分布式系统 Homework 1

张言健 16300200020

目录

1 分析对比 HDFS 与传统的分布式/网络文件系统

1.1 HDFS 架构

HDFS 的架构建立在大量普通配置的计算机组成的集群上。HDFS 采用主从架构，一个集群有多个 Master 和多个 Salve，前者成为名字节点，后者称为数据节点，理论上一个计算机可以运行多个 DataNode 进程，一个 NameNode 进程。但是实际情况汇总往往是一台计算机只运行一个 DataNode 或 NameNode。一个文件被分割成若干 Block 存储在一组 DataNode 上。

NameNode 负责打开、关闭和重命名文件以及目录，同时建立 Block 与 DataNode 之间的映射。

DataNode 负责响应客户的读/写需求，同时在 NameNode 的指挥下实现 Block 的建立、删除以及复制。

1.2 其与传统文件系统的比较

HDFS 的 DataNode 以及 NameNode 都是运行在 Linux 上，因此 Linux 是 HDFS 的底层文件系统

1.2.1 目录树上的差别

Linux FS 由 Dentry 来组织，dentry 下可以有多个 dentry, dentry 的末尾节点链接一个 INode。

而 HDFS 由 INodeDirectory 来组织，每个 INodeDirectory 下可以含有多个 INode.

1.2.2 数据块上的差别

Block 是 Linux FS 操作的最小单元，大小相等，大小一般为 4MB，与 DataNode 之间的关系是固定的。HDFS 的最小单元也是 Block, 大小可以由用户定义，默认为 64MB，与 DataNode 之间的对应关系是动态的。

1.2.3 索引节点上的差别

Linux FS 中的每个问价都由一个 INode 代表，INode 中定义一组外存上的 Block。

HDFS 中 INode 是目录树的单眼，HDFS 的目录树以此为基础建立。它们分为两类，一类是文件，指向一组 Block，一类是 INode，指向一组子 INode。

1.2.4 目录项上的差别

Dentry 是目录树的核心数据结构，目录树由一个子父 Dentry 搭建起来，包含软硬连接的实现，而在 HDFS 中不存在 Dentry。

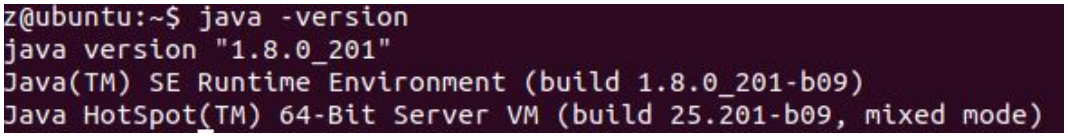

图 1: java 安装成功

1.2.5 设计侧重点差别

HDFS 在设计上的侧重点是节约 NameNode 内外存空间，一方面 HDFS 没有 Dentry，INode 没有子节点。另一方面，INodeDirectory 没有以一组 Block 的形式存储在外存。

2 部署报告

2.1 部署实验设置

操作系统	Ubuntu 18.04
JDK	java 1.8.0.201
Hadoop	3.2.0

2.2 配置环境过程

2.2.1 安装 SSH、配置 SSH 无密码登陆

```
sudo apt-get install openssh-server
```

为了避免每一次使用 SSH 都要输入密码，利用 ssh-keygen 生成密钥，并将密钥加入到授权中：

```
cd ~/.ssh/
ssh-keygen -t rsa
cat ./id_rsa.pub >> ./authorized_keys
```

登陆本机

```
ssh localhost
```

2.2.2 安装 java 环境

从官网上下载 java 安装，安装后结果如图 1

```
z@ubuntu:/usr/local$ hadoop/bin/hadoop version
Hadoop 3.2.0
Source code repository https://github.com/apache/hadoop.git -r e97acb3bd8f3befd2
7418996fa5d4b50bf2e17bf
Compiled by sunilg on 2019-01-08T06:08Z
Compiled with protoc 2.5.0
From source with checksum d3f0795ed0d9dc378e2c785d3668f39
```

图 2: Hadoop 安装成功

2.2.3 安装 hadoop 环境

从官网上下载 java 安装，安装后结果如图 2

2.2.4 配置 bashrc 环境

在 /.bashrc 末尾添加如下环境内容

```
export HADOOP_HOME=/usr/local/hadoop
export HADOOP_INSTALL=$HADOOP_HOME
export HADOOP_MAPRED_HOME=$HADOOP_HOME
export HADOOP_COMMON_HOME=$HADOOP_HOME
export HADOOP_HDFS_HOME=$HADOOP_HOME
export YARN_HOME=$HADOOP_HOME
export HADOOP_COMMON_LIB_NATIVE_DIR=$HADOOP_HOME/lib/native
export PATH=$PATH:$HADOOP_HOME/sbin:$HADOOP_HOME/bin
```

