

# 实验三：倒排索引实验报告

小组成员：MF20330007 陈明远 MF20330003 陈迪

MG20330095 朱志威

## ● 设计思路：

本次实验通过输入小说文件来获取词频，其中 Map 部分的输入为 line，输出为((world, bookname),1)，然后通过自己设计的 combiner 将临时文件整合在一起，形成((world, bookname), sum),由于 key 值也是一个键值对，所以要重写 partitioner 函数，使得里面的 hash 函数的 key 只对 world 起作用，接着通过 reduce 函数将相同词的 bookname 和 sum 整合在一起，计算出平均数，然后输入所有 bookname 和出现次数。代码如下：

## ● InvertedIndex 代码：

```
1. import org.apache.hadoop.conf.Configuration;
2. import org.apache.hadoop.fs.Path;
3. import org.apache.hadoop.io.IntWritable;
4. import org.apache.hadoop.io.Text;
5. import org.apache.hadoop.mapreduce.Job;
6. import org.apache.hadoop.mapreduce.Mapper;
7. import org.apache.hadoop.mapreduce.Reducer;
8. import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
9. import org.apache.hadoop.mapreduce.lib.input.FileSplit;
10. import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
11. import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
12. import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;
13. import org.apache.hadoop.mapreduce.lib.partition.HashPartitioner;
14. import org.apache.hadoop.util.GenericOptionsParser;
15.
16. import java.io.IOException;
17. import java.util.ArrayList;
```

```

18. import java.util.List;
19. import java.util.StringTokenizer;
20.
21. public class InvertedIndex {
22.     public static class InvertedIndexMapper extends Mapper<Object, Text, Text, IntWritable> { //自定义自己的 mapper, 输入的 key 是小说文件, 输出一个单词和词频
23.         @Override
24.         protected void map(Object key, Text value, Context context)
25.             throws IOException, InterruptedException {
26.             FileSplit fileSplit = (FileSplit) context.getInputSplit();
27.             String filename = fileSplit.getPath().getName(); //获取文件的名字
28.             String line = value.toString().toLowerCase();
29.             StringTokenizer itr = new StringTokenizer(line);
30.             while (itr.hasMoreTokens()) {
31.                 Text word = new Text();
32.                 String temp = itr.nextToken();
33.                 if (filename.indexOf(".txt.segmented") > 0) { //因为输出要去掉.txt.segmented
34.                     word.set(temp + "," + filename.substring(0, filename.indexOf(".txt.segmented")));
35.                     context.write(word, new IntWritable(1));
36.                 } else {
37.                     word.set(temp + "," + filename.substring(0, filename.indexOf(".TXT.segmented")));
38.                     context.write(word, new IntWritable(1)); //只有写入 context 才会生效
39.                 }
40.             }
41.         }
42.     }
43. }
44.
45. public static class SumCombiner extends Reducer<Text, IntWritable, Text, IntWritable> { //定义自己的 combiner
46.     private IntWritable result = new IntWritable();
47.     @Override
48.     public void reduce(Text key, Iterable<IntWritable> values, Context context)
49.         throws IOException, InterruptedException {
50.         int sum = 0;
51.         for (IntWritable val : values) {
52.             sum += val.get();

```

```

53.         }
54.         result.set(sum);
55.         context.write(key, result);
56.     }
57. }
58.
59.     public static class NewPartitioner extends HashPartitioner<Text, IntWritable> { //定义自己的 partitioner, 重写哈希函数
60.         @Override
61.         public int getPartition(Text key, IntWritable value, int numReduceTasks) {
62.             String term = new String();
63.             term = key.toString().split(",")[0]; //将哈希函数的 key 变为词, key[0]为词
64.             return super.getPartition(new Text(term), value, numReduceTasks);
65.         }
66.     }
67.
68.     public static class InvertedIndexReducer extends Reducer<Text, IntWritable, Text, Text> {
69.         private Text word1 = new Text();
70.         private Text word2 = new Text();
71.         String bookname = new String();
72.         static Text CurrentItem = new Text(" ");
73.         static List<String> postingList = new ArrayList<String>();
74.
75.         @Override
76.         public void reduce(Text key, Iterable<IntWritable> values, Context context)
77.             throws IOException, InterruptedException {
78.             int sum = 0;
79.             word1.set(key.toString().split(",")[0]);
80.             bookname = key.toString().split(",")[1];
81.             for (IntWritable val : values) {
82.                 sum += val.get();
83.             } //将不同的 map 中统计的词频加在一起
84.             word2.set(bookname + ":" + sum);
85.             if (!CurrentItem.equals(word1) && !CurrentItem.equals(" ")) { //
                相同词出现在不同小说中要整合在一起。
86.                 StringBuilder out = new StringBuilder();
87.                 long count = 0;
88.                 for (String p : postingList) {
89.                     out.append(p);

