Hadoop 程序编写

上海交通大学 OmniLab

http://omnilab.sjtu.edu.cn

2013 年 11 月 27 日更新

目录

1 Hadoop MapReduce 运行的一般过程 **2**

2 使用 Java 编写 Hadoop MapReduce 程序 **2**

2.1 示例程序运行 2

2.1.1 示例代码`WordCount.java` 2

2.1.2 编译 Java 程序代码 3

2.1.3 准备数据 4

2.1.4 运行作业 4

2.2 程序模块说明 6

2.2.1 InputFormat and OutputFormat 7

2.3 MapReduce New API 示例代码 7

3 使用 Python 编写 Hadoop MapReduce 程序 **7**

4 使用 C++ 编写 Hadoop MapRedducech 程序 **7**

5 TODO **7**

6 参考资料 **7**

1 Hadoop MapReduce 运行的一般过程

2 使用 Java 编写 Hadoop MapReduce 程序

2.1 示例程序运行

2.1.1 示例代码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);
    }
}
```

2.1.2 编译 Java 程序代码

检查环境变量CLASSPATH是否设置为 CDH4 对应的位置:

```
$ echo $CLASSPATH
/opt/cloudera/parcels/CDH/lib/hadoop/*:/opt/cloudera/parcels/CDH/lib/hadoop/client-0.20/*:
```

编译代码:

```
$ mkdir wordcount_classes
```

```
$ javac -d wordcount_classes WordCount.java
```

生成 Jar 包：

```
$ jar -cvf wordcount.jar -C wordcount_classes/ .
```

2.1.3 准备数据

随便抓取一些文本数据作为`WordCount`程序的输入。输入数据需要从本地复制到 HDFS 中。

```
$ mkdir input
$ curl http://www.sjtu.edu.cn > input/file01.txt
$ curl http://net.sjtu.edu.cn > input/file02.txt
$ hadoop fs -mkdir /user/jianwen/wordcount
$ hadoop fs -copyFromLocal input /user/jianwen/wordcount/input
$ hadoop fs -ls /user/jianwen/wordcount/input
Found 2 items
-rw-r--r--   3 jianwen hadoop      74758 2013-11-21 14:35 /user/jianwen/wordcount/input/file01.txt
-rw-r--r--   3 jianwen hadoop      33553 2013-11-21 14:35 /user/jianwen/wordcount/input/file02.txt
```

2.1.4 运行作业

使用`hadoop`命令提交作业，在参数中指定 jar 包、Java 主类、输入文件夹、输出文件夹。

```
$ hadoop jar wordcount.jar org.myorg.WordCount \
/user/jianwen/wordcount/input /user/jianwen/wordcount/output
```

运行很快就能完成，在运行过程中显示的作业信息如下：

```
13/11/21 14:38:41 WARN mapred.JobClient: Use GenericOptionsParser for parsing the arguments. Appli
13/11/21 14:38:41 INFO mapred.FileInputFormat: Total input paths to process : 2
13/11/21 14:38:41 INFO mapred.JobClient: Running job: job_201311150128_0050
13/11/21 14:38:42 INFO mapred.JobClient:  map 0% reduce 0%
13/11/21 14:38:48 INFO mapred.JobClient:  map 100% reduce 0%
13/11/21 14:38:53 INFO mapred.JobClient:  map 100% reduce 41%
```

```
13/11/21 14:38:54 INFO mapred.JobClient:  map 100% reduce 100%
13/11/21 14:38:55 INFO mapred.JobClient: Job complete: job_201311150128_0050
13/11/21 14:38:55 INFO mapred.JobClient: Counters: 33
13/11/21 14:38:55 INFO mapred.JobClient:   File System Counters
13/11/21 14:38:55 INFO mapred.JobClient:     FILE: Number of bytes read=56585
13/11/21 14:38:55 INFO mapred.JobClient:     FILE: Number of bytes written=25028678
13/11/21 14:38:55 INFO mapred.JobClient:     FILE: Number of read operations=0
13/11/21 14:38:55 INFO mapred.JobClient:     FILE: Number of large read operations=0
13/11/21 14:38:55 INFO mapred.JobClient:     FILE: Number of write operations=0
13/11/21 14:38:55 INFO mapred.JobClient:     HDFS: Number of bytes read=120091
13/11/21 14:38:55 INFO mapred.JobClient:     HDFS: Number of bytes written=51565
13/11/21 14:38:55 INFO mapred.JobClient:     HDFS: Number of read operations=150
13/11/21 14:38:55 INFO mapred.JobClient:     HDFS: Number of large read operations=0
13/11/21 14:38:55 INFO mapred.JobClient:     HDFS: Number of write operations=288
13/11/21 14:38:55 INFO mapred.JobClient:   Job Counters
13/11/21 14:38:55 INFO mapred.JobClient:     Launched map tasks=3
13/11/21 14:38:55 INFO mapred.JobClient:     Launched reduce tasks=144
13/11/21 14:38:55 INFO mapred.JobClient:     Data-local map tasks=3
13/11/21 14:38:55 INFO mapred.JobClient:     Total time spent by all maps in occupied slots (ms)=9
13/11/21 14:38:55 INFO mapred.JobClient:     Total time spent by all reduces in occupied slots (ms
13/11/21 14:38:55 INFO mapred.JobClient:     Total time spent by all maps waiting after reserving
13/11/21 14:38:55 INFO mapred.JobClient:     Total time spent by all reduces waiting after reservi
13/11/21 14:38:55 INFO mapred.JobClient:   Map-Reduce Framework
13/11/21 14:38:55 INFO mapred.JobClient:     Map input records=1468
13/11/21 14:38:55 INFO mapred.JobClient:     Map output records=5378
13/11/21 14:38:55 INFO mapred.JobClient:     Map output bytes=121396
13/11/21 14:38:55 INFO mapred.JobClient:     Input split bytes=399
13/11/21 14:38:55 INFO mapred.JobClient:     Combine input records=5378
13/11/21 14:38:55 INFO mapred.JobClient:     Combine output records=1891
13/11/21 14:38:55 INFO mapred.JobClient:     Reduce input groups=1587
13/11/21 14:38:55 INFO mapred.JobClient:     Reduce shuffle bytes=69083
13/11/21 14:38:55 INFO mapred.JobClient:     Reduce input records=1891
13/11/21 14:38:55 INFO mapred.JobClient:     Reduce output records=1587
13/11/21 14:38:55 INFO mapred.JobClient:     Spilled Records=3782
13/11/21 14:38:55 INFO mapred.JobClient:     CPU time spent (ms)=196290
13/11/21 14:38:55 INFO mapred.JobClient:     Physical memory (bytes) snapshot=42758049792
13/11/21 14:38:55 INFO mapred.JobClient:     Virtual memory (bytes) snapshot=238107385856
13/11/21 14:38:55 INFO mapred.JobClient:     Total committed heap usage (bytes)=146269863936
13/11/21 14:38:55 INFO mapred.JobClient:   org.apache.hadoop.mapreduce.lib.input.FileInputFormatCo
13/11/21 14:38:55 INFO mapred.JobClient:     BYTES_READ=108311
```

可以在 HDFS 上直接查看运行结果，当然还也可以把结果复制到本地。

