1995

Experimental Study of TCP and UDP Protocols for Future Distributed Databases

Xiangning Liu

Lebin Cheng

Bharat Bhargava
Purdue University, bb@cs.purdue.edu

Zhiyuan Zhao

Report Number:
95-046

EXPERIMENTAL STUDY OF TCP AND UDP PROTOCOLS FOR FUTURE DISTRIBUTED DATABASES

Xiangning Liu
Lebin Cheng
Bharat Bhargava
Zhiyuan Zhao

Department of Computer Sciences
Purdue University
West Lafayette, IN 47907

CSD-TR-95-046
July 1995

Experimental Study of TCP and UDP Protocols for Future Distributed Databases

Xiangning Liu, Lebin Cheng, Bharat Bhargava, and *Zhiyuan Zhao*

Department of Computer Sciences
Purdue University
West Lafayette, IN 47907
E-mail: {xl,lcheng,bb,zhao}@cs.purdue.edu

Contents

Abstract

Experiments were conducted to measure the data communication performance of the TCP and UDP/IP protocols on both currently-existing computer networks and the ATM (Asynchronous Transfer Mode) network of the future. Such directly observed information regarding performance and availability will be useful to developers of both

applications and network systems. Of particular relevance to database applications is our discussion of the impact of network behavior on the performance and availability of distributed database systems, especially on databases that are scaled up in multiple dimensions. We suggest that database systems should adapt to different environments to enhance efficiency and reliability. We weigh the merit of TCP and UDP for use with different applications on currently-existing networks and the future ATM network. Abnormal network behavior observed in connection with the transmission of messages of particular sizes via TCP over an ATM network is discussed and solutions are proposed. Finally, we attempt a description of the ATM-based database systems in the future.

Keywords: Adaptability, availability, performance, scale up, TCP, UDP/IP, Internet, LAN (Local Area Network), WAN (Wide Area Network), ATM.

1 Introduction

Future database applications, such as multimedia applications and digital libraries, will involve large volumes of data, long distances between distributed sites, large numbers of sites, and large data units with complex structures [SSU91]. An increase in the number of data items in a database is termed a *volume scale-up*, while an increase in the number of sites is a *topological scale-up*. An increase in intersite distance is a *metric scale-up*, and a *structural scale-up* occurs when data items become complex objects. When databases scale up in all these dimensions, communication of data items assumes a highly significant role in system performance and availability.

At present, TCP and UDP are the network protocols most commonly used for data communication. Most distributed database systems are implemented with TCP or UDP, and this is likely to remain the case into the foreseeable future. The aspects of network performance and behavior to be discussed in this paper can therefore form the basis of further database performance analysis and simulation. Database designers can use these results to choose a suitable network mechanism.

It has frequently been suggested that the computer networks of the future will be based on the fast ATM (Asynchronous Transfer Mode) network [Vet95]. The current network structure, with its slow speed and unreliability, will not be adequate to the needs of the future applications, which will be scaled up in all dimensions. The ATM Forum has been formed by a large group of companies which have made a commitment to the support of ATM development. For these reasons, databases in the future are likely to incorporate the ATM technology. The behavior and construction of such distributed database systems are discussed in this paper, with examples from local experiments cited.

1.1 Research Overview

Experiments were conducted using TCP (Transmission Control Protocol) and UDP (User Datagram Protocol) [PSC81, Com91], with objects of various sizes and with several transmission options, on currently-existing LANs and WANs. Communication delay and failure rates were measured. Experiments were also conducted using 19 NASA image files. On the basis of these results, we analyzed the respective merits of TCP and UDP and the various transmission options.

Local experiments using the ATM indicated that the desired performance level can be achieved for the transmission of messages smaller than 4 Kbytes. With messages of certain sizes, however, TCP is not fast enough to utilize the bandwidth provided by the ATM network at the physical layer. The implementation of the TCP protocol becomes in itself a performance barrier. The potential advantages of using the ATM to efficiently transmit large messages are thus unacceptably negated.

In one experiment, we removed the ATM switch, directly connected the two end-point hosts with fiber optic cable, and compared these results to those of tests in which an ATM switch was used to connect the end points. Little difference was observable between the results of the two tests. This experiment shows that the ATM switch itself is not responsible for the performance downgrade. The problem was caused by the mismatch of the computer

system, especially the operating system network software, with the ATM physical layer.

1.2 Related Research

Many tools and methods have been developed to measure various aspects of computer network performance and availability. Layered refinement [PKL91] permits the effective measurement of the performance of different layers of TCP/IP by filtering noise. [Gol92] introduced network measurement of end-to-end performance to predict Internet behavior. [CJRS89] described the overhead entailed in running TCP and IP by identifying the normal paths through the compiled code of TCP implementation. Many efficient communication mechanisms are provided by experimental operating systems. Examples include V [Che88], Mach [Ras86], Amoeba [TRvS+90], Sprite [OCD+88], and x-kernel [PHOR90]. The development of distributed databases has greatly benefited from the results of these research efforts.

Intense research activity directed toward the development of the computer networks of the future is taking place in both industrial and academic settings. [Kun92] discussed the high-speed local area network in which gigabit bandwidths will change dramatically the domain of future computer applications. [Vet95] surveyed the basic concepts and technology of the ATM. [PR95] presented a simulation study for the TCP protocol of the problem of data flooding on ATM switches. Solutions such as increasing the ATM switch buffer and additional congestion control methods were proposed to address the mismatch between ATM technology and the existing TCP implementation. The poor performance of TCP has also been reported in a number of studies [Rom93, AM94]. [CL94] discussed the TCP buffer problem of UNIX 4.3 BSD-Tahoe implementation for ATM communication. [CS92] discussed some internetworking strategies and issues that arise when ATM networks are used to carry the TCP/IP protocol suite on WANs.

At Purdue University, an experimental RAID (Robust Adaptable and Interoperable Distributed) database system [BR89b, BR89a] was developed. With RAID, the adaptability

of the communication subsystem is addressed through the UDP protocol. Several major improvements have since been implemented to achieve high performance [BZM91, MB91]. These enhancements, which result in an approximately 70% improvement in response time (e.g., a decrease from 40 to 22 milliseconds), include:

- Using the efficient UDP protocol, rather than TCP, to provide services of simple naming, communication surveillance, and long datagram mechanisms for large data items;
- Supporting lightweight communication, reduced kernel interaction, minimal message copying, and physical multicasting; and
- Enforcing a policy of immediate rescheduling to explicitly pass control over interprocessing communication on one machine.

The WANCE tools developed as part of RAID support emulation of distributed transaction systems over a WAN [ZB93] by taking advantage of the echo facility on each Internet host. The RAID project has continued to investigate efficient communication mechanisms which incorporate both the TCP protocol and the ATM network. These investigations, which are directed toward the anticipated topologic, metric and volume scale-ups of future database applications, are described in this paper.

1.3 Structure of the Paper

The remainder of this paper is organized as follows. In Section 2, we present results of experiments conducted on existing networks. Two sets of experiments on a LAN and a WAN are discussed. In Section 3, we discuss and analyze experiments conducted on an ATM network. The problem of TCP implementation on the ATM is described and solutions are proposed. In Section 4, we draw conclusions and provide directions for future work.

2 Data Communication on Existing Networks

When database systems scale up topologically, metrically, or in data size, the impact on the widely-used Internet with its TCP and UDP/IP protocols must be considered. Our experiments use the Ethernet as the LAN (local area network) physical data transfer medium. For WAN (wide area data) communication, the Internet is the testbed which is an integration of many networks.

TCP is a connection-oriented, flow-controlled, end-to-end transport protocol that provides reliable and ordered stream delivery of data [Pos81]. UDP is a simple connectionless datagram transport protocol that provides peer-to-peer addressing and fast but unreliable and unordered delivery of data [Pos80]. The two protocols are thus best suited to different purposes and applications. Based on experimental performance measurements, we provide guidelines for selection between these two communication protocols.

2.1 Local Area Network (LAN) Experiments

Problem Statement

The purpose of these experiments was to measure and compare data communication performance of TCP and UDP on a LAN for 1) short messages in traditional databases and 2) large messages in future applications such as digital libraries. UDP is faster than TCP but is less reliable. Measurements of message round-trip time and loss rate made in the course of these investigations will permit an objective assessment of the tradeoff between speed and reliability. The result of this experiment will be the basis for the discussion of database metric scale-up.

The development of new applications has brought with it increasing demands for shorter delays, larger bandwidths, lower packet loss rates, and higher throughput. The capacities of the physical layer may be incompletely exploited by inefficient high-level software. Applications which do not choose appropriate and properly-configured protocols will not make

optimal use of network capabilities.

Procedure

All experiments on the local area network were conducted between two Sun Sparc workstations `raid9.cs.purdue.edu` and `raid11.cs.purdue.edu` connected with 10 Mbps Ethernet. An extension of the `ping` programs originally authored by M. Muuss [Ste90] was used as the experimental vehicle. In this set-up, a sender process running on one machine sends messages to the echo receiver on another machine. The receiver will not echo the message back to the sender until it receives the entire message. The round-trip time is then calculated by the sender using the time interval between the delivery of a message and the arrival of the echo message. Each experimental trial consisted of 50 message echoes, and the experiment was repeated twenty times. The experiments were conducted using three TCP modes and two UDP modes. These modes were as follows:

- **Individual-connection TCP:** A connection is established and closed for each message sent. Some applications can send several messages through one TCP connection, while in other instances the number of messages passed through a given connection is unpredictable. A system can maintain only a limited number of simultaneous connections. The closing of a connection may increase the overhead for later messages but will allow other applications to use these limited connection resources.

