STUDIA UNIV. BABEŞ–BOLYAI, INFORMATICA, Volume **XLV**, Number 2, 2000

AUTOMATIC SUPPORT FOR IMPROVING INTERACTION WITH A WEB SITE

ALINA CÂMPAN AND DARIUS BUFNEA

Abstract. In this paper we describe a method to make a Web site easier navigable by its users. In the same time, this method provides support for the Webmaster to raise the quality of the site with minimal effort. These goals are achieved by automatic creation of *orientation Web pages*. The new adds-on to the site are generated by exploiting the data accumulated in Web server access logs, being thereby a feedback to the users' "footprint". Web mining techniques are used in order to extract the meaningful information from log data.

1. Introduction

For a Web site, to be appreciated by its visitors, it is significant not only its visual aspect, the interest of the information or/and the quality of services it offers! It also counts, in a great extent, how easy the users retrieve within the site the information they are interested in. If this retrieval involves searching in long chains of unclearly linked documents, it is very likely that the user will give up searching, leave the site and possibly never come back. In the case of a company that sales its products or services on the Internet, this will mean loosing clients, clearly an undesired effect.

As defined in [7], the quality of a Web site is lower as the user's effort to find the pages that match his area of interest is growing. Most often this effort is measured as a count of links followed by the visitor until he finds the desired information. It is more realistic to assume this effort as being a trade-off between the effort to choose in every visited page the link to follow next, and the number of visited pages. We are interested in reducing this effort.

We must also note that maintaining a complex Web site can be a difficult task. The Webmaster has to face several challenges:

- The site has to contain up-to-date information;

2000 *Mathematics Subject Classification.* 68T05,68T10,68U35.

1998 *CR Categories and Descriptors.* I.5.2. [**Computing Methodologies**] : Pattern Recognition – *Design Methodology*; I.5.3. [**Computing Methodologies**] : Pattern Recognition – *Clustering*; H.3.3. [**Information Systems**] : Information Storage and Retrieval – *Information Search and Retrieval.*

- The users may seek different information at different times, and the site must be structured in a way to permit easy access, whatever the visitors' goals may be.

So, a Web site is a dynamic structure, its design may be object to changes in time. These changes materialize in new pages and links, added sometime in unlikely places.

Our purpose is to come in response to the Webmaster needs, by helping him to maintain a good quality site for the users (quality as we talked about few paragraphs above). We are entitle to sustain that the solution we shall describe does help to improve the interaction with the Web site, both for the Webmaster and, consequently, for the visitors.

Previous work. The problem of *adaptive Web sites* — "sites that automatically improve their organization and presentation by learning from visitor access patterns" — was enounced in the AI community ([5]). There are known two ways of addressing this problem. One is the *customization* of the Web site (we will not refer to this). The other, more recent, approach is the *optimization* of the site's structure to make it easier to use for all visitors. This is the trend followed by the authors in [5, 6, 7]. More precisely, they investigate the data accumulated in Web server access logs and identify a number of cohesive, possibly overlapping clusters of pages that they conclude, based on users access patterns detected in logs data, that are related to a particular topic. For some of these clusters are synthesized index pages which contain a link for every page in that cluster. The methods used in the above mentioned papers are AI traditional clustering techniques adapted to the specific of the problem.

In this paper, we proceed similarly as in [5, 6, 7]. Namely, we want to create *orientation pages* with links to the related-by-content pages of the site. But we propose a different manner to partition the site: using Web mining methods instead of AI clustering techniques. Also, this partitioning will be made with more accuracy, as we shall see.

The point to start from is raw Web server data. Taking into account a user browsing behavior model (proposed in [3]), we separate important *content page* references from references used for *navigational* purposes. Than we construct *content transactions* that correspond to the content pages visited by a user in one session. The obtained transaction repository is than mined for association rules with Web mining algorithms. Pages in every association rule give us a cluster for which we can synthesize orientation pages. Keeping in view that we try to facilitate the access to the site, and not to overhead it, we develop orientation pages only for those clusters which pages are not already linked in the site [5].

