Information Processing and Management 50 (2014) 41–53

Contents lists available at ScienceDirect

Information Processing and Management

journal homepage: www.elsevier.com/locate/infoproman

An empirical investigation of Web session workloads: Can self-similarity be explained by deterministic chaos?

Scott Dick *, Omolbanin Yazdanbaksh, Xiuli Tang, Toan Huynh, James Miller

Department of Electrical & Computer Engineering, University of Alberta, Edmonton, AB, Canada

ARTICLE INFO

Article history:
Received 13 June 2011
Received in revised form 5 April 2013
Accepted 12 July 2013
Available online 14 August 2013

Keywords:
Web traffic
Session workload
Traffic modeling
Chaos theory
Nonlinear time series analysis

ABSTRACT

Several studies of Web server workloads have hypothesized that these workloads are self-similar. The explanation commonly advanced for this phenomenon is that the distribution of Web server requests may be heavy-tailed. However, there is another possible explanation: self-similarity can also arise from deterministic, *chaotic* processes. To our knowledge, this possibility has not previously been investigated, and so existing studies on Web workloads lack an adequate comparison against this alternative. We conduct an empirical study of workloads from two different Web sites: one public university, and one private company, using the largest datasets that have been described in the literature. Our study employs methods from nonlinear time series analysis to search for chaotic behavior in the web logs of these two sites. While we do find that the deterministic components (i.e. the well-known "weekend effect") are significant components in these time series, we do not find evidence of chaotic behavior. Predictive modeling experiments contrasting heavy-tailed with deterministic models showed that both approaches were equally effective in modeling our datasets.

1. Introduction

By one estimate, over two *billion* people now use the Internet (Staff, 2010). This user population now demands rich-media content in their Web experience. They want to play massively-multiplayer online games, receive IPTV, and store their data in the "cloud" – and all of this consumes huge amounts of bandwidth. Estimates of Internet traffic growth continue to show large year-over-year compound growth, reaching 1300 exabytes by 2016. Web traffic continues to be a large segment of this traffic, accounting for 5.4 exabytes of traffic per month in 2012 and is expected to rise to 17.5 exabytes per month in 2016, a compound annual growth rate of 35% (*peak* traffic is growing even faster) (Staff, 2012). Moreover, the number of web *users* worldwide is also increasing rapidly, rising from 2.03 billion in 2010 to 2.80 billion by 2015 (Staff, 2010). These massive traffic volumes mean that performance testing of network systems using realistic data is essential, and yet infeasible; it is simply not possible to indefinitely store a sample of network data covering an operationally significant period of time. Instead, modern performance testing relies on traffic generators to produce synthetic network traffic with the same characteristics as real-world data (Botta et al., 2012). This in turn requires accurate characterization of that traffic. Since Leland et al.'s landmark paper (Leland et al., 1994), network traffic in general has been known to be self-similar; informally, this means that observations of traffic intensity will tend to be identically distributed irrespective of the time scale of these observations.

* Corresponding author. Address: Department of Electrical & Computer Engineering, University of Alberta, 2nd Flr, ECERF Bldg., Edmonton, AB T6G 2V4, Canada. Tel.: +1 (780) 492 6216; fax: +1 (780) 492 1811.

E-mail addresses: dick@ece.ualberta.ca (S. Dick), xiuli@ualberta.ca (X. Tang), huynh@ece.ualberta.ca (T. Huynh), jm@ece.ualberta.ca (J. Miller).

0306-4573/$ - see front matter © 2013 Elsevier Ltd. All rights reserved.
http://dx.doi.org/10.1016/j.ipm.2013.07.004

More recent findings also extend this result to Web sessions (a measure that captures user behavior), which have also been found to be self-similar (Arlitt & Jin, 2000; Arlitt & Williamson, 1997).

Leland et al.'s classic explanation for self-similarity in network traffic rests on two points: first, network traffic had been found to be self-similar in earlier research (Meier-Hellstern et al., 1991). Second, Leland proposes that Ethernet traffic can be viewed as the aggregation of many renewal–reward processes (Mandelbrot, 1969; Taqqu & Levy, 1986) whose inter-renewal times exhibit infinite variance (i.e. follow a heavy-tailed distribution). An aggregate of many of these processes is known to be self-similar. This explanation is also posited by Arlitt to account for self-similarity in Web session lengths (Arlitt & Jin, 2000). However, this account neglects a possible alternative explanation for self-similarity: *deterministic chaos*. It is well-known that Web traffic (indeed all network traffic) has periodic components that are driven by the time of day, the day of the week, etc. – a phenomenon popularly referred to as the "weekend effect." These deterministic dynamics also operate at a range of time scales (albeit tending towards longer scales), and could potentially play a significant or even dominant role in causing Web sessions to appear self-similar. Chaotic maps have been previously proposed as an alternative explanation for self-similarity in network traffic (see for instance Erramilli et al., 1995; Mondragon et al., 2001), but have not been examined for Web sessions.

Web session generators are used in much the same way as network traffic generators; that is, they provide a data source for load testing of Web applications. The risk in leaving the possibility of nonlinear determinism unexplored is that simulated session data may in fact diverge from reality much more than is currently expected. Given the traffic and user-population growth described above, it seems plain that any widely-used Web application will face an increasingly challenging load profile as time progresses. Thus, load-testing of Web applications will only grow more stringent with time. However, as the frequency of Web sessions in a test increases, any discrepancies between the Web session generator and real-world behavior will be magnified. If self-similarity in Web sessions arises from deterministic components, then the traffic models used in load testing are ultimately misleading. Thus, until and unless the alternative of deterministic chaos is investigated, the model for Web session self-similarity proposed in Arlitt and Williamson (1997) and employed in Goseva-Popstojanova et al. (2006a) – and thus all session–generation systems based on them – will remain debatable. Such models are heavily used during the development of web services to analyze the performance and capacity of various server configurations and software alternatives (push vs. pull services). These models become even more critical when we consider the next generation of cloud-based infrastructure and services based upon service-orchestrated architecture (SOA) models.

Our goal in this article is to provide a rigorous empirical investigation of the possibility of deterministic chaos in Web session durations. Using the methods of nonlinear time series analysis (Kantz & Schreiber, 2003), we will investigate web server logs from two organizations (one public university, one private company) that cover much longer periods of time than earlier studies of self-similarity in Web sessions. We will first test the supposition that deterministic, rather than stochastic, processes (a necessary precondition to being chaotic in nature) best explain the datasets. We will then attempt to extract a chaotic invariant from these datasets, and thus evaluate this alternative explanation of Web session self-similarity. Finally, we will compare two forecasting algorithms (one deterministic, one stochastic) on these datasets, to determine which approach more accurately models network traffic.

The remainder of this article is organized as follows. In Section 2, we provide essential background. In Section 3, we describe the methodology for our data collection and analysis, and we attempt to extract chaotic invariants in Section 4. We describe our predictive modeling experiments in Section 5, and close with a summary and discussion of future work in Section 6.

