# 基于 Hadoop 的用户搜索行为分析①

宋芳琴

(绍兴职业技术学院, 绍兴 312000)

**摘 要**: 用户搜索网页行为的分析是目前信息搜索的研究的热点, 本文针对云计算中的并行计算搜索存在的检索速度慢, 效率低等缺点提出了一种基于 Hadoop 海量用户搜索网页行为的方法, 该方法主要是在网页 PageRank 算法的基础上, 将用户影响因子, 时间向量和网页相关性因素加入到算法中, 使得改进后的 PageRank 算法得到了提高, 进一步提高用户搜索网页行为的效率, 实验中通过使用优酷实验室中的查询日志分析证明了本文的算法具有良好的效果, 并对云计算中的用户行为分析具有一定的指导意义.

**关键词**: Hadoop 用户搜索 行为分析 海量日志 PageRank 算法

## Analyzing Users' Searching Behavior Based on Hadoop

SONG Fang-Qin

(Shaoxing Vocational & Technical College, Shaoxing 312000, China)

**Abstract**: The analysis of users' behavior of searching Webpages is the hotspot of current information searching. This paper focus on the weakness in the parallel calculation search of cloud calculation, like slow research speed, low efficiency and so on, a method based on Hadoop for mass users to search Web-pages is proposed, in which users' impact factors, time vector and Web-related factors are added to the algorithm based on the PageRank algorithm so as to further improve the efficiency for users in searching Web-pages. Analysis of query log in Youku laboratory is used in the experiment to prove algorithm in this paper has good effect as well as some guiding significance for users' behavior analysis in cloud computing.

**Key words**:Hadoop; user searching; behavior analysis; massive log; PageRank algorithm

伴随着云计算概念的出现, 越来越多的信息通过互联网进行共享和传播, 网络信息膨胀速度已经呈现指数级增长. 在此背景下, 云计算下的搜索引擎快速发展成为了人们获得信息的重要手段. 目前, 美国的斯坦福大学提出了 PageRank[1], IBM 提出了 HITS 技术[2,3]. 其次, 在 Web 查询模式下产生了一些新的可用信息, 这些可用的信息从另一个侧面反应了用户搜索的某些行为, 从而能够帮助云计算服务器来分析用户信息的搜索质量, 对用户的行为进行分析. 大型搜索引擎访问按日来计算访问量可以达到亿级, 记录的用户查询日志文件对象都是海量文件. 基于这些海量日志的特点, 传统的数据存储和计算方法已经难以适应搜索引擎用户行为分析, 因此针对这个问题, 本文在 Hadoop 架构下, 对海量网页信息进行搜索, 在 PageRank 算法的基础上, 将用户影响因子, 时间向量和网页相关性因素加入到算法中, 使得改进后的 PageRank 算法得到了提高, 有效的提高了用户搜索行为的效率, 仿真实验表明本文的分析具有一定的优越性.

## 1 Hadoop架构和Hive

Hadoop 架构是一种开源组织分布式的计算框架, 并且在大型企业中使用的非常广泛[4], 该架构中主控节点(JobTracker)将云计算模型中的 Map 任务和 Reduce

① 基金项目:浙江省高等学校访问工程师校企合作项目
收稿时间:2015-04-02;收到修改稿时间:2015-05-07

任务发给空闲的计算节点(TaskTracker)执行并进行监控运行任务, 因此, Hadoop 架构具有成本低, 效率高和可靠性高的特点. HDFS 是一种采用 Master/Slave 架构, 组成由一个管理节点 NameNode 和若干个数据节点 DataNode 构成[5], 其作用主要是把大型的文件分割成若干个的 Block 小块分散存储在不同的 DataNode 上, 其中每一个 Block 将数据复制到不同的数据节点上, 从而可以具有容错的功能. Hive[6]是一个基于 hadoop 的数据仓库的基础架构. 它主要用来处理 HDFS 中的数据机制,是一种可以对海量数据进行提取转化加载的框架. 主要是通过 MapReduce 框架来进行实现的, 因此 Hive 架构就具有了高效的处理海量数据, 保证了生成的 MapReduce 任务是高效的. 在实际的云计算应用中, Hive 可以高效的对 TB 甚至 PB 级的数据进行处理.

