An Improved Website Structure Optimizing Algorithm

Mingjun Li[1], Mingxin Zhang[2], Jinlong Zheng[2] and Ying Lu[1]

[1]School of Computer and Information Engineering,
Harbin University of Commerce, Harbin, 150028, China
[2]School of Computer Science & Engineering, Changshu Institute of Technology,
Changshu, 215500, China
wdbjwl@126.com, mxzhang163@163.com

Abstract

The improved algorithm selects expected locations (for adding navigation links) in backtracks set at the point of the earlier and the less backtracks, which avoids effectively negative impact to the accuracy of the overall analysis by the long access sequence. The experimental results show that the improved algorithm can find expected pages effectively, thus can achieve the target of adjustment and reorganization of website.

Keywords: *web log mining; expected page; website structure; navigated path*

1. Introduction

With the explosive growth of the number of website in the whole world, the information capacity and complexity of each site are also increasing rapidly. This adds the difficulty for browsing, because visitors can't find the location of page which they expect to access in thousands of links. So how to arrange the structure of website properly so is a crucial task for visiting effectively and accurately. Besides, the diversity of websites (including B2C, B2B, business site and directory sites, *etc.*), making the problem more complicated. Site structure optimization technique is based on the visitor's browsing behavior, optimizing the site structure and making the users visit the site which they expected quickly and accurately in order to reduce the time users' access. Visitors can significantly reduce the "unnecessary" clicks; reach the target page quickly, thereby reducing the number of requests to Web servers to reduce the burden on the server [1].

There is a contradiction between the actual structure of websites and the user's usage habits: the customer always search the location of pages where their expected along with the site's hierarchical structure, but actually the target pages appear in other positions [8-10], Ramakrishnan Srikant present an algorithm to automatically find pages in a website whose location is different from where visitors expect to find them, the key point of the algorithm is: visitors will backtrack if they do not find the page where they expect it: the point from where they backtrack is the expected location for the page. Expected locations with a significant number of hits are presented to the website administrator for adding navigation links from the expected location to the target page.

In this paper, an improved algorithms for selecting the set of navigation links is presented, which mainly optimizes the way to choose navigation links at the point of the earlier and the less backtracks, takes into account that users might try multiple expected locations for a target page.

2. Related Works

2.1. Website Structure Optimization Model：Website structure optimization system model generally consists of four main stages: data acquisition, data preprocessing, and pattern discovery and pattern analysis, shown in Figure 1:

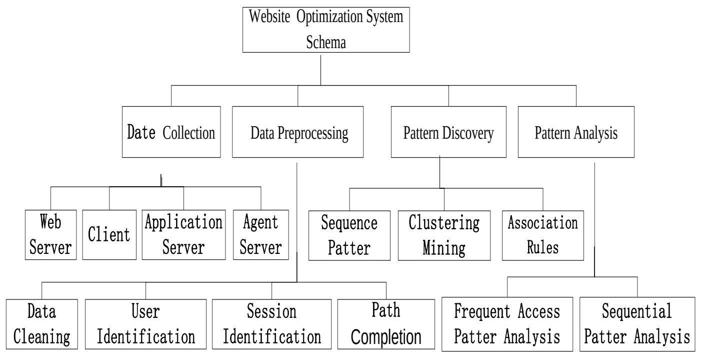

Figure 1. The Basic Mode of Website Structure Optimization

2.2. Visitors Browse Mode: website pages can be viewed as a tree hierarchy structure, in general, Consider the case where the visitor is looking for a single specific target page 7. We expect the visitor to execute the following search strategy:

1）Start from the root 1A.

2）While (current location C, C is not the target page T) do

If any of the links from C , C seem likely to lead to 1A,follow the link that appears most likely to lead to1A

Else, either backtrack and go to the parent of C, with some (unknown) probability, or give up with some probability.

