目录

1. 试验一：优化 dfs.replication 文件副本数

1.1. 实验目的

完成本实验，您应该能够：

- 掌握 HDFS 集群调优策略之 dfs.replication
- 理解 dfs.replication 配置如何影响文件上传过程中的时间
- 掌握 dfs.replication 参数的设置

1.2. 实验要求

- 熟悉 Linux 命令
- 理解 HDFS 的原理
- 熟悉 HDFS 文件操作
- 熟悉 HDFS 集群参数配置

1.3. 实验环境

本实验所需之主要资源环境如表 1-1 所示。

表 1-1 资源环境

服务器集群	3 个节点，节点间网络互通，各节点配置：2 核 CPU、2GB 内存、30G 硬盘
运行环境	CentOS 7.4 （gui 英文版本）
用户名/密码	root/password hadoop/password
服务和组件	HDFS、YARN、MapReduce 等，其他服务根据实验需求安装
测试数据大小	115.3MB、631.39 MB、1.23G

1.4. 实验过程

1.4.1. 实验任务一：HDFS 文件副本数设置为 3

1) 配置参数

```
[hadoop@master ~]$ cd /usr/local/src/hadoop
[hadoop@master hadoop]$ vi ./etc/hadoop/hdfs-site.xml
        <property>
                <name>dfs.replication</name>
                <value>3</value>
```

</property>

2）上传文件，并统计所需时间

```
[hadoop@master hadoop]$ hdfs dfs -mkdir /input1
[hadoop@master hadoop]$ time hadoop fs -put /opt/software/115.txt /input1
real    0m16.761s
user    0m3.971s
sys     0m0.376s
[hadoop@master hadoop]$ time hadoop fs -put /opt/software/631.txt /input1
real    1m30.733s
user    0m6.350s
sys     0m1.247s
[hadoop@master hadoop]$ time hadoop fs -put /opt/software/1230.txt /input1
real    2m21.532s
user    0m7.267s
sys     0m1.880s
```

此处需要说明的是，通过 time 命令量测出来该指令执行时所消耗的时间会因为具体的集群性能不同可能会存在差异，但是在相同的条件基础上得到的数据还是有参考价值的。

通过浏览器可以查看到上传之后的文件会有 3 个副本，分别存放在 master、slave1、slave2 上。

查看 master 和 slave1 名称节点的状态,若 master 节点状态为 active,则输入网址 **master:50070(本次启动网址)** 点击 utilities 查看各个文件详情,否则网址为 slave1:50070,

[hadoop@master mapreduce]$ hdfs haadmin -getServiceState master

active

[hadoop@master mapreduce]$ hdfs haadmin -getServiceState master

standby

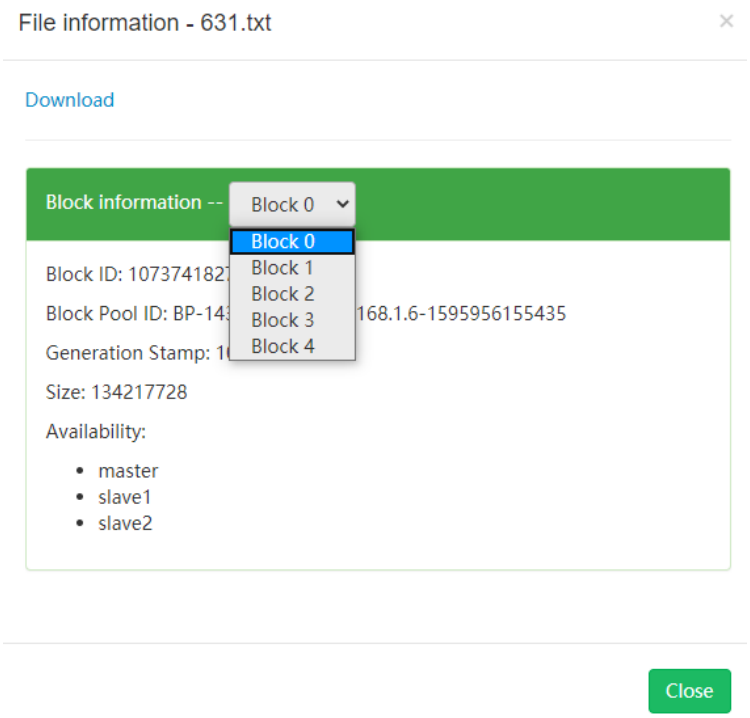


图 1-2 副本数为 3 时 HDFS 中 631.txt 文件信息

若文件只有 2 个副本,则检查 datanode 启动数量,命令为 ”hadoop fsck –locations” 查看 Number of data-nodes 是否为 3,不是 3 则启动各个主机的 datanode

3）运行 mapreduce 官方实例：

[hadoop@master mapreduce]$ cd /usr/local/src/hadoop

[hadoop@master hadoop]$ hadoop jar /usr/local/src/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.1.jar wordcount /input1 /output1

