Map-Reduce Programming

王耀聰 陳威宇

jazz@nchc.org.tw

waue@nchc.org.tw

國家高速網路與計算中心(NCHC)

Outline

- 概念
- 程式基本框架及執行步驟方法
- 範例一：
  - —Hadoop 的 Hello World => Word Count
  - —說明
  - —動手做
- 範例二：
  - —進階版 => Word Count 2
  - —說明
  - —動手做

程式基本框架

Program Prototype (v 0.18)

Map 區

Reduce 區

設定區

```
Class MR{
    Class Mapper …{
    }
    Class Reducer …{
    }
    main(){
        JobConf conf = new JobConf( “MR.class” );
        conf.setMapperClass(Mapper.class);
        conf.setReduceClass(Reducer.class);
        FileInputFormat.setInputPaths(conf, new
    Path(args[0]));
        FileOutputFormat.setOutputPath(conf, new
    Path(args[1]));
                JobClient.runJob(conf);
}}
```

Map 程式碼

Reduce 程式碼

其他的設定參數程式碼

程式基本框架

Class Mapper

```
1  class MyMap extends MapReduceBase
   implements Mapper < INPUT KEY , INPUT VALUE , OUTPUT KEY , OUTPUT VALUE >
2  {
3  // 全域變數區
4  public void map ( key INPUT KEY , value INPUT VALUE ,
       OutputCollector < OUTPUT KEY , OUTPUT VALUE > output,
              Reporter  reporter) throws IOException
5          {
6          // 區域變數與程式邏輯區
7          output.collect( NewKey, NewValue);
8          }
9  }
```

程式基本框架

Class Reducer

```
1  class MyRed extends MapReduceBase
   implements Reducer < INPUT KEY , INPUT VALUE , OUTPUT KEY , OUTPUT VALUE >
2  {
3  // 全域變數區
4  public void reduce ( INPUT KEY key, Iterator< INPUT VALUE > values,
           OutputCollector< OUTPUT KEY , OUTPUT VALUE > output,
           Reporter  reporter) throws IOException
5          {
6          // 區域變數與程式邏輯區
7          output.collect( NewKey, NewValue);
8          }
9  }
```

Class Combiner

- 指定一個 combiner ，它負責對中間過程的輸出進行聚集，這會有助於降低從 Mapper 到 Reducer 數據傳輸量。
- 可不用設定交由 Hadoop 預設
- 也可不實做此程式，引用 Reducer
- 設定

  —JobConf.setCombinerClass(Class)

