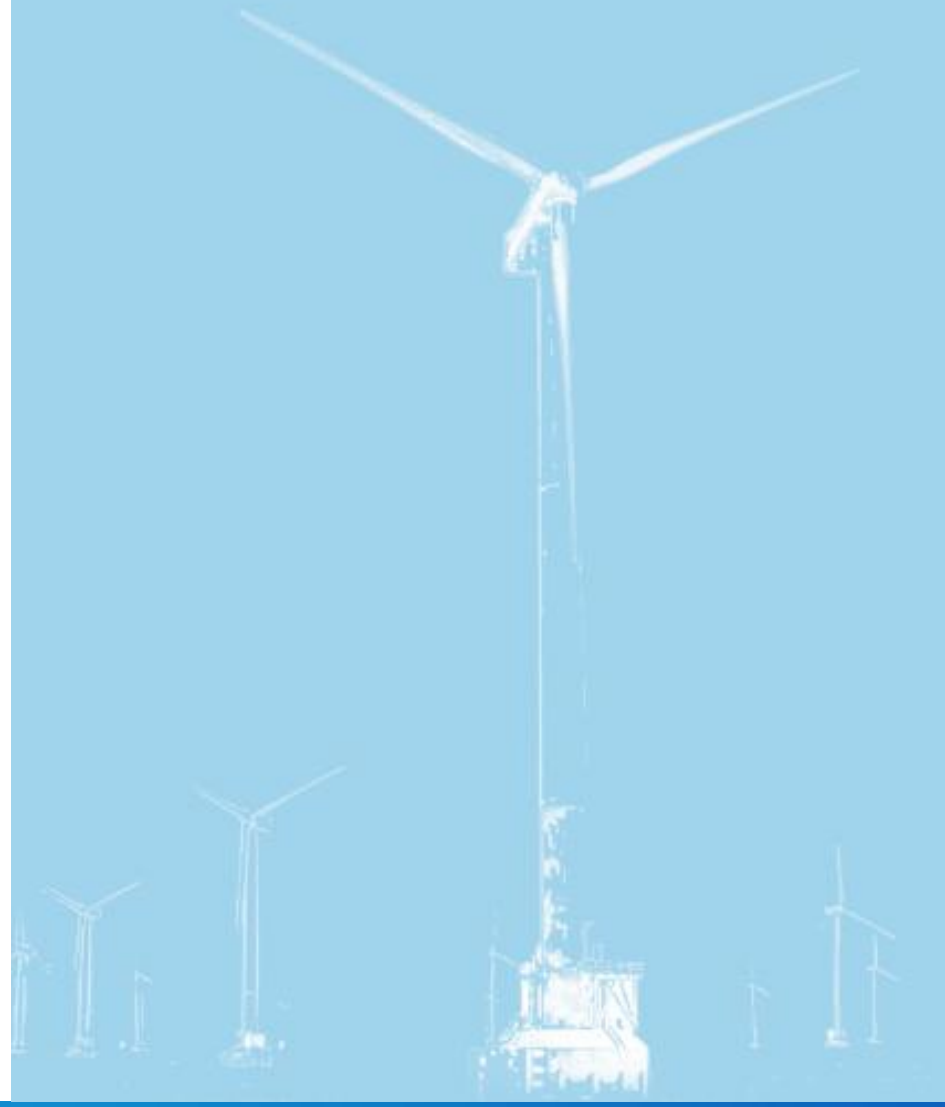
Hadoop分布式计算框架MapReduce

Hadoop离线计算— 杜黎明

目录

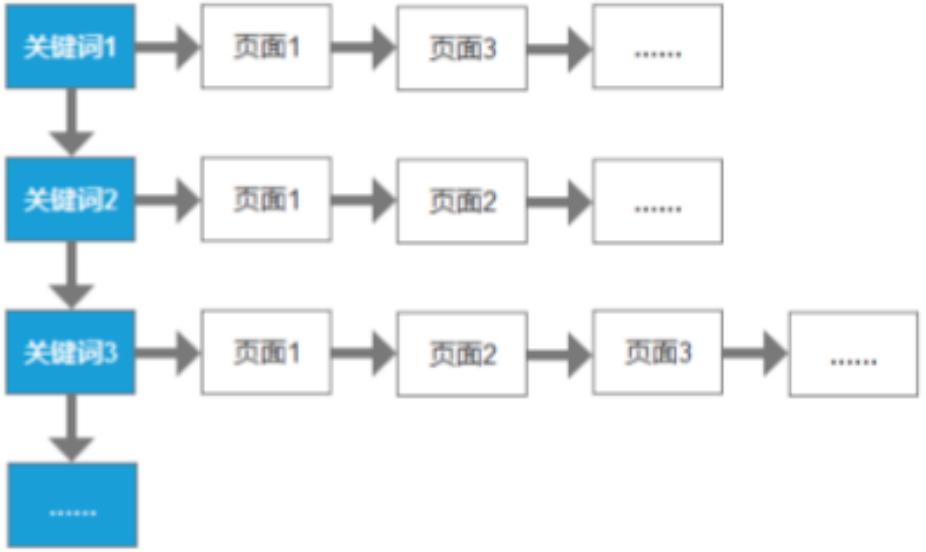


实现简单的倒排索引

需求：对于给定的文档，确定每个单词存在于某个文档，同时在文档中出现的次数

正向索引（forward index）从逻辑上讲其实就是一个大数组，数组中每个元素就是一个文档的属性集合

倒排索引（inverted index）中的每一项都包括一个属性值和具有该属性值的各记录的地址。它不是由记录来确定属性值，而是由属性值来确定记录的位置



实现简单的倒排索引

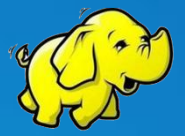
倒排索引简单的可以理解为全文检索某个词
例如：在a.txt 和b.txt两篇文章分别中查找统计hello这个单词出现的次数，出现次数越多，和关键词的吻合度就越高

解题思路：

Map端对文件统计每个单词出现的次数，输出类似<{hadoop,a.txt},2>
Map端输出前要先进行Combine过程，最终输出类似< hadoop, a.txt:2>
Reduce端继续对相同单词进行合并，最终输出类似<hadoop, a.txt:2 b.txt:5>

在hadoop平台上编写mr代码分析统计各个单词在两个文本中出现的次数
其实也只是WordCount程序的改版而已~

MapReduce应用案例-网站kpi数据统计



1.browser：用户使用的浏览器统计

2.ips：页面用户独立ip数统计

3.pv：网站pv量统计

4.source：用户来源网址统计

5.time：时间段用户访问量统计

```
|---+----1----+----2----+----3----+----4----+----5----+----6----+----7----+----8----+----9----+----0----+----1----+----2----+----3----+----4----+----5----+----6----+--
1  ip地址  客户端用户名    请求时间        请求方法        请求页面        http协议信息    返回的状态码    发送的页面字节数        从什么页面跳转进来      用户使用的客户端
2  120.197.87.216 - - [04/Jan/2012:00:00:02 +0800] "GET /home.php?mod=space&uid=563413&mobile=yes HTTP/1.1" 200 3388 "-" "-"
3  123.126.50.73 - - [04/Jan/2012:00:00:02 +0800] "GET /thread-679411-1-1.html HTTP/1.1" 200 5251 "-" "Sogou web spider/4.0(+http://www.sogou.com/docs/help/webmasters.htm
