Problems Modeling Web Sites and User Behavior

Spencer Rugaber, Nissim Harel,
Srihari Govindharaj
Georgia Institute of Technology
{spencer,nissim,srihari}@cc.gatech.edu

Dean Jerding
Infinovate, Inc.
djerding@comcast.net

ABSTRACT
As the world wide web has grown in size and scope, so too has the demand for analysis tools that can help web site providers determine how their sites are being used. Early analysis approaches focused primarily on accesses to web documents as recorded in web server logs. More recent techniques create a model of a site, and the natural modeling approach is to use a directed graph, where pages are denoted by nodes and links are modeled by edges. The process of creating the model and then analyzing the corresponding visitor traffic, however, is fraught with difficulties. The contribution of this paper is a catalog of problems gathered from extensive experience modeling web sites to determine site structure and analyze user behavior.

1. MOTIVATION
Web sites have become a central part of business marketing, sales, and customer service. Complex businesses engender complex web sites comprising thousands of pages. Poorly organized sites can give an unfavorable impression, reduce sales opportunities, and frustrate customers, possibly leading to expensive direct contact with customer service representatives in situations where a significantly less expensive web solution already exists.

It is therefore desirable for a company to organize its site as effectively as possible. To do so requires analyzing the site and the traffic it experiences. Analysis in turn requires modeling. A typical modeling approach uses a directed graph with nodes denoting pages and edges denoting links. This paper is derived from experience modeling a variety of sites. Specifically, it presents a catalog of stumbling blocks that may be encountered when using such graph models.

2. PAGES AND NODES
There are several problems with representing a web page as a node in a graph. The page itself must be parsed in order to determine its relation to other pages. Certain kinds of pages such as forms and frames must be treated specially. Finally, dynamically created pages of various sorts significantly compromise the precision of analysis.

2.1 Parsing
To model a web page as a node means that it must be possible to determine what other pages the given page refers to. This requires parsing (structurally analyzing) the page. Parsing, in turn, requires knowledge of the type of the page, usually determined from its MIME type [8]. Undoubtedly the most commonly occurring type of page is an HTML document. The following problems occur when parsing HTML documents.

1. HTML has gone through many versions.. Parsing a site may mean applying parsers with different levels of compliance to HTML standards, depending on the versions of HTML used to compose its pages.

2. Because most browsers are forgiving, many supposed HTML pages actually violate HTML syntax. In this sense, a site analyzer must be flexible with respect to what it actually accepts.

While parsers for HTML are readily available, web sites may serve a wide variety of other file types. For text files, parsing to detect URLs is straightforward. But many other types of files can contain links. For example, Microsoft Office files, PDF, and Macromedia Flash can all contain URLs. Parsers for these types of files may not be available.

3. Proprietary file types may not be readily parseable, leading to incomplete graph models.

2.2 Interaction

HTML includes the ability to express interactive *forms* using interactors such as buttons, text-entry areas, and menus. User interactions are sent to a program (*script*) running on the page's server. Technologies supporting this capability include JSP [10], ASP [1], PHP [13], and CGI [5]. The program analyzes the received data detailing the user interaction and produces a new web page as a result.

4. A page analyzer may be unable to comprehensively generate the set of all possible user inputs. This means that for a site containing forms, it may be impossible to determine the actual reachable target pages.

When user interaction data is encoded to be sent to the server, it is included either as part of the *query string* field of the URL using a *GET* method or as an encoded message body in a *POST*. For *GETs*, due to the vagaries of the browser and the way forms are written, identical interactions may be encoded with query parameters in a different order or with duplicate entries. It may be necessary to canonicalize the query (put it in a standard order), in order to adequately process it.

5. URL canonicalization may be required to adequately process query strings.

Scripts may be simple, or they may invoke arbitrarily complex functionality. One common type of web service is when an HTML form acts as a front end for a database. The recipient script processes the supplied user data and constructs an SQL query to an associated relational database. The results of the query are formatted by the script into a web page for return to the browser. By standard, the *GET* should not alter the state of the data on the server. However, this rule is often not followed in practice. Additionally, multiple *GETs* with the same parameters can return different results if the data has changed on the server due to other reasons. In general, server state can affect script processing and returned pages.