- **No-delay TCP**: Small packets are sent immediately by setting the option TCP_NODELAY. By default and following the small message avoidance protocol, the TCP software saves small packets in buffers instead of sending them immediately [Nag84]. The TCP_NODELAY option allows users to force a packet to be sent immediately after it is created.

- **TCP** or single-connection TCP: Regular TCP messages, by default, are sent within a single connection without invoking the TCP_NODELAY option.

- **Non-aggressive UDP**: Large messages are divided into 8 Kbyte packets which are sent at short intervals by UDP. A `for` loop was used to ratchet up an integer 2000

times, thus generating the interpacket interval used in our LAN experiments. The length of this interval is system-dependent.

- **UDP** or aggressive UDP: For large messages, each 8 Kbyte packet is sent immediately following the previous packet.

The following extensions were made to the ping programs:

- Provision of options to support separate connection experiments (for Individual-connection TCP) and no-delay transmission of small packets (for No-delay TCP)).
- Support for large messages of up to 4 megabytes, an extension from the 8 Kbyte maximum of the original programs.
- Measurement of the precise round-trip time. The echo receiver awaits arrival of the entire message before returning it to the sender. The original ping program starts to echo back when part of the message has arrived.
- Automatic adjustment of the time interval between the individual messages sent out by the sender. If the interval (1 second by default) is too short and the messages are large, the system may break down if a new message is sent before the previous messages return.
- Avoiding intermediate output. Under UDP, large messages are divided into small packets. If the sender must output the round-trip time after the arrival of each packet, the accumulated round-trip time measured for a large message will be artificially increased by the lag of the print statements which involves slow device I/O. For example, as shown in Figure 1, test results with the original program seem to indicate that UDP was sometimes slower than TCP. When the extra print statements were removed, this abnormal behavior disappeared.
- Adding an option to adjust the interpacket interval for non-aggressive UDP data transmission.

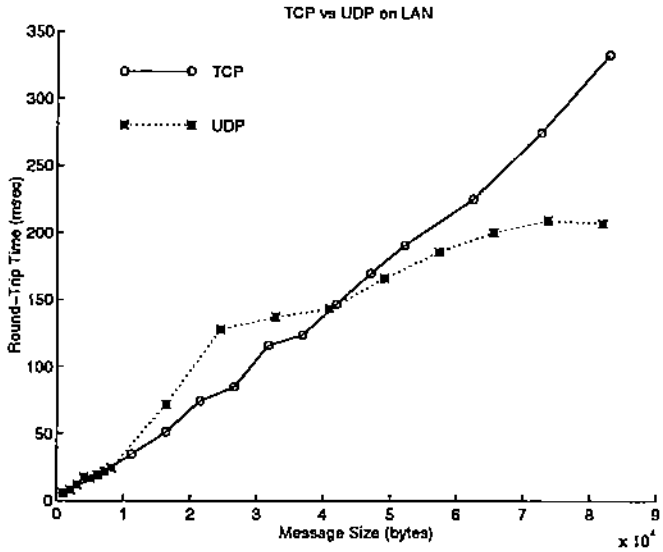

Figure 1: Experimental results with the original ping program

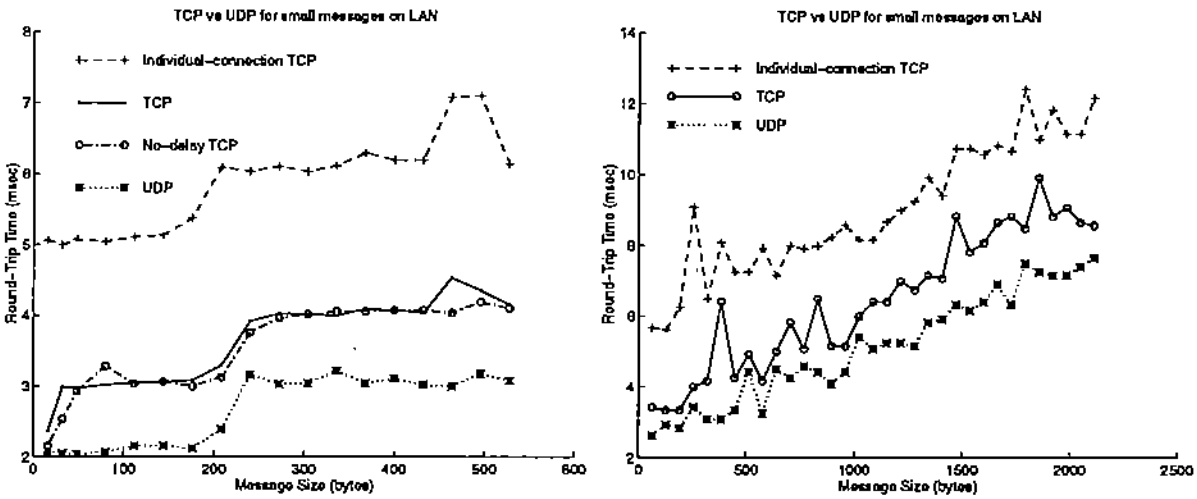

Figure 2: Round-trip time for small messages in a LAN a) shorter than 512 bytes; b) shorter than 2 Kbytes.

Table 1: Large NASA image files

SIZE	FILE CONTENT
6988	**earth-round.gif:** Sharp contours, green on blue globe. Res 187x158
7708	**earth1.gif:** Very sharp contours, green on blue globe. Res: 160x160
17027	**gal_line.gif:** Red on black, a whole line of only dots. Res: 450x450
29668	**gal_green.gif:** Green on green, lots of dots, striations of colors. Res: 384x330
35543	**comet.gif:** White eye, blue tail, tail fades into background. Res: 512x480
60379	**mars.gif:** Huge circle of light brown shades. Res: 340x340
74058	**surface.gif:** Sloping surface of white and blue, sharp contours, shades. Res: 550x450
80385	**jupiter.gif:** Huge circle of red and yellow shades, yellow text on black. Res: 710x765
97835	**gal_blue.gif:** Blue on blue, some dots. Res: 607x373
104365	**hubble.costar.gif:** Shades of concentric red, orange, yellow colors; shading, text. Res: 566x384
114323	**earth_detail.gif:** Blurred contours; text; pink color; black bg.. Res: 1152x864
135701	**eclipse2.gif:** A huge number of red shades. Res: 784x630
153634	**4gal_red.gif:** Bright red, orange; black and white dots; shading. Res: 441x400
175405	**sf.gif:** Sharp boundary contours, blue, white and red colors. Res: 500x500
205747	**ast_spray.gif:** Black background, lots of small particles. Res: 701x659
236199	**mitwave1.gif:** Orange and white shades, delicate, multicolored ridges. Res: 1024x1024
279786	**earth_highres.gif:** Blurred contours; text; blue color; black bg.; Res: 1152x864
406851	**text+image.gif:** Text, many dots, subtle shading. Res: 936x867
486430	**eclipse1.gif:** A huge number of orange and yellow shades. Res: 1280x1024

Results

We conducted experiments using messages of various sizes:

1. Figures 2 a) and b) show the round-trip times for messages under 512 bytes and 2 Kbytes, respectively. No UDP packet loss was detected in this case.

2. Figure 3 provides results for larger messages of between 1 and 80 Kbytes in size. Since the TCP_NODELAY option would have no impact on these results, we did not repeat the test with this option involved.

3. Figure 4 shows the results obtained from sending very large messages of between 50 and 1000 Kbytes.

4. Figure 5 provides the results of sending large multimedia NASA files. The sizes and descriptions of these files are listed in Table 1.

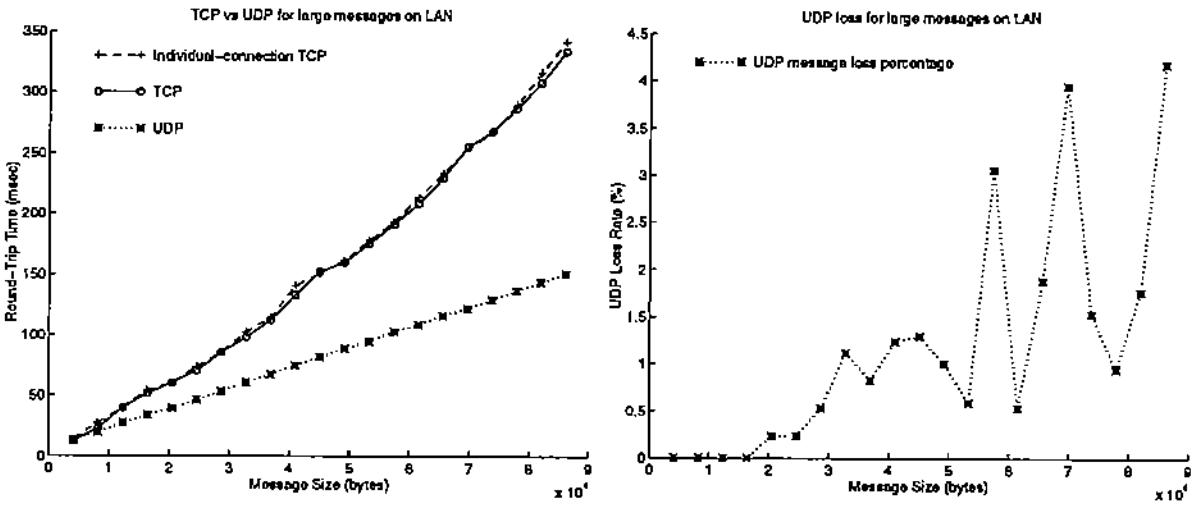

Figure 3: a) Round-trip time for large messages; b) UDP packet loss rate for large messages

