WWW TRAFFIC MEASURE AND ITS PROPERTIES

MARCUS R. KEOGH-BROWN
SCHOOL OF MEDICINE HEALTH POLICY AND PRACTICE
UNIVERSITY OF EAST ANGLIA
NORWICH NR4 7TJ
M.KEOGH-BROWN@UEA.AC.UK
UK
AND BARBARA BOGACKA
SCHOOL OF MATHEMATICAL SCIENCES
QUEEN MARY
UNIVERSITY OF LONDON
LONDON
E1 4NS
UK
B.BOGACKA@QMUL.AC.UK

Abstract. We present a method to extract a time series (Number of Active Requests (NAR)) from web cache logs which serves as a transport level measurement of internet traffic. This series also reflects the performance or Quality of Service of a web cache. It has long-memory properties but is not self-similar and does not have a heavy-tailed distribution.

However, the long-memory and autocorrelation structure of NAR are preserved through aggregation, that is the aggregated series has similar statistical properties to the original one. We call this property aggregation similarity.

Aggregation similarity is a very useful property, which makes management of large data sets easier and speeds up the asymptotic properties of time series.

Key Words: Long memory process; Cache log data; Number of Active Requests

1. Introduction

Over recent years the internet has become an essential tool for communication and data transfer. As is the case for most computer related services, users constantly require faster, more flexible and more reliable performance from their internet transfers and, in order to provide for these requirements, a greater understanding of the traffic is needed.

The perspective of our work is more closely related to the internet user's view of the traffic than the underlying machine view. We do not study each tiny item of data (packet) in detail, but instead, we study the data at a fairly high level, monitoring its behavior over long timescales. By examining the data in this way, some of the properties of detailed packet level traffic are filtered out, but we are able to study the Quality of Service (QoS) perceived by the internet user and we hope our studies will help to provide ways in which this QoS can be improved.

Various attempts have been made to model web traffic. There is a vast literature in engineering journals on some traffic measures and their properties, see for example [12] and [7] which study file popularity and fitted Zipf curves to internet data, [14], a study of data from personalised web content to improve caching of personalised content, or [13], a study of TCP packet flows and of a new method for timing out TCP flows to reduce delays. However, there is much less attention given to the problem in the literature on data analysis. In fact, there are either very theoretical papers on long memory processes (for example [10] provides a description of parameter estimation methods for long-memory FARIMA processes and fractional differencing of a time series, and [4] describes short and long range processes together with their properties) or there are papers presenting some simulation studies of internet traffic behaviour (like [3], which presents a tool to simulate web cache traffic and [1] which presents a simulation of long-range dependent packet trains which commence according to a poisson process). Each of these pieces of research makes a valuable contribution to their field either by providing statistical methods or tools to use in analysis, or by studying aspects of network traffic, or by producing synthesised data which can test theoretical web cache or networking situations. Our research takes a different approach: rather than focussing on an attribute of web users or requests, we study internet traffic at a high level over long timescales and thus gain a view of the overall quality of service provided by a web cache.

What we propose in this paper is a new traffic measure based on real data sets. We examine its statistical properties and suggest further analysis.

In Section 2 we present the data sources (caches and their log files), we briefly discuss the problems with working on the row data and we introduce a new variable, called Number of Active Requests (NAR), a traffic measure built on the available information in the cache log files. In Section 3 we introduce a definition of a very interesting new property (aggregation-similarity) we have discovered in all NAR series studied in this work. We further use this property to examine the behaviour of NAR and so of the internet traffic. Section 4 briefly describes the numerical programs used for the calculations. We conclude in Section 5.

2. Number of Active Requests

2.1. **Data Source.** Internet traffic datasets have been extracted and analysed in many different ways. However, internet and network datasets are very difficult to obtain, and, even when they are released for analysis, only certain items of data are provided. The data chosen for our analysis was web cache logs.

The web cache logs we use were provided by three main organisations: NLANR (National Laboratory for Applied Network Research), the University of Essex and Queen Mary, University of London. Our main study is based on long-term traces from eleven web caches, nine of which are administered by NLANR. The list of the caches and their location is given in Table 4 in Appendix A. The University of Essex cache is one of two web caches serving University of Essex computer users. These two caches were configured in round-robbin sharing formation. This cache has used the JANET server network as a parental cache in the past, but, at the time we obtained log files, it no longer utilised this link. The JANET cache network is a caching service provided jointly by Manchester University and Loughborough University for the UK Academic Research Community. The Queen Mary cache is the single web cache serving Queen Mary, University of London and is linked to the JANET cache network. However, the choice of situation under which this link to JANET is utilised is determined by the Queen Mary Web Cache Administrator.

Although the NLANR caches are administered by the same organisation, the user communities and rules governing the caches' service vary. The NLANR **stp** and **sj** caches serve as network caches on internet or network exchange points. Their brief job description is to cache files transferred through the network exchange points on which they are situated. For the other NLANR caches, the setup is slightly different. Individual users wishing to subscribe to the NLANR caching service may query the **sd** cache. Otherwise, the cache administrator of the organisation wishing to subscribe to the caching service may choose to use any two of the remaining NLANR caches (**bo1, bo2, pa, pb, rtp, sv, uc**). It is reasonable to assume that the choice of cache would be made according to locality of the web cache or the parental links of that cache. The web cache size, in terms of the number of requests received per day is given in Table 1 in Appendix A.

As Table 1 shows, Essex University is the largest of our caches receiving, on average, more than 2 million requests per day. **sv**, **rtp** and **pb** are the next largest caches, each receiving over 1 million requests per day.

For completeness we should point out that values in Table 1 are calculated from the data sets used for our analysis and to which we will refer throughout this work. The precise periods of data collection for our study are in Table 2 in Appendix A.

2.2. **Raw Data.** Web cache logs are vast. They frequently contain up to 2 million rows of data per day, each representing a single file request. Every row is separated into 10 columns, each representing an attribute of that request. However, just three of the data columns contain numerical values useful for analysis of traffic as a whole, the others represent attributes of the file.

The three raw data items of interest are Request Time (the time at which the file transfer was completed by the cache), Elapsed Time (the duration of the file transfer in milliseconds) and File Size (the size in bytes of the transferred file). An example of a log file is given in Appendix B. However, Request Time is an approximately linear set of time points making it difficult to use by itself for traffic analysis. The other two attributes, Elapsed Time and File Size, have inconsistent statistical properties. A single day data of Elapsed Time or of File Size may be uncorrelated on one day, but significantly correlated on the next day. Similarly, there may be huge bursts of very large Elapsed Time or File Size values which cluster together on one day, but just a few isolated bursts or large values the next day.

These statistical inconsistencies would make raw data difficult to analyse, since a conclusion which might hold for one period of data might not apply to another period of data. In addition to this, as we have already mentioned, the web cache log datasets are vast, and, for some caches it would be very difficult to even consider a single day's raw data.