2. Problem description

We reach our goal of synthesizing orientation pages in three steps, as we said above. Each one of these steps is detailed next in one paragraph.

2.1. **Content Transactions Identification.** In order to group into transactions the elementary page references which a Web server access log contains, we consider the following user behavior model. First, we make a visit coherence assumption, which states that the pages a user visits during one visit *session* tend to be *conceptually related* [5]. Even if this is not always a valid assumption, accumulating statistics over long periods of time and for many users will reduce the noise till extinguish. Secondly, during a visit of a site, a user treats the pages either as *content* pages, either as *navigational* pages. We accept as content pages those pages with information the user is interested in. The pages where he looks for links to the desired data are considered navigational ones.

To meet our goals we shall need to identify *content transactions*, from the log data. By a content transaction we mean all of the content references a user makes in one session. Mining these content transactions will produce the associations between the content pages of the site, therefore the *clusters of pages within the site related by their content*; so, we do more than just find the most popular navigational paths and the pages these paths consist in.

We introduce now the notions we need and describe formally how to find content transactions.

Let L be the set of Web server access log entries completed with user identification information. An entry $l \in L$ includes the client IP address $l.ip$, the client user id $l.uid$, the URL of the accessed page $l.url$ and the time of access $l.time$. Local browser cache, masquerading and proxy servers can distort the accuracy of the data collected by the Web server and make user identification a difficult to accomplish task. Some solutions to user identification problem are given in [8].

We order the log entries after $l.uid$ and $l.time$ and we develop first a repository of general transactions.

Definition 1. *A* general transaction t *is a triple:*

$$t = \langle ip_t, uid_t, \{(l_1^t.url, l_1^t.time), \ldots, (l_m^t.url, l_m^t.time)\}\rangle$$

where $l_k^t \in L, l_k^t.ip = ip_t, l_k^t.uid = uid_t, k = 1, \ldots, m.$

From the general transactions, we identify the set of reference length transactions contained in the log data.

Definition 2. *A* reference length transaction t *is a triple:*

$$tr = \langle ip_{tr}, uid_{tr}, \{(l_1^{tr}.url, l_1^{tr}.time, l_1^{tr}.length), \ldots, (l_m^{tr}.url, l_m^{tr}.time, l_1^{tr}.length)\}\rangle$$

where $l_k^{tr} \in L, l_k^{tr}.ip = ip_{tr}, l_k^{tr}.uid = uid_{tr}, k = 1, \ldots, m,$
and $l_k^{tr}.lenght = l_{k+1}^{tr}.time - l_k^{tr}.time, k = 1, \ldots, m-1.$

We make some observations regarding the above definition.

Obviously, the last reference in each general transaction has no next time to use in determining the reference length. We assume that all of the last references are content ones and their length is, say, one hour (in [3], they are also excluded in the process of calculating the cut-off time). From one general transaction, we can generate one or more reference length transactions, as follows. During a user visit may appear large amounts of time between two page references. In case of such interruptions in user's navigation, longer than a threshold $TMax$ we establish, we decide to break the initial general transaction in two or more smaller reference length transactions, for which every reference (except the last one) is shorter than $TMax$. This makes sense, because resuming a visit after a long inactivity may be very well interpreted as the beginning of a new *session*. Therefore, we complete definition 2 with the following condition:

$$l_k^{tr}.length < TMax, k = 1, \ldots, m-1 \text{ and } l_m^{tr}.length \geq TMax.$$

Different users can use the same page in different manners, which are for navigational or for content purposes. We need to differentiate between these two alternatives. In most cases it is not possible to categorize a page based on its content; it is more realistic to make the distinction based on how much time the visitor spends on the page. A cut-off threshold between the medium time associated with the navigation references and the content references can be assumed or calculated — one possibility is mentioned in [3]. We denote this cut-off time by C. Having this cut-off time, we define a content transaction as follows:

