S-Net: A Software System for Analyzing Packet Header Databases

Jin Cao, William S. Cleveland, and Don X. Sun

Abstract— **S-Net is a software system designed for analyzing large databases of Internet packet headers. S-Net runs on Unix, and analysis is carried out using the S language for visualization and data analysis. There are three goals in its design: (1) to allow, despite the large size of databases, comprehensive analysis — detailed analysis down to the level of individual observations, together with analysis by summaries of behavior that aggregate many observations; (2) to provide an extensible environment so that analysts can readily tailor methods to the specific objectives of their analyses; and (3) to provide an implementation based on public domain software, making S-Net readily accessible to network researchers. S-Net is available at http://cm.bell-labs.com/stat/InternetTraffic/S-Net.**

I. Introduction

There have been many useful tools developed by network researchers for analyzing Internet traffic data. A comprehensive survey is available at CAIDA [5]. Most of these available tools provide summary reporting for variables such as packet counts, byte counts, and flow characteristics.

To provide a more flexible, comprehensive analysis environment for packet header data, we have developed S-Net [7]. It runs on Unix and uses C, Perl, and the S language for visualization and data analysis ([1], [9]). The S-Net software consists of database management tools written in C, Perl, Unix commands, and S, and analysis tools written in S. Computationally intensive data processing and database management are carried out using Perl programs, C programs, and Unix commands. Summary and detailed analysis, and less computationally intensive database management, are carried out using S.

A. Goals

Goal 1 in designing S-Net and building database and analysis tools is to provide for the study of large amounts of packet header data, but at the same time enable comprehensive analysis: detailed characterization, visualization, and modeling of the data, down to the level of individual observations, and summary statistics that summarize behavior. One approach to the analysis of large databases is to use only summary analysis: carry out queries that retrieve aggregates of the detailed data, and then analyze the resulting reduced information. While the study of summaries is important, it cannot always succeed in exploiting fully the information in a large database.

Goal 2 is to make comprehensive study flexible. S-Net has a collection of tools that accommodate widespread tasks carried out by network researchers; but in addition, S-Net provides an environment that allows an analyst to readily alter tools or build entirely new ones. In other words, the system is extensible.

Goal 3 is to implement S-Net using public domain software to make it readily accessible to network researchers.

The authors are members of the Mathematical Sciences Research Center, Bell Labs, Murray Hill. Contact E-mail: dxsun@research.bell-labs.com. This research was supported, in part, by DARPA under federal contract No. F30602-00-C-0034.

B. Extensibility

Flexibility of analysis in S-Net is provided by the design of Unix and S, which allow ready extensibility, so the user is not bound by only those tools in the system.

Network researchers are familiar with the design of Unix, and why this allows ready extensibility, but S is less widely known. But in many ways, S, which won the ACM Software System Award for 1999, reflects the Unix programming design. It gives the analyst the flexibility to tailor analyses to the specific questions being asked of the data. S contains many functions written to carry out statistical analysis. They are not compiled, come as source with the system, and are interpreted by the S executive when they are invoked. S-Net includes S functions for carrying out specific methods we have found useful for analysis of header databases, but the analyst is not bound by these routines. All S-Net tools come as source and can be used as is, modified, or ignored by someone who wants to achieve the purpose of the tools in a different way.

S-Net tools can be *out-of-box*, which means S functions are used without modification; they can be *modified out-of-box*, which means simple changes to functions are carried out; or they can be *roll-your-own*, which means major changes are made or wholly new functions are written.

S has another dimension of extensibility. Tools written in C or using Unix system commands can be called from within S-Net by S commands and have the results returned as S objects. So many tools in the above CAIDA survey could be readily incorporated into S-Net.

C. Public Domain

Our implementation uses the Linux version of Unix, and the R version ([14], [13]) of the S language, both in the public domain.

D. Comprehensive Analysis

Comprehensive analysis, detailed and summary, is achieved by five aspects of S-Net: (1) a dataflow that progresses from raw packet header Unix files, to primary and secondary UNIX files, and then to S objects; (2) time blocking with repetitive analysis — break headers from a single monitor into many time blocks, use the same analysis for all blocks, and then combine the results of the analyses; (3) a combination of tools that show the detail in the data and tools that summarize behavior; (4) mechanisms for taking into account, the immense impact that the magnitude of statistical multiplexing has on the properties of many traffic variables; (5) multipage visualizations using trellis display in S, a visualization framework that allows the analyst to create a single display with perhaps hundreds of pages and tens of panels per page. Collectively, these aspects amount to a strategy for approaching the analysis of packet header data.

E. Contents of Paper

Extensibility and a public domain-system came easily; we simply used languages and systems with high extensibility and chose public-domain versions of the languages and systems. The effort in S-Net was building a system that allowed comprehensive analysis. The purpose of this article is to convey the components of S-Net and how their design enables comprehensive analysis of packet header data. In doing so we show, as examples, the use of a few S-Net tools in two major studies of Internet traffic variables. Sections II and III present the examples, Section IV discusses analysis strategy for comprehensive analysis. Section V lists out-of-box tools and variables. Section VI discusses hardware, implementations, and databases.

II. FSD: Open-Loop Internet Packet Traffic Modeling

In a series of studies using S-Net, we investigated the statistical properties, including the long-range dependence (LRD), of four packet traffic variables: packet arrival times, inter-arrival times, packet counts, and byte counts ([2], [3], [4]). (We omit the discussion of the byte counts here.) We discovered that the magnitude of statistical multiplexing had a dramatic effect on the statistical properties. We found that very simple statistical models, FSD models, which consist of a fractional ARIMA plus white noise, do an excellent job of describing the properties and how they change with the magnitude. The FSD models can be used for open-loop generation of packet arrivals and sizes.

In one study, we analyzed 2526 packet traces, 5 min or 90 in duration, from 6 monitors [3]. Table I gives information about the traces. COS and AIX are from the NLANR database [10], NZIX is from the NZIX-II database [11], and BELL is from the Bell Labs database [6]. The six monitors get packets from 15 interfaces. For NLANR, each interface has its own monitor. At BELL, two local interfaces, the two directions on the wire, are multiplexed because transmission is half-duplex. At NZIX, monitoring is carried out on a port that receives multiplexed mirrored streams from the 9 interfaces on the switch. For these two monitors, we study the full stream, but also the individual interface streams because queueing is minor in both cases. (2 interfaces for NZIX are ignored because of very low loads.) The final result is 15 interfaces, including the two multiplexed monitor streams. The table shows the time interval for each trace, the link speed, the link layer protocol, and the mean of the log base 2 of the average number of active connections for the traces in the group.

To study the data, we used many S-Net visualization tools, detailed and summary. First, individual measurements of the packet traffic variables are displayed for each trace. Each detailed tool produces one display per trace, which means 2526 displays for the tool. How we managed this is described in Section IV-E. Then each summary metric for all traces are displayed together in one plot.

A. Notation

We will introduce mathematical notation to explain the tools used in the study.