4  116.205.130.2 - - [04/Jan/2012:00:00:02 +0800] "GET /popwin_js.php?fid=6 HTTP/1.1" 200 32 "http://www.itpub.net/forum-6-1.html?ts=28" "Mozilla/4.0 (compatible; MSIE 8.
5  218.186.15.10 - - [04/Jan/2012:00:00:16 +0800] "GET /forum.php?mod=ajax&action=forumchecknew&fid=61&time=1325606260&inajax=yes HTTP/1.1" 200 92 "http://www.itpub.net/f
```

MapReduce应用案例-电信运营商用户基站停留数据统计

原始数据分为位置和网络两种

位置数据格式为：

用户标识 设备标识 开关机信息 基站位置 通讯的时间

example:

0000009999 0054785806 3 00000089 2016-02-21 21:55:37

网络数据格式为：

用户标识 设备标识 基站位置 通讯的时间 访问的URL

example:

0000000999 0054776806 00000109 2016-02-21 23:35:18 www.baidu.com

需要得到的数据格式为：

用户标识 时段 基站位置 停留时间

example:

00001 09-18 00003 15

用户00001在09-18点这个时间段在基站00003停留了15分钟

两个reducer：

1.统计每个用户在不同时段中各个基站的停留时间

2.在1的结果上只保留停留时间最长的基站位置信息

MapReduce应用案例-电信运营商用户基站停留数据统计

Hadoop离线计算

位置数据

IMSI	IMEI	UPDATETYPE	LOC	TIME
…				
A	001	0	X基站	2013-09-12 09:00:00
A	001	2	Y基站	2013-09-12 09:45:00
…				

两种数据最大的差别在于文件名

数据文件名以POS开头

数据文件名以NET开头

上网数据

IMSI	IMEI	LOC	TIME	URL
…				
A	001	X基站	2013-09-12 09:15:00	www.baidu.com
A	001	Y基站	2013-09-12 09:30:00	www.google.com
…				

MapReduce应用案例-电信运营商用户基站停留数据统计

认为用户在任何时间的停留位置都取决于之前一次位置更新的基站位置

时间间隔超过超过60分钟的判定为关机

IMSI	IMEI	UPDATETYPE	LOC	TIME
...				
A	001	0	X基站	2013-09-12 09:00:00
A	001	2	Y基站	2013-09-12 09:45:00
...				

IMSI	IMEI	LOC	TIME	URL
...				
A	001	X基站	2013-09-12 09:15:00	www.baidu.com
A	001	Y基站	2013-09-12 09:30:00	www.google.com
...				