网址**:master:8088** 点击 history 跳转,查看 mapreduce 详情

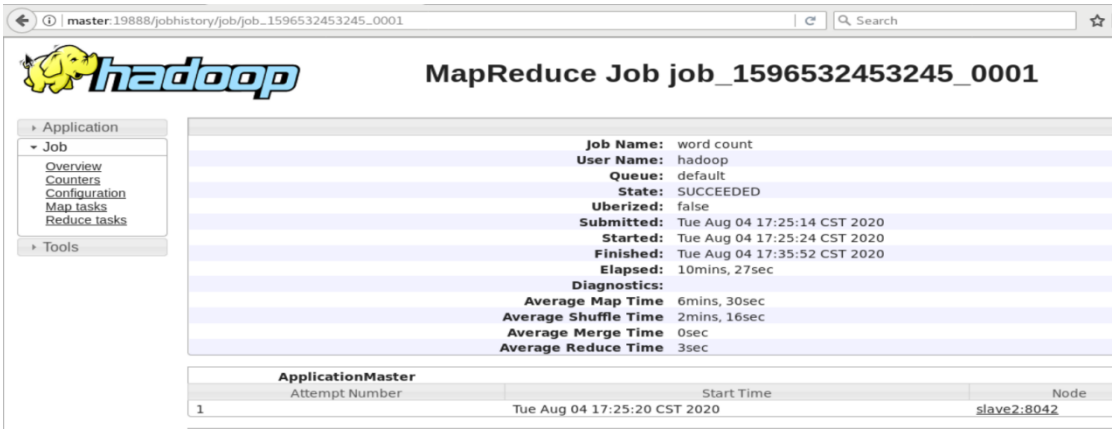


图 1-3 副本数为 3 时对 631.txt 数据执行 MR 操作

4）运行结果：

Elapsed: 10mins, 27sec
Average Map Time 6mins, 30sec
Average Shuffle Time 2mins,16sec
Average Merge Time 0sec
Average Reduce Time 3sec

1.4.2. 实验任务二：HDFS 文件副本数设置为 2

1) 配置参数

[hadoop@master hadoop]$ vi /usr/local/src/hadoop/etc/hadoop/hdfs-site.xml

```
<property>
        <name>dfs.replication</name>
        <value>2</value>
</property>
```

2）文件副本数也可以在上传文件的时候设置

```
[hadoop@master hadoop]# hdfs dfs -mkdir /input2
```

```
[hadoop@master hadoop]$ time hadoop fs -D dfs.replication=2 -put /opt/software/115.txt
/input2
real    0m14.076s
user    0m3.914s
sys     0m0.369s
[hadoop@master hadoop]$ time hadoop fs -D dfs.replication=2 -put /opt/software/631.txt
/input2
real    1m7.316s
user    0m6.300s
sys     0m1.334s
[hadoop@master hadoop]$ time hadoop fs -D dfs.replication=2 -put /opt/software/1230.txt
/input2
real    2m11.959s
user    0m7.962s
sys     0m1.952s
```

通过浏览器可以查看到上传之后的文件只有 2 个副本，分别存放在 slave1、master 上。

网址　**master:50070**　点击 utilities 查看各个文件详情

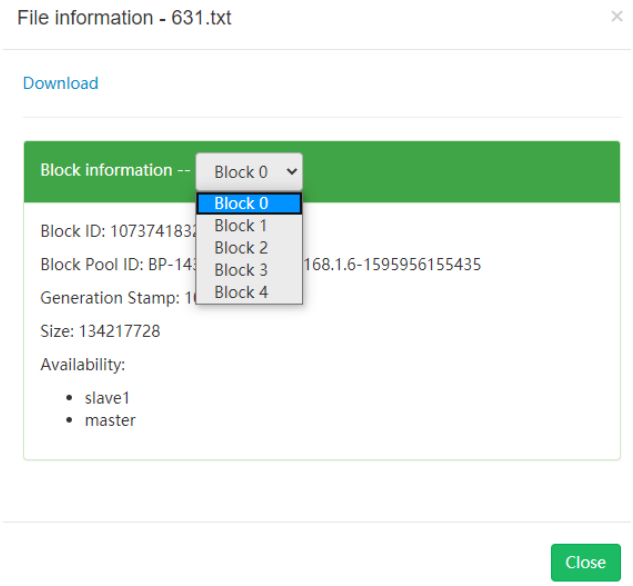


图 1-4 副本数为 2 时 HDFS 中 631.txt 文件信息

3）运行 mapreduce 官方实例：

[hadoop@master hadoop]$ hadoop jar /usr/local/src/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.1.jar wordcount /input2 /output2

网址**:master:8088**　点击 history 跳转,查看 mapreduce 详情

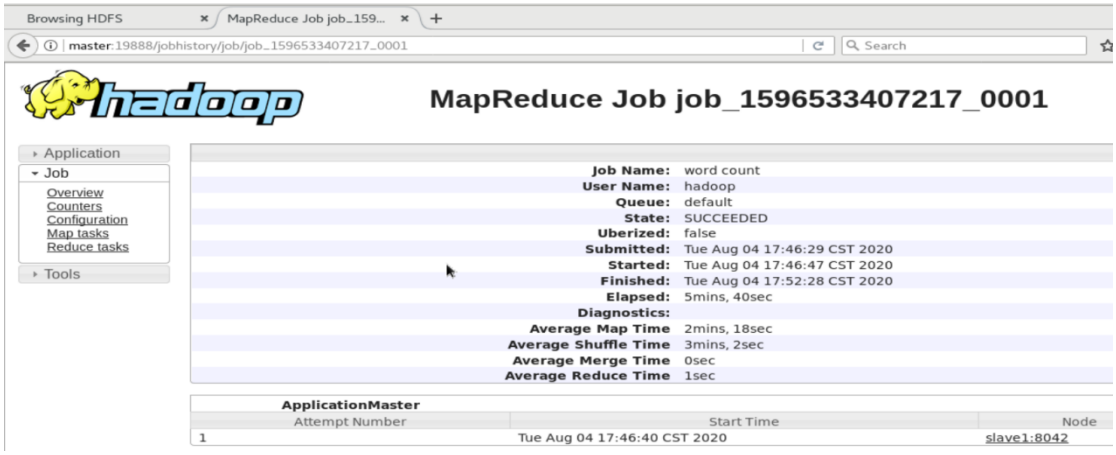


图 1-5 副本数为 2 时对 631.txt 数据执行 MR 操作

4）运行结果

Elapsed: 5mins, 40sec

Average Map Time 2mins, 18sec

Average Shuffle Time 3mins,2sec

Average Merge Time 0sec

Average Reduce Time 1sec

1.4.3．实验分析总结

通过多次试验对比，测试数据大小 631.39MB，在本次试验环境下测试，HDFS 文件副本数设置为 3 时作业花费的时间均比设置为 2 时的要多。

Hadoop 的备份系数是指每个 block 在 hadoop 集群中有几份，系数越高，冗余性越好，占用存储也越多。如果只有 3 个 datanode，但却指定副本数为 4，是不会生效的，因为每个 datanode 上只能存放一个副本，所以这里试验通过设置副本数为 2 和 3 进行测试。

