A Study of Distributed Transaction Processing in Wide Area Network

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

Yongguang Zhang

Report Number: 94-016
A STUDY OF DISTRIBUTED
TRANSACTION PROCESSING
IN WIDE AREA NETWORKS

Bharat Bhargava
Yongguang Zhang

Computer Sciences Department
Purdue University
West Lafayette, IN 47907

CSD TR-94-016
March 1994
A Study of Distributed Transaction Processing in Wide Area Networks *

Bharat Bhargava
Yongguang Zhang
Department of Computer Sciences
Purdue University
West Lafayette, IN 47907

{bb,ygz}@cs.purdue.edu

Purdue Technical Report CS-TR-94-016
March, 1994

Abstract

Our research presents an experimental study that evaluates distributed transaction processing under the wide area network environments. Experiments that study the performance of wide area network communication have been completed. Several problems in message delivery and their impact in distributed transaction processing have been identified. We observed large variations in the communication delay and pattern of failures. We also conducted experiments by connecting sites around the world using an emulation tool we developed. We measured the performance of the industrial standard DebitCredit benchmark transactions for sites involved in Indiana, Illinois, Texas, Finland, and Hong Kong. We observed that the communication latency in wide area network is a decisive factor for the transaction performance, and the size of the messages does not have a strong influence. We concluded that for a small multi-programming level between three and five yielded better throughput. We identified the bottleneck of a transaction processing system in a wide area network environment to be the atomicity control, three-phase commit was not a tolerable solution. We also suggested some directions of improvement in both communication facilities and transaction processing algorithms that we continue our research efforts.

*This research is in part supported by Army Research Lab (Software Technology Branch) and AT&T.
1 Introduction

There are two dimensions of scaling in distributed systems namely, horizontal scaling and vertical scaling. Horizontally scaled distributed systems comprise of a large number of participating sites (e.g. in hundreds of sites). Vertically scaled distributed systems consists of geographically dispersed sites interconnected by long-haul networks. Most previous research efforts in the distributed systems have assumed (either implicitly or explicitly) an environment with a small number of sites interconnected by a local area network (LAN). It makes it difficult to realize and certainly unobvious to comprehend as to how the applications and the theory developed in such an environment would behave in a horizontally or vertically scaled systems. One such application that interests us, is the distributed transaction processing in these “newer” environments. The understanding of the implications of migrating the system to a different and perhaps more unreliable environment will not only have a significant impact on the future applications for distributed systems, but also the design of the existing applications [OV91, SSU91, Sto89]. Our study entails the development of mechanisms for scaling the Distributed Transaction Processing (DTP) to the wide area network (WAN) environment and conduct experiments to analyze the effects of this movement on various components of DTP, like concurrency control and atomicity control.

The communication software has been deemed as a crucial component of DTP software [Svo88, Spe86]. All the other components of a DTP system rely on this for high-performance and possibly reliable communications. Atomic commit protocols require the exchange of control messages among the participating sites for ensuring the global commit/abort of an active transaction. Replication control protocols require the transmission of both the data items and the control (e.g. votes) messages for maintaining the consistency of replicated copies. Some distributed deadlock detection algorithms exchange information periodically to determine and break the cycles in distributed “wait-for” graph. Monitoring services such as surveillance controller [HZB92] sends Lam_alive messages for node/link failure detection, to all the other sites. The control messages although small (usually ranging in tens of bytes), are frequently exchanged. For some components, the frequency of these exchanges will determine their effectiveness. For example, in the deadlock detection component, the more often the messages are exchanged, the more prompt will be detection of a cycle. Overall, the DTP demands high performance and reliable communication. These requirements become more crucial and difficult to satisfy in WAN environment.

The performance of the communication software is largely dependent on the underlying communication media. Networks with different technologies and characteristics have been merged by the internetwork connections. Thus, the communication network spanning large number of geographically dispersed hosts will vary in speed, reliability, and processing capability. The range of these parameters across networks is growing [Com88, Tan88, Wit91]. For example, a distributed system spanning both ATM and ethernet networks has bandwidth variations between 145 Mb/s to 10 Mb/s.

From the preceding discussion, the need for designing efficient and reliable communication facilities and scalable algorithms for DTP software components, is apparent. We have experimented with the transaction processing on the Internet using different protocols for concurrency and atomicity. We have made an attempt to understand the impact of multi-programming levels on these protocols in this “new” environment. We report the results of the performance evaluation of these protocols in the WAN environment. We discuss the problems encountered, solutions proposed and our experiences in moving the transaction processing from LAN to WAN are discussed to act as
2 Communication in Wide Area Network

We have conducted a series of measurements to study the performance of communication in WANs. We present several results and observations obtained through the experiments. We analyze them and see how communication affects transaction processing in WANs.

2.1 Purpose of the study

<table>
<thead>
<tr>
<th>Issues</th>
<th>LAN</th>
<th>WAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scale</td>
<td>small (10-100 sites)</td>
<td>large (over 100 sites)</td>
</tr>
<tr>
<td>Geographic span</td>
<td>within a mile</td>
<td>over 100 miles</td>
</tr>
<tr>
<td>Topology</td>
<td>bus, ring</td>
<td>hierarchical</td>
</tr>
<tr>
<td></td>
<td></td>
<td>interconnected</td>
</tr>
<tr>
<td></td>
<td></td>
<td>irregular mesh</td>
</tr>
<tr>
<td></td>
<td></td>
<td>multiple hops,</td>
</tr>
<tr>
<td>Routing</td>
<td>simple (direct link or 1-2 hops)</td>
<td>dynamic</td>
</tr>
<tr>
<td>Speed</td>
<td>very high (10-1Gbps)</td>
<td>low (1Kbps-10Mbps)</td>
</tr>
<tr>
<td>Error rate</td>
<td>low</td>
<td>high</td>
</tr>
<tr>
<td>Variation</td>
<td>small</td>
<td>large</td>
</tr>
<tr>
<td>Ownership</td>
<td>private</td>
<td>public</td>
</tr>
<tr>
<td>Access</td>
<td>under the same authority</td>
<td>shared &amp; public network</td>
</tr>
<tr>
<td></td>
<td></td>
<td>user has no control</td>
</tr>
</tbody>
</table>

Table 1: Differences in LAN and WAN environments

WAN differs from a local area network (LAN) in several ways (see Table 2.1). The performance characteristics are significantly different:

1. LANs are usually fast and very reliable. The communication delay and failure pattern are uniform and small for all hosts. In the Internet, error rates are higher, communication delays large and non-uniform [Wit91]. The links between two sites in the Internet can range from 56KB/second (lease line) to 45MB/second (T3 link). The number of hops (gateways) between two sites can vary from a few to a very large number.

