Int J Digit Libr (2008) 9:139–149
DOI 10.1007/s00799-008-0043-z

REGULAR PAPER

Workload analysis for scientific literature digital libraries

Huajing Li · Wang-Chien Lee · Anand Sivasubramaniam · C. Lee Giles

Published online: 23 September 2008

Abstract Workload studies of large-scale systems may help locating possible bottlenecks and improving performances. However, previous workload analysis for Web applications is typically focused on generic platforms, neglecting the unique characteristics exhibited in various domains of these applications. It is observed that different application domains have intrinsically heterogeneous characteristics, which have a direct impact on the system performance. In this study, we present an extensive analysis into the workload of scientific literature digital libraries, unveiling their temporal and user interest patterns. Logs of a computer science literature digital library, CiteSeer, are collected and analyzed. We intentionally remove service details specific to CiteSeer. We believe our analysis is applicable to other systems with similar characteristics. While many of our findings are consistent with previous Web analysis, we discover several unique characteristics of scientific literature digital library workload. Furthermore, we discuss how to utilize our findings to improve system performance.

H. Li (✉) · W.-C. Lee · A. Sivasubramaniam
Department of Computer Science and Engineering,
The Pennsylvania State University, University Park,
PA 16802, USA
e-mail: huali@cse.psu.edu

W.-C. Lee
e-mail: wlee@cse.psu.edu

A. Sivasubramaniam
e-mail: anand@cse.psu.edu

C. L. Giles
Department of Computer Science and Engineering,
College of Information Sciences and Technology,
The Pennsylvania State University,
University Park, PA 16802, USA
e-mail: giles@ist.psu.edu

Keywords Workload analysis · Prediction · Benchmark

1 Introduction

Compared with general-purpose search engines such as Google,[1] digital libraries are dedicated to specific domains. Typically, digital libraries are limited to some tightly correlated topics. Unlike traditional websites which host Web files, large-scale digital libraries provide search facility over datasets collected by autonomous data harvest processes (e.g., crawlers, metadata harvesters, etc.). The size of datasets hosted by a digital library is typically much larger than traditional websites. Indices are built to facilitate the search process. An index I in a large-scale text digital library can be viewed as a collection of *inverted lists*, each of which consists of postings of document identifiers corresponding to a term to be queried. Via such an index, a query Q is processed by looking up postings associated with terms specified in Q. Auxiliary information regarding a document such as term frequency and term positions are kept in the index to facilitate calculation of the ranking scores of documents. Top ranked documents are fetched and returned to users.

Facing the increasing challenges of scalability in data and user requests, it is important to investigate the patterns and characteristics of system workload. An in-depth understanding of workloads can benefit digital libraries in various ways: (a) workload study may reveal the temporal patterns and user interests in requests, which may in turn be exploited to predict future requests; (b) workload analysis may facilitate system diagnosis to identify potential service bottlenecks. A large number of works have been performed to study user behaviors in large-scale web systems, such as search

[1] http://www.google.com.

engines [12,14,18–20]. However, they are only focused on one or some partial aspects of the traffic. To the best of our knowledge, there is no comprehensive workload analysis for the domain of large-scale digital libraries.

In this paper, we focus on large-scale digital libraries for scientific uses. Archives of scholar publications in electronic form have been built to serve the scientific community. Besides dedicated document retrieval utilities, automatic citation indexing (ACI) [10] plays an important role in such systems. As a good example, CiteSeer[2] is a Web-based scientific literature digital library which focuses on the domain of computer and information science. In this study, we analyse CiteSeer logs to investigate request patterns received at the server-side. We take proper preparation to make this work generic to scientific literature digital libraries rather than specific to one particular system. We carefully distinguish robot traffic from user traffic and study them, respectively. Two categories of characteristics (i.e., temporal and user interests) that have a direct impact on system performance are analyzed. We continue to discuss the implication and significance of the workload analysis to improve performance of digital libraries. In summary, the primary contributions of this work are threefold:

- We characterize the workload of scientific literature digital library in a set of attributes, each of which has a direct impact on system performances.
- We study the temporal patterns of user think times and session arrival intervals. We find that the access intervals are independent and identically distributed (see Sect. 4 for details), and thus cannot be predicted well with time-series models.
- We study the patterns of user requests sent to digital libraries. The result suggests that user interest patterns can be well captured with Zipf-law. In addition, we study the temporal locality and correlation of user issued requests.

Based on our workload analysis, we discuss the implications and suggest directions in tuning and improving the performance of scientific literature digital libraries. Although our focus is on scientific literature digital libraries, the research methodology presented in this paper is appropriate for applying to other application domains.

The rest of the paper is organized as follows. Related research is reviewed in Sect. 2. Preliminary processing of CiteSeer logs is introduced in Sect. 3. We will present our efforts in analyzing temporal aspects and user interests of scientific literature digital library workload in Sects. 4 and 5, respectively, followed by a discussion of applications of the discovered characteristics of the workload in Sect. 6. Finally, we give our concluding remarks and future work plans in Sect. 7.

[2] http://citeseer.ist.psu.edu.

2 Related work

Workload analysis is used in many applications for system performance tuning and benchmarking. [8] shows that Web access can be modeled by heavy-tailed distributions. The study also suggests that Web traffic shows self-similarity. With the property of self-similarity in Web traces, the correlational structure in a time series of events remains unchanged regardless measure scales. This feature can be explained by facts such as self-similar property of underlying file system statistics and user behavior. Self-similarity has been found in I/O workloads as well [11]. Some previous works study user behaviors in search engines by analyzing logged data. As one of the first few papers discussing search engine user behavior, Hoelscher [12] analyzed query logs of Fireball, a popular German search engine, showing a large portion of users only browse the first page of returned results. Silverstein et al. [20] studied AltaVista search engine and found that the majority of the users submitted only one query before the session ended. Markatos [18] shows the existence of locality in the queries for Excite search engine. The results are later confirmed in [19]. A later work on AltaVista shows over 67% of the logged queries are requested only once, while other few queries are requested frequently [14]. Again, the Zipf distribution is observed. However, none of them gives a comprehensive picture of the workload.