2. Background and related work

In the classic study by Leland et al. (1994), the number of Ethernet packets per time unit on a busy network was plotted for time periods ranging from 100 s to 0.01 s; the resulting time series seemed to have identical characteristics at all length scales, i.e. the traffic was self-similar. Multiple subsequent studies have confirmed this result. This fact has several implications for network engineers. Firstly, source models for network traffic can be very simply constructed using heavy-tailed distributions; indeed, Leland et al. proposed that the reason why network traffic was self-similar was that file sizes (and hence transfer sizes) on the network followed a heavy-tailed distribution. Secondly, common measures of the "burstiness" of network traffic become heavily dependent on the time scale of the observation if traffic is self-similar, which is a highly undesirable characteristic (Leland et al., 1994). However, our concern in this article is Web traffic and Web system workloads, which are strongly influenced by user behaviors. Merely recording the number of bytes sent or received gives little insight into user interaction dynamics, and so a different metric is required.

2.1. The Web session metric

Modern Web applications are complex pieces of software that are often mission-critical (e.g. for e-commerce vendors). Today, browsers must render a huge variety of active content (Javascript code, Flash movies, ActiveX controls, etc.), hosted from a wide variety of software platforms. Users navigate through this tidal wave of information (or at least attempt to) by following links within a website's structure; user navigation patterns, and the activities they engage in while on a site, are a prime focus of the discipline of *Web usage mining* (Bordogna & Pasi, 2010; de Campos et al., 2010; Mobasher et al., 2000;

Srivastava et al., 2000; Tseng et al., 2008). They are also a key focus of the literature on web systems reliability. Unlike more traditional software systems, the primary workload for Web systems is interactive search and retrieval, rather than computation. Furthermore, the user population is ill-defined; every person on the planet with an Internet connection is a potential user, and so no "standard" user profile exists. Thus, Web systems engineers must use fundamentally different metrics to measure their workloads, and to establish design goals for their systems (Tian et al., 2004).

Some of the metrics proposed for website workload measurement are drawn from the network engineering literature: the number of requests made to the server (*hits*), the number of bytes transferred, and the number of users (actually the number of unique IP addresses) would be familiar to any network engineer. They are measured from the server logs kept by each individual Web server. However, these are not particularly effective in understanding the workload of a Web system:

- The number of hits means little without knowing what objects were requested.
- The number of bytes transferred illustrates bandwidth usage but tells us nothing about the (interactive) operations leading to those transfers.
- The "number of users" is misleading because most individual users do not have a static IP address. Usually, IP addresses are dynamically assigned from a limited pool, and so one address may correspond to several users.

A more effective metric (which captures aspects of both traffic volume and user behavior) is the number of Web sessions and their duration (Tian et al., 2004). A Web session is defined as a sequence of actions taken contemporaneously by a particular user at a particular website. The length of a session thus depends on the nature of the users' interaction with a website; for example, a user watching a feature-length movie streamed from a website will have a much longer interaction (and receive a much greater volume of data) than a user who merely checks the current weather on a news site. This metric is unique to Web traffic, because it captures the notion of user behavior, as recorded by the sequence of HTTP requests made to the site by the user.

The Web session is one of the most popular units of measure for website traffic analysis, workload analysis, and user behavior modeling, and has been utilized by numerous researchers (Arlitt & Jin, 2000; Arlitt & Williamson, 1997; Arlitt et al., 1998; Goseva-Popstojanova et al., 2004, 2006a, 2006b). Cherkasova and Phaal (1998) proposed the Web session metric, with further discussion in Menasce et al. (1999, 2000); note, however, that Web sessions were discussed in an earlier paper by Crovella and Bestavros (1997). Since that time, the Web session metric has been employed in Web mining studies to create personalization algorithms (Eirinaki & Vazirgiannis, 2003; Mobasher et al., 2000); studies of search engine utilization (Jansen & Spink, 2003); and was the proposed unit of measure in a load-managing algorithm for improved quality of service (Cherkasova & Phaal, 2002). Sessions offer much finer-grained information than the standard number of users metric, and can be the basis for much more detailed investigations (e.g. studying the number of bytes transferred, errors and types of errors per session (Goseva-Popstojanova et al., 2006a; Huynh & Miller, 2009), the number of requests, the length of the session, and inter-session arrival times (Arlitt & Jin, 2000).

Despite its popularity, the Web session is a difficult metric to capture. HTTP is a *stateless* protocol, meaning that no history of actions is maintained between one user request and the next. Web applications that require state information, for instance, often use session cookies to *simulate* a stateful connection (although this workaround introduces security vulnerabilities (Tappenden & Miller, 2009)). In order to determine the end-point of a session, the typical approach is to define a Session Time-out Threshold (STT), which defines a period of user inactivity that signals an end to a given session. In recent work, Goseva-Popstojanova et al. (2006a, 2006b) use STT = 30 min; this is a common value used by other researchers (Berendt, 2001; Mahoui & Cunningham, 2000; Mat-Hassan & Levene, 2005; Spiliopoulou et al., 2003). This value was rounded up from a suggested STT of 25.5 min due to Catledge and Pitkow (1995), who claimed that the most "statistically significant" events occurred within 1.5 standard deviations (2.5 min) from the mean time between each user event (9.3 min). By contrast, Huynh and Miller (2009) proposed a probabilistic model to determine the STT from the Web logs of a site, meaning that the STT would be site-dependent. Although this approach is attractive as websites vary greatly, the current study will employ the 30-min threshold for comparability with previous studies.

3. Methodology

3.1. Chaotic invariants

Determining whether or not a given time series is chaotic is a complex process, in which human interpretation plays a significant role. There is no test that determines whether a finite time series arises from a stochastic, or a nonlinear deterministic, process. The current literature allows us to conclude that deterministic chaos is present *if* a chaotic invariant can be measured from the time series and has certain values; such invariants include Lyapunov exponents and the correlation dimension. The correlation dimension has been used to investigate a wide variety of complex systems (e.g. machining processes (Rajesh & Namboothiri, 2010), climatology (Chattopadhyay & Chattopadhyay, 2008) and software reliability modeling (Dick et al., 2007). The correlation dimension is currently considered to be the most robust chaotic invariant in any time series with noise contamination, which perhaps helps explain its popularity. The correlation dimension D is used to determine if the state-space attractor of a time series has a fractal geometry; if it does, then the time series exhibits chaotic behavior. A set

technically has a fractal geometry if its topological dimension D_T is strictly less than its Hausdorff dimension D_H. However, computing D_H is extraordinarily difficult for an arbitrary set, and so the criterion $D_T < D_H$ is extremely difficult to evaluate. The correlation dimension D is an estimate of D_H, but is defined as a limit; this means that we cannot directly compare D_T to D and obtain a valid result for a finite time series. We can, however, investigate the values of D; the topological dimension is always integer-valued, while D and D_H are real-valued. Thus, if D is clearly measured to be a non-integer, we can conclude that the time series appears to be chaotic (Kantz & Schreiber, 2003; Yamaguti et al., 1997).