此外，HDFS 采用一种称为机架感知的策略来改进数据的可靠性、可用性和网络带宽的利用率。这种策略在不损坏可靠性和读取性能的情况下，改善了写的性能。在大多数情况下，HDFS 的副本系数是 3，HDFS 的存放策略是一个副本存放在本地机架节点上，另一个副本存放在同一机架的另一个节点上，第三个副本存放在在不同机架的节点上。这种策略减少了机架间的数据传输，提高了写操作的效率。机架错误的概率远比节点错误的概率小，所以这种策略不会对数据的可靠性和可用性造成影响。与此同时，因为数据只存在两个机架上，这种策略减少了读数据时需要的网络传输带宽。在这种策略下，副本并不是均匀地分布在机架上。当没有配置机架信息时，全部节点 hadoop 都默认在同一个默认的机架下，无论物理上是否属于同一个机架，都会被认为是在同一个机架下。

2. 试验二：设置 dfs.block.size 数据块大小

2.1. 实验目的

完成本实验，您应该能够：

- 掌握 HDFS 集群调优策略之 dfs.block.size
- 理解 dfs.block.size 配置如何影响文件上传过程中的时间，如何配置合适
- 掌握 dfs.block.size 参数的设置

2.2. 实验要求

- 熟悉 Linux 命令
- 理解 HDFS 的原理
- 熟悉 HDFS 文件操作
- 熟悉 HDFS 集群参数配置

2.3. 实验环境

本实验所需之主要资源环境如表 2-1 所示。

表 2-1 资源环境

服务器集群	3 个节点，节点间网络互通，各节点配置：4 核 CPU、2GB 内存、20G 硬盘
运行环境	CentOS 7.4 （gui 英文版本）
用户名/密码	root/password hadoop/password
服务和组件	HDFS、YARN、MapReduce 等，其他服务根据实验需求安装
测试数据大小	631.39 MB

2.4. 实验过程

2.4.1. 实验任务一：设置 dfs.block.size 为 128M

1) 配置参数

删除原 hdfs 上/input1 和/input2 /output1 /output2 文件

[hadoop@master hadoop]$ hdfs dfs -rm -r -f /input1 /input2 /output1 /output2

新建 input1 和 input2 文件

[hadoop@master hadoop]$ hdfs dfs -mkdir /input1 /input2

[hadoop@master hadoop]$ vi /usr/local/src/hadoop/etc/hadoop/hdfs-site.xml

```
<property>
    <name>dfs.block.size</name>
    <value>134217728</value>
</property>
<property>
    <name>dfs.replication</name>
    <value>3</value>
</property>
```

2）上传数据到 HDFS

[hadoop@master hadoop]$ hadoop fs -put /opt/software/631.txt /input1

网址　**master:50070**　点击 utilities 查看文件详情

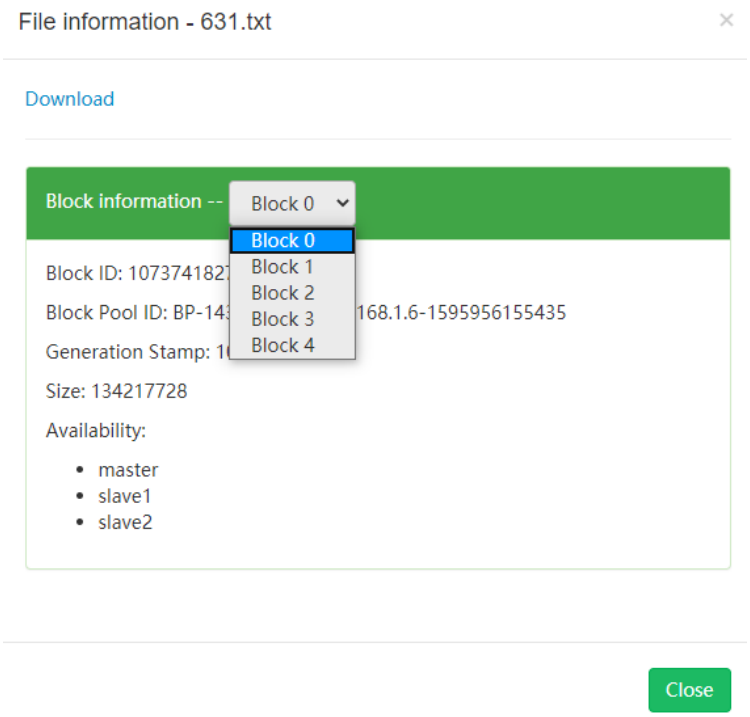


图 2-2 数据块大小为 128M 时 HDFS 中 631.txt 文件信息

3）运行官方 MapReduce 实例查看运行结果

[hadoop@master hadoop]$ hadoop jar /usr/local/src/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.1.jar wordcount /input1 /output1

运行结果如下：

```
Job Counters
        Launched map tasks=5
        Launched reduce tasks=1
```

Data-local map tasks=5
Total time spent by all maps in occupied slots (ms)=1227254
Total time spent by all reduces in occupied slots (ms)=8775
Total time spent by all map tasks (ms)=1227254
Total time spent by all reduce tasks (ms)=8775
Total vcore-seconds taken by all map tasks=1227254
Total vcore-seconds taken by all reduce tasks=8775
Total megabyte-seconds taken by all map tasks=1256708096
Total megabyte-seconds taken by all reduce tasks=8985600

