网页重要度排序模型分析

【摘要】

本文以建立数学模型的方式解决了网页重要度排序问题。我们通过一系列统计分析的方法，结合MATLAB编程语言，在只考虑超链链接的情况、及更为一般的情况下建立了两个不同的数学模型，有效提升了网页重要度排序的准确度。

对于问题一，我们参照了PageRank Algorithm【1】，得到了只考虑超链链接数量及其权重的网页重要度排序。

对于问题二，我们详细讲解了我们的算法，代码及注释可见附录。

对于问题三，我们选取了中山大学官网，中山大学教务系统网等十个网页，采用问题一中的数学模型，得到了这十个网页的重要度排序，其中中山大学官网的重要度最高，符合我们的认知，客观上验证了我们算法的准确性。

对于问题四，我们改进了PageRank Algorithm，考虑更多的影响因素：HTML结构，外部链接的数量，网页可信度。采用灰色关联分析体系构建模型，得到每个网页的综合得分，以此为依据对这十个网页做出更为准确的重要度排序。

关键字：PageRank Algorithm　灰色关联分析体系　MATLAB编程语言

一【问题重述】

1.1问题背景

一般来说，重要的网页通常会有许多其他网页的超链接导向它，但假如只通过被指向次数这一指标进行重要度排序，有些商家会通过建立许多空壳网站指向自己的网站来提高自己的网站的重要度。我们希望能防止这种作弊。

1.2问题提出

在搜索引擎中，我们往往需要对网页做重要度排序，其中一种思路是为网页的超链接赋予权重。本文根据题目要求与相关数据，依次提出并回答以下问题：

1. 根据附件一中9个网页Ti(i=1,2....9)的互相指向情况,用算法排出他们的重要度顺序。
2. 讲解上一个问题中的算法。
3. 任选10个和中山大学相关的网页，通过算法排出这些网页的重要度顺序。
4. 改进上述的算法，考虑更多对网页重要性的影响因素，用优化后的算法排出选择的10个和中山大学相关的网页的重要度。

二【基本假设】

1. 网页的重要性是由链接到它的网页的数量和重要性决定的。在分配网页重要度的过程中，只考虑排序模型中已给定的网页。
2. 在网页随机地跳转过程中，我们会随机地选择当前网页的一个链接到达另一个网页，对于一个没有任何链接的网页，会在跳转过程中随机选择一个网页。
3. 好的网站很少会链接到坏的网站，反之不成立。

三【符号说明】

P	网页
I(P)	网页P的重要度
S(P)	其他拥有超链接到网页P的网页集合
N(P)	网页P上所有超链接的数量和
$M(P_i, P_j)$	网页Pi到Pj网页的超链接数量和
H	所以网页之间的超链接矩阵
St(H)	超链接矩阵的平稳向量
D	悬挂点网页
T	在网页上停留的时间
R	随机矩阵
F	修正矩阵

G	谷歌矩阵
Er	网页HTML检测中产生的错误数量
Li	网页指向外部网站的链接数

四【模型分析】

4.1原网页排序算法的分析

4.1.1基本求解方法及对求解目标的数学化

我们认为，一个网页的重要度是由其他网页上链接到该网页上的超链接数量，以及这个外部网页的重要度决定。实际上，假设网页P_i有n个指向其他网页的链接，如果这些链接中的某一个指向到网页P_j，那么也就相当于P_i将其重要度的1/n赋给了P_j，网页P_j的重要度就是所有指向P_j的其他网页所贡献的重要度之和。

换言之，假设其他拥有超链接到P_i的网页集合为$S(P_i)$，那么我们有：

$$\mathbf{I}(P_i)=\sum_{P_j\in S(P_i)}\frac{\mathbf{I}(P_j)\times\mathbf{M}(P_j,P_i)}{\mathbf{N}(P_j)}$$

由以上算式我们可以得到每个网页重要度的度量标准，但是在这里我们很容易地就可以看出该算式的递归性，以至于不能仅仅依靠它来得出最好的重要度排序结果。为进一步计算，我们引入超链接矩阵(HyperLink Matrix)的概念， 将网页的超链接矩阵记为H(P),
其中第i行第j列的元素记为H_{ij}，那么我们有：

$$\mathbf{H_{ij}}=\begin{cases}\mathbf{M}(P_j,P_i)/\mathbf{N}(P_j)\text{如果}\mathbf{P_j}\in\mathbf{S}(P_i)\\ 0\text{上述条件不成立}\end{cases}$$

与此同时，我们还需要定义一个n行单列向量I，I = [$I(P_i)$]，此单列向量描述了n个网页各自的重要度，结合上文的网页重要度计算公式，我们可以得到：

$$\mathbf{I}=\mathbf{HI}$$

由于我们网页排序算法的最终结果即为求出各网页的重要度排序，换言之即为求解I，那么根据上述结果，可以推出I向量是矩阵H的特征值为1的特征向量，为了便于表示，我们将I记为H的平稳向量St(H)。

通过引用附件一的数据，结合我们上述的超链接矩阵构建的标准，运用Matlab软件，详细代码**参考**附录7.1，我们可以得到如下的计算结果：

```
>> webpages_analysis
     0     0     1     0     2     2     0     3     1
     0     0     0     0     0     0     3     0     0
     2     0     0     0     3     1     0     1     0
     0     1     0     0     0     0     4     0     0
     1     0     1     0     0     1     0     1     0
     1     0     2     0     1     0     0     1     0
     1     2     0     1     1     0     0     0     0
     1     0     1     0     0     0     0     6     0
     0     0     0     0     0     0     0     0     0
```

