Technical Report

Department of Computer Science
and Engineering
University of Minnesota
4-192 EECS Building
200 Union Street SE
Minneapolis, MN 55455-0159 USA

TR 99-022

Discovery of Interesting Usage Patterns from Web Data

Robert Cooley, Pang-ning Tan, and Jaideep Srivastava

May 23, 1999

Discovery of Interesting Usage Patterns from Web Data (# 387)

Robert Cooley[*], Pang-Ning Tan, Jaideep Srivastava[†]
{cooley,ptan,srivasta}@cs.umn.edu
Department of Computer Science
University of Minnesota

March 5, 1999

Abstract

Web Usage Mining is the approach of applying data mining techniques to large Web data repositories in order to extract usage patterns. As with many data mining application domains, the identification of patterns that are considered *interesting* is a problem that must be solved in addition to simply generating them. A necessary step in identifying interesting results is quantifying what is considered uninteresting in order to form a basis for comparison. Several research efforts have relied on manually generated sets of uninteresting rules. However, manual generation of a comprehensive set of evidence about beliefs for a particular domain is impractical in many cases. Generally, domain knowledge can be used to automatically create evidence for or against a set of beliefs. This paper develops a quantitative model based on *support logic* for determining the interestingness of discovered patterns. For Web Usage Mining, there are three types of domain information available; *usage, content*, and *structure*. This paper also describes algorithms for using these three types of information to automatically identify interesting knowledge. These algorithms have been incorporated into the Web Site Information Filter (WebSIFT) system and examples of interesting frequent itemsets automatically discovered from real Web data are presented.

Keywords: data mining, world wide web, frequent itemsets, web mining, access patterns.

[*]Supported by NSF grant ASC 9554517
[†]Supported by ARL contract DA/DAKF11-98-P-0359

1 Introduction and Background

The World Wide Web continues to expand at an amazing rate as a medium for conducting business and disseminating information. Even with evolving standards and technology, the ability to thoroughly analyze the usage of a Web site remains, and will grow, as an important capability for Web administrators. Design of a Web site centers around organizing the information on each page and the hypertext links between the pages in a way that seems most natural to the site users, to facilitate their browsing. For small sites, an individual Web designer's intuition along with some straightforward usage statistics may be adequate for predicting and verifying the users' browsing behavior. However, as the size and complexity of a Web site increases, the statistics provided by existing Web log analysis tools [WLA, HLC, FWP] may prove inadequate, and more sophisticated types of analyses will be necessary. *Web Usage Mining*, which is the application of data mining techniques to large Web data repositories, adds powerful techniques to the tools available to a Web site administrator for analyzing Web site usage.

Web Usage Mining techniques developed in [CMS99, CPY96, SZAS97, SF98, ZXH98, PE98] have been used to discover frequent itemsets, association rules [AS94], clusters of similar pages and users, sequential patterns [MTV95], and perform path analysis [CPY96]. Several research efforts [NW97, JFM97] have considered usage information for performing *Web Content Mining* [CMS97]. An overview of some of the challenges involved in Web Content Mining is given in [Vai98].

The notion of what makes discovered knowledge interesting has been addressed in [PSM94, ST96, LHC97, PT98]. A common theme among the various criteria for interestingness is the concept of *novelty* or *unexpectedness* of a rule. Results that were previously known by the data analyst are not considered interesting. In Web Usage Mining, as with many data mining domains, thresholds for values such as *support* and *confidence* are often used to limit the number of discovered rules to a manageable number. However, high thresholds rarely discover any knowledge that was not previously known and low thresholds usually result in an unmanageable number of rules. The approach advocated by [LHC97, PT98] is to identify a set of *beliefs*, and use the set as a filter for identifying interesting rules. Rules that confirm

existing beliefs are deemed uninteresting.

In a more general sense, both the discovered knowledge and any expectations defined from domain knowledge can be considered as pieces of evidence providing support *for* or *against* a particular belief. There can be multiple sources of evidence pertaining to any given belief about a domain, some of them possibly contradictory. Also, as pointed out in [LHC97], evidence about some of the beliefs is likely to represent vague concepts, and require a framework with fuzzy logic [Zad79] capabilities. While the frameworks developed in [LHC97] and [PT98] are designed to handle some of these conditions, neither can handle all of them. Fortunately, the problem can be mapped to Baldwin's *support logic* [Bal87] framework. Support logic is specifically designed to handle reasoning about multiple sources of evidence with both boolean and fuzzy logic. The framework is built around defining *support pairs* for every piece of evidence.[1]

Another problem that exists with the identification of interesting results is the generation of an initial set of evidence about beliefs from domain knowledge. Both [LHC97] and [PT98] rely on manually generated sets of evidence. For [PT98], beliefs are only defined as interesting if there is conflicting evidence, so unless a fairly comprehensive set is created, many interesting results can be missed. [LHC97] has a broader definition of interestingness that includes results that provide evidence about a belief not covered by domain knowledge. However, without a comprehensive set of evidence from domain knowledge, this method will end up misclassifying many results.

The Web Usage Mining domain has several types of information available that can be used as surrogates for domain knowledge. Using this information, a large and fairly comprehensive set of evidence can be automatically generated to effectively filter out uninteresting results from the Web Usage Mining process.

The specific contributions of this paper are:

- Development of a general quantitative model of what determines the interestingness of discovered knowledge, based on Baldwin's support logic framework [Bal87].
- Development of an approach for the automatic creation of an initial set of evidence

[1] In order to avoid confusion with the standard data mining definition of support, Baldwin's support pairs will be referred to as *evidence pairs* for the rest of this paper.

about a belief set.