Discussion

For small messages, the experiments show that UDP is the most efficient mechanism, with almost no message loss noted. In view of the absence of message loss, TCP need undertake

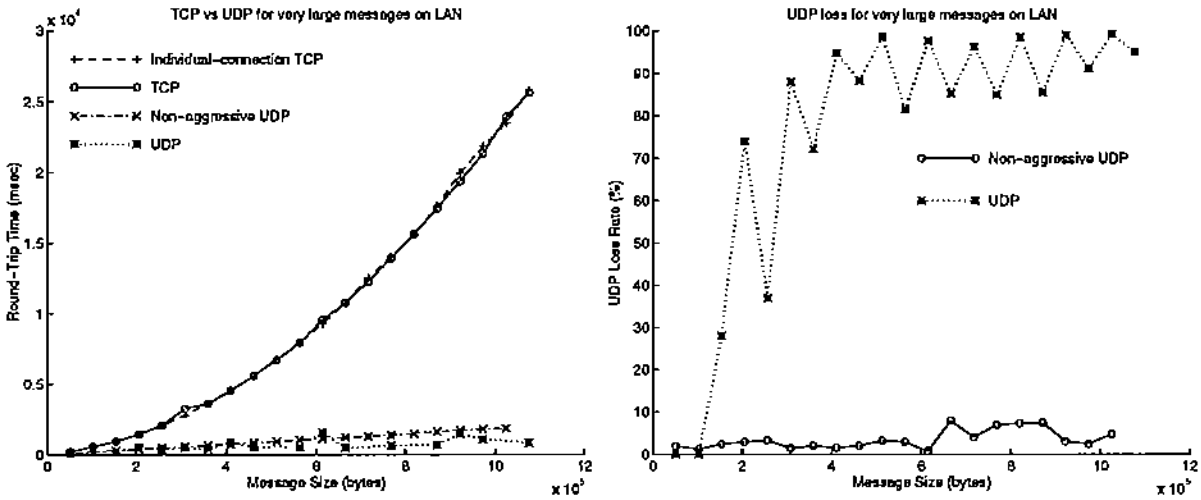

Figure 4: a) Round-trip time for very large messages; b) UDP packet loss rate for very large messages

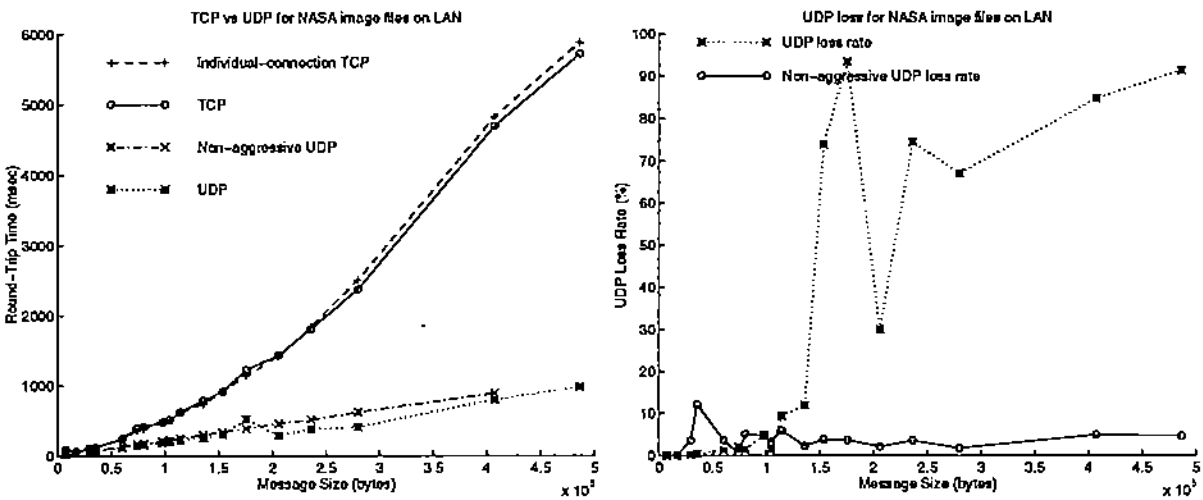

Figure 5: a) Round-trip time for large files; b) UDP packet loss rate for large files

no retransmission; its performance here is therefore comparable to that of UDP. Furthermore, TCP_NODELAY has no effect when only one small message is sent. The effects of TCP_NODELAY will be discussed further in Section 3.2. Connection establishment has been noted to add significant overhead for the sending of small messages. Therefore, the reestablishment of connections should be avoided for such messages.

It is clear that, while UDP is much faster than other mechanisms for messages of larger size, its reliability gradually declines under these circumstances. The Ethernet employs the CSMA/CD (Carrier Sense Multiple Access/Collision Detection) mechanism to avoid collisions of transmitted signals. When messages become larger and more IP packets must thus be generated for each message, the chance of collisions and out-of-order delivery becomes significantly higher. For this reason, TCP must retransmit large amounts of data when sending large messages, thus degrading its performance.

The extent of misordered data and data loss increases significantly when the message size exceeds 200 Kbytes. The overhead caused by connection re-establishment, in this case, is negligible.

For very large messages, non-aggressive UDP transmission can reduce packet loss significantly while causing little increase in speed-related overhead. This situation is depicted in Figure 4. In these experiments, the loop

```
for(i=0;i<2000;i++)
    ;
```

were used to generate the interval between each UDP packet. Experiments show that intervals longer than these may increase transmission delays with little further improvement in reliability. Shorter intervals, on the other hand, may not be enough to bring significant improvement. The number of iterations is system-dependent.

2.2 Wide Area Network (WAN) Experiments

Problem Statement

These experiments were intended to investigate the questions discussed above within the context of a wide area network (WAN). Based on the result, database topological and metric scale-up is discussed with respect to the data communication.

Procedure

The ping program as described in Section 2.1 was again employed for these experiments. An echo server was installed on a Sun Sparc workstation at Stanford University; unlike the common echo server on port 7, this server has the capacity to handle the large messages used for these experiments. The input data from the LAN experiments was used here again.

Results

The experimental results for messages of different sizes were as follows:

1. Figure 6 shows round-trip time for messages under 1 Kbytes. The UDP packet loss was small (less than 7%), and the TCP_NODELAY option had no impact in this case.

2. Figure 7 provides results obtained from sending large messages of between 1 and 80 Kbytes.

3. Figure 8 shows the results of sending very large messages of between 50 and 1000 Kbytes. The overhead arising from connection time is negligible in this case, but there is an unacceptably high UDP loss rate. It therefore proved impossible to gather a sufficient sample of round-trip times.

4. Figure 9 provides results obtained from sending the same large NASA multimedia data files discussed in Section 2.1 and detailed in Table 1.

Discussion

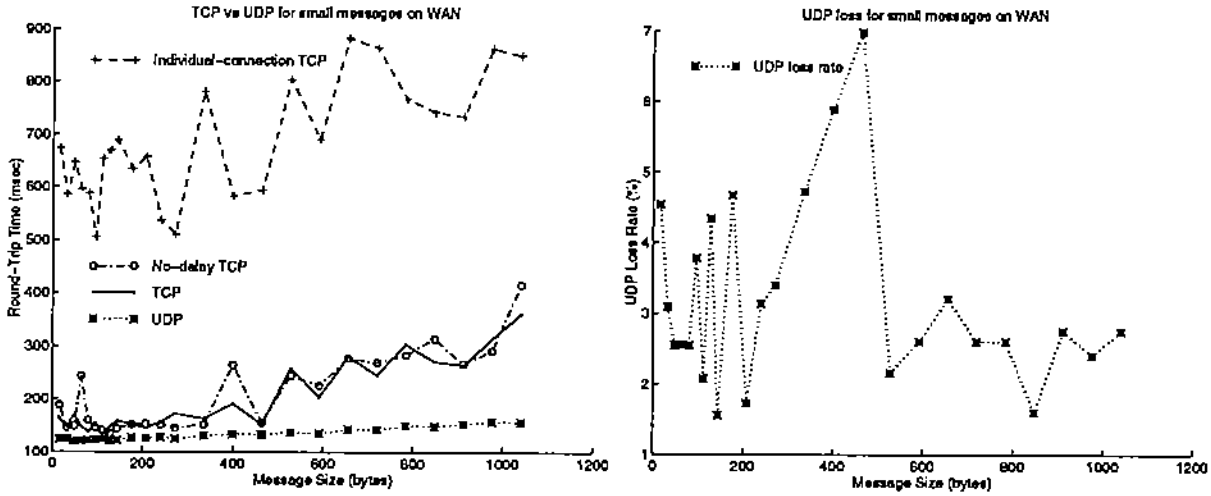

Figure 6: a) Round-trip time, b) UDP packet loss rate for small messages over WAN