2.3. **A New Series.** It was our intention to find a dataset which was manageable in size, accurately reflected the properties of internet traffic and had consistent statistical properties between different timescales. We introduce a new variable called *Number of Active Requests* or $NAR(t), t = 0, 1, 2, \ldots$. $NAR(t)$ is the number of file requests that are undergoing transfer from a web cache at time t. The diagram in Figure 1 explains the method of calculating the values of $NAR(t)$. For example, at $t = 3$ there are five files being transferred from the cache, so $NAR(3) = 5$ for this cache. For our purposes, we always calculate NAR at one second intervals. Time series plots of a week-long NAR for two different caches are shown in Figures 2 and 3.

NLANR **rtp** cache remains active almost all of the time, whereas Queen Mary cache is frequently inactive. Both caches show daily peaks with much shorter peaks at weekends, although the pattern of daily peaks is more clearly shown in the Queen Mary plot. The **rtp** cache plot is typical of NAR activity in NLANR caches, where a NAR value of zero is rare. Similarly the University of Essex cache remains active at all times, though the daily trends are less visible. The Queen Mary cache has the most clearly defined daily pattern of the caches we have studied. All of the cashes studied present the NAR series as having local trends and cycles and peaks, but over a long period being stationary. This suggest that it is a long memory process, which we examine in Section 3.1

The calculation of the NAR data series was implemented using a Perl program. Perl was selected for its' ability to process large files. A broad illustration of the algorithm implemented in the Perl program is shown in Figure C in Appendix C. The Perl code is available on request from the first author.

[11] presents a study of internet data in perhaps the most similar way to our studies that we have seen. Traffic flows, where the length of a flow is the time from the start of the first packet transmission to the end of the last, are considered. Flow lengths are plotted on the horizontal axis in the order in which they occur with horizontal length proportional to flow length, and are plotted vertically according to a random shift element. These plots are similar to our Figure 1, when rotated through 90 degrees, where the lines indicating requests are ordered randomly. The idea behind this visualisation is to distinguish the short transfers (mice) from the long flows (elephants) causing the possibly heavy-tailed nature of the data distribution. The authors agree with the popular theory that long flows also lead to long-range dependence of the data (significant correlation at large lags), and illustrate that the heavy tail of the data distribution is attributable to a very few elephants.

Although no conclusion section is featured in the paper, the authors show that random sampling of heavy-tailed data does not produce a representative sample since elephants are easily omitted and that sub-sampling such data to produce a representative sample would be a very difficult task.

These traffic flows correspond to the internet data which NAR time series measures. The main difference between the traffic data presented in [11] and the NAR is that NAR measures the number of requests active at a regular specific time point, and we model these values as a time series, while [11] considers the number of large (elephant) and small (mice) file transfers that occur within wide and narrow intervals of time. Therefore, we concentrate on the number of requests that are active at various time points but do not consider the duration of each of those requests directly. Conversely [11] studies the number of large or small transfers in an interval, without having any direct knowledge of the business in that interval.

3. Properties of the Number of Active Requests

3.1. **Long-Memory.** Here we follow the definition of long memory given by [2], though the original reference for long memory process is [8]. First we introduce notation which we use throughout the paper. Let $X = \{X_1, X_2, \ldots\}$ be a random stationary process with a finite expectation $E(X) = \mu$, a finite variance $Var(X) = \sigma^2$ and covariances $Cov(X_t, X_{t+\tau}) = \gamma(\tau)$, $t = 1, 2, \ldots$, $\tau = 0, 1, \ldots$. That is

$\gamma(0) = \sigma^2$ and $\rho(\tau) = \frac{\gamma(\tau)}{\gamma(0)}$ is the autocorrelation function at lag τ. Also, we denote by $f(\lambda)$ the spectral density function of the series at frequency λ.

Furthermore, we denote by

$$X^{(m)} = \left\{X_1^{(m)}, X_2^{(m)}, \dots, X_t^{(m)}, \dots\right\}, \tag{3.1}$$

where

$$X_t^{(m)} = \frac{1}{m}\left(X_{tm-m+1} + \dots + X_{tm}\right), m = 1, 2, \dots, \tag{3.2}$$

is a new aggregated process with the aggregation parameter m. That is, process $X^{(m)}$ is built up by replacing each non-overlapping set of m values of X with their mean.

In Figures 4 and 5, we show an example of the autocorrelation function and periodogram for the NAR process. According to the definition of a long-memory process, which we recall below, both plots indicate that NAR has this property.

Definition 3.1. From [2].

A stationary process $X = \{X_1, X_2, \dots\}$ is called long-range dependent or long-memory process, if it satisfies the following equivalent conditions

(i): $\sum_{\tau=0}^{\infty} \gamma(\tau)$ is divergent,
(ii): $f(\lambda)$ is singular at $\lambda = 0$,
(iii): $m\sigma_m^2 \to \infty$ as $m \to \infty$,
where σ_m^2 denotes variance of the aggregated series $X^{(m)}$.

The slow autocorrelation decay in Figure 4 suggests that the NAR fulfills criterion (i), while criterion (ii) in the above definition is exhibited by the periodogram plot in Figure 5. As the frequency approaches zero, the values of the periodogram increase rapidly indicating a pole in the spectrum at zero.

The long memory parameter commonly used in the literature (see for example [5]) is the so called Hurst parameter H. For long-memory processes $H \in (\frac{1}{2}, 1)$, and the autocorrelation function is asymptotically approximated by

$$\rho(\tau) \underset{\tau\to\infty}{\longrightarrow} c\tau^{2H-2}. \tag{3.3}$$

As $H \to 0.5$, we expect rapid decay of the autocorrelation function. On the other hand as $H \to 1$ the autocorrelation function tends to a constant, which means that there is strong correlation for all lags.

The concepts of long-memory and self-similarity are frequently connected, which leads us to consider whether NAR also exhibits self-similarity. In order to investigate this property, we aggregate the time series.

Although the aggregation factor $m = 1000$ is very large, Figures 6 and 7 have a clear resemblance to Figures 2 and 3 respectively. This suggests that the original and aggregated datasets, though different in size, have similar distribution. We consider the benefits of this fact in the next section.

3.2. **Aggregation Similarity.**

The definition of self-similarity, see [2] or [4], declares that the re-scaled, or in our case aggregated, process will be equal in distribution to the original process.

Therefore a self-similar process will satisfy the property

$$\rho^{(m)}(\tau) \approx \rho(\tau), \tag{3.4}$$

where $\rho^{(m)}(\tau)$ denotes the autocorrelation function of the aggregated series $X^{(m)}$ at lag τ.

However, if we examine the autocorrelation function of the NAR time series for various aggregation levels, see Figure 8, we see that the autocorrelation function of the NAR time series satisfies a different property, namely

$$\rho^{(m)}(\tau) \approx \rho(m\tau). \tag{3.5}$$

This is a property which we have examined at long and short timescales, for large and small autocorrelation lags for all caches and we have found this relation to hold without exception for all NAR datasets. This is not a property that we have seen described elsewhere. We therefore give a formal definition as follows.

Definition 3.2 (aggregation-similarity)**.** We call a second-order stationary process with long-memory satisfying condition (3.5) an *aggregation-similar process.*

Since NAR data does not satisfy (3.4), it cannot be self-similar in the usual sense, but nevertheless, aggregation similarity is a very useful property. The two main advantages of aggregation similarity are

(1) Analysis of large datasets is possible using aggregation.
(2) The asymptotic properties of the original data are sped up using aggregation.

