MapReduce Using Single Node HDFS

Data Intensive Computing 587 Project 2 Report

Magizharasu Thirunavukkarasu
Person # 50026983

SUMMARY:

The project consists of a single node HDFS on which the MapReduce Programs are run to find word count and relative count of co-occurring words by pairs, stripes method.

The input is a large text collection from the Gutenberg free ebook source. The text files are moved on to the HDFS system and then MapReduce programs are run to get the output required. The word count and relative frequencies of the co-occurrences of words are computed.

PROJECT OBJECTIVES:

This project meets the following objectives

- Set up a single node HDFS using VMPlayer
- Move input data onto the HDFS system
- Run MapReduce programs to compute word-count with custom Pratitioner and set number of Reduce tasks as 2 .
- Run MapReduce Stripes and Pairs program to compute relative count of co-ccurrence of words. The programs should have a Custom Pratitioner and 2 Reducers.
- The input size is varied and a comparison chart is drawn based on the runtime for various inputs.

PROJECT IMPLEMENTATION

Word Count:

Here a custom Mapper,Reducer and Partitioner are written and the number of reduce tasks is set to 2 using “setNumReduceTasks(). The Custom Partitioner class is written in order to divide the words into respective reducers based on the starting letter. The mapper emits and word,1 and reducer sums the values of the same key to give the final count of each word.

Pairs:

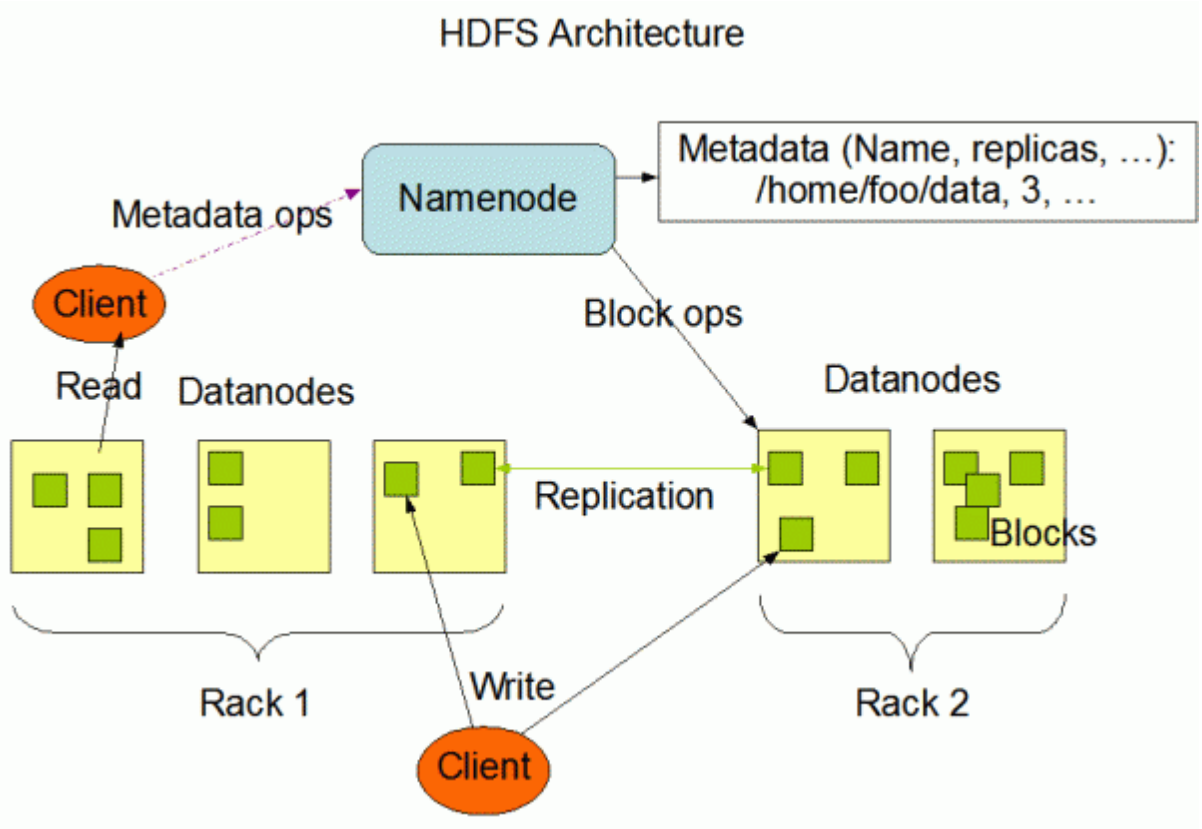
It is similar to word count, but here we have the key as a pair (word1,word2). Here a custom data-type is defined using the TextPair class. The mapper emits each word-pair and count as 1. Along with each pair (word1,”*”) is emitted to aid in relative count computation. In the reducer the count of each unique pair is summed up and divided by the (word1,”*”) count. Thus the word-pair and relative count is emitted out from the reducer.