```

```

90.         out.append(";");
91.         count += Long.parseLong(p.substring(p.indexOf(":") + 1))
    ;//将不同小说中词频数加载一起，目的是为了求平均值
92.     }
93.     StringBuilder out1 = new StringBuilder(String.format("%.2f",
        ((double) count / postingList.size()))).append(", " + out);//求平均值
94.     if (count > 0) {
95.         context.write(word1, new Text(out1.substring(0, out1.lastIndex
            tIndexOf(";")).toString()));
96.     }
97.     postingList = new ArrayList<String>();
98. }
99.     CurrentItem = new Text(word1);
100.     postingList.add(word2.toString());
101. }
102.
103.     @Override
104.     protected void cleanup(Context context) throws IOException, Interru
        ptedException {//处理最后一个词
105.         StringBuilder out = new StringBuilder();
106.         long count = 0;
107.         for (String p : postingList) {
108.             out.append(p);
109.             out.append(";");
110.             count += Long.parseLong(p.substring(p.indexOf(":") + 1));
111.         }
112.         StringBuilder out1 = new StringBuilder(String.format("%.2f", ((
            double) count / postingList.size()))).append(", " + out);
113.         if (count > 0) {
114.             context.write(word1, new Text(out1.substring(0, out1.lastIn
                dexOf(";")).toString()));
115.         }
116.     }
117. }
118.
119.     public static void main(String[] args) throws Exception {
120.         Configuration conf = new Configuration();
121.         String[] otherArgs = new GenericOptionsParser(conf, args).getRemain
            ingArgs();
122.         if (otherArgs.length != 2) {
123.             System.err.println("Usage: input output");
124.             System.exit(2);
125.         }
126.         Job job = Job.getInstance(conf, "InvertedIndex");//建立任务

```

```

127.      job.setJarByClass(InvertedIndex.class);
128.      job.setMapOutputKeyClass(Text.class);
129.      job.setMapOutputValueClass(IntWritable.class);
130.      job.setInputFormatClass(TextInputFormat.class);
131.      job.setOutputFormatClass(TextOutputFormat.class);
132.      job.setMapperClass(InvertedIndexMapper.class);
133.      job.setCombinerClass(SumCombiner.class);
134.      job.setPartitionerClass(NewPartitioner.class);
135.      job.setReducerClass(InvertedIndexReducer.class);
136.      job.setOutputKeyClass(Text.class);
137.      job.setOutputValueClass(Text.class);
138.      FileInputFormat.addInputPath(job, new Path(otherArgs[0]));
139.      FileOutputFormat.setOutputPath(job, new Path(otherArgs[1]));
140.      System.exit(job.waitForCompletion(true) ? 0 : 1);
141.  }
142. }