网址**:master:8088** 点击 history 跳转,查看 mapreduce 详情

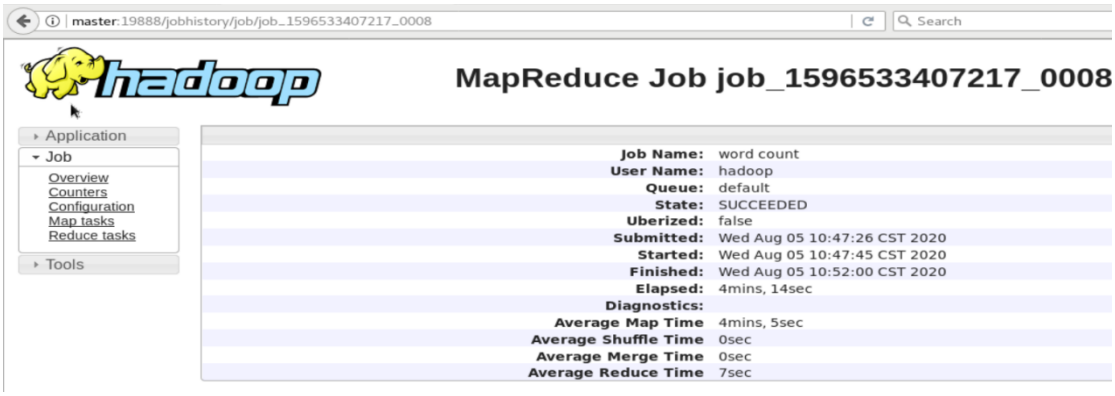


图 2-3 数据块大小为 128M 时对 631.txt 数据执行 MR 操作

结果显示：测试数据 631.39MB，块大小 128MB，map 阶段任务数为 5，map 阶段平均时间：4mins, 5sec。

2.4.2．实验任务二：设置 dfs.block.size 为 256M

1）配置参数

```
[hadoop@master hadoop]$ vi /usr/local/src/hadoop/etc/hadoop/hdfs-site.xml
    <property>
        <name>dfs.block.size</name>
        <value> 268435456</value>
    </property>
```

2）上传数据到 HDFS

```
[hadoop@master hadoop]$ hadoop fs -put /opt/software/631.txt /input2
```

网址　**master:50070** 点击 utilities 查看文件详情

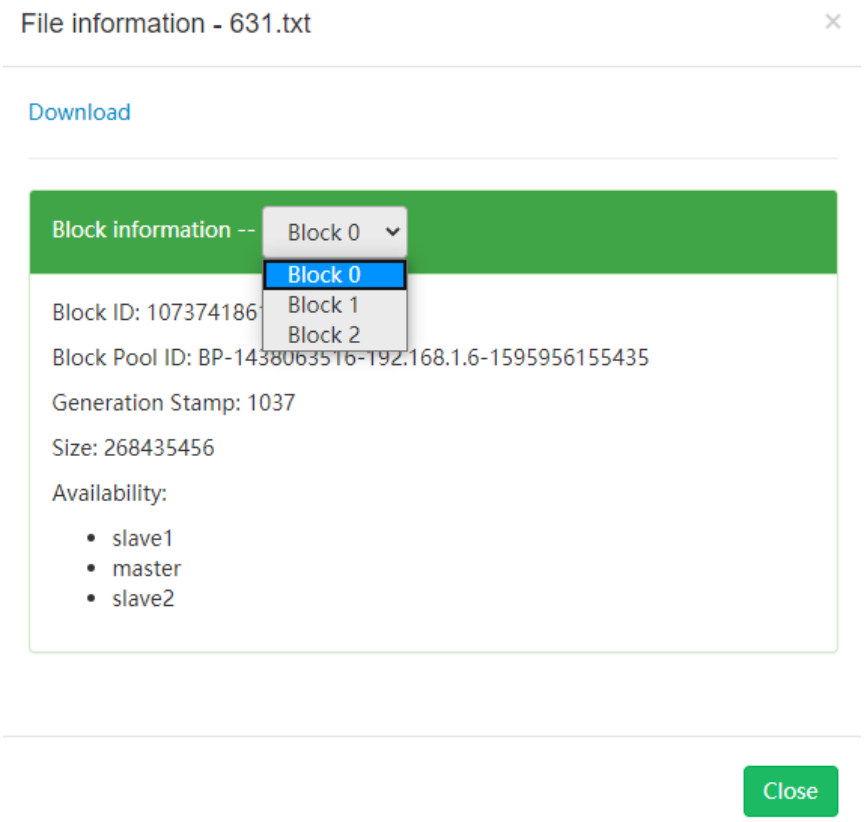


图 2-4 数据块大小为 256M 时 HDFS 中 631.txt 文件信息

3）运行官方 MapReduce 实例查看运行结果

[hadoop@master hadoop]$ hadoop jar /usr/local/src/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.1.jar wordcount /input2 /output2

运行结果如下：

Job Counters
Killed map tasks=1
Launched map tasks=4
Launched reduce tasks=1
Data-local map tasks=4
Total time spent by all maps in occupied slots (ms)=279168
Total time spent by all reduces in occupied slots (ms)=29240
Total time spent by all map tasks (ms)=279168
Total time spent by all reduce tasks (ms)=29240
Total vcore-seconds taken by all map tasks=279168
Total vcore-seconds taken by all reduce tasks=29240
Total megabyte-seconds taken by all map tasks=285868032
Total megabyte-seconds taken by all reduce tasks=29941760

网址**:master:8088** 点击 history 跳转,查看 mapreduce 详情

图 2-5 数据块大小为 256MB 时对 631.txt 数据执行 MR 操作

结果显示：测试数据 631.39MB，块大小 256MB，map 阶段任务数为 3，map 阶段平均时间：1mins, 23sec。

2.4.3．实验分析总结

通过两个实验对比，两次任务数据均为 631.39MB，block 设为 128MB 时，测试文件被切分为 5 个 block，产生 map 任务数为 5；block 设为 256MB 时，测试文件被切分为 3 个 block，产生 map 任务数为 4，这里因为配置限制，无法使用更大的数据进行测试，如果测试数据过大，产生的 map 任务数就会越多，map 任务个数太多会影响处理效率，如果数据过大，需要将 block size 设置更大些。