2. A LAN does not require routing. Even in multi-LAN environment, routing is simple and almost fixed. In the Internet, source routing is dynamic. The routes between sites can change from time to time [PKL91].

3. A LAN consists of a few hosts (e.g. around hundred maximum) under the same administrative unit. Internet connects hundreds of thousands of hosts all across the world, spanning
several organizational boundaries. Thus, it is subject to conflicting administrative policies and different usage characteristics.

These differences between LAN and WAN makes many of the communication solutions applicable to LAN unfounded in a WAN. For example, physical multicasting, widely accepted for distributed transaction processing in a LAN environment, is inadequate for WAN environment.

Purpose of the measurements

To study the implications of the WAN communication on the performance of DTP, we need to study the performance of transaction message delivery in the Internet. We are specially interested in two measurements: the round-trip time of a transaction message and the message loss rate. By round-trip time we mean the time for a message to travel back and forth across the Internet. In our DTP model one site sends a request message to another site and gets a reply message back. By loss we mean that the transport service of the Internet fails to deliver the message in time. This is important since a lost message will not only block the transaction and can cause it to be rolled-back and restarted, but also increase the contention of shared data such as the indices.

2.2 Design of the Measurements

We first identified the variables of our study. Based on our understanding of the Internet, we believed that three factors are most important in determining the performance of message delivery: the physical connection, the size of message, and the cross-traffic. The physical connection between two sites includes the distance, the type of links, and the number of hops (gateways). It determines the lower bound of the communication latency. To establish their relation, we needed to study how the performance changes across different sites in the Internet. The size of message also affects the transmission time, which is also worthy studying. The cross-traffic causes delays and losses, however it is much more difficult to establish the its connection with communication performance, because there is no easy way to determine the cross-traffic on a particular link. One way to circumvent the problem is to determine the traffic pattern by large in the Internet setting. We suspected that it is a function of the working hours, since Internet usage determined the traffic pattern. To verify this, we studied how the time of the day and date of the week affects the message delivery performance in the Internet.

In summary, we designed measurements in three dimensions: the time dimension, in which we repeated experiments from time to time, the site dimension, in which we repeated experiments using many different sites, and the size dimension, in which we varied the size of the messages.

Measurement tools. In Raid and many other DTP systems, UDP is used to meet the performance and real-time requirements [Spe86, Svo86, BZM91]. Therefore, we shall measure the round-trip time and loss rate for messages delivered in UDP protocol. A good measurement tool for this is the Unix ping program. It sends ICMP (Internet Control Message Protocol) echo request messages to a specified host and wait for replies. Any host in the Internet that receives this echo request message will formulate an identical echo reply message and return it to sender. An ICMP message is a good measurement for a UDP message since they both are encapsulated in an IP datagram and delivered in the same manner. Therefore they have the same round-trip time and the same probability of loss.
Parameters. We carefully choose the parameter to reflect the reality of message passing in DTP. We form an echo request message to be the transaction message body, the UDP header, plus a timestamp and the sequence number. For control message, the packet size formed this way is 64 bytes long. The timestamp and the sequence number are used to correspond the reply message to its request message can calculate the time elapsed (i.e., the round-trip time). Each sampling usually consists of a batch of messages that simulate several rounds of messages for a transaction. To eliminate cold start effect, we add one artificial message in the beginning of each batch and drop this datum from samples. We set the time-out value for each message to be 10 seconds. It a reply message does not come back during that time, it is considered lost. We can count the loss by subtracting the number of echo messages sent from the number of reply messages received.

2.3 Procedure and Results

Measurement I: Time variant

We chose raid8.cs.purdue.edu (a SparcStation in our laboratory) as the sender site and airmics.gatech.edu (a SparcStation in Atlanta, Georgia) as the echo site. Each batch of polls consisted of 22 messages (plus one for cold start effort elimination). The messages were sent continuously at one-second intervals. We repeated such sampling every 5 minute in 4 weeks (from 1/08/1992 to 2/05/1992). The data consist of 8,200 batches that involve more than 180,000 samples.

We compressed the data into hourly mean time for round-trip (MTFRT) by calculating the means and variances of the round-trip times in each hour. Since our measurement involved 22 valid polls every 5 minute, we have 264 samples every hour. We calculated the means, variances, as well as 95% confidence intervals of the round-trip time samples in each hour. The percentage of messages lost in each hour is also calculated this way.

Figure 1 shows this MTFRT and loss rate. The X-axis is the actual time when the measurement was taken. Table 2 shows the distribution of the hourly average round-trip time and average loss rate. Figure 2 shows the distribution of the round-trip time and loss rate in normal work days. The graphs are made by accumulating all the sample data that were measured on Tuesday, Wednesday, and Thursday, totally 12 work days in our 4 weeks experiment period. Both sites were in eastern time zone when the measure took place.

From the results we can see the large variance in the round-trip time, and the unreliability of the Internet. 103 response time samples (out of 8270) are lost or not delivered in time. Although the average MTFRT was approximate 105ms, there is still a total of 23 hours out of 4 weeks when the average round-trip time is larger than 130ms. Over 7.8% of the messages were lost in that time. The round-trip time for successful messages, ranges from 84ms to 622ms and has a variance of 424. When the mean time for round-trip goes up, the variance of the round-trip time tends to be larger, and the messages loss also tends to be more frequent. Although a change in these two parameter can happen every quickly, the periods of degraded performance are not necessary to be transient. For example, there was a two-hour failure (no messages can arrive in time) in January the 14th, and a long term semifailure (message delivery is much slower than normal and the loss is high) that lasted a day from January the 20th to the 21st.

Cold start effect is the phenomenon that the round-trip time of the first packet in each batch is much longer than the subsequent ones. It is due to the start-up costs such as warming up host-name to IP address resolution and routing table, etc. The common solution is to drop the first packet in the batch of samples [PKL91].
We can see from the results that the message loss rate is high in the afternoon work hour, and low in the early morning before work hour. The message round-trip time, on the other hand, does not have a strong correlation with the time of the day. The message delay is a little bit longer around noon. But the dependency between the round-trip time and the time of the hour seems very interesting. It takes much longer to deliver a message during the 5 minutes past every hour.
### Table 2: Distribution of the hourly MTFRT

(see the upper graph in Figure 2). We guess the reason is that many hosts and gateways run some hourly jobs at the hour time, which increase the load of the systems. However, the message loss rate does not seem to be affected.

