to encompass 90 percent of all interarrival gaps. Different researchers have used different MAIGs, ranging from 500 milliseconds to 50 seconds, depending on the network measured.

In contrast, we divided up the traffic into application-level *conversations*. We define a *conversation* to be a stream of packets travelling between the end points of an association, delimited by a twenty-minute silence; an association is in turn defined as a <protocol, source address, source socket, destination address, destination socket> tuple. A twenty-minute silence is longer than FTP's idle connection timeout value of fifteen minutes. Early on we experimented with a five-minute silence rule. The difference in results was minimal. Alternatively, we could have detected the TCP connection establishment handshakes between a source and destination pair and used them to determine the beginning and end of a conversation. This required maintaining a state machine and associated timers for every live connection. Lack of memory space prevented us from doing so.

In the case of FTP, conversations can subsume multiple TCP connections. We clumped several TCP connections into one conversation because each FTP session initiates one FTP-control and zero or more FTP-data connections. We also clumped back-to-back and concurrent FTP sessions between the same source-destination IP-address pair into one conversation. Similarly for NNTP, a new TCP connection is set up for each article sent to the initiator of the conversation—though not vice versa. We clumped back-to-back and concurrent TCP connections used by NNTP.

While FTP uses a control connection to send control packets and a separate data connection for each file transferred, NNTP can use one connection for both handshaking and sending of multiple news articles. Since we wanted the distribution of the number of articles sent per NNTP conversation, and the distribution of article sizes, we needed a scheme for distinguishing individual articles from packets used for handshaking. We had to recognize the first and last packets of an article. From the NNTP documentation and by sampling several articles posted to the Internet, we observed that articles have a minimum size of 250 bytes. Since NNTP tries to send articles using the largest allowable packet size possible, we used the 250-byte minimum size to detect the start of an article transfer. We further noticed that for a given burst of large-sized packets, the last one of the burst usually has its PUSH flag set, while those in the middle of the burst do not. Thus we used the first packet with its PUSH flag set to mark the end of an article transfer. Notice that if the last packet of an article arrived out of sequence an error would be introduced.

Since we want to model the characteristics of transport layer traffic in general, independent of TCP itself, we further decided to drop all TCP-specific traffic. We dropped TCP connection establishment packets and all zero-byte packets, assuming that these were acknowledgement packets. We also filtered out all retransmitted packets.[3] Retransmitted packets were detected by matching their sequence numbers against those of the last 128 packets from the same conversation. Most retransmitted packets match one recently transmitted within the previous 64 packets. The oldest retransmitted packet detected in the analysis of the traces was at position 104 into the buffer. Since we threw away retransmissions, we also threw away most of the keep-alive packets, which share a single sequence number. This also meant that every now and then we saw a lonesome keep-alive as a conversation transferring a single 1-data-byte packet. We filtered out all such false conversations in our analysis. For the Bellcore trace, we further noticed that 50% of all NNTP conversations between Bellcore and Rutgers consisted of a single 6-data-byte packet. After closer examination, we attributed those conversations to an implementation fault at either Bellcore or Rutgers. Our traffic pattern analyzer filtered out all such conversations.

3. Characterization of Application Conversations

We now detail several interesting properties about the applications responsible for most wide area TCP/IP traffic. In Section 4 we describe the application-specific artificial workload models. The observations below are presented under five general categories: traffic breakdown, bulk data

[3]Retransmitted packets accounted for between 0.3% to a little below 3% of all packets belonging to an application.