Stripes:

Here a hash map data structure is used to store the word and its other co-occurring words in the current input. The count of each co-occurring word is updated as pairs are encountered. After the whole of input is processed the mapper emits the map.

The reducer receives the map and first computes the total count of co-occurring words from all the mappers and then the relative count is obtained by dividing the count of each co-occurring word by sum of all the counts of the co-occurring words. And Stripes are emitted out of the reducer.

HDFS System :



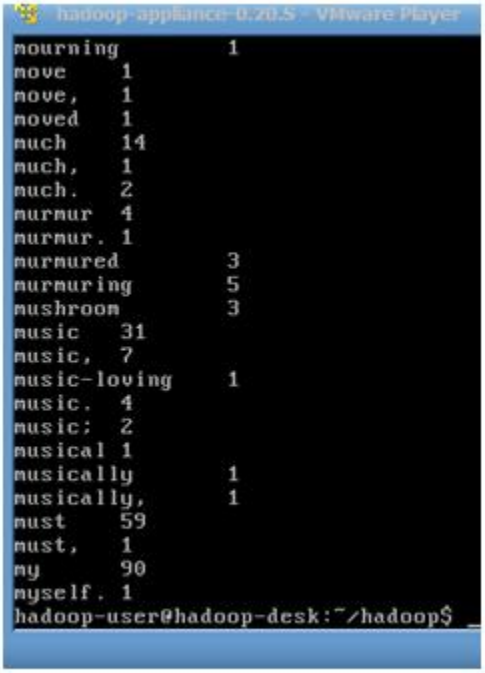
The Hadoop Distributed File System (HDFS) is a distributed file system designed to run on commodity hardware. It has many similarities with existing distributed file systems. However, the differences from other distributed file systems are significant. HDFS is highly fault-tolerant and is designed to be deployed on low-cost hardware. HDFS provides high throughput access to application data and is suitable for applications that have large data sets. HDFS relaxes a few POSIX requirements to enable streaming access to file system data. HDFS was originally built as

infrastructure for the Apache Nutch web search engine project. HDFS is now an Apache Hadoop subproject.

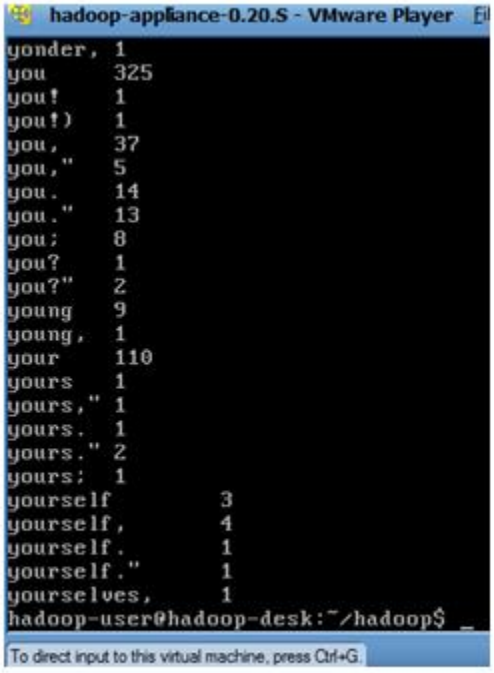
The HDFS is a distributed, scalable, and portable filesystem written in Java for the Hadoop framework. Each node in a Hadoop instance typically has a single datanode; a cluster of datanodes form the HDFS cluster. The situation is typical because each node does not require a datanode to be present. Each datanode serves up blocks of data over the network using a block protocol specific to HDFS. The filesystem uses the TCP/IP layer for communication; clients use RPC to communicate between each other. The HDFS stores large files (an ideal file size is a multiple of 64 MB), across multiple machines. It achieves reliability by replicating the data across multiple hosts, and hence does not require RAID storage on hosts.