Example 1: Figure 2 shows a hierarchically organized website, the visitor is looking for a single target page 7. She starts at the root, expects to find page 7 under 2A, goes to 2A and then 3A, realizes that 7 is not under 3A, backtracks to 2A, tries 3B,realizes that 3B does not contain 7 either, backtracks all the way to the root, and finally finds 7 under 2C. The expected locations for target page 7 are 3A and 3B, and the actual location is 2C.So finally gets a traversal path {1A, 2A, 3A, 2A, 3B, 2A, 1A, 2C, 7} If pages are cached in the browser, the web log will only contain{1A,2A, 3A, 3B, 2C, 7}, the Access pattern shown in Figure 2:

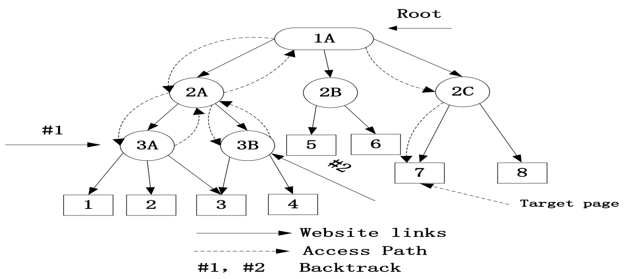

Figure 2. The Access Pattern

2.3. The Discovery of Expected Location: Ramakrishnan Srikant, *et al.*, [2] proposed propose an algorithm to automatically find pages in a website whose location is different from where visitors expect to find them. The algorithm can also find the expected location of these pages named expected page, in this algorithm, the way for getting backtrack points page in a traversal path is based on the fact that if there is no link between pages P1 and P2, the visitor must have hit the "back" button in the browser to go from P11 to P2, so P1 is just the backtrack points page which we need.

By this way the algorithm first generates a set of backtrack pages, and then selects the proper page which can be viewed as the expected page, For each visitor and target page, the set of backtrack pages generated by the corresponding access subsequence can be defined as follows:

Let $\langle T, (BK_1, BK_2, \ldots BK_n), \mathrm{Pre}\rangle$ be the set of backtrack pages, T is target page, $\{BK_1, BK_2, \ldots BK_n\}$ is the sequence of backtrack points; Pre is the actual location of the target page. In this example, we choose five group sets of backtrack which are generated by the visitors when to search for the target page T.

Table 1. The Corresponding Backtrack Set

Search records	Backtrack location sequence			
	BK_1	BK_2	BK_3	BK_4
Record$_1$	P_1	P_2		
Record$_2$	P_1			
Record$_3$		P_3	P_4	P_1
Record$_4$	P_3	P_2		
Record$_5$	P_4	P_2		

The algorithm of choosing the proper expected pages named Optimize Benefit: was also proposed by Ramakrishnan Srikant, it can be described as follows: Recommend the set of pages that optimize benefit to the website, where benefit is estimated based on the fraction of people who might give up on not finding a page. This is a greedy algorithm that attempts to maximize the benefit to the website of adding additional links.

The algorithm can be described as follows:

```
For i=1 to m // m is number the backtrack pages
    Support $(P_i)$=0
    for j=1 to n do // n is the number of records
      Calculate Benefit $(BK_{ij})$

// $BK_{iJ}$ is backtrack page, calculate the benefit of finding the target page in the $P_i$ the expected location
    Support $(P_i)$= Support $(P_i)$+ Calculate Benefit $(BK_{ij})$
```

For P1 in record1 shown in Table 1,if adding the target links to P1,the visitor can avoid visitor's two backtracks(P1, P2), in case of record2,the backtrack P1 can also be avoid, so:

$S(P_1) = 2+1+1 = 4$ $\quad S(P_2) = 1+4+1+1 = 7$

$S(P_3) = 3+2 = 5$ $\quad S(P_4) = 2+2 = 4$

The support of P_2 is lowest, so P_2 is viewed expected page, drop P_2 and all items that follow P_2, resulting in the following Table 2:

Table 2. Corresponding Backtrack Set After P_2 Was Dropped

Search records	Backtrack location sequence			
	BK_1	BK_2	BK_3	BK_4
Record_1	P_1			
Record_2	P_1			
Record_3				
Record_4	P_3			
Record_5	P_4			

3. The Improvement of Website Structure Optimizing Algorithm

3.1. The New Algorithm

3.1.1. As little as possible backtracks: Since we added the hint from these backtrack pages to the destination page are by choosing some of the pages, reduced the length of the access path, from statistical view, so which page can be used to set hinge the target page to reduce the number of our back-tracks is a very important factor.

3.1.2. As early as possible backtracks: For finding a particular page, if the previous node traversed in the path has a link to the target page, it would be more effective to improve search efficiency, and therefore, for the improvement of site structure, the front of the nodes should have more influence.