Property (1) is particularly useful in the context of the vast cache logs from which the NAR data is extracted. As we have already commented, it would be very difficult to analyse a one week NAR time series, calculated at one second intervals. However, by aggregating the NAR series, we obtain a much smaller, more manageable dataset which bears similar statistical properties to the original.

Property (2) is demonstrated in the next section where we discuss estimation of the long-memory parameter H and a distribution tail index α. Also, estimation of the values of autocorrelation function of an NAR series for a very large maximum lag is time consuming and, for long-term NAR datasets, quickly goes beyond the power of the average computer as the lag increases. However, using the property (3.5), we see that the m aggregated NAR dataset autocorrelation function at lag τ, may be used to estimate the autocorrelation of the original NAR at $m\tau$.

These properties of aggregation-similarity have been used in the estimation of the distribution tail index and the Hurst parameter.

3.3. **Estimation.** It is reasonable to expect that the sample autocorrelations $\hat{\rho}(\tau)$ for sufficiently large τ can be used to estimate H. As we are interested in the strength of the correlation rather than in its direction we use the model $|\hat{\rho}(\tau)| = c\tau^{2H-2}$ to which we apply a non-linear least squares fit. This yields estimates of c and H. An example of an autocorrelation function and this fit is shown in Figures 9 and 10 respectively.

We use aggregated series $X^{(m)}$ in order to speed up the asymptotic behavior of the autocorrelation function. Also, $X^{(m)}$ is a smaller, more manageable size data set. If, as expected, the data is aggregation-similar, then we can use the aggregated series for analysis, knowing that it has similar statistical properties to the original series. Because of the re-scaling property of aggregated similarity, the sample autocorrelation function for say $\tau = 10,000$ in the original series should correspond to the estimate for $\tau = 10$ in the aggregated data set with $m = 1000$. This makes the calculations very fast.

Other methods, presented for example in [2], are linear fit plots using long-memory properties. One is a linear fit to the plot of $\log(\hat{\rho}(\tau))$ against $\log \tau$.

The other method uses the fact that for long-memory processes, the variance of the mean decays with the sample size more slowly than for the uncorrelated data,

$$var(\bar{X}) \approx ak^{2H-2}, \tag{3.6}$$

where $\bar{X} = \frac{1}{k}\sum_{i=1}^{k} X_i$ and $a > 0$. Then the estimate of the slope of the linear regression fit

$$\log var(\bar{X}) \approx \log a + (2-2H)\log k \tag{3.7}$$

gives the estimate of H.

We examine plots of these three methods for comparison in Figure 11. Of these three methods, the non-linear autocorrelation fit is the only one that provides a consistent estimate of H which is relatively invariant to changes in aggregation and maximum lag of the autocorrelation. The consistence is maintained for all NAR datasets considered here. The other two methods are successful for some datasets, but the inconsistencies and variability in some cases make it difficult to trust those estimation methods in general. We therefore select the non-linear autocorrelation fit as our preferred method of initial estimation of H.

The Hurst parameters for NAR data vary between 0.75 and 0.85 for the caches we study. This confirms the long-memory property of the NAR series. Table 5 in Appendix A gives the estimates of H and its standard deviation for all the caches considered in this paper.

Aggregation similarity can also be used when examining the distribution of NAR enabling conclusions to be drawn from smaller aggregated datasets that are applicable to the original NAR data. We consider the distribution of the NAR data in the next section.

3.4. Marginal Distribution of NAR.

The probability distribution of the NAR data is skewed (see Figure 12). However, if we examine the body of the distribution shown in Figure 12, it is clear from the small scale of the vertical axis on the probability histogram that a significant proportion of the data does not feature in the main body of the plot. Since NAR cannot take negative values, this suggests that a reasonable proportion of the data falls into the tail of the distribution. This raises the question 'Is our NAR data heavy-tailed?'

[6] presents theory and estimation methods for the tail index, α, of heavy-tailed probability distributions. We include their definition.

Definition 3.3 (Heavy-Tailed Distribution)**.** A random variable X follows a *heavy-tailed distribution* (with *tail index* α) if

$$P(X > x) \approx cx^{-\alpha} \text{ as } x \to \infty,\ 0 < \alpha < 2. \tag{3.8}$$

where c is a positive constant, the $\approx$ implies that the ratio of the two sides tends to 1 as $x \to \infty$.

This distribution has infinite variance and, if $\alpha \leq 1$, it has infinite mean. Hence, we should not expect this definition be satisfied by our data. Anyway, we followed the computer program in [6] called aest to estimate the tail index α of our data. We have also applied another heavy tail estimator given by [9]. However, in each case the estimator failed to converge or to produce a reasonable tail index estimate. For the aest estimator, the estimate of α was very variable and often greater than 2. We followed the guidelines of [6] to randomize the data to remove correlations, but the estimates were still unreasonable. Many of our Hill estimates failed to converge also preventing us from obtaining a valid estimate of the distribution tail. We therefore attempted to model the NAR variable with several distributions including Pareto, Weibull, and Gamma using non-linear regression. None of these distributions could be well fitted to all of the datasets we considered. Hence we decided to focus on a fit to the tail part of the distribution only.

For this purpose we used an exponential function

$$b \exp(\alpha x), \tag{3.9}$$

where $x > x_0 \geq 0$, $b < 1$ and α denotes a tail index, to model the tail of the distribution, that is the tail of Complementary Cumulative Distribution Function (CCDF) $\bar{F}(x) = P(X > x)$. x_0 is chosen as a point at which the probability mass function attains maximum. The function (3.9) gives a very good fit to the tail of the NAR data for all the caches; an example of a fit is given in Figure 13.

However, the shape of the NAR distributions differ between different caches and different weeks and some of these distributions are easier to fit than others. Figure 14 shows an example of the more awkward shape which is difficult to find an accurate fit for its tail.

We should point out that x_0 can occur at various points in the distribution and, as a result, we ignore anywhere between approximately 5% and 70% of the distribution by omitting the data before the peak. Table 3 in Appendix D gives the estimates of b and α, as well as the percentage of the data used for the tail fit, that is the percentage of all $x > x_0$. The estimates were calculated for 24 five-day weeks for one of the caches considered. The very different values of the ignored part of the sample mass function indicate that the peak or mode of the distribution varies greatly between weeks, so it would be very difficult to find a distribution to fit all data for all timescales. Similar behavior occurs for all the caches we examine.

4. Numerical Calculations

The programs we have produced for this analysis were written in Perl (for data preparation) and in S programming language for statistical analysis and graphs. They can be used in both Windows and Unix environments. The Perl programs included the NAR production program, aggregation programs, and pre-processing programs so as not to allow our calculations to be affected by anomalies in the data. Computer programming language S was used for statistical analysis (in S-plus 2000) to produce plots and programs for Hurst estimation, tail index estimation and other tasks.

The main consideration for the data analysis programs is the size of the data files being processed. Our main NAR calculation program, *totalqd.pl* was designed to process an unlimited number of log files of unlimited size. Each line of data is sequentially processed since, in most cases, it is not possible for the entire data set to be held in memory at one time. The limitations on this program's use are therefore determined by the computer running the program. Similar considerations were made for the simpler aggregation programs we have used.