There have been efforts in designing Web workload generators, which also give good insights into workload characteristics. SURGE [2] presents representative Web workloads for network and server performance evaluation. User equivalence (UE) is generated in SURGE to represent a population of a known number of users. In each UE, six factors of Web characteristics are statistically modeled, which include file sizes, request sizes, popularity, embedded references, temporal locality, and OFF times. The generated trace is compared with other synthetic workloads, showing that SURGE well models the actual Web traffic. More importantly, a SURGE-generated trace is found to have the self-similarity property of actual workload data. However, SURGE cannot be applied to generate user traffic for large-scale digital libraries due to the uniqueness of this application domain versus generic websites. Query processing inside a distributed search engine architecture is studied in [1], which focuses on issues such as performance bottleneck and index server load balancing. Other network workload generators, such as [21], emulate protocol-based traffic to benchmark the performance of Web servers. From an ISP's perspective, Bent et al. [3] analyzes characteristics of a server that hosts many websites.

A recent workload study emphasizes generating accurate I/O requests for TPC-H queries to evaluate the performance of disk subsystems with 22 queries of the TPC-H workload [26]. The analysis is performed at the disk block-level. The request arrival-pattern (inter-arrival times) and access-pattern (request location and requested data size) are synthesized. In the database community, Chaudhuri et al. [7] addresses the problem of identifying primitives which best summarize SQL workloads to databases. The interest in workload analysis continues to grow as more applications can benefit from workload study, such as adaptive system tuning [16,22], testing [25] and diagnosis [27]. Manavoglu et al. [17] shows that the models extracted from user workload can be applied in improving search engine services.

There also exist many Web benchmark projects where typical Web application scenarios are identified and the corresponding Web workload traffic is synthetically generated. Well-known examples include TPC-App[3] (an application server and Web service benchmark), TPC-H[4] (a benchmark for decision-support applications), TPC-W[5] (a transactional Web benchmark), and SPECweb2005[6] (the next-generation SPEC benchmark for evaluating the performance of Web servers). However, none of them aims at providing benchmark suites for large-scale digital libraries.

3 Data preparation

To characterize the workload of scientific literature digital libraries, it is imperative to look into the usage logs. Take CiteSeer as an example, a typical logging entry is listed below:

1114070813.127 event context 0 1782747 0 446930 ip: 128.255.54.*[7] agent: Generic

This entry consists of five attributes:

- *Time stamp* recording the arrival time of the request.
- *Request type* indicating the service to be invoked in order to answer the request.
- *Request specification* specifying the parameters of a request which is dependent on request types. For example, query terms are specified to find related documents (document query); document identifiers are specified for retrieving documents (document retrieval).
- *IP address* providing address information of clients.
- *Agent type* indicating the types of clients. We can infer from the agent type whether the request is from a user or a robot.

[3] http://www.tpc.org/tpc_app/default.asp.

[4] http://www.tpc.org/tpch/default.asp.

[5] http://www.tpc.org/tpcw/default.asp.

[6] http://www.spec.org/web2005/.

[7] We suppressed the last section of the IP address.

Three preliminary data preparation tasks are performed before analysis: (1) elimination of irrelevant requests; (2) sessionization; (3) robot detection.

3.1 Elimination of irrelevant requests

CiteSeer provides many services, many of which are either irrelevant to digital library functionalities (e.g., user feedback) or hardly used (e.g., online paper submission). Besides, we do not want to limit the applicability of our study to CiteSeer. These requests are eliminated from the log.

CiteSeer has more than 40 request types recorded in its logs. After comparing services provided in CiteSeer and Google Scholar, some unique services of CiteSeer are eliminated. Meanwhile, some similar service types are grouped into generic categories. For instance, in CiteSeer, after finding a document, a user can ask for documents from the same source, co-citation documents and text-based similar documents. These requests are all grouped into a *related* request type, suggesting that this service returns to users related documents with any metric.

A simplified log as a result consists of six unique request types: including (1) document query (query for documents), (2) citation query (query for citations), (3) document retrieval (download a document), (4) related (find related documents given a document), (5) external URLs (access other websites from CiteSeer), and (6) document specification (view document details).

3.2 Sessionization

To facilitate temporal analysis of the workload, the entire trace is decomposed into a series of sessions in order to model intra-session intervals and inter-session intervals. This sessionization process can be divided into two steps: (1) correct mapping of activities to different users; and (2) correct separation of activities belonging to different visits of the same user. Since the CiteSeer system does not record user identities and session information in its logs, we employee some heuristics in the process. Berendt et al. [5] suggests that page stay interval performs well for short but temporally dense sessions, which are common for digital libraries. Therefore, the following heuristics are adopted:

- The IP address is used as the identity of users. We regard every IP address as a single user, for which requests from the same IP address are taken as from the same user.

Although it is not definitely accurate, this is the best available approach.
- The log entries belonged to an IP address are grouped and recorded sequentially by their time stamps. If we find the interval of successive visits by a user exceeds a time threshold, the latter request is taken as the start of a new session. In our experiment, we set the time threshold to be 1,800 s (30 min).

3.3 Robot detection

It has been observed that the requests initialized by robots (crawlers, Web spiders, etc.) contribute considerably to the network traffic. Unlike users, robots start crawling from some given seed URLs and continue to probe the Internet via Web links. From their behaviors it can be inferred that robot access pattern is quite different from users. Based on our best knowledge, previous Web workload studies do not single out robots from the traffic and study their behaviors separately. Taking into consideration the significant Web traffic generated by robots, we study the behaviors of robots and users, respectively.

