

《MapReduce海量数据处理》

实验3 倒排索引

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **学院名称** | **：** | 计算机科学与技术系 | | | | | |
| **小组成员** | **：** | 林喜鹏 李朝阳 饶璐 唐玉婷 | | | | | |
| **学号** | **：** | MG1833046 MF1833035 MF1833059 MF1833070 | | | | | |
| **小组组长** | **：** | 林喜鹏 | | | | | |
| **时间** | **：** | 2018 | 年 | 10 | 月 | 30 | 日 |

1. 实验要求

**实验任务**

请实现课堂上介绍的“带词频属性的文档倒排算法”。

在统计词语的倒排索引时，除了要输出带词频属性的倒排索引，还请计算每个词语的“平均出现次数”（定义见下）并输出。

“平均出现次数”在这里定义为：

*平均出现次数=词语在全部文档中出现的频数总和 / 包含该词语的文档数*

假如文档集中有四本小说：A、B、C、D。词语“江湖”在文档 A 中出现了 100次，在文档 B 中出现了 200 次，在文档 C 中出现了 300 次，在文档 D 中没有出现。则词语“江湖”在该文档集中的“平均出现次数”为（100 + 200 + 300) / 3 = 200

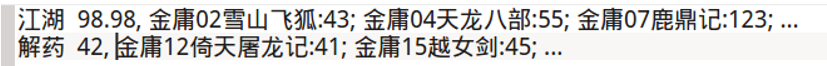
注意 这两个计算任务请在同一个 MapReduce Job 中完成。

**输出格式**

对于每个词语，输出一个键值对，该键值对的格式如下：

*[词语] \TAB 平均出现次数，小说1:词频；小说2:词频；小说3:词频；...；小说N:词频*

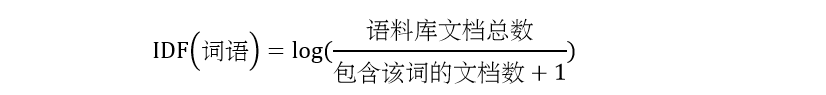
输出中的小说名需要去掉“.txt.segmented”的文件名后缀。 下图展示了输出文件的一个片段（图中内容仅为格式示例）：



**选做内容**

1）使用另外一个 MapReduce Job 对每个词语的平均出现次数进行全局排序，输出排序后的结果。

2）为每位作家、计算每个词语的 TF-IDF。TF 定义为某个词语在某个作家的所有作品中的出现次数之和。IDF 定义为:



输出格式：作家名字，词语，该词语的 TF-IDF。

**提交格式**

压缩包,包括:源代码、JAR 包、JAR 包执行说明及实验报告。

1. Map和Reduce的设计思路

要使得两个计算任务请在同一个 MapReduce Job 中完成，则除了基本的Mapper和Reducer之外，还必须重载Combiner和Partitioner,具体如下：

Map:

1. Mapper

输入:key是文件当前的偏移量，value是文件当前行内容

输出:key是word#filename，value是1

获取当前处理文件名filename,然后对value值进行切分，得到多个word值，此时将每个word和filename拼接到一起作为输出key，其计数值为1，也即value为1。

1. Combiner

输入:key是word#filename，value为对应的[1, 1, 1, …]

输出:key是word#filename，value是同一key下的累加和

作用是将Mapper输出的中间结果相同key部分的value累加，减少向Reduce节点传输的数据量。

Reduce:

1. Partitioner

输入:key是word#filename，value则是累加和

输出:key是word#filename，value则为累加和

这里是为了保证同一word都在同一Reduce节点进行处理。

1. Reducer

输入:key是word#filename，vaule为[累加和1, 累加和2, …]