TABLE I

Trace Group: name including length of trace • Number: number of traces • Link: speed and link layer protocol • $\log_2 c$: log base 2 average number of active connections

Trace Group	Number	Link	$\log_2 c$
AIX1(90sec)	23	622mbps PoS	13.09
AIX2(90sec)	23	622mbps PoS	13.06
COS1(90sec)	90	156mbps ATM	10.83
COS2(90sec)	90	156mbps ATM	10.81
NZIX(5min)	100	100mbps Eth	10.75
NZIX7(5min)	100	100mbps Eth	9.60
NZIX5(5min)	100	100mbps Eth	8.66
NZIX6(5min)	100	100mbps Eth	7.85
NZIX2(5min)	100	100mbps Eth	7.32
NZIX4(5min)	100	100mbps Eth	7.17
BELL(5min)	500	100mbps Eth	6.97
NZIX3(5min)	100	100mbps Eth	6.54
BELL-IN(5min)	500	100mbps Eth	5.98
BELL-OUT(5min)	500	100mbps Eth	5.94
NZIX1(5min)	100	100mbps Eth	4.42

Packet Variables

We begin with packet variables, which have many values per trace. Consider packets arriving on a link for a single trace where $j = 1$ corresponds to the first arriving packet, $j = 2$ to the second, and so forth. We suppose that the timestamps occur at the beginning moment of transmission. For packet counting purposes we divide the time block of each trace into sub-blocks of equal length. The length of the time blocks in this presentation is 100 ms.

q_j = the packet sizes, headers plus data but not link layer encapsulation.

q_j^{cap} = the encapsulated packet sizes, headers plus data plus link layer encapsulation.

a_j = the arrival times.

$t_j = a_{j+1} - a_j$ = the inter-arrival times. $t_j^* = \log t_j$.

p_i = the number of a_i falling into the ith sub-block. $p_i^* = \log p_i$.

The j-th packet, whose encapsulated size is q_j^{cap}, is transmitted during the beginning of the interval from a_j to a_{j+1}, whose length is t_j.

$t_j^{min} = q_j^{cap}/link.speed$ = the time to transmit packet j. Suppose there is no measurement error. We have $t_j \geq t_j^{min}$. If the $(j+1)$-st packet is back-to-back with the j-th, $t_j = t_j^{min}$.

$b_{1j} = q_j^{cap}/t_j$ = the one-packet bandwidth. This variable is the highest resolution bandwidth measurement.

Violations of the inequality $t_j \geq t_j^{min}$ can occur from measurement error resulting in $t_j < t_j^{min}$. Of course, there can be small violations due to timer resolution, res, so to assess non-resolution error we work with $t_j^{min} \pm res$.

Packet Summary Variables

We now turn to packet summary variables, which have one value per trace.

ψ_k = the percent of packets that are back to back with k or more following packets. Packet j is taken to be back-to-back with packet $j+1$ if $t_j < t_j^{min} + res$.

ν = the coefficient of variation of the p_i, the standard deviation divided by the mean.

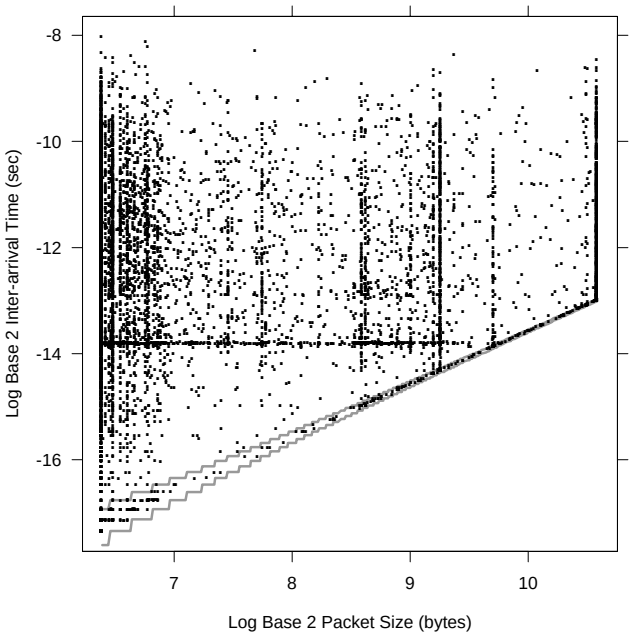

Fig. 1. Log inter-arrival time is graphed against log encapsulated packet size for an NZIX(5min) trace. This out-of-box tool is described in Section II-B.

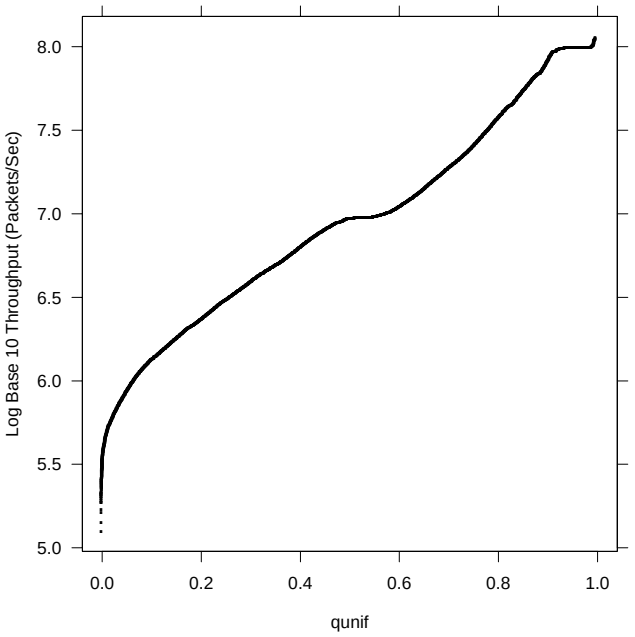

Fig. 2. Quantiles of log one-packet bandwidth are graphed against uniform quantiles for the same trace as in Figure 1. This out-of-box S-Net tool is described in Section II-B.

$c =$ the average number of active connections, where we average across all times in each trace. $c^* = \log(c)$.

$\tau_1 =$ the one-step entropy of a time series and is applied to three time series — q_j, $t_j^{1/6}$, and $\log(p_i)$ — where each series is normalized to have mean 0 and variance 1. τ_1 is the error of prediction of the series from its infinite past. $0 < \tau_1 \leq 1$. If $\tau_1 = 1$ the series is independent (uncorrelated); if τ_1 is close to 0, the series is highly dependent in the sense that it can be predicted reliably from the past.

B. Time-Stamps and Back-To-Back Packets

Figure 1 applies one visualization tool of S-Net to one trace from the NZIX(5min) trace group. For this tool, $\log(t_j)$ is graphed against $\log(q_j^{cap})$ for each packet. There are two curves at the lower envelope of the plot: $\log(t_j^{min} \pm res)$ vs. $\log(q_j^{cap})$. Any point below the lower curve, an impossible value if there were no measurement error aside from timer resolution, occurs because of delays in writing the timestamps. Figure 2 shows another tool, a log one-packet bandwidth quantile plot. On the plot, the i-th largest log one-packet bandwidth is graphed against $(i - 0.5)/number.of.observations$. Values of the one-packet bandwidth greater than $line.speed$ (100 mbps in this case) are the result of measurement error.

In Figure 1, no point is below the lower boundary, so the measurements pass this test with flying colors. In Figure 2, a small fraction of points are slightly above 100 mpbs. NZIX-II was the top monitor in our study with virtually no points below the lower boundary on the inter-arrival-size plot. This is no surprise given the immensely high quality of the measurement operation. The remaining monitors did show violations, the effects of monitors getting behind, but the effects were minor.