**Measurement II: Site variant**

This measurement is to how difference one can send a transaction message from one site to different other sites in a large scale wide area systems. We choose *major* hosts in the .edu domain (computers in American educational institutes) as the subjects of our measurement. *Major* hosts denote the hosts that are important to each individual network. They are usually large computers used for backbone hosts, mail homes or mail exchanging gateways, name servers, etc. They can represent all other hosts of the same local network. Also, these major hosts are usually fast, stable, and close to the campus backbone.

To get a list of the major hosts for our measurement, we started from the official DDN Internet Host Table, maintained by NIC (Network Information Center). The DDN Internet Host Table consists of the mapping from IP addresses to host-names. Although this table may not be completely updated, it contains the important hosts in many campuses. These "official" hosts form a substantial fraction of the major hosts in the Internet. Starting from this table, we extract all the host-name within .edu domain and the corresponding IP addresses. This results in over 2,500 hosts. To ensure that the mapping from IP addresses to host-names is up-to-date, we queried the name server for the authoritative answer. and there was no name server answer from a few hosts. After removing the incorrect mapping and obsolete hosts, as well as hosts that are in Purdue campus within domain purdue.edu (we are not interested in local hosts), we had a total of 2108 hosts for experiment. These 2108 hosts span over 514 IP networks\(^2\), all of which are registered in NIC.

\(^2\)Although there can be many subnetworks in each IP networks, as long as they have the same netid in their IP addresses, they are considered the same IP network, and will be treated as the same destination network in IP routing [MP85, RP87].
That is, about 4 major hosts per IP network.

The sender site is still "raid8 as in the first experiment. We repeatedly sent transaction messages
The distribution of the average round-trip time to a host can be reached in around 200ms. The variation, however, is extremely high. The standard deviation for the round-trip time to these 1843 major hosts is 231. On the contrary, the variation of round-trip time between any two hosts in a local network is usually very small. We have repeated the same measurement for 413 hosts in the Purdue University Computer Center Network, a separated network from our department network. Our statistical result for these 413 local hosts shows a standard deviation of only 6.3.

This also shows the “clustering” effect in the Internet topology. This shows that communication between one site to many different sites in another local network will have the similar performance, which could be represented by a major host of that network. Therefore, the latency from one network to another network can be used to estimate the communication delay between two hosts in these two networks.

Measurement III: Correlation between size and round-trip time

Besides control messages, data messages are also involved in DTP. Unlike control messages, data messages are usually longer and not fixed in size. For large messages, an increase in round-trip
time is expected. On the other hand, the round-trip time will not be proportional to the size of a message, because the computation involved in messages delivery, such as source routing and flow control, is not related to the size of the message. There are two questions: how the size of messages will affect the the round-trip time, and how the size of messages will affect the reliability?

To answer these questions, we used various size of messages in this measurement. The message round-trip time betweenector.cs.purdue.edu and rainbow.uta.edu (a SparcStation located in UT Arlington, Texas) is measured. Each sample of the experiment consists of 63 batches of the messages. of 63 batches of messages sent in 1 second interval. Each batch had 10 messages of the same size and was sent in one second interval. Messages in the first of the 63 batches are all 32-byte long, messages in the second batch are 64-byte, and so on, with the messages in last batch being 2016-byte long.3 The whole set of experiment was repeated 72 times to give the sample size of each data point of 720. We averaged the round-trip time of all the messages with the same size. The result is shown in figure 5. We plotted the average message loss rate under different message sizes in the figure.

Figure 5 indicates a strong correlation between the the message size and the average round-trip time, but the message loss rate is not a function of the message size. The average round-trip time increases linearly when the message size is below 1472. However, when the message size is bigger than 1472, the round-trip time surprisingly tends to be constant and climbs at a much slower rate.

One interpretation is that this is due to the IP fragmentation. Both sites in the experiments, ector and rainbow, are all connected to the Internet via the local Ethernet. Since the maximum Ethernet frame is about 1,500 bytes, any messages that is larger than 1,500 bytes will be fragmented into 2 messages by the network interface, each traveling independently, and will be reassembled at

3We could not send out ICMP echo messages larger than 2K due to the OS limitation.
the destination host. Because of the stop-and-forward nature of the Internet delivery, sending a message in two fragments could benefit due to the back-to-back "pipeline effect" and is faster than sending the message in one packet.

This seems contradictory to the common understanding that transferring data in large size packet would improve performance. Although sending messages in small size packet reduces the response time, it demands higher bandwidth, posts higher traffic for the gateway (in term of number of packets), and increases the probability of message loss (splitting one message into two would double the chance that one of the message is delivered in time). It is worth investigating the mechanism to fragment a message to achieve best performance.

2.4 Conclusion

In summary, from these measurements we obtained the follow observations.

- First, we observed that there are large variations in parameters such as communication delay and message loss. The variations exists in two dimensions: along the time axis and across the networks.

- We observed that the time of day has strong influence on the message delivery. It is noticed that the message loss rate is much higher in the afternoon working hour, and much lower in

Figure 5: The relations between the message size and the average round-trip time, average message loss rate.
the early morning before the working hour. The message round-trip time, on the other hand, does not have a strong correlation with the time of the day, except for the hourly peaks, which we suspect are caused by hourly jobs scheduled to run on gateways.

- We observed that message delivery had a unbalanced performance across the wide area networks, although most of the hosts can be reached within 400ms round-trip. The data also shows the "clustering" effect in the Internet topology. The communication between one site to many different sites in another local network has the similar performance, which could be represented by a major host of that network. Therefore, the latency from one network to another network can be used to estimate the communication delay between two hosts in these two networks.

- We also observed that for small messages that fit in a IP datagram without fragmentation, there is an approximate linear correlation between the transit time and the size of a message. However, the messages loss is not affected by the size.

2.5 Impact on Distributed Transaction Processing

The performance analysis of communication in the Internet, reported in the previous section will also have a significant impact on distributed transaction processing on the Internet.

The time to deliver a transaction message in the WAN is a number of magnitude longer than in a LAN. While it takes only a few milliseconds to deliver a message in a LAN [AK91], on the Internet it is several hundreds of milliseconds to send a message across the continent [GL91]. This means that a transaction stays longer in the system, implying the larger lock holding time for data items, if two phase locking is used for concurrency control. This leads to increased contention to the database, affecting the throughput adversely.

The already difficult problem of finding a "good" value for timeout in LAN is further aggravated in WAN environment. Timeout is used in DTP systems to trigger special treatment for the transactions that cannot be finished in time. The timeout value usually equals a constant multiplied by the number of read/write operations in the LAN environment. In a WAN, this flat timeout rate is not adequate. As the CPU and disk I/O performance improves, most of the time spent for a transaction is in the waiting for the messages to be delivered. Thus, the timeout value for a transaction must be dependent on the number of remote messages and their destinations.

