金融大数据处理技术 实验2

171870604 袁满杰

1.环境部署

为了方便之后多机的部署，采用Docker配置Hadoop环境，构建Dockerfile如下：

```
1  FROM ubuntu
2  MAINTAINER While
3  RUN mkdir /usr/local/java \
4         && apt-get update\
5           && apt-get install -y openssh-server vim\
6             && ssh-keygen -t rsa -f ~/.ssh/id_rsa -P ''\
7               && cat /root/.ssh/id_rsa.pub >> /root/.ssh/authorized_keys\
8                 && sed -i 's/PermitEmptyPasswords yes/PermitEmptyPasswords
   no /' /etc/ssh/sshd_config\
9                   && sed -i 's/PermitRootLogin without-
   password/PermitRootLogin yes /' /etc/ssh/sshd_config\
10                    && echo " StrictHostKeyChecking no" >>
   /etc/ssh/ssh_config\
11                      && echo " UserKnownHostsFile /dev/null" >>
   /etc/ssh/ssh_config\
12                        && echo "root:123456" | chpasswd
13
14 ADD jdk-8u221-linux-x64.tar.gz /usr/local/java/
15 ADD hadoop-3.2.1.tar.gz /
16
17 ENV JAVA_HOME /usr/local/java/jdk1.8.0_221
18 ENV JRE_HOME ${JAVA_HOME}/jre
19 ENV HADOOP_HOME /hadoop-3.2.1
20 ENV CLASSPATH .:${JAVA_HOME}/lib:${JRE_HOME}/lib
21 ENV PATH ${JAVA_HOME}/bin:${HADOOP_HOME}/bin:${HADOOP_HOME}/sbin:$PATH
22
23 CMD [ "sh", "-c", "service ssh start; bash"]
```

其中第三行的`RUN`安装vim和ssh服务并配置密钥（这里我同上一个实验一样使用配置本机免密访问本机后构建镜像，从而使得该镜像的所有容器有相同的RSA公私钥实现相互免密访问）；

第14~15行`ADD`拷贝宿主机上的jdk和Hadoop安装包进行解压/安装；

最后设置环境变量及启动ssh服务。

以此为基本Hadoop环境的镜像构建容器进行以下三个实验。

2. 单机模式运行Hadoop样例程序

首先运行结果如下：

```
2019-10-02 06:19:02,752 INFO mapreduce.Job: Job job_local269050207_0002 running in uber mode : false
2019-10-02 06:19:02,752 INFO mapreduce.Job:  map 100% reduce 100%
2019-10-02 06:19:02,753 INFO mapreduce.Job: Job job_local269050207_0002 completed successfully
2019-10-02 06:19:02,758 INFO mapreduce.Job: Counters: 30
        File System Counters
                FILE: Number of bytes read=1336130
                FILE: Number of bytes written=3348954
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
        Map-Reduce Framework
                Map input records=1
                Map output records=1
                Map output bytes=17
                Map output materialized bytes=25
                Input split bytes=117
                Combine input records=0
                Combine output records=0
                Reduce input groups=1
                Reduce shuffle bytes=25
                Reduce input records=1
                Reduce output records=1
                Spilled Records=2
                Shuffled Maps =1
                Failed Shuffles=0
                Merged Map outputs=1
                GC time elapsed (ms)=0
                Total committed heap usage (bytes)=2862612480
        Shuffle Errors
                BAD_ID=0
                CONNECTION=0
                IO_ERROR=0
                WRONG_LENGTH=0
                WRONG_MAP=0
                WRONG_REDUCE=0
        File Input Format Counters
                Bytes Read=123
        File Output Format Counters
                Bytes Written=23
root@a103cd11c7cc:/hadoop-3.2.1# cat output/*
1       dfsadmin
root@a103cd11c7cc:/hadoop-3.2.1#
```

其次，具体步骤上.....按照实验手册上教程来就好了，唯一遇到的两个问题：

- Hadoop运行时不认JAVA_HOME环境变量：

  > ERROR: JAVA_HOME is not set and could not be found.

- 貌似不允许root运行Hadoop：

  > ERROR: Attempting to operate on hdfs namenode as root
  > ERROR: but there is no HDFS_NAMENODE_USER defined. Aborting operation.