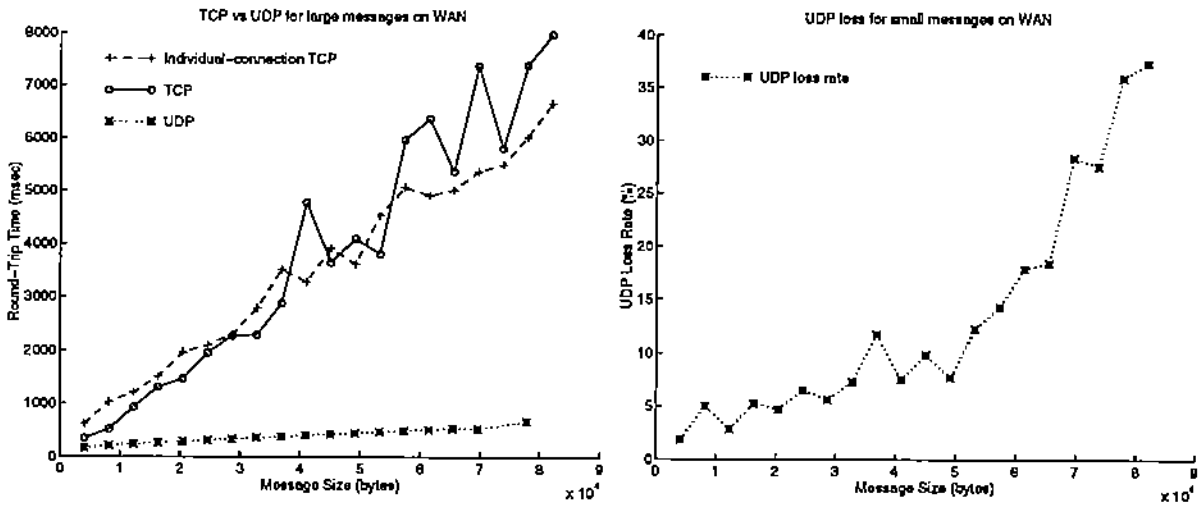

Figure 7: a) Round-trip time, b) UDP packet loss rate for large messages over WAN

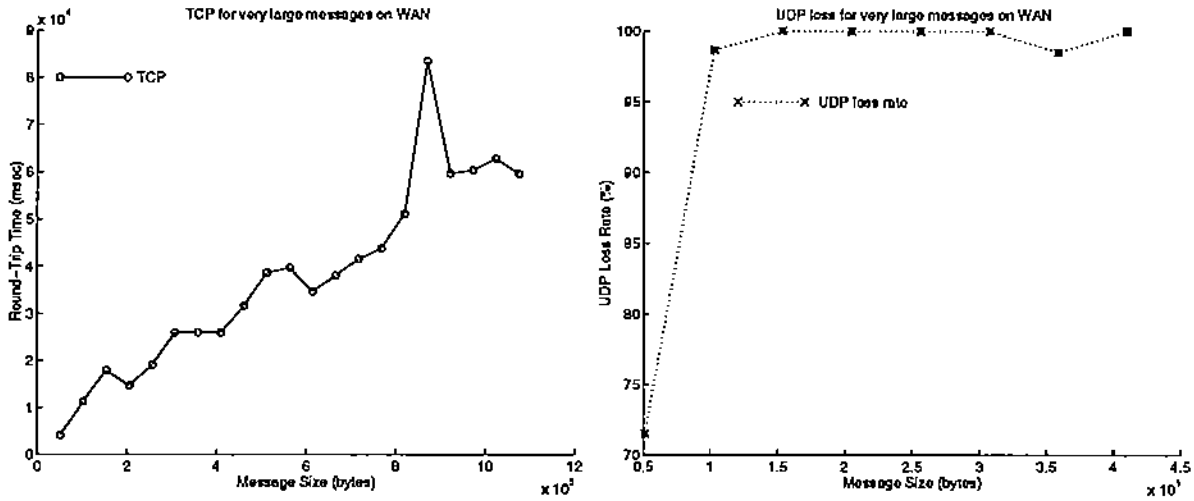

Figure 8: a) Round-trip time for very large messages; b) UDP message lost rate for very large messages

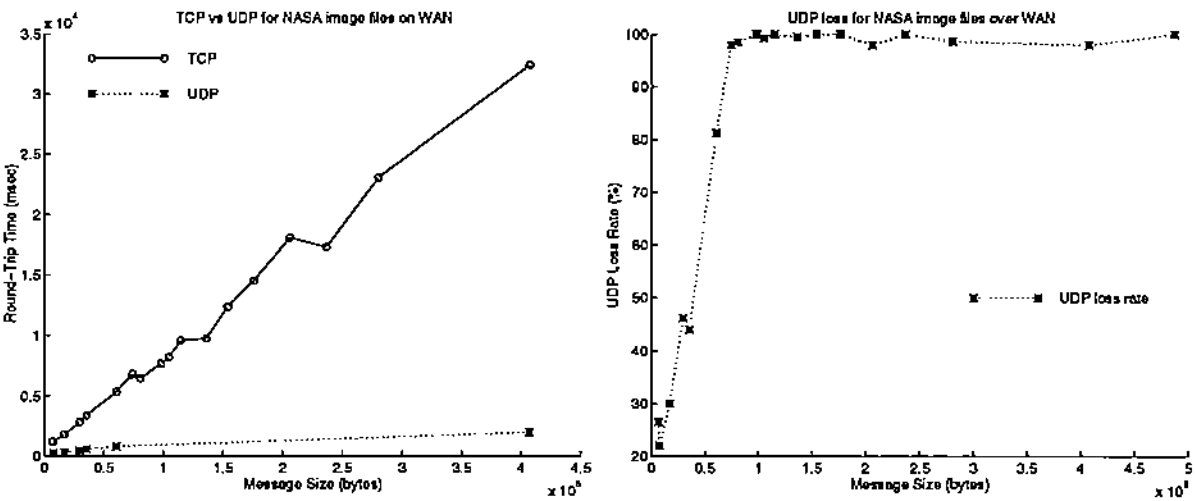

Figure 9: a) Round-trip time for NASA image files on WAN; b) UDP packet loss rate for NASA image files

These results from the LAN and WAN experiments lead to the following conclusions:

- In general, TCP is recommended for data communication in database systems, especially when databases scale up topologically and metrically. Despite its speed advantages, the reliability of UDP is inadequate for remote communication and large data transmission. Data loss and out-of-order messages may cause transaction failure from which the database may have difficulty recoverying. When databases scale up on topological and metric dimension, the problem becomes more severe. A reliable data communication service is therefore essential, and TCP is preferable in this regard. On a Local Area Network (LAN), the efficiency of TCP is usually adequate for the transmission of small messages and is only slightly less than that of UDP.

- UDP should be used only for time-critical applications, such as video services and digital libraries, that may tolerate a certain amount of data loss. Since the aggressive sending of continuous UDP packets will result in heavy data losses, transmissions should instead be spaced at short intervals. Experimental results using this method on a LAN show a significant reduction in losses, and, although the speed is somewhat decreased, it is still much faster than TCP. Unfortunately, this method does little to improve UDP reliability in a WAN context.

- Applications which act immediately upon partially-received data can also employ UDP. For example, an image can be partially or roughly displayed to a user on the basis of partial data received from a remote site, with the image completed as the remaining data arrives. Lost packets can be retransmitted for later arrival. The flexibility of UDP is well suited to this approach.

3 Data Communication on ATM

3.1 ATM Technology

Database scale-up on topological, metric, and volume dimensions requires fast and reliable data communication. ATM (Asynchronous Transfer Mode) is a family of protocols

supporting both circuit and packet-switching services. ATM cells, or fixed-length packets, form the basic data units. An ATM cell, as defined by ITU (formerly CCITT) recommendation I.361, contains 48 octets of data and 5 octets of control information. This fixed-length cell and the early binding of routing information during the connection setup make the ATM suitable for high-speed data communication. Its bandwidth reservation and graceful multiplexing also render it suitable for multi-media traffic. Four major benefits of the ATM discussed in [KW95] are scalability, statistical multiplexing, traffic integration, and network simplicity. These characteristics are the very things required by database scale-up.

Many applications will benefit from the flexibility in switching and high speed provided by the ATM. For example, consider the case of an oil company with several computing centers generating large numerical simulations of drill sites. Engineers would find it useful to display these results on their graphical workstations, an application requiring a high-throughput network. As another example, consider a digital medical application transmitting dozens of x-ray images involving several gigabits of information. In many cases, real-time access is required for collaborative diagnosis between physicians in different locations. This large quantity of data can only be delivered in real time by networks running at multimegabit speeds. Ideally, the collaborating physicians may wish to set up video conference connections for their discussion. The ATM provides flexible multiplexing technology to meet these needs.

The ATM was originally designed for the B-ISDN (Broadband Integrated Services Digital Network), a network intended to integrate time critical and data-heavy graphic, imaging, video, and audio services. As database systems scale up in multiple dimensions, they must also involve applications of this sort, for which the ATM will be the most suitable platform.

3.2 Experiments on ATM with TCP and UDP

In order to take full advantage of an ATM network, adaptation must be made to match the ATM with the popular TCP/IP and UDP protocols upon which most existing applications are built. We conducted experiments to investigate the data communication characteristics

of TCP/IP and UDP when applying to an ATM network. The experiment was designed to observe the impact of this change in the lower layer on the upper-level database systems. We expected that the ATM would provide fast and reliable data transmission for the applications. Due to the resource limitation, we can only conduct experiments on an ATM LAN. The scale-up experiments are only conducted on the data volume dimension. But we still can use the results for the discussion of scale-up on topological and metric dimension.

3.2.1 Experimental environments

Our experiments involved two Sun Microsystems SPARCstation IPCs, called `percival` and `fibonacci`, running SunOS 4.1.3 with the 4.3 BSD-Tahoe TCP implementation. As shown in Figure 10, each host connects to a Fore System ASX-100 ATM switch via 100 Mbp/s multi-mode fiber cables, which supplement the conventional 10 Mbp/s Ethernet. Both hosts use Fore System SBA-200 ATM adapter cards, the driver of which supports ATM Adaptation Layer 5 (ALL5). The Fore ATM switch comes with a dedicated processor and special-purpose software for user configurations.