（1）附件一中每个网页指向其他网页的链接数矩阵

0	0	0.2857	0	0.2500	0.2000	0.2000	0.1250	0
0	0	0	0.2000	0	0	0.4000	0	0
0.1111	0	0	0	0.2500	0.4000	0	0.1250	0
0	0	0	0	0	0	0.2000	0	0
0.2222	0	0.4286	0	0	0.2000	0.2000	0	0
0.2222	0	0.1429	0	0.2500	0	0	0	0
0	1.0000	0	0.8000	0	0	0	0	0
0.3333	0	0.1429	0	0.2500	0.2000	0	0.7500	0
0.1111	0	0	0	0	0	0	0	0

（2）超链接矩阵的初步求解结果

4.1.2对超链接矩阵构建算法的进一步优化

在现实生活中，我们会经常浏览到一些无法指向外部网页的界面，即N(P)=0,我们往往无法从这个网页到达其他任何的网页。在实际的网页重要度计算中，这些网页总是从其他外部网页获取到重要度，但是却不赋给其他网页任何的重要度，我们将这种类型的网页称为悬挂点D。

想象我们随机地在网上跳转网页，也就是说，当我们访问一个网页时，一秒钟后我们随机地选择当前网页的一个链接到达另一个网页。例如，我们正访问含有$N(P_j)$个链接的网页P_j，其中一个链接引导我们访问了网页P_i，那么下一步转到网页P_i的概率就是$M(P_j, P_i)/N(P_j)$。由于跳转网页的过程往往是随机的，我们用T_j表示停留在网页P_j上的时间，那么我们从网页P_j转到网页P_i的时间为$T_j/N(P_j)$。如果我们转到了网页Pi，那么我们必然是从一个指向它的网页而来。这意味着:

$$T_i = \sum_{P_j \in S(P_i)} \frac{T_j \times M(P_j, P_i)}{N(P_j)}$$

我们将上述算式论述过程称为重要度的概率化解释。

通过比较上下算式，我们可以看出$I(P_j)$与T_j之间是明显存在着一个等价关系，并且这也是我们生活中经验的反照：

越是重要的网页，我们越是在其界面上花费的浏览时间越多。

当分析进行到这一步时，我们往往会发现一个比较重要的相关性矛盾：

1. 当进入到悬挂点网页时，我们就会被困在悬挂点网页无法离开，此时的浏览时间近似于正无穷。

2. 我们不能从悬挂点网页转到其他外部网页，这并不意味着悬挂点网页很重要。我们称此相关性矛盾为悬挂点问题。

从附件一中的数据分析可得，P_9上并没有任何指向P_1~P_8的超链接，因此我们有：P_9是本次网页重要度排序中的悬挂点D_0。于是，为了解决初步超链接矩阵所导致的相关性矛盾问题，避免重要度的随机性误差出现，我们引入随机矩阵R，让R辅助修正超链接矩阵H。再结合重要度的概率化解释，规定网页排序向量中所有元之和为1，为化解本题第一问中的悬挂点问题，不妨令修正矩阵F为：

F =

0	0	0	0	0	0	0	0	0.1111
0	0	0	0	0	0	0	0	0.1111
0	0	0	0	0	0	0	0	0.1111
0	0	0	0	0	0	0	0	0.1111
0	0	0	0	0	0	0	0	0.1111
0	0	0	0	0	0	0	0	0.1111
0	0	0	0	0	0	0	0	0.1111
0	0	0	0	0	0	0	0	0.1111
0	0	0	0	0	0	0	0	0.1111

（3）阵算法，化解悬挂点问题的修对于附件一数据，进一步加强超链接矩正矩阵F

更进一步地讲，一般情况下的修正矩阵F的构造条件如下：

1. F的初始矩阵为nXn的全零矩阵。

2. 当出现悬挂点时，我们可以在初步的超链接矩阵上观测到悬挂点网页的列向量为零向量，此时，对应地将修正矩阵F的对应列加以1/n，满足概率化解释中排序向量元之和为1的总法则。

最终地，我们将随机矩阵R定义为：

$$R = H + F$$

0	0	0.2857	0	0.2500	0.2000	0.2000	0.1250	0.1111
0	0	0	0.2000	0	0	0.4000	0	0.1111
0.1111	0	0	0	0.2500	0.4000	0	0.1250	0.1111
0	0	0	0	0	0	0.2000	0	0.1111
0.2222	0	0.4286	0	0	0.2000	0.2000	0	0.1111
0.2222	0	0.1429	0	0.2500	0	0	0	0.1111
0	1.0000	0	0.8000	0	0	0	0	0.1111
0.3333	0	0.1429	0	0.2500	0.2000	0	0.7500	0.1111
0.1111	0	0	0	0	0	0	0	0.1111

（4）对于附件一数据得出的随机矩阵R

4.1.3对超链接矩阵的最终优化----构建谷歌矩阵，求解最终平稳向量

一般而言，我们在寻求特征值固定的矩阵特征向量之时，常常引入**幂法**来找寻特征值的绝对值最大的特征向量。根据我们求解的目的，我们需要使得最终所需的平稳向量的特征值1为当前矩阵的最大特征值，下面我们分为两个方面来进行讨论：

1.假定随机矩阵R的特征值为λ_i，并且有1 >= λ_i.

