3 Traffic Patterns of Individual Workstations

In Chapter 2 we have examined the aggregate, system-wide network traffic. Our next step will be to analyze the traffic from the perspective of the basic units that generate traffic—individual workstations. Although individual workstations are not the only conceivable units for studying network traffic—two alternatives are individual processes and clusters of clients workstations—they are the most logical choice. Individual (client) workstations are associated with individual (human) users and, at least to a certain degree, produce traffic when users are active. In this chapter we will perform several types of stochastic analyses on the arrival processes associated with the receive and the send queues of individual workstations.

A workstation's send queue process will be constructed by extracting packets whose source address is the workstation's network address from the general traces. Similarly, packets whose destination address is the workstation's address will be selected from the general traces and assembled together as the receive queue process. These will include packets originating from the workstation's file server as well as packets from other workstations. In the case of a file server, the transmit queue will contain packets addressed to several clients, some or all of which may be simultaneously active, and the receive queue will contain packets from all the clients. Thus, for file servers the packet arrival processes are the superposition of more basic client-workstation processes. As discussed in Chapter 2, each of the records of these extracted (sub)traces will contain timing information that will allow us to compute interarrival times.

Of course, the processes obtained in the manner described above will not be the same as those that would be recorded if the only machines on the network were source and destination machines. On an Ethernet, as the offered load increases, the network contention also increases, resulting in collisions, and, ultimately, in transmission delays. Although we will not consider the effects of collisions, in this chapter we will show that in the Sun Engineering network there is only a moderate amount of interaction among the various machines, i.e., the proportion of time in which two or more pairs of communication machines are active simultaneously is small. However, we make no claim that our network packet-arrival processes have the same stochastic properties as the packet-arrival processes that would be produced in a contention-free network.

In the next section we will look from a macroscopic time scale at the load patterns produced by individual workstations. These patterns consist of sequences of busy and idle segments, which we will analyze in Section III. In Section IV we will look at histograms of interarrival times, and associate protocol properties with the peaks in the histograms.

The processes we are dealing with are nonstationary (Section V). By extracting from each arrival process a subset of the interarrival times, which we define as the ``busy period'' in Section VI, we will obtain processes with better stationary properties. Using these subsets, we will study the autocorrelation coefficients of packet arrival processes in Section VII. Finally, we will discuss whether network serialization and sharing or shared file servers produce correlations in pairs of otherwise unrelated processes.

II. The Network Load of Individual Workstations

The various client workstations listed in Table 3.3 each generate roughly the same total volume of traffic during the working hours of a day. Yet, user behavior— busy/idle cycles of packet transmission—varies a great deal from workstation to workstation; differences in the work patterns of individual users account for this variation. It is instructive to look at the Ethernet load generated by the send queue and at the load directed to the receive queue of a couple of workstations.

Figures 3.1 and 3.2 display these loads for the workstation *station 3*. (Each vertical line represents the average load over a 30-second interval.) We notice that communication follows an on-off behavior typical of a person's work pattern. The spikes in the graphs indicate that there is burstiness, by which term we imprecisely refer to a high peak-to-mean ratio, when the load is measured at 30-second intervals. However, it is unclear whether this interval or a shorter one is appropriate for studying burstiness. In Chapter 4 we will introduce a formal definition of (interarrival time) variability and propose quantitative techniques for measuring it.

FIGURE 3.1. NETWORK LOAD (AS PERCENTAGE OF TOTAL) WORKSTATION *STATION 3* (SEND QUEUE)

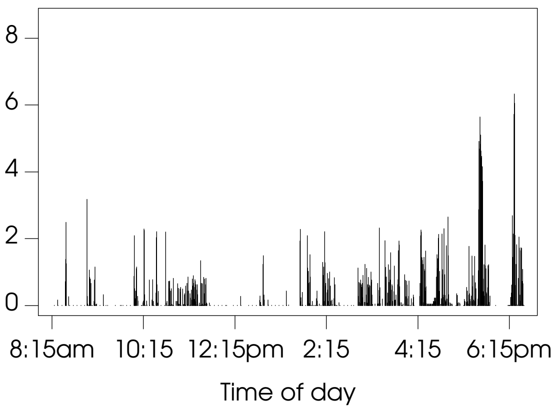
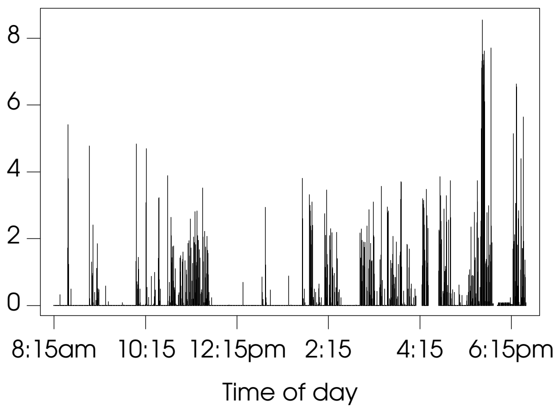


FIGURE 3.2. NETWORK LOAD (AS PERCENTAGE OF TOTAL) WORKSTATION *STATION 3* (RECEIVE QUEUE)



Station 3, a diskless Sun-3 model 50 with 4 Mbytes of physical memory, does not create a high load on the network in spite of its small memory, which could potentially cause a high paging rate. Nor did any of the other Sun-3/50's generate a high load. In the Berkeley study we saw that individual Sun-3/50's on occasion generate enough load to utilize one-fifth of the Ethernet raw bandwidth. This did not occur in the Sun environment, most likely because there was a better match between users' workloads and hardware capabilities than at Berkeley. In a university, where each machine is commonly shared by several student users, workstations tend to have shorter pauses (i.e.,

periods of idle time). Furthermore, students often overwork their hardware, sometimes at the expense of performance; in a company, the loss of productivity resulting from insufficient hardware power is not tolerable, and more powerful hardware is obtained for heavy users, while lighter users receive less powerful machines. Another reason for the network load difference is lower paging rates, which are the result of the improvements, mentioned earlier, in the virtual memory management scheme of SunOS 4.0. Finally, we note again that in the Sun network there are fewer diskless workstations, machines that depend on network communications for sending and receiving even the smallest bit of information to and from their file systems, the temporary file space, and the swapping area.

The dots visible in Figure 3.1 can be attributed to periodic messages exchanged between client and server, and are produced by small software tools that are frequently run on client workstations. Such tools typically monitor some performance index on the server (or on other machines), for instance the CPU load, displaying graphically the information on the user's display. Monitoring is performed by periodic packet transmittal, which introduces a deterministic component in the interarrival times, complicating our analysis but affecting the network load very little. In Figure 3.2, these periodic packets also appear as horizontal lines when relatively few other packets are sent.

FIGURE 3.3. NETWORK LOAD (AS PERCENTAGE OF TOTAL) WORKSTATION *STATION 16* (SEND QUEUE)

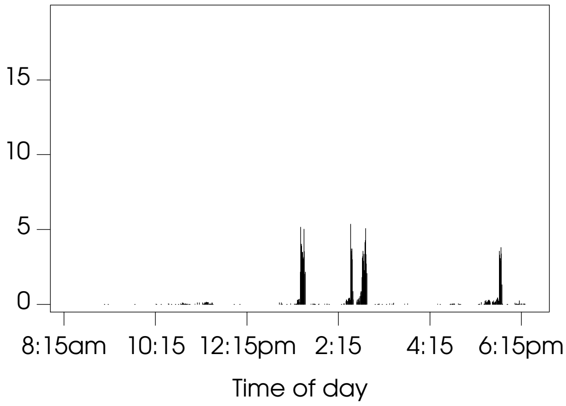
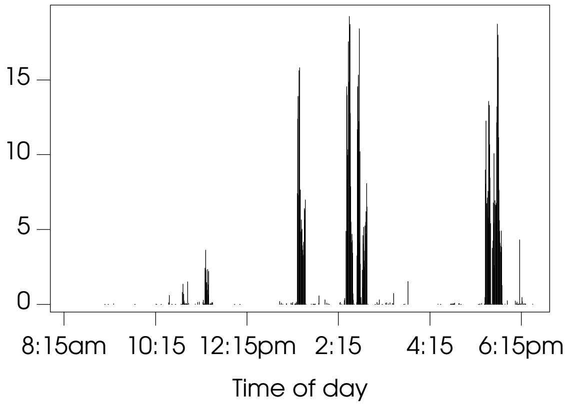


FIGURE 3.4. NETWORK LOAD (AS PERCENTAGE OF TOTAL) WORKSTATION *STATION 16* (RECEIVE QUEUE)



As another example of the types of traffic patterns produced by a client workstation, we show in Figures 3.3 and 3.4 the load generated on the Ethernet by those packets sent and received by *station 16*, a Sun-4 model 260, with 16 Mbytes of main memory and a 250-Mbyte local disk. Local disks on client workstations are typically configured to hold the root file system and the swapping partition, to both of which much of the UNIX disk traffic is directed. This disk traffic produces relevant network traffic if the machines are diskless. Hence, workstations with local disks, as in the *station 16* example, display lower network utilization. The figures show that while *station 16* does not keep the network as busy as as *station 3* does, *station 16* produces peak network loads that are higher and longer than *station 3*'s because its faster architecture can sustain higher transmission data rates. This pattern is probably caused by the workstation transferring data from other workstations to its local disk during its busy periods. The high number of 1500-byte packets in Figure 3.6, plot of the packet length distribution in the receive queue, indicates the higher activity of its receive queue.

FIGURE 3.5. PERCENTAGE OF PACKETS VS. PACKET LENGTH
WORKSTATION *STATION 16* (SEND QUEUE)

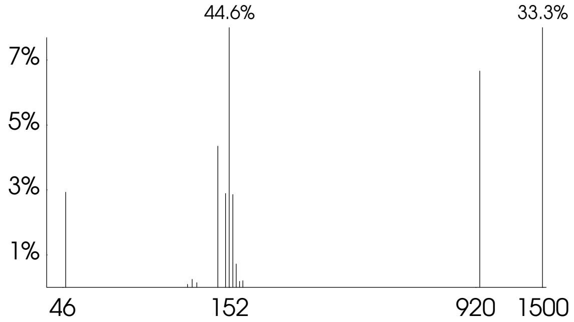


FIGURE 3.6. PERCENTAGE OF PACKETS VS. PACKET LENGTH
WORKSTATION *STATION 16* (RECEIVE QUEUE)

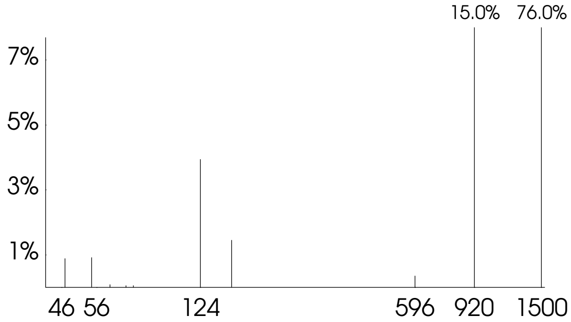


Figure 3.5 shows that medium-sized packets are over-represented in comparison with the network average shown in Figure 3.15. In contrast, medium-sized packets are under-represented in the receive queue (see Figure 3.6). With such unusual packet size distributions and network load patterns, traffic models based on *station 16* data will not be representative of typical-traffic patterns.

III. Busy and Idle Phases

Figures 3.1 through 3.4 reveal on a macroscopic time scale of busy/idle intervals just how variable network traffic patterns driven by individual user behavior can be. Busy periods—time intervals during which the network load is significantly different from zero—can range from 30 seconds to 30 minutes; idle periods from a few seconds to several hours. We have studied the autocorrelation coefficients of the lengths of busy and idle segments for the send and receive queues of several workstations. The values of the coefficients suggest that these processes could be modeled as alternating renewal processes, i.e., as processes in which successive intervals are sampled from two independent probability distributions. The series of correlation coefficients of an alternating renewal process alternates between a positive and a negative value [19, page 89]. The autocorrelation coefficients of the estimated busy/idle phases do alternate, although they decrease as the computed correlation is for busy/idle phases farther and farther apart (i.e., as the lag increases) and eventually become negligible. Another model, perhaps more suitable for these types of data, would be a first-order autoregressive process of the form $X_{i+1} = \alpha X_i + Y_i$, where Y_i is a purely random process and the coefficient α is negative. The autocorrelation coefficients of this type of autoregressive process also alternate, but their absolute values decrease as the lag increases.

Beside these general observations, we will not attempt to describe the stochastic properties of the busy/idle periods because we do not have enough data to make parameter estimation statistically meaningful. To discriminate between different alternative models requires longer data sets, perhaps spanning several days. Instead, we will focus on the busy periods alone. Busy periods are the most important aspect of packet arrival processes: it is during these intervals that queue buildups occur; that network contention is reflected in the response time; and that, in general, protocol and system performance determines the shape of the arrival processes themselves. Later in this chapter we will provide a formal definition of ``busy period'' and characterize in detail busy periods of several client workstations and file servers.