用户A在X基站

停留了30分钟

MapReduce应用案例-电信运营商用户基站停留数据统计

输入数据

IMSI	IMEI	UPDATETYPE	LOC	TIME
...				
A	001	0	X基站	2013-09-12 09:00:00
A	001	2	Y基站	2013-09-12 09:45:00
...				

IMSI	IMEI	LOC	TIME	URL
...				
A	001	X基站	2013-09-12 09:15:00	www.baidu.com
A	001	Y基站	2013-09-12 09:30:00	www.google.com
...				

根据文件名提取字段

IMSI	LOC	TIME
...		
A	X基站	2013-09-12 09:00:00
A	Y基站	2013-09-12 09:45:00
...		

IMSI	LOC	TIME
...		
A	X基站	2013-09-12 09:15:00
A	Y基站	2013-09-12 09:30:00
...		

IMSI	LOC	TimeFlag	TIME
...			
A	X基站	09-17	1386579600
A	Y基站	09-17	1386582300
A	X基站	09-17	1386580500
A	Y基站	09-17	1386581400
...			

计算时间所属时间段
把日期转换为UNIX格式

MapReduce应用案例-电信运营商用户基站停留数据统计

IMSI	TimeFlag
...	
A	09-17
A	09-17
A	09-17
A	09-17
...	

LOC	TIME
X基站	1386579600
Y基站	1386582300
X基站	1386580500
Y基站	1386581400

Map

输出

IMSI	LOC	TIME
...		
A	X基站	2013-09-12 09:15:00
A	Y基站	2013-09-12 09:45:00
...		

IMSI	LOC	TIME
...		
A	X基站	2013-09-12 11:15:00
A	Y基站	2013-09-12 09:30:00
...		

以IMSI和TimeFlag作为Key

以LOC和TIME作为VALUE

IMSI	LOC	TimeFlag	TIME
...			
A	X基站	09-17	1386579600
A	Y基站	09-17	1386582300
A	X基站	09-17	1386580500
A	Y基站	09-17	1386581400
...			

1. 计算时间所属时间段
2. 把日期转换为UNIX格式

MapReduce应用案例-电信运营商用户基站停留数据统计

Hadoop离线计算

IMSI	TimeFlag
…	
A	09-17
A	09-17
A	09-17
A	09-17
…	

LOC	TIME
X基站	1386579600
Y基站	1386582300
X基站	1386580500
Y基站	1386581400

IMSI	TimeFlag
…	
其他	其他
…	

LOC	TIME
……	……

IMSI	TimeFlag
…	
A	09-17
…	

LOC	TIME
X基站	1386579600
Y基站	1386582300
X基站	1386580500
Y基站	1386581400

具有相同Key的数据发送到同一Reducer

IMSI	TimeFlag
…	
A	09-17
…	

LOC	TIME
X基站	1386579600
X基站	1386580500
Y基站	1386581400
Y基站	1386582300

对时间排序

MapReduce应用案例-电信运营商用户基站停留数据统计

Hadoop离线计算

IMSI	TimeFlag
...	
A	09-17
...	

LOC	TIME
X基站	1386579600
X基站	1386580500
Y基站	1386581400
Y基站	1386582300
OFF基站	1386608400

计算停留时间

IMSI	TimeFlag
...	
A	09-17
...	

LOC	STAY_TIME
X基站	15分钟
X基站	15分钟
Y基站	15分钟
Y基站	435分钟

在最后添加特殊基站
时间是这个时段的结束时间

IMSI	TimeFlag
...	
A	09-17
...	

LOC	TIME
X基站	1386579600
X基站	1386580500
Y基站	1386581400
Y基站	1386582300

MapReduce应用案例-电信运营商用户基站停留数据统计

Hadoop离线计算

IMSI	TimeFlag
...	
A	09-17
...	

LOC	TIME
X基站	1386579600
X基站	1386580500
Y基站	1386581400
Y基站	1386582300
OFF基站	1386608400

计算停留时间

IMSI	TimeFlag
...	
A	09-17
...	

LOC	STAY_TIME
X基站	15分钟
X基站	15分钟
Y基站	15分钟
Y基站	435分钟

删除大于60分钟的
汇总统计各个基站

IMSI	TimeFlag
...	
A	09-17
...	