6. The content returned by a given *GET* request can vary depending on server state.

In the simplest case, the requested web page already exists on the server. The HTTP response returns code 200 and the requested URL is returned in the location field. However, when processing *POSTs*, new content can be created on the server, and a different URL location can be returned. Unfortunately this can even happen for *GETs*—the script can always change the URL or redirect the request. The fact that the address of the new page can be produced by the server, rather than occurring explicitly on the web page, is problematic.

7. Form-based page referencing complicates analysis of the outgoing links from a page.

More troubling is when the script processing the form data constructs a new page and sends it to the browser for viewing. It is often the case that the constructed page never actually exists as a file on the server.

8. Pages may be constructed dynamically by a program running on the server. Knowledge of server state may be used in web page construction that an analysis tool cannot duplicate.

An analogous set of problems arise with dynamic HTML (DHTML [6]). DHTML extends HTML in non-standard ways to include browser-executed Java applets or scripts, such as JavaScript [7], and style information, such as provided by Cascading Style Sheets [3]. These technologies not only complicate or prevent parsing, such as was described in Problems 1 through 3, but they may generate URLs that do not explicitly occur in the page.

9. Parsing and analysis of pages containing DHTML may either be complex or infeasible.

As the use of the web expands, more and more application capability is provided by web sites. Some of the capability makes use of server state. To the extent that user behavior and page presentation depend on server state, analysis based only on log files is weakened.

10. Server state can affect user behavior in ways that are invisible to log file analysis.

2.3 Frames

Although understanding the static structure of a web site is necessary to improving it, some analyses need to go beyond the question of what pages exist to consider what the user actually sees. Such a situation obtains when a site makes use of HTML *frames*. A frame acts like a web document embedded inside another page. The embedding is recursive, allowing an arbitrary complex nesting hierarchy. Because URLs can refer to specific frames within a page, the simple model of one node per page breaks down.

11. Modeling frames means that the simple model of one node per page is inadequate.

If a page contains frames, and if a frame may hold further pages, each of which can change, there is a combinatorial explosion in the number of different web displays a visitor can see. If it is of value to know exactly what the visitor is experiencing, then the various combinations must be modeled.

12. Modeling frames may lead to an explosion in the number of nodes in the graph.

Even more complex than frames, is handling *portlets* [11]. Portlets enable users to specify both the layout and content of a web page. Content is selected from a variety of available components such as news, weather, sports, etc. To the extent that analysis of visible content is desired, portlets lead to problems similar to Problem 12; to the extent that the user can dynamically determine content, portlets lead to problems similar to Problems 8 through 10.

13.Portlet technology severely compromises analysis of web page content.

2.4 Special Types of Pages

Other modeling problems arise for certain kinds of pages. For example, a common feature of a web site is a page that enables the visitor to see the overall structure of the site and quickly visit any of the site's pages. These pages are called *site maps*. Site maps, however valuable they are to the visitor, can complicate a site model. They generally have the property that they link to many if not all of the pages in the site.

In addition, depending on site policy, every site page may contain a link to the site map. Consequently, the site model will contain a node with links from all other nodes. Not only does this add clutter to any visualization of the model, but laying out such a graph for presentation can hide interesting structural features of the site such as clusters of related pages. Therefore, it may be desirable to eliminate site maps from graph models.

14.Site map pages may need to be treated specially in order to improve visualization.

Similar to site maps are *search-request* and *search-results* pages. Many sites allow visitors to search based on occurrences of keywords. Sometimes the searches are conducted from existing site pages, such as the home page, and sometimes a special search-request page exists. To the extent the latter is used, is suffers the same problems as a site map page does, because most pages link to it.

15.Search-request pages may need to be treated specially in order to improve visualization and analysis of a site.

After a search request is made, it is processed by the server, which is responsible for constructing a page that contains links to matching pages. To the extent that search-result pages are constructed, they suffer the same problems as any constructed page (see Problem 8). Moreover, search-result pages have the potential for linking to any subset of site pages. In this way, search-result pages act like site maps. The possibility of having a large number of graph nodes, each capable of having a large number of outgoing edge, not only obfuscates the visualization but potentially distorts the original intent of the model—to delineate important features of a site's structure.

16.Search-result pages may have to be treated specially to improve site analysis.

A related problem concerns *exit* pages. An exit page may be the "official" way to leave a site, particularly if site security is an issue. For example, on an exit page, a user may be warned they are leaving the site or be explicitly asked to log out of the site. Like site maps, exit pages have the property that they are the targets of many internal links.