IV. Histograms of Interarrival Times

In this section we will analyze histograms of interarrival times. Assuming that the series are stationary, interarrival-time histograms are, except for a scale factor, biased estimates of the marginal probability density functions of the interarrival times. In fact, the probability that interarrivals X_i's assume values in an interval of size δ centered at x can be estimated by

$$\hat{\Pr}(x,\ \delta) = \frac{\sum b(X_i)}{N}$$

where N is the number of elements in the time series and b is the function

$$b(t) = \begin{cases} 1 \ \text{ if } \ x - \dfrac{\delta}{2} \le X_i < x + \dfrac{\delta}{2} \\ \\ 0 \ \mathit{otherwise} \quad . \end{cases}$$

If the time series is stationary, the estimated probability is unbiased. Hence

$$\Pr(x,\ \delta) = \mathrm{E}(\hat{\Pr}(x,\ \delta)) = \lim_{N\to\infty} \hat{\Pr}(x,\ \delta) = \lim_{N\to\infty} \frac{\sum b(X_i)}{N} \ .$$

Since $\Pr(x,\ \delta)$ is related to the probability density function by

$$\Pr(x,\ \delta) = \int_{x-\frac{\delta}{2}}^{x+\frac{\delta}{2}} p(t)dt$$

by definition, if δ is small we can approximate the integral with the product $p(x)\delta$. Rearranging, we obtain

$$p(x) = \lim_{\delta\to 0} \frac{\Pr(x,\ \delta)}{\delta} = \lim_{\substack{N\to\infty \\ \delta\to 0}} \frac{\hat{\Pr}(x,\ \delta)}{\delta} = \lim_{\substack{N\to\infty \\ \delta\to 0}} \hat{p}(x) \ ,$$

where we have defined

$$\hat{p}(x) = \frac{\hat{\Pr}(x,\ \delta)}{\delta} = \frac{\sum b(X_i)}{N\delta} \ , \qquad (3.1)$$

a sample estimate of the probability density function.

From equation (3.1) we can compute the expected value of $\hat{p}(x)$:

$$\mathrm{E}(\hat{p}(x)) = \frac{\mathrm{E}(\hat{\Pr}(x,\ \delta))}{\delta} = \frac{\Pr(x,\ \delta)}{\delta} = \frac{1}{\delta} \int_{x-\frac{\delta}{2}}^{x+\frac{\delta}{2}} p(t)dt \ .$$

Thus, in general,

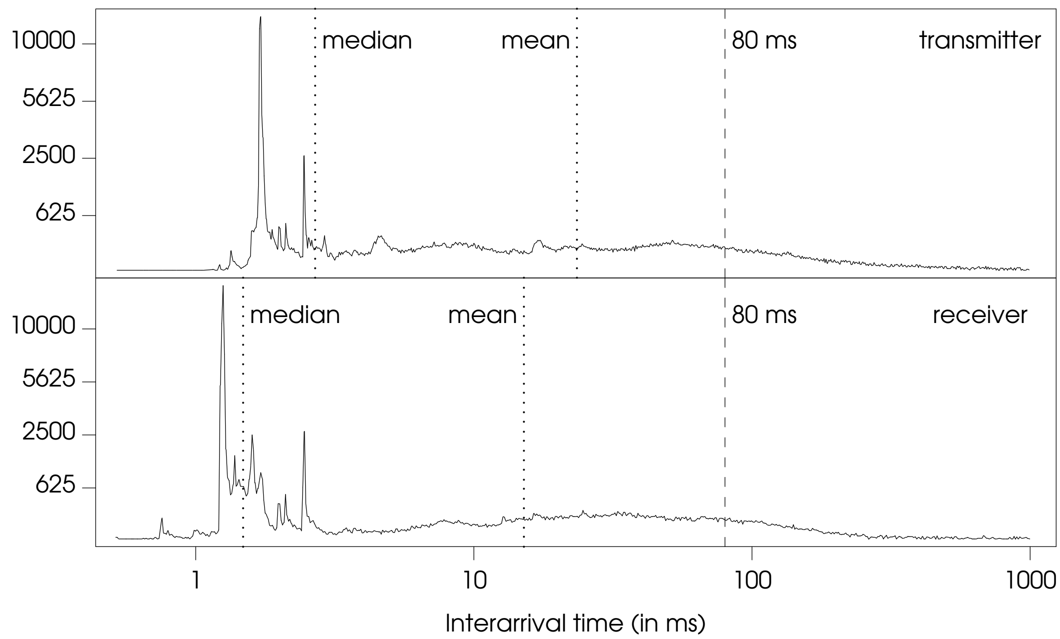
$$E(\hat{p}(x)) \neq p(x) ,$$

that is, $\hat{p}(x)$ is a biased estimate of $p(x)$. In addition, it can be easily shown that the bias is proportional to δ, the size of the interval.

In the figures of this section, we have chosen to indicate on the ordinates the values of the counts $\sum b(X_i)$ instead of the scaled version. This will allow the reader to relate more directly the height of the various points to the size of the data sets.

In Figure 3.7 we plot the histograms of the interarrival times of packets generated by the send queue and received by the receive queue of the client *station 25*. Each histogram contains 100 000 interarrival times. The abscissas of each graph were modified by a logarithmic transformation; on the vertical axis, we report the square roots of the bin counts.

FIGURE 3.7. HISTOGRAM OF INTERARRIVAL TIMES – CLIENT *STATION 25*

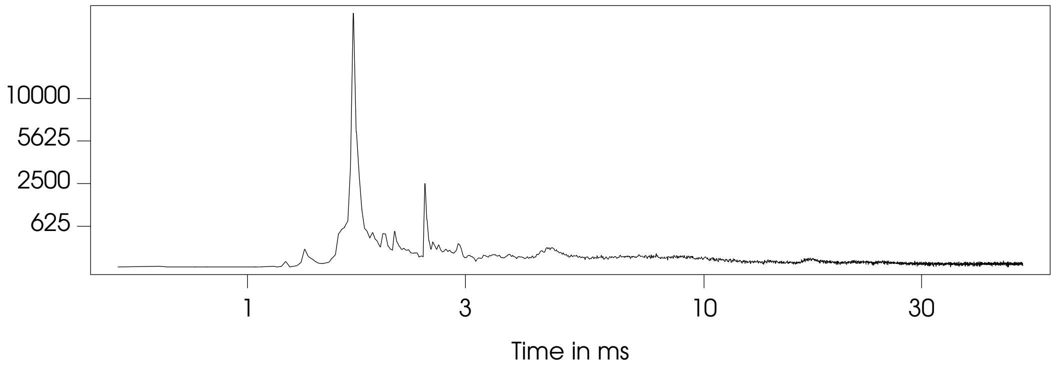


In Figure 3.7, and in the following histograms presented in this section, the estimation procedure is rather sophisticated: the size of the bins over which the histogram is computed is not constant, but increases exponentially, that is, the size of the bins is constant in logarithmic coordinates. The histogram is computed in a standard way, by counting the number of interarrival times that fall in each bin. Hence, in order to ``correct'' the shape distortion introduced by this method, one would have to multiply each point by a value obtained from a negative exponential curve.

Using exponentially increasing bins has several advantages. First, a single histogram can capture interarrival times spanning more than three orders of magnitude, which would not have been possible with equally spaced bins. Second, since the majority of interarrival times are concentrated in the 1 to 3 ms range, smaller bins on the lefthand side region of the histograms allow us to display the curves with higher resolution in that region. Third, since the variance of interarrival times within classes of communication events grows with the increase in the mean interarrival times of these classes of events, the procedure matches more closely the size of the bins to the variance (or, more properly, to the standard deviation of interarrival times) and permits the representation of all of these classes as peaks in the histograms. In contrast, in a standard histogram, classes of interarrival events whose standard deviations are larger than the size of the bins could be invisible. For instance, without considering network contention and collisions, a class of closely spaced interarrivals, such as those resulting from protocol fragmentation, shows a small variance and produces a narrow peak. Instead, periodic (but less frequent) events generated by programs that access a peripheral (such as a magnetic disk) show a much larger variance, and need a larger bin in order to appear as peaks. Finally, a fourth advantage of this procedure is that it amplifies the values of the histogram for large interarrival times allowing us to discriminate trends and properties that we would otherwise miss: for instance, without this feature we would not have seen the decrease in the interarrival times after 100 ms in Figure 3.7.

In order to provide an explicit comparison between a histogram with constant bin size and one with exponentially increasing bin sizes, in Figure 3.8 we show a constant-size-bin histogram that corresponds to the first part of the top graph in Figure 3.7.

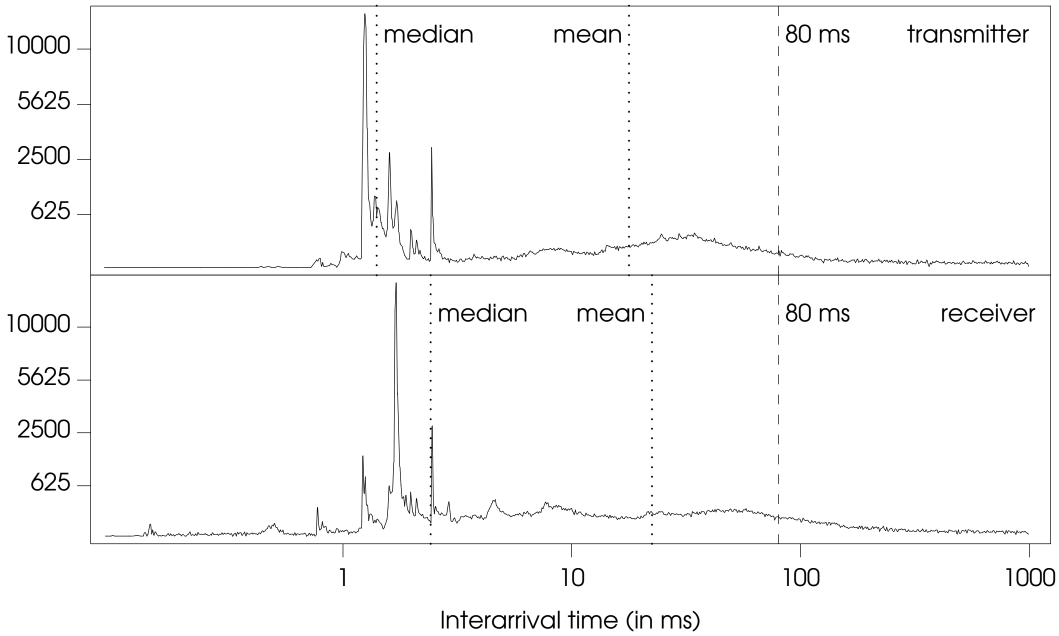
FIGURE 3.8. CONSTANT-SIZE-BIN HISTOGRAM OF INTERARRIVAL TIMES – CLIENT *STATION 25* (TRANSMITTER QUEUE)



In this figure we have used the same convention as in Figure 3.7: the abscissa is displayed in a logarithmic scale, and on the ordinate we plot the square root of the counts. Observe that the highest peak in Figure 3.8 is higher than the highest in Figure 3.7 since we use a bin that is actually larger than the bin used for the corresponding region in Figure 3.7, which, therefore, has a higher resolution in that region. It is important to note that this higher resolution does not involve higher computational costs; on the contrary, in Figure 3.7—showing histograms with only 800 bins—bins below 5.4 milliseconds are smaller than the constant-size bin in Figure 3.8; this figure contains 2000 points, and hence has a bin of about 25 microseconds. Because of the logarithmic x-axis scale, which results in a higher concentration of points in the right half of the curve, in Figure 3.8 the histogram is finely jagged for times larger than 10 milliseconds. Also, notice how the curve is stretched and flattened in this area, making it difficult to discern any trends or patterns. Changing the y-axis scale does not help: as mentioned above, there are too few points in each bin for any significant pattern to show up in the representation of Figure 3.8. Returning to Figure 3.7, we can appreciate how much easier it is to see the small bump at 13 milliseconds than it is in the constant-size-bin histogram.

A drawback of our variable-size-bin procedure, indicated by our analysis of the statistical estimate of probability density functions, is the increasing bias as interarrival times lengthen. However, since we do not plan to use these histograms (or probability densities) for quantitative analysis, but only to show qualitative properties of our arrival processes, this is not a major problem. Furthermore, given the nonstationarities present in our processes, histograms are affected by other errors that are difficult to quantify. Thus, it is not clear whether regular histograms could be used quantitatively anyhow.

FIGURE 3.9. HISTOGRAM OF INTERARRIVAL TIMES – SERVER *STATION 23*



In Figure 3.9 we show histograms of interarrival times for the send and receive queues of file server *station 23*. Although this figure and Figure 3.7 look roughly similar, there are several differences worth noting. The first portions of the histograms, showing interarrival times up to 10 milliseconds, depend on protocol time constants. As a result, the two workstations produce peaks at about the same points in the top histograms of each figure. However, whereas the highest peak in *station 25*'s send queue occurs at 1.71 milliseconds, the highest peak in *station 23*'s case occurs at 1.25 milliseconds. Since 1.2 milliseconds is the minimum time between two 1500-byte messages, *station 23* must be sending these fragmentation packets back-to-back. (In Chapter 2, we have discussed the consequences of minimum-interframe packets on the measurements.) These peaks correspond to the interarrival times between successive 1500-byte packets generated by the IP protocol fragmentation of 8-Kbyte NFS messages (see Appendix A for details on this type of message fragmentation). In *station 25*, these messages are write messages; in *station 23* they are replies to read requests. The relative shift can be explained by the difference in speed between the two CPUs: *station 25* is a 15 MHz Motorola 68020 and *station 23* a 16 MHz Sparc, which, in spite of the comparable clock rate, has a much faster architecture. (Analytical and measurement studies [33,49,64] have pointed out that the bottleneck in the file server is most often the CPU, and, as a result, a network system of this type is most often configured with the fastest machine acting as a file server for several, less powerful clients.)

