北京市科协青年人才成长 / 出版学术专著项目资助

大数据技术
在大规模储能系统中的应用

李相俊　高飞　刘松　惠东 等　编著

中国电力出版社
CHINA ELECTRIC POWER PRESS

北京市科协青年人才成长/出版学术专著项目资助

大数据技术在大规模储能系统中的应用

李相俊　高飞　刘松　惠东　贾学翠　编著

中国电力出版社
CHINA ELECTRIC POWER PRESS

内 容 提 要

能源互联网将承载和推动第四次工业革命，而储能是其中不可分割的重要组成部分。随着分布式或集中式的大容量电化学储能系统建设的不断深入和推进，大规模储能系统（电站）运行和设备监测产生的数据量呈指数级增长，这就需要大数据技术作为存储和快速处理的技术支撑。本书重点阐述了大数据技术在大规模储能系统中的应用。

本书共分 9 章，分别介绍了大数据处理架构 Hadoop；大数据存储、处理与分析技术；数据挖掘基础；储能电池性能分析与评价；储能电池（组）寿命分析与评价；大规模储能系统海量电池数据采集、存储及管理技术；大规模储能系统海量电池数据管理平台；大规模储能系统海量电池数据分析与电池运行状态评价；大规模储能系统海量电池数据云服务平台。

本书以通俗易懂、深入浅出的方式，为从事大规模储能、大数据技术的科研人员提供参考和借鉴，也可供高等院校从事储能和大数据技术研究的师生阅读。

图书在版编目（CIP）数据

大数据技术在大规模储能系统中的应用 / 李相俊等编著. —北京：中国电力出版社，2018.9

ISBN 978-7-5198-1609-4

Ⅰ. ①大… Ⅱ. ①李… Ⅲ. ①数据处理－应用－储能 Ⅳ. ①TK02-39

中国版本图书馆 CIP 数据核字（2018）第 159868 号

出版发行：中国电力出版社
地　　址：北京市东城区北京站西街 19 号（邮政编码 100005）
网　　址：http://www.cepp.sgcc.com.cn
责任编辑：邓　春　刘　薇（010-63412787）
责任校对：黄　蓓　李　楠
装帧设计：张俊霞　左　铭
责任印制：邹树群

印　　刷：三河市百盛印装有限公司
版　　次：2018 年 9 月第一版
印　　次：2018 年 9 月北京第一次印刷
开　　本：787 毫米×1092 毫米　16 开本
印　　张：14.5
字　　数：332 千字
印　　数：0001—1000册
定　　价：68.00 元

前 言

能源互联网将承载和推动第四次工业革命，而储能是其中不可分割的重要组成部分。近年来，世界各地都在积极开展储能技术的研究工作。目前，电化学储能尤其是电池储能相关技术得到了快速发展。随着分布式和集中式电池储能系统建设的不断深入和推进，电池储能系统（电站）运行和设备监测产生的数据量呈指数级增长，这就需要相应的存储和快速处理技术作为支撑。

大数据是随着信息技术尤其是近些年的互联网和物联网技术的发展而产生的一个新的趋向。大数据技术是大型电池储能系统状态分析与诊断的重要手段。事实上，人们已将基于大数据及相关分析处理技术的对科学和工程问题的探究统称为“数据密集型”科学，并将之视为与传统的理论科学、实验科学和计算科学比肩的第四种研究范式。数据量的增大会由量变转化为质变，随之而来的是大数据相关问题的两个方面：工程问题和科学问题。电池储能应用中，上述问题将体现为传统数据库系统及相应数据处理系统向大数据应用系统升级改造的工程技术问题，以及对电池储能系统相关累积数据的深入洞察理解和关联价值解析等的科学问题。