## 2 用户搜索相关工作

### 2.1 各索引查询的权重

本文通过以查询文档中的词权重值作为单元项, 将单元项组成的项集与查询序列重点词的权重一起作为单元项组成的集合. 定义 $X_{i,j}$ 表示经过搜索引擎进行分词后的权重, $Y_{i,j}$ 表示查询序列中的某个分词的权重. 因此得到如下结果:

$$f(d_j,t)=\frac{\vec{d_j}\square\vec{t}}{|\vec{d_j}|\times|\vec{t}|}=\frac{\sum_{i=1}^{t}X_{i,j}\times Y_{i,j}}{\sqrt{\sum_{i=1}^{t}X_{i,j}^2}\times\sqrt{\sum_{i=1}^{t}Y_{i,j}^2}} \tag{1}$$

$$X_{i,j}=\frac{fre_{i,j}}{\max_i fre_{i,j}}\times\log\frac{N}{n_i} \tag{2}$$

其中, $\vec{d_j}=(X_{1,m},X_{2,m},\cdots X_{i,m})$, $\vec{t}=(X_{1,n},X_{2,n},\cdots X_{i,n})$. $fre_{i,j}$ 表示第 $i$ 个关键词在第 $j$ 个网页文档中频率, $\max_i fre_{i,j}$ 表示总频率, $\log\frac{N}{n_i}$ 表示逆文档频率指数, $N$ 表示全部网页中资源的数目, $n_i$ 表示第 $i$ 个关键词出现的网页文档的总数.

### 2.2 搜索模型建立

本文建立一个云条件下的搜索模型, 整个模型主要是一个网站, 网站中的网页链接关系主要是基于 Internet 的结构, 用户搜素过程中使用了正确的搜索词, 网页搜索结果基于用户的感兴趣的程度, 搜索日志的数据具有真实性.

定义 1. 设定网页的数目为 $N$, 使用 $A_1,A_2\cdots A_n$ 表示 $n$ 个页面, 则矩阵 $A=a_{ij}$ 表示页面之间的连接关系.

$$a_{ij}=\begin{cases}1 & A_i\ link\ A_j\\ 0 & no\ link\end{cases}$$

使用有向图 $G=(V,E)$ 来表示以上的网页之间的连接关系, 设定 $V=\{A_1,A_2,\cdots A_n\}$ 来表示节点的集合, $E=\{A_i\to A_j\}$ 表示网页之间的连接关系.

定义 2. 对于某一个网页 $x$, 如果在 $[0,t]$ 的时间内, 若网页被单击一次, 则设定为设定为 1, 即 $click(x)=1$, 否则为 0.

定义 3. 在某一个时间段 $[0,t]$ 内, 对一个搜索行为 $q$, 倘若存在若干个检索结果, 因此可以将这些网页表示为 $x_1,x_2,\cdots x_n$. 假定网页 $x$ 在一天内被进行了 $m$ 次检索, 则查询网页 $x$ 点击数为 $U_p=\sum_{i=1}^{m}click(x,q)$. 其中 $U_p$ 表示用户查询相关性向量, $click(x,q)$ 表示在第 $i$ 次查询过程中, 网页 $x$ 在 $[0,t]$ 中被点击的次数.

### 2.3 需要考虑模型的因素

本文模型研究主要是基于网页 PageRank 算法基础上, 根据用户对网站访问的频率和偏爱程度来分析用户行为. 同时需要将用户行为对网页的比重进行考虑, 通过排序来计算综合权重, 给出用户搜索结果. 但实际过程中, 这样的算法存在一定的不足:

1) 用户行为影响因子. 假设在一段时间内对于任何一个检索行为 $q$, 用户的点击用量 C, 由于用户在点击的时候可能忽略了有关返回结果的 URL 消息, 因此用户在单击网页的过程中的概率在很大程度上受到了来自结果在返回网页中的位置影响. 因此网页被单击的概率与检索有着非常高的相关度, 为了避免出现某些网页排列处于底部而导致用户无法看到并且点击的情况. 因此采用如下公式来进行平衡弥补这个缺陷.

$$U_q=\sum_{i=1}^{n}c(pos(A,q))*click(A,q) \tag{3}$$

式中, $pos(A,q)$ 表示对于网页 $A$ 来说的查询中平衡因子, $click(A,q)$ 表示查询中的网页 $A$ 的点击次数, $U_q$ 值越高代表网页用户越受欢迎, 反之则出现在网页列表的最底端.

