10/11/12/96 8 SC 1

SANDIA REPORT

SAND96-8249 • UC–405
Unlimited Release
Printed August 1996

ATM-Based Cluster Computing for Multi-Problems Domains

H. Y. Chen, J. M. Brandt, R. C. Armstrong

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94551
for the United States Department of Energy
under Contract DE-AC04-94AL85000

SAND96-8249
Unlimited Release
Printed August 1996

UC-405

ATM-Based Cluster Computing for Multi-Problem Domains

H. Y. Chen and J. M. Brandt
Infrastructure and Networking Research Department
and
R. C. Armstrong
Distributed Systems Research Department
Sandia National Laboratories

Abstract

This study evaluates the performance of an Asynchronous Transfer Mode (ATM) local area network (LAN) for general as well as parallel distributed computing. General distributed computing uses client-server based applications that employ Remote Procedure Call (RPC) on top of the TCP/UDP/IP protocol. These applications typically require high throughput, good response time, and fairness. In contrast, parallel applications favor much simpler models of computation which require more direct access to data among processors. To efficiently run these programs, the network needs to achieve hardware speed. This paper describe our experience in building a multi-programmed distributed computing environment using Digital Equipment Corporation's (DEC) AN2 ATM switch and Alpha workstations. We extend our study to include more elaborate network using simulation results.

Acknowledgments

The authors gratefully acknowledge efforts of Chien, Fang, Rich Palmer, and Christine Yang for their technical input and helpful comments. We also extend our thanks to Jim Hutchins for providing the UDP-echo utility. Chandu Thekkath and Hal Murray from Digital Equipment Corporation were extremely helpful in providing special software and technical support in setting up our research PVM environment.

Contents

Acronyms and Abbreviations

AAL5	ATM Adaptation Layer 5
API	Application Programming Interface
ATM	Asynchronous Transfer Mode
BSD	Berkeley Software Distribution
DMA	Direct Memory Access
HOL	Head Of Line blocking
IP	Internet Protocol
LAN	Local Area Network
QoS	Quality of Service
PIM	Parallel Iterative Matching
PVM	Parallel Virtual Machines
RPC	Remote Procedure Call
RTT	Round Trip Time
SAR	ATM Segmentation and Reassembly
SONET	Synchronous Optical NETwork
TCP	Transmission Control Protocol
UDP	Unreliable Datagram Protocol
VC	Virtual Connection

1. Introduction

This study evaluates a high-speed local area network that can support multiple problem domains in distributing computing. Specifically, we evaluate commodity ATM [1] network components (e.g. switches, adapters, etc.) that will satisfy the performance requirements of RPC-based [2] as well PVM-based [3] applications.

Asynchronous Transfer Mode (ATM) is the latest network technology that can offer scalability in network size and speed. Moreover, using fast cell-switching in hardware, it can control delay jitters, thereby providing integrated services with different quality of service (QoS) requirements. Parallel to the advances made in networking technology, there have been significant improvements in processor technology; current high-end workstations have comparable processing power to those in dedicated, tightly-coupled supercomputers. Therefore, it is becoming increasingly desirable to use a cluster of workstations on a high-speed network as a loosely-coupled supercomputer.

The goal of this study is to build a high performance LAN that can serve both RPC-based as well as parallel applications. While RPC-based application perform well using this cluster environment, it would also allow parallel applications to use its idle cycles as a loosely coupled supercomputer. We seek to satisfy the performance requirements for both applications in terms of throughput, response time, and fairness. In addition, for parallel applications, optimization techniques are employed to achieve hardware speed in both network and processor.

The organization of this paper is as follows. Section 2 describes components of this cluster computing environment. Section 3 presents results and analysis for generic as well as parallel distributed computing. Section 4 summaries and describes future work.

2. The cluster computing environment

The major components of our cluster environment include: 1) AN2's fair queuing [4], 2) credit-based, ATM-level flow control [5], and 3) workstation kernel memory mapping [6] on 170 Mhz DEC Alpha workstations.

2.1. AN2 Fair Queuing

The DEC AN2 is a 16 x 16, input-buffered, cross-bar switch. It is internally non-blocking because there is at least one path between all non-conflicting input and output pairs. In order to avoid head-of-line (HOL) blocking [7] and achieve *fair queuing*, this switch uses per-VC buffer management and an parallel-iterative-matching (PIM) scheduling algorithm [4]. The fairness achieved is at an ATM cell granularity. The following three steps are iterated several times within one OC-3c time slot.