2.随机矩阵R的特征值为λ_i，并且存在λ_i>1。

在实际的计算中，上述两种情况往往都是有可能发生的，为采用幂法求解，我们称符合条件一的随机矩阵R为本原的，反之为非本原。

对于任意给定的两个网页，如果我们有一定存在的一条回路使得我们从第一个网页转到第二个网页，那我们称这个网页集合为强连通的，反之则为弱连通的。结合悬挂点的概念，我们发现，一切具有悬挂点的网页集合皆为弱连通，即其对应的随机矩阵为可约的。实际上，通过查阅资料我们得知，如果矩阵R不可约，那么一定存在一个所有元均为正数的平稳向量。

为了简化计算，使用幂法求解，并且最终得到一个所有元皆为正数的平稳向量，我们引入谷歌矩阵进行对随机矩阵R的进一步修正：

为得到一个本原且不可约的矩阵，我们将修正随机跳转网页的方式。就目前来看，我们的随机跳转模式由随机矩阵R确定：或者是从当前网页上的链接中选择一个，或者是对没有任何链接的网页，随机地选取其他网页中的任意一个。为了做出修正，首先选择一个介于0到1之间的参数α。然后假定随机跳转的方式略作变动。具体是，遵循矩阵R的方式跳转的概率为α，而随机地选择下一个页面的概率是1-α。若记所有元均为1的n×n矩阵为J，那么我们就可以得到谷歌矩阵G（Google matrix），公式如下：

$$G = \alpha \times R + (1-\alpha)\frac{J}{n}$$

注意到G为随机矩阵，因为它是随机矩阵的组合。进而，矩阵G的所有元均为正，因此G为本原且不可约。从而，G存在唯一的平稳向量I，后者可以通过幂法获得。

同时，参数α的作用是一个重要因素。若α=1，则G=S。这意味着我们面对的是原始的网络超链结构。然而，若α=0，则G=1/nJ。也即我们面对的是一个任意两个网页之间都有连接的网络，它已经丧失了原始的网络超链结构。显然，我们将会把α的值取得接近于1，从而保证网络的超链结构在计算中的权重更大。结合谷歌矩阵的创始者，PageRank Algorithm的提出者谢尔盖•布林和拉里•佩奇选择α=0.85。本文服从此准则，使用的α皆为0.85.

0.0167	0.0167	0.2595	0.0167	0.2292	0.1867	0.1867	0.1229	0.1111
0.0167	0.0167	0.0167	0.1867	0.0167	0.0167	0.3567	0.0167	0.1111
0.1111	0.0167	0.0167	0.0167	0.2292	0.3567	0.0167	0.1229	0.1111
0.0167	0.0167	0.0167	0.0167	0.0167	0.0167	0.1867	0.0167	0.1111
0.2056	0.0167	0.3810	0.0167	0.0167	0.1867	0.1867	0.0167	0.1111
0.2056	0.0167	0.1381	0.0167	0.2292	0.0167	0.0167	0.0167	0.1111
0.0167	0.8667	0.0167	0.6967	0.0167	0.0167	0.0167	0.0167	0.1111
0.3000	0.0167	0.1381	0.0167	0.2292	0.1867	0.0167	0.6542	0.1111
0.1111	0.0167	0.0167	0.0167	0.0167	0.0167	0.0167	0.0167	0.1111

（5）对于附件一数据，通过算法得出的谷歌矩阵

1 至 6 列

0.3372 + 0.0000i	0.0344 + 0.0000i	0.1369 + 0.0000i	-0.2148 + 0.0000i	0.1613 - 0.1987i	0.1613 + 0.1987i
0.1393 + 0.0000i	-0.2931 + 0.0000i	-0.0169 + 0.0000i	-0.3985 + 0.0000i	-0.0066 - 0.0013i	-0.0066 + 0.0013i
0.2948 + 0.0000i	0.1279 + 0.0000i	0.1814 + 0.0000i	0.0180 + 0.0000i	-0.2363 + 0.4631i	-0.2363 - 0.4631i
0.0863 + 0.0000i	-0.1295 + 0.0000i	-0.0037 + 0.0000i	-0.2344 + 0.0000i	0.0151 + 0.0034i	0.0151 - 0.0034i
0.2931 + 0.0000i	-0.0443 + 0.0000i	0.3098 + 0.0000i	-0.2161 + 0.0000i	0.6092 + 0.0000i	0.6092 + 0.0000i
0.2098 + 0.0000i	0.0199 + 0.0000i	0.2675 + 0.0000i	0.1348 + 0.0000i	-0.4000 - 0.3401i	-0.4000 + 0.3401i
0.2252 + 0.0000i	-0.5080 + 0.0000i	-0.0304 + 0.0000i	0.8186 + 0.0000i	0.0150 - 0.0132i	0.0150 + 0.0132i
0.7652 + 0.0000i	0.7870 + 0.0000i	-0.8822 + 0.0000i	0.0635 + 0.0000i	-0.0928 + 0.0510i	-0.0928 - 0.0510i
0.0799 + 0.0000i	0.0057 + 0.0000i	0.0376 + 0.0000i	0.0290 + 0.0000i	-0.0647 + 0.0358i	-0.0647 - 0.0358i

7 至 9 列

0.5018 + 0.0000i	-0.2294 + 0.0000i	-0.1788 + 0.0000i
-0.0238 + 0.0000i	-0.2708 + 0.0000i	-0.5623 + 0.0000i
0.0197 + 0.0000i	0.0637 + 0.0000i	0.2119 + 0.0000i
0.0398 + 0.0000i	0.1895 + 0.0000i	0.7113 + 0.0000i
-0.7584 + 0.0000i	-0.2028 + 0.0000i	-0.0068 + 0.0000i
0.3782 + 0.0000i	-0.0376 + 0.0000i	-0.0207 + 0.0000i
0.0382 + 0.0000i	-0.3725 + 0.0000i	-0.2950 + 0.0000i
-0.0389 + 0.0000i	0.0537 + 0.0000i	0.0278 + 0.0000i
-0.1566 + 0.0000i	0.8062 + 0.0000i	0.1126 + 0.0000i