通过两个实验运行效率对比，block 为 128MB 时，map 阶段平均时间：4mins, 5sec，block 为 256MB 时，map 阶段平均时间：1mins, 23sec，对比发现，设置过大的 block 时，运行效率并没有得到有效提升，因为从磁盘传输数据的时间会明显大于寻址时间，导致程序在处理这块数据时，变得非常慢。

HDFS 的 blocksize 需要根据实际业务数据的大小进行调整，过大过小都不合适。因为文件的读取速度包含：寻址时间（HDFS 中找到目标文件 block 块所花费的时间）和传输时间。但是文件块越大，寻址时间越短，但磁盘传输时间越长；而文件块越小，寻址时间越长，但磁盘传输时间越短。

blocksize 块大小设置规则：

1）HDFS 中平均寻址时间大概为 10ms；

2）经过前任的大量测试发现，寻址时间为传输时间的 1%时，为最佳状态，所以最佳传输时间为：

10ms/0.01=1000s=1s

3）目前磁盘的传输速度普遍为 100MB/s，最佳 block 大小计算：

100MB/s*1s=100MB

所以设置 block 大小为 128MB。

4）在实际中，磁盘传输速率为 200MB/s 时，一般设定 block 大小为 256MB；磁盘传输速率为 400MB/s 时，一般设定 block 大小为 512MB。

3. 试验三：设置 dfs.datanode.data.dir 磁盘目录

3.1. 实验目的

完成本实验，您应该能够：

- 掌握 HDFS 集群调优策略之 dfs.datanode.data.dir
- 理解磁盘目录的作用
- 掌握 dfs.datanode.data.dir 参数的设置

3.2. 实验要求

- 熟悉 Linux 命令
- 理解 HDFS 的原理
- 熟悉 HDFS 文件操作
- 熟悉 HDFS 集群参数配置

3.3. 实验环境

本实验所需之主要资源环境如表 3-1 所示。

表 3-1 资源环境

服务器集群	3 个节点，节点间网络互通，各节点配置：4 核 CPU、2GB 内存、20G 硬盘
运行环境	CentOS 7.4 （gui 英文版本）
大数据平台	root/password hadoop/password
服务和组件	HDFS、YARN、MapReduce 等，其他服务根据实验需求安装
测试数据大小	631.39 MB

3.4．实验过程

3.4.1．设置两个 data 目录

hadoop 的 dfs.datanode.data.dir 是设置 datanode 节点存储数据块文件的本地路径，通常可以设置多个，用逗号隔开：

删除原 hdfs 上/input1 和/input2 /output1 /output2 文件

[hadoop@master hadoop]$ hdfs dfs -rm -r -f /input1 /input2 /output1 /output2

新建 input1 和 input2 文件

[hadoop@master hadoop]$ hdfs dfs -mkdir /input1 /input2

[hadoop@master hadoop]$ vi /usr/local/src/hadoop/etc/hadoop/hdfs-site.xml

```
<property>
    <name>dfs.datanode.data.dir</name>
    <value>/usr/local/src/hadoop/tmp/hdfs/dn,/home/hadoop/data/dfs/data </value>
</property>
    <property>
        <name>dfs.block.size</name>
        <value>134217728</value>
    </property>
```