1. Each unmatched input port sends a request to all output ports for which it has cells to send.
2. Grants are issued, at random, by unmatched output ports to requesting input ports.
3. If the input port accepts the grant, it notifies the output port, which disqualifies the output port from participating in the next iteration.

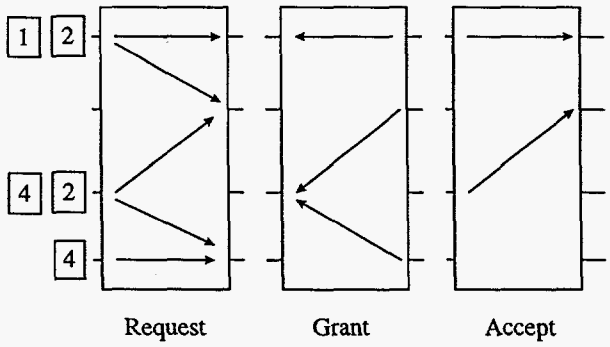


Figure 1. One iteration of the parallel-matching process.

Figure 1 illustrates the three steps of one iteration in a 4 x 4 switch. In the first iteration, five requests are made, three are granted, and two accepted. The one unmatched input-

output pair, from input port-4 to output port-4, will be resolved during the second iteration. After the second iteration, no more pairing is necessary. These steps are executed independently and in parallel at each input and output port in order to quickly match conflict-free pairs of inputs to outputs. The AN2, with its 800-Mbps switching speed, can support four iterations of the parallel-matching process per OC-3c time slot. A previous study [4] demonstrated that within four iterations, this algorithm can achieve 99.9% efficiency in a 16 x 16 cross-bar switch.

2.2. ATM level flow control

AN2's *per hop, per VC, credit-based flow control* promises a zero-loss ATM network. As depicted by Figure 2, a sender is initialized with a credit balance which is equal to the buffer space at the receiver. Credit balance of a VC is decremented after a cell is sent. When the receiver has successfully forwarded a cell, a credit is returned to the sender and its credit balance is incremented. Credits are piggybacked onto either a data cell or a null cell traveling in the opposite direction. The buffer requirement for the per hop, per VC, credit-based flow control is equal to the product of the targeted VC bandwidth and the per-hop round trip time (RTT). The targeted VC bandwidth is chosen to be less than or equal to the link bandwidth. This hop-by-hop flow control mandates per-virtual-circuit (VC) buffer management. The data- and credit- path of one VC is illustrated in Figure 3.

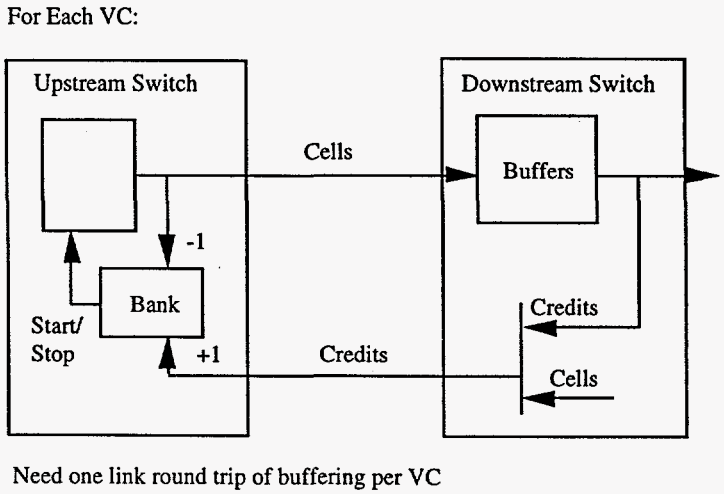


Figure 2. DEC's AN2 Credit-based Flow Control Algorithm.

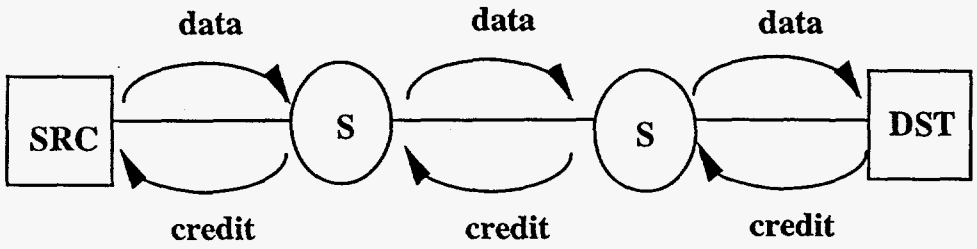


Figure 3. Credit-based flow-control scheme.