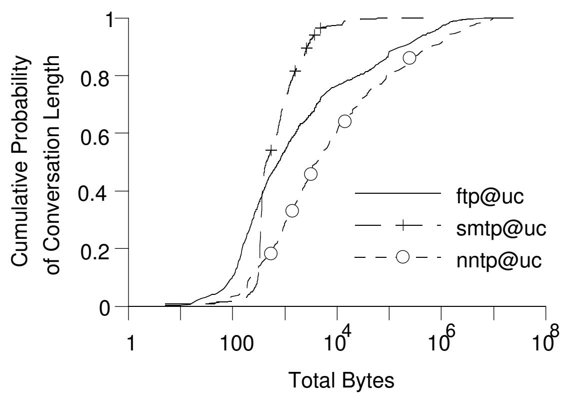
transfer applications, interactive applications, traffic flow, and wide-area network locality. We are interested in such questions as:

- How does TCP traffic break down into interactive and bulk traffic?
- How "bulky" is the data transferred by bulk applications?
- What are the characteristics of interactive applications in terms of bytes transferred, burstiness, duration, and interarrival time?
- Is traffic flow unidirectional or bidirectional?
- Is there network-pair locality on wide-area networks and how many concurrent conversations are there between such network pairs?

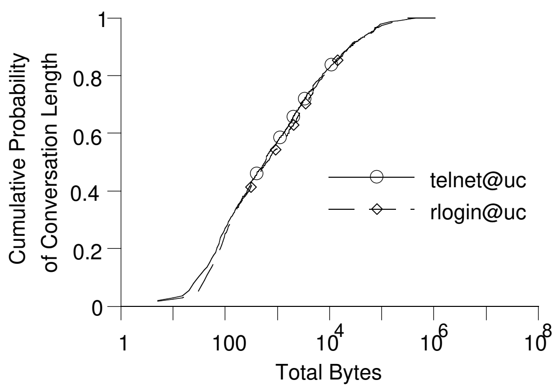
Table 3 summarizes our most important observations about TCP traffic.

Depending on the application, 60-90% of the conversations categorized as bulk transfer send less than 10 kilobytes of data. Bulk transfer is request-response in nature, with responses larger than 1 megabyte responsible for only 15-20% of all bytes transferred.
A large portion of bulk transfer applications, which are responsible for more than 50% of observed network traffic, show bidirectional traffic flow.
Over 90% of interactive conversations send fewer than 1,000 packets and 50% of interactive conversations last less than a minute and a half. Packets belonging to interactive applications are mostly smaller than 512 bytes.
Interactive applications can generate 10 times more data in one direction than the other, using packet sizes ranging from 1 byte to 512 bytes.
A uniform plus exponential distribution best models interarrival times of packets belonging to interactive applications.

Table 3: Selected Observations.



(a) Bulk transfer



(b) Interactive applications

Fig. 1: Total bytes transferred per unidirectional conversation.

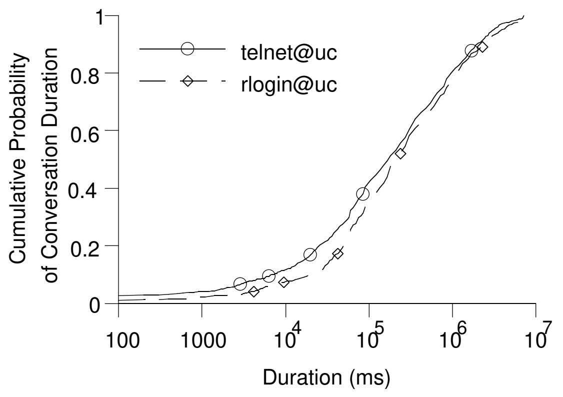
3.1. Traffic Breakdown

For lack of a more accurate workload model, previous studies that simulate flow control, congestion control, multiple access protocols, and traffic dynamics in general have been forced to assume a rather simple traffic model [7, 23, 22, 25, 26, 27]. These studies either used a continuous bulk transfer or an arbitrary mix of bulk and interactive traffic.