Definition 3. *A* content transaction *t is a triple:*

$$tc = \langle ip_{tc}, uid_{tc}, \{(l_1^{tc}.url, l_1^{tc}.time, l_1^{tc}.length), \ldots, (l_m^{tc}.url, l_m^{tc}.time, l_m^{tc}.length)\}\rangle$$

where $l_k^{tc} \in L, l_k^{tc}.ip = ip_{tc}, l_k^{tc}.uid = uid_{tc}, k = 1, \ldots, m,$
and $C < l_k^{tc}.length < TMax, k = 1, \ldots, m-1, l_m^{tc}.length \geq TMax.$

From every reference length transaction we obtain one content transaction by removing the references shorter than the cut-off time C.

2.2. **Mining for Large Content-page Sets.** We properly format the content transactions from the repository R we obtained as described in paragraph 2.1, to be suited for the type of data mining we want to perform. Because temporal information is not needed for the mining of association rules, we exclude it from our set of transactions. We do this next.

Let $P = \{p_1, p_2, \ldots, p_n\}$ be the set of pages within the site. Every such p_i has a unique corresponding url that appears in the Web server access log entries and, consequently, in the content transactions in R, and which uniquely identifies the page within the site.

Definition 4. *We define an application f over R, which transforms a content transaction in a* mining transaction*, corresponding to the relation below:*

$$f(\langle ip, uid, \{(l_1.url, l_1.time, l_1.length), \ldots, (l_m.url, l_m.time, l_m.length)\}\rangle) = \\ = \{p_{k1}, \ldots, p_{km}\},$$

where

$$tc = \langle ip, uid, \{(l_1.url, l_1.time, l_1.length), \ldots, (l_m.url, l_m.time, l_m.length)\}\rangle \in R$$

and p_{ki} is the page that corresponds to $l_i.url$.

Each mining transaction is uniquely identified by a *tid* in the resulting set of mining transactions (we denote this set by D).

Every mining transaction (we refer to it simply as transaction from now on) tm is therefore a set of pages such that $tm \subseteq P$.

Definition 5. *Let X be a set of pages. A transaction tm is said to* contain *X if and only if $X \subseteq tm$.*

a) An association rule *is an implication of the form $X \Rightarrow Y$, where $X \subset P$, $Y \subset P$ and $X \cap Y = \emptyset$.*

b) The rule $X \Rightarrow Y$ holds in the transaction set D with confidence *c if $c\%$ of transactions in D that contain X also contain Y.*

c) The rule $X \Rightarrow Y$ has support *s in the transaction set D if $s\%$ of transactions in D contain $X \cup Y$.*

Given the set of transactions D, the problem of mining association rules is to generate all association rules that have support and confidence greater than a user-specified minimum support (*mins*) and minimum confidence (*minc*) respectively.

This problem of mining association rules can be decomposed into two subproblems. First, find all sets of pages (page sets) that have transaction support above minimum support — which means that they are contained in a sufficient number of transactions such that the page set to have its support larger than *mins*. We call *large page sets* those page sets with minimum support condition satisfied. Once all large page sets are obtained, we use them to generate the desired rules.

There are proposed various algorithms for solving the mining association rules problem ([1, 2, 4]). For what we want to do it is sufficient to limit ourselves at finding the large page sets. We find suitable for this the algorithms described in [2].

2.3. **How We Synthesize Orientation Pages.** Since we choose to mine the mining transactions obtained from *content transactions* in D, we are entitled to say that we will obtain *large content-page sets*. What signifies, in practice, such a large content-page set? Discovering an association rule $X \Rightarrow Y$ in D means that is very frequent the situation when if a user visits the pages in X, he will also visit pages in Y. The support $X \cup Y$ of that rule gives us therefore a cluster of pages

grouped together based on certain common feature. In our case, they are grouped together corresponding to the criteria that they are related by their content. As we said, the association rules are determined from the calculated large page sets. The large content-page set from which $X \Rightarrow Y$ is derived is $X \cup Y$ (however, from $X \cup Y$ it is possible to obtain more that one association rule!). The discussion above justifies why we found sufficient to determine the large page sets, and not to continue with identifying the association rules.