17.Exit pages may have to be treated specially to improve graph visualization and analysis.

Similar situations arise for links to other common page types such as privacy pages and home pages.

Another example of a special kind of page can be found at a site providing weather reports. From the point of view of the site, the entry page for a particular geographical location changes day-to-day or even more frequently. From the point of view of the user, the same URL is used for access regardless of the day. For the purposes of modeling, it may be desirable to treat all versions of a page as a single node. This requires special processing at the time of modeling.

18.Pages that over time serve identical roles may need to be treated as if they were a single page during modeling.

Alternatively, it may be of value to treat the various versions of a page as separate entities. For example, if the weather on a particular day was notable, the site administrator might wish to know how visits to it differed from the day before or the day after.

19.Pages that share a URL may need to be modeled as separate nodes.

Another form of sharing takes place at sites where users can login. It may be the case that logging in actually takes the user to a separate, but similar-looking site, perhaps with access to some extra files or providing an extra measure of security. From the point of view of the user, there is only one site, and it may be desirable for any site model to reflect this. On the other hand, it may be the case that a single URL denotes different pages (at different sites) depending on whether the user is logged in.

20.Sites that support user login may require special processing. Identical URLs may identify different pages, and, depending on the purpose of the analysis, it may be desirable to treat the pages as either separate or identical.

3. LINKS AND EDGES

Links are just as important as pages in modeling a web site. Links denote hypertext references indicating target pages a user may visit. The natural representation for a link is a directed edge in a graph. However, as with nodes, difficulties arise when using a simple graph to model complex page interconnections.

One class of difficulty arises because a given page may contain more than one reference to the same target page. That is, there might need to be two edges between the same pair of nodes. Simple graph models do not allow for this possibility, so the modeler is left with the choice of information loss (using a single edge to denote multiple connections) or using a more complex modeling technique, such as hypergraphs [9].

21.Multiple, identical URLs on a page cannot be modeled using simple directed graphs.

Another, similar situation arises with so-called *popup pages*. A user can click on a link to visit a new page and, as a side effect, an additional window is displayed, often containing some form of advertisement. That is, a single link actually traverses to two pages.

22.Popup pages require one apparent link to be modeled with two edges.

Popups can also occur when a page is left. While with page-entry popups, an originating page and single URL are associated with two or more target pages, with page-unload popups, an originating page and *any* target URL can lead to the appearance of the popup.

23.Popups on page unloads can lead to multiple edges for any number of target URLs.

A URL can refer not only to a page, but to a specific location (*anchor*) within the page. An edge incoming to a node is insufficient by itself to indicate where in the page the URL points.

24.Links to anchors cannot be directly modeled with simple edges.

URLs can refer to frames within pages. Problem 11 expresses the limitation that simple nodes cannot adequately model frames. A further problem arises due to the `TARGET` attribute of the HTML `<A>` tag. `TARGET`s indicate how the target page will be displayed. For example, if the new page is to entirely replace the current page in the browser window, then `TARGET=_top` can be specified. Alternatively, if the new page should replace the existing page within the frame containing the link, then `TARGET=_self` can be used. Other options allow the new page to be displayed in an new window (`TARGET=_blank`) or to have the new page display in the frame that contains the URL's frame (`TARGET=_parent`). If the purpose of the modeling is to indicate the different displays the user can experience, then `TARGET`s must be included in the model.

25.The combination of nested framesets and `TARGET` attributes can lead to a combinatorial explosion in the size of a graph.

Web sites are not self-contained. That is, web pages link to other pages not on the site, and other sites may contain referring pages. It is of course possible to ignore these situations, but it may be useful to know where visitors are coming from and where they are going. Similarly, it may be of value to know which pages are the first pages at a site that visitors see (*entry* pages) and the pages from which they leave the site (*exit* pages). Hence, it may be of value to include information in the model concerning pages and links that are not part of the site.

26.External pages and links either significantly enlarge a model or require special handling.

4. SERVER AND SITE ISSUES

To model a site, we must have a clear definition of what a site is. A naive definition for *site* is a set of pages whose URLs share a common server name. However, several complications arise with this simple definition.

The first complication is that there are two ways of naming the same server—its IP address and the corresponding host name. A further complication is that server naming may be qualified by port number.