本书作者团队多年来一直从事电池特性分析、电池储能系统规划设计、集成与运行控制以及电力大数据应用等方面的科研和实践工作，熟悉我国电池储能系统集成与应用朝着智能化、信息化发展的技术方向和关键问题，致力于促进电池储能及大数据的应用技术发展。2012 年起，先后承担了国家电网公司“大容量储能电站监控技术研究”、“大规模电池储能电站运行技术研究与应用”等科技项目。2016 年起，还承担了国家重点研发计划专项课题“基于新型锂离子电池的大规模储能系统集成及应用示范”等重大研究课题，深感肩负的知识普及、技术引领、科技成果转化的责任重大。为了帮助从事电池储能技术相关行业的广大读者理解究竟什么是大数据、怎样应用大数据、如何开展电池状态分析与评价、如何构建海量数据管理及云服务平台、当前的主要问题、未来发展趋势等基础问题，作者团队以多年的理论与实践积累为基础，出版了这本《大数据技术在大规模储能系统中的应用》科技著作。期望使读者能从中了解大数据与云技术在电池储能系统中是如何应用的，以及对储能电池（组）性能、寿命、一致性等是如何进行分析与评价的。同时为相关领域的科研人员在以后的深入研究和实践中提供参考和助益，为电池储能系统（电站）的建设、

运行、维护，以及电力大数据的应用贡献绵薄之力。

本书共分 9 章，第 1 章介绍大数据处理架构 Hadoop；第 2 章介绍大数据存储、处理与分析技术；第 3 章介绍数据挖掘基础；第 4 章介绍储能电池性能分析与评价；第 5 章介绍储能电池（组）寿命分析与评价；第 6 章介绍大规模储能系统海量电池数据采集、存储及管理技术；第 7 章介绍大规模储能系统海量电池数据管理平台；第 8 章介绍大规模储能系统海量电池数据分析与电池运行状态评价；第 9 章介绍大规模储能系统海量电池数据云服务平台。

本书在编写过程中，刘松副教授编写了第 1、2、3 章，高飞高级工程师编写了第 4、5 章，李相俊教授级高级工程师编写了第 6、7、8、9 章并统稿，惠东教授级高级工程师、贾学翠工程师参与编写了第 8 章。另外王林助理研究员、刘佳琦助理研究员、袁涛博士、郑岳久博士等为本书的编写提供了许多素材，王向前工程师、郑昊工程师、姚继峰高级工程师等参与了本书部分研究工作，马锐、李跃等硕士研究生参与了本书的辅助工作，在此表示感谢。本书还借鉴了一些已公开发表的研究成果和网络上的资料，均在参考文献中注明，在此对这些成果涉及的作者和刊物一并表示感谢！同时感谢北京市科协青年人才成长/出版学术专著项目对本书出版的资助。

由于作者水平有限，兼时间仓促，书中难免存在疏漏之处，敬请广大读者批评指正。

编者

2017 年 12 月

大数据技术
在大规模储能系统中的应用

目 录

1

大数据处理架构 Hadoop

本章讲解大数据下的 Hadoop 处理架构，对 Hadoop 的核心组件进行讲解并给出具体安装及应用实例。总体来说，Hadoop 是一个开发和运行处理大规模数据的软件平台，是 Apache 的一个用 Java 语言实现的开源软件框架，可在大量计算机组成的集群中对海量数据进行分布式计算。Hadoop 框架中最核心的设计就是：Hadoop 分布式文件系统（Hadoop Distributed File System，HDFS）和分布式计算基本框架 MapReduce。HDFS 提供了海量数据的存储，MapReduce 提供了对数据的计算。其核心组件的种类和功能可是目前主流应用系统，是数据应用中常用文件存储组件的基础；MapReduce 编程困难，效率低；YARN 是一种分布式资源调度，它可以接收分配给每个集群节点的计算任务，相当于大数据操作系统，具有良好的通用性和良好的生态支持；ZooKeeper 可以被理解为一个小的高性能数据库，用以协作发布订阅函数，并在生态系统中提供许多组件，检测节点的失败（如心跳检测）。如确认消息是否准确到达，防止单点失效，处理负载均衡等。后续章节将针对大数据存储、处理与分析等相关技术及系统组件进行进一步讲解。