We begin our analysis by reconstructing the state space of the process that generated the time series; while we cannot do so uniquely, the *delay embedding* technique allows us to create an equivalent state space (i.e. the two are related by a smooth, invertible mapping). We would then typically use a noise-reduction technique, as the correlation dimension only tolerates a noise amplitude of 2–3% of the signal. We then calculate the correlation sum, given by

$$C(\varepsilon) = \frac{2}{N(N-1)} \sum_{i=1}^{N} \sum_{j=1}^{N} \Theta(\varepsilon - \|x_i - x_j\|) \tag{1}$$

where N is the number of delay vectors in the time series, ε is a neighborhood, Θ represents the Heaviside step function, and x_i, x_j are delay vectors with $i \neq j$. The correlation sum counts the number of pairs of delay vectors that are within an ε-neighborhood of each other. For small values of ε and infinite N, $C(\varepsilon) \propto \varepsilon^D$, and the correlation dimension is defined as

$$D = \lim_{\varepsilon \to 0} \lim_{N \to \infty} \frac{\partial \ln C(\varepsilon)}{\partial \ln \varepsilon} \tag{2}$$

With N being finite for any real-world time series, we cannot directly use this definition. Instead, we plot the local slopes of the correlation sum against the neighborhood ε on a semi-logarithmic scale for several values of the embedding dimension. If for all embedding dimensions $m > m_0$ there is region where the curves all clearly plateau at a single value (or more commonly within a tolerance of a value), then that value is the correlation dimension. The extent of this plateau is considered the "scaling region," the range of length scales for which self-similarity is observed. Unlike the ideal of self-similarity, real-world data is noisy (denying us access to infinitesimal length scales), while data points become too sparse at large length scales for the correlation sum to be valid. Note that the correlation sum can be computed automatically, while the correlation dimension has to be determined through interpretation of this plot (Kantz & Schreiber, 2003).

There are some important caveats to using the correlation dimension. First, it is only intended as a measure of spatial correlation, and can be confounded by temporal correlations (which should occur in any dataset). Thus, data points separated by less than some threshold length of time should be excluded from the correlation sum; this threshold can be determined from the space–time separation plot. Secondly, the time series is assumed to be stationary; this assumption can also be checked using the space–time separation plot (a stationary time series will exhibit a plateau or level oscillation; a non-stationary one will not have this behavior). Finally, the correlation dimension can only quantify determinism that is known to be present; it cannot itself exclude the possibility that the time series is stochastic. A stochastic process does not follow an attractor in phase space, *but it could seem to* in a finite time series. Thus, a test for deterministic behavior is needed. In the literature, the most general alternative that can be tested is that the time series arises from a linear Gaussian process that has been distorted by a nonlinear observation function. This is the null hypothesis in the *method of surrogate data*, which consists of an elaborate shuffle of the elements of the time series to match the rank statistics of a linear Gaussian process. We generate $(2/(1-\chi)) - 1$ realizations of a Gaussian process, and create a shuffled dataset matching each one. This gives us a confidence level of χ (typically 95%). The shuffled datasets and the original are then compared using a test statistic that measures some aspect of determinism; time reversal asymmetry, or a nonlinear prediction error, are typical possibilities. If the statistic for the original dataset is maximal or minimal, we can reject the null hypothesis; this rank-based approach is necessary as the distribution of the test statistics is unknown (Kantz & Schreiber, 2003).

3.2. Data collection

Server logs from the websites of two separate organizations are investigated in the current paper. The first is the website of the Department of Electrical and Computer Engineering at the University of Alberta[1] ("ECE"). The website is important to both the public presence and operation of the Department, but is not mission critical. The website is dynamic, using the ColdFusion[2] scripting language, as well as the Apache HTTP Daemon. Log files covering a period of 11 months (albeit not continuously) were obtained from the system administrators. These logs record approximately 2.42 million hits with 203,896 "unique" visitors and 22.6 GB of data transferred. The data from this website represent a baseline that is similar to the bulk of websites previously studied in Goseva-Popstojanova et al. (2006a); a high-volume, non-commercial, non-mission-critical website.

The second website studied ("Site A") is operated by a company that specializes in online databases. This is a commercial website that is crucial to the operator's business. The website charges customers for the time used to access its online database; hence, an outage directly translates into lost revenue. Users of this website can be characterized as customers intending to purchase a product or register for a training course. The website is one of the company's core revenue streams, and

[1] http://www.ece.ualberta.ca/.
[2] http://www.macromedia.com/software/coldfusion.

Table 3.1
Dataset characteristics in previous studies.

Source	Log data	Log duration	Requests	Data transferred (GB)
Goseva-Popstojanova et al. (2006a)	NASA-Pvt1	20 weeks	23 thousand	0.5
	NASA-Pvt2	20 weeks	92 thousand	0.2
	NASA-Pvt3	20 weeks	489 thousand	2.2
	NASA-Pub1	20 weeks	93 thousand	9
	NASA-Pub2	20 weeks	732 thousand	6.7
	NASA-Pub3	20 weeks	108 thousand	4.6
	CSEE	6 weeks	5.8 million	80.9
	WVU	3 weeks	37.9 million	97
	ClarkNet[a]	2 weeks	3.3 million	27.6
	NASA-KSC	2 months	3.5 million	62.5
	Saskatchewan	7 months	2.4 million	12.3
Goseva-Popstojanova et al. (2006b)	WVU	1 week	15.8 million	34.5
	ClarkNet[a]	1 week	1.7 million	13.8
	CSEE	1 week	397 thousand	10.1
	NASA-Pub2	1 week	39 thousand	0.3
Current study	Site A[a]	41 months	1.9 million	33.5
	ECE	51 weeks	2.4 million	22.6

[a] Denotes commercial website.

thus clearly qualifies as a mission-critical system. Pages are dynamically generated; technologies deployed in the delivery of this website include the PHP[3] scripting language, MySQL[4] database, and Apache HTTP Daemon. Log files covering 27 months of operation (again, not continuously) were obtained. These logs record 1.9 million hits, with 63,500 "unique" visitors and 33.5 GB of data transferred. It is believed that this log represents the longest period of capture and the only truly "mission critical" log reported within the research literature. Table 3.1 provides a summary of the log data used in previous studies, as well as the current study. Note that a heavy-tail analysis of the ECE and Site-A data has been performed in Miller and Huynh (2010), which found that this model did *not* fit the data.