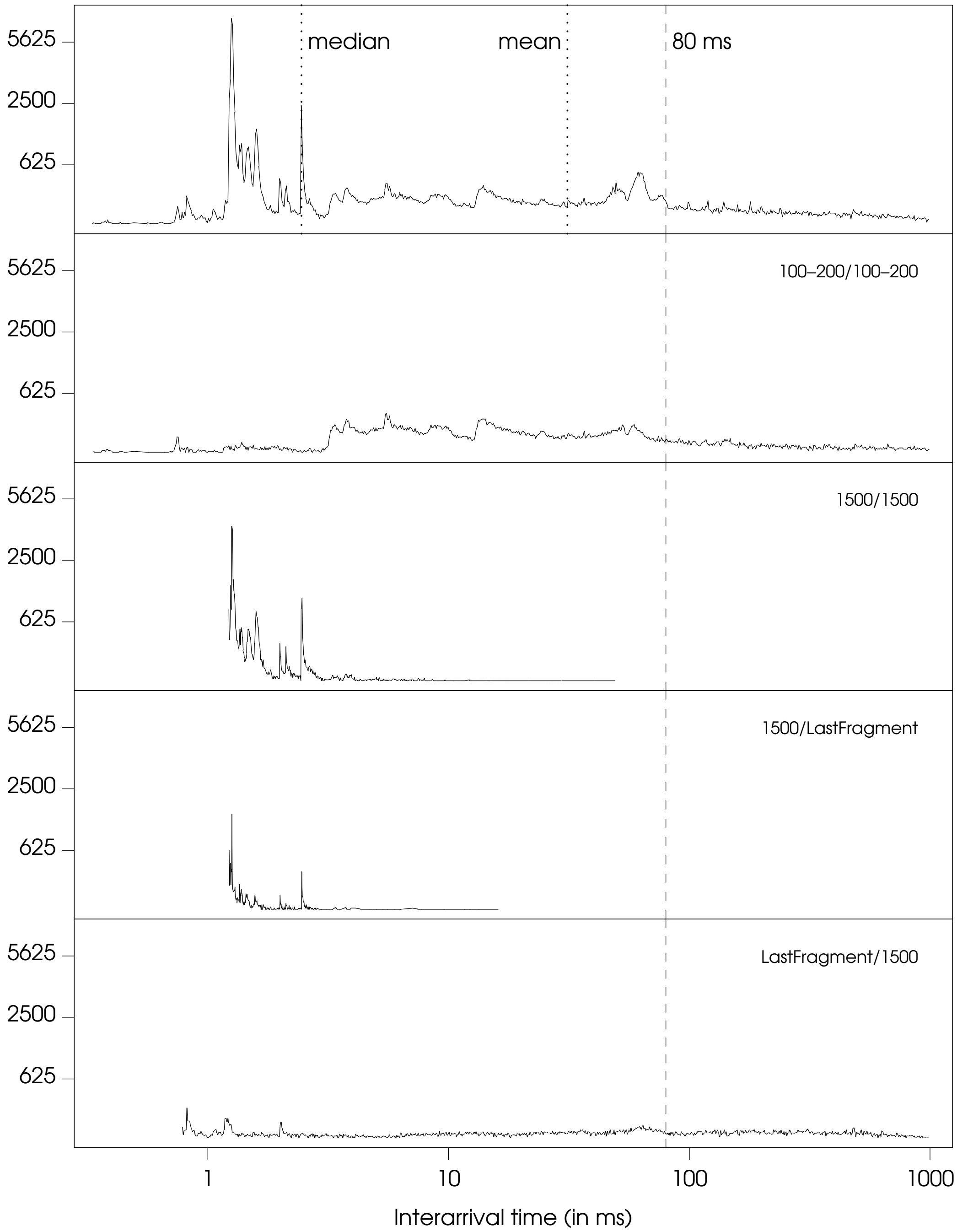
The same reasoning explains why the peaks of the transmitter are shifted to the right of those of the receiver in Figure 3.7 (the receiver queue of a client is related to the transmit queue of its file server). Because *station 23* is a file server, the situation is reversed in Figure 3.9. Also, since its receive queue is the result of the overlapped transmission of several clients of different speeds, the result is the multiplicity of peaks we see in the lower graph. *Station 23*'s simultaneous communication with several other workstations also explains why there is a nonzero probability that times between packets arriving at its receive queue are very small. This is shown by the extensions of the points to the left of 1 millisecond in Figure 3.9.

In the top graph of Figure 3.10 we show a histogram of 100000 interarrival times produced by packets arriving at the send queue of client workstation *station 15*. Although this machine has a local disk, it was relatively busy communicating with file servers and other clients, and its the estimated marginal distribution of interarrival times looks quite similar to those we have seen before. In the other four graphs in Figure 3.10, we show some principal components of the total traffic produced by the send queue. First, in the second graph from the top, we have the histogram produced by successive NFS procedure calls, the packets that in Figure 3.15 occupied the region of packet lengths between 100 and 200 bytes.

We recognize that these packets generate interarrival times that occupy primarily the middle region of the histogram, running from 4 to 80 milliseconds. NFS remote procedure calls require substantial processing time, and hence more time between successive send or receive operations, because they often require disk accesses, for instance during path-name translations; because of more complex protocol headers; because of the need for marshaling argument data types; because of the need to convert the data to an external representation so that it can be received by machines with other number representations or storage format conventions; and so on. In this region the curve is more or less flat, which means, recalling the way in which the variable-size-bin histogram works, that the interarrival time density function should be a negative exponential. The time between events in a Poisson process is exponentially distributed. However, it is doubtful that a Poisson process would model the data appropriately, since it appears that successive arrivals in the ``100-200/100-200'' class are strongly correlated.

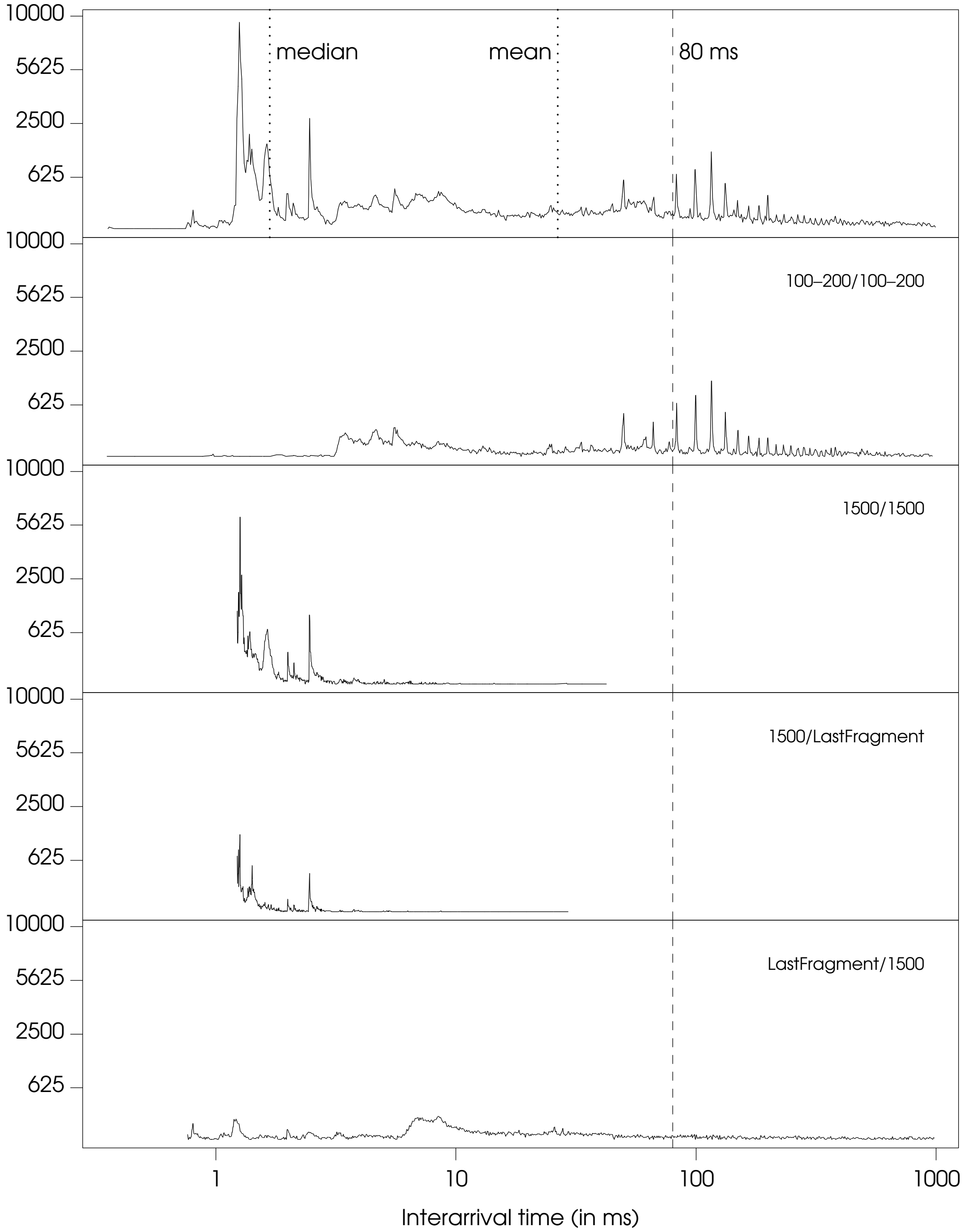
When the IP protocol layer fragments 8-Kbyte messages, it generates six fragments, the last of which is shorter; this produces four ``1500/1500'' interarrival times and one interarrival that we will call ``1500/LastFragment''. The distribution of these times is shown in the next two plots of Figure 3.10. These plots indicate that the interarrival times generated by the last fragments are not different from those of the previous ``1500/1500'' fragments. Peak locations and histogram domains are very similar. Peak values in the first of the two histograms are in a ratio of 4 to 1 to the peak values in the second histogram; this ratio corresponds exactly to the ratio between the frequencies of ``1500/1500''

FIGURE 3.10. HISTOGRAMS OF INTERARRIVAL TIMES – CLIENT *STATION 15* (SEND QUEUE)
BREAKDOWN OF INTERARRIVAL TIMES BY PACKET LENGTHS



pairs and ``1500/LastFragment'' pairs. We see that protocol fragmentation produces small interarrival times and calculate that about 44.5 percent of the interarrival times belong to these two classes. We observe also that the smaller peaks in these plots may be the result of transmission delays due to network contention and transmission reschedulings caused by collisions.

FIGURE 3.11. HISTOGRAMS OF INTERARRIVAL TIMES – CLIENT *STATION 15* (RECEIVE QUEUE)
BREAKDOWN OF INTERARRIVAL TIMES BY PACKET LENGTHS



Most often these 8-Kbyte messages are part of larger user-level data transfers; thus, it is quite common that two or more of them follow each other. If this happens, the packet after the last fragment of one message is the first 1500-byte fragment of the following message and it is interesting to look at the distribution of the times between these two classes of packets. We show this distribution in the last histogram of Figure 3.10. In this case we have a flat curve across the entire domain and the same considerations we made above for the middle part of the second plot apply here as well.

In Figure 3.11 we show plots equivalent to those of Figure 3.10 for the receive queue of workstation *station 15*. These histograms are, in general, quite similar to the previous ones. A distinctive feature, however, is the comb-like sequence of peaks in the aggregate histogram, which would not have been as apparent without the use of variable-size bins. As the second plot shows, the peaks are caused by interarrival times between medium-sized NFS packets. We do not know the source of these consecutive short NFS packets, although we have determined that all of the packets follow each other within a relatively short interval of real time. Most likely, they were generated by a particular user application. Because these interarrival process patterns are unusual, we decided not to be concerned with this level of detail in the models.

Another interesting feature of the *station 15* receive-queue histograms appears is in the last plot (the ``LastFragment/1500'' plot), in which there is a small peak centered at 8 milliseconds and extending from about 7 to 9 milliseconds. These delays between successive 8-Kbyte messages appear to be caused by disk accesses: *station 15*'s receiver activity consists of packets coming mostly from its file server.

Finally, as an example of the histograms produced by a less utilized workstation, in Figure 3.12 we display graphs of interarrival times for client *station 11*. These plots were based on roughly four-hour traces rather than 100 000 interarrival times. Since the traffic of this workstation is highly unbalanced between the transmit and receive queues, had we considered only 100 000 arrivals for each queue, we would have covered two quite different intervals of real time. Here, also, we have truncated the plots at the interarrival time of 1 second because there were quite a few long interarrivals that would have excessively compressed the more interesting areas of the histograms.

The swollen intermediate part and the absence of the typical high peaks between 1 and 2 milliseconds in the top graph, representing the interarrival times between packets transmitted by *station 11*, indicates that the transmitter did not transfer many data. The NFS protocol caches for 30 seconds file descriptor information (see Appendix A), which then needs to be revalidated—a process that generates medium-sized interarrival times. Since NFS is a request-response protocol, the receiver graph also shows a developed intermediate area, if compared with Figures 3.7 and 3.9. Like *station 16*, *station 11* is not a workstation on which to base typical-traffic models.