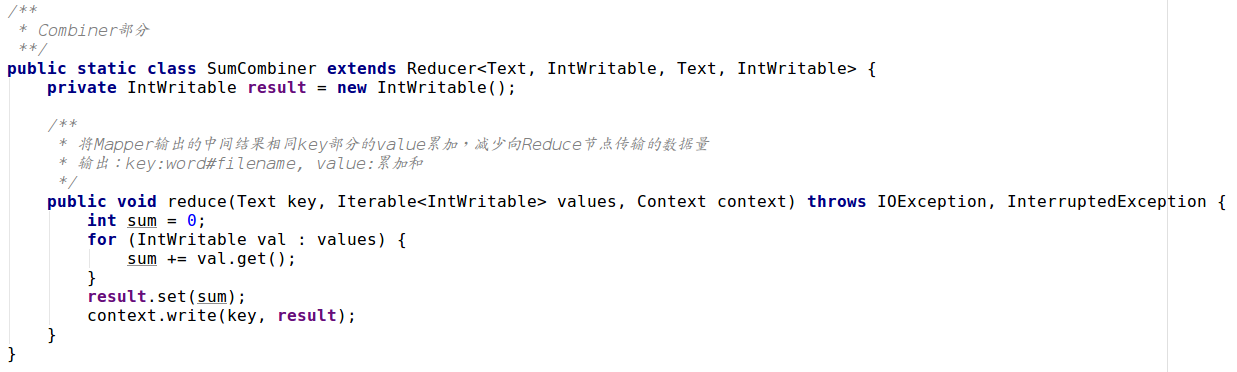
输出:key是word，value为[平均出现次数，filename:词频;filename:词频;…]

Reducer稍微麻烦一点，首先需要利用Reduce节点输入的key值都是有序的，将key拆分，对于同一word，每次都保存其filename和词频，并统计其总出现次数和总出现文档数；当同一word处理完后，计算平均出现次数，将其与filename及其词频作为value输出。

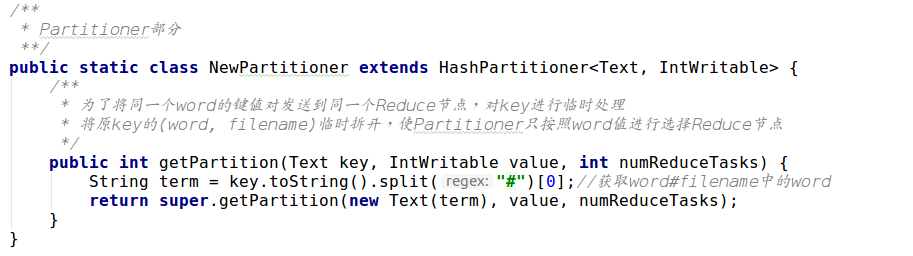
1. Map和Reduce代码

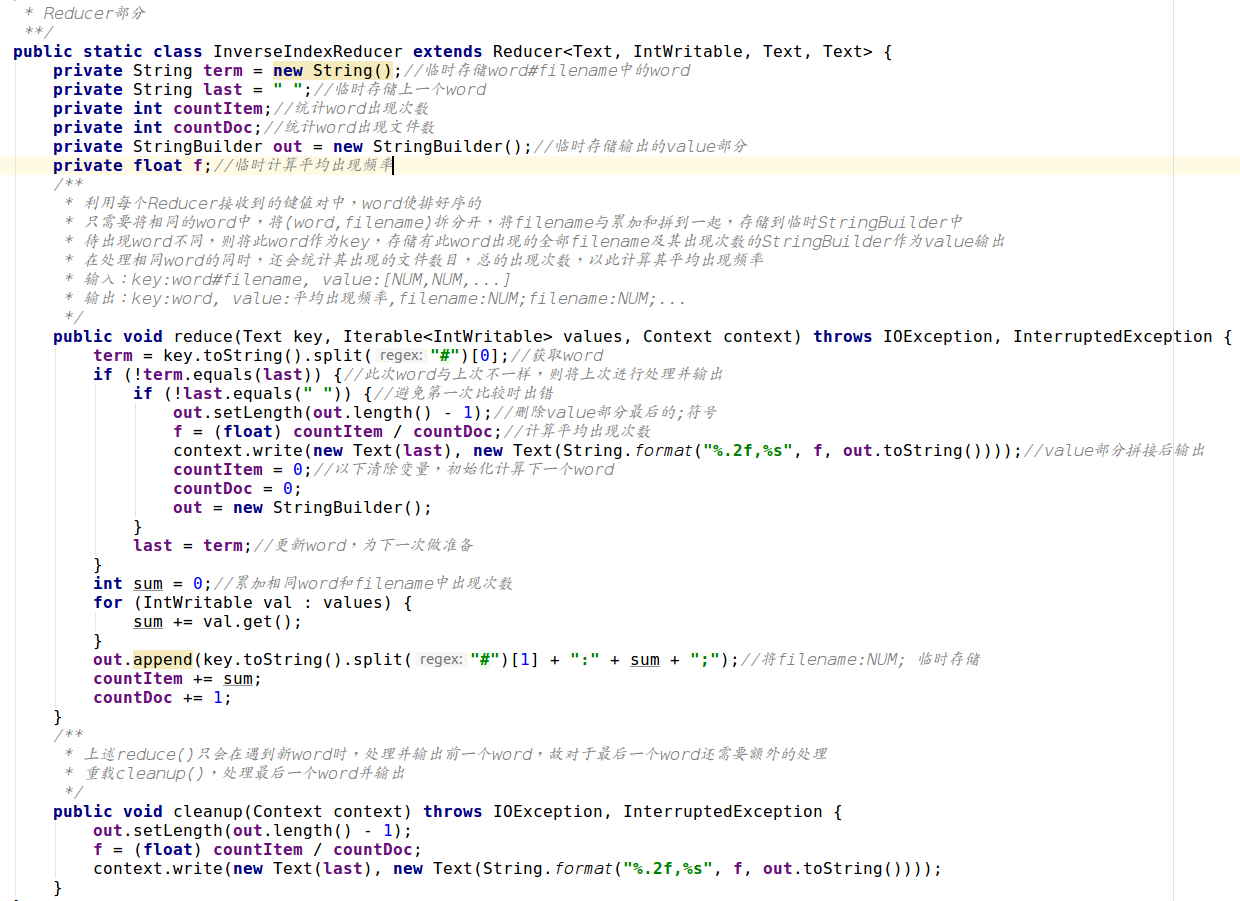
Map:





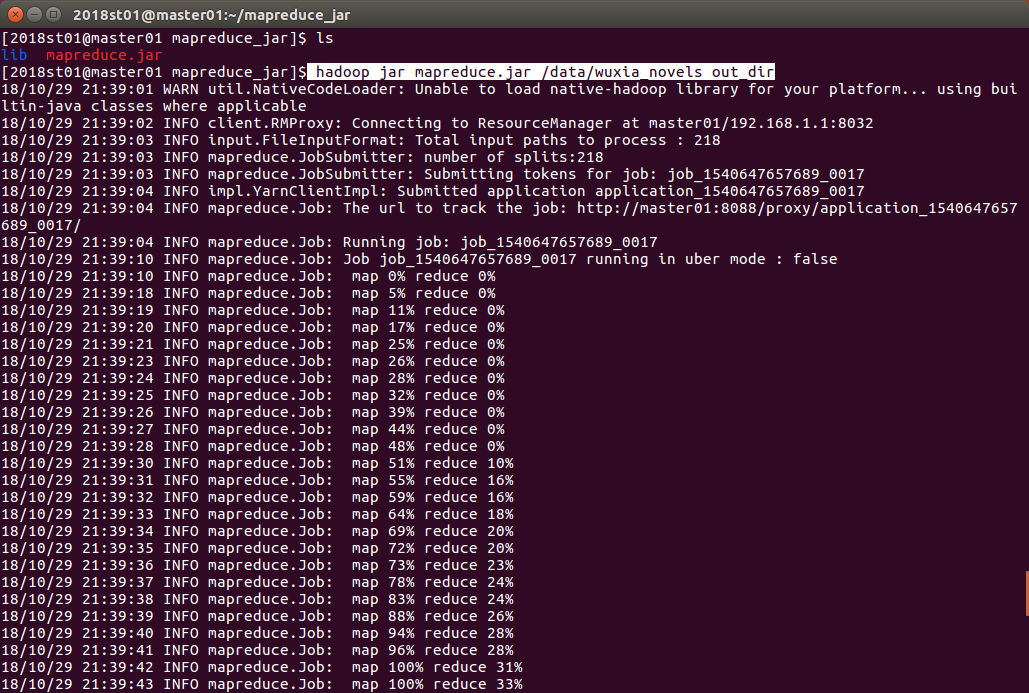
Reduce:

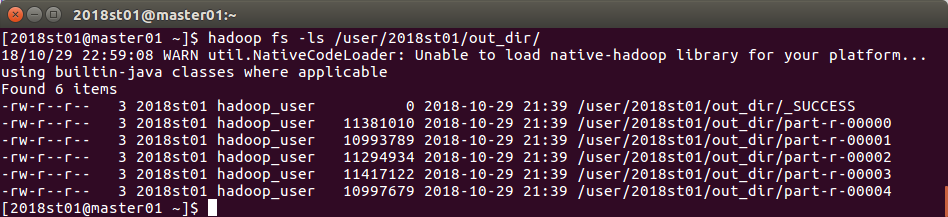




1. 本机实现后将jar包传到集群进行测试并检查输出文件

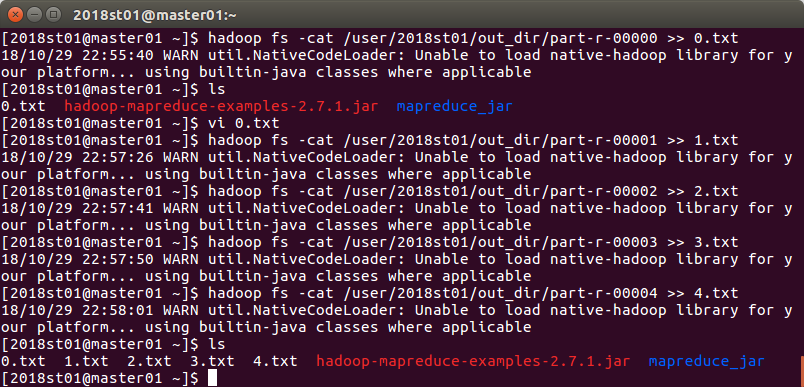
使用5个reduce节点，执行情况如下：



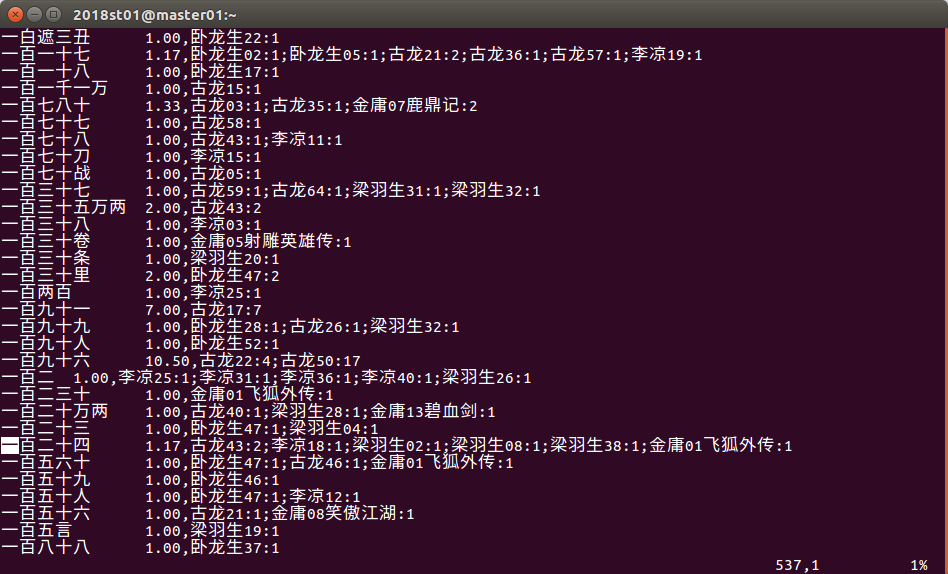


1. 输出结果文件的部分截图

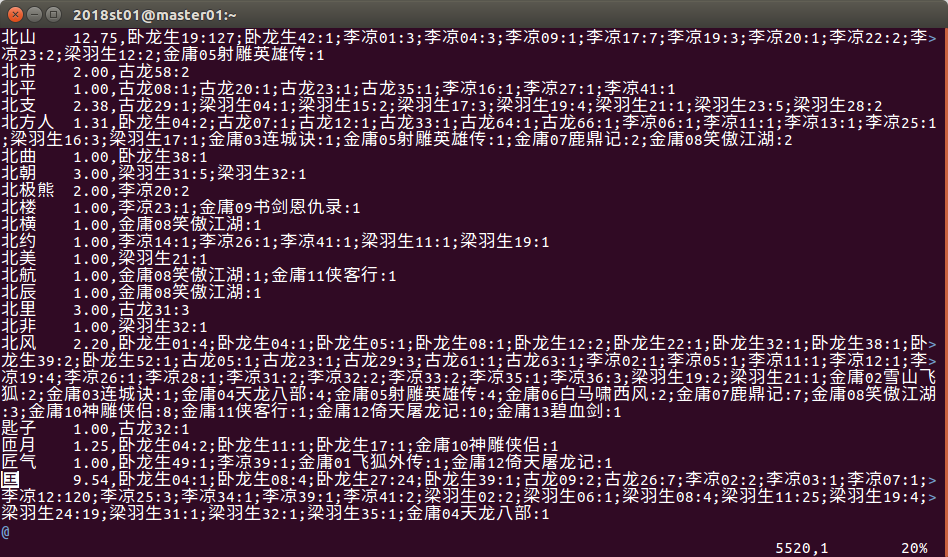
输出文件内容过多，为了便于查看，我把它们重定向到txt文件，这样也方便用vim进行搜索