2.3 伪分布式部署

2.3.1 修改 core-site.xml 文件

修改./etc/hadoop/内的 core-site.xml 文件, 添加如下内容

```
<configuration>
    <property>
        <name>fs.defaultFS</name>
        <value>hdfs://localhost:9000</value>
    </property>

    <property>
        <name>hadoop.tmp.dir</name>
        <value>/usr/local/hadoop/tmp</value>
        <description>Abase for other temporary directories.</description>
    </property>
</configuration>
```

2.3.2 修改 hdfs-site.xml 文件

修改./etc/hadoop/内的 hdfs-site.xml 文件, 添加如下内容

```
<configuration>
    <property>
        <name>dfs.replication</name>
        <value>4</value>
    </property>
    <property>
        <name>dfs.name.dir</name>
        <value>/usr/local/hadoop/hdfs/name</value>
    </property>
    <property>
        <name>dfs.data.dir</name>
        <value>/usr/local/hadoop/hdfs/data</value>
    </property>
    <property>
        <name>dfs.http.address</name>
        <value>localhost:50070</value>
    </property>
</configuration>
```

2.3.3 NameNode 及 DataNode 的配置

执行 NameNode 的格式化:

```
./bin/hdfs namenode -format
```

开启 NameNode 和 DataNode 守护进程。使用 jps 判断是否配置成功，如图 3 ，需要注意到此时应该包含 NameNode 以及 DataNode。成功配置之后可以访问 http://localhost:50070（在 hdfs-site.xml 中的地址）来查看如图 4 的详细信息

```
./sbin/start-dfs.sh
jps
```

图 3: 第一次 jps 的结果

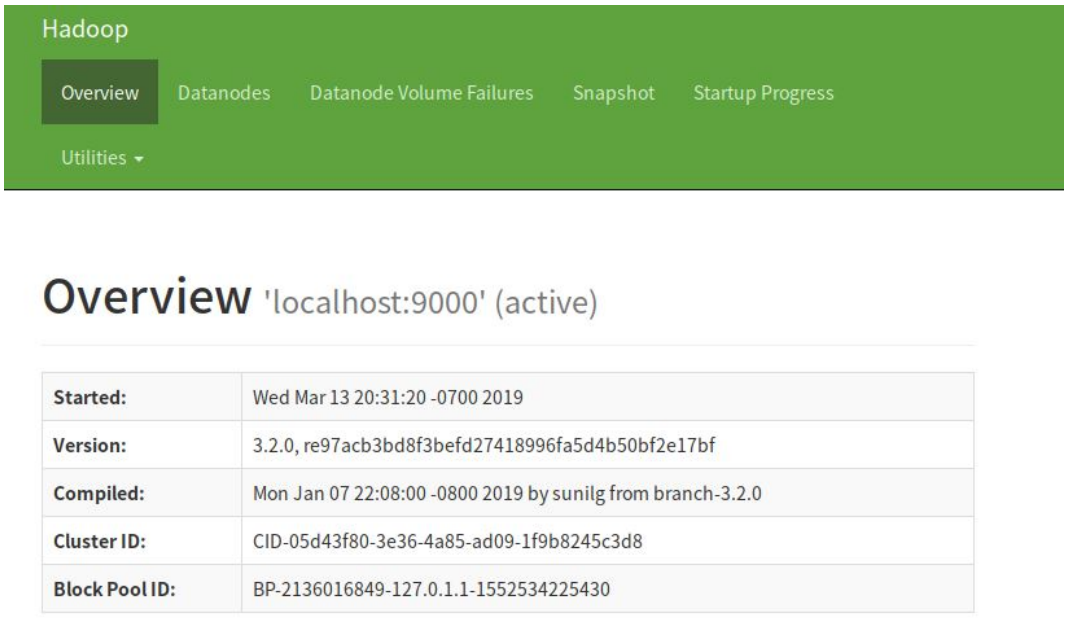

图 4: 浏览器详细信息

2.4 yarn 的配置

2.4.1 修改 mapred-site.xml 文件

添加如下内容

```
<configuration>
    <property>
        <name>mapreduce.framework.name</name>
        <value>yarn</value>
    </property>
</configuration>
```