Although the websites examined by previous studies have higher traffic density, the periods covered are shorter. The long collection period for our data may provide several benefits over shorter periods:

- User behavior can be expected to evolve over time, and thus only an incomplete snapshot of a user's behavior would be captured in a short period. For instance, a new user on a Wiki may simply read articles; once familiar with the site, the user may choose to post comments, provide feedback or even participate in editing articles.
- The website also evolves over time. Advertising campaigns or public announcements can create time-dependent changes in website traffic. An example is the 1500% traffic increase GoDaddy.com observed following a Super Bowl ad campaign.[5]
- Well-known periodic patterns (e.g. the "weekend effect" (Arlitt & Jin, 2000) will distort short-term data collection. Longer collection periods will reveal these patterns for what they are. In addition, major Web events can create short-term traffic spikes; popular YouTube videos are known to generate millions of hits in a short time before the site returns to normal traffic.
- Any quality-of-service (QoS) problems on the website can significantly impact short-term traffic. Users will only wait for a matter of seconds before navigating away if QoS is poor.

3.3. Data preparation

The log files are stored in the Common Log Format[6] for the ECE website, and the Combined Log Format[7] for Site A. A custom log parser was created in Ruby to extract the log data and store it in a relational database. This approach to data collection can be interpreted as a deep log analysis technique (Mahoui & Cunningham, 2000; Mat-Hassan & Levene, 2005; Mobasher et al., 2000). An estimate of session length requires both a start time and an end time; these times are determined from the first and last requests deemed to be a part of the session. We thus remove all sessions that consist of only one request from the dataset. Furthermore, both the ECE and Site-A datasets contain periods of more than a day where no data is available. We cannot simply delete these intervals, as this would cause a spurious high-frequency spike in the time-series data. Instead, we split ECE into four segments (around the three gaps in the data), and split Site-A into two segments around the single gap in that dataset.

[3] http://www.php.net.
[4] http://www.mysql.com.
[5] http://www.comscore.com/press/release.asp?press=742.
[6] http://httpd.apache.org/docs/1.3/logs.html#common.
[7] http://httpd.apache.org/docs/1.3/logs.html#combined.

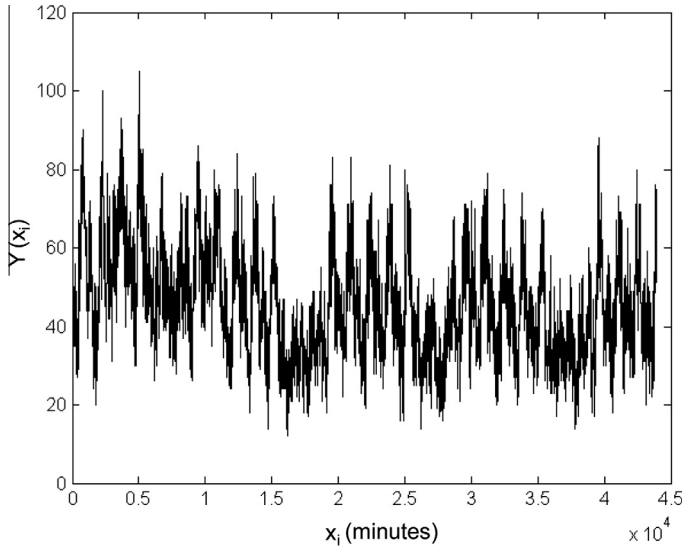

Fig. 4.1. Dataset ECE-1.

The log files contain requests from robots and other automated systems that should be removed as they are not actual requests from web users. Automated systems are classified as systems that repeatedly request a resource from the website after a set period. For example, upon investigation of Site A's logs, requests from two monitoring services are identified. The first service requests a resource from Site A every 30 min while the second service requests a resource from Site A every 66 min. The resources these services request are unique and not publicly available; hence removing them simply involves identifying requests for these resources in the log. Robots that automatically request the "robots.txt" resource are also removed from both Site A and ECE log files.

We finally construct a time series from the "cleaned" log data. This time series records the number of sessions active on the website during an interval of time. The length of this interval is 1 min, meaning that our longest time series consists of over 1.3 million observations. The count-per-interval data representation also prevents complications in the data analysis that would arise if the observation intervals in the time series were not uniform (we would have to transform the data to uniform sampling using a Poincaré section (Kantz & Schreiber, 2003). Algorithms for nonlinear time series analysis always assume that the observations are real-valued; integer-valued data requires special handling, as data points in phase space fall exclusively on a grid (which will appear deterministic, even if the data arise from a purely random process!) Following techniques in Kantz and Schreiber (2003), we thus add white noise in the interval $[-0.5, 0.5]$ to each data point. This results in "shaking the points off the grid" (Kantz & Schreiber, 2003), and thus breaking up the false "structure" in the time series. After testing for determinism in the resulting dataset, we then process the dataset with a nonlinear noise reduction technique before attempting to extract chaotic invariants.

4. Data analysis

4.1. Delay embedding

Figs. 4.1–4.6 present the number of Web sessions per minute in our datasets. Figs. 4.1–4.4 present datasets ECE1 to ECE4, respectively, while Figs. 4.5 and 4.6 present datasets Site A-1 and Site A-2, respectively. One particular concern emerged on analysis of the Site A-2 data: there was a period of anomalous data during which the number of sessions was identical for an extended period. As we believe that this was a data collection problem, we removed the first 300,000 data points from Site A-2; the reduced dataset in Fig. 4.6 covers the period 4/27/2006 to 5/1/2008. The peak session rate for each dataset is presented in Table 4.1.

To determine the embedding delay and dimension for each dataset, we have used the autocorrelation plot, the mutual information statistic and the technique of false nearest neighbors. For the sake of brevity, we omit the details of this process, noting merely that we follow the approach described in Kantz and Schreiber (2003). We summarize the delay embeddings we have selected in Table 4.2.

4.2. Test for deterministic behavior

With the delay embeddings determined for all six datasets, we can now use the method of surrogate data to determine if any of these datasets appear to be predominantly deterministic in nature. We will proceed with noise reduction and

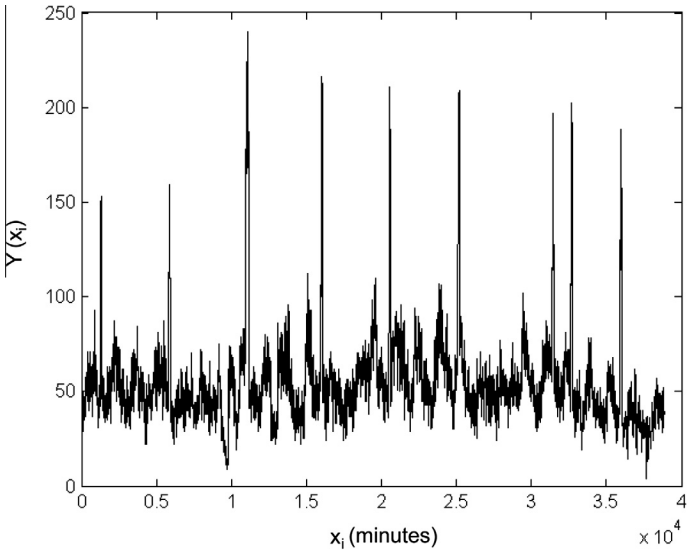

Fig. 4.2. Dataset ECE-2.