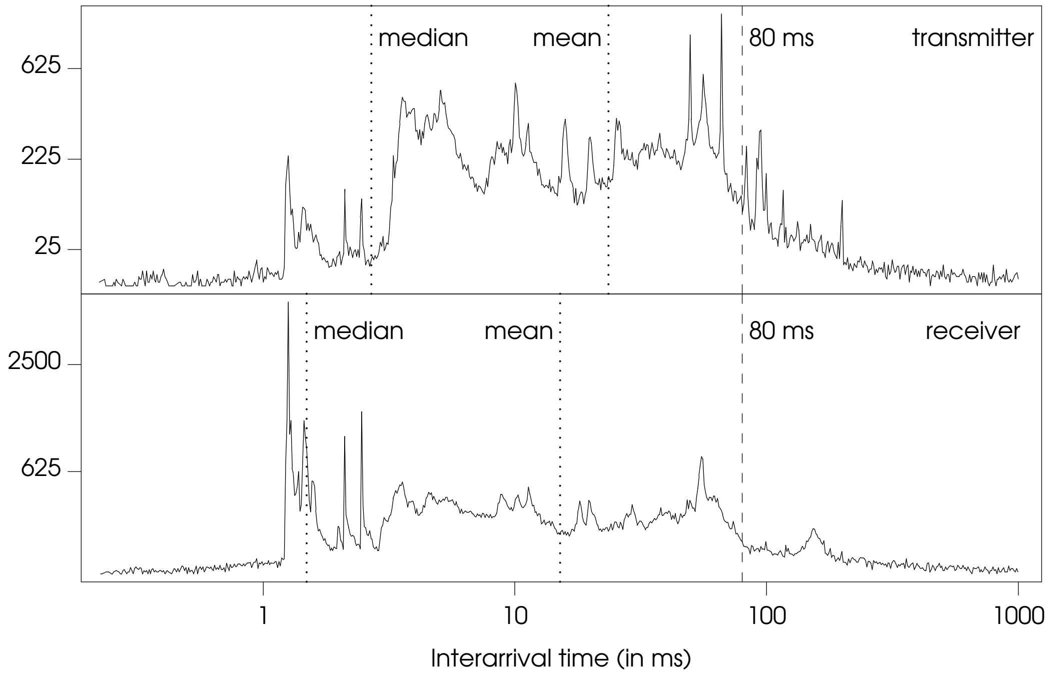
The histograms of interarrival times that we have discussed in this section are related to the marginal probability distributions of interarrival times. Alone, they do not fully describe the stochastic processes we are studying; it is necessary to represent the joint probability structure of each of these processes, that is, for interarrival times,

$$F_{X_1 X_2 \cdots X_n}(x_1, x_2, \ldots, x_n) = \Pr(X_1 \le x_1, X_2 \le x_2, \cdots, X_n \le x_n) \tag{3.2}$$

for all n.

Using joint distribution and density functions is very complicated; however, since most of the properties of stochastic process that are of interest in the description of packet arrival processes depend only on the first- and second-order joint moments [8, Chapter 6], we will complement the histogram analysis with the study of the

FIGURE 3.12. HISTOGRAM OF INTERARRIVAL TIMES – CLIENT *STATION 11*



correlation properties of packet arrival processes. We will carry out the correlation analysis after discussing, in the next section, whether the arrival processes are stationary.

V. Stationarity of Packet Arrival Processes

In this section we determine whether the stream of packet arrivals of a single user workstation constitutes a stationary point process. A process is (strictly) stationary if its joint distribution (3.2) is invariant under translations in time (or serial number), that is, formally, if

$$F_{X_{i+1}X_{i+2}\cdots X_{i+n}}(x_{i+1}, x_{i+2}, \ldots, x_{i+n}) = F_{X_{j+1}X_{j+2}\cdots X_{j+n}}(x_{j+1}, x_{j+2}, \ldots, x_{j+n})$$

for all i, j, and n. The property of stationarity is a highly desirable one for a point process: stationary processes benefit from a wealth of analysis techniques and lead to simpler models. Unfortunately, as is often the case, only rarely do ``real-life'' processes conform to this simplifying property.

Requiring that the marginal and joint probability distributions be time invariant is quite restrictive. Since we will base our descriptions of packet-arrival point processes only on first- and second-order properties, we will also choose a weaker definition of stationarity: a process is stationary if its first- and second-order descriptors are time invariant. In addition, since we will work both with count and interarrival time descriptions, we observe that the two are complementary aspects of an arrival process: a count process and an interarrival time process. The count and interarrival time processes are related because the sum of the counts in an interval of size t is less than or equal to k if and only if the sum of k interarrival times is larger than t. If we assume that a process has a point at the origin, we can write

$$\Pr(\sum_{i=1}^{n} c_i \le k) = \Pr(\sum_{j=1}^{k} X_j > t) \ .$$

Although from a strictly probabilistic viewpoint, if either of these processes is stationary, the other is not necessarily so (for details see [19, page 67]), from a practical viewpoint we will assume that the stationarity of one of the two processes will imply the stationarity of the other.

Classical time series analysis considers departures from the assumption of stationarity that have the form $X_i = Y_i + Z_i$, where the series Z_i is stationary and Y_i is not. If the series Y_i varies slowly with respect to Z_i, it is called a *trend*. A good practical rule regarding stationarity is given by Granger [32], who defines ``trend in mean'' as comprising all frequencies whose wavelengths exceed the length of the time series. This definition is important because it sheds light on the notion of ``relevant time scale'' that we will introduce shortly.

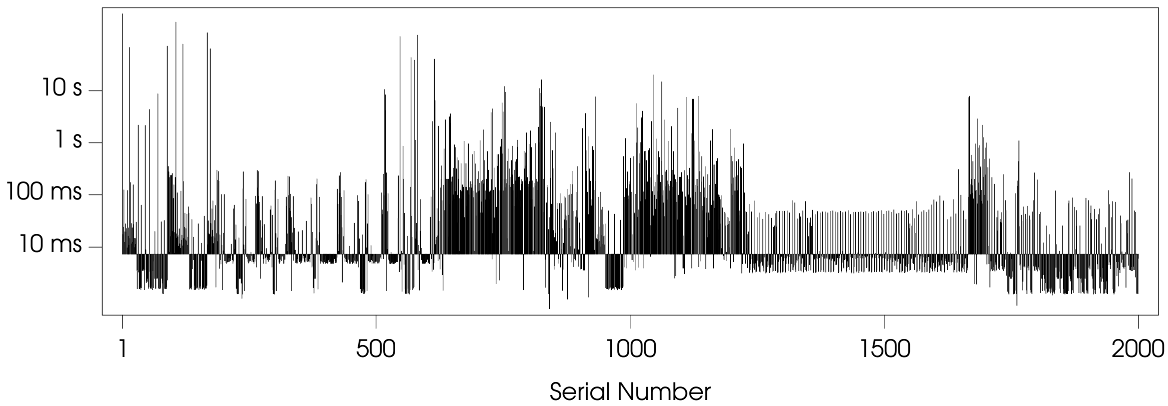
Series with trends can be dealt with by transforming the data so as to remove the trend. A typical transformation is by way of a linear filter:

$$W_i = \sum_u a_{i-u} X_u \ .$$

If the filter is high-pass, the low frequency components, i.e., the trends, in the original series are removed. A special high-pass filter can be obtained by differencing the series—an approach stressed by Box and Jenkins who maintain that a time series should be differenced until it becomes stationary [10]. (The differencing operation replaces two successive points in a time series with their difference.)

Another technique used to remove trends in classical time series analysis is by means of an interpolating curve. The differences between the curve and the original series are then taken, the residual providing estimates of the local variations. However, this is feasible only when the number of points in the series is small.

FIGURE 3.13. TIMES BETWEEN SUCCESSIVE ARRIVALS



The use of data-transformation techniques is not advisable with our arrival processes because much of the information we need in order to correlate properties of the processes to protocol and system time constants would be lost. Moreover, nonstationarities in our arrival processes do not satisfy the definition of trend given above. As an example, in Figure 3.13 we plot 2000 interarrival times, amounting to almost half an hour of real time, for the receive queue of *station 11*. On the vertical axis we report the interarrival times through a logarithmic transformation and on the abscissa we plot the serial number. An artificial zero line has been placed at the value corresponding to the median of this series; in this way, we can distinguish graphically short interarrival times embedded into sequences of long ones, since the former are printed toward the bottom of the graphs and the latter toward the top. As clearly seen in this figure, the sequence consists of segments, which we call phases, ranging from a few units to a few hundred interarrivals, in which the characteristics of the arrival process appear constant. Transitions between phases occur abruptly, that is, with high frequency components, thus, these transitions could not be classified as ``trends'' according to Granger's definition.

The question of whether a time series is stationary can also be a question of time scale. For example, all would agree that the load patterns in Figure 3.13 seem, to visual inspection, nonstationary. Yet, this particular user might follow the same work patterns and produce roughly the same workload in successive half hours. Or the work (and network load) patterns might follow a daily or weekly cycle. Therefore, if we looked at a period of several days, instead of just at half an hour, the data might look more clearly stationary.

There are at least two time scales of interest in our study. First, the busy/idle time scale about which we have already commented in the previous sections. This involves time intervals from many seconds to many minutes during which the network load generated by a workstation changes appreciably. A second time scale of interest, one over smaller time intervals, ranging from a few milliseconds to a few seconds, is one over which the packet rate of the sender's queue changes appreciably. A fine time scale is more important than a coarse one in the context of our study of the variability of packet arrival processes that are produced and consumed in an interactive system, in which response time is the key performance index. Hence, for us the ``relevant time scale'' will be the second: a time scale that will capture the queuing dynamics of packet arrival processes.

An important consequence of our approach is that we can drop long interarrival times since they will not affect the performance of the system over small time periods. In particular, we may exploit this in order to remove nonstationary components from the packet traces. It is clear that a nonstationary sequence at the coarser time scale will result in a nonstationary sequence at the finer time scale. If we, instead, restrict our attention to a subset of the arrival stream, for instance to a subset corresponding to a region of high network load, the resulting processes at the finer time scale may be stationary. We will explore this idea in our study of busy periods.

VI. The Busy Periods

In this and the next section, we will focus exclusively on busy periods of packet transmission, as defined below. Our methodology will be to extract from the full traces the sets of interarrival times occurring during busy periods, and to treat these as a single sequence. This approach is analogous to our extraction from the aggregate traces of the packets transmitted by a particular workstation and computation of their interarrival times: in both instances, we narrow the objects of our stochastic analysis to certain subsets of the general traces.

Our first step is to define the subset consisting of *busy periods*. A busy period could simply be defined as a period during which a user is active at the workstation. But drawing the line between busy and idle spans on this basis is far from satisfactory. In a long compilation task (or any long computation accessing the file system) that proceeds without user supervision, CPU and network activities are not related to the presence of a user at the workstation. Conversely, it is quite possible for a user to be active, without generating any network traffic. A more satisfactory criterion for the definition of busy period would be interarrival time length: a workstation is deemed to be busy when the interarrival times at its send and receive queues are shorter than a specified length.

To choose the cut-off point, we return to the histograms of interarrival times that we previously examined in Section IV. In each of those figures, which we have truncated at 1000 milliseconds, we see that at approximately 80 milliseconds the curve drops quite noticeably. (Dashed lines have been drawn at the 80 millisecond mark.) The longer interarrival times do not pop up randomly in the time series, but tend to occur in clusters during periods of low activity such as when a user logs off or leaves a workstation idle. Typically, only less than eight percent of the points occur after the 80 millisecond mark; yet, because of their large values and their occurrence in clusters, this small fraction of interarrival times accounts for most of the nonstationary behavior of a workstation. Hence, we chose 80 milliseconds to be the cut-off point: whenever the interarrival time exceeds this value, the machine is not deemed to be busy. We experimented with a few other values for the cut-off point in the range from 60 to 100 ms; no significant difference was evidenced and we conclude that the sensitivy of the following analyses to the 80 millisecond mark is minimal.

VII. Autocorrelation Coefficient Series

Correlation is a measure of causation in the data, and is often (but not always) associated with cause-effect phenomena. For example, the transmission of a packet may generate other transmissions within short, predictable times. We have already seen that this is the case for packet fragmentation of 8 Kbyte NFS messages. As another example, because of packet serialization over the network, there is a causal relation between interarrival time and packet length, since transmission of one packet must be completed before the next packet can be sent. Correlation coefficients provide a quantitative measure of the linear correlation among successive elements of a series.

The sample autocorrelation function of a series is defined as the (auto)covariance function divided by the variance of the series:

$$\rho_k = \frac{\mathrm{cov}(X_i,\, X_{i+k})}{\mathrm{var}(X)} = \quad (k = \cdots, -1, 0, 1, \cdots),$$

where the index k is called the *lag* and the covariance is by definition

$$\mathrm{cov}(X_i,\, X_{i+k}) = \mathrm{E}\big((X_i - \bar{X})(X_{i+k} - \bar{X})\big)\ .$$

Since the covariance at lag 0 is equal to the variance, the autocorrelation coefficient at lag 0 for any series is always 1.

It is clear that a sequence of interrival times all smaller or all larger than the mean will result in a positive contribution to the autocorrelation coefficients. Since it is quite common in packet arrival processes to find relatively extended sequences of short or long interarrivals, we would expect to find positive correlation coefficients.

It is rather difficult to establish confidence intervals for the series of autocorrelation coefficients. We can, however, define the y-axis interval within which, with some probability α, all correlation coefficients should lie were successive elements of the series mutually independent. If a significant number of correlation coefficients is found outside this interval, we will consider the series correlated and discard the assumption of independence.

If a sequence is sampled from a random process so that the samples are independent and identically distributed, all autocorrelation coefficients should be 0 except for the coefficient at lag 0, which is always 1. In this case, Kendall and Stuart [45] show that the mean and the variance of the ρ_k's are

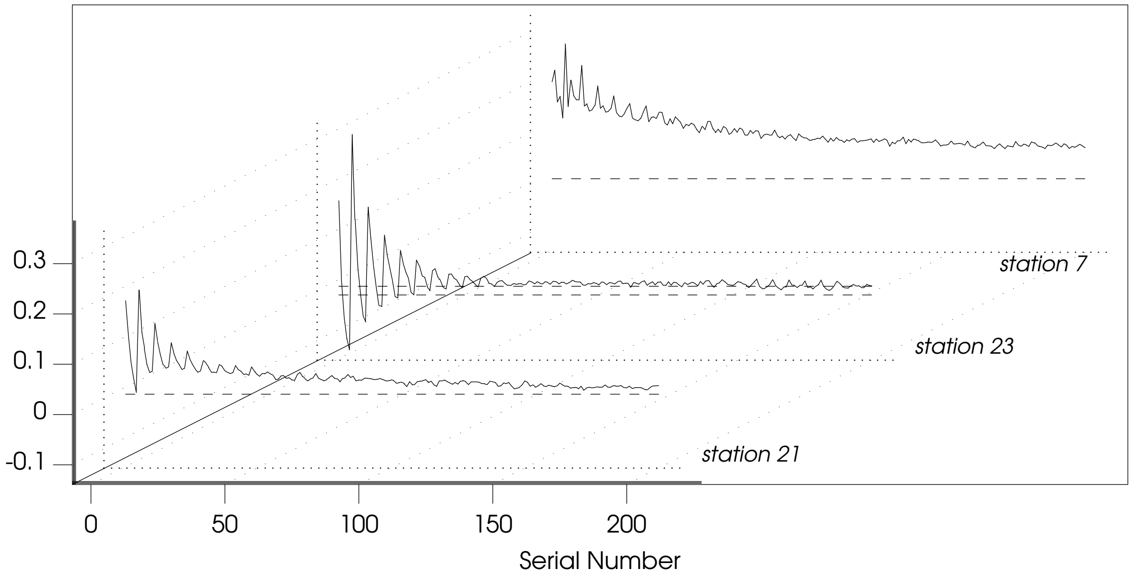
$$\mathrm{E}(\rho_k) \approx -\frac{1}{N} \quad \text{and} \quad \mathrm{var}(\rho_k) \approx \frac{1}{N}\ ,$$

where N is the cardinality of the series. They also show that, under weak conditions, ρ_k is asymptotically normally distributed. Therefore, the $100(1-\alpha)$ confidence limits for the autocorrelation coefficients are $-1/N \pm z_{\alpha/2}/\sqrt{N}$, where $z_{\alpha/2}$ is such that, for the standard normal distribution function, $\Pr(|Z| > z_{\alpha/2}) = \alpha$.