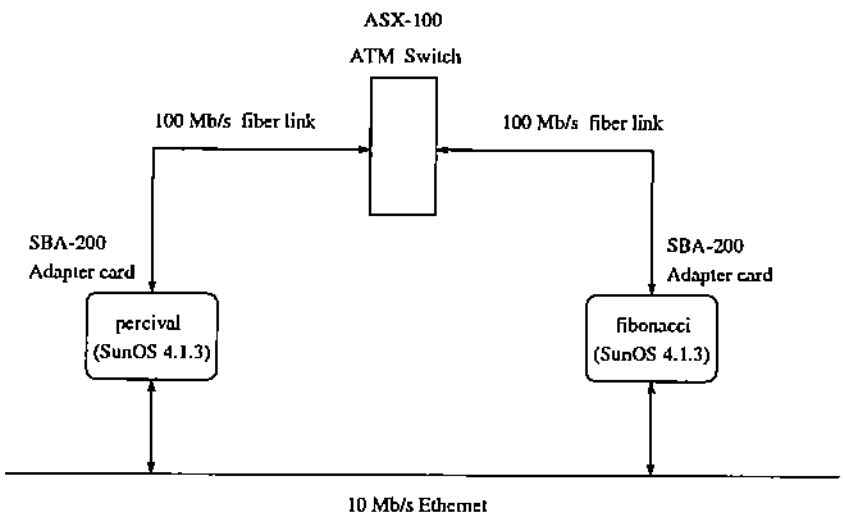

Figure 10: Experimental environment setup with Fore ATM switch

3.2.2 Experiment I: TCP Implementation on ATM

Problem Statement

These experiments are intended to measure the performance of the TCP and UDP protocols on an ATM network and to investigate the relative merits of an ATM network. Compared to the conventional Ethernet, the message transfer delay on an ATM network, with its high-speed fiber cable, is greatly reduced. TCP and UDP were developed before the existence of the ATM, and problems may therefore arise from the combination of new and existing technologies.

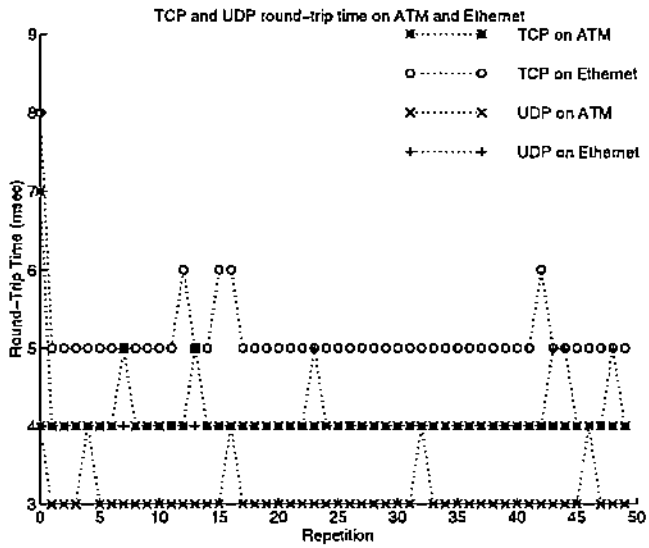

Figure 11: Round-trip time for 1 Kbyte messages with TCP and UDP using ATM and Ethernet

Procedure and results

In the experiments shown in Figure 11-13, 50 messages were sent repeatedly through both the ATM and Ethernet interfaces using the `ping` program described in Section 2.1. Figure 11 illustrates the results for the transmission of 1 Kbyte data. In this case, the ATM proved faster than the Ethernet for both UDP and TCP. Figure 12 shows the abnormal behavior encountered when sending 8 Kbyte messages with TCP on the ATM. Round trip times in

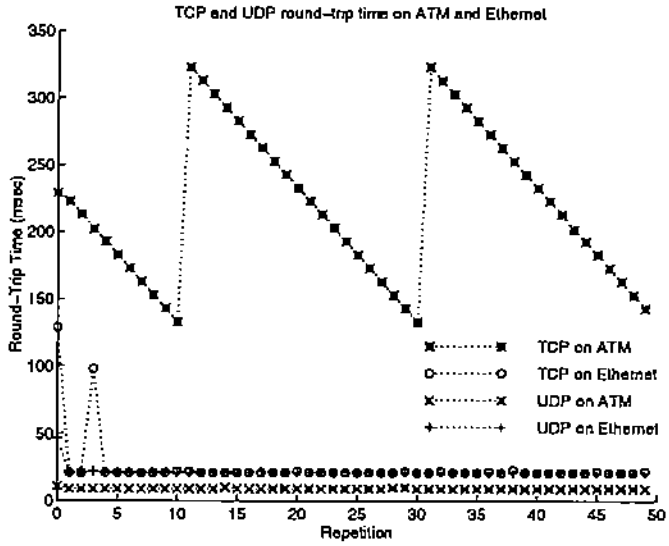

Figure 12: Round-trip time for 8 Kbyte message with TCP and UDP using ATM and Ethernet

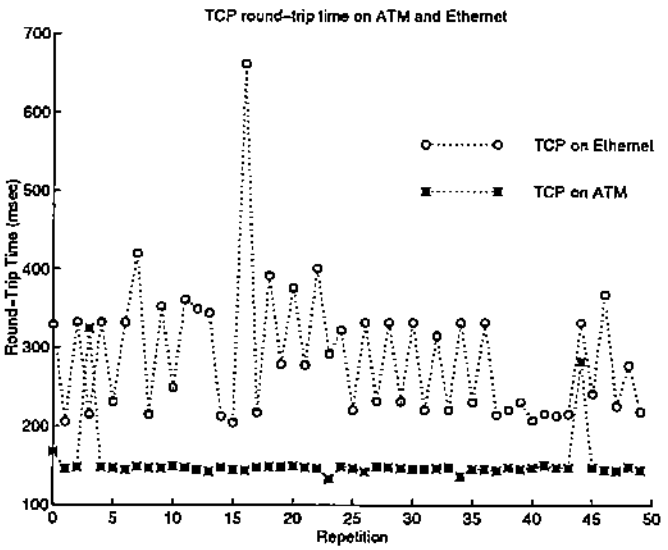

Figure 13: Round-trip time for 100 Kbyte message with TCP using ATM and Ethernet

this context fluctuated widely and were generally long. In contrast, Figure 13 reveals stable and short round-trip times for messages sent over the ATM network and long and oscillating round-trip times on the Ethernet. Figure 14 provides experimental results for sending files of size discussed in Section 2.1. While the average round-trip times on the ATM is long and almost constant for small files, the round-trip times measured for large files fits well with predictions. These large-file times are shorter on the ATM than on the Ethernet and increase with the size of files sent.

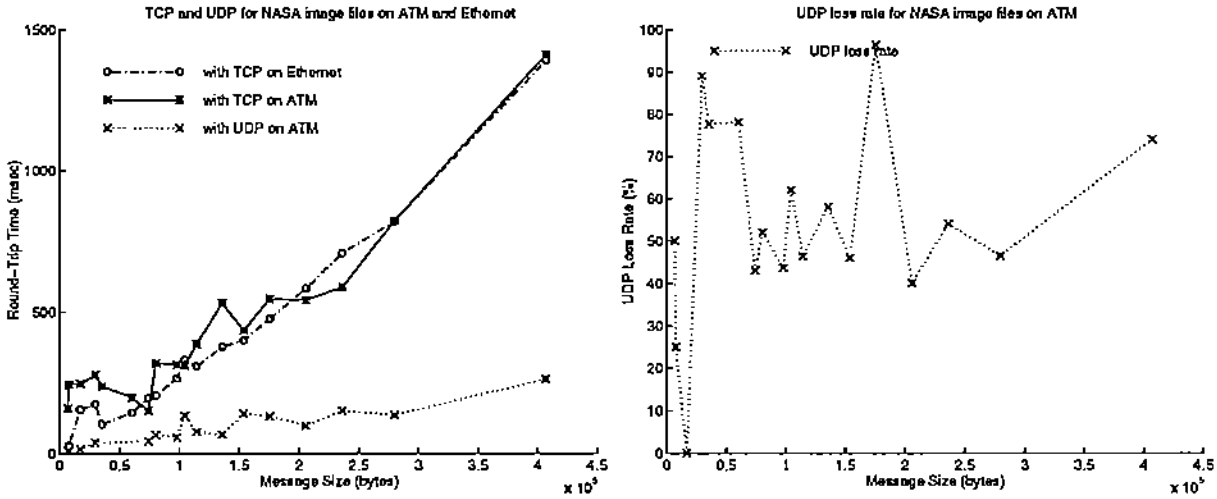

Figure 14: NASE image files on ATM a) Round-trip time; b) UDP packet loss rate

Discussion

The origin of the abnormal behavior noted above lies in the implementation of the TCP/IP protocol. Both [CL94] and [LMK90] provide detailed discussions on this question. Those factors which are most pertinent to our experiments are:

- Sender and receiver TCP buffer sizes;
- Physical MTU (Maximum Transmission Unit);
- TCP MSS (Maximum Segment Size);
- TCP buffer implementation and data delivery mechanism;

- The small packet avoidance algorithm; and
- The silly window syndrome avoidance algorithm of TCP;

By default, the sender and receiver window sizes are 16 Kbyte. The MTU of the Ethernet is 1500 bytes, while the MTU of the ATM is 9188 bytes. Accordingly, the TCP software sets the MSS to 1460 bytes for the Ethernet and 9148 bytes for the ATM. The MSS for TCP is calculated by the formula MSS=MTU−40, where 40 is the TPC and IP header size.

TCP Buffers are implemented as chains of *mbufs* [LMK90] in the SunOS 4.1.3. Each mbuf can store either up to 112 bytes of data or a pointer to a 1 Kbyte data page. The TCP output routine allocates mbufs to contain the TCP headers and the data. As long as 4 Kbyte of data is available in the buffer, the operating system invokes network routines to transmit a packet.