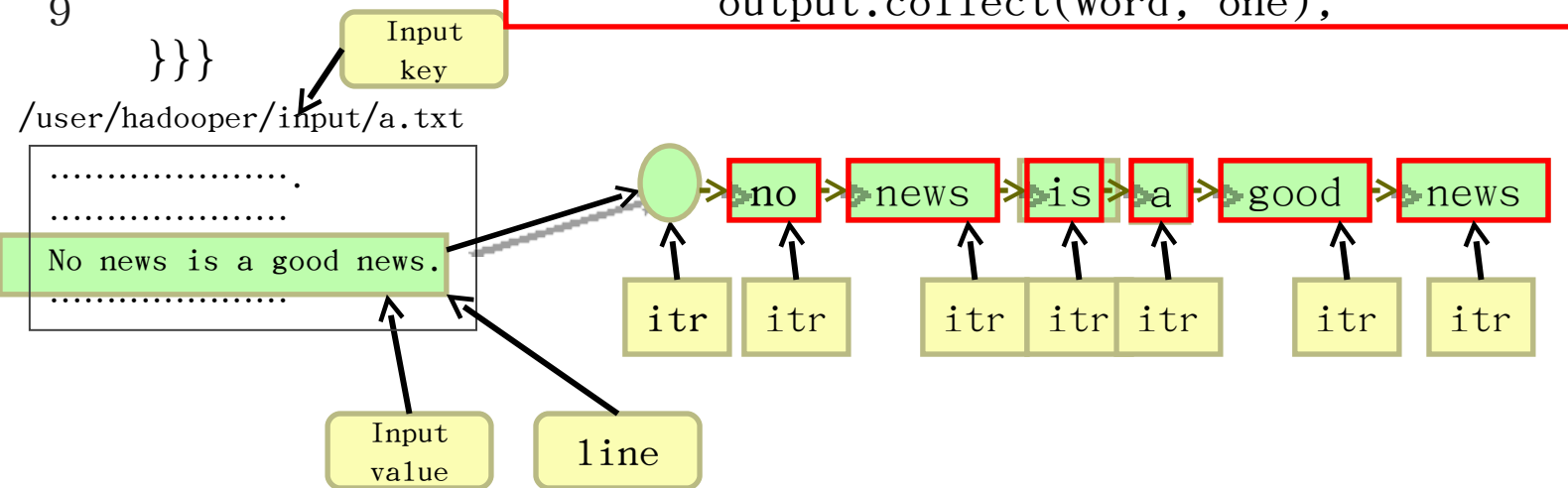
Run Job

- runJob(JobConf)
  - —提交作業，僅當作業完成時返回。
- submitJob(JobConf)
  - —只提交作業，之後需要你輪詢它返回的 RunningJob 句柄的狀態，並根據情況調度。
- JobConf.setJobEndNotificationURI(String)
  - —設置一個作業完成通知，可避免輪詢。

範例程式一

Word Count Sample (1)

```
1   class MapClass extends MapReduceBase implements
    Mapper<LongWritable, Text, Text, IntWritable> {
2           private final static IntWritable one = new IntWritable(1);
3           private Text word = new Text();
4           public void map( LongWritable key, Text value,
              OutputCollector<Text, IntWritable> output, Reporter
              reporter) throws IOException {
5                 String line = ((Text) value).toString();
6                 StringTokenizer itr = new StringTokenizer(line);
7                 while (itr.hasMoreTokens()) {
8                       word.set(itr.nextToken());
9                       output.collect(word, one);
    }}}
```