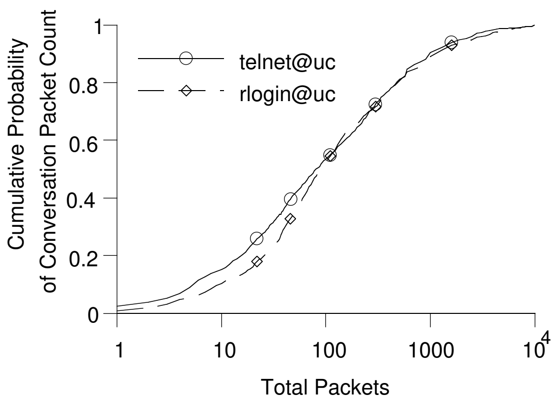
TCP traffic consists of bulk and interactive traffic as commonly assumed. Table 1[4] shows the distribution of number of bytes, packets, and conversations attributed to each application. Even though bulk applications send more data than interactive ones, interactive conversations still send 5-10% of network bytes and 25-45% of network packets.

We think it important to realize that interactive applications are responsible for 25-45% of all Internet packets. Simulations that model internetwork traffic as mostly large bulk transfers may overestimate the benefit of mechanisms proposed to improve bulk transfer performance. Most existing studies evaluate the robustness of designs and algorithms under worst case loads, but fail to contrast their performance to that of equally robust designs or algorithms when running under average loads.

[4]The applications which appear in boldface are the ones we concentrate our study on.



(a) Conversation duration



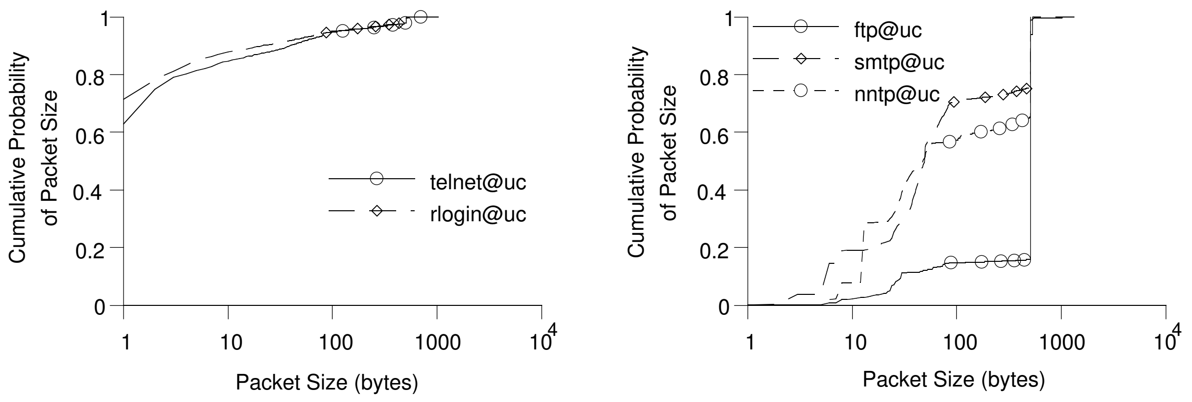
(b) Packets transferred

Fig. 2: Duration and packets transferred per conversation for interactive applications.

3.2. Bulk Data Transfer

Many simulation studies commonly overestimate the amount of data sent by bulk data transfer applications such as FTP. Transfer sizes usually range from 80K to 2M bytes, or simply continue to the end of the simulation run [7, 23, 24, 26, 27]. Figure 1a shows that about 60-90% of bulk transfer conversations transfer less than 10K bytes. Incidentally, our data shows that conversations that transfer more than one megabyte of data are responsible for 40-50% of all bytes transferred. However, as explained in section 3.4, bulk transfer conversations usually transfer more than one item per conversation—an *item* is a news article for NNTP or a file for FTP, with items larger than one megabyte making up only 15-20% of all bytes transferred. This observation correlates with the observation made in Reference 29 that most files are small.

If this is true of Internet source traffic in general, then it should be taken into account in future internetwork simulations. To the extent that simulated algorithms employ feedback mechanisms (such as congestion or flow control) [8], it is important to know that in most sessions data transfer will complete before any such feedback is received. Also, bulk transfer applications typically wait at least one network round-trip time between exchanging items. Therefore, as currently implemented, bulk applications, like NNTP, that appear to make exchanges of large amount of data actually make hundreds of exchanges of small data pieces separated by at least one network round trip time. We believe these observations are important and long lasting because.for delay insensitive applications such as NNTP, this application design may be desirable in that it regulates the rate at which the applications send data. The emergence of voluminous real-time traffic will not make existing delay-insensitive traffic disappear.