查看part-00000部分截图:

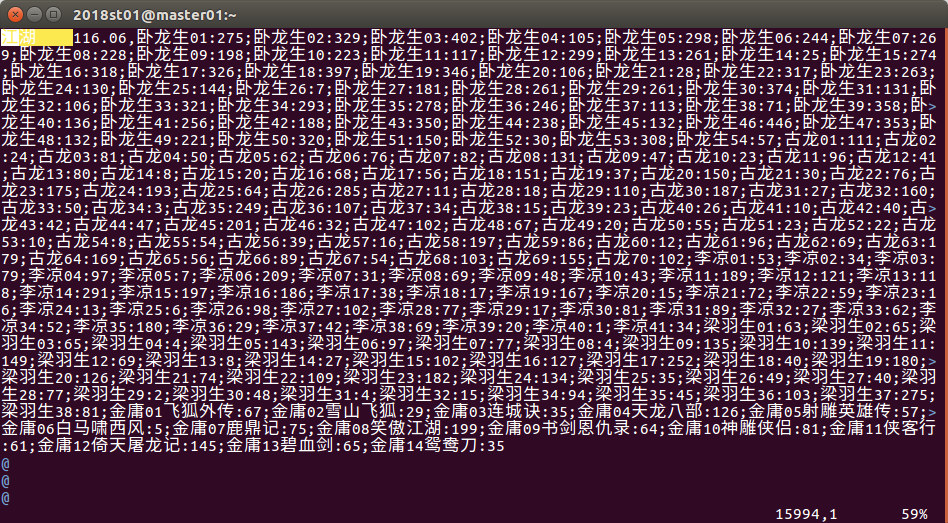


查看part-00002部分截图:

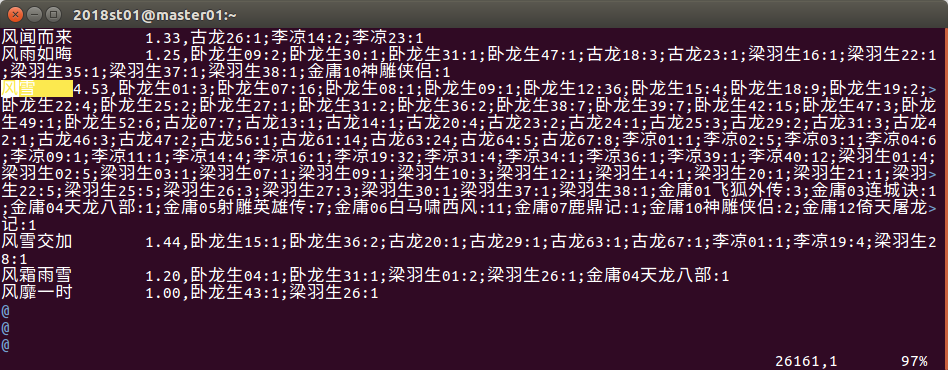


1. “江湖”、“风雪”两个单词的输出结果

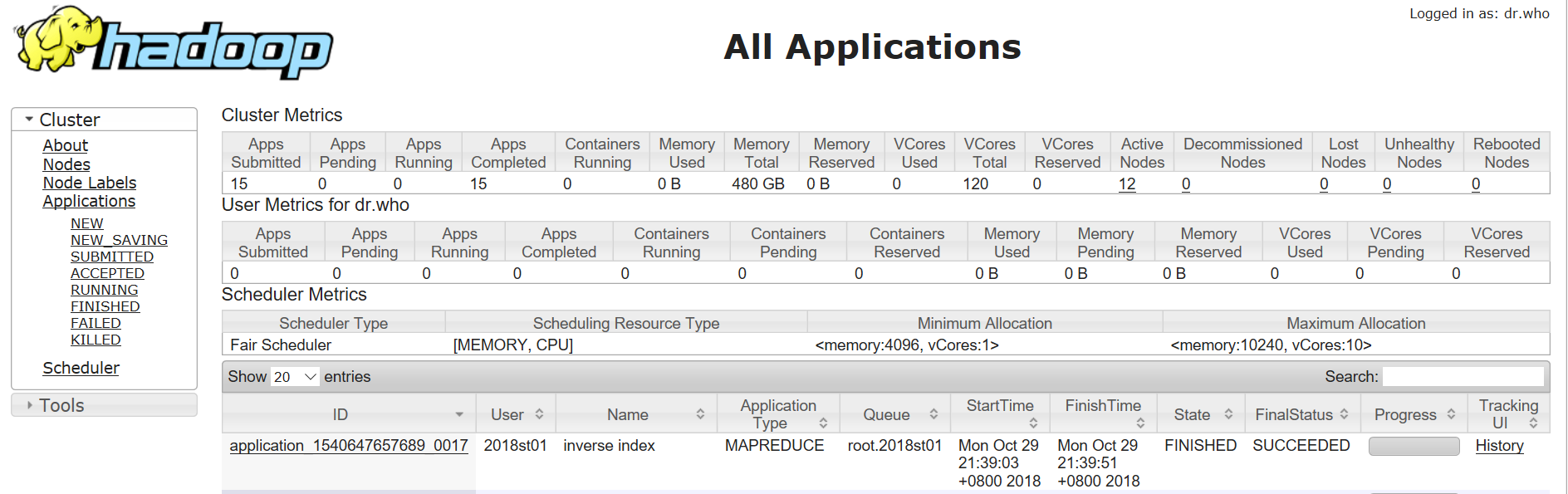
“江湖”:

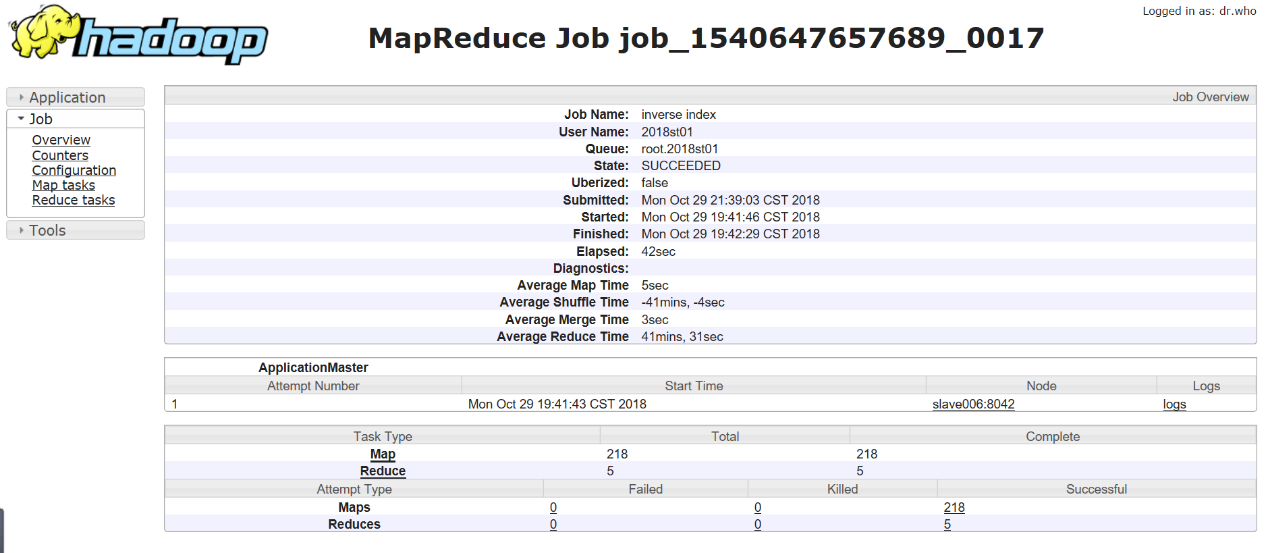


“风雪”:



1. WebUI执行报告截图





1. 实验总结

本次实验主要包括统计词语的倒排索引并输出带词频属性的倒排索引和计算每个词语的平均出现次数两个任务。实验的主要难点在于如何设计和实现Map和Reduce，这部分在课件上其实已经有非常详细的介绍了。实验结果达到了预期实验要求。通过本次实验我们提高了对MapReduce的算法设计与实现有了更加深刻认识，同时对Combiner和Partitioner的作用也有了进一步的理解。