2.4.2 修改 yarn-site.xml 文件

添加如下内容

```
<configuration>
    <property>
        <name>yarn.nodemanager.aux-services</name>        <value>
   mapreduce_shuffle</value>
    </property>
</configuration>
```

2.4.3 启动 yarn

运行如下命令，运行结果如图 5 所示，此时应该多了 NodeManager 和 ResourceManager 两个后台进程。随后访问 localhost:8088/cluster，如图 6 查看任务运行情况。至此伪分布式部署完毕。

```
start-yarn.sh
mapred --daemon start historyserver
jps
```

3 参考

分析对比 HDFS 与传统的文件系统参考:
分布式文件系统 Hadoop HDFS 与传统文件系统 Linux FS 的比较与分析
Hadoop 伪分布式的配置参考:
Ubuntu 18.04.1 LTS 搭建 Hadoop 环境

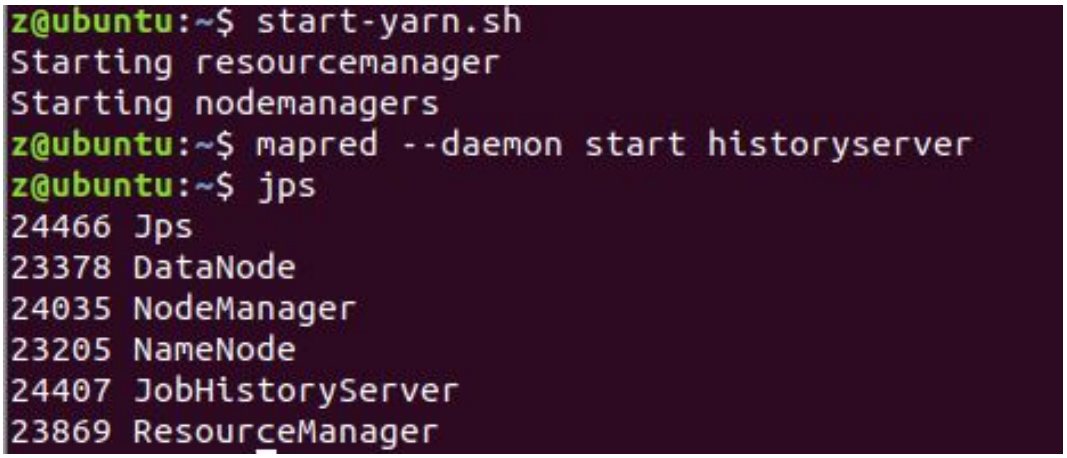

图 5: 第二次 jps 的结果

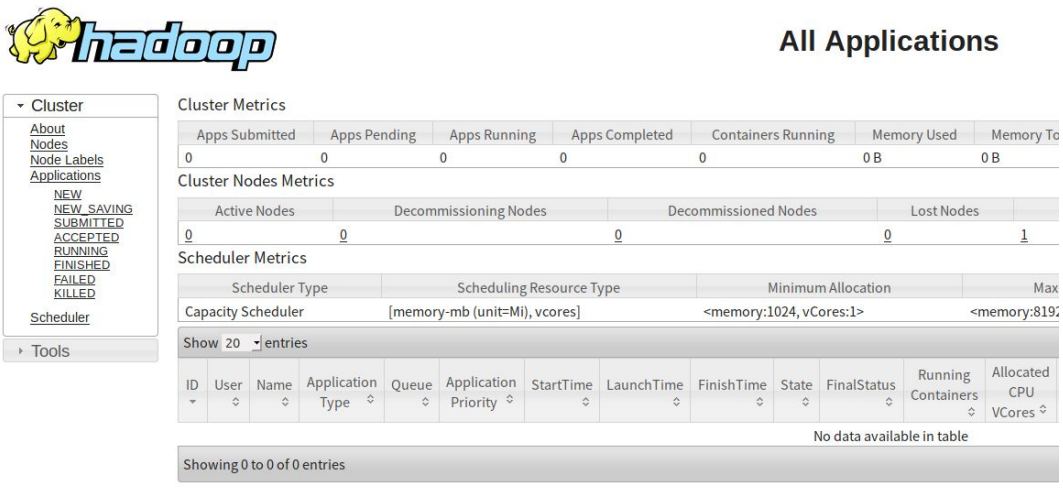

图 6: 任务运行情况