```

最终输出的部分结果和“江湖”和“风雪”两个单词的词频如下：

- 部分结果：

File - /user/2021st06/output/part-r-00000

Page 55 of 26673

一个个 1.73,卧龙生02.春秋笔:1;卧龙生05.飞花逐月:1;卧龙生06.飞铃:1;卧龙生09.风雨燕归来:1;卧龙生10.黑白剑:1;卧龙生11.黑白双娇:1;卧龙生13.剑无痕:1;卧龙生14.剑仙列传:3;卧龙生16.金笔点龙记:3;卧龙生17.金凤剪:1;卧龙生18.金剑雕翎:2;卧龙生19.惊鸿一剑震江湖:1;卧龙生20.梦幻之刀:1;卧龙生21.妙绝天香:2;卧龙生26.情剑无刃:2;卧龙生28.神州豪侠:1;卧龙生29.双凤旗:2;卧龙生32.天鹤谱:2;卧龙生34.天龙甲:2;卧龙生36.天香飘:3;卧龙生40.无名篇:1;卧龙生41.无形剑:1;卧龙生43.血剑丹心:4;卧龙生44.烟锁江湖:3;卧龙生46.摇花放鹰传:2;卧龙生47.一代天骄:1;卧龙生49.幽灵四艳:1;卧龙生51.玉手点将录:1;卧龙生54.指剑为媒:2;古龙02.白玉雕龙:1;古龙04.边城刀声:1;古龙08.苍穹神剑:1;古龙19.大人物:1;古龙24.护花铃:1;古龙27.血鹦鹉:1;古龙29.剑毒梅香:1;古龙30.剑客行:1;古龙32.剑玄录:6;古龙43.陆小凤06凤舞九天:1;古龙44.陆小凤07剑神一笑:2;古龙46.那一剑的风情:1;古龙58.情人箭:1;李凉02.霸枪艳血:1;李凉03.百败小赢家:1;李凉04.本尊分身:1;李凉07.赌棍小狂侠:1;李凉10.滑头傻小子:1;李凉13.江湖急救站:3;李凉15.江湖一担皮:1;李凉20.狂侠南宫鹰:1;李凉21.六宝江湖行:1;李凉22.矛盾天师:3;李凉23.妙贼丁小勾:2;李凉25.魔手邪怪:2;李凉26.奇神杨小邪续集:8;李凉27.奇神杨小邪:6;李凉34.天下第一当:3;李凉36.小鬼大赢家:1;李凉37.小鱼吃大鱼:3;李凉38.笑笑江湖:2;李凉40.新蜀山剑侠传续:1;李凉41.杨小邪发威:4;梁羽生02.冰川天女传:2;梁羽生03.冰河洗剑录:2;梁羽生07.弹指惊雷:1;梁羽生11.广陵剑:2;梁羽生12.瀚海雄风:1;梁羽生17.江湖三女侠:2;梁羽生19.狂侠天娇魔女:1;梁羽生20.联剑风云录:1;梁羽生23.龙虎斗京华:1;梁羽生26.女帝奇英传:2;梁羽生30.散花女侠:1;梁羽生38.云海玉弓缘:2;金庸04.天龙八部:1;金庸06.白马啸西风:1;金庸07.鹿鼎记:2;金庸08.笑傲江湖:1;金庸09.书剑恩仇录:1;金庸11.侠客行:1

一个中国 10.14.卧龙生01.鏖战:10;卧龙生02.春秋笔:14;卧龙生03.翠袖玉环:11;卧龙生04.地獄门:2;卧龙生05.飞花逐月:7;卧

图 1:部分输出结果(1)

