

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	3
2	使用 Java 编写 Hadoop MapReduce 程序	3
2.1	示例程序运行	3
2.1.1	示例代码 <code>WordCount.java</code>	3
2.1.2	编译 Java 程序代码	4
2.1.3	准备数据	5
2.1.4	运行作业	5
2.2	程序模块说明	7
2.2.1	作业参数	7
2.2.2	Mapper Reducer	8
3	使用 Python 编写 Hadoop MapReduce 程序	10
3.1	Map 函数文件 <code>mapper.py</code>	10
3.2	Reduce 函数文件 <code>reducer.py</code>	10
3.3	在 Hadoop 上运行	11
4	TODO	12

目 录	2
5 参考资料	12

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 r)
            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 r)
            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.setInputFormat(TextInputFormat.class);
    conf.setOutputFormat(TextOutputFormat.class);

    conf.setMapOutputKeyClass(Text.class);
    conf.setMapOutputValueClass(IntWritable.class);
    conf.setOutputKeyClass(Text.class);
    conf.setOutputValueClass(IntWritable.class);

    conf.setMapperClass(Map.class);
    conf.setCombinerClass(Reduce.class);
    conf.setReducerClass(Reduce.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 wordcount/input
$ hadoop fs -ls 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 wordcount/input 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 wordcount/output/part-00001

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

2.2 程序模块说明

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, 在 `JobConf` 类对象实例 `conf` 中设置作业参数;
2. 使用 `runJob` 启动作业;
3. 作业在 **MapReduce** 框架中运行, 用户最终在先前指定的输出文件路径中看到运行结果。

2.2.1 作业参数

WordCount 示例程序中设置的作业参数包括:

- 设置作业名 `setJobName`;
 - 设置输入文件格式 `setInputFormat`, 以及输出文件的格式 `setOutputFormat`;
 - 设置 **Map** 函数的输出键值对的类型 (`setMapOutputKeyClass`, `setMapOutputValueClass`), 以及 **Reduce** 函数输出的键值对类型 (`setOutputKeyClass`, `setOutputValueClass`);
 - 设置本次作业使用的 **Map** 函数 (`setMapperClass`), **Combiner** 函数 (`setCombinerClass`), **Reduce** 函数 (`setReducerClass`);
 - 设置本次作业的输入文件路径 (`setInputPaths`), 以及输出文件路径 (`setOutputPath`)。
- 在示例程序中, 这两个参数是在启动程序的命令行参数中指定的。

Hadoop MapReduce 可以从多种数据文件读入数据, 也可以将计算结果保存到多种数据文件中, 如纯文本文件 (text files)、键值对序列文件 (sequence files)、

数据库文件 (DB files) 等, 用户还可以根据自己的需求进行扩展, 实现额外的输入/输出文件类型。

特别值得注意的是输入文件类型的选择。Hadoop 支持的输入文件类型必须实现 `InputFormat<K, V>` 接口, 文档说明了文件类型必须实现的几个功能:

1. 验证作业输入参数的有效性;
2. 将输入文件分成逻辑片 (logical InputSplits), 每个逻辑片指定给一个 Mapper 处理。分片功能被封装在 `getSplits()` 函数中;
3. 实现或调用合适的记录阅读器 (RecordReader), 将逻辑分片正确无误地解析为一条条记录, 供 Mapper 使用。

WordCount 示例程序中, 使用的输入文件类型是 `TextInputFormat`。`TextInputFormat` 实现了 `InputFormat<LongWritable, Text>` 接口, 并且实现了适合于处理文本输入文件的记录阅读器 `LineRecordReader`。`TextInputFormat` 将文件做逻辑分片后, 将逻辑分片按行解析, 输出的记录类型为 `<IntWritable, Text>`, 其中 Key 为该行结束位置在文件中的偏移量, Value 是该行的文本内容。Map 函数从输入文件获得输入数据, 因此接收的键值对类型应该与之保持一致。

另一块需要注意的是 Map 函数和 Reduce 函数的输出键值对数据类型。首先, 在作业参数中指定的输出类型, 应该与函数声明的类型一致; 其次, Map 函数的输出要作为 Reduce 函数的输入, Reduce 函数的输出也要可以作为 Reduce 函数的输入, 因此作业环境的 Map 输出类型、作业环境中的 Reduce 输出类型、Reduce 函数声明的输出类型这三者应该是相容的。为简便起见, `setOutputKeyClass` 和 `setOutputValueClass` 同时指定了 Map 和 Reduce 的 Key/Value 类型, `setMapOutputKeyClass` 和 `setMapOutputValueClass` 仅用在极少数需要额外定制的情况。