Queueing at the link input causes packets to be back-to-back, which in turn causes, if the amount of queueing is large enough, changes in the statistical properties of the t_j and the p_i. We explored values of ψ_k, for $k = 1$, 3, and 5. Figure 3 is a trellis display of ψ_1 against c^* for each of the 15 trace groups. There are 15 panels, one per trace group, and each point on a panel

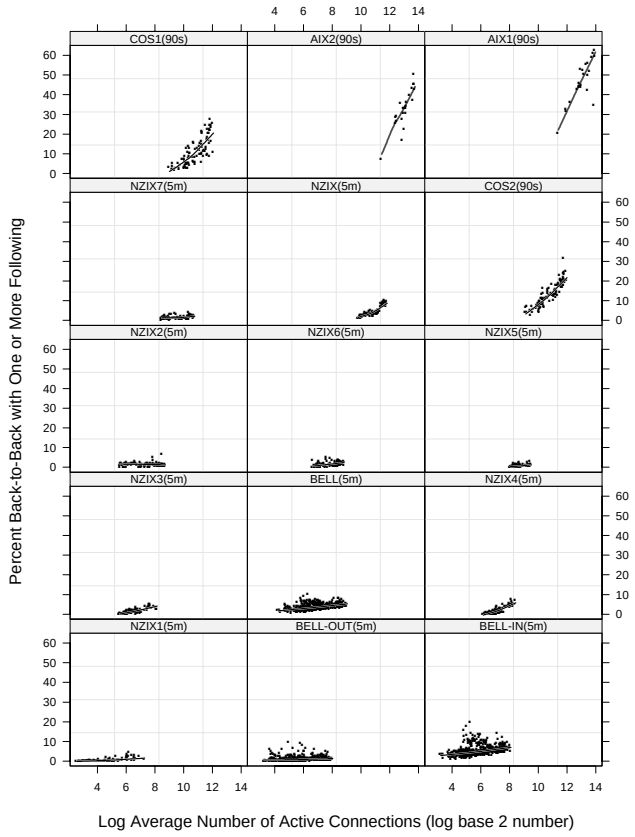

Fig. 3. The percent of packets back-to-back with one or more following packets is plotted against log average number of active connections. This modified out-of-box S-Net detail tool is described in Section II-B.

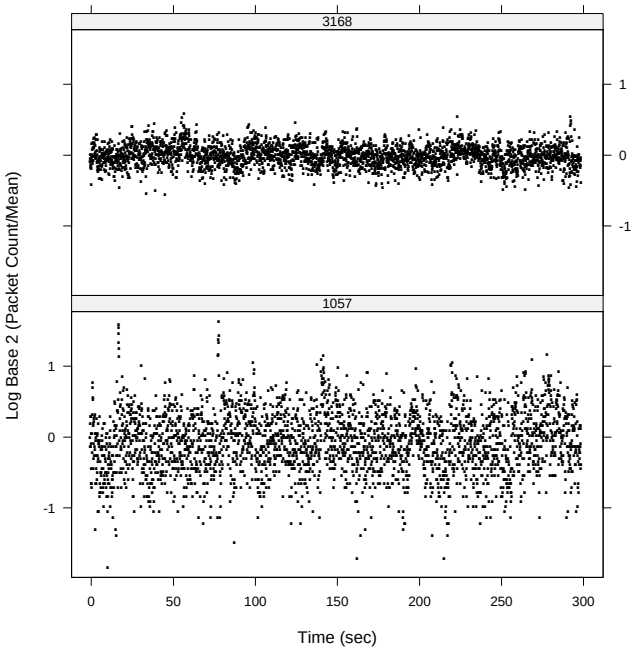

Fig. 4. Log 100 ms packet counts divided by their mean are graphed against time for two NZIX(5min) traces. This modified out-of-box S-Net tool is described in Section II-C.

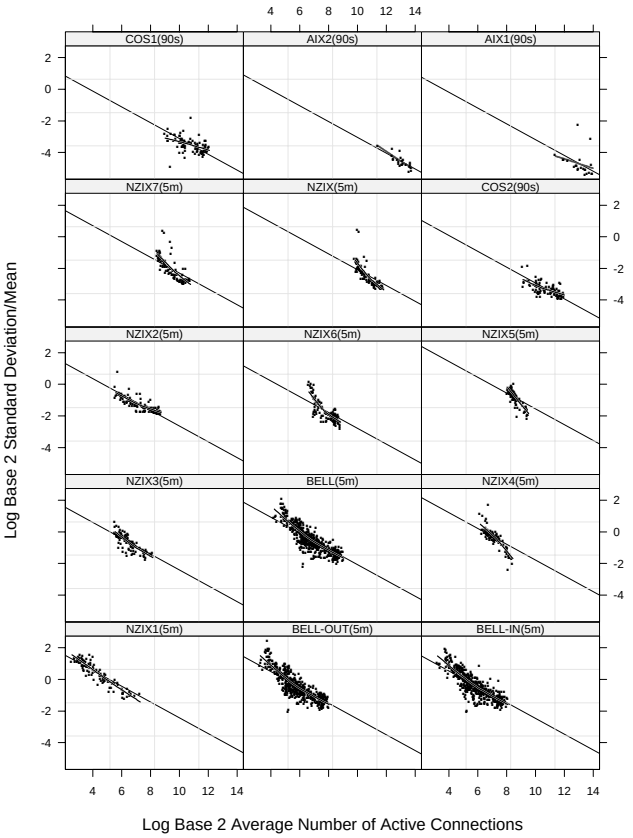

Fig. 5. Log coefficient of variation (standard deviation divided by mean) of packet counts is plotted against log average number of active connections. This modified out-of-box S-Net tool is described in Section II-C.

displays ψ_1 against c^* for one trace in the group. The panels are ordered, left to right and then bottom to top by the trace group means of c^*, shown in column 4 of Table I. For all trace groups, ψ_1 increases with c^* as expected. The values of ψ_1 vary substantially from panel to panel reflecting very different amounts of one-or-more packet queueing among the trace groups.

C. Statistical Variability of Packet Counts

The mathematical theory of multiplexed point processes stipulates that the statistical variability of the p_i should decrease with the trace summary measure, c, the average number of active connections. Our trace summary measure of the amount of statistical variability will be the coefficient of variation, ν, the standard deviation of the p_i divided by the mean. Theory says ν should decline like $c^{-0.5}$ until non-trivial queueing sets in. This means that the variation in $p_i/\bar{p}$ for a trace, where $\bar{p}$ is the mean of the p_i, should decline across traces as c increases.

To check the validity of the theory, the tool in Figure 4 graphs the log base 2 of 100 ms $p_i/\bar{p}$ for two traces from the NZIX(5min) trace group. For the bottom panel, c, shown in the strip label at the top, is 1057 connections, and for the top panel, it is 3168 connections. Clearly the variability is much greater in the bottom panel than in the top.

Figure 5 graphs $\log(\nu)$ against $c^* = \log(c)$ for each of the 15 trace groups. The lines are the best fitting (least-squares) line with slope -0.5. If the theory applies, the pattern of the points should follow the line. In fact, agreement is quite good, surprising for trace groups COS and AIX because Figure 3 shows their values of ψ_1, the percent of back-to-packet packets with one or more following, gets quite large, which should cause ν to stabilize; it is likely that for packet counts with intervals lengths smaller than 100 ms, it would do so.

D. Long-Range Dependence (LRD)

The mathematical theory of multiplexed point processes stipulates that the LRD of the packet inter-arrivals, t_j, and the packet sizes, q_j, should always be present, but should have less and less influence as the magnitude of multiplexing increases; c, the average number of active connections, is a summary measure of the magnitude for each trace. The theory also stipulates that the LRD of the packet counts, p_i, should remain the same with c until the amount of input queueing becomes non-trivial; but because the statistical variability of the p_i decreases with c, the LRD becomes less and less salient because the magnitude of the bursts due to LRD decreases. Let $\tilde{q}_j$, $\tilde{t}_j$, and $\tilde{p}_i$ be the normalized time series of q_j, $t_j^{1/6}$, and $\log(p_i)$, with mean 0 and variance 1. We use the power spectrum and the one-step entropy summary measure, τ_1, measure to study the LRD of $\tilde{q}_j$, $\tilde{t}_j$, and $\tilde{p}_i$.