2.3. Kernel memory mapping

Previous studies demonstrated that, over generic networks, only the very coarse-grained parallel applications can achieve reasonable speed-up. While commodity network components, such as ATM switches, TCP/IP drivers, ATM adapters, and its device driver, perform well for RPC applications, the protocol processing, multiple data copies, and operating system interrupt overheads severely penalize the communication aspects of parallel computing relative to the computation components. In order to achieve an acceptable cost/performance tradeoff for a wider spectrum of parallel applications, we strive to achieve hardware speed on network links as well as in workstation processors.

As mentioned earlier, we choose the parallel virtual machine (PVM) run time system for our parallel applications. The PVM software package allows a network of computers to appear as a single concurrent computational resource. We can reduce end-to-end latency by eliminating TCP/IP, since their function already exists in a flow-controlled ATM network.. DEC's System Research Center at Palo Alto has provided us with a modified set of PVM library routines [18], which send and receive messages directly over the flow-controlled ATM interface, bypassing TCP/IP. In addition, a modified driver for DEC's ATM host adapter (OTTO) allows PVM library routines to access kernel memory, thereby reducing the number of data copies necessary for a given message transfer.

The DEC ATM adapter, OTTO, performs ATM segmentation and reassembly (SAR) of data packets in host memory. The device driver will properly encapsulate and checksum data packets pointed to by a memory address; the adapter then transfers fragments (48 bytes) of the packets using DMA from host memory to the network. Similarly, the payload of ATM cells from the network is DMA'ed directly into host memory to be reassembled. This design eliminates the store-and forward latency normally experienced by adapters that carry the ATM SAR in the adapter memory. To further improve end-to-end latency, the device driver provides the modified PVM library routines with the ability to map its kernel memory using the *ioctl* system call interfaces. Figure 4 depicts the buffer layout of the kernel mapping process.

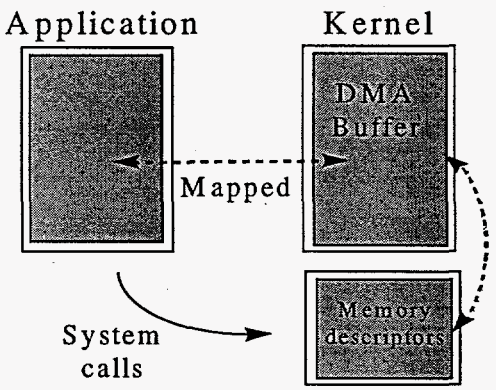
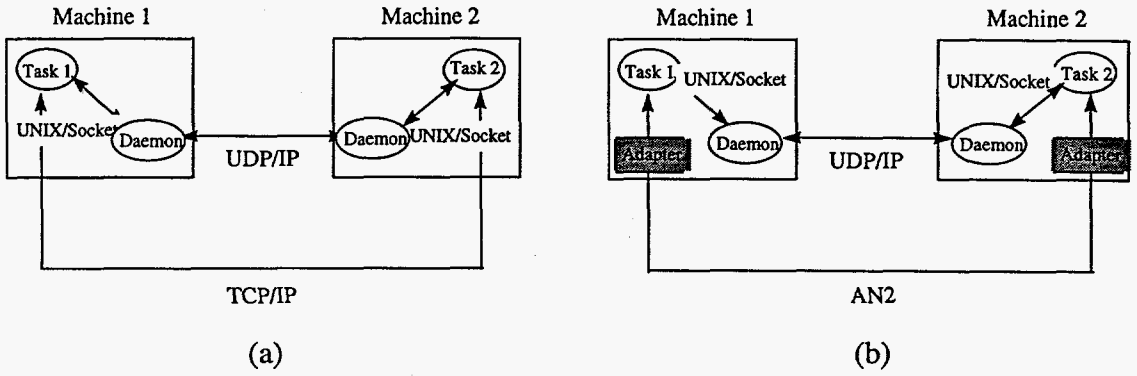


Figure 4. Buffer layout for the kernel memory mapping process.