For statistical analysis of such large data sets, a clear visualisation of the data through plots was important as was the ability to produce our own programs for calculations such as Hurst estimation. S-plus 2000 provided these graphical and programming capabilities. Size was once again an important consideration, and calculation of functions such as the autocorrelations for very large lags was problematic and required both time and a powerful computer. Earlier calculations were performed by a computer with 1GHz processor and 512MB of memory, later calculations used a 3.1GHz machine with 1GB of memory. The programs are available on request from the first author.

5. Conclusions

In this paper we have presented a method to calculate the time series we call Number of Active Requests. This data set exhibits a useful property called aggregation similarity which can be shown experimentally for different timescales and different web caches. The NAR data exhibits long-memory properties, but is not self-similar in the usual sense.

Aggregation similarity permits the use of smaller and therefore more manageable datasets for statistical analysis in place of the original NAR series. This aggregated dataset not only reflects the asymptotic properties of original series' autocorrelation function, but also illustrates the approximate behavior of the original data. This allows conclusions to be drawn on the original data from the aggregated data.

Aggregation similarity has been used to estimate the Hurst parameter for the NAR data. It is well suited to the non-linear estimation method we have used.

Analysis of the distribution of NAR data has also been made. This revealed a skewed distribution which is not heavy-tailed. We have shown that the tail of the NAR distribution, though variable in the proportion of data it represents, can be well approximated by the exponential function.

We therefore present NAR as a long-memory time series which is not self-similar and does not have a heavy tailed distribution. However, it does exhibit aggregation similarity which is a very useful property for analysis of large data sets.

In this paper we have illustrated the calculations with graphs obtained for various caches. They well represent all the caches we studied.

NAR can also be used to study the Quality of Service (QoS) provided by a web cache. A series which is prone to more rapid upward than downward bursts might indicate a web cache that does not provide a good service at busy times since its service is not as fast as the arriving requests. Conversely, a web cache, such as those studied here, are prone to rapid bursts of activity, followed by corresponding downward bursts which indicate that the cache, though prone to rapid activity increase, is able to manage these increases in workload.

The properties of NAR presented in this paper make it a very good measure of internet traffic. Its further analysis may show some traffic properties, such as business and burstiness. It may also be used for cache performance comparison. These aspects of this new variable will be presented in our future work.

References

[1] M. Barenco. A simple stochastic model for long-range dependence. Preprint, 2001.

[2] J. Beran. *Statistics For Long-Memory Processes.* Chapman and Hall, New York, 1994.

[3] G. Bilchev, C. Roadknight, I. Marshall, and S. Olafsson. WWW cache modelling toolbox. *Proceedings of the 4th International Web Caching Workshop,* San Diego, California, March 31 - April 2 1999, 1999.

[4] D. R. Cox. Long-range dependence: a review. *Statistics: An Appraisal, Proceedings 50th Anniversary Conference, Iowa State Statistical Library*, pages 55–74, 1984.

[5] M. E. Crovella and A. Bestavros. Explaining world wide web traffic self-similarity. *Technical Report, Tr-95-015 Boston University Computer Science Department*, 1995.

[6] M. E. Crovella and M. S. Taqqu. Estimating the heavy tail index from scaling properties. *Methodology and Computing in Applied Probability*, **1**:1–21, 1999.

[7] C. R. Cunha, A. Bestavros, and M. E. Crovella. Characteristics of WWW client-based traces. *Technical Report TR-95-010, Boston University, July 1995*, 1995.

[8] C. W. J. Granger and R. C. Joyeux. An introduction to long-range time series models and fractional differencing. *Journal of Time Series Analysis,*, **1**:15–29, 1980.

[9] B. M. Hill. A simple general approach to inference about the tail of a distribution. *The Annals of Statistics*, **3**:1163–1174, 1975.

[10] J. R. M. Hosking. Fractional differencing. *Biometrika*, **68**:165–176, 1981.

[11] J. S. Marron, F. Hernandez-Campos, and F. D. Smith. Mice and elephants visualization of internet traffic. In *Proceedings of Compstat 2002,* Berlin, August 2002, 2002.

[12] I. Marshall and C. Roadknight. Linking cache performance to user behaviour. *Computer Networks and ISDN Systems,*, **30**:2123–2130, 1998.

[13] B. Ryu, D. Cheney, and H. Braun. Internet flow characterization: Adaptive timeout strategy and statistical modelling. In *Proceedings of Passive and Active Measurement Workshop (PAM2001),* Amsterdam, April 2001, 2001.

[14] W. Shi, R. Wright, E. E. Collins, and V. Karamcheti. Workload characterization of a personalized web site and its implications for dynamic content caching. *New York University Technical Report TR2002-829*, 2002.

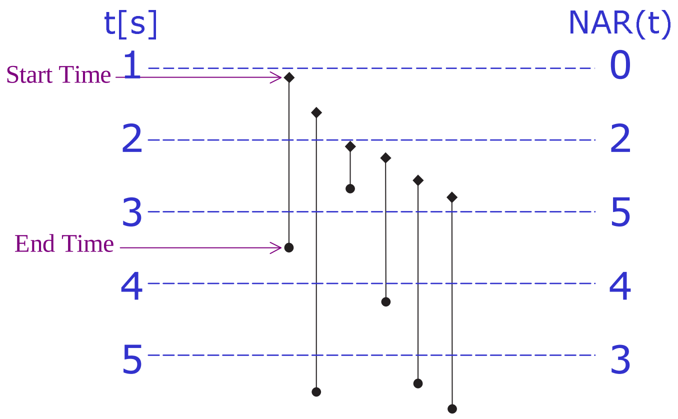


FIGURE 1. **Diagram of Number of Active Requests Calculation. End Time - Start Time = Elapsed Time**