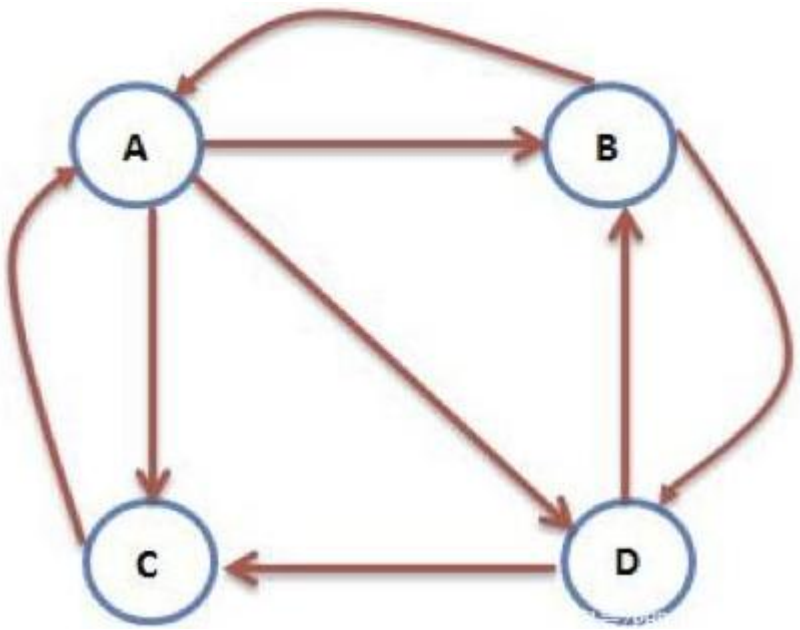
LOC	STAY_TIME
X基站	30分钟
Y基站	15分钟

输出数据

MapReduce应用案例-PageRank算法

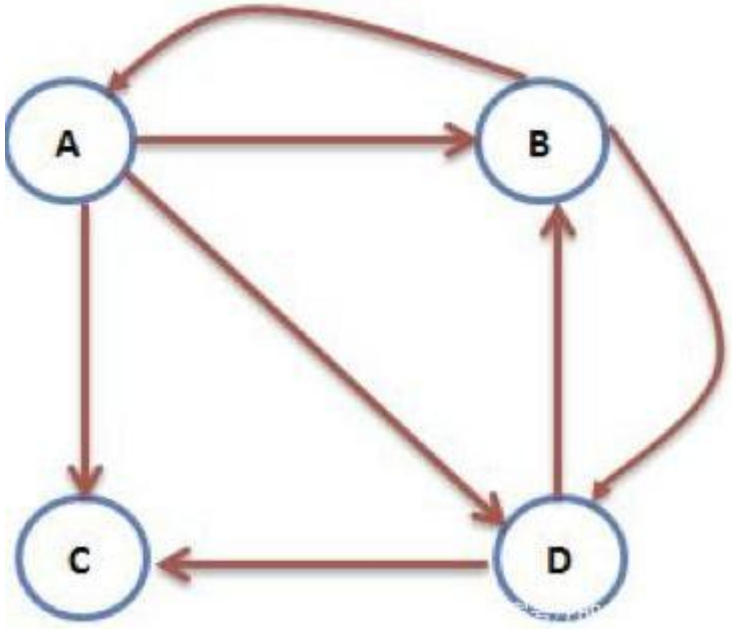
- PageRank是Google专有的算法，用于衡量特定网页相对于搜索引擎索引中的其他网页而言的重要程度。它由Larry Page 和 Sergey Brin在20世纪90年代后期发明。PageRank实现了将链接价值概念PageRank是Google的核心算法，用于给每个网页做评分，是google在“垃圾中找黄金”的关键算法，这个算法成就了今天的google。
- 作为排名因素。
- PageRank有两大特性：
  - ✓ PR值的传递性：网页A指向网页B时，A的PR值也部分传递给B
  - ✓ 重要性的传递性：一个重要网页比一个不重要网页传递的权重要多

MapReduce应用案例-PageRank算法



A网页的pagerank值由网页B和网页C的pagerank贡献而来，因为B网页有两个外链，假设等概率贡献，则贡献给A的值为自身的一半。

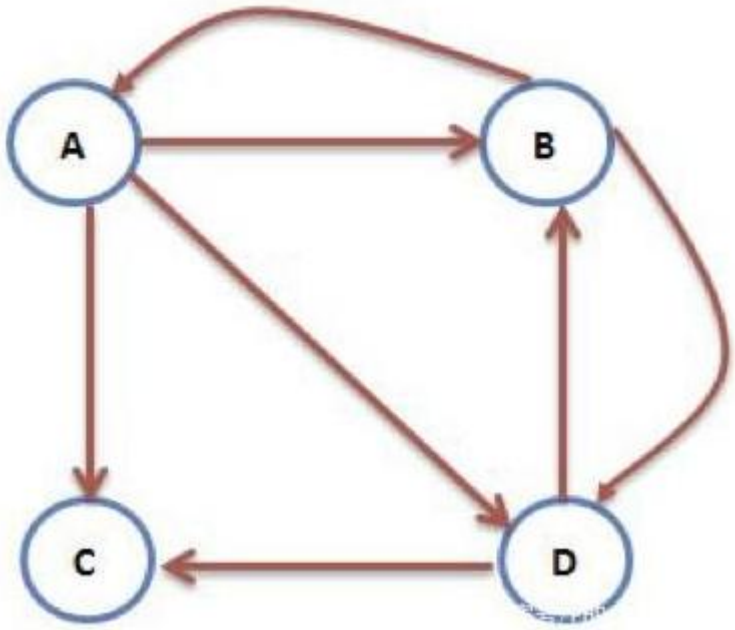
pg（A） = pg（C）/1 + pg(B) /2



此处C因为没有外链，所以我们假设他给其他所有网页都贡献了pg，于是：

pg(A) = pg（C）/4 + pg(B) /2

MapReduce应用案例-PageRank算法



用户在浏览网页的过程中，直接输入新网址进行浏览，即一个网页都有可能跳转到任意其他网页，于是产生如下公式：**pg(A) = （a * pg(B) /2） + (1-a) / 4)**