Web robot session studies indicate that by collecting Web access attributes, such as agent type, access average time and total request pages, most robot sessions can be accurately labeled [23]. Based on what are recorded in the logs, the following heuristics are adopted. First, we infer the type of a client from the agent type attribute in a log entry. Basically, we use a robot agent string list[8] to identify obvious robots. However, some robots do not declare themselves when they access CiteSeer, disguising as real users. To locate such unfriendly robots, we gather statistical information from sessions. If the average session length is extremely long (more than 100) or a portion (10%) of intra-session intervals are particularly small (less than 0.5 s), the IP address is labeled with "robots" because they show extremely odd behaviors different from normal users.

3.4 Statistical summary

Two-week-length CiteSeer logs are collected and analyzed in this paper. Table 1 gives the statistical summary of the usage logs. From the number of requests and the number of the unique requests, we can infer that many requests do not repeat frequently. Also, we can see that for most sessions, the session length is in a relatively small value. The detailed distribution of session length is shown in Fig. 1, from which we can observe that most sessions have a length of 2 or 3.

[8] http://www.pgts.com.au/pgtsj/pgtsj0208d.html.

Table 1 Statistical summary of the analyzed logs

Summary	Trace
Number of document query requests	601,337
Number of document retrieval requests	2,273,233
Number of distinct document query requests	334,731
Number of distinct document retrieval requests	393,416
Average session length	2.839

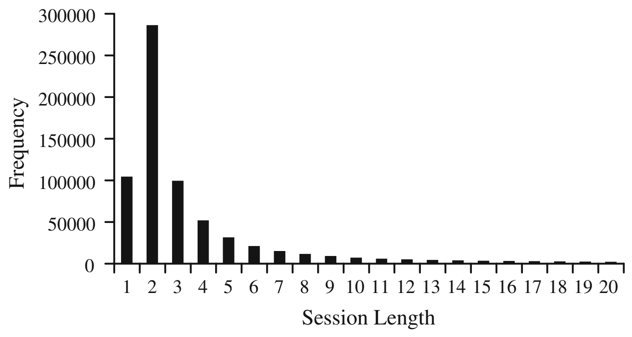


Fig. 1 Session length distribution

4 Temporal analysis

In this section, we present our analysis on temporal aspect of the workload. First, some key notions (i.e., self-similarity, short-range dependency, and long-range dependency) are introduced in Sect. 4.1. The methodology used to analyze the workload is presented in Sect. 4.2. Observations obtained from our temporal analysis are summarized in Sect. 4.3.

4.1 Background

4.1.1 Self-similarity

Self-similarity commonly exists in Web traffic [8]. It can be explained by assuming the existence of a heavy-tail distribution in workload attributes, including file sizes, transfer times and user think times. A random variable X follows a heavy-tailed distribution if

$$P[X > x] \sim x^{-\alpha}$$

as $x \to \infty, 0 < \alpha < 2$.

Regardless the behavior of a distribution for small values of the random variable, it is heavy-tailed if the asymptotic shape of the distribution is hyperbolic.

Intuitively, self-similarity means that the attributes considered are independent of the scale. Take time-series as an example, a process is called self-similar when its statistics are independent of the time scale. It can be expected that averaging over equal periods of time does not influence the statistical characteristics of the process. Formally, a time-series

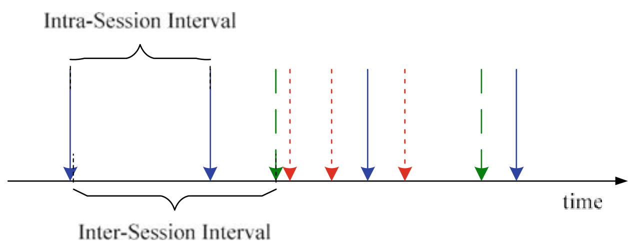


Fig. 2 Request stream received by the server

X_t $(t = 1, 2, ...)$ is said to be *exactly second-order self-similar* if

$$X_t \overset{d}{=} m^{-H} \sum_{i=m(t-1)+1}^{mt} X_i$$

for $1/2 < H < 1$ and all $m > 0$, where $\overset{d}{=}$ means an equivalent distribution. This suggests a methodology for testing self-similarity in a time-series. The series is separated into non-overlapping blocks with the same size m. All observations in each block are aggregated, by which a new m-aggregated time-series is constructed. If the new series is statistically identical with the original series, scaled by a factor of m^{-H}, the original time-series is *self-similar*. This method is adopted by self-similarity tests in Sect. 4.3.1.

4.1.2 Short- and long-range dependence

In a self-similar time-series where observation bursts can be found in a wide range of time-scales, the distribution exhibits long-range dependence (LRD) [8]. Long-range dependence exists when the current observation is highly correlated with observations far away in time. On the other hand, short-range dependence (SRD) [8] shows that the current observation is only correlated with recent observations. If we analyze the correlation functions of an SRD time-series, the correlation value decreases dramatically to a very small value, while the LRD series retains considerably significant correlation values for distant observations.

4.2 Methodology

The temporal behavior of requests can be decomposed. In system logs, all visits are collected and merged into a continuous record, which is a stream of the requests imposed on the server. However, keep in mind that this stream is comprised of many individual sessions, which provides a method to divide the trace into smaller units.

Figure 2 illustrates the composition of the request stream, where requests from different sessions are represented by different line patterns. Following this direction, two sets of time intervals are identified, as marked in the figure.

- *Intra-session interval* the time interval between two subsequent requests in one session.
- *Inter-session interval* the time interval between subsequent sessions.