The modifications to the PVM library replace the socket system calls that interface to TCP with the *ioctl* system calls that provide direct mapping to the adapter's kernel memory. The original PVM application programmer interface (API) and the PVM daemon are preserved, thereby allowing existing PVM programs to access our environment transparently. The communication path for the traditional and the AN2 cluster environment are contrasted in Figure 5.



(a) (b)

Figure 5. Communication path for : a) traditional PVM and b) ATM PVM.

In addition, the modified PVM implementation pre allocates free lists of all common data structures used by PVM routines, reducing the overhead of run-time memory allocation system calls.

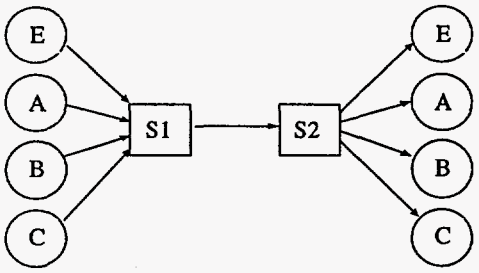
3. Results and analysis

3.1. Throughput and response time

A two-node fan-in/fan-out configuration (Figure 6) is used to evaluate the RPC response-time performance. Using this configuration, we interconnected one UDP-based echo session and from one to three concurrent TCP-based TTCP sessions. We measured the Round-Trip-Times (RTT) of UDP packets of 64, 128, 256, 512, 1024, and 2048 bytes with from one to three background TTCP streams. With fair queuing in place, the theoretical delay for a UDP echo packet is defined as follows:

RTT = (end-system delay) + (switching delay) + (propagation delay) + (n∗m∗cell-transmission delay) (1)

where n is the number of active sessions, and m is the number of cells per packet. This definition implies that active sessions are serviced fairly within the granularity of an ATM cell-time. We normalize our measured RTT to its theoretical upper-bound in order to put it in proper perspective, and we will refer to the normalized value as its relative RTT degradation.



All link distances = 4 Km and rates = OC3c
E is UDP echo session and A, B, & C are TTCP sessions

Figure 6. Two-node fan-in / fan-out configuration.

Results of the response time study are summarized in Figures 7 and 8. Plots of the echo RTTs against UDP packet sizes are shown in Figure 7 representing RTT values both in the absence of congestion and with incremental degrees of congestion. Figure 8 plots the relative performance degradation for the tests with two and three background TTCP sessions to show the effects of congestion. We consider relative degradation values near one to be close to the optimal RTT performance as defined in Equation 3. As shown in both figures, AN2's per-VC queuing mechanisms together with its fair scheduling algorithm provided the VC isolation and fairness necessary to ensure optimal response time even in the presence of multiple competing bulk data transfers. Within each test scenario (1, 2, and 3 concurrent TTCP sessions), RTT increases with increasing packet size due to additional transmission delay. Figure 8 further demonstrates that because the relative degradation is around 1, the increases in RTT with the same packet size is largely due to the added scheduling delay in the AN2 due to increasing competitions for the bottle neck link.

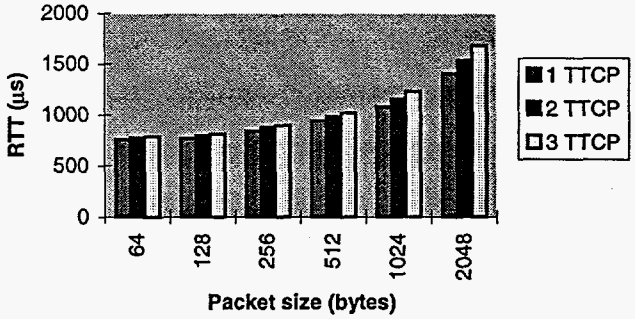


Figure 7. UDP echo RTTs vs. Packet sizes with one, two, and three TTCP's.