- Identification of specific algorithms for automated discovery of interesting rules in the Web Usage Mining domain.

- Presentation of results from a Web Usage Mining system called the Web Site Information Filter (WebSIFT) system, using data collected from an actual Web site.

The rest of this paper is organized as follows: Section 2 defines the different types of Web data and information abstractions suitable for usage mining. Section 3 develops a general support logic based framework for defining and combining evidence about a domain. A formal definition of interestingness is also given in this section. Section 4 presents an overview of the WebSIFT system and Section 5 describes algorithms used by the WebSIFT system for automatically identifying interesting rules and patterns. Section 6 summarizes some results from tests of the WebSIFT system on a Web server log. Finally, section 7 provides conclusions.

2 Data Sources and Information Abstractions

Web Usage Mining analysis can potentially use many different kinds of data sources, as discussed in [PPR96]. This paper classifies such data into the following broad types[2]:

- **Content:** The *real* data in the Web pages, i.e. the data the Web page was designed to convey to the users. This usually consists of, but is not limited to text and graphics.

- **Structure:** The data which describes the organization of the content. *Intra-page* structure information includes HTML or XML tags of various kinds, the sequence in which they appear, etc. The principal kind of *inter-page* structure information is hyper-links connecting one page to another.

[2]Active data, which includes the different kinds of code fragments contained in a Web page such as applets and scripts, along with dynamic HTML, are becoming more prominent in the Web, but are not addressed in this paper.

- **Usage:** The data that describes the pattern of usage of Web pages, such as IP addresses, page references, and the date/time of accesses. This information can be obtained from Web server logs.

- **Administrative:** Administrative data includes information that can not be automatically obtained from the server logs. This includes user registration data, etc.

Several information abstractions can be derived from one or more types of data listed above, which in turn can be used to perform different kinds of analyses. The following abstractions are used in this paper:

- **Page view:** Due to the stateless connection model of the HTTP protocol, a single request by a user in the form of a mouse click on a hypertext link often results in several file requests in addition to the HTML page. This group of files make up what is sometimes referred to as a *page view*. By using the intra-page structure, all of the files for a particular page view can easily be identified in a server log and aggregated into a single data structure.

- **Click-stream:** A *click-stream* is the sequence of pages accessed by a user (or group of users) during navigation of the Web. In general such a click-stream spans multiple web sites, with one or more visits to a particular site, making its construction an especially challenging problem. In addition, a number of service requests are fulfilled directly from proxy servers or local caches, about which the Web server has no information at all, and thus there can be substantial gaps in the click stream.

- **User transaction:** It is widely believed that individual HTTP requests is too fine a granularity to analyze user behavior. However, this is exactly the level at which data is collected in Web server logs. [CMS99] has identified the concept of a Web *user transaction*, which is a unit of meaningful interaction between the user and the Web server. This is done by clustering a sequence of HTTP requests from a user, with sufficient proximity in time, into a user transaction.

- **Page type:** Web pages can be classified into various types based on their content, structure and other attributes. For example, the ratio of the number of links in a page

to the number of text units (say words) can be used as a measure for classifying pages into various types such as *navigational*, *content*, or *hybrid*. This issue is discussed in detail in [CMS99].

Various kinds of analyses can be performed on these abstractions to extract knowledge useful for a variety of applications. A specific type of analysis is to make assertions about the aggregate usage behavior of all users who visit pages of a Web site. For example, the assertion can be made that a pair of pages that have structural proximity (due to hyperlinks between them) and/or content proximity (since they have information on closely related topics), are likely to be visited together often. Analysis of structure and content information can be used to make the initial assertion, and subsequent analysis of usage data can be used to examine the truth of such an assertion.

Note that in the above assertion, words such as *likely* and *often* are used rather than *will* and *always*. In an inductive analysis scenario with many sources of uncertainty as identified above, the first set of words more accurately captures the nature of assertions that can be made, making standard predicate logic too brittle a reasoning framework. Hence, the framework of *support logic* [Bal87] is used for analysis, as described in the next section.

3 Evaluation of Beliefs in a Support Logic Framework

3.1 Measures of Interestingness

The ultimate goal of any data mining effort is to provide the analyst with results that are interesting and relevant to the task at hand. [ST96] defines two types of interestingness measures - *objective* and *subjective*. *Objective* measures rate rules based on the data used in the mining process. Thresholds on *objective* measures such as confidence,support, or chi-square [BMS97] are invaluable for reducing the number of generated rules, but often fall well short of the goal of only reporting rules that are of potential interest to the analyst.

For *subjective* measures of interestingness, [ST96] defines two criteria to evaluate rules and patterns. A rule is *unexpected* if it is "surprising" to the data analyst, and *actionable* if the analyst can act on it to his advantage. The degree to which a rule is *actionable* depends

on its application. Consider the use of association rules to restructure a Web site. Since the topology or content of a Web site can be modified based on any discovered information, all rules are *actionable* for this application. [PT98] formally defines the unexpectedness of a rule in terms of its deviation from a set of beliefs. [LHC97] has a broader definition of interestingness that includes discovered rules that are not specifically covered by an initial set of beliefs. In other words, a rule that doesn't contradict an existing belief, but points out a relationship that hadn't even been considered is also interesting. While both [LHC97] and [PT98] give examples of small sets of manually generated beliefs, neither addresses the issue of automated generation of a realistic belief set from a large amount of data.