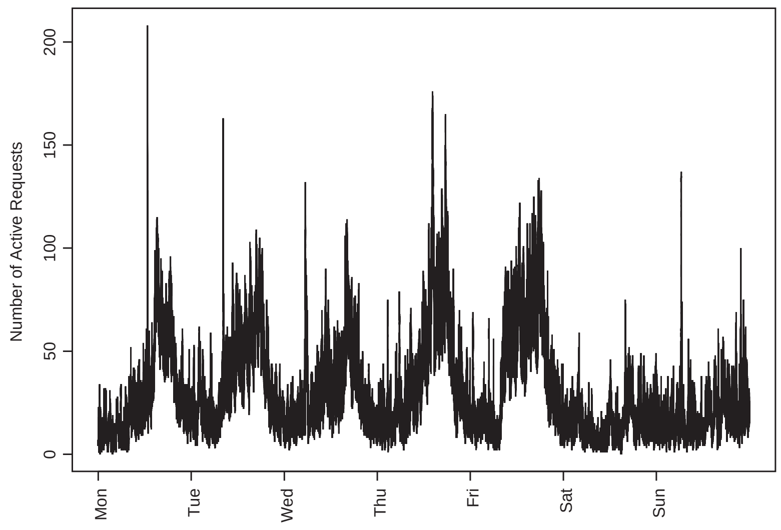


FIGURE 2. **NAR Plot for NLANR rtp cache 7-13 January 2002**

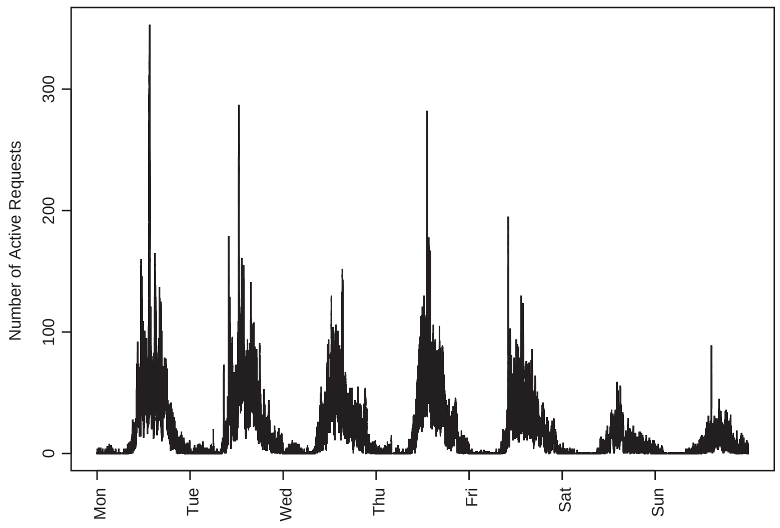


FIGURE 3. **NAR Plot Queen Mary cache 7-13 January 2002**

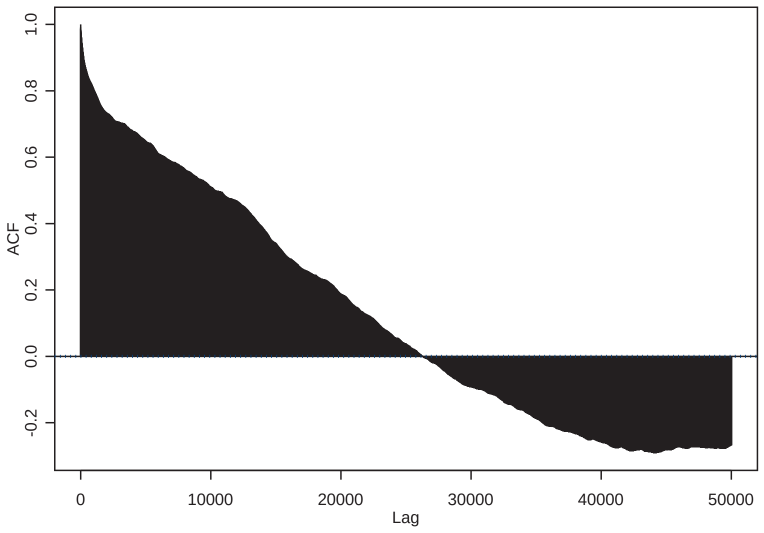


Figure 4. **NAR Correlogram for NLANR rtp cache, 7-11 January 2002, maximum lag = 50000**

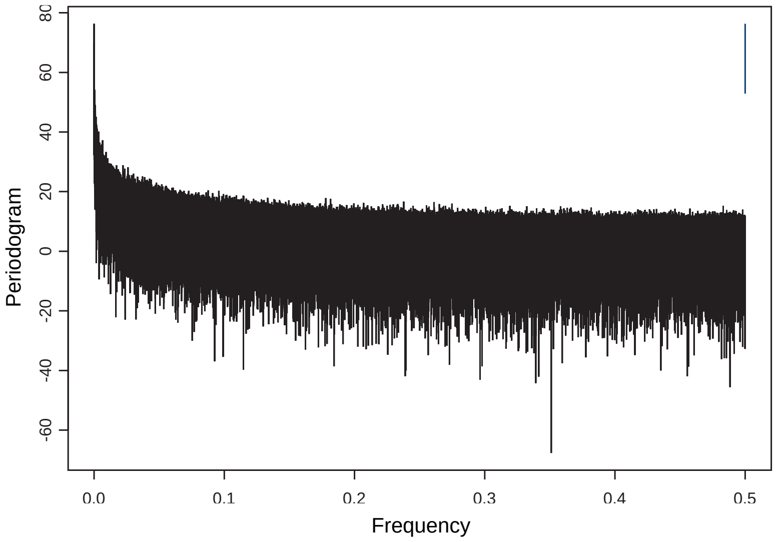


Figure 5. **Periodogram plot for NLANR rtp cache, 7-11 January 2002**

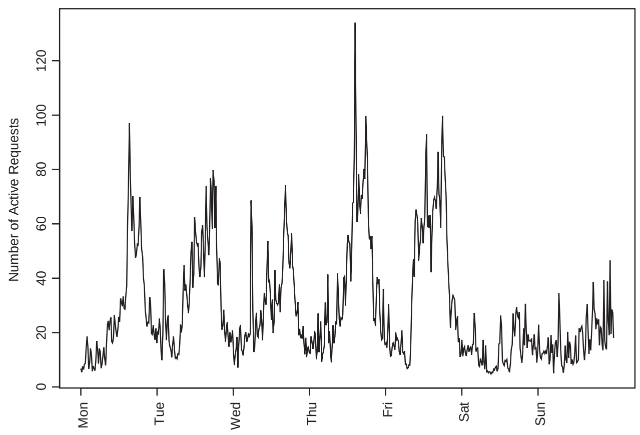


Figure 6. **Aggregated *NAR* Plot, m=1000, for NLANR rtp cache, 7-13 January 2002**

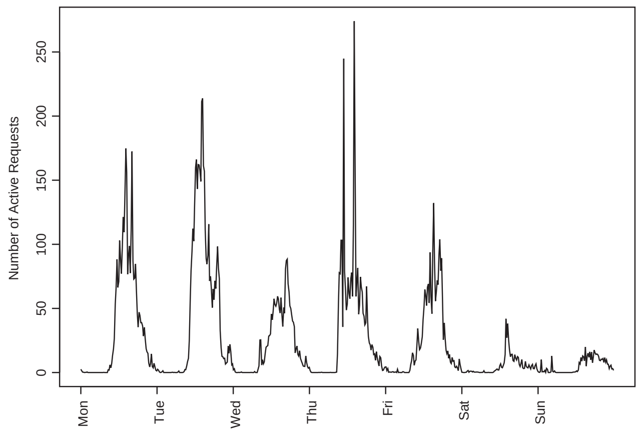


Figure 7. **Aggregated *NAR* Plot, m=1000, for Queen Mary cache, 7-13 January 2002**

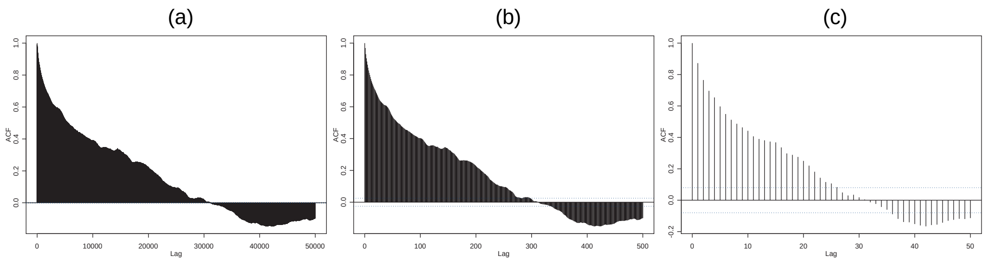


FIGURE 8. **Correlogram of** NAR **for NLANR sv cache, 15-19 January 2002.** (a) m**=1, maximum lag = 50,000, (b)** m**=100, maximum lag = 500, (c)** m**=1000 maximum lag = 50**