The two streams are studied independently because the intra-session interval is a client side behavior, decided solely by an individual client, whereas the inter-session interval is a server side aggregate. When studying time intervals, the logged request entries are segregated and organized into appropriate sessions. We first study each time stream (intra- and inter-session interval) to test correlation assumptions such as the existence of independent identically distributed (IID) and self-similarity. We also study the impact of time over requests by looking into aggregated user request patterns.

4.3 Analysis results

4.3.1 Study of correlations

Have a good understanding of the correlations within a given time-series may help to predict future accesses and to determine query processing strategies. Given a time-series as the input, the autocorrelation function (ACF) is used to study the correlation within the stream. ACF measures the similarity between the series X_t and its shifted version X_{t+k}, where k is the lag. The formal definition of a sample autocorrelation function is given as:

$$\rho(k) = \frac{E[(X_t - \mu)(X_{t+k} - \mu)]}{\sigma^2}$$

where μ, σ are the sample mean and standard deviation, respectively. ACF values can be plotted with different lags. If the ACF decays hyperbolically to zero, then the process shows LRD, where distant observations have significant influences. On the contrary, if the ACF is large for small lags and decreases dramatically when the lag increases, SRD exists. Intuitively, if none of the ACF values for shifted series is significant, we can assume the time-series to be IID.

In our analysis, we treat robots and users separately. Figure 3 gives the ACF plots for inter-session streams. It is evident that the ACF values drop dramatically as the lag increases and remain close to 0 for large lags, which shows that the session intervals are independent.

To study intra-session streams, we randomly select representative sessions from robot and user categories. Here, we deliberately select those sessions that do not show abnormal behaviors. The corresponding ACFs are plotted in Fig. 4.

It can be seen from the ACF plots that most of the absolute correlation values are extremely low. Figure 4a shows periodical spikes, which reach a maximum value of 0.2, implying that some periodical pattern may exist for this robot. However, those spikes never reach 0.25, indicating the correlation

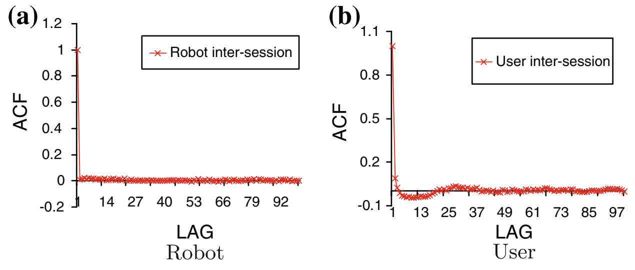


Fig. 3 ACF for inter-session intervals

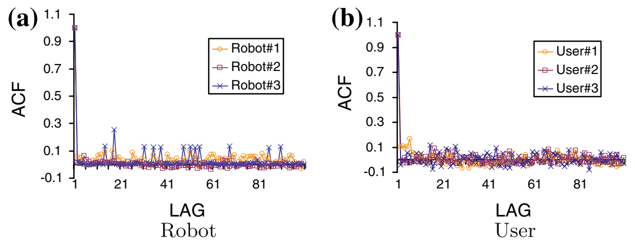


Fig. 4 ACF for intra-session intervals

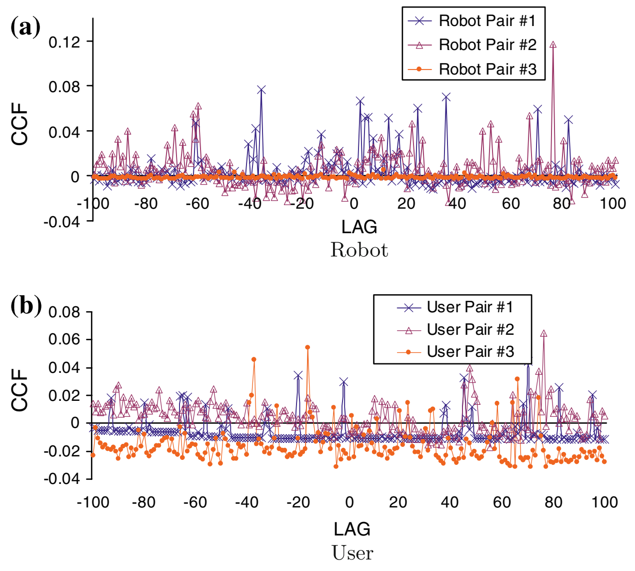


Fig. 5 CCF for intra-session interval pairs

is still weak. Hence, all the results indicate that no significant LRD or SRD exists and thus IID seems to be a better fit.

It is also useful to study the correlation among multiple streams. Here we need to know if the correlation exists between the intra-session intervals of different sessions. The cross-correlation function (CCF) computes the similarity between two streams given a lag k. Our analysis results are shown in Fig. 5, in which multiple pairs of intra-session intervals are compared.

Again we study the behavior of robots and users independently. In each category, we randomly select three pairs of intra-session intervals as input streams. The results in Fig. 5 indicate no strong correlation existing between different sessions. Most of the CCF values are smaller than 0.1, with only a few spikes reaching 0.12. Although only a small portion of sessions are compared in this experiment, considering the diverse behaviors of clients and the fairness of our session selection policy, we conclude that for most cases the sessions are not heavily correlated.

Table 2 Self-similarity test results

Stream	Variance-time	R/S	Periodogram	Whittle
Inter(R)	0.689	0.636	0.547	0.521
Inter(H)	0.721	0.559	0.257	0.539
Intra(R1)	0.553	0.507	0.482	0.500
Intra(R2)	0.768	0.520	0.499	0.553
Intra(R3)	0.706	0.591	0.555	0.532
Intra(R4)	0.403	0.571	0.575	0.522
Intra(R5)	0.410	0.534	0.812	0.599
Intra(H1)	0.499	0.653	0.736	0.592
Intra(H2)	0.499	0.556	0.563	0.500
Intra(H3)	0.520	0.585	0.706	0.500
Intra(H4)	0.633	0.507	0.431	0.504
Intra(H5)	0.455	0.554	0.443	0.500

Although the ACF results strongly suggest the existence of IID, we can use self-similarity tests to reinforce our beliefs in this conclusion. Four methods are recommended to test self-similarity: variance-time plot, R/S plot, periodogram method and Whittle estimator [4,8]. In all of these tests, the degree of self-similarity is expressed with a single parameter H, the Hurst parameter. For self-similar series with LRD, $1/2 < H < 1$. As the H value goes up, the degree of both self-similarity and LRD increases.