3.2 Support Logic

A more general way to look at the problem of identifying the interestingness of discovered patterns is to consider each piece of information in terms of the evidence it gives *for* or *against* a given logical statement (belief). Baldwin's *support logic* [Bal87, RB89] provides a framework for this point of view. For a belief $\mathcal{B}$, evidence collected for or against $\mathcal{B}$ can be used to form an *evidence pair*, $[e_n, e_p]$, where:

$$
\begin{aligned}
e_n &= \text{necessary evidence in support of } \mathcal{B} && (1)\\
e_p &= \text{possible evidence in support of } \mathcal{B} && (2)\\
(1 - e_p) &= \text{necessary evidence in support of } \neg\mathcal{B} && (3)\\
(1 - e_n) &= \text{possible evidence in support of } \neg\mathcal{B} && (4)\\
(e_p - e_n) &= \text{uncertainty of } \mathcal{B} && (5)
\end{aligned}
$$

The values of e_n and e_p must satisfy the constraint:

$$e_n + (1 - e_p) \leq 1 \qquad (6)$$

Figure 1 shows the concepts that map to each region of a belief scale, given e_n and e_p. As an example, assume that evidence has been collected about the belief $\mathcal{B}(X, Y)$, that Web pages X and Y are related. If all of the evidence is in support of $\mathcal{B}(X, Y)$, the evidence pair

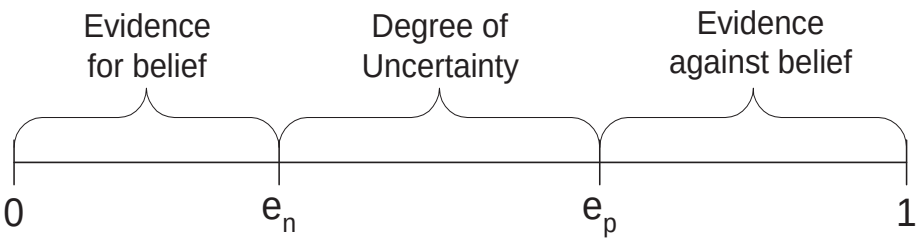

Figure 1: Evidence pair values for a belief

is $[1,1]$. On the other extreme, if all of the evidence is against $\mathcal{B}(X,Y)$, the evidence pair is $[0,0]$. If the data leads to a 25% degree of belief that $\mathcal{B}(X,Y)$ is true, and a 40% degree of belief that $\mathcal{B}(X,Y)$ is false, then $[0.25, 0.6]$ would represent the appropriate evidence pair. This says that the degree of uncertainty about $\mathcal{B}(X,Y)$ is 35%. Finally, if there is no evidence pertaining to $\mathcal{B}(X,Y)$, the evidence pair is $[0,1]$, giving an uncertainty of 100%. Independent of the type of the source for generating an evidence pair, pairs can be combined per Baldwin's *support logic programming calculus* [Bal87] to obtain a single evidence pair per belief. The basic rules are as follows:

If $\mathcal{B} : [e_{1n}, e_{1p}]$ AND $\mathcal{B} : [e_{2n}, e_{2p}]$ are two independent evidence pairs about belief $\mathcal{B}$, then conclude $\mathcal{B} : [e_n, e_p]$, where

$$K = 1 - e_{1n}(1 - e_{2p}) - e_{2n}(1 - e_{1p}) \tag{7}$$

$$e_n = [e_{1n}e_{2n} + e_{1n}(e_{2p} - e_{2n}) + e_{2n}(e_{1p} - e_{1n})]/K \tag{8}$$

$$1 - e_p = [(1 - e_{1n})(1 - e_{2n}) + (e_{1n} - e_{1p})(1 - e_{2n}) + (e_{2n} - e_{2p})(1 - e_{1n})]/K \tag{9}$$

All beliefs have a default evidence pair value of [0,1] until some data is introduced that pertains to that belief. As subsequent data relevant to a belief is analyzed, the evidence pair can be updated using equations 7, 8, and 9. For any data mining domain, the rules that are generated can be used to initialize a set of evidence pairs. A second set of evidence pairs can be generated from domain knowledge. This is shown conceptually in Figure 2, where the rectangle represents all of the beliefs that can be held about a particular domain. The region where the ovals overlap represents the set of beliefs that have evidence pairs from both the mined knowledge and the domain knowledge. This region is split into evidence pairs that

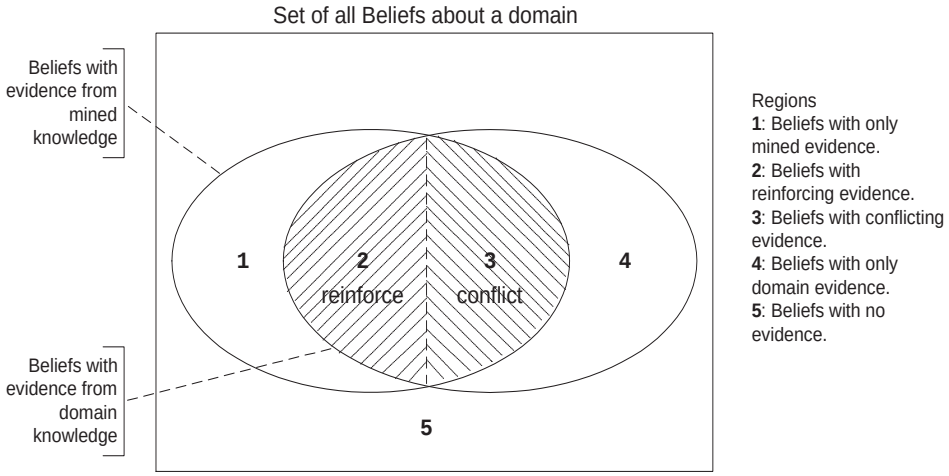

Figure 2: Evidence collected about a set of beliefs

reinforce each other, and evidence pairs that conflict with each other. An interesting result is a belief with evidence pairs from each source that are significantly different from each other. "Significantly different" can be determined by setting threshold values for differences in both e_n and e_p. The formal definition of interesting is as follows:

For a belief, $\mathcal{B}$ with an interestingness pair $\mathcal{I}(n_i, p_i)$, where

$$n_i = |e_{1n} - e_{2n}| \tag{10}$$

$$p_i = |e_{1p} - e_{2p}| \tag{11}$$

$\mathcal{B}$ is interesting if:

$$n_i > \mathcal{T}_n \quad \text{OR} \quad p_i > \mathcal{T}_p \tag{12}$$

where

$\mathcal{T}_n$ = necessary evidence threshold

$\mathcal{T}_p$ = possible evidence threshold

The ability to set different thresholds for changes in the evidence for or against a certain belief can be important in domains where one is more significant than the other[3]. Since the interestingness of a belief is defined by a real-valued pair, an ordering among interesting

[3]For example, in anomaly detection such as credit card fraud, a small change in the evidence for an occurrence of fraud is probably more interesting to an analyst than a larger change in the evidence against a fraud occurring.

Table 1: Comparison of Mined and Domain Knowledge Boolean Evidence Pairs

Mined Knowledge	Domain Knowledge	Combined Knowledge	Figure 2 Region	Interesting Belief
[0,0]	[0,0]	[0,0]	2	No
[0,0]	[0,1]	[0,0]	1	Yes
[0,0]	[1,1]	Null	3	Yes
[0,1]	[0,0]	[0,0]	4	Yes
[0,1]	[0,1]	[0,1]	5	No
[0,1]	[1,1]	[1,1]	4	Yes
[1,1]	[0,0]	Null	3	Yes
[1,1]	[0,1]	[1,1]	1	Yes
[1,1]	[1,1]	[1,1]	2	No

beliefs can also be established. In the simplest case, all evidence is either 100% for a belief, 100% against a belief, or there is no evidence about a belief. This leads to nine different cases that can occur when comparing the mined knowledge to domain knowledge. These are shown in Table 1 along with the number of the region in Figure 2 that each case maps to, and whether or not the belief is considered interesting. For the two cases where the mined knowledge and domain knowledge are in complete disagreement, the combined evidence pair is "Null." This is because completely contradictory evidence can not, and should not be automatically reconciled. Note that the definitions of interestingness from both [PT98] and [LHC97] are included in this framework. The [PT98] definition maps to region 3 of Figure 2 and the [LHC97] definition maps to regions 1 and 3. However, as shown in Table 1, the support logic framework also includes beliefs that fall into region 4 as interesting. These are the beliefs with evidence from domain knowledge that the mined knowledge does not address.

3.3 Generation of Belief Sets for Web Usage Mining

For Web Usage Mining, there are two sets of domain knowledge from which evidence pairs can be created; the content and structure data. The task of reconciling conflicting evidence from the content and structure data falls under the category of *Web Content Mining*, which is beyond the scope of this paper. The assumption is that content and structure data can

Table 2: Examples of Web Usage Information that will be automatically flagged as Interesting

Mined Knowledge	Domain Knowledge Source	Interesting Belief Example
General Usage Statistics	Site Structure	The head page is not the most common entry point for users
General Usage Statistics	Site Content	A page that is designed to provide content is being used as a navigation page
Frequent Itemsets	Site Structure	A frequent itemset contains pages that are not directly linked
Usage Clusters	Site Content	A usage cluster contains pages from multiple content clusters

be used as surrogates for the Web site designer's domain knowledge. Links between pages provide evidence in support of those pages being related. The stronger the topological connection is between a set of pages, the higher the value of e_n is set for the evidence pair. Evidence pairs based on the site content can also be automatically generated by looking at content similarity, and assigning values of e_n and e_p based on the calculated "distance" between pages. Table 2 gives some examples of the types of interesting beliefs that can be identified in the Web Usage Mining domain using the framework described in the previous section.

As a quantitative example, consider a Web page that is used 70% of the time as a content page and 30% of the time as a navigation page, as determined from a Web server log. This would lead to an evidence pair equal to [0.7,0.7] for the belief that the page is a content page. By looking at the page content and structure, an evidence pair equal to [0.6,0.9] might be created. This means that the interestingness of the belief that this page is a content page is $\mathcal{I}(0.1,0.2)$, based on equations 10 and 11. Essentially, the content and structure confirm the usage of the page, and unless the interestingness thresholds are set fairly low, the belief that this page is a content page would not be considered interesting.