即：在任意时刻，用户到达某页面后并继续向后浏览的概率为**a**，则用户停止浏览的概率为**(1-a)**，此时用户停止浏览后，可能会直接通过输入浏览器地址进行浏览网页，此时跳转到任意网址的概率都一样，于是上面的 a * pg(B) /2 表示从b跳转过来的概率，(1-a) / 4 表示直接输入网址跳转过来的概率。

- 计算公式：

$$PR(p_i) = \frac{1-d}{n} + d \sum_{p_j \in M(i)} \frac{PR(p_j)}{L(j)}$$

PR(pi): pi页面的PageRank值

n: 所有页面的数量

pi: 不同的网页p1,p2,p3

M(i): pi链入网页的集合

L(j): pj链出网页的数量

d:阻尼系数, 任意时刻，用户到达某页面后并继续向后浏览的概率。(1-d=0.15) :表示用户停止点击，随机跳到新URL的概率取值范围: 0 < d ≤ 1, Google设为0.85

MapReduce应用案例-PageRank算法

将pageRank的公式转换为矩阵标识：

$$\mathbf{R} = \begin{bmatrix} (1-d)/N \\ (1-d)/N \\ \vdots \\ (1-d)/N \end{bmatrix} + d \begin{bmatrix} \ell(p_1,p_1) & \ell(p_1,p_2) & \cdots & \ell(p_1,p_N) \\ \ell(p_2,p_1) & \ddots & & \\ \vdots & & \ell(p_i,p_j) & \\ \ell(p_N,p_1) & & & \ell(p_N,p_N) \end{bmatrix} \mathbf{R}$$

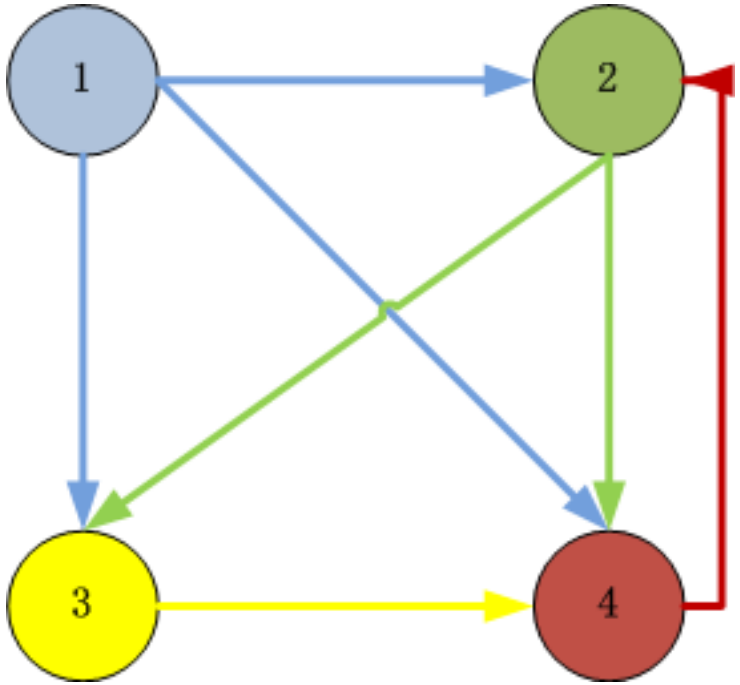
如果p_j不链向p_i，而且对每个j都成立时，$\ell(p_i,p_j)$等于0

$$\sum_{i=1}^{N} \ell(p_i,p_j) = 1,$$

l(pi,pj)表示网页j指向网页i,其值为:

l(pi,pj) = 1 / L(pj)