We apply all the four tests on the streams which are extracted and studied in the previous ACF tests, with H values listed in Table 2.

In Table 2, the first column marks the streams under study, in which character "R" represents robot streams and "H" stands for user streams. Inter-session intervals are denoted as "Inter" and intra-session intervals are denoted as "Intra". This naming schema is followed in all the following tables.

In addition to the H values listed in Table 2, the variance-time plot, R/S plot and periodogram method all return a correlation coefficient to estimate the reliability of the test. The Whittle estimator can produce confidence intervals, but its values are not shown here due to space limit. As we look into the values in Table 2, we find that most values are around 0.5, suggesting that the degree of self-similarity is not significant. Although some tests return a higher H value, its corresponding correlation coefficient indicates that the measure is not very reliable. For example, the value periodogram method only has a 39.05% correlation coefficient for intra-session intervals of robot5.

Table 3 Prediction errors using RPS toolkit's time-series models

Stream	Sample mean	AR(16)	AR(4)	BM(16)	MA(16)	MEAN	LAST	ARMA(16,16)	ARIMA(16,2,16)	ARFIMA(8,0.5,8)
Inter(R)	25	35	33	33	35	33	63	4,925	3,614	3.5e+07
Inter(H)	1.8	1.9	2.0	1.6	1.8	1.9	2.5	1.9	1.8	4.9e+06
Intra(R1)	5.3	5.4	3.8	6.7	5.4	3.7	5.4	62	5.5	6,342
Intra(R2)	6.3	7.6	6.9	5.3	7.5	6.9	7.5	603	6.2	8.8e+06
Intra(R3)	1.6	1.0	1.1	3.1	1.0	1.4	5.5	1,340	4.5	825,840
Intra(R4)	27	17	16	18	17	16	21	233,892	4,075	1.7e+08
Intra(R5)	24	18	18	14	18	18	16	199,126	19,000	2.3e+08
Intra(H1)	41	95	93	122	116	80	128	193	6,598	1.1e+10
Intra(H2)	42	23	20	22	39	21	13	507,672	3,437	8.3e+08
Intra(H3)	8.4	78	78	79	77	79	91	7,181	87	2.3e+09
Intra(H4)	5.7	10	9.9	11	10	10	21	112,888	6,059	2.5e+09
Intra(H5)	12	13	13	11	13	13	13	1,274	12	1.2e+12

In summary, the results of ACF and self-similarity tests reveal that both LRD (self-similarity) and SRD do not exist in our target time streams. Thus, we assume them to be IID.

4.3.2 *Prediction using time-series models*

To validate our assumptions made in Sect. 4.3.1, we use RPS toolkit [9] to fit various time-series models to each time interval stream in our study. Nine models are implemented in the RPS prediction library, which include three categories. MEAN (long-range mean), LAST (last-value), and BM(p) (mean over "best" window) are widely applied simple models. AR(p) (auto regressive), MA(p) (moving average), ARMA(p, q), and ARIMA(p, d, q) are examples of the Box-Jenkins linear time-series models. The final model, ARFIMA(p, d, q) is a good choice to capture LRD feature and thus can be used to predict self-similarity streams. Detailed introduction of these models can be found in [6,24].

In this experiment, our time-series intervals are segmented into identical-length blocks, which are used as the training sets for RPS toolkit. Based on the training sets, parameters of each model can be determined. The produced models are used to predict a number of future observations. The predicted values are afterward compared with real observations recorded in the stream to find the absolute errors. We take different sections from a stream and perform each test for multiple times to reduce bias in the results. For inter-session intervals, each time a size of 2,000 continuous observations are selected to predict the next 100 values. Because the size of intra-session interval stream is generally small, each time we use 1,000 observations in one intra-session stream to predict the next 10. Our experimental results are given in Table 3, in which the values written in the parentheses of the heading line are the parameters used in models.

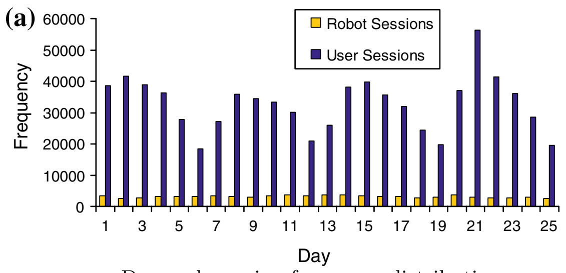
In Table 3, the second column records the mean values for all observations in the stream, which are used for comparison with absolute errors to reveal their degrees of significance. Obviously, ARMA, ARIMA and ARFIMA models are very inaccurate in data prediction, which again strongly rules out the existence of self-similarity and LRD. Other models generally produce less errors, which are still comparable with the mean value of the observations. Hence, we conclude from our study that time-series models are not a suitable choice for digital library workloads.

4.3.3 *Effects of time and date*

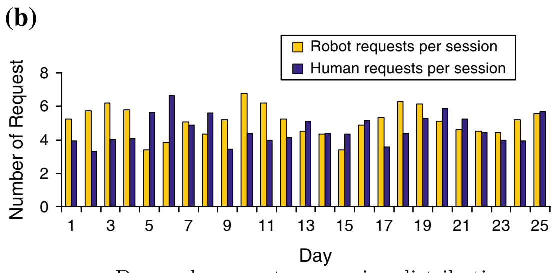
Although the test results in Sect. 4.3.1 show no strong correlation in the streams we analyzed, the exact *date* and *time* do have an effect on the request intervals. It is detected that the traffic of Mondays is much heavier than Sundays; the traffic in mornings is much heavier than midnights. These impressions indicate that a time-dependent pattern exists which remains unseen in previous tests. The pattern will be revealed when requests are aggregated with larger scales (daily, hourly).