TCP will delay the sending of small packets according to the *Small packet avoidance algorithm* [Nag84]. Thus, TCP will send packets only under the following conditions:

1. when the connection is idle;
2. when $min(D, U) \geq MSS$, where D is the size of the data to be sent and U is the usable window size. The latter is defined as the receiver window size minus the amount of outstanding data sent by the sender for which no acknowledgement has been received;
3. when all outstanding data has been acknowledged;
4. when $min(D, U) \geq max\_sndwnd/2$, where max_sndwnd is the largest receiver window size.

According to the *Silly window syndrome avoidance algorithm* [Cla82, Jac88] in TCP, the receiver should avoid small window advertisements as a waste of bandwidth. The receiver TCP will send an ACK with a window update only under the following conditions:

1. when the amount of data received $R \geq 2 * MSS$;

2. when $R \geq 35\%$ receiver window size;

3. when *tcp_fasttimo()* timer expires (every 200 milliseconds).

Based on the previous discussion, when sending a 8 Kbyte message, TCP first places 4 Kbyte of data in the mbufs. This data is transmitted immediately, since the connection is initially idle. The remaining 4 Kbyte of data is then stored in the buffer. This second data installment cannot be be sent before the ACK of the first 4 Kbyte message is returned, since 4 K $<$ $MSS = 9148$ and 4 K $<$ half of the *max_sndwnd* (which is 16 Kbyte). On the receiver's side, when the first 4 Kbyte are received, none of the conditions for sending an ACK have been met unless the *tcp_fasttimo()* timer has expired. The ACK is therefore returned to the sender with an average delay of 100 milliseconds. As a result, the sender must wait for the ACK for an average of 100 milliseconds before the next packet can be sent. Our results indicates that this delay was uniformly distributed as $U[0, 200]$. Because the same situation arose when echoed messages were returned to the sender, the overall average round trip time is 200 milliseconds when sending 8 Kbyte messages.

In the experiment shown in Figure 12, 50 messages were sent at one-second intervals. This was implemented by calling the `alarm(1)` and `signal(SIGALRM, sig_handler)` routines. The two routines were separately tested and it was found that the `sig_handler` response experienced an approximately 10-millisecond delay each time. Consequently, the delay causes the `tcp_fasttimo()` timer to expire about 10 milliseconds ahead of schedule. As a result, the round-trip time measured in the experiment seems to fall into a cycle, declining by about 10 milliseconds until reaching the lower limit, and then initiating another cycle by jumping again to the maximum time.

Based on our previous discussion, we can predict the range of message size that may suffer abnormal network behavior:

$$(12n + 4)K + 1 \leq message\_size \leq 12(n + 1)K - 1.$$

The normal ranges are :

$$12nK \leq message\_size \leq (12n+4)K.$$

We conducted experiments with messages of size 4K, 4K+1; 12K−1, 12K; 16K, 16K+1; 24K−1, 24K; and 28K, 28K +1. The test results confirm the predicted behavior. For example, Figure 15 illustrates the results of sending 12K−1 byte data and 12 Kbyte data. The delay experienced when sending 12K−1 byte messages is dramatically longer than the delay incurred when sending a 12 Kbyte message. Since 1 Kbyte and 100 Kbyte messages are not in the range in which we predicted abnormalities, as shown in Figures 11 and 13, the delay we observed is normal. This explanation also justifies the abnormalities shown in Figure 14, in that round-trip times are exceedingly large for file size in out predicted abnormal ranges.

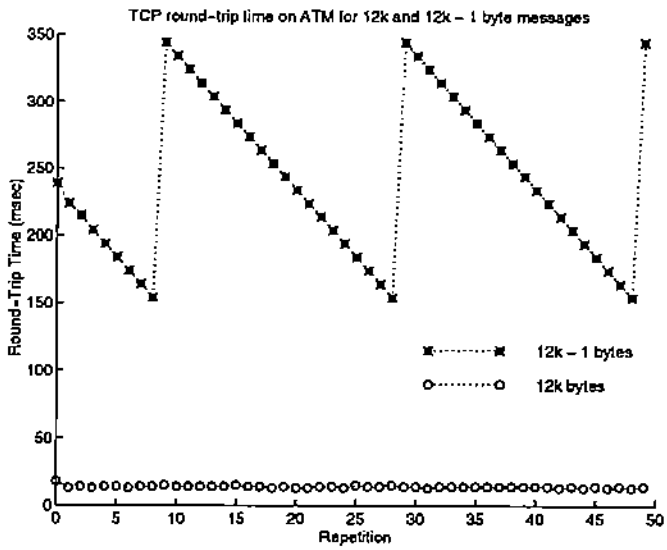

Figure 15: Transmission of 12 Kbyte and 12K-1 byte messages on ATM

A similar abnormality can also be observed on the Ethernet. Given MTU = 1500 for the Ethernet, we find that the abnormal ranges are

$$4K + (2n+1)M + 1 \leq message\_size \leq 4K + (2n+2)M - 1$$

where n = 0, 1, ...; K=1024, and M=1460.

The normal ranges are

$$4K + (2n)M \leq message\_size \leq 4K + (2n+1)M$$

where n=1, 2, 3, ...; K and M are as above.

For messages of 1 to 4 Kbyte in size, the network behavior of the Ethernet is normal. Abnormalities are found in the ranges from 4K+1 to 4K+M-1 byte. Experimental results obtained have been consistent with this prediction. While the abnormal range is 8 Kbyte and the normal range is 4 Kbyte on the ATM, the length of both the abnormal and normal ranges on the Ethernet are 1460 bytes. Therefore, if all data are uniformly distributed, the probability of abnormal behavior on the ATM is approximately 66%, versus 50% on the Ethernet. If message length distribution is skewed to small messages, the performance of an ATM network will be even worse compared with the Ethernet. In addition, since Ethernet speeds are low in comparison with the ATM network, the impact of abnormal behavior on the Ethernet is less severe.

3.2.3 Experiment II: Transmission of Data with TCP_NODELAY Option

Problem Statement

The TCP_NODELAY option, as discussed in Section 2.1 permits immediate delivery of small messages with TCP. The abnormal behavior described in the previous section was caused by the delayed transmission of packets waiting for an ACK. This problem may be addressed by using the TCP_NODELAY option to force transmission.

Procedure and results

Figure 16 provides the results from sending messages 8 Kbytes and 100 Kbytes in size, with the TCP_NODELAY set on both the ATM and the Ethernet. A comparison with Figures 12 and 13 shows that the abnormal behavior disappears when the TCP_NODELAY is set. Figures 17 and 18 provide additional test results with the TCP_NODELAY set.

Discussion

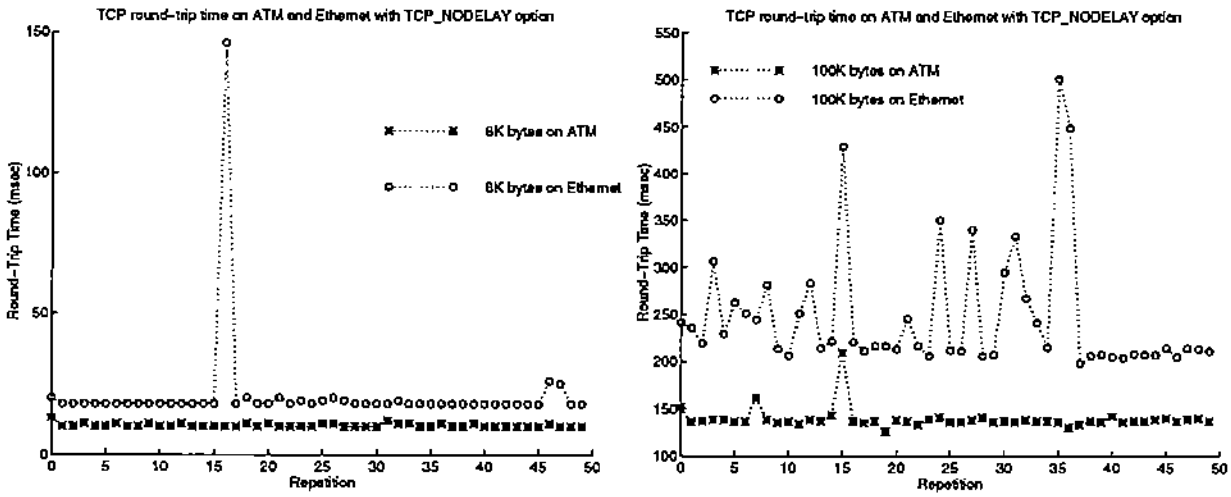

Figure 16: Transmission of data on ATM with the TCP_NODELAY option

Simply setting the TCP_NODELAY option when transmitting messages in the abnormal range is the easiest way to avoid abnormal delays, without modification of the operating system. This method also addresses the buffering problem described in [CL94]. We repeated the same test as [CL94] in the same experiment environment except that the TCP_NODELAY option was set in our tests. In comparison with the result shown in Table 2 from [CL94], Table 3 indicates that the abnormal behavior caused by the small packet avoidance algorithm disappears. One may observe that there are still cases in which the TCP throughput is abnormal in Table 3. This abnormal behavior, however, is due to the silly window syndrome avoidance algorithm. According to the silly window syndrome avoidance algorithm discussed in Section 3.2, an ACK will be delayed for 200 milliseconds unless the amount of data received is greater than either $2 * MSS$ or 35% of the receiver buffer size. On an ATM network where $MSS = 9148$, when the sender buffer size is only 16 Kbytes, the maximum size of the TCP packets sent is less than $2 * MSS$. Therefore, neither of the conditions for an immediate ACK is met when the receiver buffer is larger than 46 Kbytes. Consequently, after every delivery of 16 Kbyte data, the sender halts the transmission as the buffer filled up waiting for an ACK, while the receiver delays the ACK until its timer expires. Effectively, the TCP throughput is downgraded substantially because of the 200 millisecond delay for every 16 Kbyte message.