Hadoop 生态圈内的开源组件和产品有很多，包括本章中提到的 HDFS、YARN、ZooKeeper，还有诸如 HBase、hive、Spark、pig、kafka、flume、phoenix、sqoop 等也是常用的组件，在实际应用中，需针对不同系统的不同业务需求场景，选取组件配合进行组合使用，从而满足多方面的业务需求。

1.1 Hadoop 基础知识

1.1.1 Hadoop 的概念

今天，我们被各种各样的数据包围。人们聊天、看视频、浏览网页及更新动态等，都会使得计算机产生和保存越来越多的数据。数据的大爆发向谷歌、雅虎、亚马逊和微软等互联网巨头公司提出挑战，需要它们从与活动、交易相关的 TB 级（1TB=1024GB）和 PB 级（1PB=1024TB）数据中获取规律和隐藏信息，用于企业的业务和营销活动。而现有的工具已经不能处理如此大的数据集。

谷歌最先开发出来 MapReduce，用来应对其数据处理的需求。这个举动迅速引起其他互联网公司的关注。Doug Cutting 抓住机会，开发出了开源版本的 MapReduce，称之

为 Hadoop。随后，百度、雅虎等公司继续为其提供支持。Hadoop 发展到今天，已经成为如雅虎、淘宝、Facebook 和 Twitter 等互联网公司的计算平台的核心部分。媒体和电信行业等传统行业现在也开始采用 Hadoop 系统。

Hadoop 和大规模分布式处理技术已经成为许多程序员的一项重要技能，网络、安全和关系数据库已经成为一个高效程序员的必修课，每位程序员都应该理解分布式数据处理的基本概念。世界上的许多一流大学已经将 Hadoop 引入了他们的计算机科学教学计划中。

Hadoop 是 Apache Software Foundation 开发的一个分布式系统基础架构，是对谷歌的 MapReduce 核心技术的开源实现，应用 Java 语言进行开发。Hadoop 框架应用工程提供跨计算机集群的分布式存储和计算环境。Hadoop 可实现从单一服务器到上千台机器的扩展中，每个机器都可以提供本地计算和存储。

分布式计算是一个广泛并且不断发展变化的领域，但是 Hadoop 的独特之处在于：

（1）方便。Hadoop 可以运行在由一般机器构成的集群上，或者是云计算服务上。

（2）健壮。Hadoop 在一般商用硬件上运行时，可以从容地处理构架假设硬件频繁失效的故障。

（3）可扩展。Hadoop 通过增加集群内的节点数目，可实现扩展，从而用来处理更大的数据集。

（4）简单。用户可以简单迅速地编写出高效的并行代码。

简单方便的 Hadoop 在编写和运行大型分布式程序方面具有很大的优势。在校学生也可以快速地建立自己的 Hadoop 集群。又因为 Hadoop 的健壮性和可扩展性，它也可以承担亚马逊和雅虎等公司严苛的工作。Hadoop 的这四个优点使得 Hadoop 在学术和商业两个领域都很受欢迎。

图 1-1 描述了用户与 Hadoop 集群交互的过程。Hadoop 就是在某一地点利用网络连接的一组通用计算机，数据的存储和计算处理都由这些并行的计算机来执行。不同的用户可以从自己的客户端将计算“要求”发送到 Hadoop，这些客户端可以是远离 Hadoop 集群的台式机。但是并非所有的分布式系统的系统构建都像这幅图描述的一样。接下来，我们也简单介绍一下其他的分布式系统，以便于更好地对比理解 Hadoop。

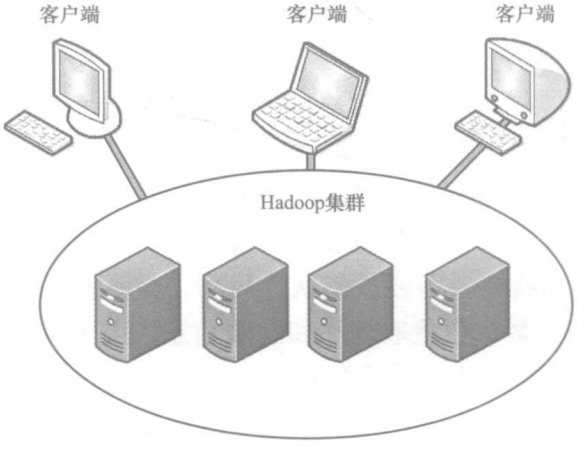

图 1-1　用户和 Hadoop 集群交互过程

1.1.2 分布式系统和 Hadoop

摩尔定律在近几十年都会伴随着我们，但是大规模数据的计算问题不能依靠制作越来越大的服务器来解决。将一组机器组织起来形成一个功能单一的分布式系统，这种解决方案已经逐渐普及。