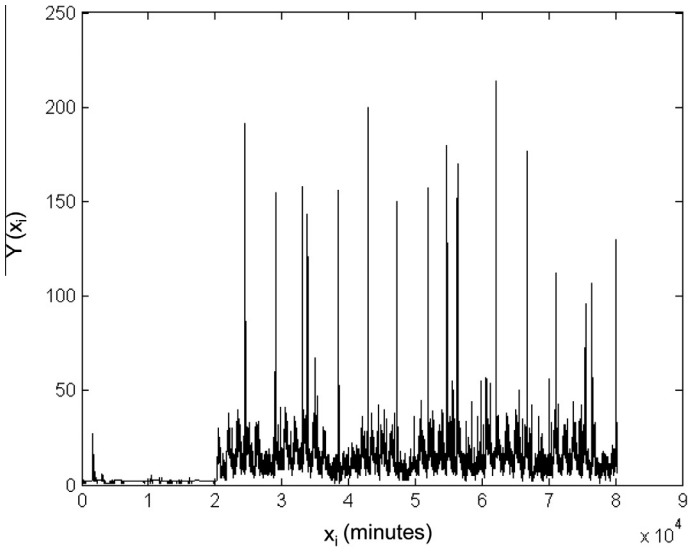

Fig. 4.3. Dataset ECE-3.

extracting nonlinear invariants only from those datasets that pass this test. In all experiments, we aim for a two-sided confidence of 95%, and so 39 surrogates are generated each time.

We obtain clear signatures of deterministic behavior in datasets ECE-1 and ECE-2, while we were not able to rule out stochastic behavior as an explanation for datasets ECE-3 and ECE-4.

Due to system limitations, we were only able to conduct surrogate data tests on segments of 100,000 time series elements at a time, corresponding to about 2½ months of data. We have therefore divided the datasets that are larger than this limit into continuous segments of 100,000 elements each. (A small number of data points at the extreme ends of each data set/segment are also discarded in order to have the first and last elements of the segments closely match; this avoids a spurious high-frequency spike when the surrogate datasets are generated using the Fourier transform (Kantz & Schreiber, 2003; Press et al., 1992).) This implies that, while a finding of no deterministic behavior is reliable, a finding of deterministic behavior is not entirely definitive; in the latter case we can only say that large segments, up to 2½ months in length, of the data sets show possible deterministic behavior. We will only proceed to extract nonlinear invariants if deterministic behavior is demonstrated for *all* segments.

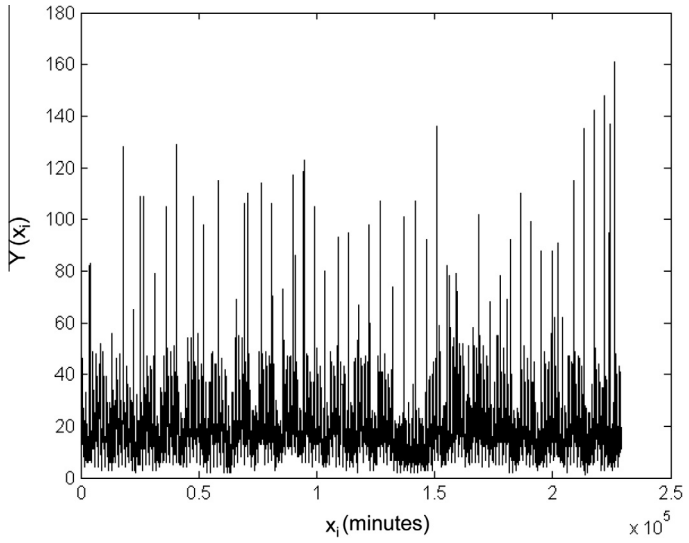

Fig. 4.4. Dataset ECE-4.

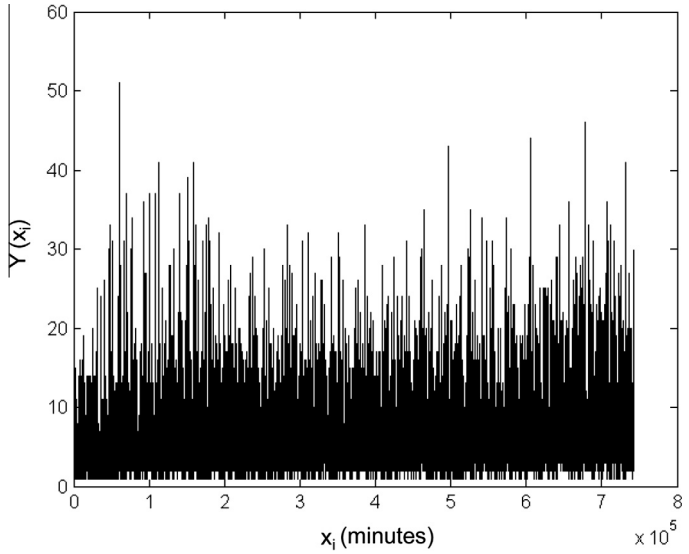

Fig. 4.5. Dataset Site A-1.

4.3. Noise reduction, stationarity, chaotic invariants

Datasets ECE-1, ECE-2 and Site A-1 appear to be generated via deterministic processes. Before attempting to extract chaotic invariants from these datasets, we need to reduce the noise in these datasets as much as possible. As discussed in Section 4.2, the noise that is known to be present in these datasets is discretization noise, which we have alleviated by adding white noise in the interval $[-0.5, 0.5]$ to each time series, to avoid biasing the results of the surrogate data test. We then employ a locally constant noise-reduction scheme to reduce the noise in the datasets. In this technique, we select a radius ε, and find all the neighbors for a given delay vector x within an ε-neighborhood of x. Then, x is replaced by the neighborhood mean. The process is repeated for all points in the datasets. The neighborhood radius ε is a free parameter, which obviously has a considerable effect on the outcome of noise reduction. Guidance for choosing the radius includes being larger than the apparent noise extent, while still being smaller than the radius of curvature of a "typical" segment of the phase-space attractor. In our work, we can definitely say that the noise radius needs to be greater than 1.0 to cover discretization noise. An upper limit cannot be determined at this point, and so we undertake parameter exploration to find a "best" value. This is judged by visual inspection of the phase portrait of the noise-reduced dataset. We want the largest possible structures to

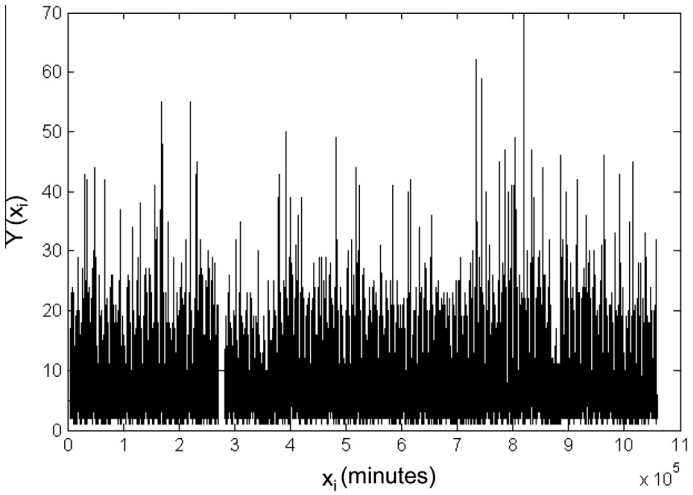

Fig. 4.6. Dataset Site A-2.