Setting the TCP_NODELAY option to avoid the abnormality, however, is not costless. Compared with Table 2, Table 3 indicates that the average throughput in the normal cases is lower in our tests. Delivery of small packets has higher overhead and thus waste available bandwidth. By setting TCP_NODELAY, the overall performance of the network is sacrificed to avoid abnormality for sending small packets. One should choose to set the TCP_NODELAY option by carefully investigate the sender and receiver buffer size, the MTU of the network, and more important, the applications existing on the network.

Receive Buffer Size (octet)

Send Buffer Size (octet)	16K	20K	24K	28K	32K	36K	40K	44K	48K	51K
16K	15.05	13.60	0.322	0.319	0.319	0.467	0.469	0.466	0.469	0.469
20K	15.99	14.60	15.07	14.87	15.40	14.24	1.095	1.095	0.548	0.549
24K	17.71	16.79	16.74	16.32	17.40	17.31	17.42	17.12	0.760	0.740
28K	16.57	17.69	17.93	18.13	18.36	19.20	19.74	19.78	18.38	18.20
32K	14.63	18.96	18.42	19.23	19.14	19.74	19.96	20.31	19.69	19.17
36K	14.33	19.22	18.12	19.82	19.77	19.92	20.56	20.49	20.13	20.20
40K	15.16	19.34	18.85	19.73	20.11	20.41	20.81	20.74	20.69	20.57
44K	14.80	19.40	18.27	20.39	20.16	20.74	20.99	20.87	20.89	20.70
48K	14.62	19.46	18.34	20.48	20.26	20.41	20.85	20.83	20.93	20.83
51K	13.92	19.41	18.26	20.50	20.06	20.21	20.88	20.91	21.21	21.06

Note 1: Throughput numbers are in megabits per second (Mb/s).
2: Shaded area indicates abnormal TCP throughput.

Table 2: TCP buffer size and mean throughput [CL94]

Receive Buffer Size (octet)

Send Buffer Size (octet)	16K	20K	24K	28K	32K	36K	40K	44K	48K	51K
16K	14.44	13.36	12.30	12.43	12.57	13.61	12.94	13.00	0.625	0.625
20K	15.74	16.49	16.04	16.00	16.87	15.23	11.85	15.40	13.92	13.82
24K	16.39	17.22	17.04	17.17	17.76	18.01	18.07	17.97	16.57	15.90
28K	12.80	17.10	17.32	17.44	17.54	17.56	18.00	17.92	16.25	16.71
32K	15.91	17.20	17.58	17.50	17.69	17.72	17.99	17.99	17.39	17.54
36K	17.16	17.84	16.84	18.04	18.06	18.39	18.09	18.37	18.39	18.20
40K	17.21	17.14	16.49	17.24	17.74	18.42	18.30	17.80	19.55	18.96
44K	16.02	15.39	14.67	15.43	14.09	15.23	14.32	17.38	19.05	17.86
48K	15.08	15.74	15.26	17.30	16.61	17.16	13.69	14.91	18.38	13.78
51K	14.00	16.33	14.56	14.47	16.22	14.35	15.80	15.63	15.50	15.45

Note 1: Throughput numbers are in megabits per second (Mb/s).
2: Shaded area indicates abnormal TCP throughput.

Table 3: TCP buffer size and mean throughput with TCP_NODELAY option

3.2.4 Experiment III: Transmission of Data with Larger Buffer Size

Problem Statement

For large messages, enlarging the buffer size may also improve network performance. The ATM network used in these experiments has an MTU=9188; this is, larger than that of the Ethernet (MTU=1500). The experimental results should show that using larger buffers improves the performance of the ATM network significantly.

Procedure and results

Figure 17 shows results for the transmission of 100 Kbyte messages with buffer sizes of 32 Kbytes and 16 Kbytes. Figure 18 provides results for large-size files. In these experiments, the ATM provides significant increment in speed over the Ethernet.

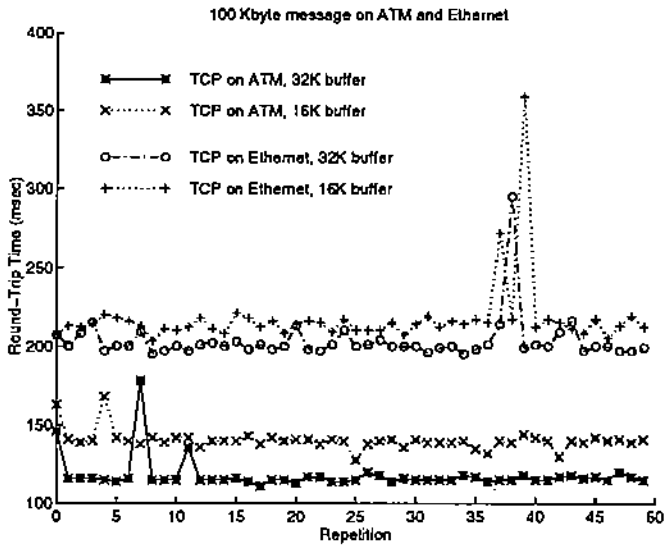

Figure 17: Round-trip time for 100 Kbyte messages on ATM with 32 Kbyte buffer and the TCP_NODELAY option

Discussion

The experimental results confirm our assumption a larger buffer size enhance ATM performance. Figures 17 and 18 indicate that ATM performance has been improved by ap-

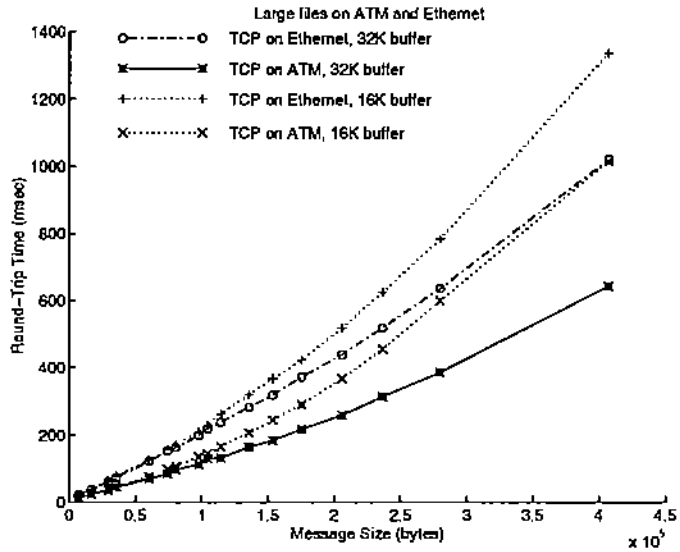

Figure 18: Round-trip time for NASA image files on ATM with 32 Kbyte buffer and the TCP_NODELAY option

proximately 30% with 32 Kbyte buffers in comparison with 16 Kbyte buffers for messages larger than 100 Kbytes. Furthermore, we observed that the performance of the Ethernet was not improved as significantly as that of the ATM when the buffer size changes from 16 Kbytes to 32 Kbytes; this improvement was only about 20%. It appears that a buffer of 16 Kbytes is sufficient to utilize the slower Ethernet, although it is insufficient for the ATM network.

3.3 Summary of TCP and UDP Performance on ATM

The results of our investigations indicate that the performance of UDP on the ATM is superior to its performance on the Ethernet. While loss rates on the ATM are about 30% lower, the unreliability of UDP still recommends against its use as the database communication protocol in most cases. The problems discussed in Section 2 with regard to the Internet apply as well to the ATM. However, while non-aggressive UDP packet delivery is not effective on an Internet WAN, its effectiveness on an ATM WAN is still unresolved, since the higher ATM speed produces lower loss rates.

Based on our investigation and analysis of the TCP/IP protocol and implementation, we have the following summary conclusions:

- Problems with the use of TCP have been discussed in [CL94, PR95, Rom93, AM94]. Our findings also indicate that implementation of TCP might fail to fully utilize the capabilities of an ATM network. The TCP protocol and its implementation must undergo further evaluation before it becomes the protocol of choice for the future network. In particular, modifications must be made to the small packet and silly window syndrome avoidance mechanism and the packet delivery mechanism.

- If it is undesirable to alter the TCP implementation, the TCP_NODELAY option may be set at the database application level to avoid abnormal behaviors.

- Users may employ a larger sender and receiver buffer. This should improve performance at the application level on a higher-capacity network such as ATM.

- If fully utilized, an ATM network provides high-throughput transmission for large messages. Therefore, ATM is ideal for multi-media applications in which large amount of higher level data transmissions are not uncommon.

- The advantages of ATM is more obvious in a WAN environment since a ATM switch route messages much faster than current routers. Reserved channel capacity can reduce the loss of cells and improve reliability. Therefore, the ATM should be the choice for the database topological and metric scale-up. As shown in both our local experiments and other research [PR95, Rom93, AM94], however, current TCP buffering mechanism, transmission implementation and congestion control protocol may not be suitable for the ATM. Further experimental study is necessary on ATM technologies in a WAN environment.

4 Conclusions

Experiments were conducted on both the conventional Ethernet network and the ATM network with the TCP and UDP/IP protocols in order to generate findings of use to devel-

opers of distributed databases. The relative merits of each scenario were assessed on the basis of these experimental data. In general, it was found that the TCP protocol is very slow but reliable for large data messages and transmission between remote sites. The UDP protocol with a few adjustments, however, is found useful for time critical applications that can tolerate a certain amount of data loss. Abnormal behaviors were observed during the performance test of TCP and UDP on an ATM LAN. We found that higher capacity of an ATM network does not necessarily mean enhanced performance to higher level protocols, if higher level protocols are not configured appropriately to take advantage of the ATM technology. Careful investigation is advisable for distributed database developer choose the right parameters for their applications.

