Lab 11 Hadoop MapReduce (2)

1 Giới thiệu

Để thiết lập một Hadoop cluster, SV chọn ít nhất là 4 máy tính. Mỗi máy có vai trò như sau:

- 1 máy làm NameNode: dùng để quản lý không gian tên (namespace) vàđi ều khiển truy cập cho hệ thống. Chỉ có một máy NameNode duy nhất trong cluster.
- 1máy làm JobTracker: theo dõi hoạt động của các node con (slave node). Chỉ một máy làm JobTracker
- DataNode: lưu giữ các file dữ liệu và quản lý vùng lưu tr ữ cục bộ của nó. Có thể có nhiều DataNode.
- TaskTracker: là các node con (slaves) thực thi các tác vụ map và reduce. Có thể có nhiều TaskTracker.

Mỗi node trong cluster có thể là master hay slave. NameNode và JobTracker có thể là cùng một máy nhưng để tăng hiệu suất cho hệ thống thì chúng nên là hai máy riêng biệt và chúng là các máy đóng vai trò master. Các máy còn lại là slave và đảm nhận vai trò DataNode lẫn TaskTracker.

Yêu cầu sinh viên thực thi ứng dụng WordCount trên hai mô hình: Pseudo-Distributed Operation và Fully-Distributed Operation để hiểu rõ hoạt động của mô hình Map/Reduce và kiến trúc HDFS (Hadoop Distributed FileSystem).

2 Nội dung

2.1 Cài đặt và sử dụng MapReduce như một Cluster

SV chọn ít nhất 4 máy và thực hiện thiết lập các bước như sau:

- Download : hadoop distribution từ một trong các liên kết sau
  - http:// hadoop.apache.org
  - http://www.cse.hcmut.edu.vn/~nathu/XLSS
- Cấu hình 3 tập tin chính trong thư mục hadoop-version/conf:
  conf/core-site.xml: → Thiết lập namenode

```
<configuration>
 <property>
   <name>fs.default.name</name>
   <value>hdfs://172.28.12.20:9000</value>
 </property>
</configuration>
```

☺ SV thay đổi địa chỉ IP tương ứng với máy đã chọn !

conf/hdfs-site.xml:

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

conf/mapred-site.xml: → Thiết lập JobJacker

```
<?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>172.28.12.45:9001</value>            // SV thay doi dia chi IP tuong ung
</property>
<property>
   <name>mapred.local.dir</name>
   <value>/home/phuong/local</value>       // SV thay doi thu muc tuong ung
```

```
</property>


<property>
 <name>mapred.map.tasks</name>
 <value>20</value>                          // SV thay so luong map task
</property>


<property>
 <name>mapred.reduce.tasks</name>
 <value>4</value>                           // SV thay doi so luong reduce task
</property>


<property>
 <name>mapred.task.tracker.http.address</name>
 <value>172.28.12.45:50060</value>  // SV thay doi dia chi IP tuong ung
</property>
</configuration>
```

- Khởi động môi trường hadoop mapreduce bằng các lệnh sau:

  Trên NameNode thực hiện lệnh

  - $bin/hadoop namenode –format
  - $bin/start-dfs.sh

  Trên JobTracker thực hiện lệnh:

  - $bin/start-mapred.sh
  - Thực hiện duyệt các trang web sau để kiểm tra xem hadoop mapreduce đã hoạt động hay chưa :
    - Namenode: http://Dia chi IP NameNode:50070
    - JobTracker: http://Dia chi IP JobTracker:50030
- Thực thi ứng dụng mẫu được cung cấp bởi hadoop:

- $bin/hadoop fs   -put conf input
- $bin/hadoop jar hadoop-example-*.jar grep input output ‘dfs[a-z.]+’
- $bin/hadoop fs –get output output
- $cat output/*

- Kết thúc môi trường hadoop mapreduce

Trên NameNode thực hiện lệnh

- $bin/hadoop namenode –format
- $bin/stop-dfs.sh

Trên JobTracker thực hiện lệnh:

- $bin/stop-mapred.sh

2.2 Thực thi ứng dụng WordCount

SV có thể sử dụng mã nguồn WordCount.java của Google như bên dưới hoặc tự viết.

```
package org.myorg;
import java.io.IOException;
import java.util.*;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapred.*;
import org.apache.hadoop.util.*;
public class WordCount {
  public static class Map extends MapReduceBase implements Mapper<LongWritable, Text, Text, IntWritable> {
    private final static IntWritable one = new IntWritable(1);
    private Text word = new Text();
    public void map(LongWritable key, Text value, OutputCollector<Text, IntWritable> output, Reporter reporter) throws
IOException {
      String line = value.toString();
      StringTokenizer tokenizer = new StringTokenizer(line);
      while (tokenizer.hasMoreTokens()) {
        word.set(tokenizer.nextToken());
```

```
        output.collect(word, one);
      }
    }
  }
  public static class Reduce extends MapReduceBase implements Reducer<Text, IntWritable, Text, IntWritable> {
    public void reduce(Text key, Iterator<IntWritable> values, OutputCollector<Text, IntWritable> output, Reporter reporter) throws
IOException {
      int sum = 0;
      while (values.hasNext()) {
        sum += values.next().get();
      }
      output.collect(key, new IntWritable(sum));
    }
  }
  public static void main(String[] args) throws Exception {
    JobConf conf = new JobConf(WordCount.class);
    conf.setJobName("wordcount");
    conf.setOutputKeyClass(Text.class);
    conf.setOutputValueClass(IntWritable.class);
    conf.setMapperClass(Map.class);
    conf.setCombinerClass(Reduce.class);
    conf.setReducerClass(Reduce.class);
    conf.setInputFormat(TextInputFormat.class);
    conf.setOutputFormat(TextOutputFormat.class);
    FileInputFormat.setInputPaths(conf, new Path(args[0]));
    FileOutputFormat.setOutputPath(conf, new Path(args[1]));
    JobClient.runJob(conf);
  }
}
```

Để thực thi ứng dụng, sv cần thực hiện:

2.2. 1 Chuyển chương trình WordCount.java thành file .jar: vd WordCount.jar

$ mkdir wordcount_class

$ javac -classpath {HADOOP_HOME}/hadoop-${HADOOP_VERSION}-core.jar -d

wordcount_classes WordCount.java

$ jar –cvf wordcount.jar –C wordcount_classes/ .

2.2.2 Thực thi chương trình

SV tạo hai tập tin file1, file2 có nội dung tùy ý, chuyển chúng vào thư mục input và thực thi lệnh bên dưới:

$ bin/hadoop jar WordCount.jar org.myorg.WordCount input output

Chú ý: Một số lệnh thao tác trên HDFS

$ bin/hadoop dfs –put <source> <dest> : cung cấp input cho chương trình

$ bin/hadoop dfs –get <dest> <source> : lấy về output của chương trình.

$ bin/hadoop dfs –rmr <dir> : xóa thư mục.

$ bin/hadoop dfs –rm <file> : xóa tập tin

3 Bài tập

1 SV thực thi chương trình WordCount ver2 của Google.

2 SV viết chương trình tính PI theo mô hình Map/Reduce.