:1;李凉17.惊神关小刀:2;李凉18.酒赌小浪子:4;李凉19.酒狂任小赌:1;梁羽生19.狂侠天娇魔女:1;梁羽生21.梁羽生传奇:1;梁羽生38.云海玉弓缘:1;金庸03连城诀:1  
一个时间 1.30,卧龙生04.地狱门:1;卧龙生06.飞铃:1;卧龙生16.金笔点龙记:1;卧龙生23.飘花令:1;卧龙生24.七绝剑:2;卧龙生41.无形剑:1;卧龙生43.血剑丹心:2;古龙08.苍穹神剑:1;古龙26.浣花洗剑录:1;古龙48.飘香剑雨:1;古龙63.武林外史:1;李凉03.百败小赢家:1;李凉11.会醉才会赢:1;李凉15.江湖一担皮:1;李凉17.惊神关小刀:2;李凉18.酒赌小浪子:4;李凉19.酒狂任小赌:1;梁羽生19.狂侠天娇魔女:1;梁羽生21.梁羽生传奇:1;金庸03连城诀:1  
一个是 1.47,卧龙生01.镖旗:1;卧龙生03.翠袖玉环:1;卧龙生06.飞铃:1;卧龙生10.黑白剑:1;卧龙生16.金笔点龙记:1;卧龙生17.金凤剪:3;卧龙生23.飘花令:2;卧龙生24.七绝剑:1;卧龙生27.琼楼十二曲:1;卧龙生28.神州豪侠:3;卧龙生31.桃花血令:1;卧龙生34.天龙甲:2;卧龙生44.烟锁江湖:2;卧龙生46.摇花放鹰传:1;古龙32.剑玄录:1  
一个月 12.12,卧龙生01.镖旗:9;卧龙生02.春秋笔:30;卧龙生03.翠袖玉环:14;卧龙生04.地狱门:11;卧龙生05.飞花逐月:6;卧龙生06.飞铃:38;卧龙生07.飞燕惊龙:7;卧龙生08.风尘侠隐:12;卧龙生09.风雨燕归来:4;卧龙生10.黑白剑:13;卧龙生11.黑白双娇:10;卧龙生12.剑气洞彻九重天:14;卧龙生13.剑无痕:25;卧龙生14.剑仙列传:1;卧龙生15.绛雪玄霜:5;卧龙生16.金笔点龙记:36;卧龙生17.金凤剪:18;卧龙生18.金剑雕翎:6;卧龙生19.惊鸿一剑震江湖:11;卧龙生20.梦幻之刀:10;卧龙生21.妙绝天香:16;卧龙生22.女捕头:10;卧龙生23.飘花令:11;卧龙生24.七绝剑:3;卧龙生25.七绝剑II还情剑:4;卧龙生26.情剑无刃:3;卧龙生27.琼楼十二曲:6;卧龙生28.神州豪侠:3;卧龙生29.双凤旗:261;卧龙生30.素手劫:374;卧龙生31.桃花血令:131;卧龙生32.天鹤谱:106;卧龙生33.天剑绝刀:321;卧龙生34.天龙甲:293;卧龙生35.天马霜衣:278;卧龙生36.天香飘:246;卧龙生37.天涯情侣:113;卧龙生38.铁剑玉佩:71;卧龙生39.铁苗神剑:358;卧龙生40.无名箫:136;卧龙生41.无形剑:256;卧龙生42.新仙鹤神针:188;卧龙生43.血剑丹心:350;卧龙生44.烟锁江湖:238;卧龙生45.燕子传奇:132;卧龙生46.摇花放鹰传:446;卧龙生47.一代天骄:353;卧龙生48.银月飞霜:132;卧龙生49.幽灵四艳:221;卧龙生50.玉钗盟:320;卧龙生51.玉手点将录:150;卧龙生52.袁紫烟:30;卧龙生53.岳小钗:308;卧龙生54.指剑为媒:57;古龙01.白玉老虎:111;古龙02.白玉雕龙:24;古龙03.碧血洗银枪:81;古龙04.边城刀声:50;古龙05.边城浪子:62;古龙06.彩环曲:76;古龙07.残金缺玉:82;古龙08.苍穹神剑:131;古龙09.楚留香01.血海飘香:47;古龙10.楚留香02.大沙漠:23;古龙11.楚留香03.画眉鸟:96;古龙12.楚留香04.借尸还魂(鬼恋传奇):41;古龙13.楚留香05.蝙蝠传奇:80;古龙14.楚留香06.桃花传奇:8;古龙15.楚留香07.新月传奇:20;古龙16.楚留香08.午夜兰花:68;古龙17.大地飞鹰:66;古龙18.梅竹梅传:151;古龙19.大地飞鹰:66;古龙20.多情剑客无情剑:150;古龙21.飞刀;古龙22.古龙23.月射天狼

图 2:部分输出结果(2)

江湖 116.06,卧龙生01.镖旗:275;卧龙生02.春秋笔:329;卧龙生03.翠袖玉环:402;卧龙生04.地狱门:105;卧龙生05.飞花逐月:298;卧龙生06.飞铃:244;卧龙生07.飞燕惊龙:269;卧龙生08.风尘侠隐:228;卧龙生09.风雨燕归来:198;卧龙生10.黑白剑:223;卧龙生11.黑白双娇:117;卧龙生12.剑气洞彻九重天:299;卧龙生13.剑无痕:261;卧龙生14.剑仙列传:25;卧龙生15.绛雪玄霜:274;卧龙生16.金笔点龙记:318;卧龙生17.金凤剪:326;卧龙生18.金剑雕翎:397;卧龙生19.惊鸿一剑震江湖:346;卧龙生20.梦幻之刀:106;卧龙生21.妙绝天香:28;卧龙生22.女捕头:317;卧龙生23.飘花令:263;卧龙生24.七绝剑:130;卧龙生25.七绝剑II还情剑:144;卧龙生26.情剑无刃:7;卧龙生27.琼楼十二曲:181;卧龙生28.神州豪侠:261;卧龙生29.双凤旗:261;卧龙生30.素手劫:374;卧龙生31.桃花血令:131;卧龙生32.天鹤谱:106;卧龙生33.天剑绝刀:321;卧龙生34.天龙甲:293;卧龙生35.天马霜衣:278;卧龙生36.天香飘:246;卧龙生37.天涯情侣:113;卧龙生38.铁剑玉佩:71;卧龙生39.铁苗神剑:358;卧龙生40.无名箫:136;卧龙生41.无形剑:256;卧龙生42.新仙鹤神针:188;卧龙生43.血剑丹心:350;卧龙生44.烟锁江湖:238;卧龙生45.燕子传奇:132;卧龙生46.摇花放鹰传:446;卧龙生47.一代天骄:353;卧龙生48.银月飞霜:132;卧龙生49.幽灵四艳:221;卧龙生50.玉钗盟:320;卧龙生51.玉手点将录:150;卧龙生52.袁紫烟:30;卧龙生53.岳小钗:308;卧龙生54.指剑为媒:57;古龙01.白玉老虎:111;古龙02.白玉雕龙:24;古龙03.碧血洗银枪:81;古龙04.边城刀声:50;古龙05.边城浪子:62;古龙06.彩环曲:76;古龙07.残金缺玉:82;古龙08.苍穹神剑:131;古龙09.楚留香01.血海飘香:47;古龙10.楚留香02.大沙漠:23;古龙11.楚留香03.画眉鸟:96;古龙12.楚留香04.借尸还魂(鬼恋传奇):41;古龙13.楚留香05.蝙蝠传奇:80;古龙14.楚留香06.桃花传奇:8;古龙15.楚留香07.新月传奇:20;古龙16.楚留香08.午夜兰花:68;古龙17.大地飞鹰:66;古龙18.梅竹梅传:151;古龙19.大地飞鹰:66;古龙20.多情剑客无情剑:150;古龙21.飞刀;古龙22.古龙23.月射天狼