SNAPSHOTS:

Output of word count :



part-r-00000



part-r-00001

For the comparison of pairs and stripes output lets take the below sample file as the input.

It is evident from the outputs that both derive at the same relative count values.

Sample input file :

```
cat rat cat cat hen dark

This is a sample input.
```

Output of pairs:

```
hadoop-user@hadoop-desk:~/hadoop$ h
oop/test3/part-r-00000
a        This     0.25
a        input.   0.25
a        is       0.25
a        sample   0.25
cat      cat      0.4
cat      dark     0.2
cat      hen      0.2
cat      rat      0.2
dark     cat      0.6
dark     hen      0.2
dark     rat      0.2
hen      cat      0.6
hen      dark     0.2
hen      rat      0.2
input.   This     0.25
input.   a        0.25
input.   is       0.25
input.   sample   0.25
is       This     0.25
is       a        0.25
is       input.   0.25
is       sample   0.25
hadoop-user@hadoop-desk:~/hadoop$
To direct input to this virtual machine, press Ctrl+G.
```

part-r-00000

```
hadoop-user@hadoop-desk:~/hado
oop/test3/part-r-00001
This     a        0.25
This     input.   0.25
This     is       0.25
This     sample   0.25
rat      cat      0.6
rat      dark     0.2
rat      hen      0.2
sample   This     0.25
sample   a        0.25
sample   input.   0.25
sample   is       0.25
hadoop-user@hadoop-desk:~/hado
To direct input to this virtual machine, press Ctrl+G.
```

part-r-00001

Output of Stripes:

```
hadoop-user@hadoop-desk:~/hadoop$ hadoop fs -cat
oop/test4/part-r-00000
a        is 0.25 This 0.25 input. 0.25 sample 0.25
cat      cat 0.4 dark 0.2 rat 0.2 hen 0.2
dark     cat 0.6 rat 0.2 hen 0.2
hen      cat 0.6 dark 0.2 rat 0.2
input.   is 0.25 a 0.25 This 0.25 sample 0.25
is       a 0.25 This 0.25 input. 0.25 sample 0.25
hadoop-user@hadoop-desk:~/hadoop$ _
```

part-r-00000

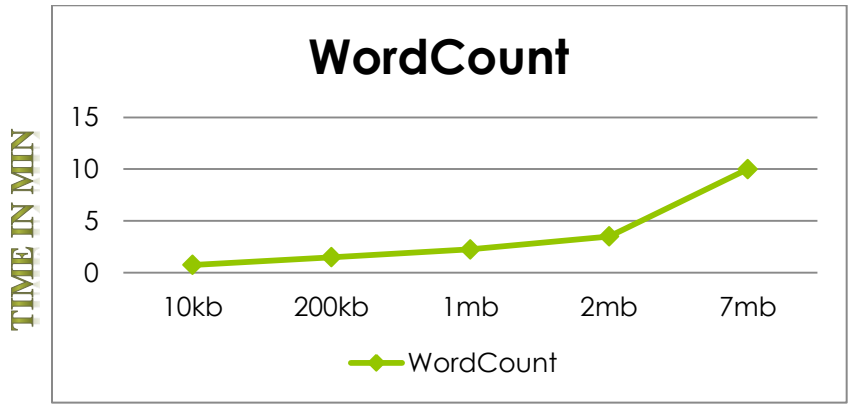
```
hadoop-user@hadoop-desk:~/hadoop$ hadoop fs -ca
oop/test4/part-r-00001
This     is 0.25 a 0.25 input. 0.25 sample 0.25
rat      cat 0.6 dark 0.2 hen 0.2
sample   is 0.25 a 0.25 This 0.25 input. 0.25
hadoop-user@hadoop-desk:~/hadoop$ _
```