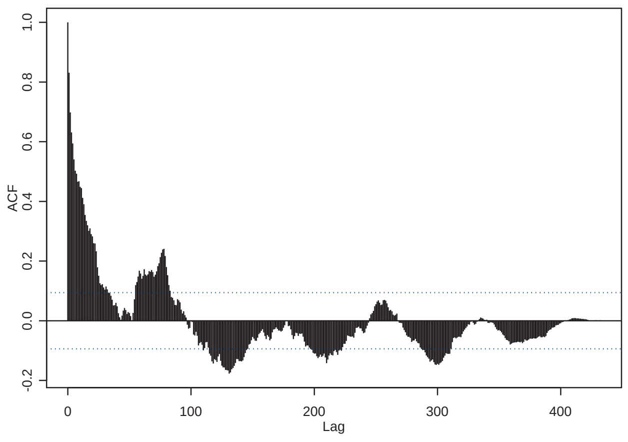


Figure 9. **Autocorrelation plot for NLANR pa 15th to 19th April 2002**

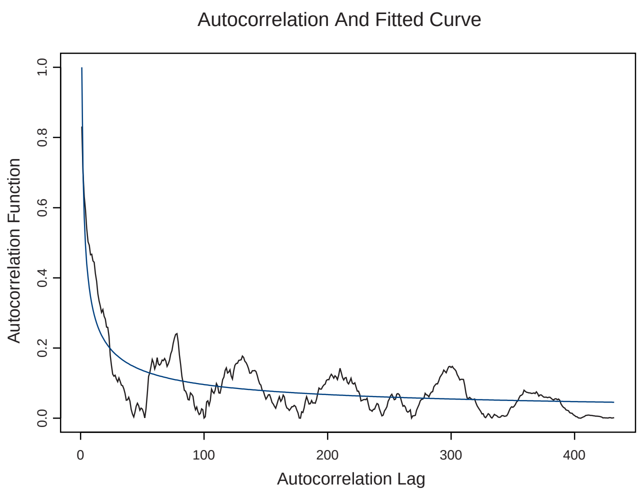


FIGURE 10. **Example of Non-linear fit to autocorrelation function NLANR pa, 15-19 April 2002**

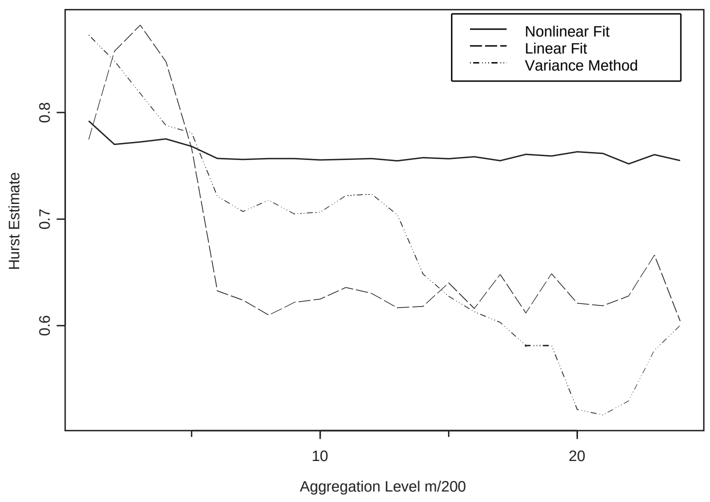


Figure 11. **Comparrison plots of three Hurst estimation methods for NLANR bo1 14th-20th January 2002**

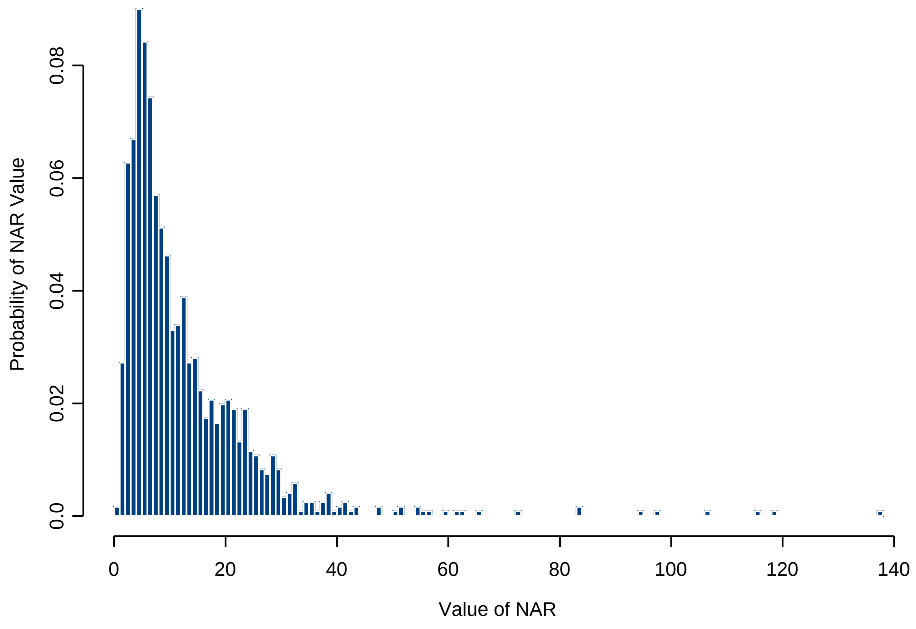


FIGURE 12. **Sample Probability distribution for NLANR bo1 cache, 21-25 January 2002**

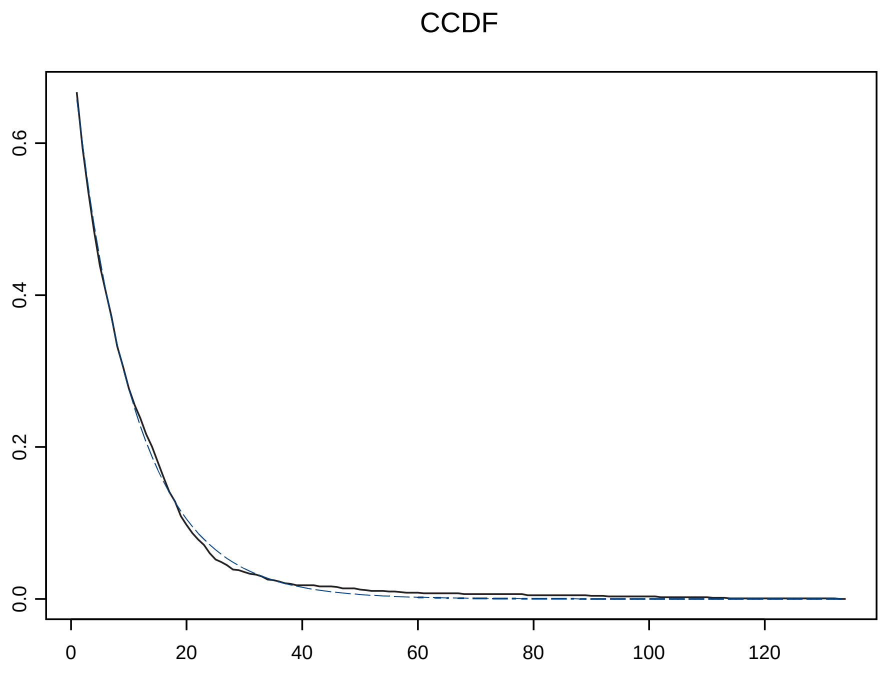


Figure 13. **Sample CCDF and the exponential Fit, dotted line are fitted values; NLANR bo1 cache, 21-25 January 2002**