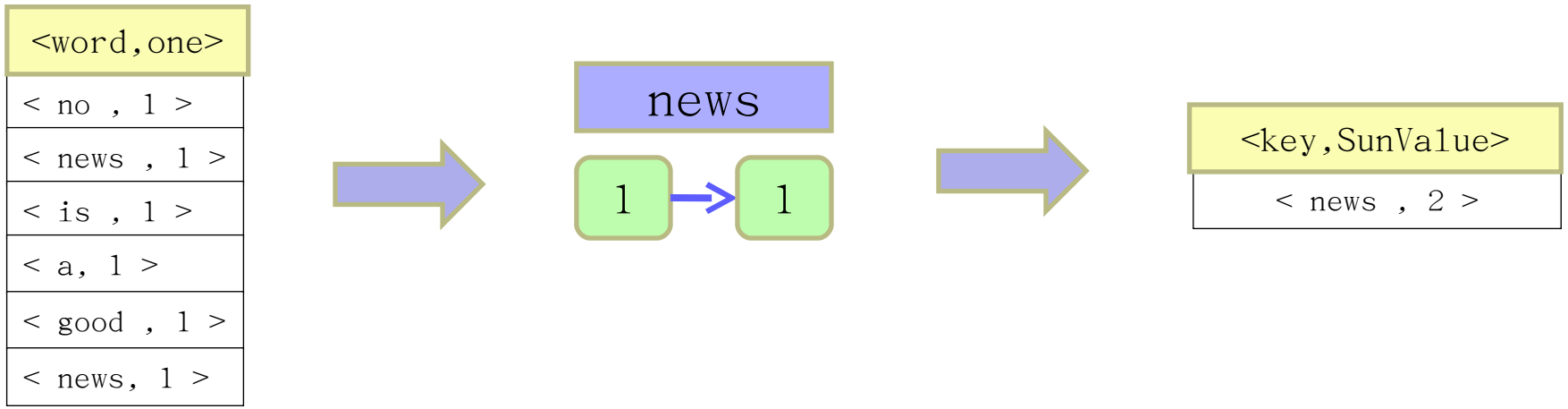


<word,one>
< no , 1 >
< news , 1 >
< is , 1 >
< a, 1 >
< good , 1 >
< news, 1 >

範例程式一

Word Count Sample (2)

```
1  class ReduceClass extends MapReduceBase implements Reducer< Text,
   IntWritable, Text, IntWritable> {
2          IntWritable SumValue = new IntWritable();
3          public void reduce( Text key, Iterator<IntWritable> values,
           OutputCollector<Text, IntWritable> output, Reporter reporter)
           throws IOException {
4                  int sum = 0;
5                  while (values.hasNext())
6                          sum += values.next().get();
7                  SumValue.set(sum);
8                  output.collect(key, SumValue);
   }}
```



範例程式一

Word Count Sample (3)

```
Class WordCount{
 main()
        JobConf conf = new JobConf(WordCount.class);
        conf.setJobName("wordcount");
        // set path
        FileInputFormat.setInputPaths(new Path(args[0]));
        FileOutputFormat.setOutputPath(new Path(args[1]));
        // set map reduce
        conf.setMapperClass(MapClass.class);
        conf.setCombinerClass(Reduce.class);
        conf.setReducerClass(ReduceClass.class);
        // run
        JobClient.runJob(conf);
}}
```

執行流程
基本步驟

編譯與執行

1. 編譯

— javac △ -classpath △ hadoop-*-core.jar △ -d △ MyJava △ MyCode.java

2. 封裝

— jar △ -cvf △ MyJar.jar △ -C △ MyJava △ .

3. 執行

— bin/hadoop △ jar △ MyJar.jar △ MyCode △ HDFS_Input/ △ HDFS_Output/

- 所在的執行目錄爲 Hadoop_Home
- 先放些文件檔到 HDFS 上的 input 目錄
- ./MyJava = 編譯後程式碼目錄
- ./input = hdfs 的輸入目錄
- Myjar.jar = 封裝後的編譯檔
- ./ouput = hdfs 的輸出目錄

範例一
動手做

WordCount1 練習 (I)

1. cd $HADOOP_HOME
2. bin/hadoop dfs -mkdir input
3. echo "I like NCHC Cloud Course." > inputwc/input1
4. echo "I like nchc Cloud Course, and we enjoy this crouse." > inputwc/input2
5. bin/hadoop dfs -put inputwc inputwc
6. bin/hadoop dfs -ls input

```
waue@vPro:/opt/hadoop$ bin/hadoop dfs -ls input
Found 2 items
-rw-r--r--   1 waue supergroup         26 2009-03-22 12:15 /user/waue/input/input1
-rw-r--r--   1 waue supergroup         52 2009-03-22 12:15 /user/waue/input/input2
waue@vPro:/opt/hadoop$
```

範例一
動手做

WordCount1 練習 (II)

- 編輯 WordCount.java
  http://trac.nchc.org.tw/cloud/attachment/wiki/jazz/Hadoop_Lab6/WordCount.java?format=raw
- mkdir MyJava
- javac -classpath hadoop-*-core.jar -d MyJava WordCount.java
- jar -cvf wordcount.jar -C MyJava .
- bin/hadoop jar wordcount.jar WordCount input/ output/

- 所在的執行目錄爲 Hadoop_Home （因爲 hadoop-*-core.jar ）
- javac 編譯時需要 classpath, 但 hadoop jar 時不用
- wordcount.jar = 封裝後的編譯檔，但執行時需告知 class name
- Hadoop 進行運算時，只有 input 檔要放到 hdfs 上，以便 hadoop 分析運算；執行檔（ wordcount.jar ）不需上傳，也不需每個 node 都放，程式的載入交由 java 處理

