In Search of Reliable Usage Data on the WWW

James Pitkow
Xerox Palo Alto Research Center
Palo Alto California 94304 USA
pitkow@parc.xerox.com

Abstract

The WWW is currently the hottest testbed for future interactive digital systems. While much is understood technically about how the WWW functions, substantially less is known about how this technology is used collectively and on an individual basis. This disparity of knowledge exists largely as a direct consequence of the decentralized nature of Web. Since each user of the Web is not uniquely identifiable across the system and the system employs various levels of caching, measurement of actual usage is problematic. This paper establishes terminology to frame the problem of reliably determining usage of WWW resources while reviewing current practice and their shortcomings. A review of the various metrics and analyses that can be performed to determine usage is then presented. This is followed by a discussion of the strengths and weaknesses of the hit-metering proposal [Mogul and Leach 1997] currently in consideration by the HTTP working group. Lastly, new proposals, based upon server-side sampling are introduced and assessed against the other proposal. It is argued that server-side sampling provides more reliable and useful usage data while requiring no change to the current HTTP protocol and enhancing user privacy.

1 Terminology

Despite several efforts and widespread agreement on the need to establish a lingua-franca for usage and demographic collection, consensus does not exist [W3C 1996]. One of the more recent and comprehensive efforts [Novak and Hoffman 1996] provides a baseline of terminology specifically designed for the advertising and measurement communities. With the intent of generating consensus, this paper will build upon their framework for classifying visitors and terminology, clarifying and introducing new terminology as needed. Readers familiar with the notions of cookies, hits, and the problems local caches and proxy-caches have on reliably determining usage may wish to skip to the next section statistical analysis.

1.2 Visitors and Cookies

The central issues regarding the classification of visitors to a Web site are the ability to uniquely identify visitors and the ability to do so reliably, especially across multiple visits to a site. Visitors to WWW sites can be separated into the following categories: unidentified, session, tracked, and identified [Novak and Hoffman 1996]. For each class of visitors, a definition is provided followed by a discussion of the methods used to achieve each level of knowledge about visitors to a site.

A *unidentified visitor* is a person who visits a Web site where no information is available about the visitor. This type of visitor does not truly exist on the Web since the Internet Protocol requires at least a machine name to return the requested information. This form of return address reveals information about the users in a similar manner to that of a phone number, where a one-to-one or a many-to-one correspondence may exist between the address and the number of users at the address. Unidentified visitors may indeed exist in other interactive digital systems, where a user's anonymity is explicitly preserved. It is arguable that anonymous proxies, programs that acts as a intermediary between clients and servers, enable users to experience the Web in an anonymous manner. While this form of server may exist to a limited number of users, it does not scale well to handle the entire user population since it effectively doubles the amount of traffic required to request a page (by first sending a request to the anonymous server which then sends a second request to the actual server) and requires a fair amount of centralization of resources, another potential bottleneck.

A *session visitor* is a visitor to a Web site where an identifier is created either explicitly via cookie[1] generation or inferred through heuristics as discussed below. This is the default type of visitor on the WWW today. Several revealing pieces of information are typically available which enable the heuristic identification of users even if cookies are not used. With each request, information about the machine name from which the visitor made the request, the type of software the visitor is using to experience the WWW, the operating system on which the software operates, and the page viewed prior to the current request is typically known. The latter piece of information is called the referrer field. While this ancillary information may enable a user to be identified within a session, it is not guaranteed to be accurate nor will it be able to reliably identify the same user in future sessions. Some heuristics for identifying users without cookies include:

- The use of Internet protocols to help determine if the user is the sole user of the machine making the request, e.g., identd, finger, etc. If a one-to-one correspondence exists between a visitor and a machine, the machine essentially becomes a unique identifier, and the visitor becomes a 'tracked visitor' as described below. These techniques fail if a visitor exists behind a proxy (as is the case if Figure 1), shares machines with other users, or the machine being used to experience the Web does not support or allow these protocols.

- To uniquely identify users suspected of existing behind proxies (see Figure 1), session limits, the site's topology (the global hyperlink structure across pages), and browser characteristics can be used. One such algorithm implemented in [Pirolli, Pitkow, and Rao 1996] checks that each incoming request is reachable from the set of already visited pages. This is done by consulting the site's topology. If all subsequent requests are made to pages that the visitor could have reached by selecting a hyperlink embedded in any of the already requested pages, the user is assumed to be the sole visitor behind the site. If requests from the same machine name occur for pages that are not reachable from the set of hyperlinks embedded in the pages already visited, multiple visitors are suspected. Multiple visitors are also suspected when pages are requested that have already been visited. The algorithm treats these cases as separate visitors and adds subsequent page to each visitor's path based upon the topology of the site. A least recently used policy is used to add pages to visitors if ambiguity exists between which user could have made the request. Visitors who do not request pages within a certain time limit are assumed to have left the site. Appropriate time-out periods are typically determined by inspection of the distribution of time between all page requests to a site. While the above algorithm performs reasonably well, it is heuristic in nature and has not been shown to reliably identify users with any measure of accuracy, especially across sessions.

A *tracked visitor* is a visitor who is uniquely and reliably identifiable across multiple visits to a site. In the earlier days of the Web, the tracking of visitors was often accomplished by inserting identifiers into the URLs issued by the server and channeling all subsequent requests through a CGI script. Not only was this method expensive computationally to the sever, but it defeated intermediary caching and did not correctly handle the exchanging of URLs between people, i.e., the person using a URL of this sort mailed to them by a friend could be incorrectly tracked as the friend. These days, this form of identification is typically accomplished by setting the expiration of an issued cookie into the far future. Thus, each time a visitor returns to the site, the same identifier will be used.