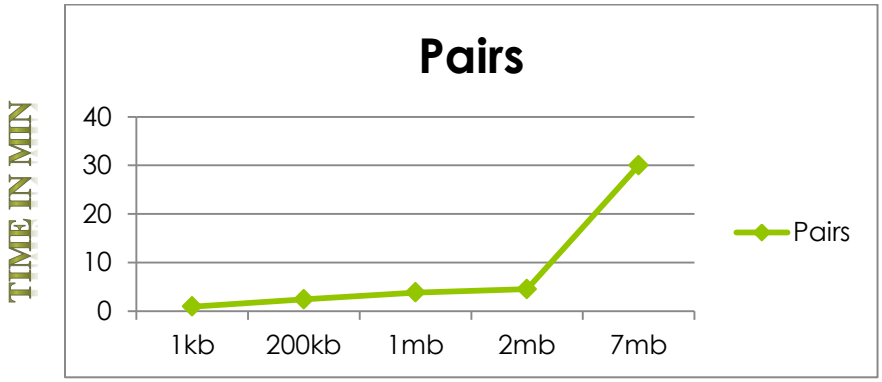
part-r-00001

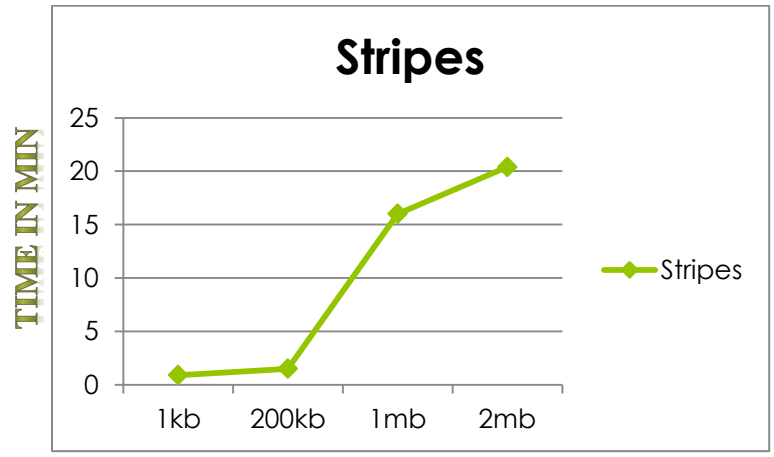
MapReduce Run on large input:

```
hadoop-appliance-0.20.S - VMware Player   File ▾ Virtual Machine ▾ Help ▾
hadoop-examples-0.20.104.1.1006042001.jar  pairs_relcount.jar
hadoop-test-0.20.104.1.1006042001.jar      README.txt
hadoop-tools-0.20.104.1.1006042001.jar     stripes_relcount.jar
inp                                        webapps
hadoop-user@hadoop-desk:~/hadoop$ bin/hadoop jar stripes_relcount.jar hdfs://loc
alhost:8020/user/hadoop/inp1 hdfs://localhost:8020/user/hadoop/stripes_output
11/12/02 08:44:57 INFO hdfs.DFSClient: Created HDFS_DELEGATION_TOKEN token 67 fo
r hadoop-user
11/12/02 08:44:57 INFO security.TokenCache: Got dt for hdfs://localhost/user/had
oop-user/.staging/job_201112020553_0023;uri=127.0.0.1:8020;t.service=127.0.0.1:8
020
11/12/02 08:44:57 INFO input.FileInputFormat: Total input paths to process : 2
11/12/02 08:44:58 INFO mapred.JobClient: Running job: job_201112020553_0023
11/12/02 08:44:59 INFO mapred.JobClient:  map 0% reduce 0%
11/12/02 08:45:32 INFO mapred.JobClient:  map 10% reduce 0%
11/12/02 08:45:38 INFO mapred.JobClient:  map 19% reduce 0%
11/12/02 08:45:41 INFO mapred.JobClient:  map 25% reduce 0%
11/12/02 08:45:44 INFO mapred.JobClient:  map 32% reduce 0%
11/12/02 08:45:59 INFO mapred.JobClient:  map 38% reduce 0%
11/12/02 08:46:03 INFO mapred.JobClient:  map 50% reduce 0%
11/12/02 08:46:23 INFO mapred.JobClient:  map 100% reduce 0%
11/12/02 08:46:32 INFO mapred.JobClient:  map 100% reduce 8%
11/12/02 08:46:37 INFO mapred.JobClient:  map 100% reduce 50%
11/12/02 08:46:54 INFO mapred.JobClient:  map 100% reduce 100%
_
To direct input to this virtual machine, press Ctrl+G.   vmware
```

The run time of word count, pairs and strips was recorded for various sizes of inputs and a graph was populated for comparison.







DESIGN DETAILS

Language: Java

IDE: Eclipse

Platform : HDFS

PROGRAMMER'S MANUAL

Eclipse IDE can be used for application development. Netbeans IDE can also be used.

Download VMPlayer and install hadoop

REFERENCES

http://hadoop.apache.org/common/docs/current/mapred_tutorial.html

http://hadoop.apache.org/common/releases.html

http://downloads.vmware.com/d/