27.IP addresses, host names and port numbers must be matched to model link targets.

Other problems arise when a given site has multiple names. One name is designated as the *canonical hostname*, and the others act as *server aliases*. For example, large corporations often purchase a set of typographically close versions of their preferred name so that confused customers end up at expected places.

28.Multiple different names can refer to the same site and consequently the same page.

Finally, it is also possible to have separate servers (physical machines) with separate IP addresses serving the same set of web pages. Determination of this situation is site specific, making detection difficult.

29.Multiple servers may serve the same web site making link resolution difficult.

The two sources of information for building models—spiders and log files—are both time dependent. That is, a spider runs at a particular time and takes a certain amount of additional time to analyze a site. Log files contain entries that cover a specific period of time. While these activities are going on, the site may change.

30.The dynamic nature of a web site may compromise the accuracy of the models produced.

An implication of the time-dependence of the data sources (spiders and log files) is that they may not be consistent with respect to each other. An explicit check may have to be made to determine if there are discrepancies. Manual intervention may be required.

31.The consistency of spidering and log files may be compromised by the dynamic nature of web sites.

5. REFINED MODELING

There are many reasons why it might be of value to model a web site. A simple graph model, as described above, enables the use of concepts from graph theory such as paths, distance and clusters. However, there might be analysis needs that go beyond what a simple directed graph can express.

A simple example of a situation where analysis needs might complicate modeling is where multiple links exist between a pair of pages. A simple approach to dealing with this problem is to treat multiple links on a page targeted at the same page with a single edge. If, however, the analyst needs to know which of the links a user traversed to visit the target page, then this approach is insufficient. A richer alternative is obtained by *tagging* the links to include additional information in URL query strings indicating which URL occurrence is being followed. Modeling in these situations requires defining an encoding convention, altering the web pages, and parsing the query strings appropriately.

32.Using different tags to discriminate identical URLs on the same page may require extra work for content providers and analysis tools.

Web sites may be quite complex, comprising thousands of pages that change from day to day. To ease analysis, it may be desirable to abstract the site in various ways. For example, it might be useful to consider a subset of site pages as a single unit, as far as analysis is concerned. A related situation arises when it is desirable to ignore the details and interconnections for a subset of pages.

Websites can be partitioned either physically or logically. Physical partitions may be served by separate machines acting conceptually as a single server. Logical partitioning may correspond to organizational units of the business or segmentation of the resources served. In any cases, a specific analysis may target or ignore a given portion of a site.

33.Logical and physical partitioning of web sites for the purpose of abstraction can require significant additional work.

Another form of abstraction typically applies when a model is being displayed graphically. If a complex site is displayed as a graph without consideration of layout, it may appear incoherent and consequently be of little value. One way of better organizing a display is to use *distance from an entry page* as a metric. That is, entry pages are given distance zero; pages that entry pages link to are distance one, and so on. If this is done, then the visual display can be organized into levels. Distance-zero nodes are placed at the top of the display, distance-one pages appear horizontally aligned in the next row down and so on.

Of course, a given page may be reached from multiple entrance pages. Moreover, even from a single entrance page, multiple paths may exist to a given page. What is desired is the length of the shortest such path. Algorithms exist for computing distance, but, in the case of rapidly changing sites, must be frequently reapplied.

34.Distances must be computed in order to adequately display levels for a site.

Many sites have clearly defined entry points, but determination of entry points at other sites may be difficult. One recourse is to obtain a list of such pages from the owner of the web site. Alternatively, entry pages can be inferred from traffic at the site.

35.Distance computation requires assumptions concerning entry pages.

Even if it is possible to cleanly model a complex web site, complications may still arise if the site changes rapidly. For example, a business may wish to know how site traffic changes as a result of a marketing campaign. To make such comparisons, it may be of value to relate today's graph model with yesterday's.

Comparing graphs is known to be a computationally difficult task. But the problem becomes even worse when the content of individual pages can change from day to day. The question arises as to whether a given page, when edited to add new content results in a new page. The simple answer is that if the URL is the same, then it is the same page, but there might be situations, particularly when pages are dynamically generated, where it is of value to identify separately generated pages in two models. This may require costly intervention on the part of the modeler.

36.Comparing models of the same site at different times may require case-by-case handling.