Figure 6 graphs the one-step entropy summary measure, τ_1, against $c^* = \log(c)$, for the $\tilde{t}_j$. Each panel shows the values for one trace group. Figures 7 and 8 do the same for the $\tilde{q}_j$ and the $\tilde{p}_i$, respectively. The theory holds up well on these plots. For the $\tilde{q}_j$, τ_1 increases toward 1. For the $\tilde{p}_i$ there is no consistent pattern. For the $\tilde{t}_j$, τ_1 increases toward 1 for most trace groups except AIX1(90sec), AIX2(90sec), NZIX7(5min), and NZIX(5min), where it decreases a bit. The tool coming next, the power spectrum, shows the decline is due to an increase of very short-range dependence and not an increasing LRD, which

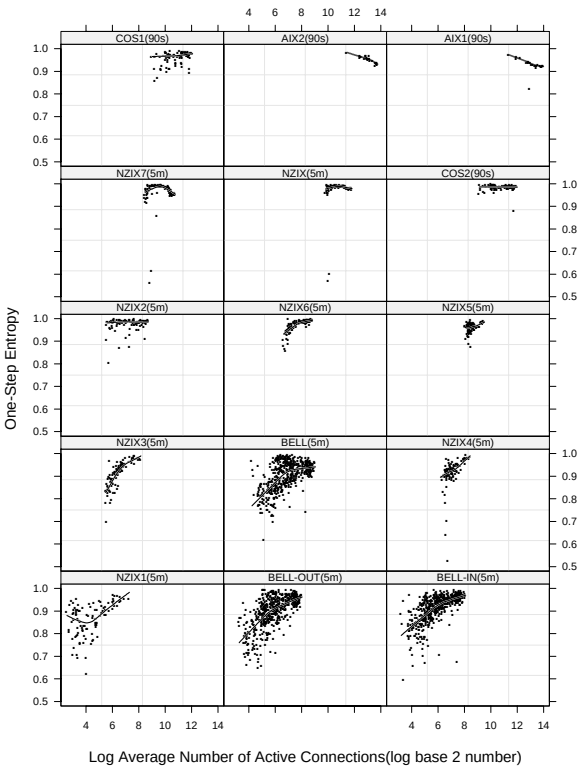

Fig. 6. One-step entropy for sixth root inter-arrival is plotted against log average number of active connections. This modified out-of-box S-Net tool is described in Section II-D.

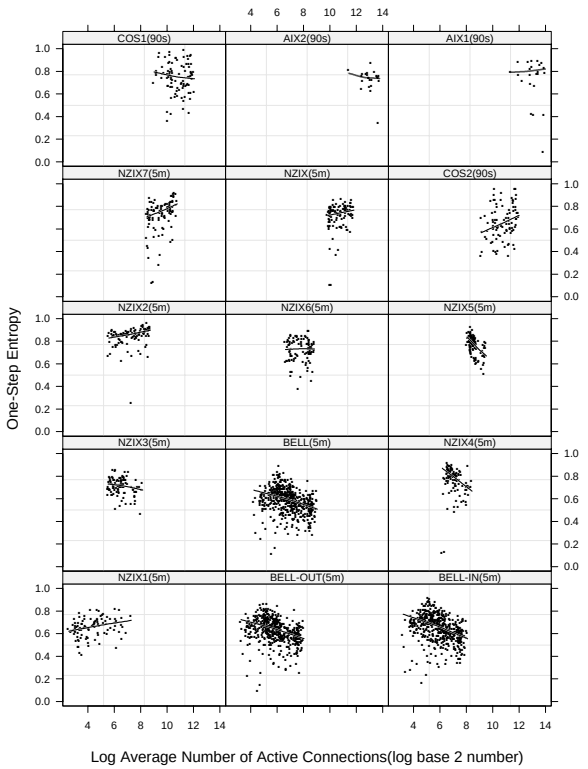

Fig. 8. One-step entropy for log 100 ms packet count is plotted against log average number of active connections. This modified out-of-box S-Net tool is described in Section II-D.

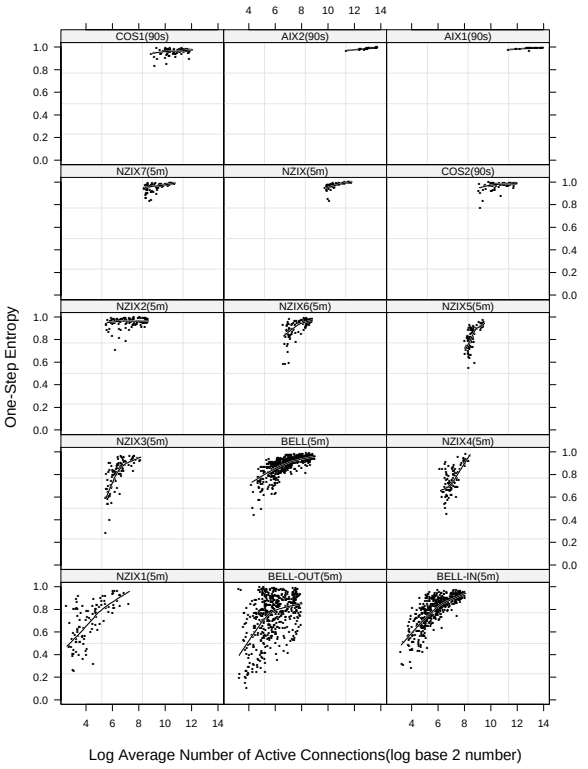

Fig. 7. One-step entropy for the packet size is plotted against log average number of active connections. This modified out-of-box S-Net tool is described in Section II-D.

is very small in influence for these trace groups.

The nature of the long-range dependence of the three time series $\tilde{q}_j$, $\tilde{p}_i$, and $\tilde{t}_j$ is persistence: a positive autocorrelation function that falls of very slowly. The persistence results in power spectra that tend to infinity at frequency 0.

Figure 9 graphs 3 power spectrum estimates for $\tilde{q}_j$ from each of the 15 trace groups. There are 45 panels on the display broken into 15 sets of three panels, one set per trace group. Going through the traces groups from left to right and then bottom to top, the order is the same as that of the groups in Figure 7. Within each set of three panels, the value of c^* for which the spectrum is estimated increases left to right and is shown at the top middle of each panel. In the figure, the general pattern for smaller values of c^* is very large power at low frequencies, a rapid rise as the frequency tends to 0, and an overall monotone decrease in power as the frequency increases from 0 cycles/packet to 0.5 cycles/packet. This is a result of the persistent long-range dependence in the size time series. But this pattern changes with c^* as follows: as c^* increases, the fraction of low-frequency power decreases and the fraction of high-frequency power increases; this means the long-range dependence decreases, and the estimates tend toward the constant spectrum of white noise. This happens for each trace group individually, going through the panels for each row. This also happens across trace groups as well; the groups toward the top with higher c^* have less long-range dependence than those at the bottom with lower

Figure 10 graphs the 3 power spectrum estimates for $\tilde{t}_j$ using the same method as in Figure 9. For each trace group other than

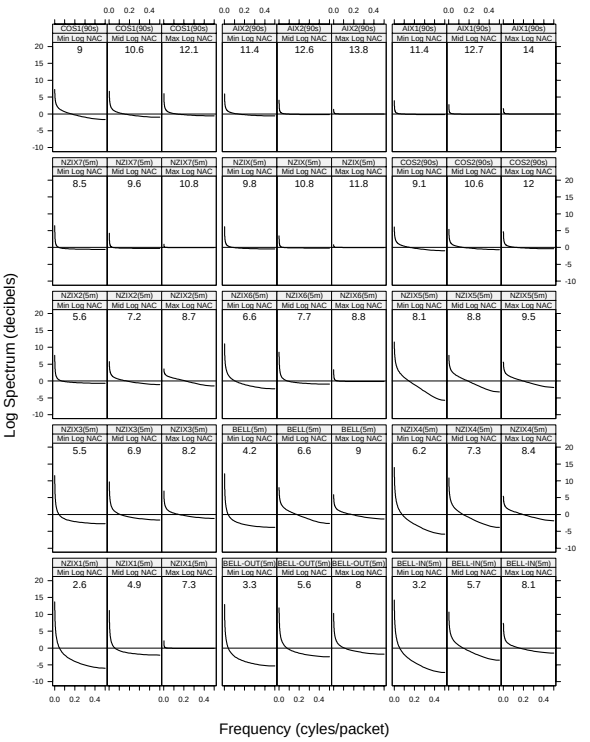

Fig. 9. For each of the 15 trace groups, the power spectrum for packet size is plotted against frequency for a set of 3 values of the number of active connections. This roll-your-own S-Net tool is described in Section II-D.