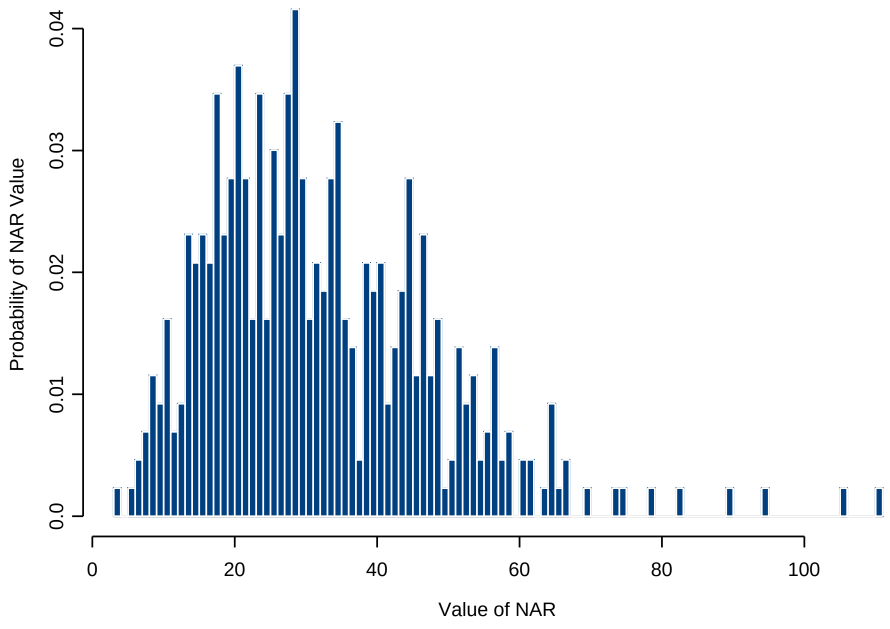


FIGURE 14. **Example *NAR* data (NLANR sd cache, 10-14 Feb 2002) a data set which is difficult to fit a distribution tail to.**

Appendix A. Tables

Cache	**Average Requests Per Day**
Queen Mary	826,044
Essex	2,095,534
NLANR **bo1**	179,853
NLANR **bo2**	197,849
NLANR **pa**	399,697
NLANR **pb**	1,004,530
NLANR **rtp**	1,618,619
NLANR **sd**	625,004
NLANR **sj**	278,335
NLANR **stp**	127,772
NLANR **sv**	1,795,244

Table 1. **Average number of requests per day for each cache**

Cache Name	**Start of First Request**	**End of Last Request**
Queen Mary	21/10/2001	26/5/2002
Essex	14/1/2002	12/6/2002
NLANR **bo1**	1/1/2002	22/6/2002
NLANR **bo2**	1/1/2002	16/4/2002
NLANR **pa**	1/1/2002	28/5/2002
NLANR **pb**	2/1/2002	29/5/2002
NLANR **rtp**	1/1/2002	28/5/2002
NLANR **sd**	1/1/2002	7/4/2002
NLANR **sj**	1/1/2002	21/6/2002
NLANR **startap**	1/1/2002	15/6/2002
NLANR **sv**	1/1/2002	21/3/2002

TABLE 2. **Measurement periods of cache datasets**

Week	$\boldsymbol{b}$	$\boldsymbol{\alpha}$	**% of Distribution Used in Fit**
1	0.95	0.94	0.75
2	1.00	0.94	0.93
3	0.81	0.93	0.51
4	1.00	0.87	0.66
5	1.00	0.88	0.78
6	1.00	0.87	0.74
7	1.00	0.84	0.56
8	1.00	0.88	0.73
9	1.00	0.86	0.68
10	1.00	0.84	0.41
11	1.00	0.83	0.68
12	1.00	0.87	0.46
13	1.00	0.96	0.75
14	1.00	0.93	0.64
15	1.00	0.93	0.69
16	0.67	0.97	0.57
17	0.41	0.96	0.40
18	1.00	0.90	0.36
19	0.70	0.96	0.72
20	1.00	0.89	0.69
21	1.00	0.92	0.50
22	0.73	0.94	0.61
23	1.00	0.93	0.69
24	1.00	0.94	0.73

Table 3: **Exponential fit parameters for NLANR bo1 cache**

Cache	**Location**
Queen Mary	Queen Mary University, London, UK
Essex	Essex University, Wivenhoe Park, Colchester, UK
bo1 and bo2 (bo.us.ircache.net)	Boulder, Colorado
pa (pa.us.ircache.net)	Palo Alto, California
pb (pb.us.ircache.net)	Pittsburgh, Pennsylvania
rtp (rtp.us.ircache.net)	Research Triangle Park, North Carolina
sd (sd.us.ircache.net)	San Diego, California
sj (sj.us.ircache.net)	MAE-West San Jose, California
stp (startap.us.ircache.net)	STARTAP the international connection point in Chicago, IL
sv (sv.us.ircache.net)	Silicon Valley, California (FIX-West)

Table 4. **location of web caches**

Cache	**Mean Hurst Estimate**	**Variance of Hurst Estimate**
Queen Mary	0.836	0.0005
Essex	0.814	0.0008
NLANR **bo1**	0.750	0.0025
NLANR **bo2**	0.764	0.0013
NLANR **pa**	0.750	0.0029
NLANR **pb**	0.750	0.0019
NLANR **rtp**	0.813	0.0012
NLANR **sd**	0.843	0.0015
NLANR **sj**	0.751	0.0044
NLANR **stp**	0.764	0.0018
NLANR **sv**	0.788	0.0014