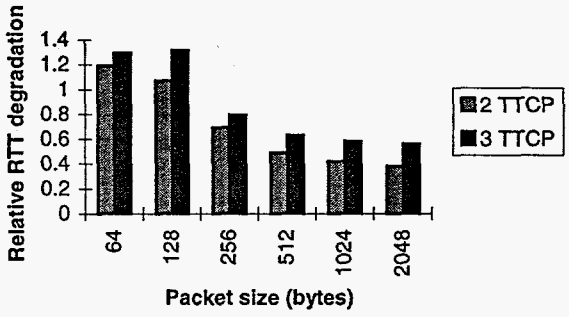
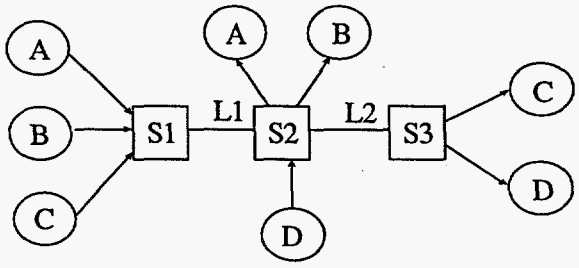


Figure 8. Relative echo RTT degradation vs. UDP packet size with two and three background TTCP sessions.

Figure 9 depicts the three-node parking-lot configuration, interconnecting four concurrent TTCP sessions, that we used to study the performance of throughput based on MAX-MIN [8] fairness, and link efficiency.



All link distances = 4 Km and rates = OC3c

Figure 9. Three-node parking-lot configuration.

According to the MAX-MIN fairness definition, a session's maximum achievable bandwidth is determined by its minimum bandwidth along its entire route, regardless of network topology or geographic distance. For our configuration, the MAX-MIN fair share for sessions A, B, and C is 1/3 of the bandwidth at the first inter-switch link (L1). Since session C is bottle-necked at the upstream link (L1), session D should absorb the remaining bandwidth of the second inter-switch link (L2). At OC-3c rate, the available link bandwidth, after SONET [9] overhead, is 150.336 Mbps. The 5-byte ATM header, 8-byte AAL5 [1] encapsulation, and 40-byte TCP/IP [10] header further reduce the available bandwidth to 135.44 Mbps (Equation 2),

$$BW = 150.336 * ((53 - 5) / 53) * ((9188 - 8 - 40) / 9188) = 135.44 \text{ Mbps} \quad (2)$$

where 9188 is the maximum transfer unit (MTU) defined by *classical IP over ATM* [11].

Accordingly, the theoretical fair shares for our configuration are:

$$A = B = C = 135.44 / 3 = 45.15 \text{ Mbps} \quad (3)$$

$$D = 135.44 - 45.15 = 90.29 \text{ Mbps} \quad (4)$$

In addition, we define link efficiency as follows:

$$\text{Link efficiency} = (\sum T_i) / \text{link-bandwidth} \quad (5)$$

where T_i is the throughput achieved by the i^{th} TTCP session.

Table 1 summarizes the results for our throughput evaluation using the three-node parking-lot configuration.

Table 1. Parking-lot TCP throughput and link efficiency.

	Session A	Session B	Session C	Session D
TCP Throughput (Mbps)	43.0	43.2	43.0	85.2
% Link efficiency (L1)*				95.4
% Link efficiency (L2)*				94.6

* %link-1 efficiency = 100*(T1 + T2 + T3)/135.4
%link-2 efficiency = 100 * (T3+ T4)/135

We calculate the percent deviation of experimental TCP throughputs from the MAX-MIN fair shares using the following formula and found that all sessions achieved fairness in our simple configuration.

$$\%\Delta = ((\text{Measured TCP throughput} / (\text{MAX-MIN fair share} * \text{link efficiency})) - 1) * 100 \quad (6)$$

3.2. Efficiency of AN2 credit-based flow

This section describes our congestion experiment with and without the AN2 credit flow control. The performance results are compared in order to demonstrate the efficiency of the AN2 credit flow control in terms of network congestion control. Because we do not have enough DEC Alpha workstations in our cluster environment, we were unable to generate enough traffic to cause switch buffer overflow, in the absence of AN2 flow control. Therefore, we verified the efficiency of AN2's credit-based flow control using simulation results. Our simulation model consists of the AN2 ATM switch and the DEC's ATM host adapter. These models are extensions of an existing simulation package from MIT, NetSim [12], which consists of other simulation components such as BSD 4.3 TCP, continuous and Poisson traffic, etc. Traffic on our simulated network is generated by applications that are based on TCP. These applications are greedy for bandwidth and will send data continuously into the simulated network, introducing congestion at the inter-switch link depicted in Figure 10.