The increased use of this technique to track users has not been without notice by the user community, where rather rudimentary but effective practices have emerged that periodically erase cookies stored on the visitor's filesystem or do not permit the storing of cookies between sessions by disabling write

---

[1] Cookies are server generated identifiers that enable the management of state between visitors and servers. They were initially designed to implement shopping baskets for the Web but have found a killer application in the tracking of user behavior. When a visitor requests a page, a server can return an identifier, a.k.a. cookie, with conditions on when and how the identifier is to be used.

permissions to the appropriate files. These practices have the effect of causing the site issuing the cookie to issue another identifier, resulting in potential over-inflation of the number of unique visitors to the site. Commercial software that performs cookie obfuscation is also emerging, e.g., PGPCookie.Cutter [PGP; 1996].

An *identified Visitor* is a tracked visitor where additional information is available. This is the most common type of visitor when persistent identifiers are employed to monitor usage. While it may appear that tracked visitors would be more common, additional information as mentioned in the above section of session visitors accompanies each request. Rough estimates of the demographics of a user can be made since the entity that owns the domain from which the request was issued can be determined somewhat reliably from InterNIC's publicly accessible domain registration database. Once the entity have been established, this information can be matched against other databases that enable the construction of user profiles. For example, if a request comes from a visitor behind a corporate proxy named xyz.com, via InterNIC's database, one can discover that the actual company that owns that domain is FooBar Corp., and then lookup information on FooBar Corp. in other sources. Many of the commercially available log file analysis programs come with databases that match domain names to core demographics, e.g., Interse [Interse 1996].

Of course, the most obvious method of collecting additional demographics of users is by asking the actual users of the site. This is routinely accomplished via online registration forms. However, GVU's most recent WWW User Survey data show that 33% of the over 14,500 respondents have falsified the information for online registration forms at least once [Pitkow and Kehoe 1996]. Over 10% reported that they provided incorrect information over 25% of the time. Although online registration is currently common practice and will remain so for the foreseeable future,

> *the information collected from online registration systems needs to be thoroughly examined on a per-site basis before reliable statements about the users of a site can be made from this information.*

Other methods for collecting demographics of users at a site include Universal Registration Systems, e.g. I/PRO's I/COUNT [I/PRO 1996]. These systems require a user to register only once in exchange for an identifier. This identifier can then be used across the set of participating sites. While the goal is to 1) make registration easier for users and 2) provide sites with valuable demographic information, scalability issues have hindered the development of most of these types of systems, and as such, few Universal Registration Systems exist in practice today.

The above classification of visitors and the accompanying definitions should provide the necessary framework to move forward and provide definitions for the terms that are used to measure the items visitors request.

1.2 Accesses and Caches

When a visitor requests and item from a Web site and the server returns the item to the user, an *access* is said to have occurred. In the common speak of the Web, this event is fondly referred to as a '*hit*.' It is widely recognized that the reporting of hits as a measure of usage is meaningless for the following reasons. First, when a user accesses a page, the client sends a request to the server in the form of a URL. If the item being requested is an HTML page, it may include other URLs embedded into its content, which are often images, audio files, video files, applets, as well as other text. When the client gets the requested HTML page back from the server, it reads the contents of the page and makes subsequent requests for the embedded URLs. Thus, a page with three embedded images results in a total of four hits as measured by the server (one hit for the HTML page, and one hit each of the three images). Since the composition of pages differs within a site as well as across sites, the comparison of hits reported by different sites is useless. For example, in Figure 1, when Visitor 1 requests Page A, Server Z will record four hits but Server Z will only record two hits when Page C is requested. What sites really want to compare is the

number of pages requested by visitors, which is often called *page views*. Page views can be crudely measured by excluding non-HTML requests from the tabulation process.

The second major problem with hits occurs because of the various levels of caching that occur on the Web. Caching is a mechanism that attempts to decrease the time it takes to retrieve a resource by storing a copy of the resource at a closer location. Prior to the fall of 1994, most browsers did not include local caches. As a result, each time a page was accessed by a user, a separate page request would need to be issued to the original site and the contents returned each time to the user. To remedy this inefficient situation, browser software began to integrate caches that utilized the user's local disk space as well as in-memory caches that stored pages in memory rather than on disk.

For example, in Figure 1, when Visitor 1 requests Page A, Page A is stored on Visitor 1's computer in the local cache. When Visitor 1 goes off to visit Page B and then returns to Page A via the 'Back' button, a request to Site Z does not need to be made. Visitor 3 on the other hand would need to issue two requests for Page A to perform the same task. Upon tabulation of the access log for Site Z, one would find two different totals for the same navigation: Visitor 1 would record only one page view for Page A whereas Visitor 3 would record two page views. Which one is correct? Depends on the definition of a page view. Since local caching has become the standard in browsers and users can select different cache management policies, *page views* typically are defined to only reflect that the page was viewed *at least once*. Given this frame work, *unique page views* represent the first time a user requests a page in a session. *Reuse page views* refers to the total times the page is viewed minus the first page view., with *total page views* being the sum of the two. Without mucking with the configuration of the server to defeat local caching, it should be clear that no guarantees can be made about the measurement of total page views.

