request-response behavior may influence congestion and transport mechanisms and should be included in simulation studies of these mechanisms.

Small packets, short conversations, and bidirectional flow all contribute to the traffic dynamics of the internetwork. These characteristics of current internetwork traffic could affect traffic segregation and oscillation studies [23, 24, 27].

3.5. Wide-Area Network Locality

Mogul reports strong locality of reference between pairs of hosts on a local area network [31]. This locality of reference means that certain hosts communicate more with one another than with other hosts. Does such locality of reference exist between host pairs or network pairs in wide-area internetworks? Figure 8 shows that it indeed occurs. For example, half of UCB TELNET conversations are directed to just 10 stub-networks.

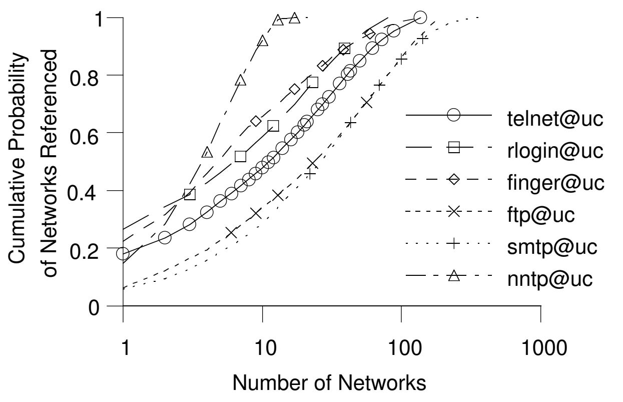
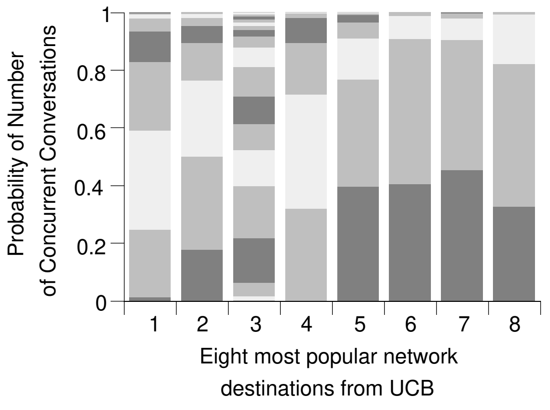


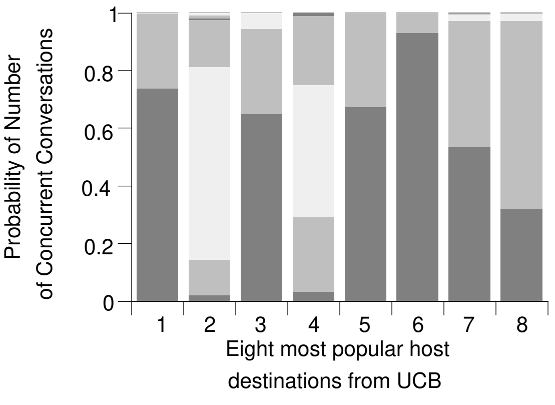
Fig. 8: Number of networks referenced by UCB.

Given network-pair locality on wide-area networks, we want to know how many concurrent conversations run between popular network-pairs. Figure 9a shows the concurrent conversations to UCB's eight most popular destination networks. In Figure 9, each band represents a number of concurrently running conversations. The band at the bottom of a bar represents the probability of finding zero on-going conversations. The next band up represents the probability of finding one on-going conversation. The third represents the probability of finding two simultaneous conversations, and so on. The third bar in Figure 9a shows that it is very probable to find more than two concurrently running conversations between the two networks the bar represents. However, this particular bar represents the traffic between UCB and Lawrence Berkeley Laboratory which are located several hundred yards from each other.

Given that we frequently find concurrent conversations between popular network pairs, how often do we find concurrent conversations between host pairs on wide area networks? Figure 9b shows that it is unlikely with the present Internet traffic, but this may change in the future. The second host-pair in Figure 9b frequently exhibits two or three concurrent conversations. This host pair connects an UCB host to an Andrew host at CMU; we suspect that we captured traces of an experiment with the Andrew File System. From this measurement of current traffic, we can say that there are not many concurrent conversations between host pairs over wide-area network.



(a) Network pair



(b) Host pair

Fig. 9: Number of concurrent conversations for the eight most popular network and host pairs coming out of UCB (see section 3.5).

4. An Artificial Workload Model

Our internetwork traffic model has two parts: a stub-network-dependent part and a stub-network-independent part. The first part models conversation arrivals for each type of network application. It determines when a new conversation arrives and what its destination is. We found these characteristics varied widely among stub networks, hence they are stub-dependent. The second part models packet arrivals within a conversation. It determines when a packet arrives and what size it has. We found these characteristics to be constant across all conversations for the same application, hence they are stub-independent.