$$PR(p_i) = \frac{1-d}{n} + d \sum_{p_j \in M(i)} \frac{PR(p_j)}{L(j)}$$



链接源页面	链接目标页面
1	2,3,4
2	3,4
3	4
4	2

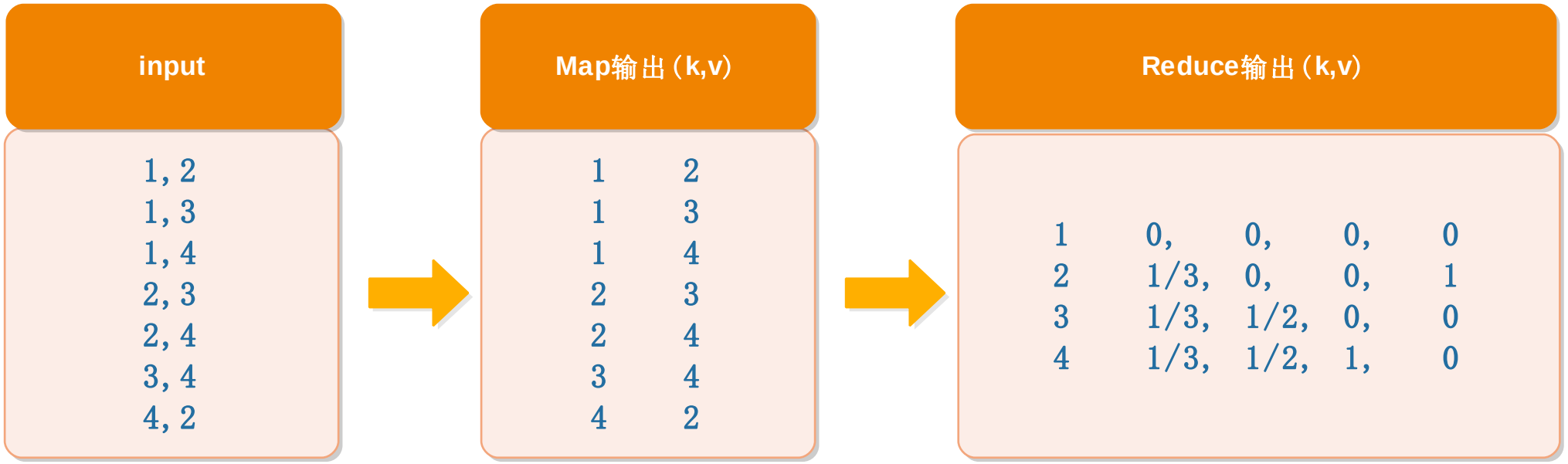
邻接矩阵：列=源链接，行=目标链接

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 1/3 & 0 & 0 & 1 \\ 1/3 & 1/2 & 0 & 0 \\ 1/3 & 1/2 & 1 & 0 \end{bmatrix}$$

概率矩阵：列=源链接，行=目标链接

- 生成邻接概率矩阵。


input
1, 2
1, 3
1, 4
2, 3
2, 4
3, 4
4, 2
Map输出(k,v)
1 2
1 3
1 4
2 3
2 4
3 4
4 2
Reduce输出(k,v)
1 0, 0, 0, 0
2 1/3, 0, 0, 1
3 1/3, 1/2, 0, 0
4 1/3, 1/2, 1, 0

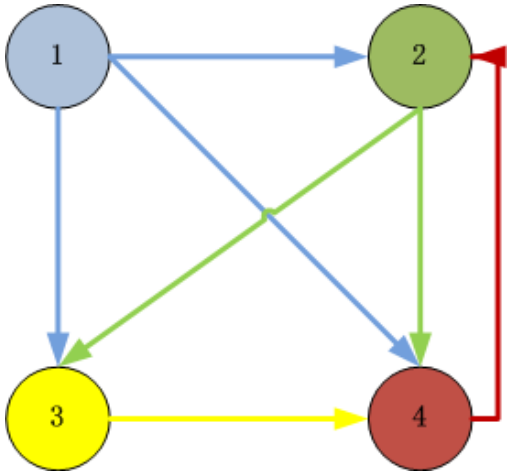
假设每个链接初始PageRank值为1。

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 1/3 & 0 & 0 & 1 \\ 1/3 & 1/2 & 0 & 0 \\ 1/3 & 1/2 & 1 & 0 \end{bmatrix} \times \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0*1+0*1+0*1 \\ \frac{1}{3}*1+0*1+0*1+1*1 \\ \frac{1}{3}*1+\frac{1}{2}*1+0*1+0*1 \\ \frac{1}{3}*1+\frac{1}{2}*1+1*1+0*1 \end{bmatrix} = \begin{bmatrix} 0 \\ 4/3 \\ 5/6 \\ 11/6 \end{bmatrix}$$