2)用户考虑时间. 用户在进行搜索的时候, 当发现相关的内容或者相近内容的时候, 用户会进行一定时间的浏览, 但是用户在网页上的时间并不代表用户

对搜索结果的满意度, 因为用户可能在网页上进行复制, 粘帖等操作, 因此用户对结果集合中任意浏览时间的长度决定了用户满意度的高低, 因此使用公式(4)来进行描述用户浏览时间权重值.

$$Time(A,q)=\frac{t_i}{\sum_{i=1}^{n}t_i} \tag{4}$$

式中, $t_i$ 表示用户在查询词集合 $q$ 的浏览网页 $A$ 的时间.

3)网页之间的相关性

在云计算的搜索过程中, 用户搜索可能会存在网页 $i$ 与网页 $j$ 的内容具有很强的相关性, 但是在搜索引擎排序之后会出现一个排名可能在前, 而另一个排名在后面的情况, 因此需要依靠平衡因子对排序比较靠后的网页进行补偿. 因此假设在某一个时间段 $[0,t]$ 内进行 $N$ 次迭代, 用户点击的网页构成网页矩阵 $C_{N\times N}$, 其中 $C_{i,j}$ 表示网页 $i$ 和网页 $j$ 被点击的次数. 如果存在 $C_{i,j}$ 和 $C_{j,k}$ 同时都大于 0, 则说明网页 $i,j,k$ 存在一定的关系, 因此得到如下的关系:

$$K(A,T_i)=K(ID_A,ID_{T_i}) \tag{5}$$

式中, $K(A,T_i)$ 表示从网页 $A$ 指向 $T_i$ 的网页关联度, $K(ID_A,ID_{T_i})$ 表示在网页关联度中根据两个网页的 ID 号查出关联值.

## 3 改进的用户搜索网页PageRank算法

PageRank 算法是用来标识网页的“等级/重要性”的一种方法, 它能够使那些 “等级/重要性”的网页在搜索结果中排名获得提升, 从而能够提高搜索结果的相关性和质量. 但是在云计算中的众多用户在访问过程中, 会根据该链接与主题的相似程度来有进行有选择的访问页面, 假设有 5 个链接的某个页面 X, 指向 A, B, C, D, E5 个页面, 与主题相似度分别为: 0.15, 0.24, 0.32, 0.42, 0.65. 因此在选择页面链接的时候, 选择 E 页面的的概率大的多. 因此改进后的页面算法中的 PageRank 需要考虑来自 2.3 节的因素, 不仅仅需要包括网页之间的直接的链接关系, 同时还需要包括隐含的简介因素在内, 因此在此基础上, 针对传统的 PageRank 公式进行改进, 对某一个页面 $X$ 的 PR 计算如下:

$$PR(X)=\frac{1-d}{N}+d*\sum_{(X,T_i)\in E}\left(\frac{PR(T_i)}{\sum_{k=1}^{M}click(T_i,X)}\right)*(\delta_1 f(X,T_i)+\delta_2 T(X,q)+\delta_3 K(X,T_i)) \tag{6}$$

式中, 参数 $\delta_1$, $\delta_2$, $\delta_3$ 分别表示了用户影响因子, 时间向量和网页相关性等因素, 并且 $\delta_1+\delta_2+\delta_3=1$, 同时满足 $d*(\delta_1 f(X,T_i)+\delta_2 T(X,q)+\delta_3 K(X,T_i))\le 1$, 这样更好的保证算法收敛. $E$ 表示互联网中网页总数, $d$ 为阻尼因子, $click(T_i,X)$ 表示网页 $T_i$ 和网页 $X$ 被同时点击的次数, 次数越高就说明两个网页之间的相关性就越大, 算法充分考虑了网页权值的计算过程中考虑了用户行为, 时间向量和网页相关性等影响.

### 3.1 影响因子的设定

有关 $\delta_1$, $\delta_2$, $\delta_3$ 的影响因子如何能够更好的确定影响算法高效的关键. 本文使用数据样本对数据因子进行分析, 对搜索日志中的数据进行分组计算, 发现影响因子对整个排序模型的干扰因素比较小. 反馈结果如表 1 所示.

表 1 参数记录列表

| 记录个数 | $\delta_1$ | $\delta_2$ | $\delta_3$ |
|---|---|---|---|
| 1000 | 0.17501 | 0.32079 | 0.42109 |
| 5000 | 0.21391 | 0.31137 | 0.41317 |
| 10000 | 0.21341 | 0.28513 | 0.41027 |
| 15000 | 0.21341 | 0.28519 | 0.41029 |

从以上表中可以发现, $\delta_1$ 的取值在[0.21340, 0.21345]之间, $\delta_2$ 的取值在[0.285010,0.285020]之间, $\delta_3$ 的取值在[0.41025,0.41030]之间. 说明单独使用相关度函数进行的排序并不能完全令用户达到满意, 同时最终的排序结果受到用户影响因子, 时间向量和网页相关性三个方面的影响.