Based on the above two different angle, we proposed the improved algorithm as follow:

```
For i=1 to n// n is number the backtrack records
    For j=1 to Count (RDi)   // j is backtrack point in every records
Adjust Benefit (BKij)
```

Adjust Benefit(BKij) is a function which adjusts way that calculates the benefit of finding the target page in the Pi th expected location, supposed Pm is the correspond page of BKij, then the concrete adjust algorithm is shown as follow:

```
for $(k=1, n=0; n < Count(RD_j) - j + 1; n++)$
        $TS(P_m) = TS(P_m) + 1 \times \varphi(k++)$
```

k is the order of page in backtrack sequence(set the order of the page P_m is zero), the page weighted function $\varphi(k)$ should meet the character that it's value is reduced with the increase of k(because the former backtrack page links added to the target page behind has the better efficiency for system performance than the behind one),as for which the weight function used can better close to the actual situation in web access, we chose the following typical function to discuss:

$$\varphi(k) = \tau^{-k} \text{ (τ is a constant and } k \in n\text{)} \tag{1}$$

$$\varphi(k) = \log(k) = c \text{ (c is a constant and } k \in n\text{)} \tag{2}$$

$$\varphi(k) = -k^{-c} + b \text{ (c is a constant and } k \in n\text{)} \quad (3)$$

In the trends of the change of functions' value with the K's increase, τ^{-k} is the closest to the actual situation, so finally we choose τ^{-k} as a weighting function, finally set the value of empirical parameters 2, because the function with the parameter 2 can reflect effectively the characters of the less and the earlier backtracks.

3.2. Algorithm Analysis

In terms of the access sequence of each user, we define the benefit value TS of each page in backtrack set if add the links from the page to target page as the sum of benefit value of each access sequence, the formulator can be shown:

$$TS(P_s) = \omega_1(P_s) + \omega_2(P_s) + \cdots + \omega_n(P_s) \quad (4)$$

m is the number of backtrack set corresponded by each access sequence, for each access record, $\omega_i(P_s)$ is an accumulative total from the beginning page to the end, in general, an access sequence $RD(P_1, \cdots, P_s, P_{s+1}, P_t)$ $1 \le s \le t$, P_t is the last backtrack point.

$$\omega_i(P_s) = 1\times 2^{-(s-1)} + 1\times 2^{-s} + 1\times 2^{-(s+1)} + \cdots + 1\times 2^{-(t+1)} \quad (5)$$

The formulator can be transformed:

$$\omega_i(P_s) = 4\times 2^{-s} - 2^{-(t-1)} \quad (6)$$

The above analysis shows that the method used in this paper takes into account the problem of the length of access path, reduces the impact to make decision because of a few long access sequences. It means that the new algorithm can avoid effectively negative impact to the accuracy of the overall analysis by the long access sequence.

3.3. Experiment and Example

We used the web server log from http://www.nwnu.edu to evaluate and compare the difference in performance between the new algorithm and old. Table 3 shows five examples the backtrack set.

Table 3. The Corresponding Backtrack Set

Search records	***Backtrack location sequence***			
	BK₁	***BK₂***	***BK₃***	***BK₄***
Record$_1$	P_1	P_2		
Record$_2$	P_1			
Record$_3$	P_5	P_3	P_4	P_2
Record$_4$	P_3	P_2		
Record$_5$	P_4	P_2		

1) apply the algorithm proposed by Ramakrishnan Srikant to analysis:

$S(P_1) = 2+1 = 3$ $\quad S(P_2) = 1+1+1+1 = 4$
$S(P_3) = 3+2 = 5$ $\quad S(P_4) = 2+2 = 4$
$S(P_5) = 4$

Set Support=5 drop P_3 and all items that follow P_3, resulting in the following table:

Table 4. Corresponding Backtrack Set After P_3 Was Dropped

Search records	*Backtrack location sequence*			
	BK₁	***BK₂***	***BK₃***	***BK₄***
$Record_1$	P_1	P_2		
$Record_2$	P_1			
$Record_3$	P_5			
$Record_4$				
$Record_5$	P_4	P_2		

2) use the new algorithm

$$TS(P_1) = (1\times1+1\times0.5)+1\times1 = 2.5$$
$$TS(P_2) = 1\times0.5+1\times0.125+1\times0.5+1\times0.5 = 1.625$$
$$TS(P_3) = (1\times0.5+1\times0.125+1\times0.125)+ (1\times1+1\times0.5) = 2.375$$
$$TS(P_4) = 1\times0.25+1\times0.125+1\times1+1\times0.5 = 1.875$$
$$TS(P_5) = 1\times1+1\times0.5+1\times0.25+1\times0.125 = 1.875$$