图 3:“江湖”输出结果(1)

酒赌小浪子:17;李凉19.酒狂任小赌:167;李凉20.狂侠南官鹰:15;李凉21.六宝江湖行:72;李凉22.矛盾天师:59;李凉23.妙贼丁小勾:16;李凉24.妙贼丁小勾续集:13;李凉25.魔手邪怪:6;李凉26.奇神杨小邪续集:98;李凉27.奇神杨小邪:102;李凉28.忍者龟:77;李凉29.神偷小千:17;李凉30.淘气世家:81;李凉31.天才混混:89;李凉32.天才混混外集:27;李凉33.天齐大帝:62;李凉34.天下第一:52;李凉35.武林嘻游记:180;李凉36.小鬼大赢家:29;李凉37.小鱼吃大鱼:42;李凉38.笑笑江湖:69;李凉39.新蜀山剑侠传:20;李凉40.新蜀山剑侠传续:1;李凉41.杨小邪发威:34;梁羽生01.白发魔女传:63;梁羽生02.冰川天女传:65;梁羽生03.冰河洗剑录:65;梁羽生04.冰魄寒光剑:4;梁羽生05.草莽龙蛇传:143;梁羽生06.大唐游侠传:97;梁羽生07.弹指惊雷:77;梁羽生08.飞凤潜龙:4;梁羽生09.风雷震九洲:135;梁羽生10.风云雷电:139;梁羽生11.广陵剑:149;梁羽生12.瀚海雄风:69;梁羽生13.还剑奇情录:8;梁羽生14.幻剑灵旗:27;梁羽生15.慧剑心魔:102;梁羽生16.剑网尘丝:127;梁羽生17.江湖三女侠:252;梁羽生18.绝塞传烽录:40;梁羽生19.狂侠天娇魔女:180;梁羽生20.联剑风云录:126;梁羽生21.梁羽生传奇:74;梁羽生22.龙凤宝钗缘:109;梁羽生23.龙虎斗京华:182;梁羽生24.鸣镝风云录:134;梁羽生25.牧野流星:35;梁羽生26.帝奇英雄传:49;梁羽生27.萍踪侠影录:40;梁羽生28.七剑下天山:77;梁羽生29.塞外奇侠传:2;梁羽生30.散花女侠:48;梁羽生31.随笔集: 笔不花:4;梁羽生32.随笔集: 笔花六照:15;梁羽生34.武当一剑:94;梁羽生35.武林天骄:45;梁羽生36.侠骨丹心:103;梁羽生37.游剑江湖:275;梁羽生38.云海玉弓缘:81;金庸01.飞狐外传:67;金庸02.雪山飞狐:29;金庸03.连城诀:35;金庸04.天龙八部:126;金庸05.射雕英雄传:57;金庸06.白马啸西风:5;金庸07.鹿鼎记:75;金庸08.笑傲江湖:199;金庸09.书剑恩仇录:64;金庸10.神雕侠侣:81;金庸11.侠客行:61;金庸12.倚天屠龙记:145;金庸13.碧血剑:65;金庸14.鸳鸯刀:35