6. TRAFFIC ANALYSIS

Ultimately, the purpose of a web site is to serve its visitors, and web site providers often have in mind classes of visitors and desirable behavior for them. For example, a site offering products for sale wants to make it as easy as possible for visitors to complete the purchasing process. This goal raises several questions: Can classes of users be identified? Can visits be categorized as successes or failures? And can sites be reorganized to increase the success-to-failure ratio? To answer these and related questions, it is desirable to analyze visit histories.

6.1 Log Files

There are typically two sources of information used to build a site model—the pages of the site itself and one or more server-generated log files. The former can be explored by a spidering program; the latter requires a log-analysis program to be written. Spider-generated data provides a static picture of a web site. Within the limits prescribed by the problems described above, a spider's picture of a site is definitive. On the other hand, a spider can say nothing about how a site is actually used by its visitors. Server logs help address this need. They include information such as the date and time of a site access, the referring page, the type of browser and operating system used, and the requested URL. Although Apache web servers are the most common, and Apache specifies the contents of its log files [2], variations exist that require extra processing.

37.Different web server programs provide log files in different formats. A general log file analysis tool must handle a variety of formats.

38.The Apache web server allows a site administrator to configure the contents of that site's log files. Once again, a general tool must be able to deal with this sort of variation.

Log files are voluminous and may contain uninteresting information. For example, if search engines regularly visit a site in order to update their indices, it may be useful to filter out this traffic.

39.Visits from search engines can obfuscate web server logs.

A log file entry normally indicates that a specific visitor was served a specific page. In the case where the user has clicked a link, the entry will contain the URL of the referring page. There are various situations, however, where this information is not available. Examples include when the *BACK* button was used, or when the user typed in the URL explicitly, or the browser's history or bookmarking mechanisms were used.

40.Log file referral data may be incomplete when direct user type-in, history access, bookmarks, or scripting are used.

Many types of user link-following requests may actually be handled by the browser. For example, links that map to another place in the same page, uses of the *BACK* button, and other occasions when the target page is cached may not appear in the log file at all.

41.A server log contains only a partial record of a user's link-following activity.

6.2 Identifying Visitors

Of course, a log file may not actually indicate who a visitor is. It only contains information on the IP address responsible for a specific page request. If multiple visitors are using the same computer, this may give the appearance of one very active user.

42.Independent users of the same computer may be confounded in a log file.

ISPs often provide single IP addresses for a collection of users. Similarly, businesses behind firewalls identify as one a community of users.

43.In situations where a set of machines shares a network access point, an entire business or community of users at an ISP may be referred to by the same IP address in web logs.

Some ISPs provide so-called *rolling IP addresses* to their clients. That is, the IP associated with a given user may change over time. IP-based visitor identification can be thus compromised.

44.Rolling IP addresses can reduce the accuracy of visitor identification.

As indicated above, IP addresses in log files are a crude source of information for identifying visitors. Some sites provide mechanisms for improving precision. For example, a *cookie* is a common means of associating data with a specific user and storing that data on the user's computer. The data can include a visitor identifier that can be retrieved upon subsequent visits to the site. Even if that user visits the site from different computers, the common user can be identified. However, some users may configure their browsers to disallow cookies; and if visitors connect from different computers, their cookie data may not be synchronized, making it more difficult to identify them.

45.Although cookies can help identify visitors, the analysis process for handling them becomes significantly more complicated.

Some large companies that use cookies may, in fact, provide multiple web sites. They must implement the cookies in such a way that the visitor can be identified across the different web sites. While the company would like to know about that user's behavior regardless of which site they visit, the company might also like to differentiate the uses. The log files can contain sufficient information to do this, but, once again, analysis becomes more complicated.

46.Multiple sites that share cookie policies can make analysis more fruitful and more complex.

Of course, users may have more than one cookie, or a given cookie may represent more than one user, and users can delete cookies. Hence, visitor identification is approximate.

47.Cookies, while improving the precision of users, do not guarantee perfect accuracy.

A related scenario occurs at sites that require users to login to access certain resources or to engage certain services. The logging process takes the user-supplied data (name and password) and examines a database. This enables a more precise identification of those users, but it does complicate the analysis.

48.Sites supporting user login can provided additional information helpful to identifying users, but analysis is thereby complicated.