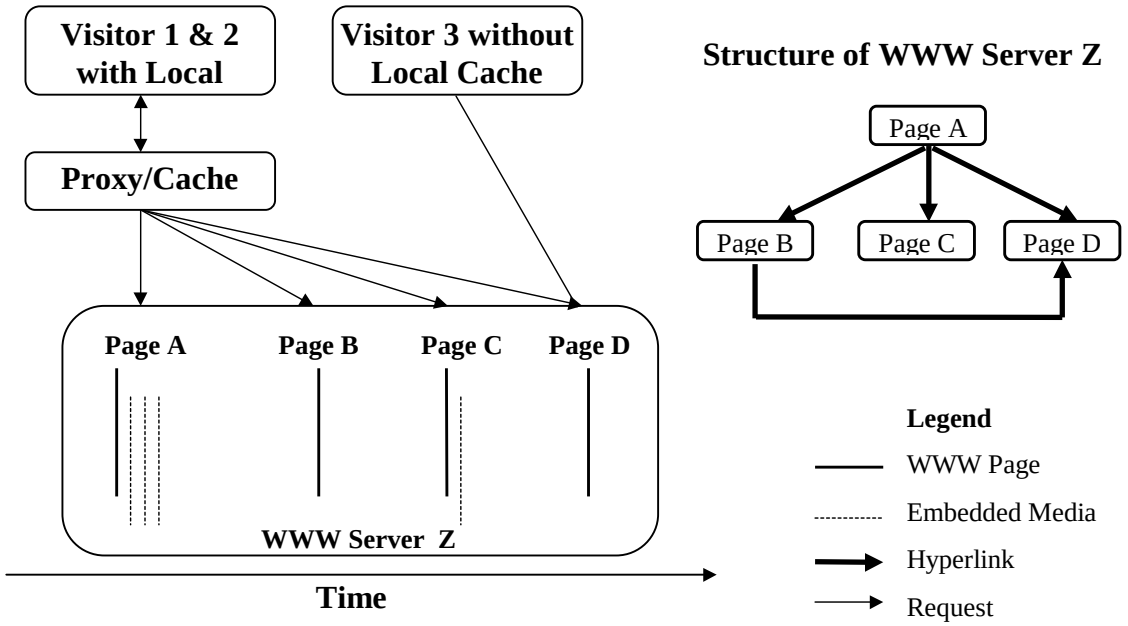

Figure1. This figure represents a visitor making requests for Pages A, B, C, and D at Site Z. Another visitor is shown making a simultaneous request for Page D. The topology of the site is depicted in the upper right hand corner. Since Visitors 1 and 2 are both behind a proxy-cache, their requests can not be differentiated without the use of either cookies or heuristics. It is also unclear from the requests make to Server Z whether Visitor 1 requested Page D by selecting the hyperlink on Page A or the hyperlink from Page B.

The other form of caching that occurs on the Web happens at the proxy level. The scenario is very similar to the one of local caches, but in this case, since the cache is shared by many users, the server may only

receive one page request even though the page was viewed by numerous users behind the proxy. In Figure 1, if Visitors 1 and 2 request the same pages with Visitor 2 making the requests at a later time than Visitor 1, the proxy-cache would only need to issue one request per page to Sever Z and use the copy stored locally to handle requests by Visitor 2. Note that combinations of local caches and proxy/caches can exist as well as multiple proxy/caches chained together. In order to help control the caching of pages by local caches as well as proxies, the HTTP has evolved to include cache specific headers.

A common resource intensive solution to the problem of reliably determining pages views and users is to use the cache specific headers in HTTP to effectively defeat all attempts at caching pages. Despite the reasons for employing such an approach, this technique is commonly referred to as *cache-busting*. This is accomplished in several ways including sending "Cache-control: proxy-revalidate", or "Expires: <past date> headers. In order for these attempts to be effective, the caches that are incurred along the way must cooperate, that is, they must obey the headers. Unfortunately, there is not way to ensure that all caches (local as well as proxy-caches) on the WWW cooperate. As a result, even with attempts to defeat caching, an unknown number of users and page views can occur without the knowledge of the originating server. This causes the total number of page views to be inaccurately reported and therefore should not be considered a trustworthy measure.

From this discussion it ought to be clear that the reliable gathering of visitor and page usage is not an easy task. The most widely used solution to these problems include the issuing of cookies to identify visitors and the defeating of caching to determine page views. This is only half of the picture however. Once the data is collected, it must be analyzed before it is of any use. The next section describes the various flavors of statistics one might want to apply to the gathered usage data.

2 Statistical Analysis

Since page requests are discrete events, they afford several forms of statistical analyzes, with descriptive statistics being the most rudimentary and most widely used for analysis of usage data on the WWW today. What one gathers from descriptive statistics of a set of events are the frequency, mean, median, mode, minimum, maximum, standard deviations, variance, and range. For accesses to WWW sites, the frequency of page views is the most commonly reported statistic, though as noted in the above section, this statistic may be difficult to determine reliably. This form of event analysis can occur on the page level as well as on the site level for all of the different types of visitors mentioned above.

2.1 Temporal Analysis

While knowledge of how frequently an event has occurred, it tells us nothing about the interactions between events. Temporal analysis of page request events can reveal several interesting metrics but assumes that visitors can be uniquely identified within sessions (session visitors, tracked visitors, or identified visitors). This requirement exists since the sequence of events is needed to construct the temporal ordering of page requests. The following discussion also assumes that the complete and correct sequence of page request is know, though as described above, this is often not possible due to local caches.