（6）上述谷歌矩阵所求的特征向量

1 至 6 列

1.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i
0.0000 + 0.0000i	0.6627 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i
0.0000 + 0.0000i	0.0000 + 0.0000i	0.4384 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i
0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i	-0.6052 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i
0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i	-0.2088 + 0.1223i	0.0000 + 0.0000i
0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i	-0.2088 - 0.1223i
0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i
0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i
0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i

7 至 9 列

0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i
0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i
0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i
0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i
0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i
0.0000 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i
-0.2083 + 0.0000i	0.0000 + 0.0000i	0.0000 + 0.0000i
0.0000 + 0.0000i	0.0676 + 0.0000i	0.0000 + 0.0000i
0.0000 + 0.0000i	0.0000 + 0.0000i	-0.0556 + 0.0000i

（7）上述特征向量所对应的特征值

根据（6）（7），我们可以很明显地看到特征值为1的情况，并且接下来的特征值都满足小于1的条件，因此我们得到了问题一的最终结果-----平稳向量I：

I =(0.3372, 0.1393, 0.2948, 0.0863, 0.2931, 0.2098,
0.2252, 0.7652, 0.0799)

所以这九个网页的重要度排序为：

$$P_8 > P_1 > P_3 > P_5 > P_7 > P_6 > P_2 > P_4 > P_9$$

详细代码**参考**附录7.2

4.2 网页排序算法应用于与中山大学有关的十个网页

我们选择了与**中山大学**有关的十个网页，分别为：

P_1 -- 中山大学官网：http://www.sysu.edu.cn/2012/cn/index.htm

P_2 -- 中山大学本科教务系统：http://uems.sysu.edu.cn/jwxt/

P_3 -- **中山大学**学生微教务系统：http://wjw.sysu.edu.cn/

P_4 -- 中山大学图书馆官网：http://library.sysu.edu.cn/

P_5 -- 中山大学教务部官网：http://jwb.sysu.edu.cn/

P_6 --
中山大学百度贴吧：http://tieba.baidu.com/f?kw=%D6%D0%C9%BD%B4%F3%D1%A7&fr=ala0&tpl=5&pn=0&&red_tag=o3157195774

P_7 -- 中山大学新闻网：http://news2.sysu.edu.cn/index.htm

P_8 -- 我的中大：http://my.sysu.edu.cn/welcome

P_9 -- 中山大学网络与信息技术中心：http://inc.sysu.edu.cn/

P_{10} -- 中山大学信息技术帮助台：http://helpdesk.sysu.edu.cn/

步骤一：我们先找到这十个网页上的互相指向情况。

与题目所给的定义相同，我们定义第i行第j列的数字为网页P_i指向P_j的超链接数。通过Web脚本语言JavaScript脚本程序代码的书写，我们成功地拿到了这十个网页的相互指向情况数据，详细代码**参考**附录7.3。

1	0	0	1	0	0	1	0	0	0
0	3	1	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	1
1	0	0	3	0	0	0	0	0	0
1	1	0	0	3	0	0	0	0	0
1	0	0	0	0	2	0	0	0	0
1	0	0	0	0	0	1	0	0	0
1	0	0	0	0	0	0	1	0	1
1	0	0	0	0	0	0	0	2	1
0	0	0	0	0	0	0	1	0	1

（8）**与**中山大学**有关的**十个网页的相互指向情况

步骤二：运用Matlab软件，得到随机矩阵：

0.3333	0	0.3333	0.2500	0.2000	0.3333	0.5000	0.3333	0.2500	0
0	0.7500	0.3333	0	0.2000	0	0	0	0	0
0	0.2500	0	0	0	0	0	0	0	0
0.3333	0	0	0.7500	0	0	0	0	0	0
0	0	0	0	0.6000	0	0	0	0	0
0	0	0	0	0	0.6667	0	0	0	0
0.3333	0	0	0	0	0	0.5000	0	0	0
0	0	0	0	0	0	0	0.3333	0	0.5000
0	0	0	0	0	0	0	0	0.5000	0
0	0	0.3333	0	0	0	0	0.3333	0.2500	0.5000

步骤三：我们参照谷歌矩阵构建法来修正随机矩阵：

0.2983	0.0150	0.2983	0.2275	0.1850	0.2983	0.4400	0.2983	0.2275	0.0150
0.0150	0.6525	0.2983	0.0150	0.1850	0.0150	0.0150	0.0150	0.0150	0.0150
0.0150	0.2275	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150
0.2983	0.0150	0.0150	0.6525	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150
0.0150	0.0150	0.0150	0.0150	0.5250	0.0150	0.0150	0.0150	0.0150	0.0150
0.0150	0.0150	0.0150	0.0150	0.0150	0.5817	0.0150	0.0150	0.0150	0.0150
0.2983	0.0150	0.0150	0.0150	0.0150	0.0150	0.4400	0.0150	0.0150	0.0150
0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.2983	0.0150	0.4400
0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.0150	0.4400	0.0150
0.0150	0.0150	0.2983	0.0150	0.0150	0.0150	0.0150	0.2983	0.2275	0.4400