Set Support=2.5 drop P_1 and all items that follow P_1, resulting in the following table:

Table 5. Corresponding Backtrack Set After P_1 Was Dropped

Search records	*Backtrack location sequence*			
	BK₁	***BK₂***	***BK₃***	***BK₄***
$Record_1$				
$Record_2$				
$Record_3$	P_5	P_3	P_4	P_2
$Record_4$	P_3	P_2		
$Record_5$	P_4	P_2		

If we use the algorithm proposed by Ramakrishnan Srikant, P_3 will be the expected page, while applying the algorithm we proposed, P_1 is expected page. It can be discovered from the final results that the new algorithm takes into account simultaneously the earlier and the less backtracks, and the result is not be affected by the few long access path (for example RD3 in this instance).Because there are long access path existed in actual situation, the visitors are likely to find a specific page rather than a series of related content pages in the view.

4. Summary and Future Directions

We proposed an algorithm to automatically discover pages in a website whose location is different from where visitors expect to find them. This problem of matching website organization with visitor expectations is pervasive across most websites. Our key insight is that select the set of navigation links (to add to expected locations) to optimize visitor time or benefit to the website.

An interesting problem for future research is that in websites without a clear separation of content and navigation, it can be hard to differentiate between visitors who backtrack because they are browsing a set of target pages, and visitors who backtrack because they are searching for a single target page. It will be interesting to explore if there are better approaches to solve this problem.

Acknowledgements

The authors like to thank Professor Mingxin Zhang, for his invaluable help. We would also like to thank an anonymous referee for his useful comments on the improvement of our presentation.

References

[1] S. Cheng and C. Xu, "The Progress of Website Structure Optimization Techniques", Application Research of Computers, vol. 6, no. 26, **(2009)**.
[2] R. Srikant and Y. Yang, "Mining Web Logs to Improve Website Organization", Proceedings of the 10th International conference on World Wide Web, **(2001)**, Hong Kong.
[3] Y. Bao, X. Huang and Z. Zhang, "Website Structure Optimization Method Based On Web Log", Computer Engineering, vol. 12, no. 29, **(2003)**.
[4] M. -S. Chen, J. S. Park and P. S. Yu, "Data mining for path traversal patterns in a web environment", The 16th IEEE International Conference on Distributed Computing Systems, **(1996)**.
[5] M. Kang and D. Cho, "Adaptive web site construction using ART", Proceedings of IEEE International Symposium on Industrial Electronics, **(2011)**, Pusan, Korea.
[6] W. -l. Lin and Y. -z. Liu, "A novel website structure optimization model for more effective Web navigation", Proceeding of the 1st International Workshop on Knowledge Discovery and Data Mining, **(2008)**, Adelaide.
[7] H. P. Jian, B. Mortazavi-asl and H. Zhu, "Mining access patterns efficiently from web logs", Proceeding of the 4th Pacific-Asia Conf. on Knowledge Discovery and Data Mining, **(2000)**, Kyoto, Japan.
[8] T. Nakayama, H. Kato and Y. Yamane, "Discovering the gap between web site designers expectations and users' behavior", Proc. of the Ninth International World Wide Web Conference, **(2000)**, Amsterdam.

Authors

Mingjun Li, associate professor, received the B.S degree in computer science from the Northeastern University, Shenyang, China, in 1982. His research interests include pattern recognition; The main research directions: information management, business automation, ERP systems, logistics and distributed computing.

Mingxin Zhang received the B.S degree in computer science from the Northeastern University, Shenyang, China, in 1982, the M.S degree in general traffic design from the Xi'an University of Architecture and Technology, Xi'an , China, in 1990, and the Ph.D. degree in software and theoretical science from the Xi'an Jiaotong University, Xi'an, China, in 2008. He has taught Computer Science at Northwest Normal University and Gansu Agricultural University as a professor. He is currently a professor of Computer Science at Changshu Institute of Technology, China. His research interest include database and system integration , intelligent control system based on network, graphic image and video information retrieval , data mining and knowledge discover.

Jinlong Zheng received the B.S degree at College of Computer Science and Engineering, Northwest Normal University. His research interests include pattern recognition and image processing.

Ying Lu, Master, work in Computer and Information Engineering, Harbin University of Commerce, China; lecturer. The major research fields: artificial intelligence, information services; e-commerce and e-government.