如果使用单机系统，一台 4 个 I/O 通道的高端机，每个通道吞吐量 100MB/s，读取 4TB 的数据也要 3h。如果利用 Hadoop，数据集会被划分为较小的块（通常为 64MB），通过 Hadoop 分布式文件系统（HDFS）分布在集群内多台机器上。这些集群可以同时并行读取数据，这样就可以有很高的吞吐量。再考虑到现有 I/O 技术的性价比，这样的一组机器比一台高端服务器更加便宜。

Hadoop 在处理数据的理念上与其他的分布式系统架构也不相同。与强调把数据向代码迁移相反，Hadoop 把代码向数据迁移。Hadoop 的集群内既包括数据又包括计算环境，客户只需将需要执行的 MapReduce 程序发送到集群内即可，而这些程序一般都很小。此外，这种代码向数据迁移的理念也被应用到 Hadoop 集群自身。在集群中，数据被拆分分布，且尽量让一台计算机计算同一段数据。

Hadoop 的出现就是为了处理密集型数据，这种代码向数据迁移的理念正好契合这种设计目标。现对于数据的数量级，代码的数量级更小，也更容易在网络上移动，更节省资源和时间。

1.1.3 Hadoop 和 SQL 数据库

Hadoop 是一个数据处理框架，而当今数据处理的主导是标准的关系数据库，那么这两者有什么区别，Hadoop 有什么优势呢？其中一点，SQL（结构化查询语言）是针对结构化数据设计的，而 Hadoop 最初是针对文本这种非结构化数据。从这一方面来看，Hadoop 是更通用的模式。

下面从四个方面简单讨论两者的区别：

（1）用向外扩展代替向上扩展。Hadoop 通过增加集群内的机器数量来扩展资源，而不是去购买一个更大的机器。

（2）用键/值对代替关系表。SQL 针对结构化查询语句，是结构化数据。Hadoop 要处理的大型数据集一般都是非结构化或者半结构化的数据，比如文本、图片和 XML 文件等文件形式。而 Hadoop 使用键值对的方式，将任何形式的数据都转化为键值对，以便于能够更灵活地处理这些数据类型。

（3）用函数式编程（MapReduce）代替声明式查询（SQL）。SQL 使用查询语句，而 Hadoop 使用脚本和代码。此外，Hadoop 还有一些 SQL 不能完成的任务，例如利用读取出的数据来建立复杂的模型或者改变图片格式，但是当数据处理非常适合于关系型数据结构时，使用 MapReduce 会不太方便。

（4）用离线批量处理代替在线处理。Hadoop 是专为离线处理和大规模数据分析而设计的，它并不适合那种对几个记录随机读写的在线事务处理模式。

1.1.4 Hadoop 的构造模块

在此之前，我们已经讨论了分布式存储和分布式计算的相关概念。那么 Hadoop 是如何实现这些思想的呢？

在一个全配置的集群上，“运行 Hadoop”意味着在网络分布的不同服务器上运行一组守护进程。这些守护进程有特殊的角色，一些仅存在单个服务器上，一些则运行在多个服务器上。它们主要有：NameNode（名字节点）、DataNode（数据节点）、Secondary NameNode（次名字节点）、JobTracker（作业跟踪节点）、TaskTracker（任务跟踪节点）。我们逐一讨论并定位它们在 Hadoop 中的作用。

（1）NameNode（NN）是 HDFS 的守护程序，记录了文件如何被拆分成数据块（block），以及这些数据块都存储到了哪里的 DataNode 节点上。NameNode 同时保存了文件系统运行的状态信息，它的主要功能是集中管理内存和 I/O 口。在 Hadoop 的集群中，NameNode 是一个单点，所以 NameNode 发生故障时整个系统将无法进行。