We summarize request access frequencies in day-scale buckets and plot them in Fig. 6. It is obvious from Fig. 6a that periodic patterns exist for user inter-session intervals. Meanwhile, from Fig. 6b, the average number of requests within a session does not change too much over time, especially for users. It is suggested that a client's behavior is basically not influenced by time factor. Thus, it is inferred that what is affected by the time factor is the inter-session interval stream, in other words, the number of sessions at a time.

Figure 7 plots the ACF values for hour-scale session frequency distributions. Unlike the ACF plots shown previously, from the plotted curve we can find obvious correlation in



Day-scale session frequency distribution.



Day-scale request per session distribution.

Fig. 6 Effects of time study

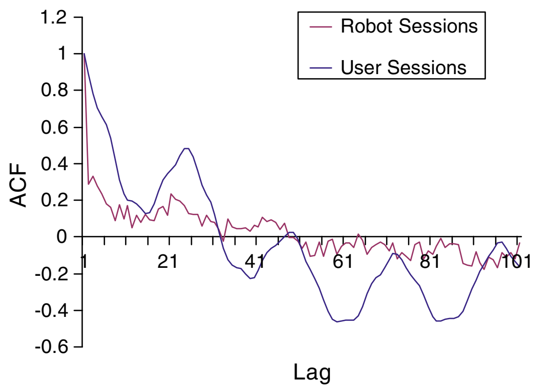


Fig. 7 ACF plot for hour-scale frequency distribution

the frequencies, which statistically confirms the existence of periodical frequency patterns.

5 Analysis of user interests

User interests in terms of searched documents, specified as request types and their parameters, exist in the workload trace (i.e., query logs) and thus has an impact on performance of services that process those requests. The actual request parameters and their meanings corresponding to various request types are different. Among services supported by CiteSeer, document query and citation query take the queried terms as their request parameter; document retrieval, related, and document specification use the targeted document identifiers as their request parameters; and external URLs service uses the specified URL as the parameter (which is not forwarded to the digital library and thus is neglected). As a result, the request types are grouped into the following two categories:

- *Query-centric*, where query terms are specified in the requests.
- *Document-centric*, where document identifiers are specified in the requests.

An analysis on the system workflow of digital libraries drives us to believe that it is important to analyze the requests in terms of user interests (i.e., represented by query terms and document identifiers). For example, the sizes of inverted lists for different query terms differ dramatically, which as a result has an impact on the response time of the system. On the other hand, each stored document has its own data volume and physical location, which may also affect the request latency significantly.

5.1 Frequency distribution

Previous studies [13] suggest that Zipf distributions are common for Web accessed items. A specific Zipf distribution follows the form of $f_i = K/i^{\alpha}$, in which f_i stands for the frequency of the ith popular observation. As we look into the logs, we find that there exist a set of *hot* query terms and documents in the requests to the system, i.e., requests for these terms and documents appear very frequently. Correspondingly, an unpopular request only contributes a small portion of the entire traffic. We rank terms and documents according to their request frequencies and plot the distributions in Fig. 8, where Y-axis shows the log-scaled request frequency, while X-axis shows the log-scaled ranking of the request, sorted by their frequencies. The distributions shown in Fig. 8 are close to straight lines, suggesting the existence of a Zipf distribution.

Maximum likelihood estimation (MLE) is applied to the frequencies to generate parameters for Zipf distributions. The parameters of the observed Zipf distributions are given in Table 4, from which it is suggested that users tend to provide more focused requests while robots issue diverse ones.

5.2 Request locality

Previous studies indicate that there is a high shareness in user requests [14, 18, 19]. In many cases, it is also valuable to investigate the locality of requests. For example, when it comes to determine the possible effectiveness of a system cache, request shareness and locality are both necessary.

