11] 徐焕良,翟 璐,薛 卫,等. Hadoop 平台中 MapReduce 调度算法研究[J]. 计算机应用与软件,2015,32(5):1-6.

[12] Apache Hadoop[EB/OL]. 2015-08-07. http://hadoop.apache.org/.

[13] 王婷娟,管会生,尹 晖. DSA 与 RSA 相结合的数字签名技术[C]//全国第 19 届(CACIS)学术会议论文集(下册). 出版地不详:出版者不详,2008:1129-1133.

[14] 严霄凤,张德馨. 大数据研究[J]. 计算机技术与发展,2013,23(4):168-172.

(上接第 126 页)

[5] Negnevitsky M. 人工智能:智能系统指南[M]. 北京:机械工业出版社,2012.

[6] 王美方,刘培玉,朱振方. 一种基于 TFIDF 的特征选择方法[J]. 计算机工程与设计,2007,28(23):5795-5796.

[7] 张可佳,李春生,姜海英,等. 时间序列下模式挖掘模型设计[J]. 计算机工程与应用,2015,51(19):146-151.

[8] Yang Y,Pedersen J O. A comparative study on feature selection in text categorization[C]//Proceedings of 14th international conference on machine learning. Nashville,US:[s. n.],2007:412-420.

[9] 吕海燕,车晓伟. 数据仓库中数据粒度的划分[J]. 计算机工程与设计,2009,30(9):2323-2325.

[10] 王晓鹏,武 彤. 生产质量控制数据仓库模型设计与实现[J]. 计算机技术与发展,2015,25(6):181-184.

[11] 王 虎,丁世飞. 序列模式挖掘研究与发展[J]. 计算机科学,2009,36(12):14-17.

[12] 卓书月. 柯西不等式及其变式的应用[J]. 民营科技,2011(9):78-78.

[13] Duda R O,Hart P E,Stock D G. 模式分类[M]. 北京:机械工业出版社,2000:36-39.

[14] 高 建,侯加根,王 军,等. 聚合物驱后砂岩储层岩石物理特征变化机制[J]. 中国石油大学学报:自然科学版,2009,33(3):22-26.

[15] 徐松辽. 影响二类聚驱油层压裂效果的原因分析[J]. 黑龙江科技信息,2012(11):63-63.