这里设置两个目录，分别为：

hadoop 安装目录下：/usr/local/src/hadoop/tmp/hdfs/dn

home 目录下：/home/hadoop/data/dfs/data

由于 home 目录下不存在上面所示文件夹，因此需要在 home 用户目录下创建文件夹：

[hadoop@master hadoop]$ mkdir -p /home/hadoop/data/dfs/data

3.4.2．通过上传文件查看数据块存储情况

1）上传文件

[hadoop@master hadoop]$ hadoop fs -put /opt/software/631.txt /input1

上传的文件大小为 631.39MB。

重启 hadoop

[hadoop@master hadoop]$ stop-all.sh

[hadoop@master hadoop]$ start-all.sh

网址　**master:50070** 点击 utilities 查看文件详情

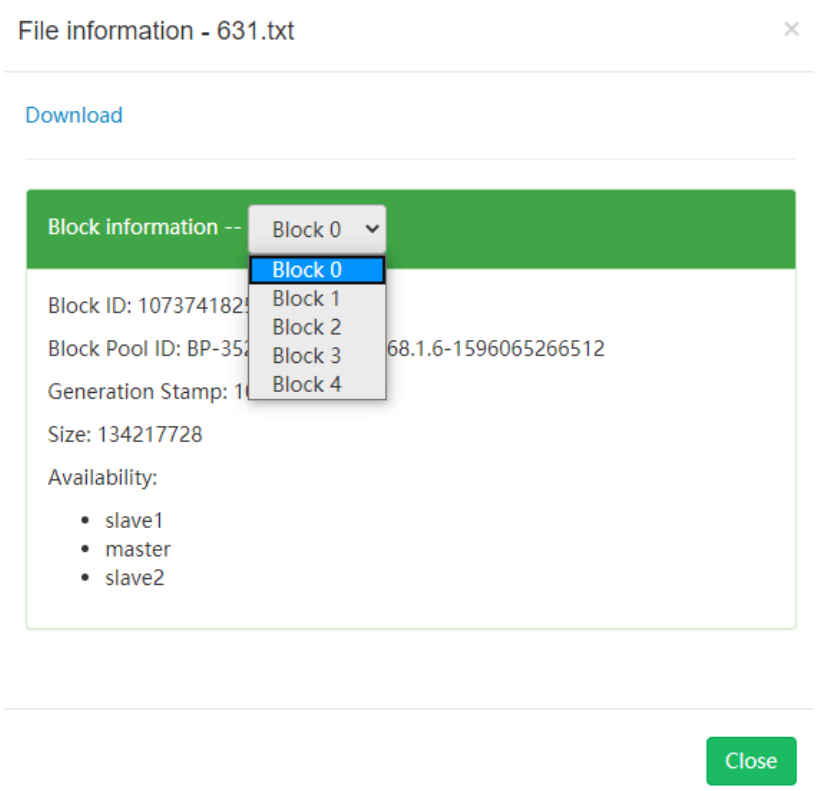


图 3-2 HDFS 中的 631.txt 文件

2）查看数据块存储情况

从图 3-2 可以看出，631.39MB 的文件产生 5 个 block，对于每个 block 的详细信息可以执行下面的命令查看：

```
[hadoop@master hadoop]$ hdfs fsck /input1/631.txt -files -blocks -locations
Connecting to namenode via http://master:50070/fsck?ugi=hadoop&files=1&blocks=1&locations=1&path=%2Finput1%2F631.txt
FSCK started by hadoop (auth:SIMPLE) from /192.168.90.39 for path /input1/631.txt at Wed Aug 05 11:25:26 CST 2020
/input1/631.txt 662064288 bytes, 5 block(s):  Under replicated BP-1893501819-192.168.90.39-1596532166179:blk_1073741967_1143. Target Replicas is 3 but found 2 replica(s).
 Under replicated BP-1893501819-192.168.90.39-1596532166179:blk_1073741968_1144. Target Replicas is 3 but found 2 replica(s).
 Under replicated BP-1893501819-192.168.90.39-1596532166179:blk_1073741969_1145. Target Replicas is 3 but found 2 replica(s).
 Under replicated BP-1893501819-192.168.90.39-1596532166179:blk_1073741970_1146. Target Replicas is 3 but found 2 replica(s).
 Under replicated BP-1893501819-192.168.90.39-1596532166179:blk_1073741971_1147. Target Replicas is 3 but found 2 replica(s).
```

0. BP-1893501819-192.168.90.39-1596532166179:blk_1073741967_1143 len=134217728 repl=2 [DatanodeInfoWithStorage[192.168.90.39:50010,DS-83568ddc-3e04-4461-a751-0bc887447457,DISK], DatanodeInfoWithStorage[192.168.90.135:50010,DS-ad64b7d3-0f1d-4ca4-b52b-f9ab0c91df81,DISK]]
1. BP-1893501819-192.168.90.39-1596532166179:blk_1073741968_1144 len=134217728 repl=2 [DatanodeInfoWithStorage[192.168.90.39:50010,DS-83568ddc-3e04-4461-a751-0bc887447457,DISK], DatanodeInfoWithStorage[192.168.90.135:50010,DS-ad64b7d3-0f1d-4ca4-b52b-f9ab0c91df81,DISK]]
2. BP-1893501819-192.168.90.39-1596532166179:blk_1073741969_1145 len=134217728 repl=2 [DatanodeInfoWithStorage[192.168.90.39:50010,DS-83568ddc-3e04-4461-a751-0bc887447457,DISK], DatanodeInfoWithStorage[192.168.90.135:50010,DS-ad64b7d3-0f1d-4ca4-b52b-f9ab0c91df81,DISK]]
3. BP-1893501819-192.168.90.39-1596532166179:blk_1073741970_1146 len=134217728 repl=2 [DatanodeInfoWithStorage[192.168.90.39:50010,DS-83568ddc-3e04-4461-a751-0bc887447457,DISK], DatanodeInfoWithStorage[192.168.90.135:50010,DS-ad64b7d3-0f1d-4ca4-b52b-f9ab0c91df81,DISK]]
4. BP-1893501819-192.168.90.39-1596532166179:blk_1073741971_1147 len=125193376 repl=2 [DatanodeInfoWithStorage[192.168.90.39:50010,DS-83568ddc-3e04-4461-a751-0bc887447457,DISK], DatanodeInfoWithStorage[192.168.90.135:50010,DS-ad64b7d3-0f1d-4ca4-b52b-f9ab0c91df81,DISK]]

Status: HEALTHY
 Total size:　662064288 B
 Total dirs:　0
 Total files:　1
 Total symlinks:　0
 Total blocks (validated):　5 (avg. block size 132412857 B)
 Minimally replicated blocks:　5 (100.0 %)
 Over-replicated blocks:　0 (0.0 %)
 Under-replicated blocks:　5 (100.0 %)
 Mis-replicated blocks:　0 (0.0 %)
 Default replication factor:　3
 Average block replication:　2.0
 Corrupt blocks:　0
 Missing replicas:　5 (33.333332 %)
 Number of data-nodes:　3
 Number of racks:　1
FSCK ended at Wed Aug 05 11:25:26 CST 2020 in 9 milliseconds

The filesystem under path '/input1/631.txt' is HEALTHY

3）查看数据块在两个 data 目录中查看存储情况（数据块分布随机）

hadoop 安装目录下

[hadoop@master ~]$ cd /usr/local/src/hadoop/tmp/hdfs/dn

[hadoop@master data]# ll

总用量 4

drwxr-xr-x. 3 hadoop hadoop 70 7 月　20 13:53 current

-rw-r--r--. 1 hadoop hadoop 11 7 月　30 10:14 in_use.lock

[hadoop@master data]# cd current/

[hadoop@master current]# ll

总用量 4

drwx------. 4 hadoop hadoop　54 7 月　30 17:22 BP-352475817-192.168.1.6-1596065266512

-rw-r--r--. 1 hadoop hadoop 229 7 月　30 10:14 VERSION

可以看出 hadoop 安装目录下的 data 文件夹中存放了 Block Pool ID: BP-352475817-192.168.1.6-1596065266512 的数据块。