At the page level, the time spent reading a page, or *reading time*, can be measured as the inter-arrival time between the request for the page and a subsequent page request. In Figure 1, the reading time for Page A by Visitor 1 is the distance along the *x-axis (time)* between the request for Page A and Page B. This measurement is subject to a fair amount of noise as the visitor's behavior can not be always be accurately determined, e.g., the visitor could be grabbing a cup of coffee, talking on the phone, or actually reading the page. Another problem is determining when users leave the site since HTTP only sends "get me this page" messages and not "I'm leaving this page" messages. However, if a statistically valid sample size is determined and the data collected, reasonable statements about the reading times of pages can be made. [Catledge and Pitkow 1995] first reported a session time-out period of 25.5 minutes, which was 1 ½ standard deviations from the mean of 9.3 minutes between user interface events. A time-out period of 30 minutes has become the standard used by log file analysis programs, e.g., Interse [Interse 1996] and I/PRO

[I/PRO 1996]. This value should be determined individually for each site being measured. At the site level, temporal analysis can reveal other useful metrics. The total duration of the visit, or *session length*, measures the amount of attention spent by visitors to the site. The time between visits, or *inter-visit period*, helps determine the periodicity with which users revisit the site. There are, of course, other interesting analyzes that can be performed with temporal data, including survival analysis, self-similar processes, auto-regression, etc.

2.2 Path Analysis

The goal of path analysis is to understand the sequence of page views by visitors. Although the connectivity of pages in a Web site mathematically is best represented as a graph, it is not uncommon for designers and users to conceptualize the space hierarchically as a tree. In this representation, the establishment of parent-child relationships is usually done via a depth-first traversal of the pages. Once this has been done, it becomes possible to talk about the depth of the tree, the number of internal and leaf nodes, and the branching factor of the tree. With a tree representation and information about the paths people take through the site, the a*verage depth* visitors take can be determined, as well as the *average number of internal and leaf nodes* accessed. If one assumes that internal nodes typically serve to facilitate navigation and leaf nodes tend to represent content, this metric provide insight into how much time users are navigating verses ingesting content. If link typing were to ever become widespread on the Web, this metric could be determined more accurately.

Of equal interest to the paths people take is where and with what frequency they enter and leave the site. *Entry points* [Pirolli, Pitkow, Rao 1996] can be identified by looking for differences between the sum of all the incoming paths to a page and the total number of requests made for the page. Large differences indicate that visitors are not relying completely upon the local topology to access the page. Likewise, *exit points* can be identified by looking for the last element in the path sequence as well as clues in the frequency of path traversal verses page requests. Descriptive statistics can be generated for each of these metrics on a per page basis.

We know that certain users will visit a page and not continue traversing the hyperlinks contained in that page. Others, however, will proceed to traverse the presented links, thus continuing down a path. *Attrition*, introduced initially with respect to the WWW by [Pitkow and Kehoe 1995], can be understood as a measure of visitors who stop traversing verses the visitors who continue to traverse the hyperlinks from a given page. Attrition is typically calculated across a group of visitors, although it can be applied to individual visitors as well. *Attrition curves* are defined as the plot of attrition ratios for all pages along a certain path.

Other methods for analyzing hypertext paths include Markov Chain analysis [Guzdial 1994], Pathfinder Networks [Schvaneveldt 1990], and subsequence analysis [Catledge and Pitkow 1995][Tauscher 1996]. Path information can also be used to cluster users as well [Yan, et al. 1996].

While all these analyses provide insight into the behavior of visitors to a site and the usage of the site's resources, they are only as good as the data they analyze. While cookies provide an already consistently implemented approach to identifying users, this is not the case for gathering reliable page views, temporal, and path information. This inability to gather solid data is the cause for much concern by the measurement community [W3C 1996]. With the aim of rectifying this situation, several proposals have been made, with the nit-metering proposal being discussed in the next section in light to the types of analyses reviewed in this section.

3 Proposed Solutions

Several solutions exist that are specifically targeted towards increasing the reliability and amount of usage information known to servers. The two most notable are [Hallam-Baker 1996], which proposes complete forwarding of access data from proxy-caches, and [Mogul and Leach 1997], which proposes a limited form

of usage reporting by proxy-caches. Unfortunately, discussion and interest around [Hallam-Baker 1996] by the HTTP Working Group has not occurred in over half a year and as such, its RFC status has been withdrawn. This, plus space considerations, limit the discussion by this paper to [Mogul and Leach 1997], which is commonly referred to as the hit-metering proposal.

Recently, [Mogul and Leach 1997] have generated a rather elegant proposal for hit-metering. Their work is aimed at removing the practice of cache-busting by sites wishing to gain an accurate count of the usage of the site's resources. The draft calls for the implementation of a new HTTP header, called "Meter", that enables proxy-caches to report usage and referral information to originating servers. Additional extensions permit the originating server more control over the use of cached data by limiting the number of times a proxy-cache returns items before requesting a fresh copy. One of the argued side-effects this system if implemented, would be reduced network traffic as well as a reduction in server resources. The closest approximation of their work outside of the Web occurs in the field of advertising and the measurement of hard-copy publications. In that field, publishers try to determine the number of users who read a publication at least once. It does not try to measure the number of times users read the publication overall. Adhering to the same model, [Mogul and Leach 1997, page 5] specify a system whose "goal is a best-efforts approximation of the true number of uses and/or reuses, not a guaranteed exact count."