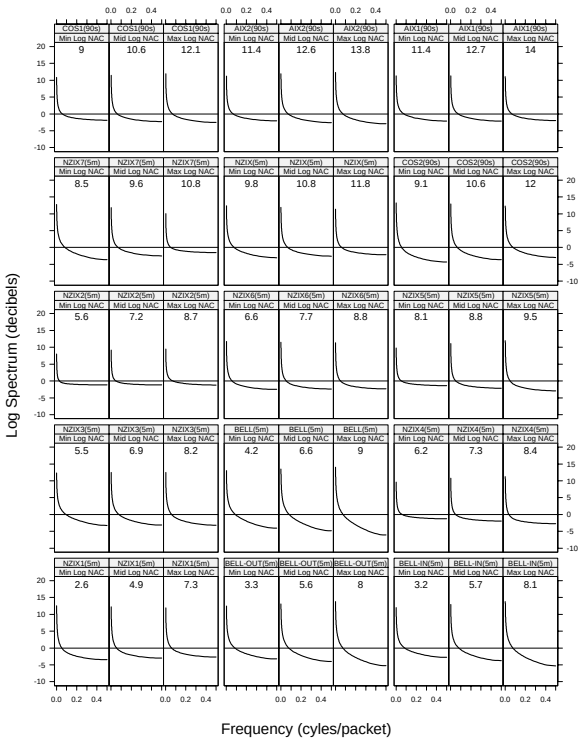

Fig. 11. For each of the 15 trace groups, the power spectrum for packet count is plotted against frequency for a set of 3 values of the number of active connections. This roll-your-own S-Net tool is described in Section II-D.

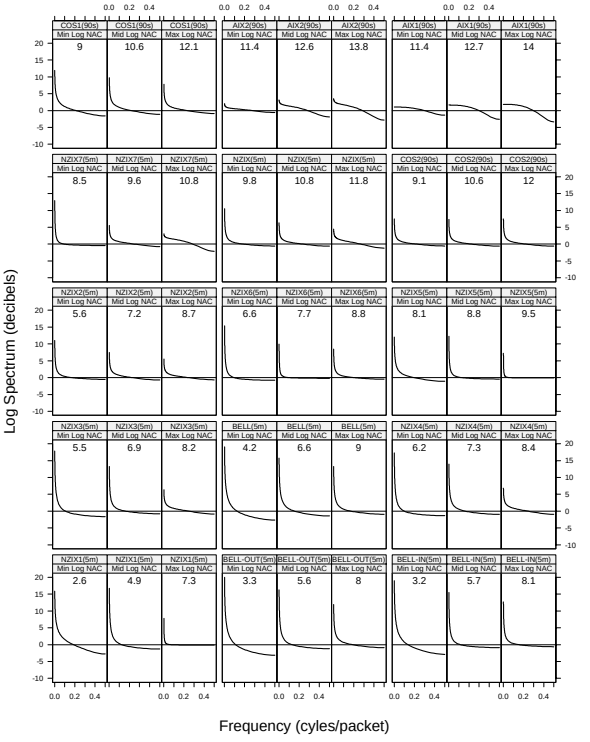

Fig. 10. For each of the 15 trace groups, the power spectrum for sixth root inter-arrival is plotted against frequency for a set of 3 values of the number of active connections. This roll-your-own S-Net tool is described in Section II-D.

for AIX1, AIX2, NZIX7, and NZIX, the general pattern is very large power at low frequencies, a rapid rise as the frequency tends to 0, and an overall monotone decrease in power as the frequency increases from 0 cycles/packet to 0.5 cycles/packet. This is a result of the long-range dependence of the $\tilde{t}_j$. But as c^* increases, the fraction of low-frequency power decreases, the fraction of high-frequency power increases, and the spectra approach the spectrum of white noise. For AIX1, AIX2, NZIX7, and NZIX, as c^* increases, the spectrum near 0 decreases but the spectrum at high frequencies decreases as well so that the end result is a log spectrum that is not flat and decreases slowly as the frequency increases. This is the spectrum of a time series with very short range dependence.

Figure 11 graphs the estimates of the power spectrum of $\tilde{p}_i$. They show the expected long-range dependence, the rapid rise at frequency 0. For each trace group, the spectra show little change with c^*. Furthermore, the spectra for different groups are very similar.

III. PackMime: Closed-Loop Internet Packet Traffic Modeling

In another study using S-Net, we developed models for HTTP application connection variables. We call the collection of resulting models, PackMime. The purpose of PackMime is to provide stochastic generation of application requests for network simulators such as NS, which in turn provides closed-loop modeling of packet traffic. As in the study of Section II, we discovered that the magnitude of statistical multiplexing had a dramatic effect on the statistical properties.

Here, we show a few tools applied to one connection variable: HTTP start times. The data are the arrival times of client HTTP SYN packets on the link for the 500 BELL(5min) traces in Table I. The magnitude of multiplexing also has a major effect on

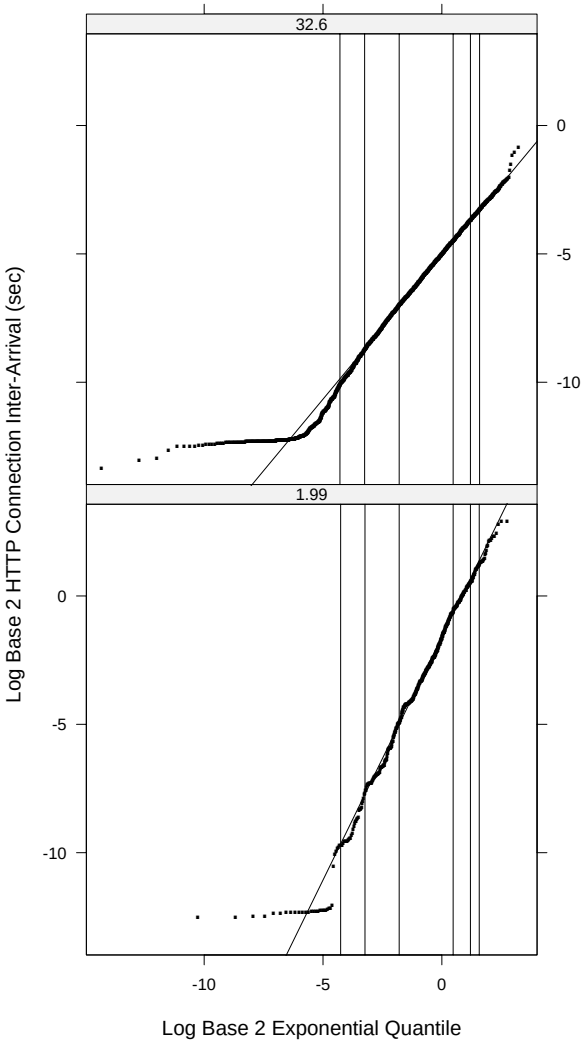

Fig. 12. Log HTTP connection inter-arrivals are graphed against the log quantiles of an exponential for two BELL(5min) traces. This out-of-box tool is described in Section III-B

the statistical properties of application connection variables, but in this case it is convenient and natural to use the number of new HTTP connections per second as the measure of the magnitude of multiplexing.