（2）DataNode（DN）是负责存储被拆分的数据块。在集群中的每个服务器都运行一个 DataNode，它负责把 HDFS 数据块读/写到本地的文件系统中。

（3）Secondary NameNode（SNN）是监控 HDFS 状态的辅助后台程序，帮助 Name Node 搜集文件系统运行的状态信息。在 Hadoop 的集群中，Secondary NameNode 与 Name Node 一样也只有一个，且是在一个单独的服务器上。在 NameNode 发生故障时，Secondary NameNode 作为备用 NameNode 使用。

（4）JobTracker（JT）的作用是连接应用程序和 Hadoop，在有任务提交到 Hadoop 集群时负责 Job 的运行，调度多个 TaskTracker。在 Hadoop 的集群中，JobTracker 也只有一个，一般部署在 Master 节点上。

（5）TaskTracker（TT）与负责存储数据的 DataNode 结合，负责某一个 Map 或 Reduce 任务。TaskTracker 还负责与 JobTracker 交互，JobTracker 如果没有收到 TaskTracker 提交出来的信息，就会判定 TaskTracker 已经崩溃，并且把任务分配给其他节点。

讨论 Hadoop 的各个守护进程之后，在图 1-2 中描绘了一个典型的 Hadoop 集群的拓扑结构。

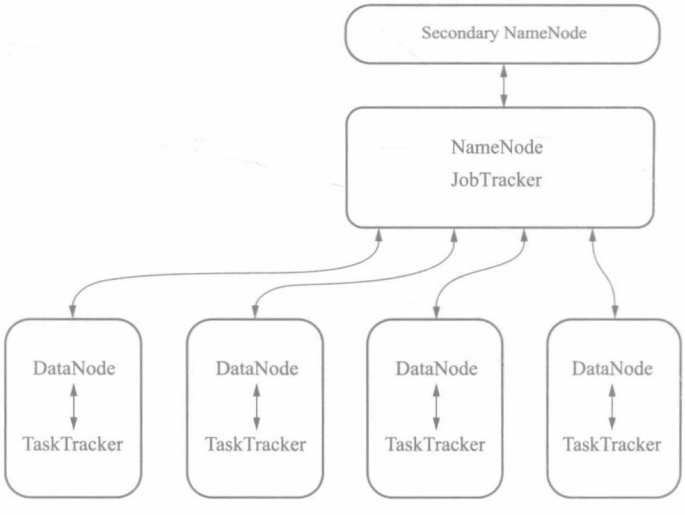

图 1-2 Hadoop 集群

这种拓扑结构的特点是在主节点上运行 NameNode 和 JobTracker 的守护进程，并使用独立的节点运行 SNN 以防主节点失效。在小集群中，SNN 也可以驻留在某一从节点上，而在大型集群中，连 NameNode 和 JobTracker 都会分别驻留在两台机器上，每个从节点均驻留一个 DataNode 和 TaskTracker，从而在存储数据的同一节点上执行任务。

1.1.5 Hadoop 环境配置

首先介绍一下安装环境：1 台 NameNode（CentOSNameNode：192.168.114.131），两台 DataNode（CentOSDataNode1：192.168.114.130，CentOSDataNode2：192.168.114.132），全部都是 CentOS6.4 系统，用户全都名为 Hadoop。具体的安装步骤如下：

1. 安装配置 JDK

首先，卸载 Linux 默认安装的 Java 版本（如果没有默认版本请忽略），使用 yum list installed|grep java 或者 rpm-qa|grep gcj 查看。卸载命令为 yum-yremove java-1.4.2-gcj-compat 或者 rpm-e--nodeps java-1.4.2-gcj-compat-1.4.2.0-40jpp.115，卸载完后查看是否成功。下载最新版本的 JDK，放到/usr/local 文件夹下：

```
sudo cp jdk-7u51-linux-x64.tar.gz /usr/local/
```