Previous research [BZM91, MB91] suggested the use of UDP to send small packets in a LAN. In a WAN environment, however, TCP is preferable, especially for large messages. Therefore, TCP is the choice of data communication for database scale-up. Data transmission in a WAN usually suffers high packet loss rate that common applications can not tolerate. Despite the low speed, TCP's guaranteed reliability makes it the protocol of choice for common applications. But exception can be found in applications such as video conferencing, which is time critical but is not affected by data loss as seriously. Using UDP datagrams, instead of TCP packets, to transmit data can provide the required high speed with acceptable qualities to applications of these kind. Furthermore, we found in our experiments that a small increase in the time interval between each UDP datagram during the transmission may decrease the data loss rate significantly. Adjustments, such as the insertion of a short `for` loop between the transmission of each datagram, are found quite effective to improve the reliability of UDP with little negative impacts on the speed. It is worth mentioning that such adjustments are machine and application dependent. Developers of a distributed database system must balance their need for reliability and speed in order to optimize the network performance of their system.

In general, an ATM network provides much higher transmission throughput compared to the conventional Ethernet. But higher level protocols must be adjusted in order to take advantage of such improvement. The TCP/IP implementation of existing systems is assuming a low speed physical network such as Ethernet, and thus may not be suitable the high throughput ATM. Abnormal behaviors are experienced during our experiment. Simple adjustment, such as setting the TCP_NODELAY option, works to address this problem. But the TCP_NODELAY option should be taken as a quick fix rather than an optimal solution because it downgrades the throughput in the average cases. Choosing larger buffers on both the sender and receiver's side address this issue while generating higher throughputs. But buffers are allocated in the host machine's memory, which is usually limited resource shared by multiple users/applications. It is not possible for a real time system to allocate arbitrarily large buffers to optimize performance for each individual application. While choosing large buffers works in our experiment to optimize data transmission throughput between two test applications on a dedicated ATM network, how well this solution works in pratice remains as an open question.

Despite the abnormalities, our experiments indicate that a high capacity ATM network will enhance network performance significantly if it is fully utilized. In the normal cases, the ATM network over-performs the Ethernet for both TCP and UDP. Considering its high capacity and the potential to grow, the ATM network is ideal for the next generation distributed database applications.

Based on our studies, we suggest future research on the following issues:

- Further study on a standardized ATM network will be necessary when a universal standard of the ATM exists. Neither the hardware nor the software of the ATM network used in our experiment is produced under a universally recognized standard. Because the issues discussed in this research are machine and application dependent, the experiment results and solutions may not apply to different systems. Research for a widely applicable solution is yet impossible because the lack of standards in the

field of ATM.

- The existing network protocols, such as TCP/IP, should undergo evolution in order to adapt to the improving physical network. Higher level applications, such as the distributed database applications, rely on TCP/IP to achieve better performance. How TCP/IP can take full advantage of the higher capacity provided by the ATM network proves crucial to the overall performance of the system.

- Performance study of a real world ATM network may reveal issues that are left out in this paper. Our experimental environment is an ideal environment, in which two dedicated machines enjoy an otherwise idle ATM network. Studies have shown that an ATM network may behave quite differently when it is saturated and is forced to drop cells [PR95, Rom93, AM94]. It is still an open issue on how to optimize the overall performance of a system in practice.

- Studies of an ATM WAN may bring interesting findings. Our performance test on the ATM network is restricted to a LAN because of our limited resource. Since ATM is intended to carry data traffics for both LAN and WAN. An ATM network of large scale should undergo thorough study when it becomes available.

- Message multi-casting on ATM is another interesting research topic. When a distributed database topologically scale up, many messages would be sent to large number of remote sites simultaneously. How well the switching mechanism of ATM could support group communication of this kind is now an open question.

Acknowledgement

The authors would like to thank Professor Douglas E. Comer for providing support for our experiments. The authors are also very grateful to John Chueng-Hsien Lin for answering many of our questions. Melliyal Annamalai selected NASA image files for our experiments. Yue Zhuge at Stanford helped us with the WAN experiments. We would like to acknowledge Rachel Ramadhyani for her input regarding the presentation of this paper.

References

[AM94] Aboul-Magd. TCP Performance in ATM Networks Employing End-to-End Flow Control. Technical Report 94-0442, ATM Forum Contribution, May 1994.

[BR89a] B. Bhargava and J. Riedl. A Model for Adaptable Systems for Transaction Processing. *IEEE Transactions on Knowledge and Data Engineering*, 1(4), December 1989.

[BR89b] B. Bhargava and J. Riedl. The RAID Distributed Database System. *IEEE Transactions on Software Engineering*, 15(6), June 1989.

[BZM91] B. Bhargava, Y. Zhang, and E. Mafla. Evolution of Communication System for Distributed Transaction Processing in Raid. *USENIX Journal Computing Systems*, 4(3):277–313, Summer 1991.

[Che88] D. R. Cheriton. The V distributed system. *Communications of the ACM*, 31(3), March 1988.

[CJRS89] D. D. Clark, V. Jacobson, J. Romkey, and H. Salwen. An Analysis of TCP Processing Overhead. *IEEE Communication Magazine*, 27(6):23–29, June 1989.

[CL94] D. E. Comer and J. C. H. Lin. TCP Buffering and Performance over an ATM Network. *Journal of Internetworking: Research and Experience*, 10(4):70–80, October 1994.

[Cla82] D. D. Clark. Window and Acknoledgement Strategy in TCP. *Request for Comments*, (RFC-813), July 1982.

[Com91] D. E. Comer. *Internetworking with TCP/IP Vol I: Principles, Protocols, and Architecture*, volume I. Prentice Hall, Inc, Englewood Cliffs, NJ, second edition, 1991.

[CS92] J. D. Cavanaugh and T. J. Salo. Internetworking with ATM WANs. Technical report, Minnesota Supersomputer Center, Inc, December 1992.

[Gol92] R. A. Golding. End-to-end performance prediction for the Internet. Technical Report UCSC-CRL-92-26, University of California at Santa Cruz, June 1992.

[Jac88] V. Jacobson. Congestion Avoidance and Control. In *Proceedings of ACM SIGCOMM'88*, pages 314–328, August 1988.

[Kun92] H. T. Kung. Gigabit Local Area Networks: A Systems Perspective. *IEEE Communication Magazine*, 30(4):79–89, April 1992.

[KW95] B. G. Kim and P. Wang. ATM Network: Goals and Challenges. *Communications of the ACM*, 38(2):39–44, February 1995.

[LMK90] S. J. Leffler, M. K. McKusick, and M. J. Karels. *The Design and Implementation of the 4.3BSD UNIX Operating System*. Addison-Wesley, Inc, Reading, MA, 1990.

[MB91] L. E. Mafla and B. Bhargava. Communication facilities for distributed transaction processing systems. *IEEE Computer*, pages 61–66, August 1991.

[Nag84] J. Nagle. Congestion Control in IP/TCP Internetworks. *Request for Comments*, (RFC-896), January 1984.

[OCD+88] J. K. Ousterhout, A. R. Cherenson, F. Douglis, M. N. Nelson, and B. B. Welch. The Sprite network operating system. *IEEE Computer*, February 1988.

[PHOR90] L. Peterson, N. Hutchinson, S. O'Malley, and H. Rao. The *x*-kernel: A Platform for Accessing Internet Resources. *IEEE Computer*, 23(5):23–33, May 1990.

[PKL91] C. Pu, F. Korz, and R. C. Lehman. A Measurement Methodology for Wide Area Internets. Technical Report CUCS-044-90, Columbia University, March 1991.

[Pos80] J. B. Postel. User Datagram Protocol. *Request for Comments*, (RFC-768), August 1980.

[Pos81] J. B. Postel. Transmission Control Protocol. *Request for Comments*, (RFC-793), September 1981.

[PR95] M. Perloff and K. Reiss. Improvements to TCP Performance in High-speed ATM Networks. *Communications of the ACM*, 38(2):90–100, February 1995.

[PSC81] J. B. Postel, C. A. Sunshine, and D. Cohen. The ARPA Internet Protocol. *Computer Networks*, 5(4):261–271, July 1981.

[Ras86] R. F. Rashid. Threads of a New System. *Unix Review*, 4(8):37–49, August 1986.

[Rom93] A. Romanow. Preliminary Report of Performance Results for TCP over ATM with Congestion. Technical report, July 1993.

[SSU91] A. Silberschatz, M. Stonebraker, and J. Ullman. Database Systems: Achievements and Opportunities. *Communication of ACM*, 34(10):110–120, 1991.

[Ste90] W. R. Stevens. *Unix Network Programming*. Prentice-Hall, Inc., 1990.

[TRvS+90] A. S. Tanenbaum, R. V. Renesse, H. van Staveren, G. J. Sharp, S. J. Mullender, J. Jansen, and G. van Rossum. Experiences with the Amoeba Distributed Operating System. *Communications of the ACM*, 33(12):46–63, December 1990.

[Vet95] R. J. Vetter. ATM Concepts, Architectures and Protocols. *Communications of the ACM*, 38(2):30–38, February 1995.

[ZB93] Y. Zhang and B. Bhargava. WANCE: A Wide Area Network Communication Emulation System. In *Proceedings of the IEEE Workshop on Advances in Parallel and Distributed Systems*, pages 40–45, Princeton, New Jersey, October 1993.