TABLE 5. **Mean Hurst estimates for caches and their variances**

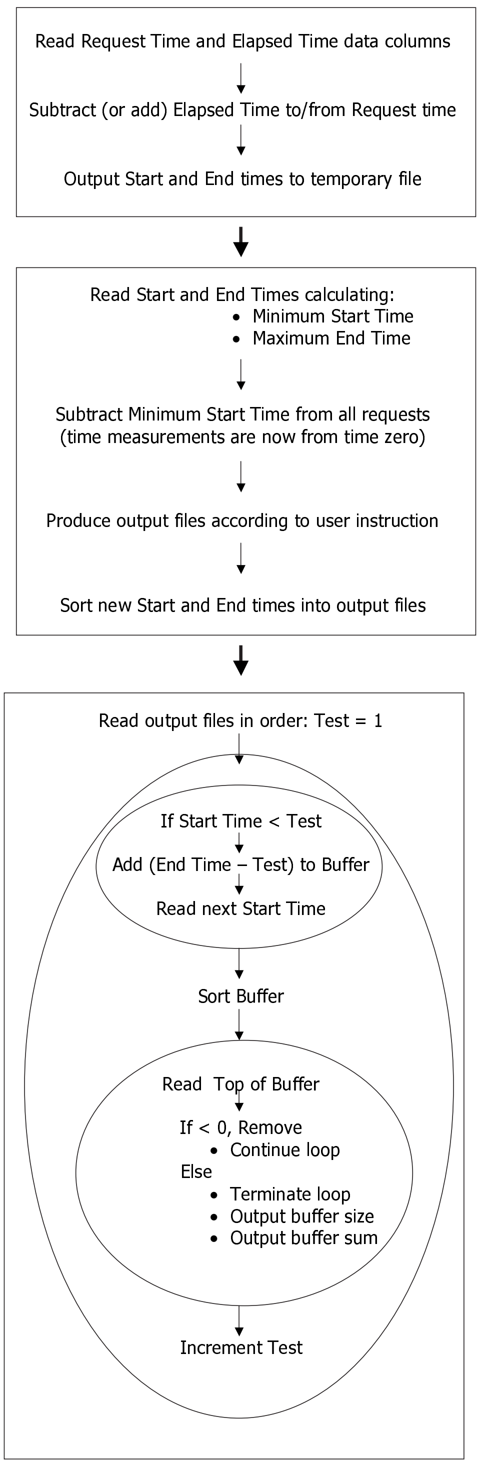
Appendix B. An example of a log file.

Columns are separated by a single space.

```
1018137607.017 314 88.121.141.160 TCP_MISS/403 2676 GET http://slashdot.org/slashdot.rdf - DIRECT/64.28.67.150 text/html
1018137607.165 8 88.121.141.160 TCP_NEGATIVE_HIT/403 2683 GET http://slashdot.org/slashdot.rdf - NONE/- text/html
1018137607.682 168 88.121.141.160 TCP_MISS/403 2676 GET http://slashdot.org/~Nugget94M/ - DIRECT/64.28.67.150 text/html
1018137607.835 12 88.121.141.160 TCP_NEGATIVE_HIT/403 2675 GET http://slashdot.org/~Nugget94M/ - NONE/- text/html
1018137607.936 253 80.21.198.72 TCP_REFRESH_HIT/200 4738 GET http://www.looksmart.com/h/info/bad_ad.html - DIRECT/64.241.242.202 text/html
1018137608.158 17 80.21.198.72 TCP_MISS/304 259 GET http://mercury.beseen.com/images/chat/heart.gif - DIRECT/209.249.66.57 -
1018137608.731 171 88.121.141.160 TCP_MISS/403 2676 GET http://slashdot.org/~Nugget/ - DIRECT/64.28.67.150 text/html
1018137608.882 11 88.121.141.160 TCP_NEGATIVE_HIT/403 2675 GET http://slashdot.org/~Nugget/ - NONE/- text/html
1018137608.971 89 80.21.198.72 TCP_MISS/304 259 GET http://mercury.beseen.com/images/chat/wink.gif - DIRECT/209.249.66.57 -
1018137609.300 45 80.21.198.72 TCP_MISS/304 260 GET http://mercury.beseen.com/images/chat/stop.gif - DIRECT/209.249.66.57 -
1018137609.339 38 80.21.198.72 TCP_CLIENT_REFRESH_MISS/304 258 GET http://mercury.beseen.com/images/chat/frown.gif - DIRECT/209.249.66.57 -
1018137610.040 164 88.121.141.160 TCP_MISS/403 2676 GET http://slashdot.org/~grub/ - DIRECT/64.28.67.150 text/html
1018137610.192 12 88.121.141.160 TCP_NEGATIVE_HIT/403 2683 GET http://slashdot.org/~grub/ - NONE/- text/html
1018137610.570 63 80.21.198.72 TCP_MISS/304 259 GET http://mercury.beseen.com/images/chat/heart.gif - DIRECT/209.249.66.57 -
1018137610.748 113 80.21.198.72 TCP_MISS/304 260 GET http://mercury.beseen.com/images/chat/rose.gif - DIRECT/209.249.66.57 -
1018137611.047 68 80.21.198.72 TCP_MISS/304 260 GET http://mercury.beseen.com/images/chat/stop.gif - DIRECT/209.249.66.57 -
1018137611.158 32 80.21.198.72 TCP_MISS/304 259 GET http://mercury.beseen.com/images/chat/wink.gif - DIRECT/209.249.66.57 -
1018137611.208 160 88.121.141.160 TCP_MISS/403 2676 GET http://slashdot.org/~Leto2/ - DIRECT/64.28.67.150 text/html
1018137611.360 6 88.121.141.160 TCP_NEGATIVE_HIT/403 2683 GET http://slashdot.org/~Leto2/ - NONE/- text/html
1018137611.791 23 80.21.198.72 TCP_CLIENT_REFRESH_MISS/304 258 GET http://mercury.beseen.com/images/chat/frown.gif - DIRECT/209.249.66.57 -
1018137611.863 59 80.21.198.72 TCP_MISS/304 259 GET http://mercury.beseen.com/images/chat/smile.gif - DIRECT/209.249.66.57 -
1018137611.900 36 80.21.198.72 TCP_MISS/304 327 GET http://www.beseen.com/smallchat.gif - DIRECT/209.249.66.60 -
1018137612.145 171 88.121.141.160 TCP_MISS/403 2676 GET http://slashdot.org/~BovineOne/ - DIRECT/64.28.67.150 text/html
1018137612.296 12 88.121.141.160 TCP_NEGATIVE_HIT/403 2683 GET http://slashdot.org/~BovineOne/ - NONE/- text/html
1018137612.402 53 80.21.198.72 TCP_MISS/304 273 GET http://bsads.looksmart.com/plainads/blank.gif - DIRECT/64.241.242.203 -
1018137612.564 236 80.21.198.72 TCP_REFRESH_HIT/304 254 GET http://17tahun.com/html/cerita/Konvensional/more27.html - DIRECT/209.103.167.115 -
1018137613.127 170 88.121.141.160 TCP_MISS/403 2676 GET http://slashdot.org/~dbaker/ - DIRECT/64.28.67.150 text/html
```

Appendix C. Perl Program

Explanatory diagram of Perl totalqd.pl program. The circles and ovals indicate loops in the program structure.


Read Request Time and Elapsed Time data columns
Subtract (or add) Elapsed Time to/from Request time
Output Start and End times to temporary file
Read Start and End Times calculating:
• Minimum Start Time
• Maximum End Time
Subtract Minimum Start Time from all requests (time measurements are now from time zero)
Produce output files according to user instruction
Sort new Start and End times into output files
Read output files in order: Test = 1
If Start Time < Test
Add (End Time – Test) to Buffer
Read next Start Time
Sort Buffer
Read Top of Buffer
If < 0, Remove
• Continue loop
Else
• Terminate loop
• Output buffer size
• Output buffer sum
Increment Test