For every set of currently unlinked content-related pages we want to generate an orientation page, comprising one link for every page in that set. Two pages are considered linked if there exists a link from one to the other, or if there exists a page that links to both of them. It certainly wouldn't make sense to include, in an orientation page, links to two pages that are already pointed by a common parent, because we would create a redundant structure equivalent with the existing parent.

We make use in constructing the orientation pages of the following obvious property that stands for large page sets. In fact, the algorithms that discover the large page sets in a transaction repository are based on this property. We enunciate it and then we introduce another concept we will use.

Remark 1. *Any subset of a large page set is also a large page set.*

Definition 6. *We call a* maximal page set *a large page set that is not contained in any other large page set.*

We explained before that every large content-page set comprises pages related by their content, and which are often visited together. Obviously, we wouldn't have any advantage in generating an orientation page for every large page set! This because every large page set that is not maximal will retrieve itself in a series of other large and maximal page sets fact that would cause many redundant orientation pages! So, we are interested in knowing only the maximal content-page sets because they give us the maximal, complete clusters of pages related by their content. To obtain them from the large content-page sets that we have previously determined is a straightforward task.

However, we are not yet at the end of our task. As we affirmed, we want to generate orientation pages for groups of related-by-content pages that are not already linked in the site So, it remains to detect, from every maximal content-page set, the subsets of pages that, two by two, are currently unlinked. We describe bellow, in an algorithmic form, a method to do this by working on a graph model.

Algorithm *ConnComp* is
Set $H = \emptyset$;
For every maximal content-page set previously determined, $M=\{p_{i1}, \ldots, p_{ij}\} \subset P$, Do
Associate to M a graph $G = (M, U)$; $U \subseteq M \times M$ where there is an edge $(p_i, p_j) \in U$ **iff** the pages p_i and p_j are unlinked (meaning that they are not linked in the sense we specified above) in our site;

Find all the connected components of G and add them to H;
End For;

End *ConnComp*

The algorithm *ConnComp* supplies us the set H of all the connected components determined for all maximal content-page sets. For every connected component in H there are two properties that follow from the way we defined G:

- All the pages in a connected component in H are content related;
- Every connected component in H has the property that each two of its pages are not linked.

We must note that it is not realistic to offer to the visitors of the site orientation pages with hundreds of links, because this wouldn't be of any help. Similarly, orientation pages with just a few links should be excluded, as not being significant enough and only burdening the site's structure. So, we agree to reasonably choose two thresholds (min and Max, $min < Max$) to limit the number of links in an acceptable orientation page.

Eventually, we generate one or more orientation pages for every connected component in H like this:

- If the connected component has between min and Max pages, we generate one orientation page containing a link for every page in the component;
- If the connected component has more than Max pages we break it into parts of Max pages each (excepting the last one, which may have between 1 and Max pages). For each part we construct one orientation page and we create a parent index to point to the orientation pages corresponding to all these parts. So, we have a two-level orientation structure. We will not take into consideration the connected components with more than Max^2 pages; we think that such cases have little chance to appear in practice for a common site. For $Max = 10$, imagine what it means "hot" access pattern that imply more than 100 pages!

We point out that every possible improvement to the site (add-on orientation pages) is reported to the Webmaster to be accepted or not. Only the content of pages is automatically supplied; the Webmaster will have to integrate the orientation pages in the overall design of the site and place them where he thinks adequate. So this approach brings in only non-destructive transformations: changes of the site that leave existing structure intact.

3. Heuristic Comparative Study

In [5] grouping the pages above their common topic simply consisted in finding collections of pages that tend to co-occur in visits. This process didn't make difference between pages that were visited only for navigational purposes and those

pages that really interested the user by their content. This fact can erroneously introduce some pages used for navigation, into one cluster of related-by-topic pages, only because they are placed on a frequently used path between two content related pages.

In turn, we identified the pages that *really* share a particular topic by making use of the concept of content transaction [3]. We explain now why this is true.