然后在目标下解压安装包即可。

```
sudo tar -zxvf jdk-7u51-linux-x64.tar.gz
```

修改配置文件：

```
sudo vim /etc/profile
export JAVA_HOME=/usr/local/java/jdk1.7.0_51
export JRE_HOME=$JAVA_HOME/jre
export CLASSPATH=.:$JAVA_HOME/lib/dt.jar:$JAVA_HOME/lib/tools.jar
export PATH=$PATH:$JAVA_HOME/bin
```

保存完毕之后，在文件末尾如下添加，然后更新文件。

```
source /etc/profile
```

最后，输入测试信息，验证配置是否正确，若出现 Java 的版本信息说明安装成功。

```
[root@localhost java]# java -version
java version "1.6.0_45"
Java(TM) SE Runtime Environment (build 1.6.0_45-b06)
Java HotSpot(TM) Client VM (build 20.45-b01, mixed mode, sharing)
```

2. SSH 无钥登录

Hadoop 在运行过程中需要对远程 Hadoop 守护进程，Hadoop 启动之后，NameNode 是通过 SSH 来启动和停止各个 DataNode 上的各种守护进程的。这就要求节点之间执行指令时要实现不输入密码就能登录，这样 NameNode 使用 SSH 无密码登录并启动 DataName 进程。Hadoop 自身具有 SSH 的资源，但是并没有安装，因此每台机器上都需要安装 SSH，实现每台机器之间的无密码登录。

这里有两台机器，IP 地址分别为：主机 A：10.0.5.199，主机 B：10.0.5.198，需要配置主机 A 无密码登录主机 A、主机 B。先确保所有主机的防火墙处于关闭状态。在主机 A 上执行如下：

```
$ssh-keygen -t rsa
```

然后一直按回车键，就会按照默认的选项将生成的密钥保存在.ssh/id_rsa 文件中。

```
$cd ~/.ssh
$cp id_rsa.pub authorized_keys
```

这步完成后，正常情况下就可以无密码登录本机了，即 ssh localhost，无须输入密码。把刚刚产生的 authorized_keys 文件拷一份到主机 B 上：

```
$scp authorized_keys summer@10.0.5.198:/home/summer/.ssh
```

正常情况下上面几步执行完成后，从主机 A 所在机器向主机 A、主机 B 所在机器发起 SSH 连接，只有在第一次登录时需要输入密码，以后则不需要。下面是一些可能遇到的问题以及解决方法：

（1）进行 SSH 登录时，出现："Agent admitted failure to sign using the key"，则执行下面命令，强行把私钥加进来。

```
$ssh-add
```

（2）如果无任何错误提示，可以输密码登录，但就是不能无密码登录，在被连接的主机上（如 A 向 B 发起 SSH 连接，则在 B 上）执行以下几步：

```
$chmod o-w ~/
$chmod 700 ~/.ssh
$chmod 600 ~/.ssh/authorized_keys
```

（3）如果执行了第（2）步，还是不能无密码登录，再试试下面的操作：

```
$ps -Af | grep agent
```

检查 SSH 代理是否开启，如果有开启的话，kill 掉该代理，然后执行下面，重新打开一个 SSH 代理，如果没有开启，直接执行下面：

```
$ssh-agent
```

还是不行的话，执行下面，重启一下 SSH 服务：

```
$sudo service sshd restart
```

（4）执行（1）时提示"Could not open a connection to your authenticationh agent"而失败，则执行：

```
$ssh-agent bash
```

3．Hadoop 安装

（1）关闭防火墙：chkconfig --level 35 iptables off（此命令要用管理员权限执行），之后重启一下虚拟机。

（2）配置机器的 host 文件：/etc/hosts（此命令要用管理员权限执行），配置完后的文件如下图：

```
[hadoop@centosnamenode ~]$ cat /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.114.131 CentOSNameNode
192.168.114.130 CentOSDataNode1
192.168.114.132 CentOSDataNode2
```