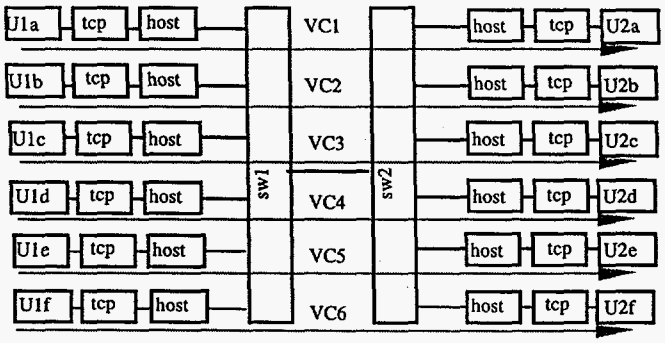


Figure 10. Simulated ATM network configuration of six concurrent TCP sessions sending data continuously

Using the simulation parameters listed in Table 2, one TCP session can achieve 110 Mbps. Therefore, this topology represents an offered network load which is roughly six-times the link capacity. We performed our simulation with and without the credit-based flow control and the results are plotted in Figure 11.

Table 2. Run-time Simulation Parameters

Simulation Parameters		Values
TCP	Window	64 KByte
	Maximum segment size (MSS)	8192 Bytes
	Protocol processing and OS overhead	300 μs
ATM adapter	Segmentation and reassembly delay	200 ns
ATM Switch	Switching latency	12 μs
	Per VC queue allocation (without flow control)	256 cells
	Per VC credit allocation (with flow control)	5 cells
Link	Transmission delay at OC3c speed	2.8 μs
	100 m propagation Delay	500 ns

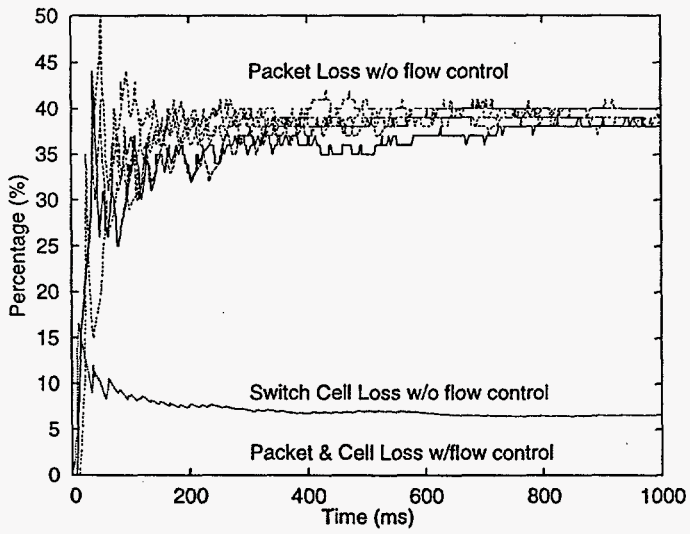


Figure 11 Percentage Cell loss and TCP Retransmission with AN2 Flow Control and without Flow Control

As shown, the credit-based flow control indeed provided a zero loss ATM network. Furthermore, we show that cell loss is very detrimental to TCP performance; a ~6% cell

loss caused ~37% TCP packet loss in this scenario. As a result, a significant amount of bandwidth is wasted in transporting cells belonging to the already corrupted packets as well as in their re-transmission.

3.3. PVM performance

Using pvm_send and pvm_recv routines, a ping-pong program was written to measure message round trip times. We conducted tests, with message size of 64, 128, 256, 512, 1024, 2048, 4096, and 8192 bytes, using the traditional as well as this ATM-based PVM system. Results of these measurements are plotted in Figure 9 and demonstrated that our implementation can offer significant performance gains, especially when the PVM messages are small.