In Figure 3.14 we show the series of autocorrelation coefficients (sometimes called *correlogram*) for the send queues of three file servers, *station 21*, *station 23*, and *station 7*. In this figure we have drawn as dashed lines the confidence intervals that allow us to reject the assumption of independence with a 99 percent certainty. While the correlograms of *station 21* and *station 7* are always positive (accordingly, we have only drawn the upper limit of the confidence bands), *station 23*'s correlogram has some negative values. This is an indication (even after the truncation of interarrival times) that the series contains some sequences of alternating short and long interarrival times, which are often artificially produced by the software monitoring tools that we mentioned in Section II and are revealed only when users are idle. We could remove from the series such sequences, but we are reluctant to manipulate the data more than we have already done. We will keep in mind, however, that some interarrival time series, such as that of *station 23*, contain sequences that may not be fully representative of user generated busy periods.

In all three workstations there are periodicities that generate small spikes at lag values multiple of six. If we recall the discussion of Figures 3.10 and 3.11 in Section IV, in which we showed that the interarrival times classified as ``LastFragment/1500'' were more or less uniformly distributed, we realize that in groups of successive 8-Kbyte transfers with high probability the ``LastFragment/1500'' interarrivals are larger than the others generated by the message fragmentation. Thus, in sequences of successive 8-Kbyte

FIGURE 3.14. AUTOCORRELATION COEFFICIENTS FOR THE SEND QUEUES OF THREE FILE SERVERS



message transfers, we have longer interarrival times spaced exactly six interarrivals apart. The impact of these is reflected upon the shapes of the correlograms. Since there are fewer and fewer of those groups, as the number of successive groups increases, the size of the periodic spikes decreases as the lag increases.

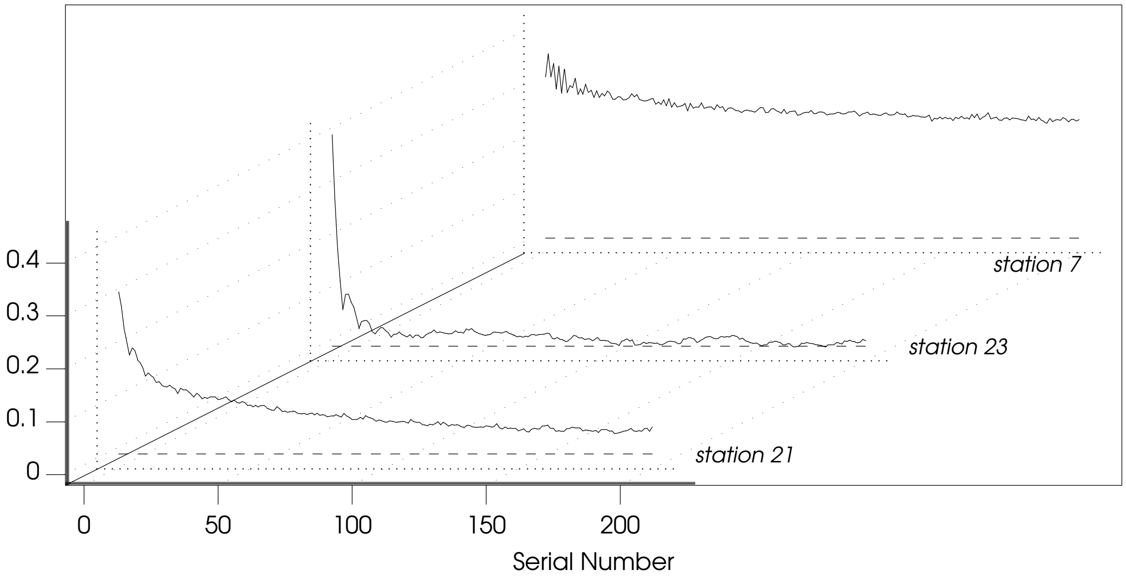
As expected, the series of correlation coefficients decreases towards zero as the lag grows. Although in the cases of *station 21* and *station 23* the coefficients approach levels indicating that after some delay the correlation effects are lost, *station 7* shows a significant long-term correlation. Since all machines run the same type of software and communicate using the same protocols, we attribute *station 7*'s long-term correlation to nonstationary components in its arrival process.

Finally, note that, had we computed the correlograms using all interarrival times, including those larger than 80 milliseconds, the details that emerge in these graphs would have disappeared, overwhelmed by the dominant nonstationary components.

In Figure 3.15 we show the correlation coefficients of the receive queues for the same three servers. Three observations deserve to be made. First, unlike the plots of Figure 3.14, those in Figure 3.15 are not spiky. Since the receive queue of a server collects the aggregate traffic from all of the server's clients, various groups of successive 8-Kbyte messages mix together, resulting in randomization of packet interarrival times.

Second, the correlation coefficients of *station 7* decrease towards zero even more slowly than *station 7*'s send queue coefficients, suggesting that nonstationary components are even more dominant here. Table 3.3 indicates why this may be so. According to that table, *station 7* receives 20 percent fewer packets and 50 percent fewer bytes than it generates. This suggests that *station 7*'s average receive-queue packet is smaller. Smaller packet size in turn suggests that there are fewer long data-transferring packets, which are associated with short interarrival times; thus, the receive queue has fewer short

FIGURE 3.15. AUTOCORRELATION COEFFICIENTS FOR THE RECEIVE QUEUE OF THREE FILE SERVERS



interarrival times than the send queue. We have verified that the fewer long packets are more unevenly distributed in the traces for which we computed the autocorrelation coefficients. This uneven distribution is the source of nonstationarity.

Finally, the short-term correlation of the receive queues is generally higher than that of the corresponding send queues. Such higher short-term correlation is produced by longer sequences of long interarrival times in the receive streams. Table 3.3 indicates that the receive queues of each of the file servers *station 21*, *station 23*, and *station 7* contain a larger proportion of small packets. In each case, the average packet size is smaller in the receive queues than in the send queues. This should be expected, since client workstations have to page in programs, an activity that requires the servers to transmit large packets. Another source of large packets are file-system read and write operations. Since reads, which are transmitted by servers, outnumber writes, which are received, the send queues of servers contain a larger number of large packets.

In Figures 3.16 and 3.17 we show the correlograms for the send and receive queues, respectively, of four client workstations: two diskless ones, *station 13* and *station 25*, and two with disks, *station 8* and *station 11*. *Station 13* and *station 25* display very similar correlation series, characterized by high initial values that subsequently decline steeply. *Station 8* with its local disks generates long sequences of medium-sized, remote-procedure-call packets. These sequences, which produce long interarrival times, occur within the time series in clusters, creating long-term autocorrelation. In *station 11*'s case, we have another indication that the data is highly nonstationary (compare with Figure 3.12 in Section IV).

Interpreting a correlogram is complicated. Often, there are spurious correlations, and the presence of nonstationary data components alters the shapes of the correlation curves. Using as point processes for our autocorrelation study what we have called busy

FIGURE 3.16. AUTOCORRELATION COEFFICIENTS FOR THE SEND QUEUES OF FOUR CLIENT WORKSTATIONS

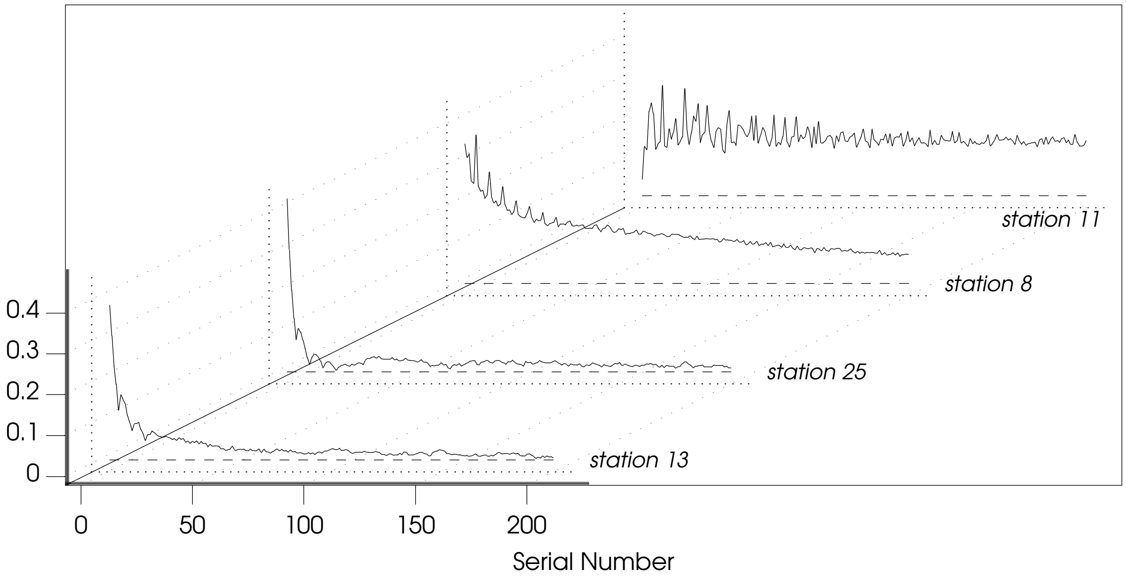
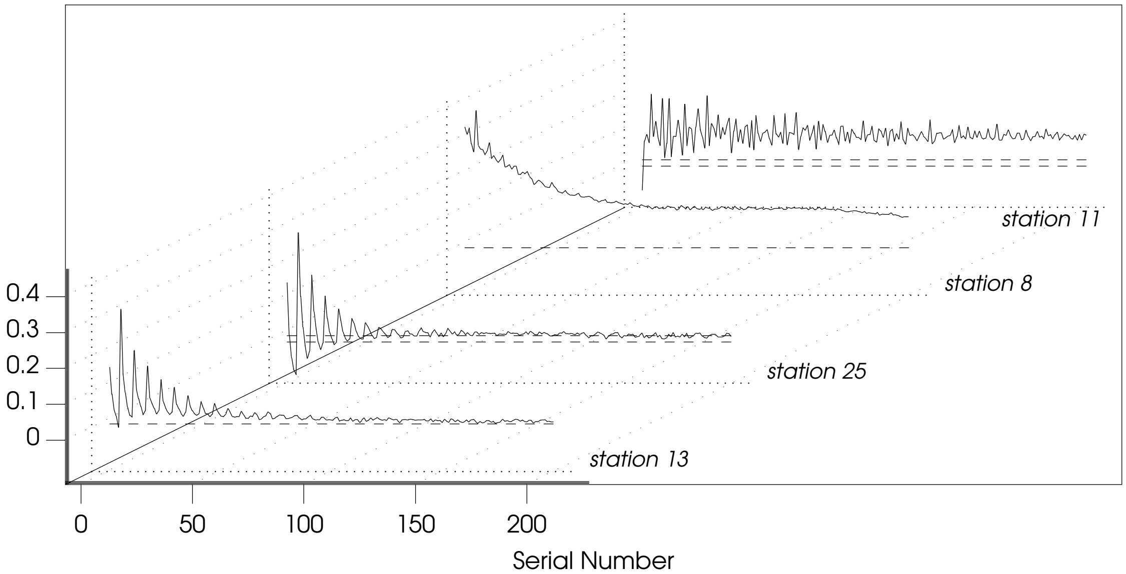


FIGURE 3.17. AUTOCORRELATION COEFFICIENTS FOR THE RECEIVE QUEUES OF FOUR CLIENT WORKSTATIONS



periods has simplified the problem of dealing with ``real-life'' processes, for many packet arrival processes (notably most diskless workstation processes) become stationary. Others are not quite stationary, but in most cases the range of ``variability'' has been reduced. Although we could have used a more aggressive technique to remove nonstationarities, our method has the fundamental advantage of not transforming the timing relationships between successive arrivals, which allows us to relate more directly interarrival times to protocol features. In Chapter 4, we will discuss a new way of looking at autocorrelation coefficients through indices that we will associate with the variability of packet arrival processes. We will also continue there our discussion of the effects of nonstationarity.

VIII. Concurrency on the Network

Analysis of the behavior of single workstations must take into account interactions between machines, if any. Without a clear understanding of such interactions, models of packet arrival processes cannot be adjusted to encompass network configurations other than the one studied. There are three major sources of interaction among workstations: first, direct communication among them, since the protocols used for this communication are request-response; second, concurrent access to the same file server; and third, serialization in the network. During all of these types of interaction there is simultaneous activity by two or more workstations. Thus, we will begin by looking at the number of concurrently communicating machines. In the next section, we will consider the effects of shared resources.