6.3 Sessions

One use of log data is to try and understand the specific behavior of a visitor. For example, if the site supports product perusal and purchase, it may be useful to know what happens with visitors who spend time reviewing and then do not purchase. To understand this kind of behavior requires associating disparate log file entries. That is, one entry might report a user visit to a company's home page; several intervening entries may correspond to other visitors; the next entry might indicate that the original visitor has decided to view a certain class of products, and so on. The history of a user's visit to a site is called a *session*. Determining sessions requires that log file entries be categorized according to the visitors involved. This is a complex and approximate process depending on both successful visitor identification as described in Section 6.2 and on determining how to group visitor log file entries into groups, each corresponding to one session.

49.Session definition is inherently approximate and error prone.

Determining visitor behavior heavily depends on being able to recreate the sequence of pages visited. Fortunately, for each page visited, the corresponding log file entry contains information about the referring page. This helps an analysis tool connect page hits to visitor paths. Unfortunately, some older servers do not supply the referring URL, significantly compromising the accuracy of the session reconstruction.

50.Some older servers do not provide referral information thereby reducing the quality of session reconstruction.

One form of approximation occurs when a visitor is mentioned in different log file entries spread out over time. Is this one long session or multiple short ones?

51.Log file entries referring to the same visitor need to be partitioned into sessions. This discrimination can be site specific.

Another problem arises when a visitor returns to a previously visited page. If the page has been cached by the visitor's browser, no log file entry will be created, further compromising of the analysis.

52.Browser page caching may hide page visits.

A given log file records visits that occurred during a specific time period. It may be the case that a visit starts during the time span covered by one log file and ends during that recorded by the next. To accurately compute the session requires identifying the visitor in the second file with that in the first.

53.Information from separate log files may have to be coordinated to compute sessions.

If the site is served by multiple servers, each with their own log files, then an accurate picture of the site requires combining these statistics.

54.Sites with multiple servers must comb log files to get an accurate picture of site traffic.

Further complicating matters is the fact that the clocks on the various servers may not be synchronized or that the log file recording schedules might not align.

55. Log file recording schedules and server clocks may cause accuracy loses in computing site traffic.

As mentioned in Problem 48, some sites support user login. While this can improve the accuracy of visitor identification, it does raise the need to identify those log file entries for the visitor that occurred before login with those afterwards.

56.Sites providing login require additional analysis to relate requested pages before and after login occurs.

Problem 47 discusses cookies. Some sites, after a user logs in and the server confirms the user's identity, record that identity in a cookie. To the extent this technique is relied on for session determination, it suffers from the problems described in Problem 47.

Session definition normally requires some heuristic to distinguish between one very long visit by a user or two separate visits by the same user. A natural rule is to treat entries separated by more than some amount of time, such as one hour, as belong to separate sessions. But modern web technology can interfere with the effectiveness of this simple rule. For example, a user visiting a sports web site that automatically refreshes a page every ten minutes might result in multiple weblog entries for that user, even though the user has left for the day. Dynamic elements such as advertisements, Flash or applets can behave similarly.

57.Session partitioning based on time of activity can be artificially distorted by page refreshes and other technology.

A problem arises in the other direction as well. For example, if a user is viewing a downloaded video, the actual session may cover a long period of time even though no subsequent log entries are being made.

6.4 Redirects, Refreshes, and URL Rewriting

When a user clicks a link or types in a URL, information is sent from the browser to the server requesting a page in response. It may not be the case, however, that the pages displayed is the one requested. The server can substitute a page and *redirect* the user to it. For example, the requested page may have been moved and a replacement page left in its place that redirects the visitor to the new target. In these situations, the log file can contain two entries: one for the originally requested URL and one for the replacement. The log file analyzer must be aware of this possibility in order to model what the user experienced and not how the server implemented the request.

58.Server redirects must be compensated for during log file analysis.

A related situation arises for pages that *auto refresh.* For example, if a visitor is tracking a sporting event, the server may push an updated version of the page to the browser periodically. In these situations, one log file entry can be written for each push. Depending on the analysis requirements, it may be desirable to coalesce the entries or to treat them separately.

59.Log file analysis tools must be able to recognize and deal with auto-refresh pages.

An even more general situation arises due to the server's ability to rewrite URLs. One common example is mapping from URLs specifying directories into the `index.html` file within that directory. Normally, these situations are treated by the server like redirects, implying multiple log file entries. Other times only the returned URL is recorded in the log file, depending on the web server.