Table 4.1
Peak session rates.

Dataset	Peak session rate
ECE-1	105 sessions
ECE-2	240 sessions
ECE-3	214 sessions
ECE-4	161 sessions
Site A-1	51 sessions
Site A-2	70 sessions

Table 4.2
Delay embeddings for the six datasets.

Dataset	Dimensions	Delay
ECE-1	8	380
ECE-2	9	460
ECE-3	9	400
ECE-4	9	400
Site A-1	9	680
Site A-2	9	550

work with, while also ensuring that the neighborhood size is not excessive compared to the size and curvature of these structures.

We next check for stationarity in the noise-reduced datasets using the space–time separation plot. Each of these datasets reaches a level oscillation, and can thus be considered stationary. We can avoid temporal correlations in computing the correlation dimension if we exclude points closer than 1500 time steps in ECE-1; closer than 500 time steps in ECE-2; and closer than 700 time steps in Site A-1.

Finally, we compute and plot the slopes of the correlation sum for each dataset. In these plots, we are searching for a region of the x-axis where all the curves for embedding dimensions greater than the actual dimensionality of the attractor plateau at a common value. We do not observe any common plateaus in these plots. We must therefore conclude that the phase-space attractors of these datasets do not appear to have a fractal geometry, and thus the datasets underlying them do not exhibit chaotic behavior.

5. Predictive modeling experiments

Section 4 described evidence that some of our datasets appear to be predominantly deterministic, using the method of surrogate data. In this section, we will conduct a parallel test of this hypothesis, using predictive models based on the two explanations for self-similarity (heavy-tailed distributions vs. nonlinear determinism) through a predictive modeling task. If our hypothesis is correct (Web session data is predominantly deterministic), then a deterministic, nonlinear

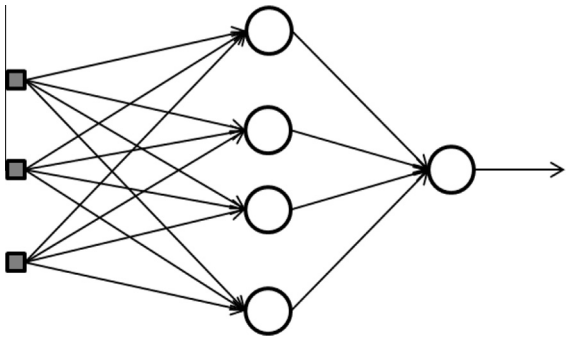

Fig. 5.1. Radial basis function network.

prediction algorithm should be significantly more accurate compared to a stochastic model. Likewise, if the hypothesis is incorrect, then the stochastic model should be significantly more accurate.

5.1. Prediction models

The two models we propose to compare are: for the deterministic model, radial basis function networks (RBF networks); and for the stochastic model, fractional auto-regressive integrated moving average (FARIMA, or sometimes ARFIMA) models. RBF networks are a well-known and widely-used neural network architecture, which are known to be universal approximators (Haykin, 2009). They have been frequently used as time-series prediction algorithms (see e.g. Chng et al., 1996; Harpham & Dawson, 2006; Lian et al., 2008); an RBF network is depicted in Fig. 5.1. It is a layered feedforward network in which the first layer neurons merely distribute the inputs to the next layer, while the second layer implements a set of radial-basis functions (Haykin, 2009):

$$y = e^{\frac{-(x-\mu)^2}{2\sigma^2}} \tag{3}$$

where x is the input vector to the neuron, μ is the center of the radial basis function, σ is its spread, and y is the output of the neuron. The third layer neurons output a weighted sum of their inputs. Training this network consists of determining the centers of the layer-2 radial basis functions (via k-means clustering) and then optimizing the layer-3 weight vector (via least-squares regression). The number of RBF neurons, and the maximum spreads of the radial basis functions, are input parameters to the training algorithm (we use the implementation in the WEKA software package).

For the stochastic alternative, we seek a model that closely matches the characteristics of the datasets (i.e., self-similarity). As self-similarity in network traffic is usually measured by the Hurst parameter, the class of Fractional ARIMA models are a natural choice. Recall that an ARIMA model is a generalization of the auto-regressive moving average (ARMA) model, given by:

$$x_t - \sum_{i=1}^{p} \phi_i x_{t-i} = \varepsilon_t - \sum_{j=1}^{q} \theta_j \varepsilon_{t-j} \tag{4}$$

where x_t is the observed value of the time series at time t, ε_t is the forecast error at time t, ϕ_i is the coefficient of the ith observation lag, θ_j is the coefficient of the jth prediction lag, and p and q are the model orders (this is usually denoted as an ARMA(p, q) model). ARMA models are only appropriate for stationary time series; ARIMA models are a generalization that accomodates a non-stationary mean. The time series is differenced as:

$$\Delta x_t = x_t - x_{t-1} \tag{5}$$

where Δx_t is the tth entry of the differenced time series. The differencing operator is applied d times, and then an ARMA(p, q) model is built for the differenced time series. The overall model is referred to as an ARIMA(p, d, q) model (Andreoni & Postorino, 2006).

In a FARIMA model, the difference parameter d can be a non-integer, and will be equal to $H - 0.5$ where H is the Hurst parameter. FARIMA models have been used for a variety of datasets exhibiting long-range dependencies (a known property of heavy-tailed distributions); and in particular have been found to be effective models for self-similar network traffic (Gorst-Rasmussen et al., 2012; Ilow, 2000; Liu et al., 1999). Based on these characteristics, we select FARIMA models as our stochastic alternative; we will use Hyndman's "forecast" package for R (Hyndman & Khandakar, 2008) to carry out our FARIMA experiments.

5.2. Methodology

Our experiments follow a common design from time-series forecasting: the dataset is split into an estimation or "training" set, and a holdout sample that is chronologically later than all data points in the training set (i.e. a chronologically-ordered

Table 5.1
RMS error for holdout samples.

Dataset	RBF network RMSE	FARIMA RMSE
ECE-1	1.624	1.660
ECE-2	1.839	1.854
ECE-3	0.758	1.541
ECE-4	1.358	1.142
Site A-1	0.824	0.821
Site A-2	0.813	0.811

single-split design). The models will be trained for the basic one-step-ahead prediction task (given a history of the dataset, predict the next observation). The models will be parameterized on the training set, and then their out-of-sample prediction accuracy will be compared on the holdout sample. In all of our experiments, we reserve 1/3rd of the dataset in the holdout sample.