最终，我们得到了该谷歌矩阵的特征向量与其对应的特征值：

0.6168	-0.8197	-0.8121	-0.7448	-0.2026	0.1667	0.0000	0.1530	0.2132	-0.2447
0.1998	-0.0188	0.0000	-0.0000	0.1181	-0.0000	-0.0000	0.4098	0.0000	-0.0000
0.0795	0.0477	-0.0000	-0.0000	0.0348	-0.0000	-0.0000	0.1708	0.0000	-0.0000
0.5843	0.3221	0.3324	0.3310	-0.6875	0.6667	0.0000	-0.3401	-0.8528	0.7969
0.0756	0.0000	0.0000	-0.0000	0.0000	-0.0000	-0.0000	-0.5920	-0.0000	0.0000
0.0855	-0.0000	-0.0000	-0.0000	0.0000	-0.0000	0.0000	0.0000	0.2132	-0.0000
0.3684	0.4567	0.4797	0.4966	-0.1939	0.1667	0.0000	0.5102	0.4264	-0.5523
0.1798	0.0869	-0.0000	-0.2483	0.4587	-0.5000	-0.5883	-0.2033	-0.0000	0.0000
0.0644	-0.0000	-0.0000	-0.0000	-0.0000	0.0000	0.7845	0.0000	0.0000	-0.0000
0.2161	-0.0750	0.0000	0.1655	0.4724	-0.5000	-0.1961	-0.1084	-0.0000	0.0000

1.0000	0	0	0	0	0	0	0	0	0
0	-0.0835	0	0	0	0	0	0	0	0
0	0	-0.0547	0	0	0	0	0	0	0
0	0	0	-0.0000	0	0	0	0	0	0
0	0	0	0	0.7210	0	0	0	0	0
0	0	0	0	0	0.7083	0	0	0	0
0	0	0	0	0	0	0.4250	0	0	0
0	0	0	0	0	0	0	0.5100	0	0
0	0	0	0	0	0	0	0	0.5667	0
0	0	0	0	0	0	0	0	0	0.5505

根据特征值矩阵，我们可以很明确地得到平稳向量St的最终解：
St=(0.6168, 0.1998, 0.0795, 0.5843, 0.0756, 0.0855, 0.3684, 0.1798, 0.0644, 0.2161)

至此，我们得出最终的网页重要度排序：

$$P_1 > P_4 > P_7 > P_{10} > P_2 > P_8 > P_6 > P_3 > P_5 > P_9$$

即：中山大学官网 > 中山大学图书馆官网 > 中山大学新闻网 > 中山大学信息技术帮助台网 > 中山大学本科教务系统网 > 我的中大 > 中山大学百度贴吧 >中山大学学生微教务系统 > 中山大学教务部官网 > 中山大学网络与信息技术中心

其中，中山大学官网的重要度排名最高，符合我们的预期，间接验证了算法的正确性。

五【模型建立与求解】

5.1新网页排序模型背景

在前三个问题中，我们选择了网页的超链链接数及其重要性作为计算网页重要度的指标，但忽略了其他因素对网页重要度的影响，因此我们添加了网页的HTML结构、外部链接的数量、网页信任度三个指标从不同侧面进行分析，忽略了其他次要因素，构成了一个信息部分明确，部分模糊的灰色系统，因而我们采用了灰色关联分析体系，对网页重要度进行完善。

5.2新网页排序模型的建立

步骤一：

选取对网页重要度影响最大的指标——
网页超链接数量及他们的重要度作为母指标X_0。$X_0=(X_{10}, X_{20}, \ldots, X_{n0})^T$ 然后引入另外三个子指标：

1.网页HTML结构的优秀指数X_1。

2.网页所有指向外部网站的跨网站超链接总数X_2，此处包括所有外部链接**不**仅限于所考虑的网页集合中的外部链接。

3.网页可信度指标X_3。

步骤二:

由于各指标的量纲不同，数量级差别也很大，为了使用这些数据进行综合性评价，我们必须对原始数据进行无量纲，无数量级的处理，此处我们采用均值化处理：

$$X_i(j)'=\frac{X_i(j)}{\sum_{j=1}^{m}X_i(j)}\times m$$

步骤三：

计算关联系数：
$$y_i(k)=\frac{(\alpha+b\rho)}{\Delta_i(k)+b\rho}$$

其中

$$\Delta_i(k)=x_{ki}-x_{0i};$$
$$\alpha=\min_{1\le k\le n}\min_{1\le k\le n}\{\Delta_i(k)\};$$
$$b=\max_{1\le k\le n}\max_{1\le k\le n}\{\Delta_i(k)\}$$

ρ为分辨系数，一般取0.5；

步骤四：

计算关联度：

$$r_i=\frac{1}{n}\sum_{k=1}^{n}y_i(k), i=1,2,3,.....,m;$$

步骤五：

求出各指标的权重：

$$w_i=\frac{r_i}{r_1+r_2+r_3+.....+r_m}, i=1,2,3,.....,m;$$

步骤六：

构造综合评价模型：

$$Z_k=w_1\times x_{k1}+w_2\times x_{k2}+.....+w_m\times x_{km}, k=1,2,3,.....,m$$

步骤七：

将各网页的指标值代入，得到各个网页的综合得分，依据得分从大到小排序，也就得到了各网页的重要度排序。

5.3模型求解

5.3.1对各项子指标关于与中山大学有关十个网页数据的收集

1.网页的超链链接重要度X_0

根据4.2，得到:

X_0 =（0.6168, 0.1998, 0.0795, 0.5843, 0.0756, 0.0855, 0.3684, 0.1798, 0.0644, 0.2161）^{T}

2.确立网页的HTML结构优秀指数X_1

我们使用全球公认的W3C标准来评判每一个网页HTML结构的优秀程度。鉴于W3C对于一个优秀网页的标准常以评分的Error数判断，为了描述Error数量与网页重要度之间存在的负相关关系，此处不妨将优秀指数X_1设为：

$$X_1 = \sqrt{Er_{max} - Er}$$

其中Er是网页HTML检测中产生的错误数量，为了便于定量分析，结合所收集到的数据，我们把最高错误数量定为Er_{max}=150，因此我们将HTML结构优秀指数取表达式为：

$$X_1 = \sqrt{150 - Er}$$

经过对与中山大学有关的十个网站的错误数的收集，结合matlab软件，我们得到第一项子指标的数据：

X_1 =（10.4000, 11.7000, 11.9000, 12.2000, 12.1000, 11.6000, 6.2000, 11.9000, 10.4000, 12.0000）^{T}

3.确定网页中所有指向外部跨网站的超链接数X_2

我们知道，一个网站越是重要，其上线的时间也相对越长，这期间也代表了一定程度上，重要网站往往比其他网站拥有更多的指向外部的链接，使得用户能够快速地从这个网站到达下一个网站，同时也会从其他网站回到这个网站。此处，我们定义第二个子指标X_2:

$$X_2 = Li$$

其中Li就代表这个网页所有指向外部网站的跨网站超链接数。

经过使用Web脚本语言JavaScript语言编写脚本程序，我们成功地拿到了与中山大学有关的十个网站的X_2数据,详细代码参考附录7.3:

X_2 =（28.0000, 2.0000, 6.0000, 10.0000, 11.0000, 14.0000, 8.0000, 21.0000, 7.0000, 4.0000）^{T}

4.采用TrustRank Algorithm[2]，计算每个网页的可信度X_3

TrustRank Algorithm是著名的谷歌搜索引擎所采用的计算网页可信度的基本算法，该算法通过定义一个基本假设：好的网站很少会链接到坏的网站，但反之不成立。进而通过半自动的方式，加上人工挑选确立一些优质网页，设为种子网页集合Seed，通过种子集合中的网页，信任程度逐渐由高到低，其他网页的可信度往往与

其是否与Seed有超链接连通，以及它与Seed之间的距离而确定。

下面简单介绍算法的详细过程，

步骤一：在排序网页中选取可信任的网页作为Seed set，此处我们假定中山大学官网是最可信的；

步骤二：基于已明确的可信任网页，得到平稳向量 r_k；

$$r_0^T = v^T$$

$$当 k \le M, 有 r_k^T = \alpha(r_{k-1}H) + (1-\alpha)v^T;$$

结合Matlab软件，我们计算出当 $\alpha = 0.85$，$k = 20$ 时，十个网页的信任度指标:

X_3 =（0.2123, 0.0863, 0.0184, 0.1656, 0.1531, 0.0000, 0.1045, 0.0482, 0.1304, 0.0812）^{T}

以下是**与中山大学有关的**十个网站的TrustRank Algorithm各项数据，详细代码**参考**附录７．4

0.1250	0	0.1250	0.1250	0.1250	0.1250	0.1250	0.1250	0.1250	0
0	0.6000	0.2000	0	0.2000	0	0	0	0	0
0	1.0000	0	0	0	0	0	0	0	0
0.2500	0	0	0.7500	0	0	0	0	0	0
0	0	0	0	1.0000	0	0	0	0	0
0	0	0	0	0	1.0000	0	0	0	0
0.5000	0	0	0	0	0	0.5000	0	0	0
0	0	0	0	0	0	0	0.5000	0	0.5000
0	0	0	0	0	0	0	0	1.0000	0
0	0	0.2500	0	0	0	0	0.2500	0.2500	0.2500

(9)TrustRank Algorithm中超链接矩阵的转置U

0.3333	0	0	0.3333	0	0	0.3333	0	0	0
0	0.7500	0.2500	0	0	0	0	0	0	0
0.3333	0.3333	0	0	0	0	0	0	0	0.3333
0.2500	0	0	0.7500	0	0	0	0	0	0
0.2000	0.2000	0	0	0.6000	0	0	0	0	0
0.3333	0	0	0	0	0.6667	0	0	0	0
0.5000	0	0	0	0	0	0.5000	0	0	0
0.3333	0	0	0	0	0	0	0.3333	0	0.3333
0.2500	0	0	0	0	0	0	0	0.5000	0.2500
0	0	0	0	0	0	0	0.5000	0	0.5000

（９）TrustRank Algorithm中的关系逆的超链接矩阵H

```
请输入种子集合的元素个数: 3
     5     9     6

请输入good种子的个数: 2
         0         0         0         0    0.5000         0         0         0    0.5000         0

    0.2123    0.0863    0.0184    0.1656    0.1531         0    0.1045    0.0482    0.1304    0.0812
```

(11)人工选择种子个数，以及最终的可信度排序

5.3.2灰色关联分析法的实现步骤：

步骤一：

选取对网页重要度影响最大的指标，即我们的第一部分中主体变量，网页的超链接数量及其权重作为母指标X_0，而其他指标作为子指标。从而构建出原始数据，如下图。

0.6168	0.1998	0.0795	0.5843	0.0756	0.0855	0.3684	0.1798	0.0644	0.2161
10.4000	11.7000	11.9000	12.2000	12.1000	11.6000	6.2000	11.9000	10.4000	12.0000
28.0000	2.0000	6.0000	10.0000	11.0000	14.0000	8.0000	21.0000	7.0000	4.0000
0.2374	0.0575	0.0122	0.1853	0.1020	0.1154	0.1169	0.0322	0.0870	0.0541

步骤二：

均值化处理后得到的数据

2.4970	0.8088	0.3218	2.3654	0.3060	0.3461	1.4914	0.7279	0.2607	0.8748
0.9420	1.0598	1.0779	1.1051	1.0960	1.0507	0.5616	1.0779	0.9420	1.0870
2.5225	0.1802	0.5405	0.9009	0.9910	1.2613	0.7207	1.8919	0.6306	0.3604
2.3740	0.5750	0.1220	1.8530	1.0200	1.1540	1.1690	0.3220	0.8700	0.5410