Autonomous control over LAN allows modification to the communication software improving the performance of DTP. [BZM91] discusses many changes, such as physical multicasting, lightweight protocols, etc that can be affected. Physical multicasting is not supported by all WANs. Direct control passing or memory mapping may not have a significant impact, because the message delivery latencies may cause a performance bottleneck. Unless dedicated links or special networks are adopted, one cannot do much to the shared public WAN such as the Internet. The performance of message delivery is determined by traffic and various other factors beyond the designer's control. Therefore, the focus of improving communication has to be shifted towards reducing the number of messages exchanged in DTP.

The mechanism to handle message loss in DTP has to be changed. In a LAN environment, a lost message will cause the DTP system to time-out the transaction and restart it later. DTP systems seldom adopt a transport service that guarantees reliable delivery but imposes high overhead [Spe86, BZM91]. This approach works fine because the message loss rate is very low. In a WAN, however,
message loss rate is much higher. The percentage of message loss is usually 5%, sometimes as high as 30%. Frequent transaction abort and restart caused by message loss will drastically degrade the overall performance of DTP. Transport protocols that have higher degree of reliability should be considered.

DTP algorithms must be able to adapt to the high variations in parameters such as communication delay and the message loss to different sites in WAN. For example, the quorum consensus replication control algorithms should consider the dynamic performance of each of the links. Such site-to-site estimated performance data are stored in a matrix structure, called cost matrix or weighted adjunct matrix. However, the values are not pre-defined and fixed but time varying, and cannot be specified as a function of geographic location of the sites. In consequence, the algorithms (such as distributed query optimization) that use static cost matrix are no longer adequate. The communication system need to collect the performance data periodically to update these cost matrices.

Surveillance facilities have helped in early detection of site and link failures and repairs by exporting an up/down vector to the DTP algorithms [HZB92]. In the WAN environment, modeling the communication as an up/down vector is not sufficient. Early detection of changes in communication performance, such as latency and message loss rate must also be considered. This will improve the performance of adaptable DTP algorithms such as the quorum consensus replication control protocol.

The large variations also pose the scalability problem. Detailed cost matrix must be maintained to reflect the reality of the network dynamics. The cost matrix size grows as $O(n^2)$ and become inordinately large and unmanageable, for large number of sites $n$. We can tackle this problem by taking advantage of the clustering effect. The latency from one network to another network can be used to estimate the communication delay between two hosts in these two networks.

3 Experiments of Raid Transaction Processing in an Internetwork

In the past two years, we have conducted a series of experiments to study distributed transaction processing in the Internet environment. The ongoing experimental research in the Raid laboratory will pave the path for better understanding of the problems in migrating the DTP to the WAN environment and suggest more concrete directions for the proposals to tackle them. In this section, we present the preliminary result of our experimental study.

3.1 Apparatus

**Distributed Transaction Processing in Raid** Raid is a reliable and adaptable distributed database system specifically designed for conducting experiments in communications and distributed transaction management [BR89]. It provides an experimental infrastructure for our research [BFHR90, BZM91].

In Raid, the functions of DTP (such as concurrency control, replica control, and atomic commitment control) are implemented in separate servers. Each server is implemented as a process in a separate address space. Servers interact with each other through a high-level communication subsystem and can be arbitrarily distributed over the network to form a set of virtual database
The servers are the Access Manager (AM), the Concurrency Controller (CC), the Atomicity Controller (AC), the Replication Controller (RC), the Surveillance Controller (SC), the Action Driver (AD), and the User Interface (UI). The user interface is a front-end that allows a user to invoke SQL-type queries on a relational database. The action driver translates parsed queries into a sequence of low-level read and write actions. The access manager is responsible for the storage, indexing, and retrieval of information on a physical device. The concurrency controller checks that read and write actions of different transactions are serializable. The atomicity controller is responsible for ensuring that transactions are committed or aborted atomically across all sites. The replication controller manages multiple copies of data objects to provide system reliability and mutual consistency of replicated data. The surveillance controller collects connectivity information about Raid sites and advertises view changes to the replication controller. Details of the Raid architecture can be found in [BR89].

**Benchmark**  The benchmark we used is based on the DebitCredit benchmark, a simple yet realistic transaction processing benchmark that has been used in many experimental studies [Gra91]. This benchmark uses a small banking database, which consists of three relations: the teller relation, the branch relation, and the account relation. Each relation has 100 tuples. The tuples are 100 byte long and contain an integer key and a fixed-point dollar value. In addition, there is a sequential history relation, which records one tuple per transaction. Its tuples are 50 bytes long and contain a teller id, a branch id, an account id, and the relative dollar value specified in the transaction. A transaction updates one tuple from each of the three relations and appends a logging tuple to the sequential history relation.

We developed a multiple sites Raid distributed database for this benchmark. All the data items were replicated in all sites. The replication control protocol we used in our experiments was ROWA (Read One Write All), i.e. the database access was handled locally, and the database update had to be done on all sites. We generated the workload (a batch of transactions) according to the benchmark, and submitted it to the system. Each transaction was generated randomly with the following parameter: 20% of the data items were marked “hot spot”, 80% of all transactions would access some items in the “hot spot”; the average number of read/write actions per transaction was 4.0 with a variance of 2.0; the average update percentage (the number of writes actions per transaction) was 50% (and 10% in some experiments if so indicated). The length of the workload (number of transactions in the batch) was sometimes 250, and sometimes 20, depending on whether we could afford to prolong the experiments. The measurement parameters are response time (of a transaction), throughput (number of transaction per second), and abort rate (percentage of the transaction failed). None of them depends on the length of the workload.

Although we have done other experiments with databases of more than ten sites, we found that the performance was usually bound by the worst connection pair. Therefore we choose only two sites in our wide area network experiments. These should give us a good understanding on how wide area networking influences the transaction processing performance.

**Experimental environment**  Our research is conducted in the Raid laboratory with access to over ten Sun SparcStations running SunOS 4.1. The Raid Ethernet is a 10Mbps individual subnet of the Purdue CS departmental network. The connection from the Purdue-CS net to the Internet backbone was first through a NSC hyperchannel in Purdue campus, and then a T1 line to the NSFNET T3 backbone. All Raid machines have local disks and are also served by departmental
file servers.

We use the Internet as our wide area network testbed. Internet connects over two million computers in over 50 countries around the world. The United States part of the Internet is organized bottom up by linking many regional networks by high-speed NSFNET backbones. It makes a good representative interconnection network for our study.