The histograms in Figure 3.18 show the number of 1-second intervals during which a given number of workstations were active. The chosen interval length of 1 second is short enough to be meaningful for the queuing behavior of transmit and receive queues, yet long enough to allow time for the statistical interaction among machines to develop. Figure 3.18 is built from a subset of all packets recorded in our traces; in order to compare more directly this figure to the following two figures, we consider only those machines in the Sun Engineering network that communicate with the six major file servers: *station 5*, *station 7*, *station 9*, *station 21*, *station 23*, and *station 28*. Between them, these machines account for 82.4 percent of the total traffic. (The remaining 17.6 percent is primarily traffic between pairs of clients as well as a small amount of internetwork traffic.)

FIGURE 3.18. HISTOGRAMS OF THE NUMBER OF 1-SECOND INTERVALS IN WHICH WORKSTATIONS WERE ACTIVE

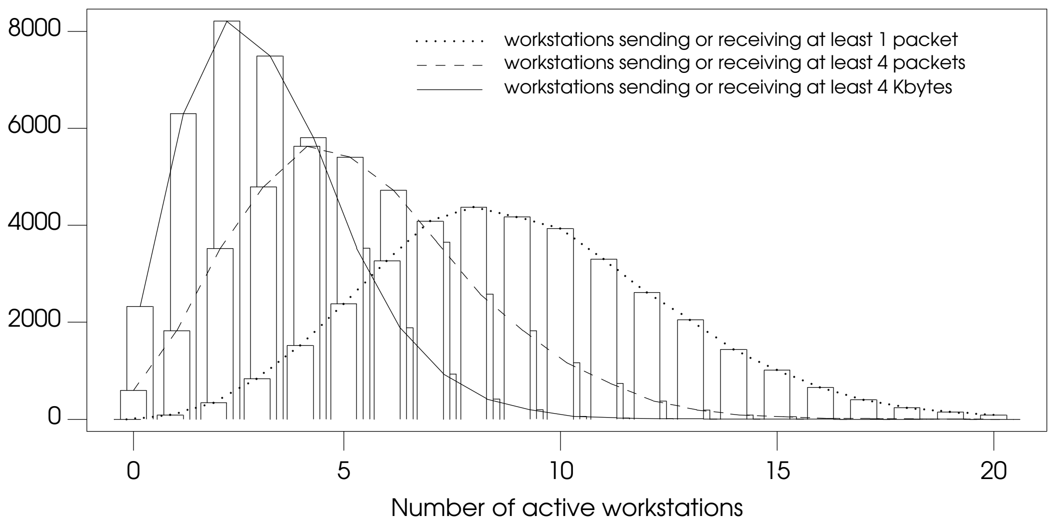


Figure 3.18 consists of three separate sets of histograms, reflecting three different criteria for defining an ``active'' workstation. In the first set, a workstation was considered active if it sent or received at least one packet during a 1-second interval; in the second

set, if it sent or received at least four packets during a 1-second interval; and in the third set, if it sent or received at least 4 Kbytes during a 1-second intervals. The second and third tests were devised to exclude workstations that transmit only occasionally or periodically but infrequently. Interpolation curves for the three sets have also been drawn. The median for the curve for the first set (indicated by a dotted line) is between 9 and 10 workstations; for the second set (indicated by a dashed line), between 5 and 6; and for the third, approximately 3.

In Figure 3.19 our unit of study is the individual file server rather than the single workstation. We use the same three criteria for an active machine as in Figure 3.18. Finally, Figure 3.20 presents histograms for workstations in communications with the busiest file server, *station 21*.

FIGURE 3.19. HISTOGRAM OF THE NUMBER OF 1-SECOND INTERVALS SHOWING ACTIVITY OF FILE SERVERS

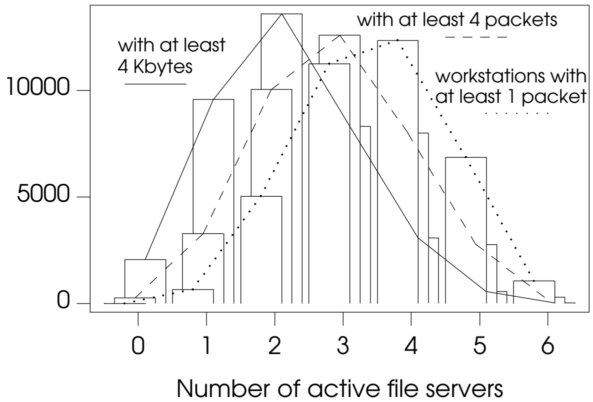
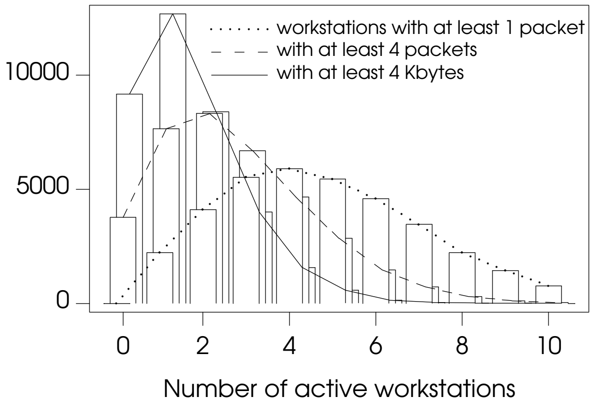


FIGURE 3.20. HISTOGRAM OF THE NUMBER OF 1-SECOND INTERVALS SHOWING ACTIVITY OF BUSIEST FILE SERVER



All these graphs show a consistent view: there is only a moderate amount of interaction among workstations and between workstations and file servers. For instance, notice that the statistical norm in Figure 3.20 in the case corresponding to the criterion that calls for 4 Kbytes to be exchanged before a machine be considered active is 1. Nevertheless, it is necessary to investigate whether this partial interaction is reflected in the second order statistics.

It is also worth estimating the probability that k or more consecutive packets are exchanged between two machines. We know that machines tend to produce bursts of messages, thus, low values for this probability distribution would also indicate that there is network contention.

In Figure 3.21 we display the sample probability that k packets are exchanged (in either direction) between two machines. (Here also we considered only those machines in the Engineering network that communicated with at least one of the six major file servers, that is *station 21*, *station 7*, *station 23*, *station 28*, *station 5*, and *station 9*.) Of course, for this probability to be meaningful, we have assumed that the distribution is stationary, when in reality one would expect that, for a given value of k, the probability becomes higher as the average network load decreases. If we call this probability $\Pr_{AB}(S = k)$, we have

$$\Pr_{AB}(S = k) = \Pr\left[\, p_n \in \{A, B\},\ p_{n+1} \in \{A, B\},\ \cdots\ ,\ p_{n+k-1} \in \{A, B\},\ p_{n+k} \notin \{A, B\} \right]$$

in which p_n identifies the nth packet, and $\{A, B\}$ represents the set of packets exchanged between machines A and B, so that $\Pr_{AB}(S = 1)$ is the probability that the next packet is not between A and B. Then,

$$\Pr_{AB}(S \geq k) = \sum_{j=k}^{\infty} \Pr_{AB}(S = j) = 1 - \sum_{j=1}^{k-1} \Pr_{AB}(S = j) \tag{3.3}$$

is the probability that k or more packets are exchanged between two machines. Probability function (3.3) is displayed in Figure 3.22.

FIGURE 3.21. PROBABILITY THAT EXACTLY K PACKETS ARE TRANSMITTED BETWEEN TWO WORKSTATIONS

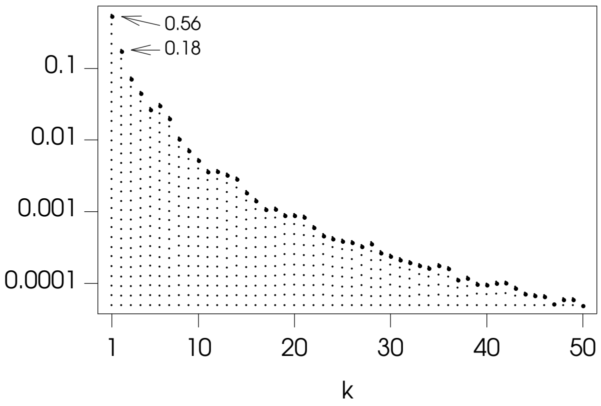
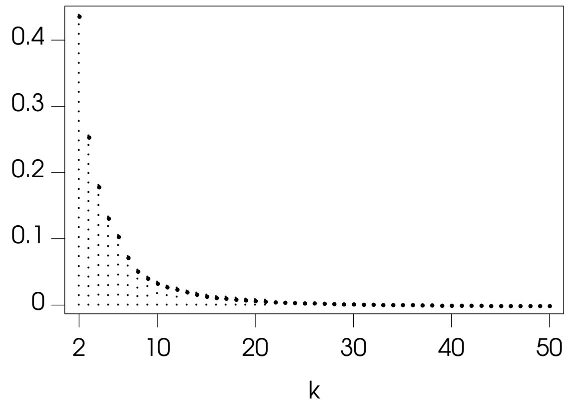


FIGURE 3.22. PROBABILITY THAT K OR MORE PACKETS ARE TRANSMITTED BETWEEN TWO WORKSTATIONS



The relatively low probabilities are additional evidence that some network contention is present. In the following section, we determine whether this network contention creates statistical interaction among workstation arrival processes.

IX. Statistical Interactions Among Workstations

Mutual interactions among personal workstations may have a significant impact on the packet arrival processes that they generate. Personal workstations, as opposed to a distributed system composed of several cooperating but autonomous nodes, which must share access to common data structures as well as other resources, contend with each other for only two types of resources: shared file servers and the shared network (in this context, ``network'' refers to the set of links, routers, and interfaces that are shareable). Although distributed applications can be built on top of personal workstations, there were few instances of distributed programs in use in the Sun system that we studied; most users accessed their workstations, the file servers, and the other resources independently of each other. Thus, interaction caused by mutual exclusion synchronization of distributed programs is not relevant. Rather, interaction arises from contention over file servers and the network. Both sources of interaction will be examined in this section.

Packet transmissions from autonomous machines originate independently of each other. They will, however, be serialized on the network whenever two stations begin to transmit simultaneously. Serialization obviously creates a causal relationship between successive arrivals. In addition, concurrent file server accesses are related by sharing of the server's CPU and disks. Yet, it is necessary to determine whether serialization and sharing result in significant interaction among workstations.

The technique we have chosen for capturing and describing possible interaction among machines is the analysis of the cross covariance (in the form of cross correlation) of various packet arrival processes detailed below. The covariance between two series X_i and Y_i of means μ_x and μ_y respectively is defined by

$$\mathrm{cov}(X,\ Y) = \mathrm{E}\big((X - \mu_x)(Y - \mu_y)\big)\ . \tag{3.4}$$

Since the covariance depends on the units in which X and Y are measured, we will use a normalized version of the covariance, the correlation coefficient, obtained by dividing Equation (3.4) by the product of the standard deviations of X and Y. It can be easily proven that the correlation coefficient between two series is a number between −1 and +1. Finally, we observe that the correlation coefficient at lag 0 between two (distinct) random series, unlike the autocorrelation for which the coefficient at lag 0 is always 1, will be in general different from 1.

The packet arrival processes that we will analyze fall into two basic categories, which we define as *opposite queues* and *parallel queues*. Opposite queues are a pair of arrival processes one at a send queue and one at a receive queue; parallel queues refer instead to either two send or two receive queues. Opposite queues can be classified into four types according to the machines they are associated with: 1) a single workstation (i.e., both the send and receive queues belong to the same workstation); 2) one client and one server; 3) two clients on different file server clusters; and 4) two independent servers.

In the single workstation case, correlation is to be expected: the nature of the transmission sent to the server determines the content and size of the response and the time it is sent. In the second case we are not sure whether to expect a causal relationship because several different clients contribute to the queue of the server. The third case considers whether or not there is correlation between opposite queues of two clients that use different file servers, allowing us to isolate the effects of the network on correlation. Finally, the fourth case, like the third, in which we consider two independent servers, helps to detect causation attributable to the network. In this last situation, we would expect a stronger causal relationship for two reasons: first, servers are more heavily used than clients, and, second, clients in one cluster often mount file systems from a server from another cluster of workstations. As a result, a client that loads a file from the second server in its memory may need to flush some of its memory pages to the swapping area, which is managed by the first server. Thus, in this example, there is a causal relationship between the send queue of the second server and the receive queue of the first.

Parallel queues can be classified into three types according to the relationship between the two machines to which the queues belong: 1) two clients from the same cluster; 2) two clients from two different clusters; and 3) two servers. Studying parallel queues is important because they replicate what occurs when processes are superimposed. In the case of two clients from the same cluster, we examine the interactions at the server and on the network of the two send (or two receive) queues. In contrast, in the case of two clients from different clusters, the queues' sole interaction occurs on the network; thus (as in case three and four for opposite queues), we can isolate the effects of the network interaction. Finally, it is also important to see how two servers behave with respect to the superposition of their queues.

We will study both opposite and parallel queues in terms of packet counts and interarrival times. Note that we cannot use subsets of measured series for these calculations. Since we are trying to see evidence of temporal cause/effect phenomena, we cannot remove a portion of the measured arrivals, as we did in the study of the autocorrelation coefficients. Instead, for each pair of arrival processes, we will include all arrivals in the time series.

Packet counts are computed by choosing a constant interval with which to divide the time axis, and counting how many arrivals occur within each interval. The choice of the interval over which to compute the packet counts is important and represents a trade-off between computational cost and accuracy: small intervals will increase cost, while large intervals will result in loss of burstiness information. In line with Cox and Lewis [19], we found that choosing the number of intervals roughly equal to the number of interarrival times gives good results. For the analysis in the rest of this chapter we have used intervals of 50 milliseconds for computing packet counts.