[hadoop@master current]$ cd /usr/local/src/hadoop/data/dfs/data/current/BP-352475817-192.168.1.6-1596065266512/current/finalized/subdir0/subdir0/

[hadoop@master subdir0]# ll

总用量 387416

-rw-rw-r-- 1 hadoop hadoop　49 7 月　21 15:52 blk_1073742006

-rw-rw-r-- 1 hadoop hadoop　11 7 月　21 15:52 blk_1073742006_1182.meta

-rw-rw-r-- 1 hadoop hadoop 359 7 月　21 16:01 blk_1073742014

-rw-rw-r-- 1 hadoop hadoop　11 7 月　21 16:01 blk_1073742014_1190.meta

可以看到，该目录下存储的 block 有：blk_1073742006、blk_1073742014 这两个数据块。

home 目录下

[hadoop@master ~]$ cd /home/hadoop/data/dfs/data/

[hadoop@master data]# ll

总用量 4

drwxr-xr-x. 3 hadoop hadoop 7 7 月　31 11:39 current

-rw-r--r--. 1 hadoop hadoop 12 7 月　31 11:39 in_use.lock

[hadoop@master data]$ cd current/

[hadoop@master current]$ ll

总用量 4

drwx------. 4 hadoop hadoop　54 7 月　31 11:39 BP-352475817-192.168.1.6-1596065266512

-rw-r--r--. 1 hadoop hadoop 229 7 月　31 11:39 VERSION

可以看出 home 目录下的 data 文件夹中存放了 Block Pool ID: BP-352475817-192.168.1.6-1596065266512 的数据块。

[hadoop@master current]$ cd /home/hadoop/data/dfs/data/current/BP-352475817-192.168.1.6-1596065266512/current/finalized/subdir0/subdir31

[hadoop@master subdir31]$ ll

总用量 264200

-rw-rw-r-- 1 hadoop hadoop 134217728 7 月　31 11:39 blk_1073749934

-rw-rw-r-- 1 hadoop hadoop　1048583 7 月　31 11:39 blk_1073749934_9110.meta

-rw-rw-r-- 1 hadoop hadoop 134217728 7 月　31 11:39 blk_1073749936

-rw-rw-r-- 1 hadoop hadoop　1048583 7 月　31 11:39 blk_1073749936_9112.meta 可以看到，该目录下存储的 block 有：blk_1073749934、blk_1073749936 这两个数据块。

3.4.3．实验分析总结

dfs.datanode.data.dir 是 HDFS 数据存储目录，是设置 datanode 节点存储数据块文件的本地路径，可以设置多个，用逗号隔开：

```
<property>
    <name>dfs.datanode.data.dir</name>
    <value>/home/hadoop/project/hadoop/data/dfs/data,/home/hadoop/data/dfs/data</value>
</property>
```

将数据存储分布在各个磁盘上可充分利用节点的 I/O 读写性能。因此在实际生产环境中，这也是磁盘不选择 RAID 和 LVM，而选择 JBOD 的原因。推荐设置多个磁盘目录，以增加磁盘 IO 的性能，提高并发存取的速度，多个目录用逗号进行分隔。但是设置多个路径时，随着数据量的增多，有可能会导致磁盘空间不均衡，因为 hadoop 默认是轮询方式写入，如上实验过程所示，如果产生的 block 数一直为奇数，目录（磁盘）1 的空间占用率会比目录（磁盘）2 的要多得多；因此也可以配置 hadoop 的另一种写入策略：根据可用空间的大小来判断写入。

```
<property>
    <name>dfs.datanode.fsdataset.volume.choosing.policy</name>
    <value>org.apache.hadoop.hdfs.server.datanode.fsdataset.AvailableSpaceVolumeChoosingPolicy</value>
</property>
```

此项配置是根据磁盘的可用空间来优先写入的策略，一般需要配合以下两个参数来使用：

dfs.datanode.available-space-volume-choosing-policy.balanced-space-threshold

默认值是 10737418240，既 10G；意思是首先计算出两个值，一个是所有磁盘中最大可用空间，另外一个值是所有磁盘中最小可用空间，如果这两个值相差小于该配置项指定的阀值时，则就用轮询方式的磁盘选择策略选择磁盘存储数据副本。

dfs.datanode.available-space-volume-choosing-policy.balanced-space-preference-fraction

默认值是 0.75f；意思是有多少比例的数据副本应该存储到剩余空间足够多的磁盘上。该配置项取值范围是 0.0-1.0，一般取 0.5-1.0，如果配置太小，会导致剩余空间足够的磁盘实际上没分配足够的数据副本，而剩余空间不足的磁盘取需要存储更多的数据副本，导致磁盘数据存储不均衡。

另外，不同的 datanode 中，block 的目录属性可以设置为不相同，不影响集群的正常运行。如果当前目录（磁盘）被占满，可以再次挂载新的目录（磁盘），继续添加 dfs.datanode.data.dir 的值。

4. 试验四：设置 mapred.local.dir 优化 IO 读写能力

4.1. 实验目的

完成本实验，您应该能够：

- 掌握 HDFS 集群调优策略之 mapred.local.dir
- 理解 mapred.local.dir 的作用
- 理解 mapreduce 运行过程

4.2. 实验要求

- 熟悉 Linux 命令
- 理解 HDFS 的原理
- 熟悉 HDFS 文件操作
- 熟悉 HDFS 集群参数配置

4.3. 实验环境

本实验所需之主要资源环境如表 4-1 所示。

表 4-1 资源环境

服务器集群	3 个节点，节点间网络互通，各节点配置：4 核 CPU、2GB 内存、20G 硬盘
运行环境	CentOS 7.4 （gui 英文版本）
用户名/密码	root/password hadoop/password
服务和组件	HDFS、YARN、MapReduce 等，其他服务根据实验需求安装
测试数据大小	631.39 MB