The proposal outlines a system where cooperating proxy-caches volunteer to keep totals of the number of uses for each page and the number of times each page is reused[2]. Cooperating proxy-caches can be arrange in an hierarchical manner rooted at the originating server, forming what the authors call a "metered subtree." Mechanisms are included in the draft which keep the counting and forwarding of page use information consistent throughout the subtree. When the originating sever encounters a proxy-cache that volunteers to meter usage, it can turn cache-busting efforts off and allow the proxy-caches to cache resources, i.e., actually allow the proxy-caches to do what they were intended to do in the first place. The tabulated metering information is then periodically forwarded to the originating server as determined by a set of rules that depend on the type of requests being issued by clients and upon deletion of the resource from the cache. The originating server is also able to specify limits on the number of times a resource is used by the proxy-cache before requesting a fresh page from the server. It is stated that this can be used to bound the amount of inaccuracy between reports.

This proposal has many strengths. First, it does not require significant changes to the existing HTPP 1.1 protocol [Fielding et al. 1997]. However, changes to the server and proxy software to implement the behavior behind the new Meter header are required. Second, it appears that it does no worse a job at reporting descriptive statistics of usage than the current practice of cache-busting, and given cooperating proxy-caches, could arguably reduce network traffic considerably, freeing up valuable resources. Third, distinctions are made and reported between unique page views and reuse page views by users behind proxy-caches, a distinction that cache-busting techniques are not able to do reliably without ancillary techniques that generate unique identifiers for each user, e.g., cookies. For these reasons, the proposal does appear to satisfy the goal of a best effort system.

However, the proposal suffers from several potentially critical weaknesses. First, the system depends upon cooperating proxy-caches. This cooperation is something that can not be forced to happen, and as a result, may not happen. Since cooperation can not be enforced, originating servers may still be tempted to implement cache-busting techniques for non-cooperative proxy-caches. Much seems to depend upon the level of trust WWW site owners have in the system. However, no mechanism exists within the proposal to determine the level of trust to place in the system. It therefore becomes difficult to determine the extent of resources that would be saved by this proposal and what level of cooperation would be required. Given this

---

[2] Couched in the terminology outlined above, "uses" are unique page views, and "reuses" are reuse page views.

uncertainty, one can not say whether it will scale well as the size and the complexity of the Internet increases.

Second, the hit-metering proposal does not enable the collection of temporal statistics or path statistics, which are collectable when cache-busting techniques and other proposals (see below section on sampling) are used. This poses a serious threat to sites wishing to do market research beyond merely the number of page views.

Third, the authors make claims about the ability of their hit-metering solution to address the problem of obtaining reasonably accurate counts of the number of *users* of a resource, not just the number of uses. They argue that the separation of unique page views from reuse page views affords the unique identification of users behind proxies. While they recognize that scenarios exist where their solution would over-report the number of users, they assert that they do not believe this to be a significant source of error. However, this assertion is not supported by any empirical data. Further research is needed before the validity of this assertion can be determined.

Finally, as the authors themselves acknowledge, no guarantees can be made that the reported usage will accurately reflect actual usage. While this is not a problem in and of itself, we do after all live in a world of approximations, their system does not enable the amount of error imposed by the system to be measured. This means though that the origin sever can not at the end of the reporting cycle determine if the usage reported is 100% correct, 75% correct, or 50% correct. As a result, the amount of error in the system could be any of the above, or all of the above, but on different days under different conditions. This is undesirable from a research and marketing perspective, since the claims can not be made about accuracy. Suppose a site notes an increase of 5% in traffic from one week to another, under this scheme, there is no way to say for sure that traffic actually increased by the stated amount—the variation could have occurred from any number of sources, all of which are impossible to determine using the hit-metering proposed solution.

Some of the variability in the system exists because the server can not control the time a resource will be deleted from the proxy-cache and hence when a report will be issued. The server can also not control when the usage limit on the resource will be achieved and hence, when that report will be issued. This causes problems in determining what data to include in each reporting cycle. To alleviate this, the server could compute the time remaining until the end of a reporting cycle for each item requested and set the "Cache-control: max-age = x" header to expire at the end of a reporting cycle. However, the proxy-cache is not required to deliver the report at the end of this period. Once could also set max-age to be small, so that the origin server receives many reports during a reporting period, but this begins to resemble cache-busting practices and defeats the proposal's alleged efficiency gains. There is no way specified in the proposal to guarantee that all usage data will be delivered by a specific time and hence what data to include in each report. Without being able to attribute reported usage correctly to each reporting period, results will not be comparable across reports, let along across sites.

Another source of variability in the proposal, though readily admitted by the authors, is the inability of the system to handle network failures and host crashes. Several real scenarios can occur. If a proxy-cache crashes, all usage data will be lost. For larger proxy-caches, the amount of usage data lost could be significant. To circumvent this, the originating server could set shorter usage limits on resources, though this reduces the alleged network gains. Likewise, if the physical network suffers a failure, the proxy-cache will be unable to deliver report data, possibly effecting a large number of servers. The origin servers have no way of knowing that a failure has occurred, or recovering the usage data. Additionally, if the origin server becomes unavailable, all attempts by proxy-caches to send reports will not succeed. No mechanism is specified for the behavior of a proxy-cache in this, or any of the above instances. So, did a significant decrease in traffic occur because of real usage of the site or because of proxy-cache failures, failures in the network, etc.?