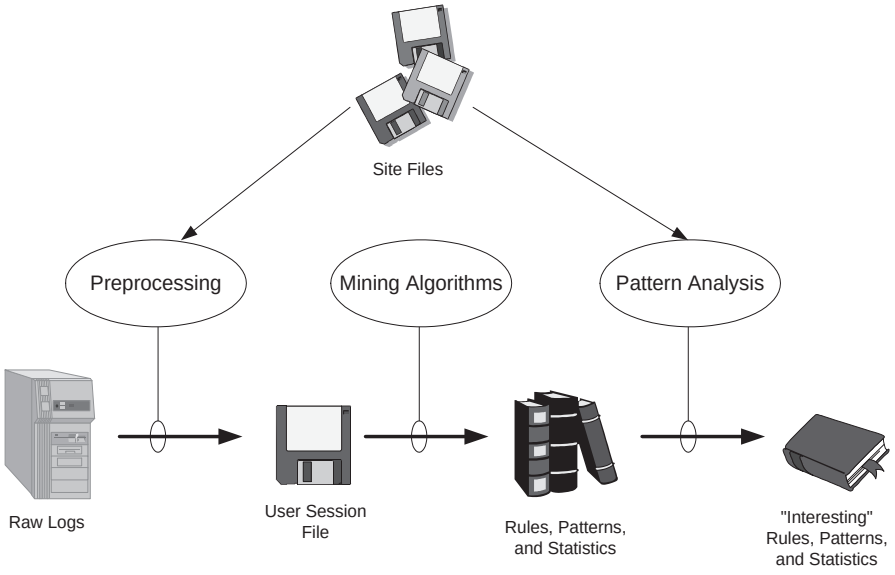

Figure 3: High Level *Web Usage Mining* Process

4 The WebSIFT System

The WebSIFT system, which is based on the WEBMINER prototype [CMS97], divides the Web Usage Mining process into three main parts, as shown in Figure 3. For a particular Web site, the three server logs - access, referrer, and agent, the HTML files that make up the site, and any optional data such as registration data or remote agent logs provide the information to construct the different information abstractions defined in Section 2. The preprocessing phase uses the input data to construct a user session file, which is the best estimate of the user's browsing behavior based on the methods and heuristics discussed in [CMS99]. In addition to being used to derive a site topology, the site files are used to classify the pages of a site. Both the site topology and page classifications are then fed into the *information filter*, which is described in detail in the next section.

The knowledge discovery phase uses existing data mining techniques to generate rules and patterns. Included in this phase is the generation of general usage statistics, such as number of "hits" per page, page most frequently accessed, most common starting page, and average time spent on each page. The discovered information is then fed into various pattern analysis tools. The WebSIFT system, shown in detail in Figure 4, has been implemented using a relational database, procedural SQL, and the Java programming language. Java

Database Connectivity (JDBC) drivers are used to interface with the database. Although algorithms have been identified and tested for individual portions of the system, only the generation and filtering of frequent itemsets, association rules, and general statistics is fully automated at this time.

5 Filtering of Knowledge based on Interestingness

The current implementation of the WebSIFT system uses two different ways to identify interesting results from a list of discovered association rules. The first approach corresponds to finding all the rules that fall into region 1 of Figure 2, the *beliefs with only mined evidence* (BME). This is accomplished by declaring rules that contain pages not directly connected to each other to be interesting. In the second approach, the WebSIFT information filter attempts to identify *beliefs with only domain evidence* (BDE). Since the site structure maps to the belief set of what pages should be accessed together, the absence of rules containing these directly linked pages should be interesting. This algorithm will use the site structure to construct pairs of connected pages, and declare the pairs that are missing from the discovered knowledge to be interesting. Frequent itemsets, a more basic form of mined knowledge, are used as the input for the algorithms. Interesting frequent itemsets can be converted to association rules or sequential patterns if so desired. In both approaches, the discovered itemsets that fall into the *reinforcing evidence region* will be labeled as uninteresting. Because both the domain evidence from the site structure and mined evidence from the frequent itemsets only provide evidence in support of a given belief, there will be no conflicting sets of evidence (region 3 of Figure 2).

5.1 The BME Algorithm

The BME algorithm removes uninteresting itemsets that confirm directly connected pages. The site structure can be modeled as a graph, with pages as nodes and hypertext links as edges. In this conceptual model, the algorithm for determining interesting frequent itemsets finds the number of connected components required to cover all pages in an itemset. If a single connected component can represent all the pages in an itemset, this confirms the belief