4.1 Stub-Dependent Models

The stub-network-dependent part of our model consists of a of stochastic process for each application responsible for outgoing traffic from a stub-network. These stochastic processes select the starting time and destination network of new conversations. We call the collection of stochastic processes from all stub networks the internetwork *traffic matrix*. Specifying the traffic matrix is hard because certain applications reference more stub-networks than do others (see Figure 8). For example, we see that half of UCB TELNET conversations are directed to just 11 stub-networks, with the other half referencing over 100 stub-networks. NNTP references just 11 stub-networks for the whole trace. Half of SMTP conversations reference over 50 stub-networks, and the other half reference 300 other stub-networks. Overall, half of UCB's conversations are directed to just 17 stub-networks. The traffic matrix must capture that some stub-networks send more traffic than others, and that some stub-networks use one application more heavily than another.

Analysis of conversation arrival times from all three traces indicates that we can approximate the application arrival processes as time-varying, independent Poisson processes with rate proportional to the stub-network's traffic breakdown (Table 1). We choose the application type of a stub-network's next conversation from the stub-network's traffic breakdown. Independence is only an assumption, because conversations depend on one another. For example, one is more likely to send mail to a stub-network shortly after fingering it than if one had never referenced it before. However this effect is not particularly pronounced in the data. We found that the types of successive conversations are independent, although we did not investigate correlations on the sequence of conversation types between a specific network pair or host pair. Hence, we model arrivals of new conversations as time-varying Poisson processes with stub-network and time-of-day dependent rates. For example, Figure 10 plots measured arrival rates of UCB conversations for several applications. By making the rate depend on time of day, it is possible to model stub-network-specific configurations. For example, at UCB, VMNET runs just four times per day at specified times, while at USC VMNET runs on demand.

A stub's conversation arrival processes must also specify destination stub-networks for each conversation. More study is needed to characterize conversation destination networks However, present technology limits existing network simulators to well less than fifty stub-networks, and typical studies simulate four to sixteen nodes. Larger size of future simulations will make the issue of representative traffic matrix more critical.

4.2 Stub-Independent Models

The stub-network-independent part of our artificial workload model consists of *source* models for an individual conversation of each popular application type. We constructed source models for five of the six applications responsible for more than 96% of wide area network bytes transmitted. The other twenty-nine applications identifiable in the traces account for the remaining 4%. Currently we model FTP, SMTP, NNTP, TELNET, and RLOGIN. The source model is implemented as a library of functions, described in Section 4.3, that can be called from a network simulator.

Modeling a source consists of two steps. First, we select the application-specific characteristics of the new conversation. If the application is bulk we select the amount of data exchanged. If it is interactive, we select the duration of the conversation. Second, we determine the sequence of packets that the conversation will send.

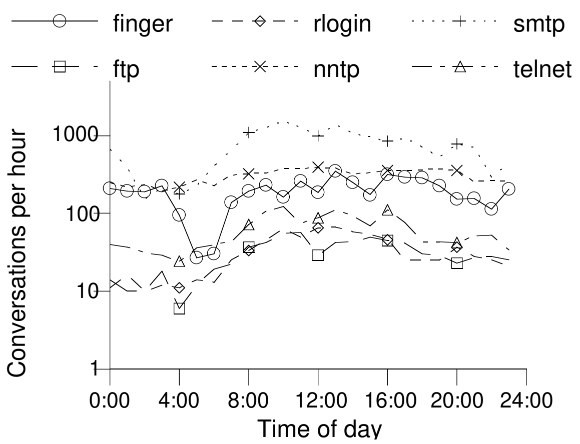


Fig. 10: Conversation arrival rate.

The first step depends on whether the conversation is bulk or interactive. If it is bulk, we choose the number of bytes transmitted in each direction from the joint distribution of bidirectional bytes transmitted. We illustrate one such distribution for SMTP in Figure 5b. This figure plots the larger side of a conversation on the x-axis and the smaller side on the y-axis. Bigger, darker marks indicate higher likelihood. If the conversation is interactive, we choose its duration from the distribution of duration. We illustrate two such distributions for TELNET and RLOGIN in Figure 2a. We show a distribution of duration for bulk protocols in Figure 2b, but do not employ it in the models because the duration of a bulk transfer depends on network bandwidth and flow control, rather than the traffic sources.

For bulk transfer such as FTP and NNTP, we also model the number of items transferred, such as the number of news articles exchanged during an NNTP conversation. Given the distribution of the number of items transferred (see Figure 6) and the distribution of the number of bytes in an item (see Figure 7), we can model the synchronous interactive phase inherent in most bulk applications. During this phase, file names, commands, and article numbers are exchanged. These interactive phases act as synchronization points; at their start, no outstanding packets exist between end points. Hence, there is at least one round trip time between bulk exchanges.

The rule for specifying packet arrival times and sizes depends on the application. For bulk applications, packet sizes and interarrival times depend on physical characteristics of the network, the bidirectional distribution of bytes transferred, and the distribution of items transferred. While their packet interarrival times depend on the network, their packet sizes depend on the application. During bulk transfers, packet sizes are a network MTU followed, if necessary, by a final smaller fragment. During control exchanges, packet sizes are smaller, corresponding to file names and commands; it is necessary to draw their packet sizes from the measured distributions (see Figure 3b).