### 3.2 实时反馈细节

本文算法实时反馈细节描述如下, 首先通过搜索引擎来获得一个结果集合, 然后当前用户在收到查询结果后点击目标网页, 获取目标网页的 ID 号, 根据网页隐含相关度值, 将网页的结果集合分别与目标网页的隐含相关度的进行比较, 最终将比较之后的隐含相关度大的网页作为新的搜索结果返回给用户. 如图 1 所示;

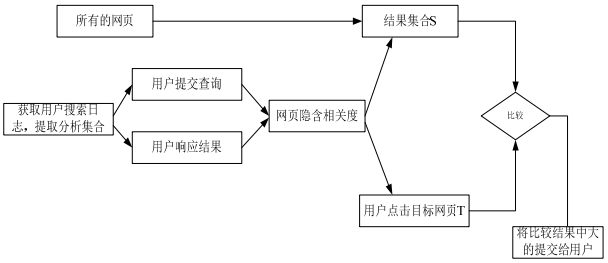


图 1 实时反馈细节比较

### 3.3 算法流程

步骤 1: 用户根据搜索目标进行网页搜索, 得到一定数量的页面;

步骤 2: 在基本 PageRank 算法的基础上, 依次引入用户影响因子, 时间向量, 网页相关性进行分析;

步骤 3: 通过用户影响因子对网页进行分析.

步骤 4: 通过时间向量对网页所需要的时间进行分析.

步骤 5: 对网页相关性之间的分析进行选择

步骤 6: 将步骤 3 到步骤 5 的结果提交给 PageRank 单元, 计算结果

步骤 7: 将结果反馈给用户.

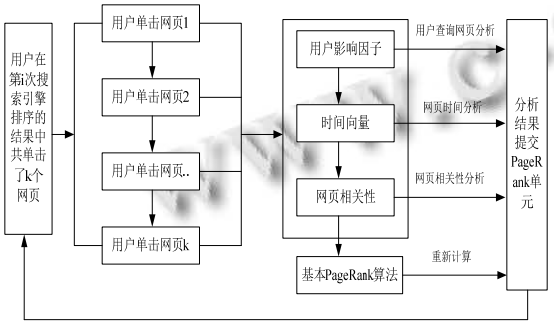


图 2　本文算法步骤

## 4　基于Hadoop架构的数据处理设计和分析

### 4.1 实验场景

本实验中使用 10 台 PC 来进行搭建基于 Hadoop 的分布式的计算平台, 将其中的一台 PC 作为服务器, 担任主控节点的功能, 同时其它的 9 台 PC 作为 TaskTracker, 硬件采用酷睿 i3, 硬盘 500G, 内存为 4GDDR3, 100Mbps, 软件采用 Windows 7,JDK1.8, Hadoop2.0 版本.

### 4.2 用户搜索分析的数据设计与实现

#### 4.2.1 数据收集

实验中使用的数据主要有优酷实验室[7]来进行提供, 本文选取在一个星期内大约 1000 万条的搜索引擎查询及用户单击日志集合, 将其中的查询记录设置为如表 2 的格式:

表 2　查询记录格式

| 字段名称 | 说明 |
|---|---|
| Query | 查询内容 |
| URL-Rank | 用户点击 URL 排名 |
| SessionID | 用户 Cookie 信息 |

#### 4.2.2 数据去重

在优酷数据集中具有大量的重复记录, 这个主要是因为用户在进行优酷搜索端中输入的关键词的顺序不同而产生的结果, 比如“周星驰 电影”以及“电影 周星驰”, 这样的用户表示方式通过点击产生的 2 条链接在后台数据库中看来只能当做一次查询结果, 需要去除重复的数据. 本文采用 Map/Reduce 并行计算模型的数据去重伪码如下:

```
Map(String No,Stirng Content)
 { String Str[]=”lineContent.spli()”;
    Collect(id,term);//收集所有数据
  }
 Reduece(String id,Tree terms)
 { While each<=terms
     {//查询词去重
     }
    Collect(id,new Terms);
 }
```