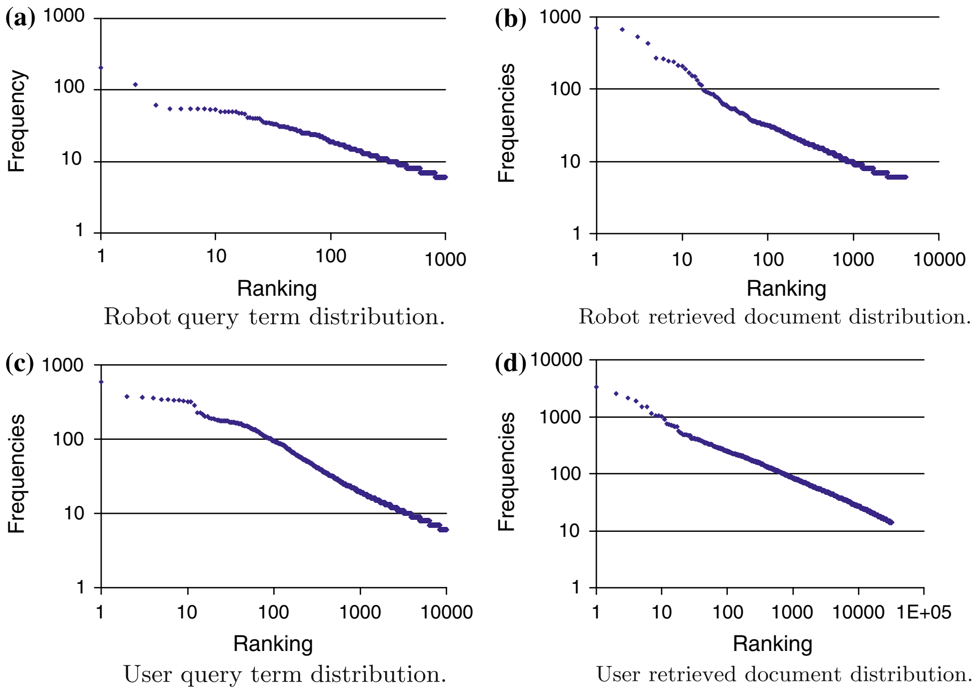


Fig. 8 Log–log scale request frequency distribution

Table 4 Zipf distribution parameters for requested item frequencies

Stream	α	K
Robot queried terms	0.473	157.40
Robot document identifiers	0.658	941.89
User queried terms	0.603	1288.25
User document identifiers	0.538	3451.44

In our analysis here, we aim at investigating the *distance* between subsequent resubmission of the same requests, where distance represents the number of other requests appearing in the interval of a request resubmission. Figure 9a, b show the distance distribution (in log–log scale) for document query and retrieval, respectively. From the plots we can see that most repeated requests have a relatively small distance. In other words, the same requests are often reissued within short intervals.

We also study user behaviors in browsing paged query results. Previous Web studies [12] reveal that users are reluctant to browse more pages than the first one. We conduct an analysis to figure out the probability for a user to view the subsequent pages. The distribution of requested page number is plotted in Fig. 9c, from which we can see most users are only interested in the top 3 pages, which contributes to 75.33% of browse requests.

5.3 Correlation study

If the system can capture a user's interest on-the-fly from previous requests and retrieve relevant documents in advance, the user may be able to retrieve the documents promptly (as he clicks on the document link). Correlation studies can assist in predicting the relevant documents in the repository to a query. For example, if we find users have a high probability to view or download d_i after querying Q, d_i probably is a good retrieval candidate for the user. We study the correlation between issued queries and the documents that are browsed afterwards. We summarize the correlation analysis of all 334, 731 distinct logged query strings in Table 5.

As we look into the queries that have high correlation with documents, we find most queries (90.72% for 10% correlational probability or higher, for instance) are unpopular, i.e., their cumulative query frequency do not exceed 5 in the logged time interval.

Correspondingly, we also analyze the correlation between document retrieval requests, which would reveal the relationship between documents. The same procedure is applied to all 393, 416 distinct document retrieval requests. The results are given in Table 6.

In-depth content analysis of involved papers shows that many highly correlated papers have some inherent relationships, including the same author papers, cited–citing pairs, and semantically similar papers. Due to the space limit, detailed analysis is not included.

6 Discussion

In this section, we briefly summarize and discuss the potential uses of the analysis results found in previous sections.

A direct application of the workload analysis result is to develop synthetic workload generators for simulated traffic of

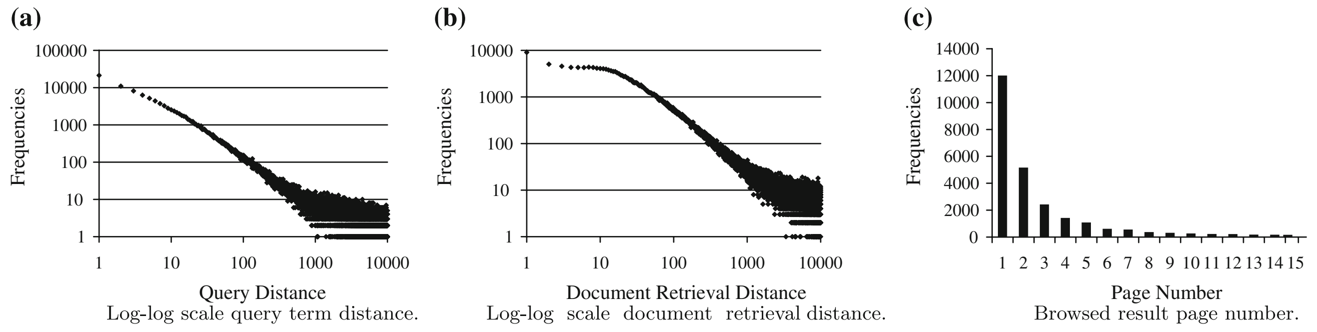


Fig. 9 Request locality distributions

scientific literature digital libraries. As we stated, we deliberately exclude logged attributes specific to CiteSeer in order to make our study applicable to general scientific literature digital libraries. Our findings, in terms of temporal correlation between users and the effects of time, can be used in such a workload generator to simulate actual workload for testing and benchmarking [15]. Our request frequency analysis shows that a small portion of popular requests contribute a significant proportion of the workload, whereas a large number of unpopular requests also form a long tail which is still significant. Combined with the finding on request locality, we can assess that system cache can be very effective for such a workload. However, special attention should be paid to the organization and management of large-scale digital library caches to prioritize popular requests. The study regarding to correlation between requests indicates that a digital library may be able to predict future user requests under certain circumstances, when a prefetching mechanism may be beneficial to the system.

7 Conclusion

Workload analysis for scientific literature digital libraries is challenging yet important. Previous works appeared in the literature are mainly focused on traditional Web traffic. In this paper, we study log traces of CiteSeer, aiming at analyzing both temporal characteristics and user interests of scientific literature digital libraries. Our analysis reveals that there is little temporal correlation within and between user sessions. In term of user interests, it is observed that user requests distribute unevenly with a small group of popular requests (on certain query terms and documents) and a long tail of unpopular requests. In addition, we observe high locality and correlation in user interests. These patterns can be utilized by the system to improve performance.

We plan to further investigate fine-grained semantics of requests. We are interested in using our workload analysis to improve performance and quality of scientific literature digital libraries by implementing workload-based traffic predication and server tuning.