60.Server URL rewriting must be compensated for by a log file analyzer tool.

7. CONTEXT

The problems catalogued in this paper have been encountered by analyzing a variety of web sites using a proprietary behavior intelligence system. This tool models web sites and processes web server logs in order to better understand visitor behavior. The problems documented here manifested themselves in scores of web sites analyzed over four years. These sites can be characterized by the following attributes:

- **Purpose:** e-commerce, news, sporting events, corporate promotion, informational, and company Intranet
- **Size:** upwards of 20,000 pages
- **Technologies:** HTML, JavaScript, CGI, PHP, ASP, JSP, Flash, Java applet, XML/XSL/XSLT, CSS, etc.
- **Traffic volume:** up to 500,000 visitors per day

8. RELATED WORK

Sirkant and Yang [15] use web logs and site structure to suggest site reorganizations. Specifically, they infer visitor navigation difficulties from time spent viewing pages and from actual and inferred use of the *BACK* button. They also make assumptions to distinguish between "content" pages and "navigation" pages. Both of these classes of heuristics can be thought of as limitations to accurate site and log file analysis.

Catledge and Pitkow [4] analyzed (browser) user-event logs to better understand user browsing behavior. In particular, they distinguished between directed search, general purpose browsing, and random (undirected) exploration. They characterized events in terms of application orientation, mid-level tasks (e.g. menu selection), and low-level interfaced techniques. To do this, they had to compute sessions, which they did with timeout heuristics. Because they had browser data, they avoided caching-related information loss. They were able to characterize user behavior in anticipation of improving browser design.

Sarukkai [14] describes a technique for modeling web server and log data to afford prediction of next link, next page, and visitor path. The model is probabilistic, making use of Markov chains. Session construction is mentioned, indicating that a time-out threshold is used. Suggested applications include navigation agents, tour generators, and personalized hubs and authorities.

Nakayama *et al.* [12] explore situations where the topology of a web site does not adequately reflect the relationship among its content pages. They compare page similarity measures to session co-occurrences to suggest site improvements. An interesting heuristic they apply distinguishes content pages from index pages by the number of links they contain. Index pages are then removed from further consideration.

9. REFERENCES

[1] Active Server Pages. Microsoft Corp. http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnanchor/html/ activeservpages.asp.

[2] Apache HTTP Server Version 1.3, Module mod_log_config, http://httpd.apache.org/docs-/mod/mod_log_config.html.

[3] Cascading Style Sheets. World Wide Web Consortium. http://www.w3.org/Style/CSS/.

[4] Lara Catledge and James Pitkow. Characterizing Browsing Strategies in the World-Wide Web. *Fourth International World Wide Web Conference,* December, 1995, Boston, Massachusetts.

[5] The Common Gateway Interface. http://hoohoo.ncsa.uiuc.edu/cgi/overview.html.

[6] Dynamic HTML. Wikipedia. http://en.wikipedia.org/wiki/DHTML.

[7] *ECMAScript Language Specification. Standard ECMA-262, 3rd edition* (December 1999). http://www.ecma-international.org/publications/files/ecma-st/ECMA-262.pdf.

[8] Freed, N. Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types. *Internet Working Task Force, Request for Comment (RFC 2046)*, November, 1996, http://www.ietf.org/rfc/-rfc2046.txt?number=2046,

[9] David Harel. On Visual Formalisms. *Communications of the ACM,* 31(5):514-530, May 1988.

[10] JavaServer Pages Technology. Sun Microsystems. http://java.sun.com/products/jsp/.

[11] JSR-000168 Portlet Specification. http://www.jcp.org/aboutJava/-communityprocess/review/jsr168/.

[12] Takehiro Nakayama, Hiroki Kato, Yohei Yamane. Discovering the Gap Between Web Site Designers' Expectations and Users' Behavior. *Fourth International World Wide Web Conference*, Amsterdam, The Netherlands, May, 2000.

[13] PHP. http://www.php.net/.

[14] Ramesh R. Sarukkai. Link Prediction and Path Analysis Using Markov Chains. *Ninth International World Wide Web Conference,* Amsterdam, May 15-19, 2000.

[15] Ramakrishnan Sirkant and Yinghui Yang. Mining Web Logs to Improve Website Organization. *WWW10,* ACM, May 2-5, 2001, Hong Kong.