WANCE tool  We have developed a Wide Area Network Communication Emulator (WANCE) tool to emulate Internet communication in a LAN environment, since we may not always have the resources to conduct experiments that run on actual sites in WAN and repeat these experiments with many different configurations. For example, changing a database site in an experiment may need to install all the Raid software at the new site.

We chose the emulation approach because it provides the real WAN communication in a LAN environment. The basic idea behind emulation is to force communication between two local hosts to go through the real Internet. To emulate a two-site system linking A and X over the WAN, we only need to find another computer B comparable to X but in the same LAN as A. We run the transaction managers in A and B instead of A and X (Figure 6). The WANCE tool can be specified to automatically route all transaction packets from A to B through X. Although our experiments are run in A and B (two SparcStations in our lab), we have the same effect of running them in A and X.

---

![Diagram of real experiment setup](image)

![Diagram of emulation experiment setup](image)

Figure 6: Configuration of emulation experiments

The justification for the emulation approach is based on the observation that the difference
between the behavior of a distributed system running on a LAN and that on a WAN is primarily due to the communication performance, not the location of the experimental host. The validation of this approach has been studied and experiments shows that it can achieve an accuracy of ±3%. Details of the emulation approach and the WANCE tool can be found in [ZB93].

3.2 Experiment I: Raid Transaction Processing in an Internetwork

Statement of the Problem  In section 2 we have shown that there is a large variation of the communication performance metrics on the Internet. Further, the time of the day has a strong influence on the effectiveness of the message delivery. Since the DTP software components rely on the underlying communication software, we can safely claim that the performance of transaction processing will demonstrate a similar behavior. These experiments strengthen our claims by measuring the response time, throughput and abort rate in distributed transaction processing between two sites on the Internet. We would like to know how the time of day influence the system performance and how the performance varies among different sites in the world. We hope to be able to predict the transaction processing performance through the experiments.

Procedure  The workload was a periodic batch of 20 transactions every 10 minutes over a 24 hours time span. We choose 10 minute interval between any two batch submissions because we believed that it would be small enough to capture the changes in the network conditions. We decided not to use smaller intervals to avoid extra messages on the Internet, which would not contribute to our studies. We choose 20 transactions as the number is large enough to generate meaningful average response time, throughput, and the abort pattern, and small enough for their execution time (plus experimental setup and bookkeeping times) to fit in the 10 minute interval.

We used the WANCE tool to conduct emulation experiments between Purdue and many other hosts across the world. We have experimented with the hosts in United States, and across the continent in Finland, Germany, India, Japan, Hong Kong, Thailand, Australia, Australia, Brazil, Zambia, etc. We are presenting the results of experiments between the hosts at Purdue, USA and Helsinki, Finland; and between the hosts at Purdue, USA and Hong Kong. The choice of these two remote sites can be explained as follows. Firstly, they are typical of the results for all the other sites we have investigated. And secondly, because these two sites cover much of the global span and many different time zones, they provide a better understanding of how network metrics behave at different hours of the day.

Data  We measured the response time, throughput, and transaction abort rate for each 20-transaction batch in these experiments. We report the case between Purdue and Helsinki, and between Purdue and Hong Kong for the reasons mentioned in the previous section.

The results are plotted in Figure 7 according to the time-of-day. We also average the data for each day and list the mean, variance, and the 95% confidence intervals for these averages in these figures.

Discussion  The data shows that the performance of DTP in WAN has large variants. Furthermore, it could drastically change at different times of the day. Transaction processing between United States (US) and Europe has better performance in night than in daytime. For example, the throughput was almost 2 transaction/second at night, but was often below 0.5 transaction/second
during the day time. This is consistent with our measurement of the Internet performance that shows the communication is slower and less reliable during the working hours. The transaction processing performance between US and Hong Kong is different. There are two reasons for this. First, the time difference between Hong Kong and Indiana, US is 13 hours. Hence when it is day in US, it is night in Hong Kong. Second, the US–Hong Kong Internet link is more reliable. We surmise that this is because much lesser traffic on this sector of the Internet. Unlike the US–Europe Internet communication, where many countries share a small number of cross-Atlantic links, the US–Hong Kong cross-Pacific link serves only Hong Kong, an area much smaller than New York city.

3.3 Experiment II: Performance of Basic Database Operations in WAN

Statement of the Problem A transaction consists of basic operations like updating a existing tuple, inserting a new tuple, and selecting a tuple. Understanding the performance of basic operations helps us to extrapolate the performance of a particular transaction given its access pattern. In this experiment, we measure the elapsed time for each of these constituent operations in a transaction.

Procedure We studied and analyzed the cost of executing three basic queries: a query selecting one tuple in a relation, a query updating one tuple of the relation, and a query inserting one tuple in the relation. We used the same two-site replicated DebitCredit database from the previous experiment. Each tuple of the relation is 100 byte long.

We configured the system to emulate a host in UTA (University of Texas in Arlington), a host in Hong Kong and a host in Finland. The same experiment was repeated in the LAN to get the data set for comparison. Measurements are repeated 100 times for each of the queries. (plus an extra first time to eliminate the cold-start effect). Query aborted due to the timeout because of
long delay or message loss in a WAN, would be restarted later.

<table>
<thead>
<tr>
<th>transaction</th>
<th>within LAN</th>
<th>Purdue-UTA</th>
<th>Purdue-Helsinki</th>
<th>Purdue-Hong Kong</th>
</tr>
</thead>
<tbody>
<tr>
<td>select one tuple</td>
<td>167</td>
<td>166</td>
<td>166</td>
<td>167</td>
</tr>
<tr>
<td>insert one tuple</td>
<td>286</td>
<td>352</td>
<td>754</td>
<td>1768</td>
</tr>
<tr>
<td>update one tuple</td>
<td>267</td>
<td>339</td>
<td>785</td>
<td>1786</td>
</tr>
</tbody>
</table>

Table 3: Response time (millisecond) for basic database queries

Data Table 3 shows the time in millisecond taken by the processing of the several basic database queries. We observe that the time to select one tuple is the same for different configurations. This could be attributed to the fact that all tuple reads can be completed using the local copies. Thus, the distance between two sites did not affect the performance of select queries. For insert and update queries, the distance played a significant role in their performance. The time for inserting or updating one tuple in the Purdue-UTA case only increases slightly over the LAN, because the communication delay between Purdue and UTA is around 35 milliseconds (averaged), causing the processing time to dominate. However, in Purdue-Helsinki and Purdue-Hong Kong experiments, the response time is notably increased. Communication delay varies between 200 and 400 milliseconds for the Purdue-Helsinki case, and between 700 and 2000 milliseconds for the Purdue-Hong Kong case. Thus the message latencies dominate the processing times for these queries.