概率矩阵　　初始PR矩阵　　第一轮PR值

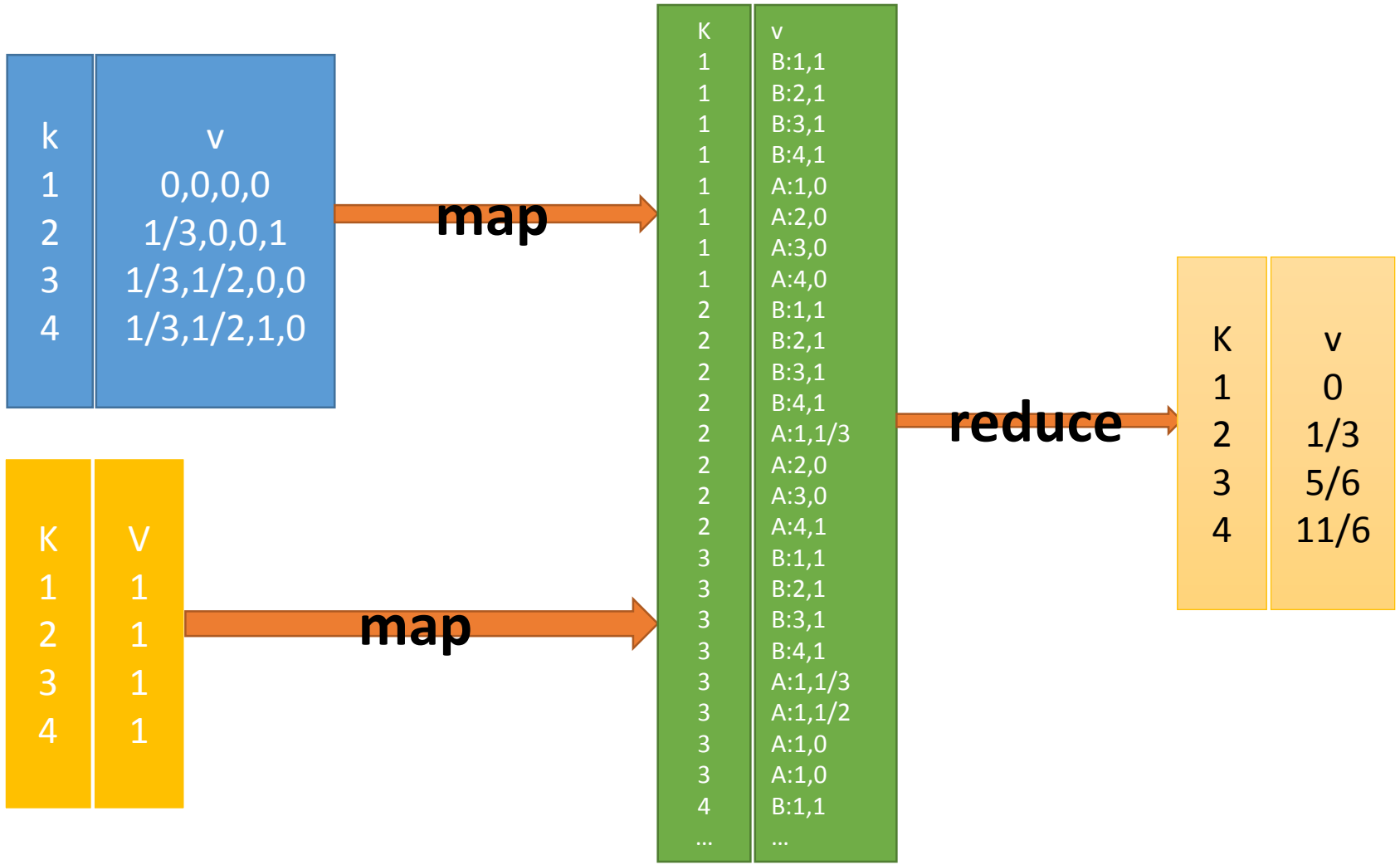
$$\sum_{p_j \in M(i)} \frac{PR(p_j)}{L(j)}$$

$$PR(p_i) = \frac{1-d}{n} + d \sum_{p_j \in M(i)} \frac{PR(p_j)}{L(j)}$$



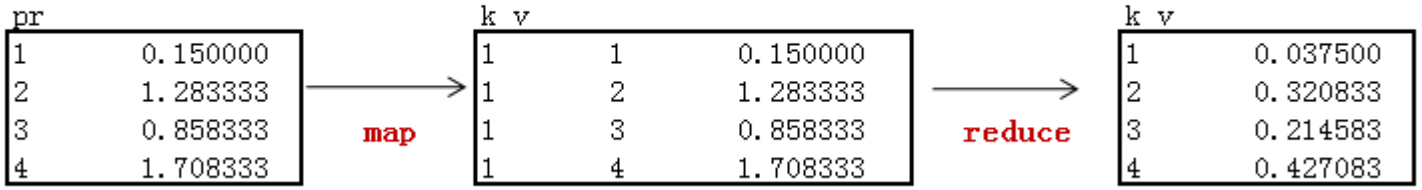
MapReduce应用案例-PageRank算法

- 实现邻接与PR矩阵的乘法



- 对PR的计算结果标准化，让所以PR值落在(0,1)区间

Normal



作业

1.用MapReduce实现PageRank算法（提供算法程序和运行结果的截图）。

2.查找Mapreduce关于计数器相关的资料，mapreduce输出结果中这些计数器都代表什么含义？如何自定义计数器？

3.自己动手实现今天所讲的案例代码，并运行处结果。

4.描述一下Mapreduce的shuffle过程。

5.描述几条你所知道的关于Mapreduce的优化策略。

作业

提供了1号风场的2015~2016年的10min数据，计算每台风机，每月份的风速功率曲线。

计算步骤详细介绍如下：

1.数据预处理

- 剔除时间为空的数据；
- 剔除时间范围超时2015和2016年的数据；
- 剔除时间重复的数据；
- 剔除超出量程的风速数据；（0~50）
- 剔除风速值、功率值为-902的数据；