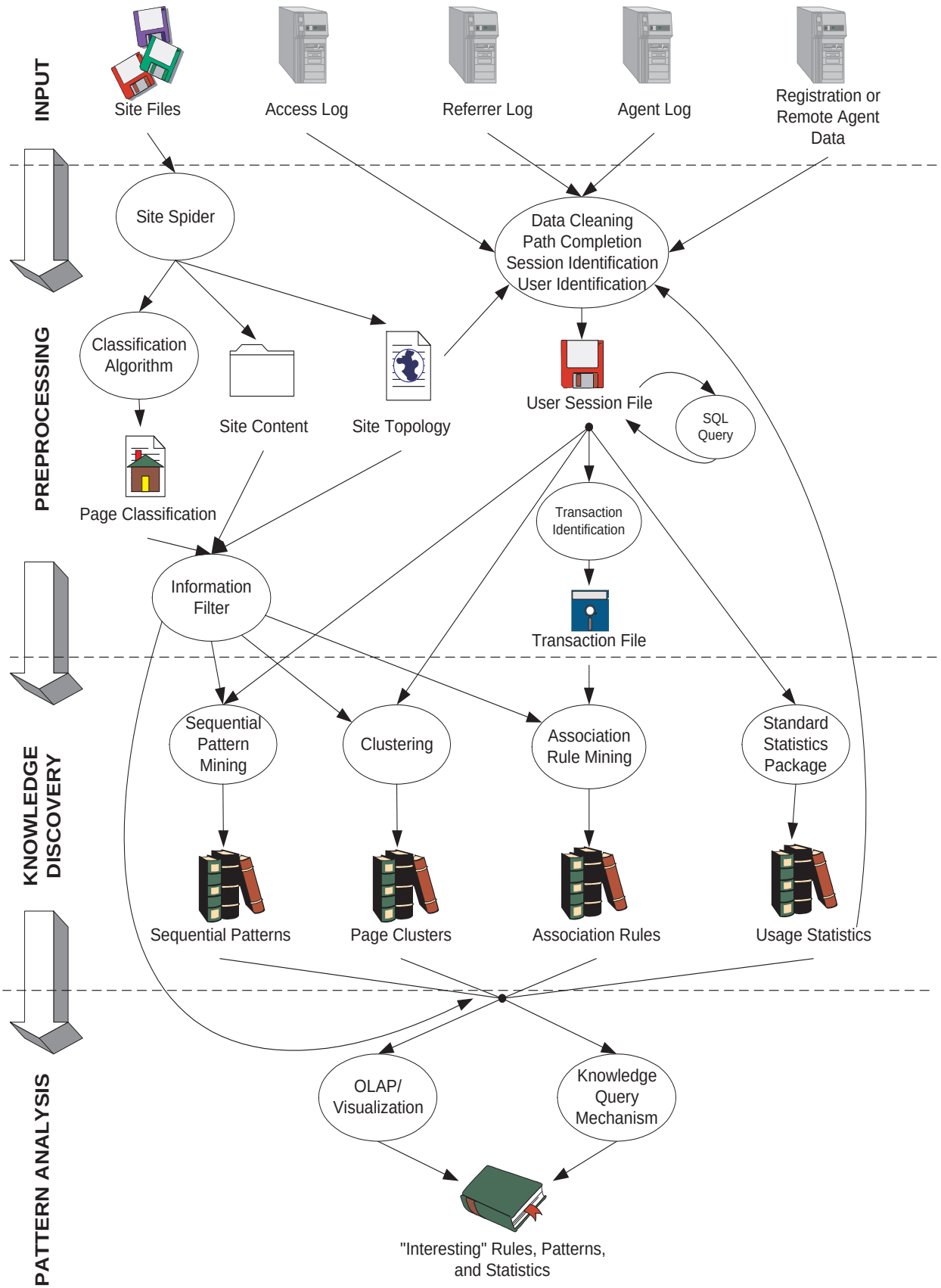

Figure 4: WebSIFT Architecture

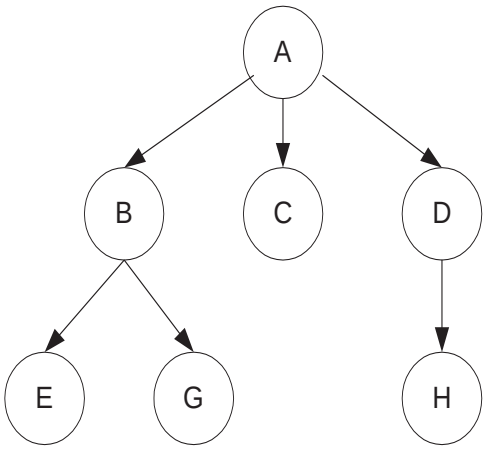

Figure 5: Sample Web Site—Arrows between the pages represent hypertext links

that the connected pages will be accessed together, and such an itemset is declared to be uninteresting. Conversely, if it requires more than one connected component to cover all of the itemset pages, this frequent itemset is reported as interesting.

5.1.1 Notation

The site structure can be represented as an undirected graph SG = $(V_{\mathrm{SG}}, E_{\mathrm{SG}})$, where V_{SG} is a set of nodes representing the pages at the site, and E_{SG} is a set of edges that represent the hyperlinks connecting these pages.

A set of frequent itemsets is denoted as F = $\{I_1, I_2, \cdots, I_k\}$ where each $I_j \subseteq V_{\mathrm{SG}}$ $(\forall j : 1 \leq j \leq k)$ is an itemset. Each itemset I_j can be represented as an undirected graph $\mathrm{FG}_j = (V_{\mathrm{FG_j}}, E_{\mathrm{FG_j}})$, where $V_{\mathrm{FG_j}}$ is the set of all items in I_j and $E_{\mathrm{FG_j}}$ corresponds to the set of undirected edges that are present among these items in the site graph SG. Note that SG is represented as an undirected graph since the only concern is whether or not the components of the graph are connected.

5.1.2 Example

As an example of how the algorithm works, consider the Web site depicted in Figure 5 and the set of frequent itemsets shown in Table 3. Initially, the set of frequent itemsets is pruned to remove itemsets that are proper subsets of larger itemsets. At the end of this pruning step, F contains only {{ B,E }, { C,E }, { A,B,C } } (single item frequent itemsets

Table 3: Frequent itemsets for sample Web site

Size	Sets
Single Item	{A},{B},{C},{E},{G}
Two Item	{A B},{A C},{B C},{B E},{C E}
Three Item	{A B C}

Table 4: The BME Algorithm

```
Algorithm BME
1. let SG = (V_SG, E_SG) denotes the site structure
2. let F = {I_1, I_2, ..., I_n} denotes the discovered frequent itemsets
3. for each itemset I ∈ F do
4.     if ∃J ∈ F such that I ⊂ J
5.         then remove I from F
6. for each itemset i ∈ F do
7.     FG ← CreateGraph(i, E_SG);
8.         if Connected(FG) > 1 then add i to Result
9. end;
```

are not used in this algorithm). Next, for each of the remaining itemsets, the algorithm uses the site structure to determine the number of connected components in the itemset. If there are two or more connected components, the itemset is declared to be interesting. It can be easily seen from Figure 5 that { C, E } is the only itemset that creates more than one connected component. Hence, it is the only itemset declared to be interesting by the BME algorithm.

5.1.3 Algorithm and Complexity Analysis

Initially, the set of frequent itemsets F is pruned to remove all the proper subsets (lines 3 to 5 of Table 4). Next, for each remaining itemset, an undirected graph FG is constructed based on the page references within the itemset (line 7). The *Connected()* function returns the number of connected components of the graph FG. If the itemset generates more than one connected component, it is added to the *Result* set.