Next, we studied the effect of varying the number of tuples in a query on its performance. We measured the response time of the queries for inserting or updating or selecting 10, 20, 30, or 40 tuples in a relation. The results are plotted in Figure 8. (Only the data from the LAN is plotted for the select queries because of their uniform performance in both LAN and WAN due to replicated data.)

Discussion We observed that the response time for each of the queries is an approximate linear function of the number of tuples in it. Both update and select queries are more sensitive to the number of tuples as they involve index read, tuple read, and locking concurrency control overheads. Insert does not need such overhead. However, the gap between response time in a WAN and that in the LAN does not increase with the the number of tuples. This implies that increasing the data size in a transaction will increase the response time in both LANs and WANs, but the decisive factor is the communication delay not the size of the message. This also suggests that it is possible to project the performance of transaction processing from LANs to WANs.

3.4 Experiment III: Multi-Programming Level and Concurrency Control

Statement of Problem The maximum number of concurrent transactions allowed in a system at any one time is called the multi-programming level (MPL) of the transaction processing. The MPL has a significant impact on selecting the concurrency control and commit protocols. The most commonly used concurrency control protocols include two-phase locking (2PL) and timestamp
ordering (TO). The widely employed commit protocols are two-phase commit (2PC) and three-phase commit (3PC). The experiment investigates the relationship between multi-programming level and the performance of transaction processing in WAN environment using these protocols. An effort is made to identify "good" MPL value for each of these different protocols and best combination of concurrency and commit protocol for DTP in WAN. The effects of changing the conflict rates in the transaction workload on the performance of these protocols is also investigated.

**Procedure** In this experiment, we have used Purdue-Texas configuration with a 250 Debit/Credit transactions workload with 10% average updates. We varied the MPL as 1, 2, 3, 4, 5, 10, 15, 20, and 25. (MPL=1 is the normal batch mode used in our previous experiments.) We measured the performance of four different protocol configurations: 2PL with 2PC, TO with 2PC, 2PL with 3PC, and TO with 3PC. For each value of MPL, we repeated the experiment 50 times and averaged the measured data.

To see how the update rate affects the results, we repeated the experiment using another workload of 250 Debit/Credit transactions with the 50% average updates.

**Data** Figure 9 shows the response time, throughput, and abort rate of the benchmark transactions using different protocols, as the MPL is varied. Two different average update rates (10% and 50%) are considered.

**Discussion** As the MPL increases, the response time increases monotonically, which is expected. The increase in MPL implies more conflicts and hence the blocking for transaction on the conflicting operations delays its completion.

The throughput increases initially for small MPL, but starts to dip for larger values of MPL. The increased throughput is due to the "non blocking" concurrent execution of the transactions.
As the MPL increases beyond a certain value, the number of conflicts and the message delays from the remote site prevents any useful work to be done at the local site, thus reducing the system utilization. Because of the presence of significant message delays in addition to the conflicts, the throughput can be maximized at a smaller MPL value than in LAN or centralized database environment. With lower average update rates (10%), this MPL value is in the range of 5 to 10. When the update rate is high (50%), the throughput maximizing MPL goes down to between 3 and 5.

The experiment suggests no significant impact of the MPL on the abort rate. This means that the unreliability of the Internet is the main contributor for the transaction aborts. The choice of concurrency control protocols strongly affect the throughput. We observed that TO produced higher throughput than 2PL, especially when the update rate is high. This suggests that we use timestamp ordering instead of the more widely used locking protocols, in the WAN environment.

2PC protocols also clearly out performed 3PC ones. Therefore we should not adopt 3PC, especially the combination of 3PC with 2PL, which caused extremely high abort rate. We will further analyze the commit protocols in the next experiments.

3.4.1 Experiment IV: Atomicity Control and Commitment

Statement of Problem  Atomicity control ensure the integrity of the distributed database. This experiment explores the cost of different commit protocols, 2PC versus 3PC, in an internetwork. When we move the operating environment to WANs, we expect extra time spent on the transaction commit phase. This experiment also identifies the bottle neck of the transaction processing.

Procedure  We continued using the two-site fully replicated DebitCredit database. The repeated the experiments using different commit protocols, the first time we used 2PC, and the second time
we used 3PC. We used a workload of 250 DebitCredit transactions and varied the distance between two database sites: in a local area network with Purdue, between Purdue and Arlington, Texas, between Purdue and Helsinki, and between Purdue and Hong Kong. We measured the total time for a transaction to stay in the system as well as the portion of time spent in the commit phase.

![Figure 10: Break down of transaction elapse time in the system](image)

**Data** Figure 10 details the total times and the portion spent in AC (Atomicity Controller), for both 2PC and 3PC and for four different database sites configuration. The percentage of time spent in AC are calculated and put on the top of each bar. At the time of each transaction processing experiment we also measured the communication delay between the two database sites. The delays are listed at the bottom of the figure, below the corresponding configurations.

**Discussion** We observe that when the communication delay increases, both the response time and the commit time increase. The time used in 3PC increases more shapely than that in 2PC. Although the contribution of 3PC on the transaction response time is not significantly larger than that of 2PC in a LAN, a system that employs 3PC will see a much bigger slowdown in a WAN. This suggests that 3PC is not a tolerable solution.

This experiment also indicates that the commit phase of transaction processing will become the bottleneck when we move our transaction processing software from LANs to WANs. It is caused by rounds of message exchanges to form a global consensus. Efforts to reduce the overhead can be directed into these directions: streamlined commit protocol, optimal replica placement, and perhaps relaxed consistency requirements, etc.
Table 4: the response time for select or update queries (in millisecond)

<table>
<thead>
<tr>
<th>Emulated Host</th>
<th>select queries</th>
<th>update queries</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>full replic</td>
<td>select replic</td>
</tr>
<tr>
<td>within LAN</td>
<td>233</td>
<td>233</td>
</tr>
<tr>
<td>within Purdue-Illinois</td>
<td>255</td>
<td>240</td>
</tr>
<tr>
<td>Purdue-Texas</td>
<td>251</td>
<td>247</td>
</tr>
<tr>
<td>Purdue-Helsinki</td>
<td>251</td>
<td>239</td>
</tr>
</tbody>
</table>

3.5 Experiment V: Replication of Complex Data in WAN

Statement of Problem To evaluate performance of both full replication and selective replication strategies, we conducted experiments using the Raid system with a complex object database. We used a “document” object-oriented database schema in this experiment. The database is distributed in a two-site system, queried by two independent users, one in each site (site1 and site2). A “document” object has four parts. Part1 and Part4 are accessed by both sites with equal probability. Part2 is accessed by site1 more frequently than by site2, with a probability of 3-to-1. Part3 is accessed by site2 more frequently than by site1, also with a probability of 3-to-1. A full replication scheme places copies of all four sections in both sites, and a selective replication scheme places sections 1 and 4 in both sites but Part2 in site1 only and Part3 in site2 only. A “no replication” configuration is also presented for comparison, which places all the parts in site1 solely.