These sources of variability seriously undermine the confidence one can place upon the numbers reported using the system proposed by [Mogul and Leach 1997], since results are not guaranteed to be either repeatable or reliable. While, these limitations do not violate their goal of a best efforts system, they essentially make the usage reported by the hit-metering proposal statistically unusable.

4 A Sampling Approach

Sampling is a form of inferential statistics that attempts to draw conclusions about a population based upon a subset of the population. This subset is called a *sample*. While the development of inferential statistics has occurred primarily in the 1900's, making is much more recent vintage than descriptive statistics, the field is back by mathematical theorems that have been proven to function properly and repeatedly across numerous disciplines [Hogg and Tannis 1993]. Sampling is commonly used in many areas, including the measurement of audience size and behavior for television (e.g., the Nielsen ratings), the gauging of political opinions and preferences, as well as measuring the population of the countries, (e.g., the United States Census Bureau will implement sampling procedures for the first time in the year 2000).

Part of the elegance of inferential statistics and sampling is that all members of the population need not be examined in order to accurately represent the entire population. One of the major requisites for sampling is that the elements of the sample are selected at random, where each element has an equal opportunity of being selected for the sample. If this is done, the mathematics behind sampling theory enable dependable estimates to be made of the characteristics of the entire population. The amount of error in the estimates can also be determined. The other major assumption of sampling theory is that the underlying distribution of events is a normal distribution. While the mathematics behind a normal distribution are a bit complex, a normal distribution can be legitimately assumed for large sample sizes, where large is commonly defined as a sample with over 35 members [Hogg and Tannis 1993]. The last thing to know about samples is that they can either be created on a continuous basis, where the sampling occurs consistently throughout the process of interest, or on a non-continuous basis, where a specific sampling period is defined and measurements made only within this sampling period.

With respect to the Web, inferential statistics prescribes sampling of usage data to understand the number of users as well as the usage of a site's resources. This approach to collecting usage information via a subset of the population is diametrically opposed to the approach taken by cache-busting and [Mogul and Leach 1997], which attempts to collect as much information about all users and usage as possible. With the increasing number of Web users, and the increasing number of pages made accessible, measuring each and every page request becomes quite expensive. For example, suppose you are one the more popular sites around, generating over say 20 million hits a day. Given that a typical entry in a log file contains around 100 bytes of data, which usually compresses at a ratio of 90%, the total amount of storage required per day is still 200 megabytes per day, or 75 gigabytes per year. While this is not a terribly large amount of data, it is roughly a third the size of all the current HTML content on the Internet [Internet Archives 1996]! Of course, there is also the time require to process that amount of data, which turns out to be a non-trivial., and typically dominates the cost equation as usage analysis. While the majority of sites on the WWW do not receive 20 million hits per day, the same principle of sampling applies—measure only as much as is necessary to gather reliable and projectable estimates of traffic at WWW sites. The proposed sampling solutions have not yet been implemented and as such are not to be taken as instructions for implementation. The value of the proposed solutions is to explicate possible approaches to gather reliable usage data in an efficient manner.

There are several ways to implement non-continuous sampling of usage on the WWW[3]. One of the most rudimentary methods, called *day sampling*, is to defeat caching only on certain days, treating all days equally. Standard cache-busting techniques like sending "Cache-control: proxy-revalidate", or "Expires: <past date> headers can be used on the desired days, with no cache-busting on the other days. The appropriate number of days to sample can be determined reliably using mathematically proven formulas. These formulas enable control over how much confidence one wants in the results. Thus, if the originating site can deal with a 10% margin of error in the reporting of its usage, it can sample fewer days than those sites wishing accuracy to within 5% margin of error (see [Hogg and Tannis 1993] or any statistical text for the exact formula).

A modification on this technique creates two groups (or strata) of users based upon the days they use the site, weekend or weekdays, and samples each group independently. The statistics gathered for each group can then be compared using a variety of standard statistical techniques, e.g., t-tests, and differences can be reliably identified if present. These comparisons could be as simple as examining the frequency of page requests or as complicated as examining the navigational behaviors of the weekday and weekend visitors to determine if the site would benefit for a lighter more playful feel on weekends.

This form of sampling has several strengths. Foremost, usage information can be gathered in a mathematically dependable manner while reducing the amount of network traffic and server resources consumed by cache-busting each and every day. The collected data is guaranteed to be as reliable within knowable limits to those achieved by consistent cache-busting. Unlike the hit-metering proposal which only collects descriptive statistics, day sampling results in both temporal and path statistical results, since the client is forced to send a revalidate message to the server. These forms of analyzes facilitate a deeper understanding of the dynamics of the site and its users. Additionally, comparisons between the non-cache-busting days and the days that implemented cache-busting can be performed to see to what degree the non-cache-busting days misreport usage. This difference can be examined over long periods of time and weights applied to the non-cache busting data to estimate usage in a dependable manner.

Still, this form of sampling is not without weaknesses. The valuable data gained by understanding how return visitors use a site can not be gathered since no information is kept about users across sampled days. Additionally, given that the usage of the resources at a site may be very time dependent, trustworthy usage information may not be available on specific days of interest, e.g., suppose you want to monitor the traffic on the day you release a product, but this day is not part of the sample. While one would still have the data as recorded by the access log, since a consistent weighting between cache-busting and non-cache-busting days may not exist, this information may be widely inaccurate and hence unacceptable to the origin site. Fortunately, several methods can be implemented which utilize continuous sampling. The most notable of these two are *IP sampling* and *user sampling*. Both of these techniques permit projectable estimates to be made for each day the server operates.

