HITS 和 推荐算法

HITS 算法是信息检索技术中的一种检索算法，和PageRank、SALSA算法一起组成目前比较流行的检索算法，众所周知，PageRank算法在Google的搜索引擎中得到实践和应用，效果就不用说了，大多技术控，都会使用Google搜索引擎来检索自己遇到的问题，其在中国被各方势力的围追堵截，依然没能阻止其熠熠生辉的光亮，如www.gfsoso.com，www.glgoo.com其实都是Google引擎的影子，可见其魅力所在呀，额，扯远了，今天的主角是HITS算法，HITS算法是Jon Kleinberg提出来的一种链接分析算法，英文全称是Hyperlink-Induced Topic Search，目前已被Teoma搜索引擎（www.teoma.com）作为链接分析算法使用，还有一个重要应用，就是科学杂志里面的影响因子排名。下面来具体讲一下HITS算法。

# 1. 基本概念

首先介绍HITS的两个基本概念：Hub页面（枢纽页面，Hub）和Authority页面（权威页面），Authority页面是指与某个领域或者某个主题相关的高质量的网页，比如搜索引擎领域，Google和百度首页即该领域的高质量网页，在视频领域，优酷和YouTube等首页即该领域的高质量网页，Hub 页面是指包含了很多高质量“Authority”页面链接的网页，比如hao123。HITS算法的目的即是通过一定的技术手段，在海量网页中找到与用户查询主题相关的高质量“Authority”页面和“Hub”页面，尤其是“Authority”页面，因为这些页面代表了能够满足用户查询的高质量的内容，搜索引擎以此作为搜索结果返回给用户。

# 2. 算法基本思想：相互增强关系

基本假设1：一个好的Authority页面会被很多好的Hub页面指向。

基本假设2：一个号的Hub页面会指向很多好的Authority页面；

# 3. HITS算法

具体算法：利用上面提到的两个基本假设，以及相互增强关系等原则进行多轮迭代计算，每轮迭代计算更新每一个页面的两个权值，知道权值稳定不再发生明显的变化为止。

步骤：

## 3.1 根集合

1） 将查询q提交给基于关键词查询的检索系统，从返回结果页面的集合总取前n个网页（如n=200），作为根集合（root set），记为root，则root满足：

① root中的网页数量较少；

② root中的网页是与查询q相关的网页；

③ root中的网页包含较多的权威（Authority）网页。

这个集合是个有向图结构：G(V,E)。

## 3.2 扩展集合base

在根集root的基础上，HITS算法对网页集合进行扩充集合base，扩充原则是：凡是与根集内网页有直接链接指向关系的网页都被扩充到集合base，无论是有链接指向根集内页面也好，或者是根集页面有链接指向的页面也好，都被扩充进入扩展网页集合base，HITS算法在这个扩充网页集合内寻找好的“Hub”页面与好的“Authority”页面。

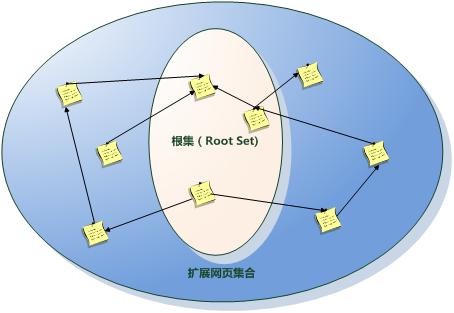


图2 根集与扩展集

## 3.3 计算扩展集base中所有页面的Hub值和Authority值

步骤：

1.） 对于任意的网页p，有；

2.）Authority更新：；

3.）Hub更新： ；

4.）正则化。

伪代码：

|  |  |
| --- | --- |
| 1 | G := set of pages |
| 2 | for each page p in G do |
| 3 | p.auth = 1 // p.auth is the authority score of the page p |
| 4 | p.hub = 1 // p.hub is the hub score of the page p |
| 5 | function HubsAndAuthorities(G) |
| 6 | for step from 1 to k do // run the algorithm for k steps |
| 7 | norm = 0 |
| 8 | for each page p in G do // update all authority values first |
| 9 | p.auth = 0 |
| 10 | for each page q in p.incomingNeighbors do // p.incomingNeighbors is the set of pages that link to p |
| 11 | p.auth += q.hub |
| 12 | norm += square(p.auth) // calculate the sum of the squared auth values to normalise |
| 13 | norm = sqrt(norm) |
| 14 | for each page p in G do // update the auth scores |
| 15 | p.auth = p.auth / norm // normalise the auth values |
| 16 | norm = 0 |
| 17 | for each page p in G do // then update all hub values |
| 18 | p.hub = 0 |
| 19 | for each page r in p.outgoingNeighbors do // p.outgoingNeighbors is the set of pages that p links to |
| 20 | p.hub += r.auth |
| 21 | norm += square(p.hub) // calculate the sum of the squared hub values to normalise |
| 22 | norm = sqrt(norm) |
| 23 | for each page p in G do // then update all hub values |
| 24 | p.hub = p.hub / norm // normalise the hub values |

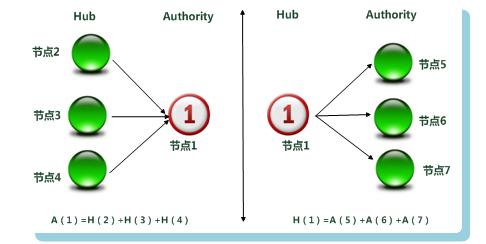


图3 Hub与Authority权值计算.