A. Notation

We will introduce mathematical notation to explain the tools used in the study.

Application Connection Variables

We begin with application connection variables, which have many values per trace.

a_j = the connection arrival times.

$t_j = a_{j+1} - a_j$ = the inter-arrival times. $t_j^* = \log(t_j)$.

Application Connection Summary Variables

We now turn to application connection summary variables, which have one value per trace.

ρ = the average number of new HTTP connections, where we average across all times in each trace. $\rho^* = \log(\rho)$.

Let w be a random variable that has a Weibull distribution with shape parameter λ and scale parameter α. Then $(w/\alpha)^\lambda = u$ where u is a unit exponential. So if $\lambda = 1$, w has an exponential distribution. For $\lambda < 1$, the upper tail of w is heavier than that of the exponential, but as λ increases to 1, the upper tail becomes less heavy. λ is a trace summary measure of the t_j that describes the shape of their marginal distribution, in particular, the heaviness of the upper tail.

B. Marginal Distribution

We studied the marginal distribution of the t_j of each trace by a Weibull quantile plot. Figure 12 shows this S-Net tool for two BELL(5min) traces. The t_j^* are graphed against the log quantiles of the exponential. The oblique line is drawn through the 25% and 75% quantiles. The vertical lines show the 5%, 10%, 25%, 75%, 90%, and 95% quantiles. The value of ρ is shown in the strip label at the top of each panel. If the pattern of the points is a line, then the t_j are well approximated by the Weibull. In such a case, the inverse of the slope of the pattern is the Weibull shape parameter, which is 1 if the Weibull is an exponential.

In Figure 12, the pattern of the points follows the quartile line well on both panels. There are deviations at the bottom end of the distribution. The empirical distribution is truncated to the transmit time of a minimum-size packet, 40 bytes plus encapsulation. This effect, which increases with the magnitude of the multiplexing, is never more than minor for the 500 Bell(5min) traces; for the two plots of the figure, the vertical lines show the truncation affects no more than 5% of the data. The inverse slopes of the patterns of the points, the Weibull shape parameters, are both less than 1, but in the top panel the shape, is considerably greater, so the distribution is closer to an exponential.

We fitted the Weibull distribution to the 500 BELL(5min) traces and plotted λ against ρ^* to see how the parameter changed with the magnitude of the multiplexing. Figure 13 shows the result; the parameter tends toward 1 with an increase in the load,

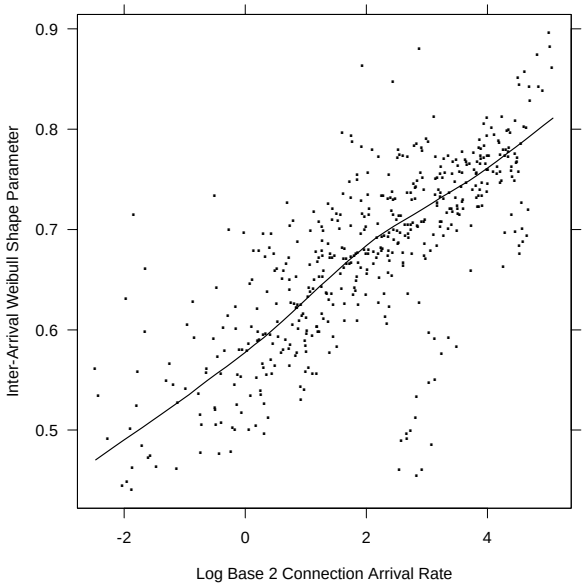

Fig. 13. Estimates of the Weibull shape parameter are graphed against the log of the number of new connections per second for the BELL(5min) traces. This out-of-box tool is described in Section III-B.

that is, the inter-arrival marginal tends toward the exponential.

C. Long-Range Dependence (LRD)

The t_j for HTTP inter-arrivals also have a component of LRD, and it, just like the inter-arrivals of packets, is reduced with the magnitude of multiplexing.

The S-Net tool in Figure 14 reveals changing properties in the t_j. In the bottom panel, for one BELL(5min) trace, $t_j^* = \log(t_j)$ is graphed against a_j. The value of ρ is 1.99 connections/sec. The log on the vertical scale is vital because the t_j vary by several orders of magnitude. The horizontal scale, however, conveys arrivals and inter-arrivals on the original scale. The top panel of Figure 14 plots another trace with ρ equal to 32.6 connections/sec.

In the bottom panel of Figure 14, the data form distinct vertical bands, taking values in the middle of the distribution of values. These are bursts of connections caused by single clicks of individual users. The larger values of the t_j^* on the plot tend to be quiescent times until the click of some user occurs. In the top panel of Figure 14, the bursty behavior at the small time scales has disappeared. Because ρ is much larger, the SYNs of more users intermingle, and the behavior of individual users is broken up. The t_j are headed toward independence.

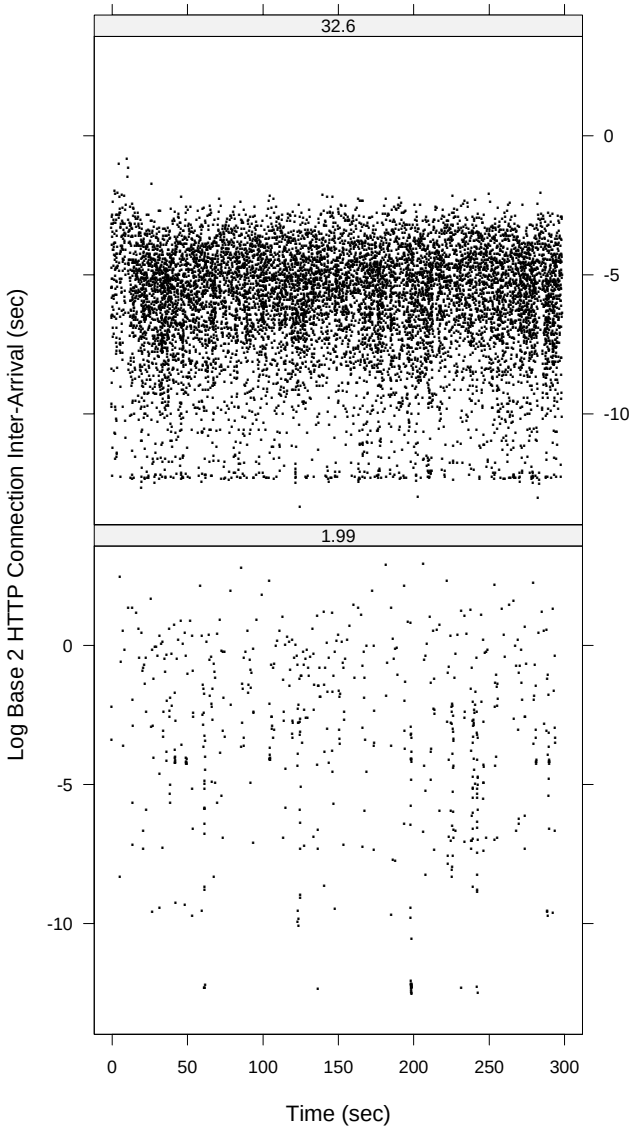

Fig. 14. Log HTTP inter-arrival time is graphed against the arrival time of the first packet of the pair that forms the inter-arrival time. This out-of-box tool is described in Section III-C.

IV. Analysis Strategy for Comprehensive Analysis

A. Data Flow

Figure 15 shows the data flow of S-Net. Primary flat-file objects are compressed flat ASCII files that form the master database in the S-Net analysis system. One example of primary files is the headers of a single monitor broken into time blocks (by packet arrival time) with the headers in a block ordered by their timestamps. This is what was accessed to carry out the analysis described in Section II. A second example, again for a single monitor, is headers of TCP packets organized by TCP connection; each time block has the packets of all connections that start in the block, the packets in the block are organized by connection, the connections are ordered by their start times (arrival times of SYN packets), and for each connection the packets are ordered by their timestamps. This is what was accessed to carry out the analysis described in Section III.