Procedure We ran one LAN experiment and four WAN experiments using the WANCE tool. For WAN experiments, one local site emulated the following Internet sites respectively, one in each separate experiment: a site in Purdue campus but not in our department, a site in Champion, Illinois, a site in Arlington, Texas, and a site in Helsinki, Finland. A fixed size of 512 bytes for each read and write was used. The experiments with the same database configuration and input data were repeated for each of the emulated site. Details of the database schema and the experiment setups can be found in [ShJZB92].

Data and Discussion The total execution time for a query involving retrieval and update of the parts for both site1 and site2 are shown in Table 4. For select queries, since only the local version is read, full replication is almost as good as the local case (site1) of no replication. Selective replication has a slight overhead since some of the copies has to be fetched remotely. However, for the update queries, selective replication is much better than full replication, as it takes into account the access patterns of the users.

Going from LANs to WANs there will be an increase in communication overheads as well as an increase in the probability of some site or line failures. We can also see the response time increases in both update and non-local select queries from the tables. But the selective replication case has the slowest increase in average response time. The benefit of selective replication case is even more when the sites are further apart.
4 Transition of Distributed Transaction Processing to WAN

The communication experiments in the Internet have embarked the rethinking of our transaction processing software. We have identified some potential problems in the transition of distributed transaction processing software from LAN to WAN. We have also raised some important questions on large scale transaction processing, and designed some experiments to answer them. We believe that the following two directions for accommodating the DTP to WAN are important: improving communication facility and tuning/modifying/rewriting the transaction processing algorithms and protocols.

4.1 Improving Communication Facility

Communication is one of the most important component in DTP. The available communication system provided by the operating systems and computer networks are not tailored for the on-line transaction processing. To improve the communication support for DTP, the following issues need to be addressed.

Reliability vs Efficiency Many DTP systems use UDP/IP in their communication facilities. UDP is a connectionless datagram transport protocol and transmits data packets as independent entity. It is a “best-delivery” protocol but does not guarantee that every packet sent will reach its destination. Since the error rate in a LAN is less than 0.001% [AK91], most DTP systems use UDP/IP for message delivery in transaction processing, to meet the performance and real-time requirements. [Spe86, Svo86, BZM91]

Large volumes of network traffic may be generated by a transaction executing index queries or queries involving large data objects. This will many times lead to the overflow of receiver buffers (either at the destination, or at a gateway). Some packets may be lost and will need retransmission with UDP/IP. In the WAN environment, the reliability of message delivery is lower than that in LAN. In some portion of the network, or over some links, the packet loss rate may be so high that even the retransmissions meet a similar fate. Such problems are not seen on a local area network.

A brute force approach is to use TCP/IP (Transmission Control Protocol), another popular protocol for the Internet. TCP is a reliable stream-oriented protocol. It requires a connection setup to start the transmission and a connection termination to close the session. TCP ensures reliability by acknowledging the successful delivery of each packet and retransmitting the lost ones. Reliable connection-oriented protocols impose severe overheads and are inefficient for the LAN environment. As connectionless protocols such as UDP are suit for the LAN environment, a scheme for integrating both kinds may be the best for DTP in WAN.

Scalability It is important for a communication scheme for DTP to be “scalable” in WAN environments. That is, the performance of communication component should not degrade as the number of sites increases. Two different communication models are used for DTP: the virtual circuit model and the stateless message passing model. The virtual circuit model The communication between two processes is via a pre-established channel (e.g. a TCP connection) in a virtual circuit model. A connection is required for every pair of communicating processes. In the stateless message passing model, each message packet is self-addressed and each process has a “message port” (e.g. the UDP port in Unix socket interface).
The first model has the problem of scalability, since both ends of the connection maintain the state information and the buffer space for the un-acknowledged packets. Furthermore, as the number of sites in a DTP system increases, the number of connections a site has to support increases. Most of the operating systems place a limit on the maximum number of established connections for a process. For example, many versions of Unix allow a maximum of 32 open sockets per process. Hence, the communication scheme must adopt the message port model for scalability of DTP in WAN.

**Two-level communication** Our solution to the communication scheme for WAN is a “two-level” scheme. In the “local” level, the efficient message passing interfaces are preserved. These include the UDP socket interface and the SE port (Simple Ethernet protocol suite) which have been proven efficient in a LAN [BZM91]. In the “global” level, adequate transport service based on the condition of the links is employed.

We define the set of local sites to be a cluster. The connections in a cluster are reliable enough to use the “unreliable” datagram protocol. The links are “reliable enough” means that the packet loss rate on the link is so low that it affects the performance of DTP marginally. Occasional packet loss is remedied by timeout in a DTP server and rolling back the current transaction to a previous checkpoint.

In this scheme DTP servers can continue to use the datagram message passing interfaces to communicate with each other in the same cluster. To communicate with a site in a different cluster, instead of sending the message directly to the destination, the server deposits it with a communication process. This process is responsible for forwarding these messages using a more reliable protocol such as TCP. This two level protocol can be tuned to different networking environment.

**4.2 Transaction Processing Algorithms**

There have been a lot of performance studies on the transaction processing algorithms in local area networks. However, their performance in a wide area network environment can only be conjectured at this time.

**Concurrency Control** In transition from LAN to WAN, the communication latency has changed from several milliseconds to several hundred milliseconds. Given a two site database in a LAN, a typical Debit/Credit benchmark transaction takes a few hundred milliseconds to finish in a Sun SPARCstation. It will take a couple of seconds to complete in a WAN environment, depending upon the distance and the traffic on the network. A transaction stays longer in the system, increasing the data contention. The transaction is blocked for a longer periods of time, degrading the throughput of the system. Furthermore, since the locks are held for longer periods of time, the chances of transactions getting in deadlocks increase, leading to higher abort rates. We need to study the kind of concurrency control protocols that can adapt to the communication delay to give higher degree of concurrency.

**Commitment** 2PC is not resilient to failures while 3PC needs an extra round of messages. In [BFHR90] we have shown that in a LAN environment the throughput of a system using 3PC is not very different from that of 2PC. We suggested that 3PC should be utilized to increase availability with only a little cost. However, we have shown in Experiments III and IV that 3PC
degrades the throughput significantly due to the overheads of sending an extra round of message. This is in contrast to previous proposal of using 3PC for the LAN environment.