Applying RBF networks to time series forecasting requires some preprocessing; an RBF network is designed to learn a static mapping from an input "feature" space to an output space, rather than a time series. However, the method of delay embeddings reviewed in Section 3 provides an elegant method for converting time-series data into a set of feature vectors; we then need only append the next observation to this vector (as our "output"), and the one-step-ahead prediction can now be determined by an RBF network (or indeed, any other function–approximation algorithm). In addition, the time dependencies between observations in the time series have now been subsumed in the delay vectors; this means that we can randomize the presentation of delay vectors to the network, allowing for a cross-validation design in our training.

Our approach for training the RBF network is therefore as follows. We first construct a delay embedding of each of the training set and holdout sample time series, using the dimensions and delays determined in Table 4.2. We conduct a parameter exploration using tenfold cross-validation on the training set. We randomly split the training set into ten partitions, combining nine into a smaller "parameterization" set and holding the remaining one out as a validation set. The RBF network is then trained on the parameterization set with a given parameter vector and its out-of-sample accuracy determined on the validation set. We repeat this process ten times, with each partition used as the validation set exactly once. We repeat this process for different parameter vectors, performing a grid search of the parameter space (in this case, the number of radial basis functions, and their spreads). We select the parameterization with the lowest mean RMS error across the ten validation partitions as the "best" parameterization on the dataset. We then train a new RBF network on the entire training set using this "best" parameterization, and finally determine its out-of-sample accuracy on the holdout sample.

For the FARIMA model, we use Hyndman's forecast package in the R environment to determine the model order (the parameters p, q, and d) on the training set. However, we are not able to compute the out-of-sample forecast directly, due to a memory leak in forecasting FARIMA models. Instead, we first obtain a model from the 'arfima' procedure on the training set. We then performed fractional differencing (using the 'fracdiff' routine) on the training and holdout sets using the d determined from 'arfima.' We then trained an ARMA model (using the model orders p and q determined from ARFIMA on the training set) on the differenced training set, and tested it (using the "forecast" package) on the differenced holdout set.

5.3. Experimental results

In Table 5.1, we present the root-mean-square (RMS) prediction errors for both models on our six holdout samples. For the ECE datasets, our results are plainly mixed. The RBF network and FARMIA are very close on ECE-1 and ECE-2, but RBF networks are substantially better on ECE-3, and FARIMA seems substantially better on ECE-4. On the Site-A data, the RBF network and FARIMA models are very close. Overall, we cannot favor one model or the other across the six datasets. We therefore find that our hypothesis (the RBF Network would be substantially more accurate than FARIMA) is not supported. While it is interesting that a purely deterministic model performs as well as a FARIMA model (and supports our earlier finding that deterministic components play an important role in these datasets), in the end we do not find sufficient evidence to overturn the existing consensus that Web sessions are best modeled as stochastic, heavy-tailed processes.

We can also say that the overall prediction quality for both models was good. The relative absolute error for the RBF networks (the ratio of the absolute error of the model vs. the absolute error obtained by simply predicting the mean value of the time series) was under 10% for all datasets, meaning an order-of-magnitude improvement over a naïve model.

6. Conclusions

Our goal in this article was to provide a rigorous empirical investigation of the possibility of deterministic chaos in Web session durations. We collected web access logs from two separate organizations: one is a department at a very large public university; the other is a private company. The university website is important to the operation of the department, but is not mission-critical; the corporate website, by contrast provides one of the core revenue streams for the company, and is thus a mission-critical website. From these logs, we extract a total of six datasets, each covering one continuous period of operation (none of which are less than 1 month). Each dataset is a time series that records the number of open Web sessions in 1-min

intervals. We then employ the methods of nonlinear time series analysis to test for the presence of chaotic behavior in these datasets. In three of the datasets (two from the university web site, one from the corporate site), we determined that the deterministic components of the signal (i.e. the well-known *weekend effect*) appear to dominate the stochastic components of the signal. However, a subsequent check for the presence of deterministic chaos was negative; we were not able to extract a correlation dimension for any of these datasets. We then contrasted deterministic and stochastic models in a time-series prediction experiment on all six datasets, finding that the two models performed very similarly. Ultimately, we found that there was insufficient evidence to favor deterministic models over the existing model of heavy-tailed processes.

In future work, we plan to replicate our predictive modeling study using web access logs from a broader array of websites. We first want to check if RBF networks and FARIMA models are still equally effective across a broad range of websites. If they are, we will seek to understand why two such radically different models would have such similar performance on these datasets. One intriguing possibility is that the mix of stochastic and deterministic components might be nearly "balanced;" if so, Web session models for load testing should incorporate this characteristic (perhaps as a fine-grained hybridization of RBF networks and heavy-tailed processes). We will also investigate intelligent algorithms for determining the length of a session. The use of a simple timeout threshold for capturing such a complex idea as user behavior is almost certainly less than optimal, and this will directly impact the accuracy of the Web session metric.

Acknowledgements

We thank the Department of Electrical and Computer Engineering at the University of Alberta, and the operators of "Site A," for providing access to their web server logs. This research was supported in part by the Natural Sciences and Engineering Research Council of Canada under Grant Nos. G121210906 and G121210670.