IP sampling uses the machine name that accompanies each HTTP request as defining the population to be sampled. For selected IP addresses, caching is defeated for all subsequent requests by that IP address, but only to for that IP address. Session time-out limits can be employed to handle the cases of when the IP address is that of a proxy server. A typical manner to select addresses at random passes each address through a one-way hashing function and divides the result by some number. The divisor and the resultant are used to determine the sampling probability. For example, if we desire to sample one out of every 1000 addresses, we use 1000 as the divisor and only defeat caching on the IP address whose resultant hash equals 1 (note that is could be any number between 0 and 999 so long as it is used consistently throughout the sampling).

---

[3] Many of these techniques have already been discussed by various members of the WWW community. However, to the best of my knowledge, none these proposals have not been formally explicated or published anywhere.

While this technique does enable continuous sampling it is not recommended since it is unclear how to generate a random sample of users due to the non-uniform nature of IP addresses. The distribution of users behind each IP address is not uniform because of 1) the presence of proxies which label all user requests from behind a firewall with the same IP address, 2) the class-based assignment of IP addresses which assigns domains to different address ranges, 3) the constantly changing ownership of domains, and 4) the constantly changing mapping of number of users behind each IP address and domain. Given these limitations, there is no obvious method to randomly sample the population via IP addresses that provides any confidence that all members of the users of the site had an equal opportunity to be selected for sampling. In the future though, if a method for randomly sampling domains can be found, reliable estimates could be made using IP sampling.

A more promising technique to obtain continuous sampling is to randomly sample the actual users. The most straight-forward implementation of this uses cookies to identify which users to sample. Rather than hash the IP address, this method hashes the cookies generated by the server to select users with a certain probability to include in the sample. This technique can be viewed as random cache-busting and therefore the implementation is rather straight forward. Once a user has been identified for sampling, caching is defeated for all subsequent requests during the sampling period. Since the sample can be determined by the hashing function and the generated cookies, a mixture between new visitors and return visitors is contained in the sample. Furthermore, by setting different "Cache-Expires" header times for different users, longitudinal tracking of user interests can be accomplished as well. However, since defeating caching can slow down browsing for visitors and hence negatively effect their behavior, it may be necessary to change the hashing algorithm periodically, place time limits on each user, or place limits on the amount of data gathered per user. This effect occurs as well with current cache-busting practices.

For sites which already issue cookies and use cache-busting, user sampling can greatly reduce the amount of network traffic and server resources consumed by the process. Since random sampling is done on a continuous basis, reliable estimates can be made about the global nature of sites usage on a per day basis—important or abnormal traffic days are not missed. Because the complete behavior of individual users is sampled, temporal and path analyses can be also performed. Furthermore, user sampling does not require any changes or extensions to HTTP 1.1. Just to make sure the main point is not missed though, the main advantage of user sampling is that reliable estimates can be made as to the total amount and nature of traffic at a Web site.

Cooperation is not necessary by any other entities other than client support of cookies and compliant proxy-caches. This compliance is no worse than that expected of current cache-busting practice. Non-cookie compliant browsers, which include browsers that do not have the ability to handle cookies correctly at the protocol level as well as users who choose not to accept issued cookies, can be addresses by double sampling. Double sampling is a method that measures the effect of non-response in a statistically valid manner by creating a random sample out of the non-responders and collecting data from them. To achieve double sampling, cache-busting can be turned on for a randomly selected group of non-cookie compliant browsers and statistically valid comparisons performed to see if differences exist between the usage patterns of cookie compliant browsers and non-cookie compliant browsers. Weights can be applied to the data if differences are found to exist, resulting once again in reliable estimates of site usage.

Non-compliant proxy-caches present a more difficult problem to which there is not immediate obvious solution. The problem of non-compliant proxy-caches plagues cache-busting, hit-metering, and sampling attempts equally. It is the author's hope that this problem can be handled via self-regulatory efforts by the Web community.

User sampling does not suffer system failures in the same manner as the hit-metering proposal. When a proxy-cache crashes, service is denied for all clients behind the firewall in the sample. Since the origin server is not relying upon the proxy-cache to send usage data, the system failure accurately reflects real usage, i.e., the clients in the sample did not continue to use the origin server resources. The same rationale

applies to network failures. When the origin server fails, sampling is not possible, but since no one is using the resource, it accurately reflects real usage.

5 Privacy Issues—What Users Think

A discussion of the collection of usage data would not be complete without incorporating privacy concerns. As is turns out, Web users place great value on the anonymous nature of the Internet and are equally protective of their privacy online. Only one in five uses feel that identifiers that can track a user at a site across sessions, i.e., cookies, ought to even exist [Pitkow and Kehoe 1996]. How then do these proposals fair in the face of such opposition?

Since cookies have already become the de-facto standard for sites interested in determining usage and the number of users, any proposal that further violates user privacy will most likely face serious opposition. Fortunately, the hit-metering and user sampling proposals both allow for more anonymity than full cache-busting, which collects a full set of longitudinal data for all users. For hit-metering, the time of the request and the machine name of the user are not reported to the origin server. However, use of the somewhat obscure "Vary" header in combination with the "User-Agent" and other headers may enable users to be uniquely identified behind proxy-caches, even if a user uses a non-compliant browser. This is no worse however than what sites can already gather by employing cache-busting.