步骤三：

计算关联系数：

0	0	0	0	0	0	0	0	0	0
1.5549	0.2509	0.7561	1.2603	0.7900	0.7046	0.9298	0.3500	0.6813	0.2121
0.0256	0.6287	0.2187	1.4645	0.6849	0.9151	0.7707	1.1640	0.3699	0.5145
0.1230	0.2338	0.1998	0.5124	0.7140	0.8079	0.3224	0.4059	0.6093	0.3338

(12) $\Delta_i(k)$ 数据

1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000	1.0000
0.3333	0.7560	0.5070	0.3815	0.4960	0.5246	0.4554	0.6896	0.5330	0.7856
0.9682	0.5529	0.7805	0.3468	0.5316	0.4593	0.5022	0.4005	0.6776	0.6018
0.8634	0.7688	0.7955	0.6028	0.5213	0.4904	0.7069	0.6570	0.5606	0.6996

(13)y数据

步骤四：

计算关联度：

1.0000　　0.5462　　0.5821　　0.6666

步骤五：

求出各指标的权重：

0.3578　　0.1954　　0.2083　　0.2385

步骤六：

构造综合评价模型：

$$Z_k = 0.3578X_1 + 0.1954X_2 + 0.2083X_3 + 0.2385X_4$$

步骤七：

将各网页的指标值代入，得到各个网页的综合得分：

2.1691　0.6712　0.4675　1.6919　0.7734　0.8671　1.0723　0.9419　0.6162　0.7295

依据得分从大到小排序，也就得到了各网页的重要度排序：

中山大学官网 > **中山大学**图书馆**官网** > **中山大学**新闻网 > 我的中大 > **中山大学百度**贴吧 > **中山大学**教务部**官网** > **中山大学**信息技术帮助台 > **中山大学**本科教务系统 > **中山大学**网络与信息技术中心 > **中山大学学生**微教务**系**统

详细代码参考附录7.5

与算法修改前相比，**中山大学百度**贴**吧**的重要度明显上升，这是因为它作为用于商业用途的网页，具有更优化的网页结构与更丰富的网页内容，符合我们的预期，客观上验证了算法的正确性。

六【参考文献】

[1] TH Haveliwala ,Topic-sensitive PageRank: a context-sensitive ranking algorithm for Web search, Knowledge & Data Engineering IEEE Transactions on , 2003, 15(4): 784-796

[2] Gyongyi, Garcia-Molina, and Pederson, Combating Web Spam with TrustRank, in Proc.,VLDB 2004

七【附录】

7.1针对附件一数据，针对问题一的初步算法Matlab代码

```
%%创建9个网页互相指向的超链接矩阵%%
store = [0,0,1,0,2,2,0,3,1;
         0,0,0,0,0,0,3,0,0;
         2,0,0,0,3,1,0,1,0;
         0,1,0,0,0,0,4,0,0;
         1,0,1,0,0,1,0,1,0;
         1,0,2,0,1,0,0,1,0;
         1,2,0,1,1,0,0,0,0;
         1,0,1,0,0,0,0,6,0;
         0,0,0,0,0,0,0,0,0];

%%初始化重要度结果数组%%
result = zeros(1, 9);
%%超链接矩阵的初始化%%
H = zeros(9, 9);
```

```
%%第i个网页的总超链接数%%
link = zeros(1, 9);
%%初始化谷歌矩阵%%
G = zeros(9, 9);
%%选定谷歌系数a为0.85%%
a = 0.85;
%%设定全一矩阵%%
J = ones(9, 9);

%%求解重要度算法部分%%
%%首先得到超链接矩阵%%
for i=1:9
    for j=1:9
    if(store(i, j)>0)
        link(i) = link(i)+store(i, j);
    end
    end
end

for i=1:9
    for j=1:9
        if(store(j, i)>0)
            H(i, j) = H(i, j)+store(j, i)/link(j);
        end
    end
end
%%将超链接矩阵进行初步修正为随机矩阵%%
for i=1:9
    H(i, 9) = 1/9;
end
%%谷歌矩阵求解%%
G = a*H + J*((1-a)/9);
%%谷歌矩阵的特征值以及特征向量,V为特征向量，C为对应的特征值%%
[V, C] = eig(G);
%%结果矩阵的输出%%
disp(store);
disp(H);
disp(G);
disp(V);
disp(C);
```

7.2可用于nXn矩阵数据的输入，初步算法的Matlab代码

```
%% 创建n个网页之间的互相指向的链接数矩阵 %%
```

```
store = xlsread('data2.xlsx');
n = input('请输入网页的数量: ');

%%初始化重要度结果数组%%
result = zeros(1, n);
%%初始化各个矩阵%%
H = zeros(n, n);
link = zeros(1, n);
G = zeros(n, n);
%%选定谷歌系数%%
a = 0.85;
J = ones(n, n);
Zero_c = zeros(n, 1);

%%求解重要度算法部分%%
%%首先得到超链接矩阵%%
for i=1:n
    for j=1:n
    if(store(i, j)>0)
        link(i) = link(i)+store(i, j);
    end
    end
end

for i=1:n
    for j=1:n
        if(store(j, i)>0)
            H(i, j) = H(i, j)+store(j, i)/link(j);
        end
    end
end
%%检验得到的结果超链接矩阵，使用随机矩阵优化%%
for i=1:n
    if(isequal(Zero_c, H(:, i)))
        H(:, i) = H(:, i) + 1/n;
    end
end
%%谷歌矩阵求解%%
G = a*H + J*((1-a)/n);
%%谷歌矩阵的特征值以及特征向量,V为特征向量，C为对应的特征值%%
[V, C] = eig(G);

disp(store);
disp(H);
```

```
disp(G);
disp(V);
disp(C);
```