2.2.2 Mapper Reducer

在 WordCount 作业中, 我们使用的 Map 类是 `org.myorg.WordCount.Map`, 类型签名如下。

```
public static class Map
    extends MapReduceBase
    implements Mapper<LongWritable, Text, Text, IntWritable>
```


map函数在函数体中实现了，其函数签名如下：

```
void map(LongWritable key, Text value,  
OutputCollector<Text, IntWritable> output,  
Reporter reporter)  
throws IOException
```

可见，Map函数实现了Mapper接口，继承了MapReduceBase抽象基类。

其中，接口Mapper的如下

```
Interface Mapper<K1,V1,K2,V2>
```

其主要功能是约定 Map 函数的输入键值对类型<K1,V1>和输出键值对类型<K2,V2>，并约定接口函数map()原型：

```
public void map(K1 key, V1 value, OutputCollector<K2,V2> output, Reporter reporter)
```

MapReduceBase则提供了与作业控制相关的函数接口，不再详述。

类似地，可以分析org.myorg.WordCount.Reduce函数的签名：

```
public static class Reduce  
extends MapReduceBase  
implements Reducer<Text, IntWritable, Text, IntWritable>
```

reduce函数的签名：

```
public void reduce(Text key, Iterator<IntWritable> values,  
OutputCollector<Text, IntWritable> output,  
Reporter reporter)  
throws IOException
```

被实现的Reducer接口，签名如下，同样对 Reduce 的输入、输出键值对类型做了约定。

```
Interface Reducer<K2,V2,K3,V3>
```

最后，map和reduce函数生成的键值对结果，都需要通过OutputCollector实例收集。

3 使用 Python 编写 Hadoop MapReduce 程序

3.1 Map 函数文件_{mapper.py}

```
#!/usr/bin/env python

import sys

# input comes from STDIN (standard input)
for line in sys.stdin:
    # remove leading and trailing whitespace
    line = line.strip()
    # split the line into words
    words = line.split()
    # increase counters
    for word in words:
        # write the results to STDOUT (standard output);
        # what we output here will be the input for the
        # Reduce step, i.e. the input for reducer.py
        #
        # tab-delimited; the trivial word count is 1
        print '%s\t%s' % (word, 1)
```

3.2 Reduce 函数文件_{reducer.py}

```
#!/usr/bin/env python

from operator import itemgetter
import sys

current_word = None
current_count = 0
word = None

# input comes from STDIN
for line in sys.stdin:
    # remove leading and trailing whitespace
    line = line.strip()
```

```
# parse the input we got from mapper.py
word, count = line.split('\t', 1)

# convert count (currently a string) to int
try:
    count = int(count)
except ValueError:
    # count was not a number, so silently
    # ignore/discard this line
    continue

# this IF-switch only works because Hadoop sorts map output
# by key (here: word) before it is passed to the reducer
if current_word == word:
    current_count += count
else:
    if current_word:
        # write result to STDOUT
        print '%s\t%s' % (current_word, current_count)
    current_count = count
    current_word = word

# do not forget to output the last word if needed!
if current_word == word:
    print '%s\t%s' % (current_word, current_count)
```

3.3 在 Hadoop 上运行

准备数据:

```
$ cd src/pyWordCount
$ hadoop fs -mkdir pywordcount
$ hadoop fs -copyFromLocal input pywordcount/
```

使用 Hadoop Streaming 运行:

```
$ hadoop jar hadoop-streaming.jar \
-file py/mapper.py -mapper py/mapper.py \
-file py/reducer.py -reducer py/reducer.py \
```

```
-input pywordcount/input -output pywordcount/output
```

4 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。
- Tasks 任务数与 Splits 文件分片数，如何有效定制这两个参数？

5 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, from Chapter 6 of “The Definitive Guide of Hadoop”: How Hadoop works <http://answers.oreilly.com/topic/459-anatomy-of-a-mapreduce-job-run-with-hadoop/>
- “HADOOP: RECORDREADER AND FILEINPUTFORMAT” <http://hadoopi.wordpress.com/2013/05/27/understand-recordreader-inputsplit/>

- Source Code:`org.apache.hadoop.mapred.TextInputFormat` <http://greppcode.com/file/repo1.maven.org/maven2/org.jvnet.hudson.hadoop/hadoop-core/0.19.1-hudson-2/org/apache/hadoop/mapred/TextInputFormat.java>
- “Writing an Hadoop MapReduce Program in Python” by Michael G. Noll <http://www.michael-noll.com/tutorials/writing-an-hadoop-mapreduce-program-in-python/>