#### 4.2.3 用户数据分析

通过 Hadoop 的架构可以从多个角度来进行数据分析和挖掘, 包括用户搜索热词, 用户单击视频分析等. 用户在优酷中的行为主要是通过优酷中的使用热词来进行搜索, 从而对这些这些搜索行为进行分析. 可以根据数据集的大小来实现存储以及计算功能. 热词分析伪代码如下:

```
Map(String No,Stirng Content)
 { String Str[]=”lineContent.spli()”;
    Collect(id,term);//收集所有数据
    While each<=terms
      { collect(term,reduce)//将数据发送至 reduce
      }
  }
 Reduece(String query,Tree values)
 {   int num=0;//设置计数器
     While each<=terms
     {num=num+values//累计访问量
     }
     Collect(query,num);
}
```

### 4.3 实验结果分析

#### 4.3.1 算法分析

本文使用 5 台 PC 机(分别是 PC1 到 PC5)搭建 Hadoop 的分布式计算平台，其中 PC1 作为 Master，运行 Jobtracker；其余四台机器运行 Tasktrcker. 其中机器配置如下: Cpu 为 Inter Core2.2Ghz,4GDDR3,500G 硬盘，软件环境为 Ubuntu12,Hadoop 0.20.3,OpenSSH.

本文将文献[9]，文献[10]进行比较，实验数据来源了开源爬虫攻击 Heritrix[11]的数据，网页数量按照 10 万，30 万，50 万，100 万进行分配，设定 5 个集群节点，分别为 10 万，20 万，30 万，40 万. 从算法在不同数据量下的比较，不同节点数目，准确率三个方面进行比较，比较结果如图 3-5 所示.

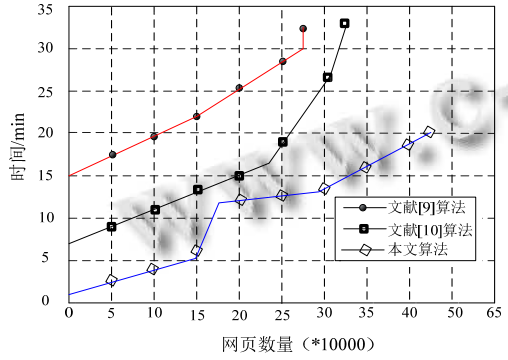


图 3 不同数量下的三种算法比较

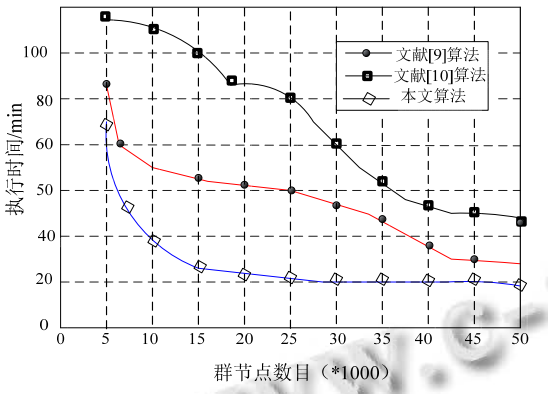


图 4 不同群节点数目下的算法比较

从图 3-5 中可以发现，伴随着数据量的不断增加，本文算法的运行时间由于两种参考文献算法，说明本文的算法在执行效率方面的提高，通过不同群节点在执行时间上的比较，发现本文算法执行时间比较平缓，这说明本文算法比较稳定，能够保证搜索结果的稳定性，进一步说明在 Hadoop 架构下的各个节点的 CPU 和内存资源能够得到充分的利用. 另外，从网页搜索的准确率方面来看，三种算法在网页数量比较大的时候准确度相差不大，主要是因为都是基于 PageRank 算法来进行排序的规则没有改变，导致网页的排列也没有变化. 当网页数量比较少的时候，本文的算法从用户影响因子，时间向量和网页相关性几个方面消除了同一个网站内的网页之间的排名的不公平性.

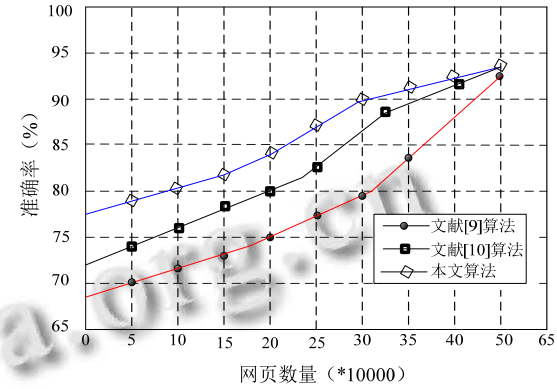


图 5 不同数据下的查询准确率

4.3.2 搜索比较分析

1)热榜分析

通过对优酷中的数据集分析，本文算法计算访问量排名在前 1000 名访问量占据了总的访问量为 75.36%，说明搜索引擎在每天处理的查询请求大部分是重复请求. 如图 6 所示.