  解决方案：

通过在hadoop-env.sh配置文件中加入以下语句可以顺利解决(顺便也为之后启动yarn配置了这个)：

```
export JAVA_HOME=/usr/local/java/jdk1.8.0_221
export HDFS_NAMENODE_USER="root"
export HDFS_DATANODE_USER="root"
export HDFS_SECONDARYNAMENODE_USER="root"
export YARN_RESOURCEMANAGER_USER="root"
export YARN_NODEMANAGER_USER="root"
```

最后顺利得到运行结果

3. 伪分布模式运行Hadoop样例程序

首先运行结果如下：

```
root@d2cb5bf1a4df:/hadoop-3.2.1# hdfs dfs -get output2 output
2019-10-03 03:42:13,197 INFO sasl.SaslDataTransferClient: SASL encryption trust check: localHostTrusted = false, remoteHostTrusted = false
root@d2cb5bf1a4df:/hadoop-3.2.1# cat output/*
5       dfs.audit.logger
3       dfs.logger
3       dfs.server.namenode.
2       dfs.audit.log.maxbackupindex
2       dfs.sh
2       dfs.audit.log.maxfilesize
1       dfsadmin
1       dfs.replication
1       dfs.namenode.servicerpc
1       dfs.namenode.rpc
1       dfs.log
1       dfs.http.address
1       dfs.namenode.ec.policies.max.cellsize
root@d2cb5bf1a4df:/hadoop-3.2.1#
```

其次，在创建容器时使用`-p`参数指定端口映射从而可以访问Hdfs的ui网页：

```
$ docker run -it -p 8000:50090 hadoop
```

之后也是具体步骤上参照实验手册来配置文件就好了；

其中在修改好配置文件之后我将容器通过`docker commit`保存为镜像，这样以后遇到问题想重来就可以直接从这一镜像生成容器进行操作；

其中同样遇到两个问题：

1.宿主机上无法访问Hdfs的ui网页：

经过分析，是由于默认情况下该端口开放在容器中localhost:50070，不对外开放，因此只有容器内部可以访问；

解决方法是在hdfs-site.xml中增加配置：

```
<property>
  <name>dfs.http.address</name>
  <value>0.0.0.0:50070</value>
</property>
```

从而将其改为0.0.0.0:50070，如此便可以在宿主机上、甚至从外网访问宿主机映射后的8000端口进入Hdfs的ui页面了。

2. 重新格式化NameNode之后无法启动DataNode:

遇到之后要提到的第三个问题时，我多次尝试过从头重新再来，因此重新格式化了NameNode，但在没有删除DataNode的情况下重新格式化NameNode会导致其ID与DataNode不同，从而在启动HDFS时会无法启动DataNode（但start-dfs.sh不会报错，在之后拷贝文件进HDFS时才会出错，通过jps命令发现DataNode没有启动）

解决方法有两种：

1. 在HDFS开启状态下进入缓存文件保存地址（默认为/tmp下），然后从name/current/VERSION文件中将第三行的ID复制下来，修改data/current/VERSION中同样位置的ID为一样的，之后就可以正常启动DataNode了
2. 直接删除整个dfs文件夹，让系统重新生成；

3. 显示map0%,reduce 100%之后看似正常结束，但输出结果为空

这个问题卡了我一天……显示上诡异的map0%,reduce 100%，任务也SUCCESS完成,但是结果为空

```
2019-10-03 03:46:19,751 INFO mapreduce.Job: Job job_local2081552833_0002 running in uber m
ode : false
2019-10-03 03:46:19,752 INFO mapreduce.Job:  map 0% reduce 100%
2019-10-03 03:46:19,752 INFO mapreduce.Job: Job job_local2081552833_0002 completed success
fully
2019-10-03 03:46:19,768 INFO mapreduce.Job: Counters: 30
        File System Counters
                FILE: Number of bytes read=633251
                FILE: Number of bytes written=1676052
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=0
                HDFS: Number of bytes written=0
                HDFS: Number of read operations=12
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=5
                HDFS: Number of bytes read erasure-coded=0
        Map-Reduce Framework
                Combine input records=0
                Combine output records=0
                Reduce input groups=0
                Reduce shuffle bytes=0
                Reduce input records=0
                Reduce output records=0
                Spilled Records=0
                Shuffled Maps =0
```

最后翻去查logs文件夹下的运行日志，发现其中问题所在：

> 2019-10-03 03:33:42,249 INFO org.apache.hadoop.ipc.Server: IPC Server handler 6 on default port 9000, call Call#67 Retry#0 org.apache.hadoop.hdfs.protocol.ClientProtocol.getBlockLocations from 172.17.0.2:45456: java.io.FileNotFoundException: Path is not a file: /user/root/input/shellprofile.d

原因在于将整个`etc/hadoop`目录拷贝到HDFS中时其中有一个`shellprofile.d`文件夹（猜测为新版Hadoop中加入的文件夹，因此手册上示例中没问题）也被移入其中，（若单机情况下用cp命令不加参数会提示文件夹无法复制从而跳过该项，之后便可以正常运行），而grep 命令时无法对文件夹处理，因此map引发错误中断，reduce对空输出进行处理最终输出也为空（可以看出mapreduce的稳定性很强……即使map程序出错也能顺利进行reduce,输出Success…）