範例一
動手做

WordCount1 練習（III）

```
waue@vPro:/opt/hadoop$ mkdir MyJava
waue@vPro:/opt/hadoop$ javac -classpath hadoop-*-core.jar -d MyJava WordCount.java
waue@vPro:/opt/hadoop$ jar -cvf wordcount.jar -C MyJava .
新增 manifest
新增：WordCount.class (讀=1516)(寫=740)(壓縮 51%)
新增：WordCount$Reduce.class (讀=1591)(寫=642)(壓縮 59%)
新增：WordCount$Map.class (讀=1918)(寫=795)(壓縮 58%)
waue@vPro:/opt/hadoop$ bin/hadoop jar wordcount.jar WordCount input/ output/
09/03/22 11:39:01 WARN mapred.JobClient: Use GenericOptionsParser for parsing the argu
ments. Applications should implement Tool for the same.
09/03/22 11:39:01 INFO mapred.FileInputFormat: Total input paths to process : 1
09/03/22 11:39:01 INFO mapred.FileInputFormat: Total input paths to process : 1
09/03/22 11:39:02 INFO mapred.JobClient: Running job: job_200903201526_0007
09/03/22 11:39:03 INFO mapred.JobClient:  map 0% reduce 0%
09/03/22 11:39:08 INFO mapred.JobClient:  map 100% reduce 0%
09/03/22 11:39:15 INFO mapred.JobClient: Job complete: job_200903201526_0007
09/03/22 11:39:15 INFO mapred.JobClient: Counters: 16
09/03/22 11:39:15 INFO mapred.JobClient:   File Systems
09/03/22 11:39:15 INFO mapred.JobClient:     HDFS bytes read=320950
09/03/22 11:39:15 INFO mapred.JobClient:     HDFS bytes written=130568
09/03/22 11:39:15 INFO mapred.JobClient:     Local bytes read=168448
09/03/22 11:39:15 INFO mapred.JobClient:     Local bytes written=336932
09/03/22 11:39:15 INFO mapred.JobClient:   Job Counters
09/03/22 11:39:15 INFO mapred.JobClient:     Launched reduce tasks=1
```

範例一
動手做

WordCount1 練習（IV）

```
waue@vPro:/opt/hadoop$ bin/hadoop dfs -cat output/part-00000
Cloud   2
Course, 1
Course. 1
I       2
NCHC    1
and     1
course. 1
enjoy   1
like    2
nchc    1
this    1
we      1
```

範例二
動手做

WordCount 進階版

- WordCount2
- http://trac.nchc.org.tw/cloud/raw-attachment/wiki/jazz/Hadoop_Lab6/WordCount2.java
- 功能
  - 不計標點符號
  - 不管大小寫
- 步驟 (接續 WordCount 的環境)

1. echo "\." >pattern.txt && echo "\," >>pattern.txt
2. bin/hadoop dfs -put pattern.txt ./
3. mkdir MyJava2
4. javac -classpath hadoop-*-core.jar -d MyJava2 WordCount2.java
5. jar -cvf wordcount2.jar -C MyJava2 .

範例二
動手做

不計標點符號

- 執行
  - —bin/hadoop jar wordcount2.jar WordCount2 input output2 -skip pattern.txt dfs -cat output2/part-00000

```
waue@vPro:/opt/hadoop$ bin/hadoop dfs -cat output2/part-00000
Cloud   2
Course  2
I       2
NCHC    1
and     1
course  1
enjoy   1
like    2
nchc    1
this    1
we      1
```

範例二
動手做

不管大小寫

- 執行
  - bin/hadoop jar wordcount2.jar WordCount2 -Dwordcount.case.sensitive=false input output3 -skip pattern.txt

```
waue@vPro:/opt/hadoop$ bin/hadoop dfs -cat output3/part-00000
and     1
cloud   2
course  3
enjoy   1
i       2
like    2
nchc    2
this    1
we      1
```

範例二
補充說明

Tool1

- 處理 Hadoop 命令執行的選項

  -conf <configuration file>

  -D <property=value>

  -fs <local|namenode:port>

  -jt <local|jobtracker:port>

- 透過介面交由程式處理

  —ToolRunner.run(Tool, String[])

範例二
補充說明

DistributedCache

- 設定特定有應用到相關的、超大檔案、或只用來參考卻不加入到分析目錄的檔案
  - —如 pattern.txt 檔
- DistributedCache.addCacheFile(URI,conf)
  - —URI ＝ hdfs://host:port/FilePath