Secondary flat-file objects are compressed flat ASCII files containing derived quantities from the primary flat-file objects. For example, for the TCP connection primary files, secondary files might be the counts of connection starts in 10 ms intervals. Secondary flat files or pieces of secondary flat files are read into

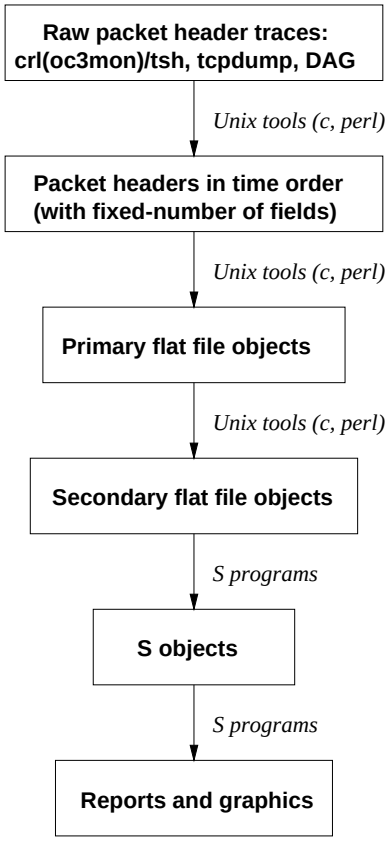

Fig. 15. Data Flow in the S-Net environment.

S as S objects.

Primary flat files underlie very large numbers of data analyses. The secondary flat files are created with more specific analyses in mind. Further database management occurs through selecting just some information to be read into S. The resulting reads create S objects that typically form the basis for a specific analysis; but in the course of the analysis many new S data objects are often created to provide more focussed access of data.

This design of the data flows contributes to the ease of comprehensive analysis. The flow from primary packet files to S objects makes it far more convenient to carry out detailed study because the analyst is typically spending time altering analyses through access of S objects and secondary flat files, rather than going all the way back up to the primary flat files (or the raw packet header traces), which is tedious and time-consuming. Of course, it does require some foresight about the likely course of an analysis and strategic thinking about how to form objects down the flow to enable more efficient analysis.

B. Time Blocking with Repetitive Analysis

We have found that a useful strategy for achieving comprehensive, detailed analysis is time blocking with repetitive analysis. Suppose we have a long trace from a monitor covering many days, weeks, or months. We break the trace into time blocks, and apply the same analysis to all blocks. The number of blocks can (and often should) be in the hundreds or thousands. Sections II and III are two examples.

A critical issue is the length of the time blocks. Some issues call for longer blocks, others shorter. First, we want stationarity of phenomena over a block. This pushes toward shorter blocks. But we need to have a time block long enough that the phenomena under study can come to completion. For this purpose we can in a sense extend beyond the block; for the example of Section III, we study all flows that begin in a 5 min block, but measure aspects of the flow that extend beyond the block. Finally, we need to take the computational burden of the tools into account; more burdensome tools call for shorter blocks.

For the example in Section II on packet traffic modeling, and the example in Section III, 5 min was a reasonable choice. The major issue that pushes toward shorter blocks is diurnal variation; the number of applications carried out has a dramatic 24 hour trend and getting much beyond about 15 min runs the risk of interference from this trend. In both cases, we want the mean magnitude of statistical multiplexing — the mean number of active connections or the mean number of new connections per second — to be constant over the block; the reason is the that a changing mean changes the properties under study, so we need to keep the blocks short enough to capture properties for a fixed mean. In our analyses, intervals with nearly the same fixed mean are repeat samples. The AIX and COS trace groups were only 90 sec, less than ideal, but the longest available. To make sure that this did not bias our results we analyzed 90 sec traces for BELL and NZIX, and found results were similar to those for 5 min except there was greater statistical variability. This suggested that 90 sec was not unreasonable for AIX and COS.

Of course, characteristics can and do change over time scales greater than the block length. This can be investigated by studying how the results of the block analyses change with time.

C. Detailed and Summary Study

It is important to have tools that allow study in the finest possible detail. In effect we seek tools that allow us to see every individual measurement. For example, in Section III, the Weibull quantile plot of inter-arrival times, and the plot of log inter-arrivals against arrivals, both show every arrival in a trace. In Section II, the plot of log inter-arrival against log encapsulated size, the plot of one-packet bandwidth, and the plot of log packet size minus the log mean against time all show individual measurements of the variables under study. Anything less runs the risk of missed effects. Summaries, with one value per trace, are extremely useful, but are not nearly enough. In Sections II and III, plots of summaries against measures of the magnitude of multiplexing are critical to the analyses, but are hardly enough.

D. Multiplexing

Statistical multiplexing needs to be at the core of many studies of Internet packet header data. The magnitude of statistical multiplexing has a large impact on the statistical properties of many variables, both packet variables and application connection variables. This is amply demonstrated in the studies cited in Sections II and III. Properties need to be related to measures of the magnitude of multiplexing such as the number of active connections or the number of connections per second. S-Net provides ample capabilities to take the magnitude of multiplexing into account.

E. Trellis and Multipage Display

Time-blocking and trellis display [8] go well together. Trellis is a framework for the visualization of multivariable databases. Its most prominent aspect is an overall visual design, reminiscent of a garden trelliswork, in which panels are laid out into columns, rows, and pages. On each panel of the trellis, a subset of the data is graphed by a display method such as a scatterplot, curve plot, boxplot, 3-D wireframe, Weibull quantile plot, or dot plot. Each panel shows the relationship of certain variables, the panel variables, conditional on the values of other variables, the conditioning variables. The strip labels at the tops of panels are indicators of the values of conditioning variables. A number of display methods employed in the visual design of Trellis display enable it to uncover the structure of data even when the structure is quite complicated. All visualizations shown in Sections II and III used trellis software implemented in the S language. For example, Figures 9, 10, and 11 are the pages of a 3-page trellis display, each page with 45 panels.

Sending a display across many pages is necessary to study a very large header database in detail. There can be hundreds or thousands of displays. Figures 1, 2, 4, 12, and 14 are single-panel or two-panel trellis displays with a single page. But in the actual displays of our analyses, there are as many panels across all pages as there are traces. So for the first three tools there are 2526 panels, the number of traces in the FSD study, and for the last two there are 500, the number in the PackMime study. This means, of course, there are a very large number of pages. Our practice is to break the panels into separate displays by trace group, and produce a single display for each group. We find it convenient to store displays as a PostScript document and study

them using Ghostview.

V. Out-of-Box S-Net Variables and Tools

The examples in Sections II and III have shown examples of S-Net tools. Now we simply provide lists of two sets of out-of-box tools, one that applies to packet variables and one that applies to application connection variables.

A. Packet Variables and Tools

Suppose that for a given time block we have measurements of the following primary packet variables for the jth packet: arrival times a_j, sizes q_j (where q_j is the size of the packet transmitted during the t_j interval), a connection id defined by source/destination IP addresses, source/destination ports, and the transport protocol (TCP,UDP). Furthermore, suppose that from these primary packet variables we compute three derived variables: inter-arrival times t_j, byte counts b_i in fixed-length intervals such as 10 ms, and packet counts p_i in fixed-length intervals. We also derive the number of simultaneous connections c_k at the start and end of each TCP/UDP connection, where the start time and end time are defined as the time of the first and last packet from the connection.