References

Andreoni, A., & Postorino, M. N. (2006). A multivariate ARIMA model to forecast air transport demand. *Presented at the European transport conf., Strasbourg, France*.
Arlitt, M. et al (1998). Workload characterization of a Web proxy in a cable modem environment. *ACM Sigmetrics Performance Evaluation Review, 27*, 25–36.
Arlitt, M., & Jin, T. (2000). A workload characterization study of the 1998 World Cup Web site. *IEEE Network, 14*, 30–37.
Arlitt, M., & Williamson, C. L. (1997). Internet web servers: Workload characterization and performance implications. *IEEE/ACM Transactions on Networking, 5*, 631–645.
Berendt, B. et al. (2001). Measuring the accuracy of sessionizers for web usage analysis. In *Proc. SIAM int. C. data mining, workshop on web mining, Arlington, VA, USA* (pp. 7–14).
Bordogna, G., & Pasi, G. (2010). A flexible multi criteria information filtering model. *Soft Computing, 14*, 799–809.
Botta, A. et al (2012). A tool for the generation of realistic network workload for emerging networking scenarios. *Computer Networks, 56*, 3531–3547.
Catledge, L. D., & Pitkow, J. E. (1995). Characterizing browsing strategies in the World-Wide Web. In *Proc. 3rd int. World-Wide Web conference: Technology, tools and applications, Darmstadt, Germany* (pp. 1065–1073).
Chattopadhyay, G., & Chattopadhyay, S. (2008). A probe into the chaotic nature of total ozone time series by correlation dimension method. *Soft Computing, 12*, 1007–1012.
Cherkasova, L., & Phaal, P. (1998). *Session based admission control: A mechanism for improving the performance of an overloaded web server*. Palo Alto, CA, USA: HP Labs.
Cherkasova, L., & Phaal, P. (2002). Session-based admission control: A mechanism for peak load management of commercial web sites. *IEEE Transactions on Computers, 51*, 669–685.
Chng, E. S. et al (1996). Gradient radial basis function networks for nonlinear and nonstationary time series prediction. *IEEE Transactions on Neural Networks, 7*, 190–194.
Crovella, M. E., & Bestavros, A. (1997). Self-similarity in world wide web traffic: Evidence and possible causes. *IEEE/ACM Transactions on Networking, 5*, 835–846.
de Campos, L. M. et al (2010). Using second-hand information in collaborative recommender systems. *Soft Computing, 14*, 785–798.
Dick, S. et al (2007). Software reliability modeling: The case for deterministic behavior. *IEEE Transactions on Systems, Man and Cybernetics, Part A – Systems and Humans, 37*, 106–119.
Eirinaki, M., & Vazirgiannis, M. (2003). Web mining for web personalization. *ACM Transactions on Internet Technology, 3*, 1–27.
Erramilli, A. et al (1995). An application of deterministic chaotic maps to model packet traffic. *Queueing Systems, 20*, 171–206.
Gorst-Rasmussen, A. et al (2012). *Why FARIMA models are brittle*. Cornell University Library.
Goseva-Popstojanova, K. et al (2006a). Empirical characterization of session-based workload and reliability for Web servers. *Empirical Software Engineering, 11*, 71–117.
Goseva-Popstojanova, K. et al. (2004). Empirical study of session-based workload and reliability for Web servers. In *Proc. IEEE int. symp. software reliability engineering, Saint-Malo, Bretagne, France* (pp. 403–414).
Goseva-Popstojanova, K. et al. (2006). A contribution towards solving the Web workload puzzle. In *Proc. int. C. dependable systems and networks, Philadelphia, PA, USA* (pp. 505–516).
Harpham, C., & Dawson, C. W. (2006). The effect of different basis functions on a radial basis function network for time series prediction: A comparative study. *Neurocomputing, 69*, 2161–2170.
Haykin, S. (2009). *Neural networks and learning machines* (3rd ed.). Upper Saddle River, NJ, USA: Pearson Education Ltd..
Huynh, T., & Miller, J. (2009). Another viewpoint on "evaluating web software reliability based on workload and failure data extracted from server logs". *Empirical Software Engineering, 14*, 371–376.
Huynh, T., & Miller, J. (2009). Empirical observations on the session timeout threshold. *Information Processing & Management, 45*, 513–528.
Hyndman, R. J., & Khandakar, Y. (2008). Automatic time series forecasting: the forecast package for R. *Journal of Statistical Software, 27*, 1–22.
Ilow, J. (2000). Forecasting network traffic using FARIMA models with heavy tailed innovations. *Presented at the proc. ICASSP, Istanbul, Turkey*.
Jansen, B. J., & Spink, A. (2003). An analysis of web documents retrieved and viewed. In *Proc. 4th int. C. internet computing, Las Vegas, NV, USA* (pp. 65–69).
Kantz & Schreiber (2003). *Nonlinear time series analysis* (2nd ed.). Cambridge, UK: Cambridge University Press.
Leland, W. E. et al (1994). On the self-similar nature of Ethernet traffic (extended version). *IEEE/ACM Transactions on Networking, 2*, 1–15.
Lian, J. et al (2008). Self-organizing radial basis function network for real-time approximation of continuous-time dynamical systems. *IEEE Transactions on Neural Networks, 19*, 460–474.
Liu, J. et al. (1999). Traffic modeling based on FARIMA models. *Presented at the CCECE, Edmonton, AB, Canada*.

Mahoui, M., & Cunningham, S. J. (2000). A comparative transaction log analysis of two computing collections. *Lecture Notes in Computer Science, 1923*, 418–423.
Mandelbrot, B. B. (1969). Long-run linearity, locally Gaussian processes, H-spectra and infinite variances. *International Economic Review, 10*, 82–113.
Mat-Hassan, M., & Levene, M. (2005). Associating search and navigation behavior through log analysis. *Journal of American Society for Information Science and Technology, 56*, 913–934.
Meier-Hellstern, K. et al. (1991). Traffic models for ISDN data users: Office automation application. In *Proc. 13th int. teletraffic congress, Copenhagen, Denmark* (pp. 167–172).
Menasce, D. A. et al. (1999). A methodology for workload characterization of e-commerce sites. In *Proc. ACM C. electronic commerce, Denver, CO, USA* (pp. 119–128).
Menasce, D. A. et al. (2000). In search of invariants for e-business workloads. In *Proc. ACM C. electronic commerce*, Minneapolis, MN, USA (pp. 56–65).
Miller, J., & Huynh, T. (2010). Investigating the distributional property of the session workload. *Journal of Web Engineering, 9*, 25–47.
Mobasher, B. et al (2000). Automatic personalization based on Web usage mining. *Communications of the ACM, 43*, 142–151.
Mondragon, R. J. et al (2001). Chaotic maps for traffic modelling and queueing performance analysis. *Performance Evaluation, 43*, 223–240.
Press, W. H. et al (1992). *Numerical recipes in C: The art of scientific computing* (2nd ed.). Cambridge, UK: Cambridge University Press.
Rajesh, V. G., & Namboothiri, V. N. N. (2010). Flank wear detection of cutting tool inserts in turning operation: Application of nonlinear time series analysis. *Soft Computing, 14*, 913–919.
Spiliopoulou, M. et al (2003). A framework for the evaluation of session reconstruction heuristics in Web usage analysis. *INFORMS Journal of Computing, 15*, 171–190.
Srivastava, J. et al (2000). Web usage mining: Discovery and applications of usage patterns from web data. *SIGKDD Explorations, 1*, 12–23.
Staff (2010). *Internet user forecast by country*. Arlington Heights, IL, USA: Computer Industry Almanac Inc.
Staff (2012). Cisco *visual networking index: Forecast and methodology, 2011–2016*. San Jose, CA, USA: Cisco Systems, Inc.
Tappenden, A. F., & Miller, J. (2009). Cookies: A deployment study and the testing implications. *ACM Transactions on the Web, 3*, 9:1–9:49.
Taqqu, M. S., & Levy, J. B. (1986). Using renewal processes to generate long-range dependence and high variability. In E. Eberlein & M. S. Taqqu (Eds.), *Dependence in probability and statistics*. Boston, MA, USA: Birkhauser.
Tian, J. et al (2004). Evaluating web software reliability based on workload and failure data extracted from server logs. *IEEE Transactions on Software Engineering, 30*, 754–769.
Tseng, V. S. et al (2008). Prediction of user navigation patterns by mining the temporal web usage evolution. *Soft Computing, 12*, 157–163.
Yamaguti, M. et al (1997). *Mathematics of fractals*. Providence, RI, USA: American Mathematical Society.