The commit phase will takes up a substantial portion of transaction time. It was shown in [SJR91] that the commitment accounts for one-third of the transaction duration in a general purpose database. Our experiments shows that the ratio increases to over 80% when we move into the wide area networks. It is important to optimize the commit protocol. Several possible optimizations that reduced the number of messages were shown in [SBCM93]. We need to conduct further experiments to study the feasibility and performance of these optimizations.

Replication Replication improves the availability and reliability of a distributed system. Full replication scheme results in improved availability for reads, but at the cost of more expensive updates. This is because of the requirement that all the replicas be mutually consistent. In the WAN environment, these updates are more expensive in terms of the number of messages exchanged, especially for large complex data such as in object-oriented databases. Selective replication scheme allocates and replicates the fragments of an object to various sites by considering the individual access behavior of the fragments [BBS92]. We believe that this scheme will out-perform full replication scheme for complex objects in the WAN environment.

Another problem is to understand the notion of consistency for replicated data in WAN environment, and whether it can be or should be relaxed. What are tradeoffs of taking such a decision.

If the quorum consensus based protocols are used in the WAN the time to form a global consensus on the data replicas will dominate because of communication latencies. Furthermore, the likelihood of a network partition is higher in a large internetwork. Thus, the replication control schemes based on strictly consistency criteria may fail. Some weaker consistency protocols have been studied in the literature. For example, epsilon-serializability permits inconsistency in database state for an epsilon time period. Anti-entropy replication algorithms, on the other hand, allow unlimited inconsistency for infinite long time but assures that the consistency will be maintained eventually.

There are many questions that remain unanswered. It remains to be explored that, given an access pattern, what is the optimal number of data replicas that will minimize the update overheads, maximizing the system availability. This may be a NP-Hard problem, in which case the need for studying the heuristics is self evident. Finally, in the weak consistency protocols, the overheads of merging inconsistent copies needs more attention to justify their applicability.

Availability Darrel Long et al has studied the availability of the Internet host and has found it to be below 86% [LCP91]. The low availability of the hosts has been attributed to many factors. The host is down for the reasons of periodic maintenance, software/hardware crashes, or a high local work load. The message losses in the communication can make it unavailable to a remote site. Thus, the host may be up and functional, but the network congestion causes the in-route gateways to drop the message packets. In either situation, the low availability of the hosts will degrade the performance of DTP.

One solution to improve the availability is to pair each database site with a backup site. This is similar to notion of disk mirroring for improved availability. The two sites are placed in different geographic locations and use different links to the internetwork backbone. However, the increased availability comes with the overheads of bringing the backup site to a consistent state, when the primary site is unavailable. The probability of both sites being unavailable simultaneously is reduced.
A detailed analysis of the availability of a replicated database can be found in [BHIF92].

4.3 Relevant Work and Related Results

Our measurements is related to some other work in experiencing the performance of WANs.

Communication in an internetwork Long et al [LCP91] used the Internet to estimate parameters such as mean-time-to-failure (MTTF) and availability of the Internet sites, and then used the results to derive an estimate of mean-time-to-repair (MTTR). Statistics data about mean-time-to-failure and mean-time-to-repaired were reported according to the computer models. His conclusion was that most of the hosts in Internet are available during 90% of the time. Golding and Long [GL91] also assessed the reliability of message delivery in the Internet. They observed that retrying messages for two or three times can significantly improve the performance. Discrete-event simulations were built based on this study. Other studies that analyze the Internet performance and model it can be found in the literature [Hei90, Pax91, Pax93, AS92].

Transaction processing in WANs Pu et al [PKL91] measured the performance such as response-time and failure rate of Camelot transaction service and Webster dictionary lookup over the wide area networks. The measurement method is called "Layered Refinement", the simultaneous data collection at some important software layers. He concluded that the performance of the wide area internetwork is dynamic and the performance of application are significantly affected by the the performance of the network. To solve the performance problem of running wide area distributed systems across the Internet, Golding developed a weak-consistency replication architecture [Go192a]. Temporal inconsistent is allowed in the system while an anti-entropy protocol will eventually bring the all copies updated [Go192b].

Large scale system Andrew/Coda are the only distributed file systems developed so far that can be classified as very large scale systems [Sat89]. They are developed by CMU with planned configuration of thousands of workstations over MAN or WAN. They show that several ideas fulfill the scalability, and requirement of shared file access over the wide area network: volume replication, client caching, callback, classifying files by access and modification patterns, and disconnected operations.

5 Conclusion and Future Work

We have studied the end-to-end communication performance in the Internet and its impact on distributed transaction processing. A large variations in communication delay, message loss rates suggest that the parameters for transaction processing algorithms cannot be determined statically. The value for the communication performance metrics are different for different sites, not exhibiting any correlation with their geographic location, and are time varying. We have made an attempt to analyze the impact of communication on DTP and suggested various techniques to improve our communication facilities.

We have presented the results of our experimental study of DTP on the Internet. We have used the industry standard DeditCredt benchmark. Similar to the communication performance, transaction processing performance metrics in the Internet have large variants. We have observed
that although different transaction control protocols have a strong influence on system throughput, their impact on response time and abort rates can not be isolated easily in a WAN environment. We observed that to maximize throughput along with a reasonable response time, a multi-programming level between 3 and 5 should be utilized. In moving the DTP from LAN to WAN, the higher communication delays makes atomicity control the bottleneck of the system. Furthermore, 3PC requires extra round of messages and should be discarded as an option for DTP in WAN. Our experimental study confirms that selective replication of complex data objects, given the access behaviors of the sites, is much better in WAN environment than full object replication.

The experiments has commenced our rethinking the DTP software, to adapt to uncontrollable and dynamically changing WAN environment. We have identified potential problems in the transition of distributed transaction processing software from LAN to WAN. We have pointed out that better communication facility and finely tuned transaction processing algorithms will be two important directions for future research. We continue our experimental studies to further our understanding of DTP in WAN.

Since the WANs are in a state of a constant flux, a DTP system should be reconfigurable according to the changing network characteristics. We plan to study the implication involved in such re-configuration in our future research. We will study the criteria of the reconfiguration, including the communication latency, bandwidth, etc. To make the dynamic reconfiguration possible, we must search for the mechanism to effectively monitor and report the performance of WANs. We believe surveillance is a good model to achieve this goal. We are currently developing the surveillance protocols and studying their contribution to the DTP.

References


27