Table 5 Correlation for query–document (Q–D) pairs

Correlation (%)	Q–D pairs	Unpopular Q percentage
1	221,981	79.13
5	41,718	86.61
10	18,903	90.72
50	2,381	95.38

Table 6 Correlation for document–document (D_1–D_2) pairs

Correlation (%)	D_1–D_2 pairs	Unpopular D_1 percentage
1	85,695	81.68
5	6,875	84.41
10	2,634	86.82
50	298	93.10

Acknowledgements This research is supported in part by National Science Foundation (under Grant no. IIS-0328881 and IIS-0534343) and Microsoft Research.

References

1. Badue, C.S., Barbosa, R., Golgher, P., Ribeiro-Neto, B., Ziviani N: Distributed processing of conjunctive queries. In: HDIR ’05, SIGIR (2005)
2. Barford, P., Crovella, M.E.: Generating representative Web workloads for network and server performance evaluation. In: SIGMETRICS ’98, pp. 151–160. July 1998
3. Bent, L., Rabinovich, M., Voelker, G.M., Xiao, Z.: Characterization of a large web site population with implications for content delivery. In: WWW ’04, pp. 522–533 (2004)
4. Beran, J.: Statistics for Long-Memory Processes. Chapman & Hall, New York (1994)
5. Berendt, B., Mobasher, B., Spiliopoulou, M., Wiltshire, J.: Measuring the accuracy of sessionizers for web usage analysis. In: Proceedings of the Web Mining Workshop at the 1st SIAM International Conference on Data Mining. Chicago, April 2001
6. Box, G., Jenkins, G.: Time Series Analysis, Forecasting and Control. Holden-Day Inc., San Francisco (1990)
7. Chaudhuri, S., Ganesan, P., Narasayya, V.R.: Primitives for workload summarization and implications for SQL. In: VLDB, pp. 730–741 (2003)

8. Crovella, M.E., Bestavros, A.: Self-similarity in World Wide Web traffic: Evidence and possible causes. IEEE/ACM Trans. Network **5**(6), 835–846 (1997)
9. Dinda, P., O'Hallaron, D.: An extensible toolkit for resource prediction in distributed systems (1999)
10. Giles, C.L., Bollacker, K., Lawrence, S.: CiteSeer: An automatic citation indexing system. In: Witten, I., Akscyn, R., Shipman, F.M. III (eds.) Digital Libraries 98—The Third ACM Conference on Digital Libraries, pp. 89–98. ACM Press, Pittsburgh, 23–26 June 1998
11. Gómez, M.E., Santonja, V.: Analysis of self-similarity in I/O workload using structural modeling. In: MASCOTS, p. 234 (1999)
12. Hölscher, C.: How internet experts search for information on the web. In: WebNet (1998)
13. Kelly, T., Mogul, J.: Aliasing on the world wide web: prevalence and performance implications. In: WWW'02, Honolulu, Hawaii, May 2002
14. Lempel, R., Moran, S.: Predictive caching and prefetching of query results in search engines. In: WWW, pp. 19–28 (2003)
15. Li, H., Lee, W.-C., Sivasubramaniam, A., Giles, L.: Searchgen: a synthetic workload generator for scientific literature digital libraries and search engines. In: JCDL '07: Proceedings of the 2007 Conference on Digital Libraries, pp. 137–146. ACM Press, New York (2007)
16. Lu, Y., Abdelzaher, T., Lu, C., Tao, G.: An adaptive control framework for QoS guarantees and its application to differentiated caching services (2002)
17. Manavoglu, E., Pavlov, D., Giles, C.L.: Probabilistic user behavior models. In: ICDM '03, p. 203. Washington (2003)
18. Markatos, E.P.: On caching search engine query results. Comput Commun **24**(2), 137–143 (2001)
19. Saraiva, P.C., de Moura, E.S., Fonseca, R.C., W.M., Jr., B.A., Ribeiro-Neto, Ziviani, N.: Rank-preserving two-level caching for scalable search engines. In: SIGIR, pp. 51–58 (2001)
20. Silverstein, C., Henzinger, M.R., Marais, H., Moricz, M.: Analysis of a very large web search engine query log. SIGIR Forum **33**(1), 6–12 (1999)
21. Simmonds, R., Williamson, C.L., Bradford, R., Arlitt, M.F., Unger, B.: Web server benchmarking using parallel WAN emulation. In: SIGMETRICS'02, pp. 286–287 (2002)
22. Streit, A.: Self-Tuning Job Scheduling Strategies for the Resource Management of HPC Systems and Computational Grids. PhD thesis, Faculty of Computer Science, Electrical Engineering and Mathematics, University Paderborn (2003)
23. Tan, P., Kumar, V.: Discovery of web robot sessions based on their navigational patterns. Data Min Knowl Discov **6**, 9–35 (2002)
24. Tran, N. Reed, D.A.: ARIMA time series modeling and forecasting for adaptive I/O prefetching. In: Proceedings of the 15th International Conference on Supercomputing, pp. 473–485, June 2001
25. Wang, Y., Rutherford, M.J., Carzaniga, A., Wolf, A.L. Weevil.: a tool to automate experimentation with distributed systems. Technical Report CU-CS-980-04, Department of Computer Science, University of Colorado, October 2004
26. Zhang, J., Sivasubramaniam, A., Franke, H., Gautam N., Zhang Y., Nagar, S.: Synthesizing representative I/O workloads for TPC-H. In: HPCA, pp. 142–151 (2004)
27. Zhang, S., Cohen, I., Goldszmidt, M., Symons, J., Fox, A.: Ensembles of models for automated diagnosis of system performance problems. In: DSN '05: Proceedings of the 2005 International Conference on Dependable Systems and Networks (DSN'05), pp. 644–653 (2005)