4.4. 实验过程

4.4.1. 设置一个临时缓存目录

1）配置参数

[hadoop@master hadoop]$ vi /usr/local/src/hadoop/etc/hadoop/mapred-site.xml

```
<property>
    <name>mapred.local.dir</name>
    <value>/home/hadoop/data/mapred/local/</value>
</property>
```

2）创建目录

[hadoop@master hadoop]$ mkdir -p /home/hadoop/data/mapred/local/

[hadoop@master hadoop]$ cd /home/hadoop/data/mapred/local/

[hadoop@master local]$ pwd

/home/hadoop/data/mapred/local

重启 hadoop

[hadoop@master local]$ stop-all.sh

[hadoop@master local]$ start-all.sh

删除 output1 output2

[hadoop@master local]$ hdfs dfs -rm -r -f /output1 /output2

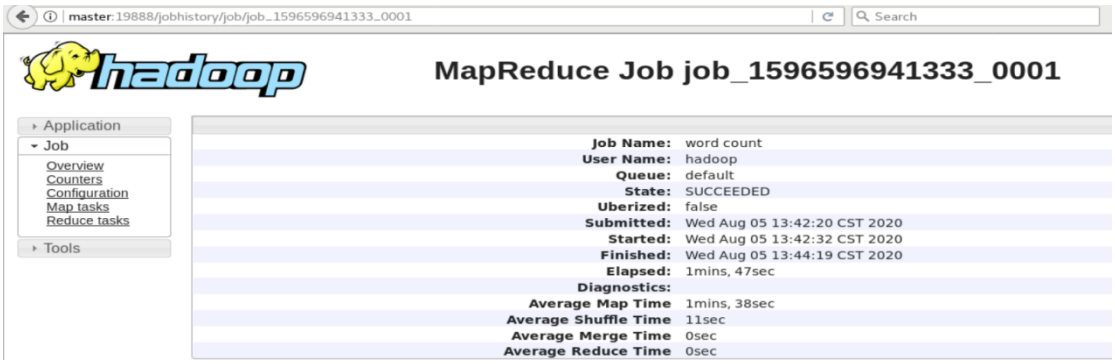
3）运行测试数据

[hadoop@master local]$ cd

[hadoop@master ~]$ hadoop jar /usr/local/src/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.1.jar wordcount /input1/631.txt　/output1

```
……
        Job Counters
                Launched map tasks=5
                Launched reduce tasks=1
                Data-local map tasks=5
                Total time spent by all maps in occupied slots (ms)=491989
                Total time spent by all reduces in occupied slots (ms)=12089
                Total time spent by all map tasks (ms)=491989
                Total time spent by all reduce tasks (ms)=12089
                Total vcore-seconds taken by all map tasks=491989
                Total vcore-seconds taken by all reduce tasks=12089
                Total megabyte-seconds taken by all map tasks=503796736
                Total megabyte-seconds taken by all reduce tasks=12379136
……
```

网址**:master:8088** 点击 history 跳转,查看 mapreduce 详情



结果：

Elapsed: 1mins, 47sec

Average Map Time 1mins, 38sec

Average Shuffle Time 11sec

Average Merge Time 0sec

Average Reduce Time 0sec

4.4.2．设置两个临时缓存目录

1）配置参数

[hadoop@master ~]# vi /usr/local/src/hadoop/etc/hadoop/mapred-site.xml

```
<property>
    <name>mapred.local.dir</name>
    <value>/home/hadoop/data/mapred/local/,/usr/local/src/hadoop/data/mapred/local/</value>
</property>
```

2）创建目录

[hadoop@master ~]# mkdir -p /usr/local/src/hadoop/data/mapred/local/

[hadoop@master ~]# cd /usr/local/src/hadoop/data/mapred/local/

[hadoop@master local]# pwd

/usr/local/src/hadoop/data/mapred/local

重启 hadoop

[hadoop@master local]# stop-all.sh

[hadoop@master local]# start-all.sh

3）运行测试数据

[hadoop@master local]# hadoop jar /usr/local/src/hadoop/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.7.1.jar wordcount /input1/631.txt /output2

……

Job Counters

Launched map tasks=5

Launched reduce tasks=1

Data-local map tasks=5

Total time spent by all maps in occupied slots (ms)=494114
Total time spent by all reduces in occupied slots (ms)=9739
Total time spent by all map tasks (ms)=494114
Total time spent by all reduce tasks (ms)=9739
Total vcore-seconds taken by all map tasks=494114
Total vcore-seconds taken by all reduce tasks=9739
Total megabyte-seconds taken by all map tasks=505972736
Total megabyte-seconds taken by all reduce tasks=9972736

……

网址**:master:8088** 点击 history 跳转,查看 mapreduce 详情

结果：
Elapsed: 1mins, 47sec
Average Map Time 1mins, 38sec
Average Shuffle Time 7ec
Average Merge Time 0sec
Average Reduce Time 1sec

4.4.3．实验分析总结

mapred.local.dir 是在 mapreduce 运行流程 MapTask 阶段临时数据存放的地方（Spill 阶段：即“溢写”，当环形缓冲区满后，MapReduce 会将数据写到本地磁盘上，生成一个临时文件），，也就是说 map 本地计算时所用到的目录，可以配置多个目录路径，因为多路径有助于利用磁盘 I/O，优化作业运行效率，建议配置在多块硬盘上。

通过以上两个实验，设置 mapred.local.dir 目录为一个目录时，任务总耗时 1mins, 47sec，map 耗时：1mins, 38sec ；设置为两个目录时，任务总耗时 1mins, 47sec，map 耗时：1mins, 38sec，由于集群配置和测试数据限制，两个实验效果差距不是特别明显，但是还是可以印证 mapred.local.dir 所配置的目录一定数量的增加还是能提高 IO 读写能力的。