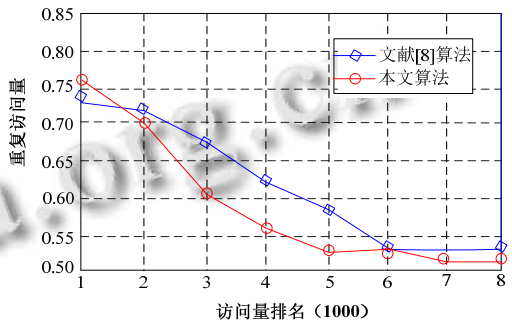


图 6 热榜分析

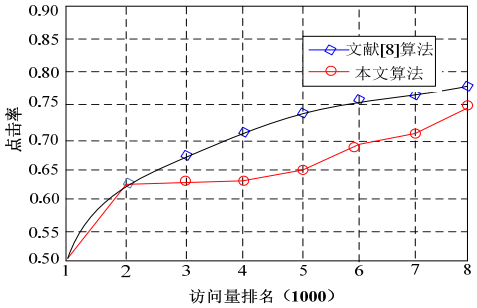


图 7 用户点击数与 URL

2)用户点击数与 URL 分析

对本实验数据集的统计分析结果如图 7 所示,可以看到用户点击的排名前 1000 的 URL 能占到所有点击率的 72.46%,与文献[8]所得出的 75%相差不大,说明论文平台算法具有有效性和正确性.

3)分布式平台效率分析

本文根据搜素日志不同的三种搜索数据的规模,分别是基本样本数据(1Mb), 一天数据(5Gb),一星期数据(40Gb),在此平台上对数据分析中的查询主题排行榜的效率进行测试分析, 如表 3 所示.

表 3 数据处理时间

| | 1 个节点 (秒) | 2 个节点 (秒) | 3 个节点 (秒) | 4 个节点 (秒) |
|---|---|---|---|---|
| 基本样本数据 | 12.829 | 17.417 | 21.821 | 25.813 |
| 一天数据 | 57.327 | 52.216 | 45.741 | 36.816 |
| 一星期数据 | 180.716 | 162.721 | 142.715 | 99.218 |

从表 3 中可以发现, 当数据规模比较小的时候,平台在进行处理的时间花费比较多, 当数据规模比较大的时候, 平台处理的时间明显减少, 这说明在 Hadoop 平台下改进的 PageRank 算法对数据规模比较大的数据处理更具有适应性.

## 5 结语

基于 Hadoop 平台下的用户搜索行为分析通过查询日志和数据挖掘技术来提高信息获取的信息, 这种技术可以应用于海量的日志文件处理, 通过对优酷数据库的数据分析, 应用 Hadoop 分布式计算框架进行搜索引擎分析, 可以有效的解决的云计算下的并行计算模型的不足, 通过分析比较本文的分析具有良好的指导和实践意义.

## 参考文献

1 Page L, Brin S, Motwani R, et al. The Pagerank Citation Ranking; Bringing Order to the Web.Techical Report, Standford Digital Library Technologies Project,2011.

2 Kleinberg JM. Authoritative Sources in a Hyperlinked Environment. Journal of the ACM, 2012,46(5):604–632.

3 Chakrabarti S, Dom B, Raghavan P, et al. Automatic Resource List Compilation by Analyzing Hyperlinked Resource List Compilation by Analyzing Hyperlink Structure and Assocaitaed Text. [2013-11-17]. http://citeseer.ist.psu.edu/chakrabarti98automatic.htm

4 PoweredBy-HadoopWiki. [2013-11-17]. http://wiki,apache.org/hadoop/PoweredBy.

5 Borthakur D. HDFS Architecture. [2012-11-17]. http:// hadoop.apache.org/common/ docs/current/hdfs_design.

6 毛国君,段立娟.数据挖掘原理与算法.北京:清华大学出版社,2009.

7 优酷实验室.[2009-11-17]. http://labs.youku.com

8 刘健,刘奕群,马少平等.搜索引擎用户行为与用户满意度的关联研究.中文信息学报,2014,28(1):73–79.

9 陈诚,战荫伟,李鹰.基于网页链接分类的 PageRank 并行算法.计算机应用,2015,35(1):48–52.

10 曹珊珊,王冲.基于网页链接与用户反馈的 PageRank 算法改进研究.计算机科学,2014,41(12):179–182.