# 4. HITS 算法存在的问题

HITS算法整体而言是个效果很好的算法，目前不仅应用在搜索引擎领域，而且被“自然语言处理”以及“社交分析”等很多其它计算机领域借鉴使用，并取得了很好的应用效果。尽管如此，最初版本的HITS算法仍然存在一些问题，而后续很多基于HITS算法的链接分析方法，也是立足于改进HITS算法存在的这些问题而提出的。

归纳起来，HITS算法主要在以下几个方面存在不足：

1.计算效率较低

因为HITS算法是与查询相关的算法，所以必须在接收到用户查询后实时进行计算，而HITS算法本身需要进行很多轮迭代计算才能获得最终结果，这导致其计算效率较低，这是实际应用时必须慎重考虑的问题。

2.主题漂移问题

如果在扩展网页集合里包含部分与查询主题无关的页面，而且这些页面之间有较多的相互链接指向，那么使用HITS算法很可能会给予这些无关网页很高的排名，导致搜索结果发生主题漂移，这种现象被称为“紧密链接社区现象”（Tightly-Knit CommunityEffect）。

3.易被作弊者操纵结果

HITS从机制上很容易被作弊者操纵，比如作弊者可以建立一个网页，页面内容增加很多指向高质量网页或者著名网站的网址，这就是一个很好的Hub页面，之后作弊者再将这个网页链接指向作弊网页，于是可以提升作弊网页的Authority得分。

4.结构不稳定

所谓结构不稳定，就是说在原有的“扩充网页集合”内，如果添加删除个别网页或者改变少数链接关系，则HITS算法的排名结果就会有非常大的改变。

# 5. HITS算法与PageRank算法比较

HITS算法和PageRank算法可以说是搜索引擎链接分析的两个最基础且最重要的算法。从以上对两个算法的介绍可以看出，两者无论是在基本概念模型还是计算思路以及技术实现细节都有很大的不同，下面对两者之间的差异进行逐一说明。

1.HITS算法是与用户输入的查询请求密切相关的，而PageRank与查询请求无关。所以，HITS算法可以单独作为相似性计算评价标准，而PageRank必须结合内容相似性计算才可以用来对网页相关性进行评价；

2.HITS算法因为与用户查询密切相关，所以必须在接收到用户查询后实时进行计算，计算效率较低；而PageRank则可以在爬虫抓取完成后离线计算，在线直接使用计算结果，计算效率较高；

3.HITS算法的计算对象数量较少，只需计算扩展集合内网页之间的链接关系；而PageRank是全局性算法，对所有互联网页面节点进行处理；

4.从两者的计算效率和处理对象集合大小来比较，PageRank更适合部署在服务器端，而HITS算法更适合部署在客户端；

5.HITS算法存在主题泛化问题，所以更适合处理具体化的用户查询；而PageRank在处理宽泛的用户查询时更有优势；

6.HITS算法在计算时，对于每个页面需要计算两个分值，而PageRank只需计算一个分值即可；在搜索引擎领域，更重视HITS算法计算出的Authority权值，但是在很多应用HITS算法的其它领域，Hub分值也有很重要的作用；

7.从链接反作弊的角度来说，PageRank从机制上优于HITS算法，而HITS算法更易遭受链接作弊的影响。

8.HITS算法结构不稳定，当对“扩充网页集合”内链接关系作出很小改变，则对最终排名有很大影响；而PageRank相对HITS而言表现稳定，其根本原因在于PageRank计算时的“远程跳转”。

# 6. 基于HITS的推荐算法

推荐系统中，consumer-product关系可以形成包含消费者和产品节点的双向图，消费者节点c和产品节点p之间的关联表示p可以表示c的部分兴趣，c的部分爱好是p，同Web page ranking相比，推荐要求的是个性化的需求，而非generally popular products，因此，我们将原先的Authority和Hub值改进，我们定义表示产品p相对于consumer 的评分，将其作为对p的“Authority”，相似的，我们定义了一个consumer repesentativeness score 表示c相对于，用来衡量作为的hub的c与兴趣关联的程度。

我们使用 来表示 的product representativeness matrix，表示产品对consumer  的representativeness score， 表示 consumer representativeness matrix, 表示consumer 相对于consumer 的representativeness score。使用Authority和hub scores来定义product和consumer的representativeness scores as  。和 ，直观上，the sum of the representativeness scores of products linked to a consumer gives the consumer representativeness score, and vice versa.

这样的定义有两个问题：1. 如果一个consumer与所有的products相关联，那么这个consumer对于所有的目标用户就具有很高的representativeness score，然而，这样的用户行为对于预测用户兴趣爱好，具有很少的信息量；2. 一个更加基本的问题在于覆盖率，PR和CR将使用identical columns 来converge to matrices，这样scores representing product ranking independent of particular consumers，因此对于推荐仅提供有限的值。

为了解决这些问题，我们定义consumer representativeness score： ， 是一个由A变化而来的 的矩阵，

 ， 表示source consumer representativeness score matrix,

 ，即， 是一个的单位阵，这个新的定义来自于network spreading-activation 模型，B的引入….

算法：

1. 基于交易行为数据，构建interaction matrix A和关联矩阵B， ， ，

 ，此处 ；

2. 设置，初始化consumer representativeness matrix： ；

3. 每一次迭代t，执行以下操作：

a. 

b.  ；

c. 正则化 ，以便于 ；

d.  。

# 7. 参考文献

[1] http://blog.csdn.net/hguisu/article/details/8013489

[2] Zan Huang, Daniel Zeng, Hsinchun Chen, "A Comparison of Collaborative-Filtering Recommendation Algorithms for E-commerce", IEEE Intelligent Systems, vol.22, no. 5, pp. 68-78, September/October 2007, doi:10.1109/MIS.2007.80