（3）配置无密码互联 SSH，在三台机器上输入：ssh-keygen -t rsa，之后会在/home/hadoop/.ssh/生成两个文件，如下：

```
[hadoop@centosnamenode .ssh]$ cd /home/hadoop/.ssh/
[hadoop@centosnamenode .ssh]$ ll |grep 'id'
-rw-------. 1 hadoop hadoop 1675 Jul 21 07:28 id_rsa
-rw-r--r--. 1 hadoop hadoop  397 Jul 21 07:28 id_rsa.pub
```

（4）将 NameNode 上面的 id_rsa.pud 文件发送给两台子节点并命名为 authorized_keys：

```
scp id_rsa.pud hadoop@192.168.114.130:/home/hadoop/.ssh/authorized_keys
scp id_rsa.pud hadoop@192.168.114.132:/home/hadoop/.ssh/authorized_keys
```

将两台 DataNode 的 id_rsa.pud 发送给主节点并合并为 authorized_keys：

```
scp id_rsa.pud hadoop@192.168.114.130:/home/hadoop/.ssh/id_rsa.pud130
scp id_rsa.pud hadoop@192.168.114.132:/home/hadoop/.ssh/id_rsa.pud132
cat id_rsa.pud130 id_rsa.pud132 > authorized_keys
```

最后三台机器的/home/hadoop/.ssh 的文件如下：

```
[hadoop@centosnamenode .ssh]$ ll
total 16
-rw-r--r--. 1 hadoop hadoop 1193 Jul 21 10:03 authorized_keys
-rw-------. 1 hadoop hadoop 1675 Jul 21 07:28 id_rsa
-rw-r--r--. 1 hadoop hadoop  397 Jul 21 07:28 id_rsa.pub
-rw-r--r--. 1 hadoop hadoop 2000 Jul 21 10:14 known_hosts
```

（5）下载软件，以 jdk1.8+hadoop-1.2.1 的软件版本为例，下载到的位置是/home/hadoop/Downloads 文件夹，如下：

```
[hadoop@centosnamenode Downloads]$ ll
total 192500
-rwxrw-rw-. 1 hadoop hadoop  38096663 Jul 21 08:05 hadoop-1.2.1-bin.tar.gz
-rwxrw-rw-. 1 hadoop hadoop 159019376 Jul 21 07:06 jdk-8u11-linux-x64.gz
```

之后用命令解压缩两个文件：

```
tar -zxvf hadoop-1.2.1-bin.tar.gz
ar -zxvf jdk-8u11-linux-x64.gz
```

然后用管理员权限将两个文件移动到/usr/local 文件夹下，并赋予 Hadoop 用户权限。

```
su: password
mkdir /usr/local/java
mv hadoop-1.2.1 /usr/local/
mv jdk1.8.0_11 /usr/local/java/
chown -R hadoop hadoop-1.2.1/
chown -R hadoop java/
chgrp -R hadoop hadoop-1.2.1/
chgrp -R hadoop java/
```

之后目录如下：

```
[hadoop@centosnamenode local]$ ll
total 48
drwxr-xr-x.  2 root   root   4096 Sep 23  2011 bin
drwxr-xr-x.  2 root   root   4096 Sep 23  2011 etc
drwxr-xr-x.  2 root   root   4096 Sep 23  2011 games
drwxr-xr-x. 16 hadoop hadoop 4096 Jul 22 04:26 hadoop-1.2.1
drwxr-xr-x.  2 root   root   4096 Sep 23  2011 include
drwxr-xr-x.  3 hadoop hadoop 4096 Jul 21 08:37 java
drwxr-xr-x.  2 root   root   4096 Sep 23  2011 lib
drwxr-xr-x.  2 root   root   4096 Sep 23  2011 lib64
drwxr-xr-x.  2 root   root   4096 Sep 23  2011 libexec
drwxr-xr-x.  2 root   root   4096 Sep 23  2011 sbin
drwxr-xr-x.  5 root   root   4096 Mar 13 16:13 share
drwxr-xr-x.  2 root   root   4096 Sep 23  2011 src
```

（6）配置 Hadoop，主要的配置文件如下：

Hadoop-env.sh：只需要加一行：

```
# The java implementation to use.  Required.
export JAVA_HOME=/usr/local/java/jdk1.8.0_11
```