The analysis of the above algorithm can be done as follows. Let n be the number of pruned frequent itemsets, $k_i = |I_i|$, and m is the average time to determine if two itemsets are connected to one another. The time complexity of this algorithm is $T = \sum_{i=1}^{n} k_i m$. Assuming

Table 5: The BFE Algorithm

Algorithm BFE
1. let S = (V_{SG}, E_{SG}) denote the site structure
2. let F = $\{i_1, i_2, \cdots, i_k\}$ denote the discovered frequent itemsets
3. for each pair (i, j) where $i \in V_{SG}$, $j \in V_{SG}$, $i \neq j$ do
4. if $(i, j) \in E_{SG}$
5. then insert $\{i, j\}$ into Temp
6. Result $\rightarrow$ Temp $-$ F

$k_{\max}$ is the size of the largest itemset, the worst-case time complexity is O($mnk_{\max}$).

5.2 The BDE Algorithm

As previously discussed, if two pages are directly linked but do not occur together in a frequent itemset, this information can be as important as the existence of frequent itemsets that are not supported by the site structure. The essential idea of this algorithm is to generate all pairs of connected pages based on the site structure, and then perform a pruning step by removing the pairs that already exist in the discovered frequent itemsets.

5.2.1 Example

Consider again the graph shown in Figure 5 and the set of discovered frequent itemsets given in Table 3. Further, suppose that only pages A, B, C, E, and G are visited often enough to meet the minimum support threshold (this is the list of single item frequent itemsets). Based on the site structure, the following set of connected pages are generated : { (A,B), (A,C), (B,E), (B,G) }. Upon removing the itemsets found in the discovered knowledge (Table 3), only {(B,G)} is reported as interesting.

5.2.2 Algorithm and Complexity Analysis

For each pair of page references in the site structure, the algorithm checks if there is an edge connecting them (line 4 of Table 5). If so, the pair of pages is added to the *Temp* set (line 5) . The final result is obtained by removing the itemsets that appear in the discovered frequent itemsets from the *Temp* set (line 6).

The above algorithm has a time complexity of $O(|V_{SG}|^2)$. This can be improved by performing a depth-first search on the site structure and adding the pair $\{i, \text{parent}(i)\}$ into the *Temp* set every time a new node i is discovered. Such an algorithm will have a complexity of $O(|V_{SG}| + |E_{SG}|)$.

6 Experimental Evaluation

6.1 Experimental Design

The experiments described in this section were performed on a Web server log from the University of Minnesota Department of Computer Science and Engineering Web site; http://www.cs.umn.edu/. The server log collects data in the combined log format, with the access, agent, and referrer data all collected in a single file. The log used was from eight days in February, 1999. The physical size of the log was 19.3 MB and it consisted of 102,838 entries in its raw form. Once preprocessing was completed, there were 43,158 page views divided among 10,609 user sessions.

6.2 Interesting Frequent Itemsets

A threshold of 0.1% for support was used to generate 693 frequent itemsets, with a maximum set size of six pages. There were 178 unique pages represented in all of the rules. Both the BME and BDE algorithms described in the previous section were run on the frequent itemsets. The BME algorithm resulted in 11 frequent itemsets being declared as potentially interesting, and the BDE algorithm resulted in 10 missing page pairs being declared as potentially interesting. Tables 6 and 7 show the interesting results identified by each algorithm.

Of the frequent itemsets shown in Table 6, the two about the graduate handbook (numbers 10 and 11) are of note because these pages are out-of-date. A page with the 1998-99 graduate handbook exists, and the links from all site pages to the older handbooks have been removed. However, since the pages were not actually removed from the site, users with bookmarks to the old handbooks continue to reference them, unaware that the informa-

Table 6: Interesting frequent itemsets identified by BME algorithm

#	Mined Support(%)	Related Pages
1	0.10	/Research/, /tech_reports/
2	0.10	/employment/, /newsletter/
3	0.10	/faculty/, /newsletter/
4	0.10	/icra99/ICRA99-Index.htm, /icra99/Notice.html, /icra99/TechnProgram.htm, /icra99/advanceprogram2.htm
5	0.10	/new/, /sem-coll/
6	0.10	/reg-info/98-99_schedule.html, /reg-info/ss1-99.html, /reg-info/ss2-99.html
7	0.11	/Research/Agassiz/, /faculty/
8	0.11	/icra99/Notice.html, /icra99/best.html
9	0.11	/icra99/Proceeding-Order.htm, /icra99/Registration.htm
10	0.22	/grad-info/, /grad-info/97-98-grad-handbook.html
11	0.25	/grad-info/, /grad-info/96-97-grad-handbook.html

tion is out-of-date. The support of these itemsets is 0.25% and 0.22%, meaning they would have been missed if the support had been set higher to limit the total number of itemsets discovered.

In Table 7, the fourth pair of pages is of note because the sole purpose of the first page is as an entry page to a particular research group's pages. However, the link from the first page is flashing and located fairly low on the page. This indicates a design problem since not all of the visitors from the first page are visiting the second.