For user sampling, since only a randomly chosen set of users is monitored with the tracking equivalence of cache-busting, most users will be able to use the site in an anonymous manner. If steps are taken to periodically change the hashing function that selects users, more users will be monitored, but fewer users will be monitored over long periods of time. One can even envision a system similar to the Neilsen Ratings where small amounts of money are provided in exchange for users agreeing to be sampled, though this introduces the risk of infusing self-selection biases into the sample. The trade-off between monitoring users for long periods of time and monitoring more users preserves *greater* privacy for *more* users than cache-busting. The comparison of sampling against hit-metering is arguable, since the it basically depends upon the value one places upon the number of users affected versus the amount of information gathered per user. Still, both provide more protection of user privacy than cache-busting.

6 Conclusion

This paper reviewed the current practices and limitation in the methods used to collected usage information on the WWW by first defining terminology. Since sever shortcomings exist in current practice, the latest proposal by the [Mogul and Leach 1997] was reviewed along with a new sampling based approach. Table 1 reviews the major dimensions along with the proposals were evaluated. Sampling techniques have the ability to gather data in a more reliable manner, permit more statistical analyzes, and perform accurately in the face of failures. Current research efforts are underway by the author that implement this form of user sampling to demonstrate their ability to reliably gather usage data in a lightweight manner.

Table 1: Comparison of usage collection proposals using cache-busting (CB) as the baseline.

Proposal	Descriptive Statistics	Temporal Statistics	Path Statistics	Protection of User Privacy	Necessary Modifications	Reliability of Gathered Data
Hit-Metering	full	some, worse than CB	none	better than CB	HTTP specs, proxy-cache software	uncertain, not worse than CB
User Sampling	full	full	full, except from non-compliant technologies	better than CB, arguably equal to hit-metering	server module	reliable, better than CB

7 Acknowledgments

This work was supported in part by an Office of Naval Research Grant N00014-96-C-0097 and in part by an Intel Foundation Graduate Fellowship.

8 References

[Catledge and Pitkow 1995] Catledge, L. and Pitkow, J. (1995) Characterizing Browsing Behaviors on the World Wide Web. *Computer Networks and ISDN Systems*, 27(6).

[Guzdial 1993] Guzdial, M. (1993). Characterizing Process Change Using Log File Data. Graphics, Visualization, and Usability Center Technical Report 93-41.

[Fielding et al 1997] Fielding, R, Gettys, J., Mogul, J. C., Frystyk, H., and Berners-Lee, T. (1997). RFC 2068—Hypertext Transfer Protocol—HTTP/1.1. UC Irvine. Digital Equipment Corporation, MIT.

[Hallam-Baker 1996] Hallam-Baker, P. (1996). W3C Working Draft for Proxy Caches. WD-proxy-960221. <URL:http://www.w3.org/pub/WWW/TR/WD-proxy.html>. Online resource.

[Internet Archives 1996] Internet Archives. (1996) <URL:http://www.archive.org>. Online resource.

[Interse 1996] Interse Corporation. (1996) <URL:http://www.interse.com>. Online resource.

[I/PRO 1996] Internet Profiles. (1996) <URL:http://www.ipro.com>. Online resource.

[Mogul and Leach 1997] Mogul, J. And Leach, P. J. (1997). Simple Hit-Metering for HTTP. Internet Draft draft-ietf-http-hit-metering-00.txt. HTTP Working Group. January 1997. This is a working draft.

[Novak and Hoffman 1996] Novak, T, and Hoffman, D. (1996) New Metrics for New Media: Toward the Development of Web Measurement Standards. Manuscript in progress.

[Pirolli, Pitkow, and Rao 1996] Pirolli, P., Pitkow, J., and Rao, R. (1996) Silk From a Sow's Ear: Extracting Usable Structure from the World Wide Web. *Conference on Human Factors in Computing Systems (CHI 96)*, Vancouver, British Columbia, Canada.

[PGP 1996] Pretty Good Privacy. (1996) <URL:http://www.pgp.com>. Online resource.

[Pitkow and Kehoe 1995] Pitkow, J. and Kehoe, C. (1995) Results from Third World Wide Web User Survey. *The World Wide Web Journal*, 1(1).

[Pitkow and Kehoe 1996] Pitkow, J. and Kehoe, C. (1996) GVU's Sixth WWW User Survey. <URL:http://www.ccc.gatech.edu/gvu/user_surveys/survey-10-1996>. Online resource.

[Schvaneveldt 1990] Schvaneveldt, R. (1999) Ed. Pathfinder Associative Networks: Studies in Knowledge Organization, Ablex Publishing Corporation, Norwood, NJ.

[Tauscher 1996] Tauscher, L. (1996). Evaluating History Mechanisms: An Empirical Study of Reuse and Patterns in World Wide Web Navigation. Master's Thesis, Department of Computer Science, University of Calgary.

[W3C 1996] World Wide Web Consortium Workshop on Internet Survey Methodology and Web Demographics. (1996). <URL:http://www.ai.mit.edu/projects/iiip/conferences/survey96/progcopy.html>. Online resource.

[Yan, et al. 1996] Yan, T. W., Jacobsen, M, Garcia-Molina, H., and Umeshwar, D. (1996) From User Access Patterns to Dynamic Hypertext Linking. *Computer Networks and ISDN Systems*, 28(11).