

实验 2：文档倒排索引算法

第一部分：实验设计说明

本部分共包括设计思路、算法设计、程序和各个类的设计说明。

1. 设计思路

倒排索引算法中每个倒排索引结构由一个词 term 和含有这个 term 的相关的多个 posting 列表组成, 每个 posting 列表中包括单词 term 在一个具体的文档中出现的描述信息, 包括文档名以及在该文档中的词频。经过这一步我们就会得到类似下面的索引结构:

$\langle \text{term}, \langle \text{doc1}, \text{tf1} \rangle \cdots \langle \text{docn}, \text{tfn} \rangle \rangle$

注: 其中 doc*i* 代表文档 *i* 的文件名, tf*i* 代表 term 在文档 *i* 中的词频;

再计算: 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

对每个 term 在全部文档中出现的频数总和 = $\sum_{i=1}^n \text{tf}_i$

最后, 得到索引结构:

$\langle \text{term} \backslash \text{TAB 平均出现次数}, \text{doc1:tf1}; \text{doc2:tf2}; \dots; \text{docn:tfn} \rangle$

2. 算法设计

Map 阶段: 对每个文件每一行内容, 进行处理得到诸如以下的多个键值对: $\langle \text{term} \# \text{filename}, \text{num} \rangle$, 下面是 Map 阶段的伪代码:

Class Mapper

```

        //for a line
        for all term t in line do
            Key <- term + “#” + filename
            Emit(key, 1)
Class Combiner:
    //process same key
    sum <- 0
    for all value in same key
        sum += value
    Emit(key, sum)
Class Partitioner
    //shuffle key-value
    //for all key-value do
        newKey <- term //get term form key
    do Partitioner with newKey...

```

Reduce 阶段：经过 Map 阶段，相同的 term 都被发送到了同一个 Reduce 节点。根据输出的 key 信息，可以得到 term 在每个文档中出现的词频，进而可以计算出 term 的平均出现次数。

```

Class Reduce
    //for key-value→<term#doci, num>
    //calculate tf of term in doci
    tfi <- 0
    for all term in doci
        tfi += value;
    get <doci, tfi>
    //计算平均出现次数
    total <- 0
    for tfi of term in all doc
        total += tfi
    average <- total / n
    key <- term
    value<-average+ “,” +doci:tfi+ “;” +...+ “;” +docn:tfn
    Emit(key, value)

```

3. 程序以及各个类的设计说明

程序中包括以下类：

FileNameInputFormat：是一个自定义 FileInputFormat，将

作业数据分割成多个 InputSplit。

FileNameRecordReader：是一个自定义 RecordReader，将 InputSplit 处理转化为若干的输入记录；

InvertedIndexMapper：对 filename 进行处理得到诸如 <term#filename, 1>的多个键值对。

SumCombiner：在 Mapper 阶段会产生大量的相同主键的键值对，因此需要使用 Combiner 对 Mapper 部分的中间结果的词频进行累加，减少向 Reduce 节点传输的数据量。

NewPartitioner：由于 Key 值为 term+filename，若使用默认的 Partitioner 进行 shuffle 处理，同一个 term 可能会被发送到不同的 Reduce 节点。所以需要将原有的 key 处理，单使用 term 作为新的 key 进行 Partitioner，这样就可以保证所有相同的 term 被发送到同一个 Reduce 节点。

InvertedIndexReducer：自定义的 Reducer 类。对上一步的中间结果<term#filename, num>进行处理，得到 term 在每个文档中对应出现的词频，再计算 term 的平均出现次数。这里注意处理最后一个单词。得到：