7 Conclusions

Using the support logic model, this paper has developed a general framework for determining the interestingness of mined knowledge. The framework leverages the power of a robust logic system based on fuzzy logic and evidential reasoning, that is capable of reasoning about evidence from multiple sources about a given belief. Both reinforcing and conflicting pieces of evidence can be handled. In addition, automated methods for generating evidence in support of beliefs have been defined and tested for the Web Usage Mining domain. From a set of almost 700 discovered frequent itemsets, 21 interesting itemsets were identified. Of these 21 interesting itemsets, two identified out-of-date information that needed to be

Table 7: Interesting page pairs identified by BDE algorithm

#	Web Pages
1	/Research/Agassiz/agassiz_pubs.html, /Research/Agassiz/agassiz_people.html
2	/Research/GIMME/tclprop.html, /Research/GIMME/Nsync.html
3	/Research/airvl/minirob.html, /Research/airvl/loon.html
4	/Research/mmdbms/home.shtml, /Research/mmdbms/group.html
5	/newsletter/kumar.html, /newsletter/facop.html
6	/newsletter/letter.html, /newsletter/facop.html
7	/newsletter/letter.html, /newsletter/kumar.html
8	/newsletter/newfac.html, /newsletter/facop.html
9	/newsletter/newfac.html, /newsletter/kumar.html
10	/newsletter/newfac.htm, /newsletter/letter.html

removed from a Web site, and one pointed out an instance of poor page design.

Future work will include implementation and testing of an information filter that takes full advantage of the fuzzy logic capabilities of the support logic framework. In addition, evidence from both Web site structure and content will be combined prior to comparison with usage evidence.

References

[AS94] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proc. of the 20th VLDB Conference*, pages 487–499, Santiago, Chile, 1994.

[Bal87] J. F. Baldwin. Evidential support logic programming. *Fuzzy Sets and Systems*, 24(1):1–26, 1987.

[BMS97] S. Brin, R. Motwani, and C. Silverstein. Beyond market baskets: Generalizing association rules to correlations. In *ACM SIGMOD International Conference on Management of Data*, 1997.

[CMS97] Robert Cooley, Bamshad Mobasher, and Jaideep Srivastava. Web mining: Information and pattern discovery on the World Wide Web. In *International Conference on Tools with Artificial Intelligence*, pages 558–567, Newport Beach, 1997. IEEE.

[CMS99] Robert Cooley, Bamshad Mobasher, and Jaideep Srivastava. Data preparation for mining World Wide Web browsing patterns. *Knowledge and Information Systems*, 1(1), 1999.

[CPY96] M.S. Chen, J.S. Park, and P.S. Yu. Data mining for path traversal patterns in a web environment. In *16th International Conference on Distributed Computing Systems*, pages 385–392, 1996.

[FWP] Funnel Web Professional. http://www.activeconcepts.com.

[HLC] Hit List Commerce. http://www.marketwave.com.

[JFM97] T. Joachims, D. Freitag, and T. Mitchell. Webwatcher: A tour guide for the World Wide Web. In *The 15th International Conference on Artificial Intelligence*, Nagoya, Japan, 1997.

[LHC97] Bing Liu, Wynne Hsu, and Shu Chen. Using general impressions to analyze discovered classification rules. In *Third International Conference on Knowledge Discovery and Data Mining*, 1997.

[MTV95] H. Mannila, H. Toivonen, and A. I. Verkamo. Discovering frequent episodes in sequences. In *Proc. of the First Int'l Conference on Knowledge Discovery and Data Mining*, pages 210–215, Montreal, Quebec, 1995.

[NW97] D.S.W. Ngu and X. Wu. Sitehelper: A localized agent that helps incremental exploration of the World Wide Web. In *6th International World Wide Web Conference*, Santa Clara, CA, 1997.

[PE98] Mike Perkowitz and Oren Etzioni. Adaptive Web sites: Automaticlly synthesizing Web pages. In *Fifteenth National Conference on Artificial Intelligence*, Madison, WI, 1998.

[PPR96] Peter Pirolli, James Pitkow, and Ramana Rao. Silk from a sow's ear: Extracting usable structures from the Web. In *CHI-96*, Vancouver, 1996.

[PSM94] G. Piatetsky-Shapiro and C. J. Matheus. The interestingness of deviations. In *AAAI-94 Workshop on Knowledge Discovery in Databases*, pages 25–36, 1994.

[PT98] Balaji Padmanabhan and Alexander Tuzhilin. A belief-driven method for discovering unexpected patterns. In *Fourth International Conference on Knowledge Discovery and Data Mining*, pages 94–100, New York, New York, 1998.

[RB89] A. L. Ralescu and J. F. Baldwin. Concept learning from examples and counter examples. *International Journal of Man-Machine Studies*, 30:329–354, 1989.

[SF98] Myra Spiliopoulou and Lukas C Faulstich. WUM: A Web Utilization Miner. In *EDBT Workshop WebDB98*, Valencia, Spain, 1998. Springer Verlag.

[ST96] A. Silberschatz and A. Tuzhilin. What makes patterns interesting in knowledge discovery systems. *IEEE Transactions on Knowledge and Data Eng.*, 8(6):970–974, 1996.

[SZAS97] Cyrus Shahabi, Amir M Zarkesh, Jafar Adibi, and Vishal Shah. Knowledge discovery from users Web-page navigation. In *Workshop on Research Issues in Data Engineering*, Birmingham, England, 1997.

[Vai98] Shivakumar Vaithaynathan. Data Mining on the Internet - a KDD-98 exhibit presentation. http://www.epsilon.com/kddcup98/mining/, 1998.

[WLA] Webtrends Log Analyzer. http://www.webtrends.com.

[Zad79] L. A. Zadeh. A theory of approximate reasoning. *Machine Intelligence*, 9:149–194, 1979.

[ZXH98] O. R. Zaiane, M. Xin, and J. Han. Discovering Web access patterns and trends by applying OLAP and Data Mining technology on Web logs. In *Advances in Digital Libraries*, pages 19–29, Santa Barbara, CA, 1998.