Lab 11: HADOOP MAPREDUCE

Biên soạn: ThS. Nguyễn Quang Hùng
E-mail: hungnq2@cse.hcmut.edu.vn

1. Giới thiệu:

- Hadoop Map/Reduce là một khung nền (software framework) mã nguồn mở, hỗ trợ người lập trình viết các ứng dụng theo mô hình Map/Reduce. Để hiện thực một ứng dụng theo mô hình Map/Reduce, sinh viên cần sử dụng các interface lập trình do Hadoop cung cấp như: Mapper, Reducer, JobConf, JobClient, Partitioner, OutputCollector, Reporter, InputFormat, OutputFormat, v.v..
- Yêu cầu sinh viên thực thi ứng dụng WordCount trên hai mô hình cluster để hiểu rõ hoạt động của mô hình Map/Reduce và kiến trúc HDFS (Hadoop Distributed FileSystem).

2. Tài liệu hướng dẫn cài đặt Apache Hadoop và MapReduce tutorial:

- Hadoop: http://hadoop.apache.org/docs/r1.1.2/#Getting+Started
- Tài liệu hướng dẫn cài đặt Apache Hadoop trên cluster (Cluster node setup): https://hadoop.apache.org/docs/r1.2.1/cluster_setup.html

- MapReduce Tutorial: http://hadoop.apache.org/docs/r1.1.2/mapred_tutorial.html

3. Chương trình ví dụ:

3.1. Cài đặt và sử dụng MapReduce

SV có thể cài đặt mô hình Single Node Mode hay Pseudo-Distributed Operation trên một máy đơn. Các bước thực hiện như sau:

- Download hadoop distribution từ một trong các liên kết sau: http:// hadoop.apache.org
- Khởi động môi trường hadoop mapreduce bằng các lệnh sau:

```
$ cd $HADOOP_HOME
$ bin/hadoop namenode –format
$ bin/start-all.sh
```

- Thực hiện duyệt các trang web sau để kiểm tra xem Hadoop MapReduce đã hoạt động hay chưa:

- Namenode: http://localhost:50070
- JobTracker: http://localhost: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

$ bin/stop-all.sh

Một số file cấu hình để thiết lập môi trường Peuso-DistribuCluster mode cho Hadoop gồm:
- Ba (3) tập tin chính trong thư mục hadoop-version/conf:

conf/core-site.xml:

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

conf/hdfs-site.xml:

```
<configuration>
<property>
<name>dfs.replication</name>
<value>1</value>
</property>
</configuration>
```

conf/mapred-site.xml:

```
<configuration>
<property>
<name>mapred.job.tracker</name>
<value>localhost:9001</value>
</property>
</configuration>
```

3.2. Chương trình ví dụ: WordCount.java

```
/* Filename: WordCount.java */
```

```
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);
       }
}
```

Biên dịch và thực thi
$ export HADOOP_HOME=<thư mục cài hadoop>
$ javac -classpath $HADOOP_HOME/hadoop-core-*.jar -d ../wordcount_classes/ WordCount.java
$ jar -cvf wordcount.jar -C ../wordcount_classes/ .
$ mkdir wordcount
$ cd wordcount
$ mkdir input
$ cd input/
$ vi file01
Hello Hadoop Goodbye Hadoop
$ vi file02
Hello World Bye World

$ bin/hadoop -fs mkdir wordcount
$ bin/hadoop dfs -put $HOME/input/file0* wordcount/input/
$ bin/hadoop dfs -ls wordcount/input
Found 2 items
-rw-r--r-- 1 hung supergroup 28 2012-05-08 08:15 /user/hung/wordcount/input/file01
-rw-r--r-- 1 hung supergroup 22 2012-05-08 08:15 /user/hung/wordcount/input/file02

$ bin/hadoop dfs -cat wordcount/input/file*
Hello Hadoop Goodbye Hadoop
Hello World Bye World

$ bin/hadoop dfs -ls wordcount/input
Found 2 items
-rw-r--r-- 1 hung supergroup 28 2012-05-08 08:15 /user/hung/wordcount/input/file01
-rw-r--r-- 1 hung supergroup 22 2012-05-08 08:15 /user/hung/wordcount/input/file02

$ bin/hadoop jar ~/wordcount.jar org.myorg.WordCount wordcount/input wordcount/output
12/05/08 08:17:38 WARN mapred.JobClient: Use GenericOptionsParser for parsing the arguments. Applications should implement Tool for the same.
12/05/08 08:17:38 INFO mapred.FileInputFormat: Total input paths to process : 2
12/05/08 08:17:38 INFO mapred.JobClient: Running job: job_201205080748_0004
12/05/08 08:17:39 INFO mapred.JobClient: map 0% reduce 0%
12/05/08 08:17:57 INFO mapred.JobClient: map 66% reduce 0%
12/05/08 08:18:17 INFO mapred.JobClient: map 100% reduce 100%
12/05/08 08:18:22 INFO mapred.JobClient: Job complete: job_201205080748_0004
12/05/08 08:18:22 INFO mapred.JobClient: Counters: 30
…..
12/05/08 08:18:22 INFO mapred.JobClient: Combine output records=6
12/05/08 08:18:22 INFO mapred.JobClient: Physical memory (bytes) snapshot=471007232
12/05/08 08:18:22 INFO mapred.JobClient: Reduce output records=5

12/05/08 08:18:22 INFO mapred.JobClient: Virtual memory (bytes) snapshot=1495506944
12/05/08 08:18:22 INFO mapred.JobClient: Map output records=8

$ bin/hadoop dfs -ls wordcount/output
Found 3 items
-rw-r--r-- 1 hung supergroup 0 2012-05-08 08:18 /user/hung/wordcount/output/_SUCCESS
drwxr-xr-x - hung supergroup 0 2012-05-08 08:17 /user/hung/wordcount/output/_logs
-rw-r--r-- 1 hung supergroup 41 2012-05-08 08:18 /user/hung/wordcount/output/part-00000

[hung@ppdslab01 hadoop-0.20.205.0]$ bin/hadoop dfs -cat wordcount/output/part-00000

WordCount v2.0

Here is a more complete WordCount which uses many of the features provided by the MapReduce framework we discussed so far.

This needs the HDFS to be up and running, especially for the DistributedCache-related features. Hence it only works with a pseudo-distributed or fully-distributed Hadoop installation.

Source Code

https://hadoop.apache.org/docs/r1.2.1/mapred_tutorial.html#Example%3A+WordCount+v2.0

- **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

Bài 1: SV thực thi chương trình WordCount có đếm tần suất xuất hiện các từ trong các văn bản.

Bài 2: SV viết chương trình tính PI theo mô hình Map/Reduce.
Bài 3: Cho trước tập các đỉnh (tọa độ trong không gian hai chiều (x, y)). Tìm đường đi ngắn nhất qua hai đỉnh cho trước. Gợi ý: hiện thực giải thuật Dijistra trên Hadoop MapReduce.