作业

提供了1号风场的2015~2016年的10min数据，计算每台风机，每月份的风速功率曲线。

计算步骤详细介绍如下：

2.数据质量统计

- 统计缺失数据数量(两条连续数据之间的时间间隔为10min，如果大于10min则表示有缺失值)；

作业

提供了1号风场的2015~2016年的10min数据，计算每台风机，每月份的风速功率曲线。

计算步骤详细介绍如下：

3.功率曲线计算

- 数据按照分数划分bin区间(bin区间的步长为0.5，最后将数据按照风速分类为如下bin区间的数据：<0,0.25>,<0.26,0.75>,<0.76,1.25>,<1.26,1.75>… …)。

例如：有4条数据的风速值分别为：0.26,0.50,0.74,1.24；则：

0.26~0.75区间的数据：0.26,0.50,0.74这三条数据

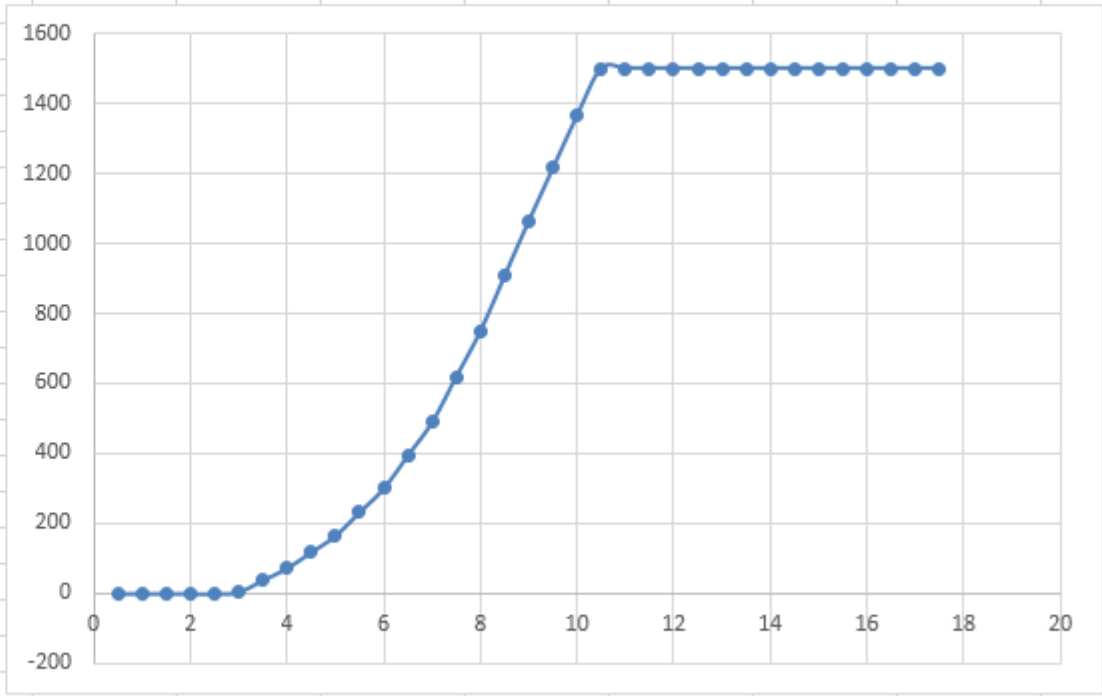
0.76~1.25区间的数据：1.24这条数据

- 统计每个bin区间内的平均风速，平均功率。

3.输出结果示例：

风机编号_年月	平均	平均功率值
WT2209_201501	0	0
WT2209_201501	0.5	0
WT2209_201501	1	0
WT2209_201501	1.5	0
WT2209_201501	2	0
WT2209_201501	2.5	0
WT2209_201501	3	4.466
WT2209_201501	3.5	38.313
WT2209_201501	4	72.16
WT2209_201501	4.5	118.415
WT2209_201501	5	164.67
WT2209_201501	5.5	232.345
WT2209_201501	6	300.02
WT2209_201501	6.5	394.66
WT2209_201501	7	489.3
WT2209_201501	7.5	619.6
WT2209_201501	8	749.9
WT2209_201501	8.5	907.1
WT2209_201501	9	1064.3
WT2209_201501	9.5	1215.1
WT2209_201501	10	1365.9
WT2209_201501	10.5	1500.6
WT2209_201501	11	1500.6
WT2209_201501	11.5	1500.6
WT2209_201501	12	1500.6

输出数据包括：风机编号_年月，风速，平均功率值



THANKS

致敬每一个看似微渺的小梦想！

致敬每一位正在路上的追梦人！