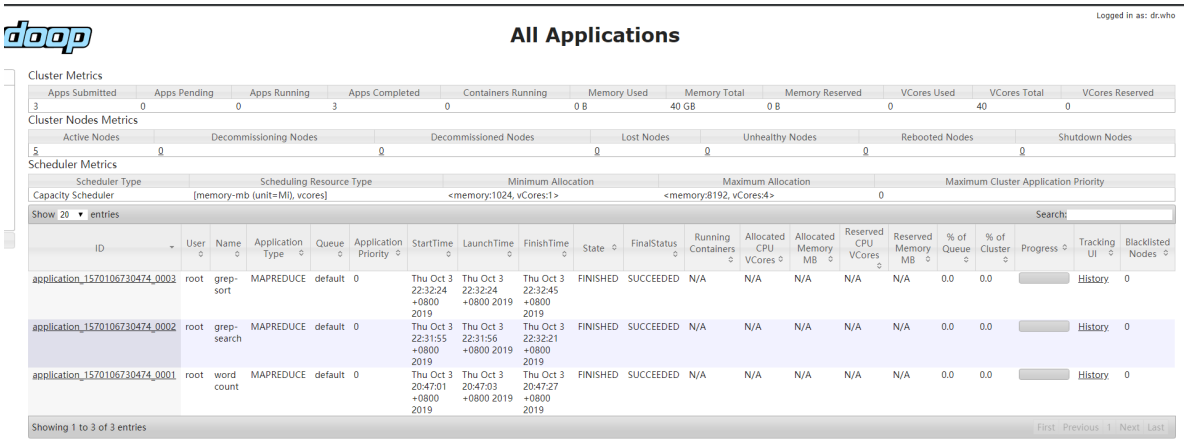
所以只要将input文件夹中这个子文件夹删除掉就可以正常运行了

4. 集群模式运行Hadoop样例程序

运行结果如下：

```
                Bytes Written=107
root@4f1a3b914bc7:/# hdfs dfs -cat output2/*
2019-10-03 14:33:03,065 INFO sasl.SaslDataTransferClient: SASL encryption trust check: loc
alHostTrusted = false, remoteHostTrusted = false
1       dfsadmin
1       dfs.replication
1       dfs.namenode.secondary.http
1       dfs.namenode.name.dir
1       dfs.datanode.data.dir
root@4f1a3b914bc7:/#
```

在其ui界面中可以看到任务完成情况如下(第一条)：



首先从之前配置好的Hadoop镜像建立一个临时容器，进入后安装实验手册改写配置文件，这里worker设为h1,h2,h3,h4,h5;主节点设为h0；此外在之前提到的hadoop-env.sh中修改也需加入其中；

之后将修改好配置文件的容器通过`docker commit`设为镜像hadoop_3，并编写一下shell脚本据此生成一批容器：

```
#! /bin/bash

for i in {1..5}
do
  docker run -dit --name=h$i --net hadoop-net hadoop_3
done
docker run -dit --name=h0 -p 8000:50090 -p 8001:8088 -p 8002:19888 --net
hadoop-net hadoop_3
```

其中for循环建立了一批worker，加入`hadoop-net`虚拟网络中；最后一行将HDFS、yarn的ui端口进行了映射建立主节点

然后通过`docker attach h0`进入其bash界面按照实验手册进行后续操作，运行得到结果。

这里只遇到一个问题：

运行时yarn报错:

大概长这样：

```
[2019-04-16 06:25:41.902]Container exited with a non-zero exit code 1. Error file: prelaunch.err.
Last 4096 bytes of prelaunch.err :
Last 4096 bytes of stderr :
Error: Could not find or load main class org.apache.hadoop.mapreduce.v2.app.MRAppMaster

Please check whether your etc/hadoop/mapred-site.xml contains the below configuration:
<property>
  <name>yarn.app.mapreduce.am.env</name>
  <value>HADOOP_MAPRED_HOME=${full path of your hadoop distribution directory}</value>
</property>
<property>
  <name>mapreduce.map.env</name>
  <value>HADOOP_MAPRED_HOME=${full path of your hadoop distribution directory}</value>
</property>
<property>
  <name>mapreduce.reduce.env</name>
```


只需按照其解释修改`mapred-site.xml`加入以下配置即可：

```
<property>
  <name>yarn.app.mapreduce.am.env</name>
  <value>HADOOP_MAPRED_HOME=${HADOOP_HOME}</value>
</property>
<property>
  <name>mapreduce.map.env</name>
  <value>HADOOP_MAPRED_HOME=${HADOOP_HOME}</value>
</property>
<property>
  <name>mapreduce.reduce.env</name>
  <value>HADOOP_MAPRED_HOME=${HADOOP_HOME}</value>
</property>
```