7.3查找网页超链接总数，以及计算网页到其他界面的链接数JavaScript代码

```
//找到网页上所有的超链接
var all_alink = document.getElementsByTagName("a");
//定义10个的网页集合
var webpages = [ 'http://www.sysu.edu.cn/2012/cn/index.htm',
      'http://uems.sysu.edu.cn/jwxt/', 'http://wjw.sysu.edu.cn/',
      'http://library.sysu.edu.cn/', 'http://jwb.sysu.edu.cn/',
      'http://tieba.baidu.com/f?kw=%D6%D0%C9%BD%B4%F3%D1%A7&fr=ala0&t
      pl=5',
      'http://news2.sysu.edu.cn/','http://my.sysu.edu.cn/welcome',
      'http://inc.sysu.edu.cn/',
      'http://helpdesk.sysu.edu.cn/'];
//得到的结果
var result = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0];
//定义当前网页的order，即该网页在网页集合中的index下标
var order;

//查找过程
for(var i=0;i<all_alink.length;i++){
      if(all_alink[i].href==='/'||all_alink[i].href==='/index'
      ||all_alink[i].href==='index.htm'||all_alink[i].href===webpages
[order]+'index.ht ml')
       result[order]++;
      else {
            for(var j=0;j<=9;j++)
      if(all_alink[i].href===webpages[j]||all_alink[i].href===webpage
s[j].substr(0,23)
                        ||all_alink[i].href===webpages[j].substr(0,
webpages[j].length-1)

      ||all_alink[i].href===webpages[j]+'index.htm')
                  result[j]++;
      }
}

//存储所得到的超链接结果
var a = [];
for(var i=0;i<all_alink.length;i++)
   a.push(all_alink[i].href);
```

7.4 TrustRank Algorithm的Matlab实现

```
%%TrustRank Algorithm%%

%%各项数据矩阵的初始化%%
store = xlsread('data2.xlsx');
store_inv = store';
n = input('请输入网页的个数: ');
H = zeros(n, n);
U = zeros(n ,n);
link = zeros(1, n);
link_v = zeros(1, n);

%%每个网页超链接数量总和的计算%%
for i=1:n
    for j=1:n
        link(i) = link(i) + store(i ,j);
        link_v(i) = link_v(i) + store_inv(i, j);
    end
end

for i=1:n
    for j=1:n
        %%经过矩阵转置之后，超链接权重矩阵的构建%%
        if(link_v(i)~=0)
            U(i, j) = store_inv(i, j)/link_v(i);
        end
        %%初始超链接权重矩阵的构建%%
        if(link(i)~=0)
            H(i, j) = store(i, j)/link(i);
        end
    end
end

disp(U);
disp(H);

%%定义元行向量，以及对应的谷歌系数的确立%%
E = ones(1, n);
S = E;
a = 0.85;

%%采用递归的方法精确化最终的Seed向量，这里采用了20作为递归次数%%
```

```
for i=1:20
    S = a.*(S*U)+(1-a).*E/n;
end

%%将得到的结果进行排序，确立种子集合%%
S_s = sort(S, 'descend');
disp(S_s);

m = input('请输入种子集合的元素个数: ');

Seed = zeros(1, m);
for i=1:m
    for j=1:n
        if(S(j)==S_s(i))
            Seed(i) = j;
        end
    end
end

disp(Seed);
%%对种子集合中的good or bad种子的确定%%
k = input('请输入good种子的个数: ');
good = Seed(1, 1:k);
v = zeros(1, n);
%%建立good种子行向量%%
for i=1:k
    v(good(i)) = 1/k;
end

%%对最终的可信度行向量的建立，同样采用递归的算法实现%%
r = v;
for i=1:20
    r = a*(r*H)+(1-a)*v;
end

disp(v);
disp(r);
```

7.5灰色关联分析建模代码实现

```
store = xlsread('end_data.xls');

n = input('请输入网页的个数: ');
average = zeros(1, 4);
```

```
%%首先建立各项指标的均值数组%%
for i=1:4
    for j=1:n
        average(i) = average(i) + store(i, j);
    end
    average(i) = average(i) / n;
end

disp(average);
%%将原始数据进行均值化处理%%
for i=1:4
    for j=1:n
        store(i, j) = store(i, j)/average(i);
    end
end
disp(store);
%%确定关联系数的各项系数%%
%%首先是det的矩阵%%
det = zeros(4, n);
a = 10000000;%%最小值%%
b = -1000000;%%最大值%%
y = zeros(4, n);

%%建立deta矩阵%%
for i=1:4
    for j=1:n
        det(i, j) = abs(store(1, j)-store(i, j));
        if(det(i, j)<a)
            a = det(i, j);
        end
        if(det(i, j)>b)
            b = det(i, j);
        end
    end
end

disp(det);
%%求解各项指标中的关联系数%%
for i=1:4
    for j=1:n
        y(i, j) = (a+b*0.5)/(det(i, j)+b*0.5);
    end
end
```

```
%%求解最终的关联系数之和%%
r = zeros(1, 4);
sum = 0;
for i=1:4
    for j=1:n
        r(i) = r(i) + y(i, j);
    end
    r(i) = r(i)/n;
    sum = sum + r(i);
end
disp(r);

%%得到最终各项指标的weight权值%%
w = zeros(1, 4);
for i=1:4
    w(i) = r(i)/sum;
end

disp(w);

%%量化最终带权值的各项指标的重要度结果%%
final = zeros(1, n);
for i=1:n
    for j=1:4
        final(i) = final(i) + w(j)*store(j, i);
    end
end

disp(final);
```