Term average_times, doc1:tf1;...;docn:tfn

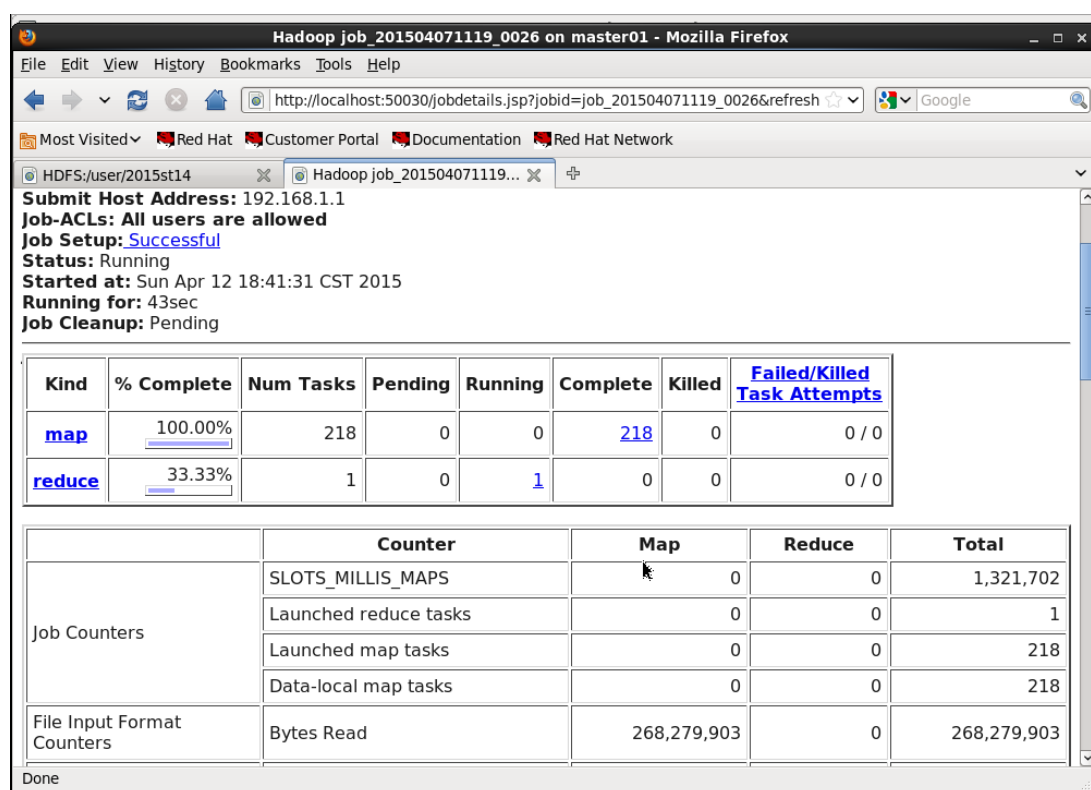
第二部分：运行与结果分析

本部分共包括实验运行和实验结果说明和分析。

1. 实验运行

在 eclipse 中打包程序 InvertedIndexer.jar

在集群中提交, 运行时 Jobtracker WebUI 截图:



运行成功截图:

master01:7 (2015st14)

Hadoop job_201504071119_0026 on master01 - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://localhost:50030/jobdetails.jsp?jobid=job_201504071119_0026&refresh

Most Visited Red Hat Customer Portal Documentation Red Hat Network

HDFS:/user/2015st14 Hadoop job_201504071119...

Job Name: inverted index
Job File: https://master01:54310/user/hadoop/mapred/staging/2015st14/staging/job_201504071119_0026/job.xml
Submit Host: master01
Submit Host Address: 192.168.1.1
Job-ACLs: All users are allowed
Job Setup: [Successful](#)
Status: Succeeded
Started at: Sun Apr 12 18:41:31 CST 2015
Finished at: Sun Apr 12 18:42:47 CST 2015
Finished in: 1mins, 16sec
Job Cleanup: [Successful](#)

Kind	% Complete	Num Tasks	Pending	Running	Complete	Killed	Failed/Killed Task Attempts
map	100.00%	218	0	0	218	0	0 / 0
reduce	100.00%	1	0	0	1	0	0 / 0

	Counter	Map	Reduce	Total
	SLOTS_MILLIS_MAPS	0	0	1,325,616
	Launched reduce tasks	0	0	1
	Total time spent by all reduces waiting after reserving slots (ms)	0	0	0

2. 实验结果说明和分析

实验结果截图：

Most Visited Red Hat Customer Portal Documentation Red Hat Network

HDFS:/user/2015st14/exp2... master01 Hadoop Map/Red...

File: [/user/2015st14/exp2_ywj_out/part-r-00000](#)

Goto :

[Go back to dir listing](#)
[Advanced view/download options](#)
[View Next chunk](#)

1993年	1.0, 梁羽生21. 梁羽生传奇:1;
19世纪	3.0, 梁羽生31. 随笔集: 笔不花:3;
19岁	2.0, 梁羽生31. 随笔集: 笔不花:2;
1+	1.0, 古龙45. 名剑风流:1;
1月	2.0, 梁羽生21. 梁羽生传奇:2; 梁羽生31. 随笔集: 笔不花:2;
2	2.84375, 卧龙生83. 碧血玉环:1; 卧龙生22. 女捕头:1; 卧龙生24. 七绝剑:2; 卧龙生26. 情剑无刃:1; 卧龙生33. 天剑绝刀:1; 卧龙生37. 天涯情侠:1; 卧龙生42. 新仙鹤神针:2; 卧龙生47. 一代天骄:1; 卧龙生50. 玉钗盟:2; 卧龙生51. 玉手点将录:1; 卧龙生54. 指剑为媒:1; 李凉09. 红顶记:1; 李凉14. 江湖双响炮:3; 李凉16. 江湖一品郎:1; 李凉17. 惊神关小刀:2; 李凉20. 狂侠南官属:1; 李凉22. 矛盾天师:5; 李凉25. 魔手邪怪:1; 李凉26. 奇神劫小邪侠集:1; 李凉29. 神偷小千:1; 李凉30. 剑气世家:22; 李凉33. 天齐大帝:1; 李凉34. 天下第一当:2; 李凉36. 小鬼大当家:2; 梁羽生15. 慧剑心魔:1; 梁羽生31. 随笔集: 笔不花:1; 梁羽生34. 武当一剑:1; 梁羽生35. 武林天骄:1; 金庸04. 天龙八部:3; 金庸05. 射雕英雄传:1; 金庸07. 鹿鼎记:1;
20	2.0, 李凉26. 奇神劫小邪侠集:1; 梁羽生31. 随笔集: 笔不花:3;
2000	1.0, 梁羽生31. 随笔集: 笔不花:1;
2005-10-19	4.0, 梁羽生31. 随笔集: 笔不花:3; 梁羽生32. 随笔集: 笔花六照:5;
2005-11-4	1.0, 梁羽生31. 随笔集: 笔不花:1;
2005-6-15	1.0, 梁羽生32. 随笔集: 笔花六照:1;
2005-8-12	5.0, 梁羽生32. 随笔集: 笔花六照:5;
2005-8-13	3.5, 梁羽生31. 随笔集: 笔不花:2; 梁羽生32. 随笔集: 笔花六照:5;

Find: ☐ Match case

实验结果中索引结构 term\t 平均出现次数, doc1:tf1;...,

对词语 1 月进行分析,发现其在两个文档中分别出现了两次,

故其平均出现次数为 2.0.

第三部分：不足与改进

本部分共包括实验中性能扩展性等方面的可能存在的不足和可能的可进。

1. 可能存在的不足

实验结果中包含了较多的停用词信息，比如数字等。

2. 可能的改进

对停用词进行处理。

附件：

附件 1：源程序 InvertedIndexer.java

附件 2：运行程序 InvertedIndexer.jar

附件 3：结果文件 exp2_2015st14_out