In the context of classifying a reference made by a user as either a navigational or a content one, there are two ways of defining transactions *3*. One is to define a transaction as all of the navigation references up to and including each content reference for a user session — *navigation-content transactions*. The other is to compound a transaction as we did in this paper, from all of the content references within a user session — *content transactions*. Mining the repository of navigation-content transactions calculated from a log and the one of content transaction determined for the same log will take us to different results. The first approach is, in a way, similar to what is described in [5] — mining navigation-content transaction would essentially give the common traversal paths trough the Web site towards content pages. Mining the content transactions produces, in turn, associations between the content pages of a site, without any information about the path followed between the pages. We point out other significant fact that devolves from our approach: Web mining on content transactions does not produce association rules that might be erroneously determined if we would consider all page references in a log. Imagine this situation: users that treat page A as a navigation page do generally go on to the page B, but users that visit A as a content page do not go on to B. In this case, including navigational references into the data mining process will classify A in the same cluster as B. Mining content transactions will not produce this fake association, because rule $A \Rightarrow B$ will not have minimum support. So, in this way, we grouped the pages related by their content with more accuracy than previously has been done.

4. Conclusions

In this paper we presented an approach to the problem of adaptive Web sites — synthesizing new pages to be added to the site in order to facilitate the retrieval of information was already proposed before. What is new is the way we establish the content of the orientation pages. We identified the clusters of pages that really share a particular topic by making use of the concept of content transaction. We outlined above the advantage of this technique.

References

[1] Agrawal R., Srikant R., *Fast Algorithms for Mining Association Rules*, In Proc. of the 20th VLDB Conference, pp. 487–499, Santiago, Chile, 1994 (http://citeseer.nj.nec.com/agrawal94fast.html).

[2] Chen M.-S., Park J. S., Yu P.S., *Data Mining for Path Traversal Patterns in a Web Environment*, In Proc. of the 16th International Conference on Distributed Computing Systems, pp. 385–392, 1996 (http://citeseer.nj.nec.com/128354.html).

[3] Cooley R., Mobasher B., Srivastava J., *Grouping Web Page References into Transactions for Mining World Wide Web Browsing Patterns*, In Knowledge and Data Engineering Workshop, pp. 2–9, Newport Beach, CA, 1997 (http://citeseer.nj.nec.com/cooley97grouping.html).

[4] Park J. S., Chen M.-S., Yu P.S., *An Effective Hash-Based Algorithm for Mining Association Rules*, In Proc. 1995 ACM-SIGMOD, pp. 175–186, 1995 (http://citeseer.nj.nec.com/park95effective.html).

[5] Perkowitz M., Etzioni O., *Adaptive Web sites: Automatically Synthesizing Web Pages*, In 15- th National Conference on Artificial Inteligence, 1998 (http://citeseer.nj.nec.com/perkowitz98adaptive.html).

[6] Perkowitz M., Etzioni O., *Towards adaptive Web sites: Conceptual framework and case study*, Artificial Intelligence 118 (2000), pp. 245–275, 2000 (http://citeseer.nj.nec.com/326006.html).

[7] Perkowitz M., *Adaptive Web Sites: Cluster Mining and Conceptual Clustering for Index Page Synthesis*, Ph.D. Dissertation, 2001 (http://www.perkowitz.net/mike/research/papers/phd.pdf).

[8] Pitkow J., *In search of reliable usage data on the WWW*, In Proc of the Sixth International World Wide Web Conference, pp. 451–463, Santa Clara, CA, 1997 (http://citeseer.nj.nec.com/242362.html).

Department of Computer Science, Faculty of Mathematics and Computer Science, "Babeş-Bolyai University, 1, M. Kogălniceanu St., RO-3400 Cluj-Napoca, Romania
E-mail address: alina@cs.ubbcluj.ro

Comunications Center, "Babeş-Bolyai University, 1, M. Kogălniceanu St., RO- 3400 Cluj-Napoca, Romania
E-mail address: bufny@cs.ubbcluj.ro