(a) Interactive applications (b) Bulk transfer

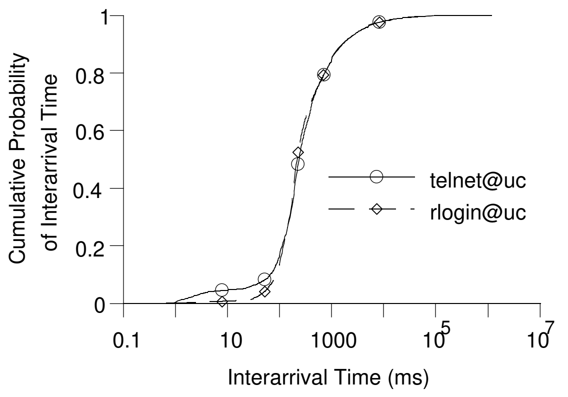
Fig. 3: Distribution of packet size by application. Packet sizes reflect only user data without protocol headers.

3.3. Interactive Applications

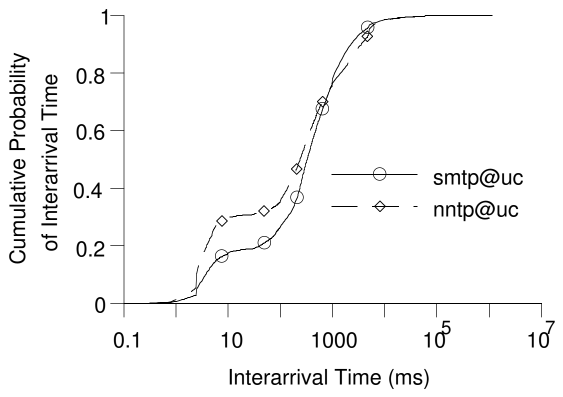
Network flow control and the Maximum Transferrable Unit (MTU)[5] determine, to a great extent, the measured statistics of bulk internetwork traffic. In contrast, Figure 1b and 2a show that about 90% of TELNET and RLOGIN conversations send less than 10K bytes, over a duration of 1.5 to 50 minutes. Figure 3a shows that about 90% of TELNET and RLOGIN packets carry less than 10 bytes of user data, which is much smaller than the MTU. Thus interactive applications are more or less unaffected by flow control and MTU size.

If interactive applications are not affected by network flow control and MTU, then the observed characteristics reflect the true nature of such applications. However, we should not assume that interactive traffic carries less data—Figure 1b shows that 80% of all interactive conversations send as much data as the average bulk transfer conversation—rather, it means that bulk transfer applications send a smaller amount of data than is often assumed.

[5]For historical reasons, wide-area TCP connections still use an MTU of 512 data bytes despite the fact that the NSFNET backbone supports 1500-byte packet.



(a) Interactive applications



(b) Bulk transfer

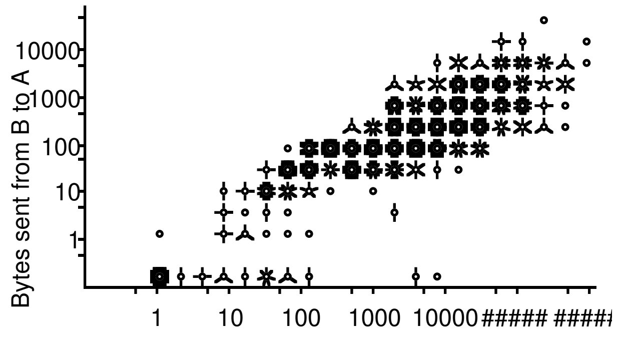
Fig. 4: Distribution of packet interarrival by application.