We developed S-Net visualization functions to study the arrival process of packet variables with focus on their marginal distribution and time correlation. These visualization functions fall into three main categories: time plots, quantile plots and spectrum plots. Let z_j be one of these packet variables. A time plot of z_j simply graphs z_j (or transformed z_j) against time of the observations. This can be used to study time characteristics of z_j. A quantile plot of z_j graphs ordered values of z_j (or transformed z_j) against the quantiles of a reference distribution. Quantile plot can be used to study the marginal distribution of a variable and to examine if the marginal distribution is well approximated by the reference distribution. We chose the reference distribution for each packet variable based on extensive empirical study. A spectrum plot of z_j graphs the raw log averaged periodogram values and smooth nonparametric power spectrum estimate of the time series of z_j against frequency. This can be used to study time dependence, in particular, long-range dependence.

The following list describes the current set of S-Net visualization functions applicable to the packet variables and gives the purpose of each function:

1. time plots of $\log t_j$, q_j, $\log p_i$, $\log b_i$ and c_k.
2. uniform quantile plot of q_j, Weibull quantile plot of t_j, normal quantile plot of $\log p_i$ and $\log b_i$, and normal quantile plot of c_k
3. spectrum plots for normalized $t_j^{1/6}$, q_j, $\log p_i$ and $\log b_i$ with mean 0 and variance 1
4. log t_j against $\log q_j$: study relationship of t_j and q_j to assess how accuracy of t_j depends on q_j
5. quantile plot of q_j/t_j: study accuracy of t_j, and look for upstream bottleneck link speeds.

For the case of multiple time blocks, we developed S-Net functions for computing block summary variables, with one measurement per block for each variable. We then study changes of these summary variables with characteristics of the time block, for example, statistical multiplexing. The following describes some of the important summary variables and the purpose of each:

1. percent of packets back-to-back with predecessor: measures the magnitude of the upstream queueing plus packets that originate back-to-back and stay that way
2. maximum likelihood estimate of the Weibull shape and scale parameter of t_j: as this measure of the distributional shape increases, the upper tail of the distribution becomes less heavy-tailed
3. coefficient of variation of p_i or b_i: measures the amount of statistical variability relative to the mean
4. entropy of the normalized $t_j^{1/6}$, q_j, $\log p_i$, or $\log b_i$ with mean 0 and variance 1: measures dependence based on the power spectrum
5. estimate of the variance ratio parameter in the FSD model of $\log t_j$, q_j, $\log p_i$, or $\log b_i$, as 1 minus variance ratio measures the extent of long range dependence in the FSD model
6. average number of simultaneous active connections: measures amount of statistical multiplexing.

B. Application Connection Variables and Tools

Our study of application connection variables has been focused on TCP connections, especially, HTTP connections. In the following, we describe the HTTP connection variables and tools we developed to study these variables.

For the j-th HTTP connection that started in a given time block, we have measurements of the following variables:

1. arrival time a_j, measured by the time of the client SYN packet
2. duration l_j measured by the time from the client SYN to the last data packet from the server
3. client file size f_j and server file size F_j, measured by the differences in sequence numbers
4. client-side round trip time r_j, measured by the time from server SYN/ACK to client ACK in the three-way handshake
5. server-side round trip time R_j, measured by the time from client SYN to server SYN/ACK in the three-way handshake,
6. server delay D_j, measured by the time between the last client data packet and the first server data packet, and then subtract the client-side and server-side round trip time (define D_j to be 0 if the result of this is less than 0)
7. the connection id defined by source/destination IP address and port.

From the arrival time a_j, we derive the connection inter-arrival time t_j.

We developed S-Net visualization tools to study these HTTP connection variables, many of which are similar to that for packet variables. The following describes a set of S-Net visualization functions applicable to these connection variables.

1. time plots of $\log t_j$, $\log l_j$, $\log f_j$, $\log F_j$, $\log r_j$, $\log R_j$ and $\log D_j$
2. Weibull quantile plot of t_j, normal quantile plot of $\log l_j$, Pareto quantile plot of f_j and F_j, Weibull quantile plot of r_j, R_j and D_j
3. spectrum plots of transformed t_j, l_j, f_j, F_j, r_j, R_j, and D_j with a $N(0,1)$ marginal distribution

We also developed some specialized tools. For example, we find that the time correlation in server file size is mostly due to the fact that there are often multiple not-modified messages from the server in response to a series of cache validation requests from the client. So we developed discrete Weibull quantile plots to study the run length of cache validations with not-modified messages from the server and run length of actual file transfers.

In addition to the connection variables discussed above, we also derived variables to study the multiple request and response exchanges within a persistent HTTP connection. For example, the client and server file size of each request and response exchange, the time gap between two request and response exchanges, and the server delay for each request and response exchange. Additional S-Net analysis and visualization tools are developed for these variables.

Similar to the packet variable, for the case of multiple time blocks, we developed S-Net functions for computing block summary variables. The following describes some of the important summary variables.

1. maximum likelihood estimates of the Weibull shape and scale parameters of connection inter-arrival variable t_j.
2. entropy of the transformed t_j, l_j, f_j, F_j, r_j, R_j and D_j with a $N(0,1)$ distribution: measures dependence based on the power spectrum
3. average connection arrival rate: measures amount of statistical multiplexing of connection arrival

VI. Hardware, Implementations, and Databases

S-Net lends itself readily to implementation on a hardware environment consisting of a cluster of Unix PCs, each storing flat-file objects and S objects on attached disk. The system can be distributed over a high speed local area network; this effectively reduces computational time in performing a specific analysis task. The disks of the individual hosts are shared through the Network File System (NFS); this helps to reduce the burden on individual disks when large amounts of data are accessed by multiple processors from different disks on different hosts.

S-Net has so far been implemented on two Linux clusters over fast networks, one at a Bell Labs Research location and one at the NCNI Research facility as part of the Helios DARPA project. The first implementation has an associated database of packet header files from the BELL database: 3 years of monitoring on the link between a Bell Labs Research network and the rest of the Internet [6]. The database associated with the second implementation is 42 hours of monitoring on the link between the University of North Carolina, Chapel Hill, and the NCNI OC48 ring [12], which carries all traffic between UNC and the rest of the Internet.

References

[1] R. A. Becker, J. M. Chambers, and A. R. Wilks. *The New S Language*. Chapman & Hall, London, 1988.

[2] J. Cao, W. S. Cleveland, D. Lin, and D. X. Sun. On the Nonstationarity of Internet Traffic. In *Proc. ACM SIGMETRICS 2001*, pages 102–112, 2001.

[3] J. Cao, W. S. Cleveland, D. Lin, and D. X. Sun. The Effect of Statistical Multiplexing on the Long-Range Dependence of Internet Packet Traffic. Technical report, Bell Labs, Murray Hill, NJ, 2002.

[4] J. Cao, W. S. Cleveland, and D. X. Sun. FSD: Open-Loop Modeling of Internet Packet Traffic. Technical report, Bell Labs, Murray Hill, NJ, 2002.

[5] http://caida.org/tools. The CAIDA Tools Site.

[6] http://cm.bell-labs.com/stat/InternetTraffic. Bell Labs Internet Traffic Research.

[7] http://cm.bell-labs.com/stat/InternetTraffic/S-Net. The S-Net System.

[8] http://cm.bell-labs.com/stat/project/trellis. Trellis Display.

[9] http://cm.bell-labs.com/stat/S. The S System.

[10] http://pma.nlanr.net/PMA. Passive Measurement and Analysis Project at NLANR.

[11] http://wand.cs.waikato.ac.nz/wand/wits/nzix/2. NZIX-II Trace Data.

[12] http://www.ncni.org. The North Carolina Networking Initiative.

[13] http://www.R-project.org. The R Project for Statistical Computing.

[14] R. Ihaka and R. Gentleman. R: A language for data analysis and graphics. *Journal of Computational and Graphical Statistics*, 5(3):299–314, 1996.