In the correlation computations, we use arrival streams generated by nine machines: three file servers—*station 21*, *station 23*, and *station 7*—and six clients. *Station 25* and *station 14* are clients of *station 23*. *Station 27*, *station 3*, and *station 13* are clients of *station 21*. *Station 11* is a client of *station 7*. *Station 25*, *station 14*, *station 3*, and *station 13* are diskless and thus depend on their servers for obtaining, in addition to general file services, the object code of programs that are executed, and swap space.

In all figures below, except for Figure 3.23, in which we needed more resolution, we show the first 200 correlation coefficients. The notation ``*station 3–>station 21*'' will be used to indicate the correlation curve between *station 3*'s send queue and *station 21*'s receive queue. Analogously, with the notation ``*station 3–station 21*'' we will indicate *station 3*'s and *station 21*'s parallel queues. Notice that we cover all cases: if there is a ``*station 3–>station 21*'' curve, there will also be the ``*station 21–>station 3*'' one. In each figure, we indicate with dashed lines the 95 percent confidence intervals for the null hypothesis that packet counts (or packet interarrival times) are not correlated (refer to Section VII for more details on the meaning of these confidence bands).

A glance at Figures 3.23 through 3.32 quickly reveals that all the queues, with the noticeable exception of those in Figure 3.23 and those in Figure 3.25, appear to be uncorrelated. There are several spurious correlation coefficients (for instance in the graphs of Figures 3.28 and 3.29), which are generated by the few large sample points present in the series. In addition, although not every autocorrelation curve is within its the confidence interval, in almost all cases the coefficients are very small in magnitude and display random fluctuations. Only opposite queues for packet counts of type 1, i.e., belonging to the same workstation (in Figure 3.23), and opposite queues of type 2 for packet counts (in Figure 3.25) show noticeable correlation. (In type-2 opposite queues, in which we correlate a queue of a client and that of its server, the correlation is not as pronounced as in type-1 queues, but we have indicated earlier that, since many clients contribute to the queue of the server, the causal effects may be reduced.) The

FIGURE 3.23. CROSS CORRELATIONS BETWEEN OPPOSITE QUEUES: PACKET COUNTS
SEND AND RECEIVE QUEUES ON SAME WORKSTATIONS

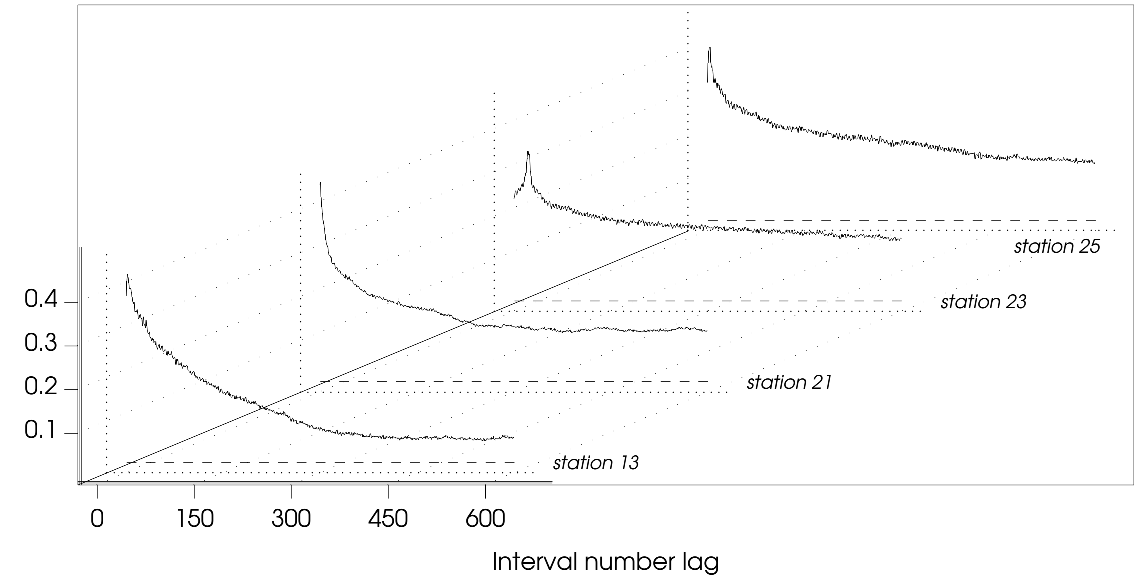
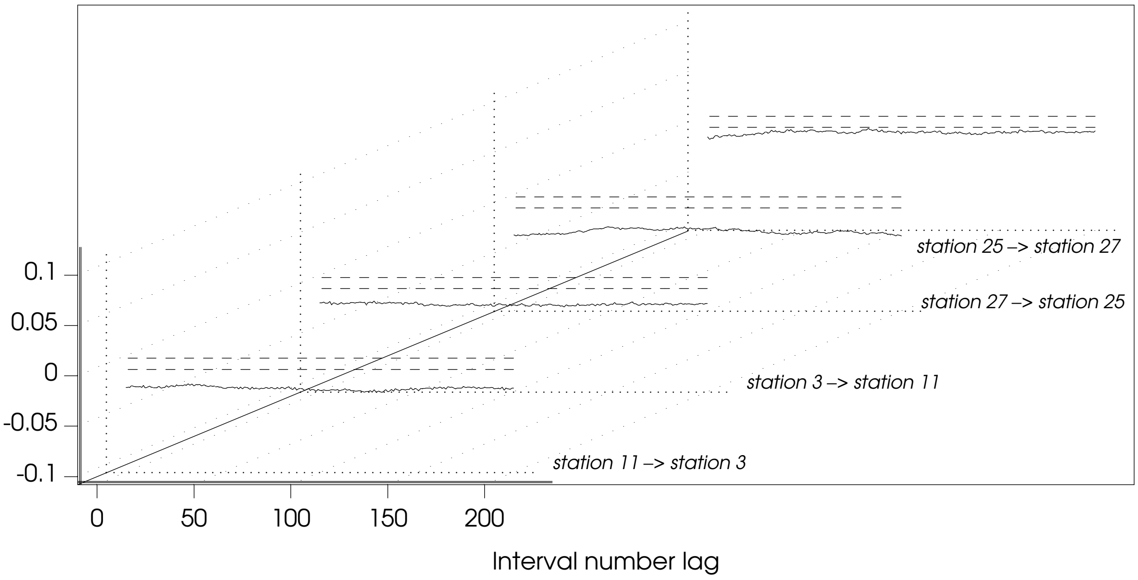


FIGURE 3.24. CROSS CORRELATIONS BETWEEN OPPOSITE QUEUES: PACKET COUNTS
SEND AND RECEIVE QUEUES ON CLIENTS OF DIFFERENT CLUSTERS



association detected between the send and receive streams of a workstation is attributable to the use, by many applications, of remote procedure call protocols, which suspend themselves until an answer to the sole outstanding message is received, resulting in interlocks between send and receive operations. In addition, since in systems that use request-response protocols operations are typically initiated by clients (and not by servers), pauses in a user's activity that result in long interarrival times in the send stream

FIGURE 3.25. CROSS CORRELATIONS BETWEEN OPPOSITE QUEUES: PACKET COUNTS
SEND AND RECEIVE QUEUES ON A CLIENT WORKSTATION AND A FILE SERVER

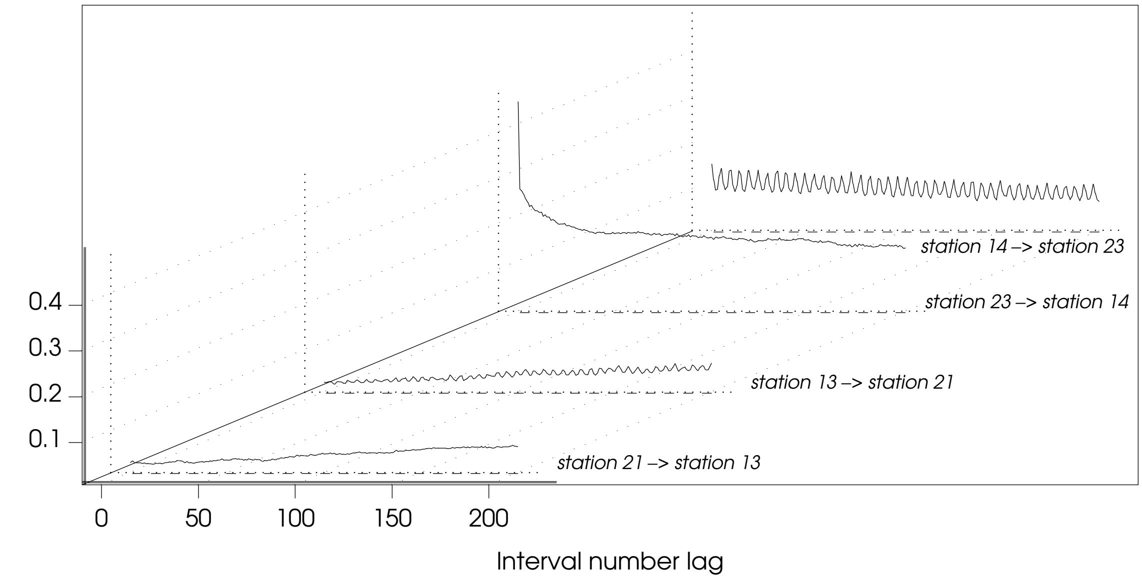
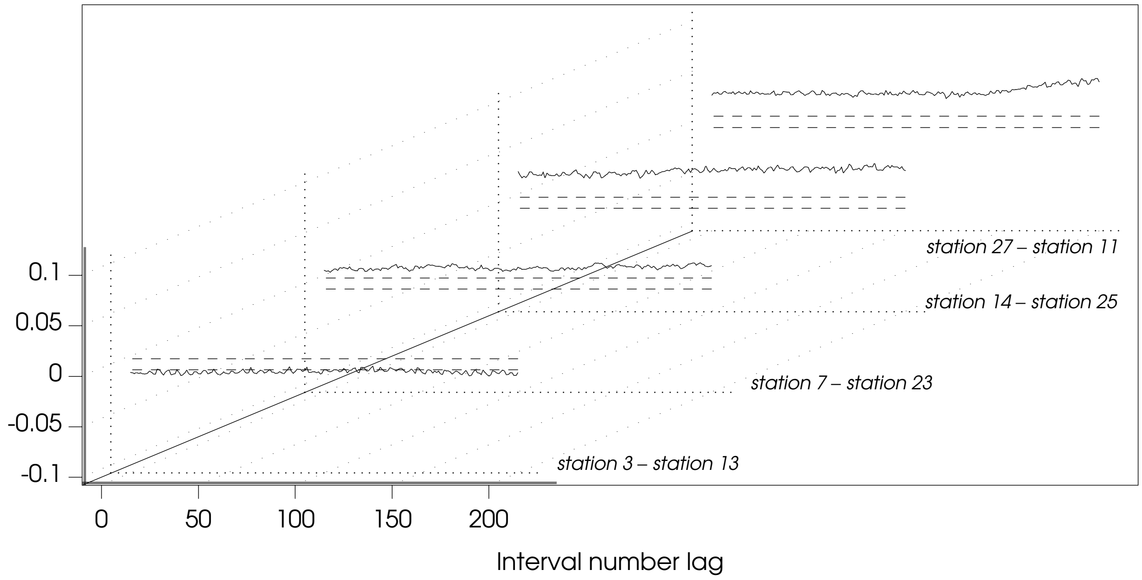


FIGURE 3.26. CROSS CORRELATIONS BETWEEN PARALLEL SEND QUEUES: PACKET COUNTS



are almost invariably associated to pauses in the receive streams.

In all other cases, if any causal relationships do exist, they are overwhelmed by the large variability of the arrival processes. In fact, it is the greater variability of interarrival time processes in comparison to that of packet count processes (as we have observed earlier, the range of the probability distribution functions for packet counts is rather limited) that washes out the correlation in Figure 3.28, which displays the cross correlations