还可以多加一行 export HADOOP_HOME_WARN_SUPPRESS="TRUE"来屏蔽"WARM：HADOOP_HOME is deprecated"提醒。

core-site.xml：

```
[hadoop@centosnamenode conf]$ cat core-site.xml
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<!-- Put site-specific property overrides in this file. -->

<configuration>
  <property>
    <name>fs.default.name</name>
    <value>hdfs://CentOSNameNode:9000</value>
  </property>
  <property>
    <name>hadoop.tmp.dir</name>
    <value>/usr/local/hadoop-1.2.1/tmp</value>
  </property>
</configuration>
```

这里将 hadoop.tmp.dir 更改了位置，避免被 Linux 重启的时候删掉。

hdfs-site.xml：

```
[hadoop@centosnamenode conf]$ cat hdfs-site.xml
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<!-- Put site-specific property overrides in this file. -->

<configuration>
  <property>
    <name>dfs.replication</name>
    <value>2</value>
  </property>
  <property>
    <name>dfs.name.dir</name>
    <value>/hdfs/name</value>
  </property>
  <property>
    <name>dfs.data.dir</name>
    <value>/hdfs/data</value>
  </property>
</configuration>
```

配置文件的冗余是 2，并且重新配置了 dfs.name.dir，dfs.data.dir 的位置，关于位置选定，首先用命令：df-h 查看一下磁盘空间：

```
[hadoop@centosnamenode conf]$ df -h
Filesystem            Size  Used Avail Use% Mounted on
/dev/sda2              18G  3.4G   14G  20% /
tmpfs                 495M  228K  495M   1% /dev/shm
/dev/sda1             291M   33M  243M  12% /boot
```

然后选择较大的那个下面建立 HDFS 文件夹，并且把权限赋予 Hadoop，三台机器全都如此操作。

```
mkdir /hdfs
chgrp -R hadoop /hdfs/
chgrp -R hadoop /hdfs/
```

这里注意，千万不要继续往下创建 name 文件夹和 data 文件夹，这会造成 Hadoop namenode -fomart 不成功。

为了保护数据，不允许对已经存在的 dfs.name.dir 格式化。

1）mapred-site.xml：

```
[hadoop@centosnamenode conf]$ cat mapred-site.xml
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<!-- Put site-specific property overrides in this file. -->

<configuration>
  <property>
    <name>mapred.job.tracker</name>
    <value>CentOSNameNode:9001</value>
  </property>
</configuration>
```

2）Masters：

```
[hadoop@centosnamenode conf]$ cat masters
CentOSNameNode
```

3）Slaves：

```
[hadoop@centosnamenode conf]$ cat slaves
CentOSDataNode1
CentOSDataNode2
```

（7）至此 Hadoop 就配置好了，用 scp-r 命令将 Hadoop 和 Java 的文件夹 copy 到两个子节点，然后在三台机器上用管理员权限修改/etc/profile 文件，加入几行配置如下：

```
export JAVA_HOME=/usr/local/java/jdk1.8.0_11
export JRE_HOME=$JAVA_HOME/jre
export CLASSPATH=.:$JAVA_HOME/lib/dt.jar:$JAVA_HOME/lib/tools.jar
export HADOOP_HOME=/usr/local/hadoop-1.2.1
export PATH=$PATH:$JAVA_HOME/bin:$HADOOP_HOME/bin:$HADOOP_HOME/sbin
```

之后用命令：source/etc/profile 重新加载一下配置文件，集群的基本配置就完成了。最后需用 java-version 命令检查一下 java 是否配置成功：

```
[hadoop@centosnamenode conf]$ java -version
java version "1.8.0_11"
Java(TM) SE Runtime Environment (build 1.8.0_11-b12)
Java HotSpot(TM) 64-Bit Server VM (build 25.11-b03, mixed mode)
```

（8）回到 NameNode，运行命令：

```
hadoop namenode -format
start-all.sh
```

然后用命令 jps 检查一下，主节点：