In contrast to bulk traffic, packet interarrival times of interactive traffic depend on the user. Users' keystrokes generate “byte-sized” packets with a uniform plus exponential interarrival time distribution. The destination process sends a response for every packet that it receives; occasionally it returns a large response (see Figure 3a). A close inspection of the interarrival times of TELNET and RLOGIN packets presented in Figure 4a reveals that 10% of the time, interarrival times are less than 100 milliseconds. These short interarrival times occur for two reasons. First, when the destination sends a response greater than a network MTU, its packets arrive in rapid succession. These back-to-back MTUs account for roughly a quarter of the interarrival times less than 100 milliseconds. Second, network queueing and operating system unresponsiveness can deliver single key strokes to the destination in rapid succession. Back-to-back single data-byte packets constitute roughly three quarters of these short interarrival times.

Application Type	Routine Name
Interactive	float telnet_duration() float telnet_interarrival() int telnet_pktsize()
Bulk Transfer	int ftp_nitems() int nntp_nitems() int ftp_itemsize() int nntp_itemsize() int smtp_itemsize() int ftp_ctlsize()

Table 4: Distribution functions included in the traffic library.

4.3 Generating Conversation Characteristics

The application-specific source models are implemented as a library to be linked with a network simulator. Table 4 presents a summary of our library of routines, *tcplib*[6]. Figures 11 and 12 illustrate how these routines can be used to generate simulated TELNET and FTP traffic. Because curve fitting loses information, and since it makes no difference to the simulator whether there exists an analytical representation of the distributions, *tcplib* generates random numbers by the inverse transform method [31]. Each routine inverts a piecewise linear representation of the measured distribution. Below, we briefly describe the inverse transform method.

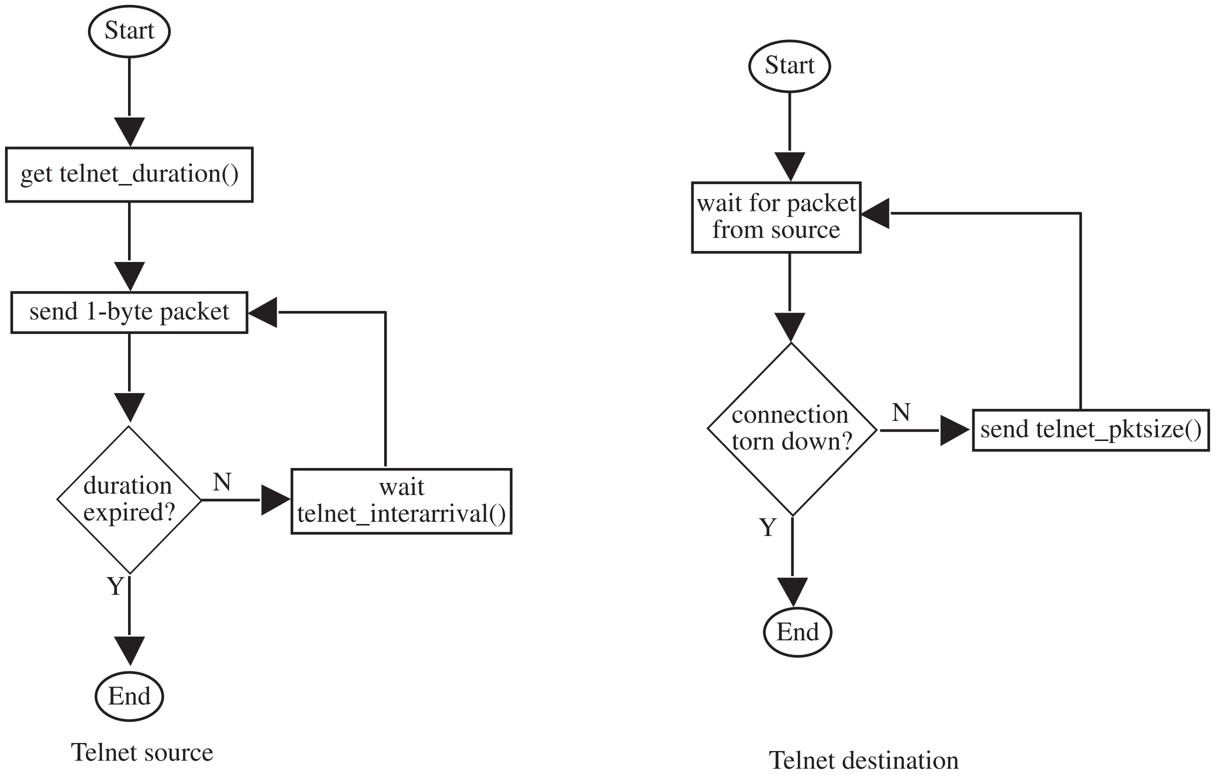


Fig. 11: Flow chart for controlling TELNET and RLOGIN conversations.

[6]tcplib is available from anonymous ftp from jerico.usc.edu in directory pub/jamin/tcplib