In most traffic models used in existing simulations or testbed studies, conversations are assumed to last anywhere from 500 seconds, 600 seconds, to “keep on forever” [7, 23, 24, 27, 29]. Figure 2a shows that the duration of interactive conversations is highly variable. This fact, along with the small number of packets per conversation (see Figure 2b), might influence steady-state feedback assumptions, as well as per-packet processing time with respect to gateway algorithms.

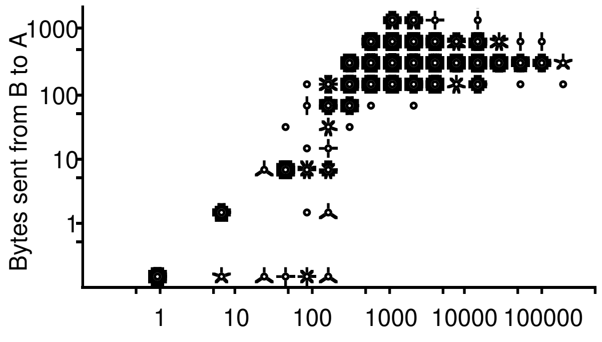
Finally, our data shows that while interarrival times for bulk data transfers exhibit the packet-train phenomenon, interarrival times for interactive applications should be modeled by a uniform plus exponential random time (see Figure 4a). Section 4 describes this phenomenon in more detail.

3.4. Directionality of Traffic Flow

Most simulations of gateway queueing [7, 23, 26] have assumed unidirectional data flow. Figure 5 shows that a large percentage of traffic, both interactive and bulk, is bidirectional. In other words, simulations should generate traffic in both directions.



(a) Telnet



(b) SMTP

Fig. 5: Bidirectionality of traffic flow.

Figure 3b and 4b together affirm that many bulk transfer applications contain a request-response phase, which causes a synchronization point where no data is flowing in either direction. In turn, this synchronization point causes classic packet train behavior: a handshake followed by a big burst. For example, NNTP sends a query, waits for a response, and then does a bulk transfer, while FTP sets up a separate TCP connection for each item transferred. Since we tracked conversations in a unidirectional fashion, we found that 60% of all FTP and NNTP

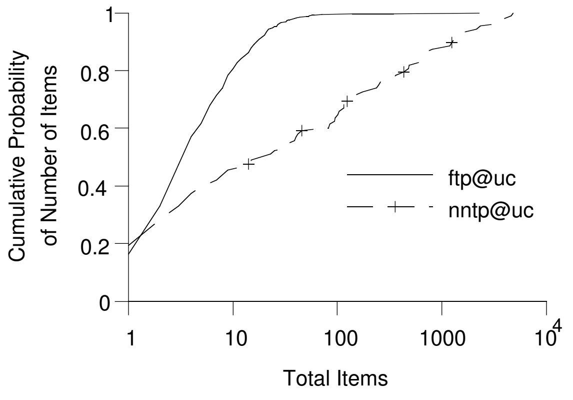


Fig. 6: Number of items per conversation.

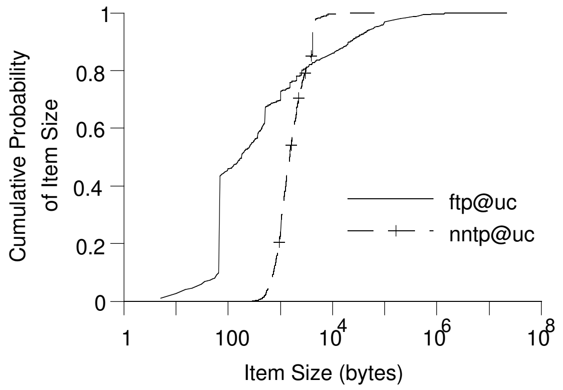


Fig. 7: Distribution of item sizes.

conversations do not send any items. However, when paired with their corresponding conversation going in the other direction, we found that bytes flow bidirectionally due to protocol handshaking—in the case of NNTP—or user control—in the case of FTP. Ignoring conversations that send zero items, Figure 6 shows distribution of the number of items sent per conversation for FTP and NNTP, and Figure 7 shows the distribution of the item sizes. This