Cancel

Download

图 4:“江湖”输出结果(2)



风雪 4.53,卧龙生01.镖旗:3;卧龙生07.飞燕惊龙:16;卧龙生08.风尘侠隐:1;卧龙生09.风雨燕归来:1;卧龙生12.剑气洞彻九重天:36;卧龙生15.绛雪玄霜:4;卧龙生18.金剑雕翎:9;卧龙生19.惊鸿一剑震江湖:2;卧龙生22.女捕头:4;卧龙生25.七绝剑II还情剑:2;卧龙生27.琼楼十二曲:1;卧龙生31.桃花血令:2;卧龙生36.天香飘:2;卧龙生38.铁剑玉佩:7;卧龙生39.铁苗神剑:7;卧龙生42.新仙鹤神针:15;卧龙生47.一代天骄:3;卧龙生49.幽灵四艳:1;卧龙生52.袁紫烟:6;古龙07.残金缺玉:7;古龙13.楚留香05蝙蝠传奇:1;古龙14.楚留香06桃花传奇:1;古龙20.多情剑客无情剑:4;古龙23.孤星传:2;古龙24.护花铃:1;古龙25.欢乐英雄:3;古龙29.剑毒梅香:2;古龙31.剑气书香:3;古龙42.陆小凤05幽灵山庄:1;古龙46.那一剑的风情:3;古龙47.怒剑狂花:2;古龙56.七种武器06离别钩:1;古龙61.失魂引:14;古龙63.武林外史:24;古龙64.湘妃剑:5;古龙67.英雄无泪:8;李凉01.暗器高手:1;李凉02.霸枪艳血:5;李凉03.百败小赢家:1;李凉04.本尊分身:6;李凉09.红顶记:1;李凉11.会醉才会赢:1;李凉14.江湖双响炮:4;李凉16.江湖一品郎:1;李凉19.酒狂任小赌:32;李凉31.天才混混:4;李凉34.天下第一当:1;李凉36.小鬼大赢家:1;李凉39.新蜀山剑侠传:1;李凉40.新蜀山剑侠传续:12;梁羽生01.白发魔女传:4;梁羽生02.冰川天女传:5;梁羽生03.冰河洗剑录:1;梁羽生07.弹指惊雷:1;梁羽生09.风雷震九洲:1;梁羽生10.风云雷电:3;梁羽生12.瀚海雄风:1;梁羽生14.幻剑灵旗:1;梁羽生20.联剑风云录:1;梁羽生21.梁羽生传奇:1;梁羽生22.龙凤宝钗缘:5;梁羽生25.牧野流星:5;梁羽生26.女帝奇英传:3;梁羽生27.萍踪侠影录:3;梁羽生30.散花女侠:1;梁羽生37.游剑江湖:1;梁羽生38.云海玉弓缘:1;金庸01.飞狐外传:3;金庸03.连城诀:1;金庸04.天龙八部:1;金庸05.射雕英雄传:7;金庸06.白马啸西风:11;金庸07.鹿鼎记:1;金庸10.神雕侠侣:2;金庸12.倚天屠龙记:1

CancelDownload

图 5：“风雪”输出结果

- 输出结果路径： /user/2021st06
- WebUI 报告：

Yadooop

MapReduce Job job\_1604923458283\_0171

Job Overview

Job Name: InvertedIndex

User Name: 2021st06

Queue: root.2021team06

State: SUCCEEDED

Uberized: false

Submitted: Mon Nov 23 20:42:47 CST 2020

Started: Mon Nov 23 20:41:10 CST 2020

Finished: Mon Nov 23 20:42:35 CST 2020

Elapsed: 1mins, 24sec

Diagnostics:

Average Map Time: 3sec

Average Shuffle Time: 2mins, 15sec

Average Merge Time: 1sec

Average Reduce Time: -1mins, -5sec

ApplicationMaster		Start Time	Node	Logs
Attempt Number	1	Mon Nov 23 20:41:06 CST 2020	slave018:8042	logs

Task Type	Total	Complete	
Map	218	218	
Reduce	1	1	
Attempt Type	Failed	Killed	Successful
Maps	0	0	218
Reduces	0	0	1