FIGURE 3.27. CROSS CORRELATIONS BETWEEN PARALLEL RECEIVE QUEUES: PACKET COUNTS

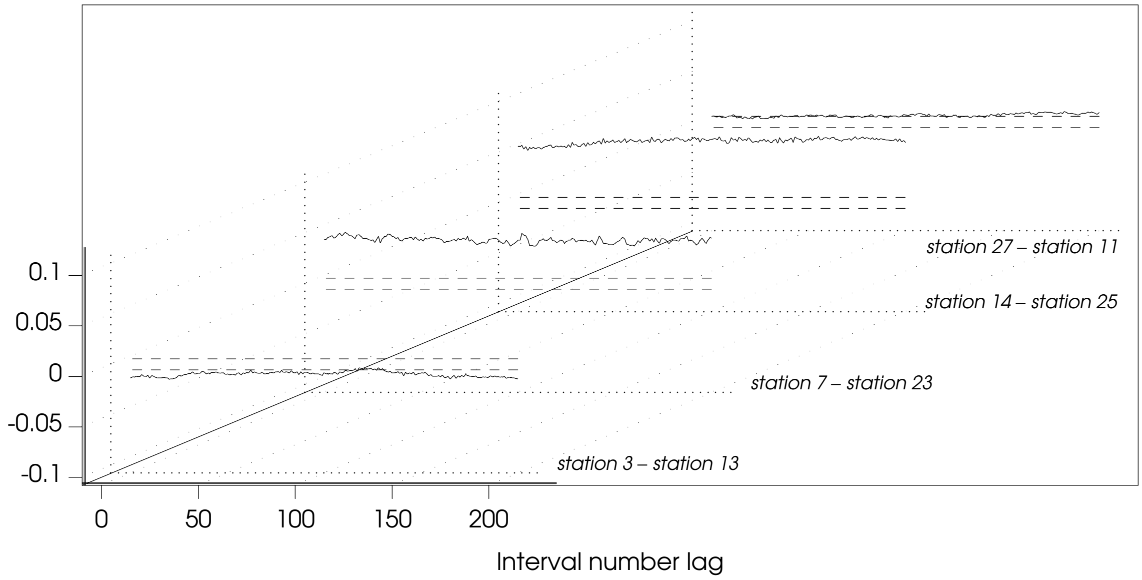
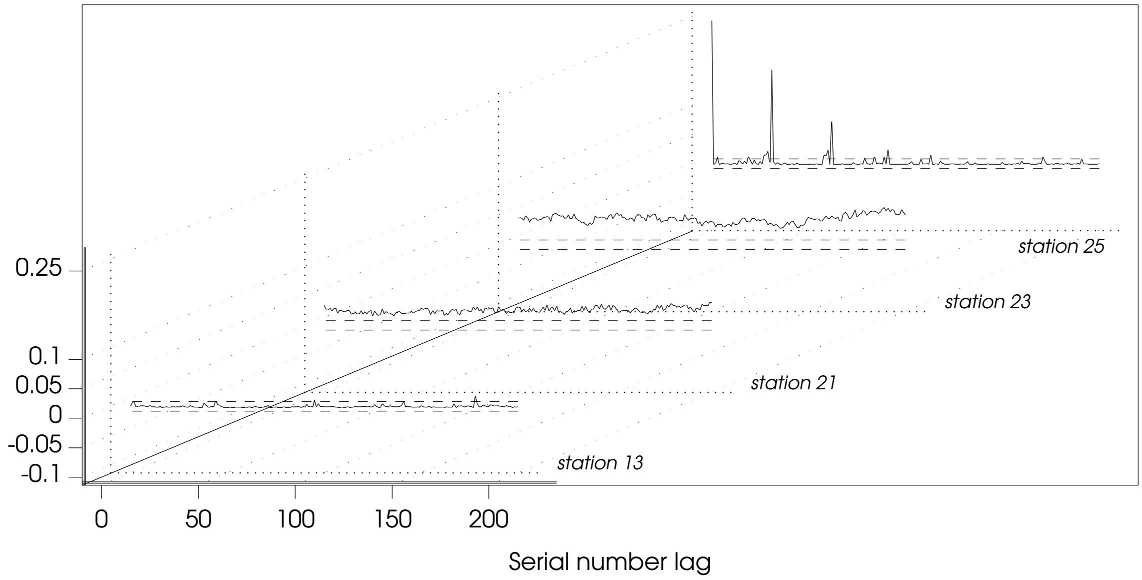


FIGURE 3.28. CROSS CORRELATIONS BETWEEN OPPOSITE QUEUES: INTERARRIVAL TIMES
SEND AND RECEIVE QUEUES ON SAME WORKSTATIONS



between opposite queues on the same workstation for interarrival time processes.

X. Summary

In this chapter we have analyzed the properties of the traffic generated and received by single client and file server machines. From the prospective of a macroscopic time scale, workstations produce network loads that can be divided into light (or

FIGURE 3.29. CROSS CORRELATIONS BETWEEN OPPOSITE QUEUES: INTERARRIVAL TIMES
SEND AND RECEIVE QUEUES ON CLIENTS OF DIFFERENT CLUSTERS

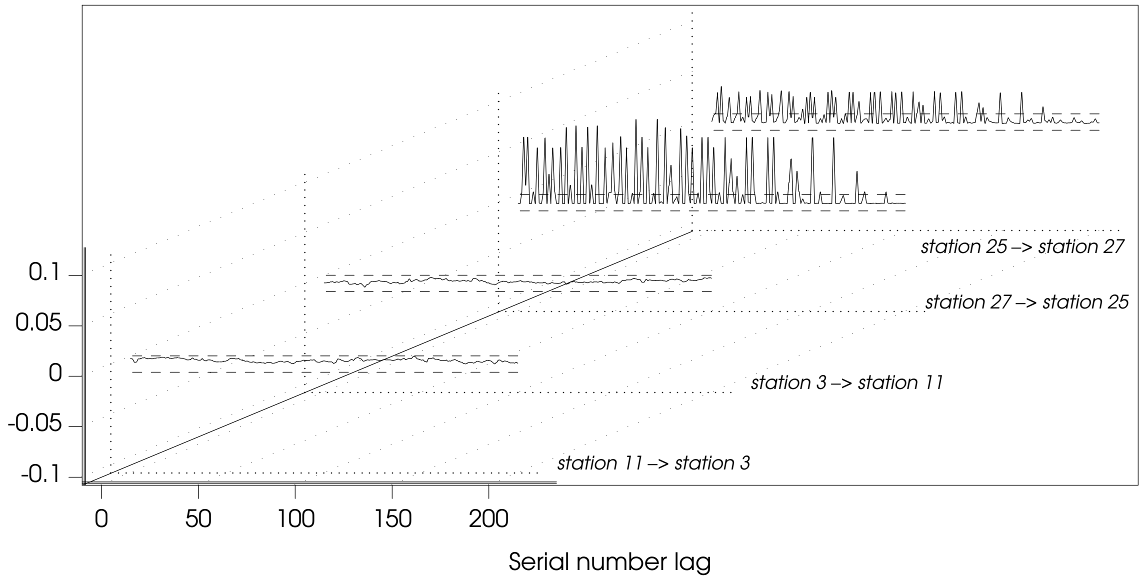
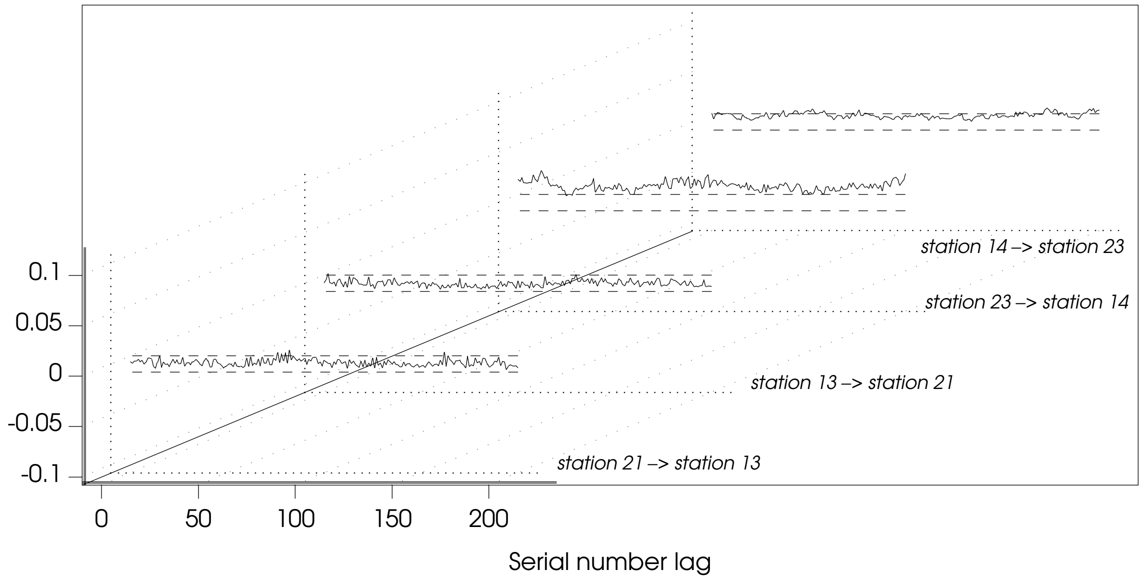


FIGURE 3.30. CROSS CORRELATIONS BETWEEN OPPOSITE QUEUES: INTERARRIVAL TIMES
SEND AND RECEIVE QUEUES ON A CLIENTS WORKSTATION AND A FILE SERVER



nearly zero) and heavy phases. We argued, in Section III, that traffic models of variability need only be concerned with what we have termed ``busy'' periods: intervals of time during which the arrival process produces a ``substantial'' arrival rate. In Section VI, we then considered what constitutes a substantial arrival rate. We concluded that the busy period should be defined as the interval of time during which interarrival times are no longer than 80 milliseconds.

FIGURE 3.31. CROSS CORRELATIONS BETWEEN PARALLEL SEND QUEUES: INTERARRIVAL TIMES

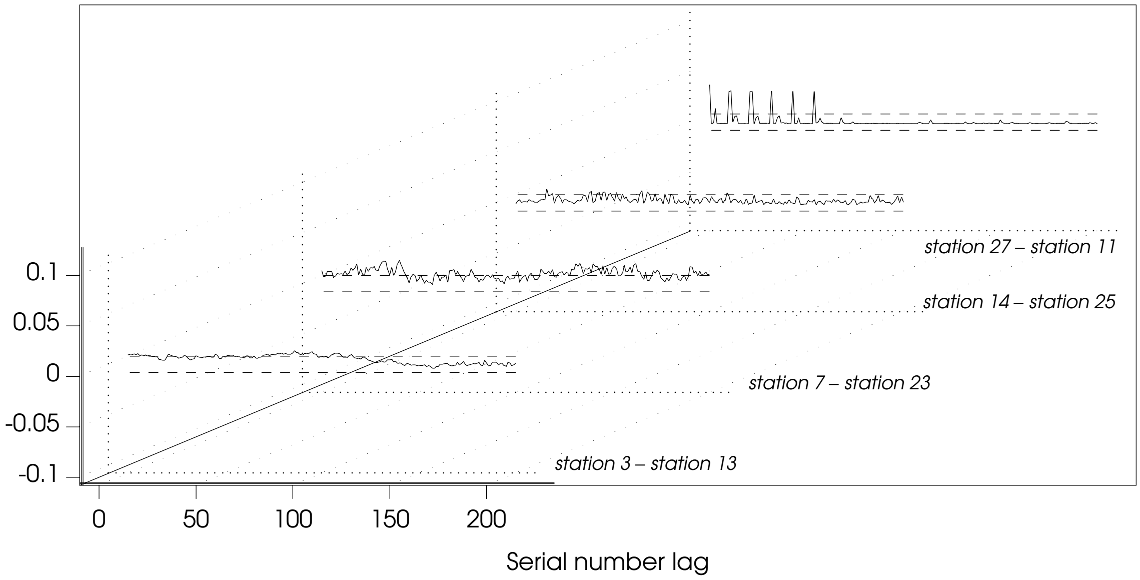
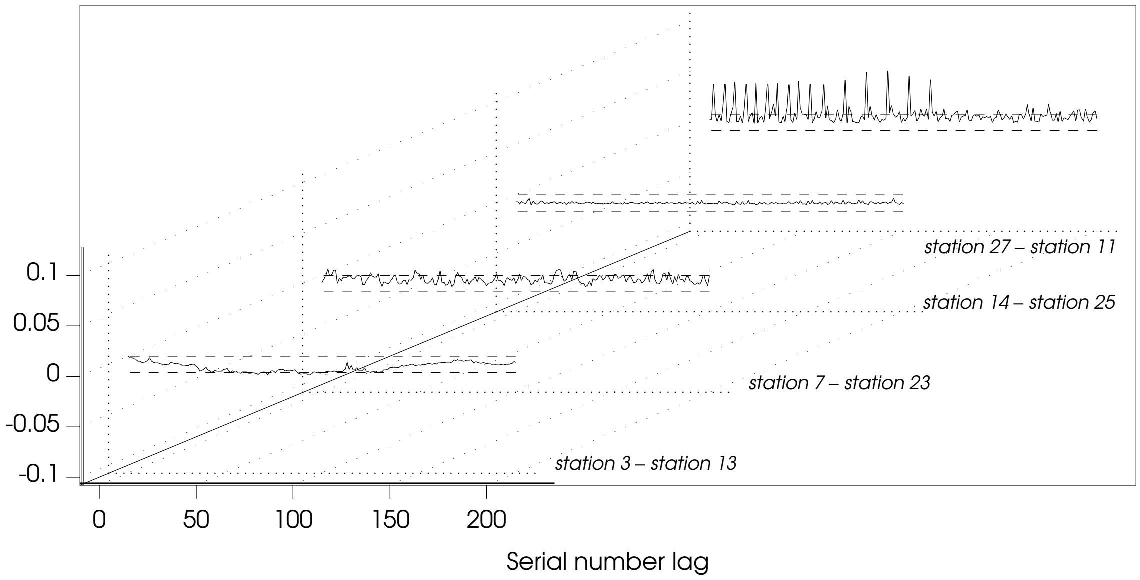


FIGURE 3.32. CROSS CORRELATIONS BETWEEN PARALLEL RECEIVE QUEUES: INTERARRIVAL TIMES



The major reason for focusing on busy periods rather than on the entire sequence of interarrival times is that busy periods are probably more stationary than the full series. However, upon closer examination, we discovered that, although in most cases busy periods appeared to be weakly stationary processes, there were also some that were clearly nonstationary.

We looked at histograms of interarrival times, which are related to the marginal probability density functions of interarrival times. The decomposition of the histograms by packet-length classes has shown that short interarrival times are produced by long packets, which are transmitted in quick sequence by the IP fragmentation layer. Sequences of shorter packets, produced individually by the NFS protocol layer, tend to generate long interarrival times. Typically, packet arrival processes from file servers and diskless machines are more stationary than those from machines with local disks. Whereas in file servers and diskless machines short interarrival times are more uniformly distributed because of the high communication demands of these machines, in machines with local disks we often observe long stretches of medium-sized packets and long interarrival times, which produce appreciable changes in the characteristics of the packet arrival processes.

We studied the autocorrelation structure of packet arrival processes during busy periods. Often, interpreting the series of correlation coefficients is not simple, but, whenever possible, we have tried to relate the shape of the series to protocol and system features.

We then considered the interactions among various types of packet arrival processes, and found that the only non-negligible statistical dependence occurs between the send and receive processes produced by the same workstation.