```
$ hadoop fs -ls /user/jianwen/wordcount/output
Found 146 items
-rw-r--r--   3 jianwen hadoop          0 2013-11-21 14:38 /user/jianwen/wordcount/output/_SUCCESS
drwxr-xr-x   - jianwen hadoop          0 2013-11-21 14:38 /user/jianwen/wordcount/output/_logs
-rw-r--r--   3 jianwen hadoop        505 2013-11-21 14:38 /user/jianwen/wordcount/output/part-0000
-rw-r--r--   3 jianwen hadoop        236 2013-11-21 14:38 /user/jianwen/wordcount/output/part-0000
-rw-r--r--   3 jianwen hadoop        255 2013-11-21 14:38 /user/jianwen/wordcount/output/part-0000
...

$ hadoop fs -cat /user/jianwen/wordcount/output/part-00001

$ hadoop fs -copyToLocal hadoop fs -cat /user/jianwen/wordcount/output ./wordcount-output
```

2.2 程序模块说明

使用 Java 开发 Hadoop 应用程序，使用的主要接口 (Interface) 如下，它们是 Hadoop MapReduce 的基本构建模块 (Building blocks)。在开发过程中，用户可以自己编写类实现这些接口，也可以使用 Hadoop 框架预先实现的类。

- `InputFormat`, `OutputFormat`: 控制 MapReduce 输入、输出中间数据的格式，即键值对的格式；
- `Mapper`: 实现 Map 函数的逻辑；
- `Reducer`: 实现 Reduce 函数的逻辑；
- Partitioner
- Reporter
- OutputCollector

除此以外，Hadoop 还包含一个辅助工具包`lib`，提供作业管理、数据分割、数据合并等辅助功能。

以 WordCount 程序为例，运行`hadoop jar`载入`org.myorg.WordCount`类后，程序执行流程如下：

1. 从`Wordcount.main()`函数开始执行；
    1. 在`JobConf`类对象实例中`conf`设定作业参数，包括：作业名、
    2.

2.2.1 InputFormat and OutputFormat

实现了`Input`和`Output`

2.3 MapReduce New API 示例代码

3 使用 Python 编写 Hadoop MapReduce 程序

4 使用 C++ 编写 Hadoop MapRedducech 程序

5 TODO

- 在命令行，而不是代码中指定这些参数：作业名、并行 Mapper/Reducer 数量；
- Java Makefile http://geosoft.no/development/javamake.html
- 0.20 引入了以`Context Objects`的新 API，这部分应该如何介绍和使用？目前新与“旧” 两套 API 都有比较广泛的使用。

6 参考资料

- Cloudera Hadoop Tutorial http://www.cloudera.com/content/cloudera-content/cloudera-docs/HadoopTutorial/CDH4/Hadoop-Tutorial.html
- Apache Hadoop Commands Guide http://hadoop.apache.org/docs/r1.2.1/commands_manual.html
- Java Doc: Package `org.apache.hadoop.mapred.lib` http://hadoop.apache.org/docs/current/api/org/apache/hadoop/mapred/lib/package-summary.html
- Hadoop Wiki: WordCount Example in Hadoop New APIs http://wiki.apache.org/hadoop/WordCount
- StackOverflow: Is it better to use the mapred or the mapreduce package to create a Hadoop Job? http://stackoverflow.com/questions/7598422/is-it-better-to-use-the-mapred-or-t

- “Upgrading to the new MapReduce APIs” by TOm Hughes-Croucher http://www.slideshare.net/sh1mmer/upgrading-to-the-new-map-reduce-api
- “Anatomy of a MapReduce Job Run with Hadoop” by Tomwhite http://answers.oreilly.com/topic/459-anatomy-of-a-mapreduce-job-run-with-hadoop/
- “Hadoop MapReduce 任务执行流程源代码详细解析” http://blog.csdn.net/riverm/article/details/6826200