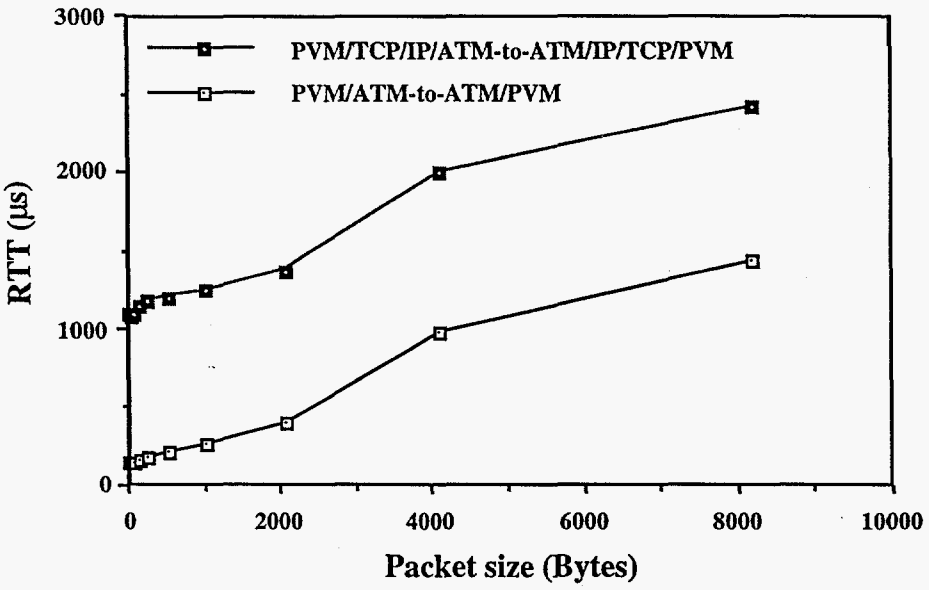


Figure 9. Round-trip latencies of PVM applications using different transport mechanisms.

A scaleable parallel molecular dynamics code has been compiled to run on eight DEC Alphas interconnected *via* the DEC AN2 ATM switch without source code modification. This model uses a mesoscopic approach between the strict continuum and atomistic levels. It is useful for quantifying processes that involve the manufacture of state-of-the-art microchips, where neither atomistic nor continuum approaches are applicable.

Models of this sort require large (~1 megabyte) messages and can profit from the switched point-to-point network that ATM provides. Using this cluster environment, the speedup in calculation is achieved by a special PVM implementation that eliminates processing overhead. This, together with ATM's point-to-point switching technology, has more than doubled the efficiency of communication. However, since this application is coarse grained, there is no significant performance gain. We will continue to evaluate the performance characteristics for finer grained parallel models.

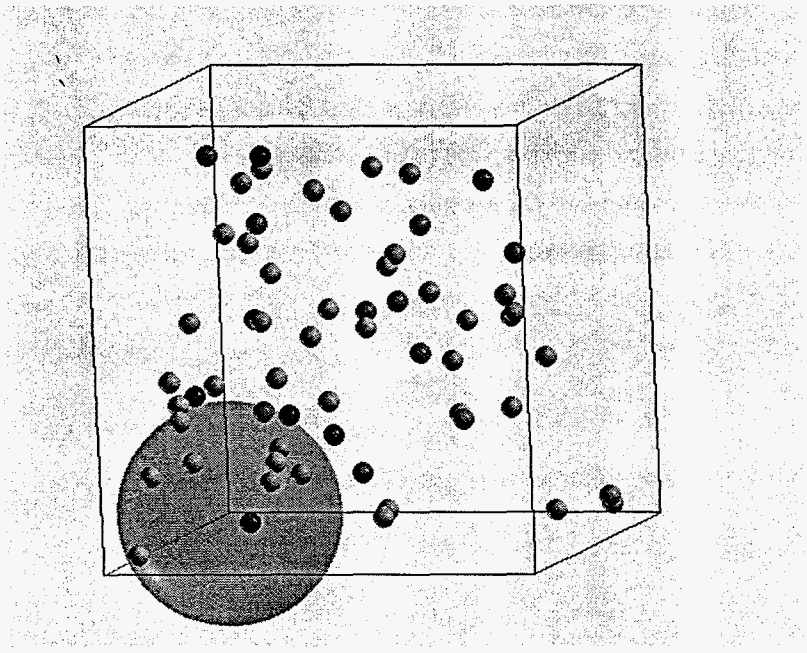


Figure 10. Molecular Dynamics over ATM.

4. Conclusions and Future Work

Our study has shown that ATM switches with Per-VC buffer management and fair scheduling can provide optimal throughput, response time and fairness for distributing computing. Because packets are fragmented for ATM adaptation, cell-loss can be extremely damaging to TCP performance. Therefore, it is important that the switch has a functional flow-control mechanism. Our study demonstrated that the AN2 credit-based flow control can indeed provide a loss-less environment.

While we have demonstrated in our tests that TCP works well with DEC's credit flow control, the processing overhead of the TCP/IP protocol stack introduces delays that are

undesirable for distributed applications. We have reduced end-to-end latency by eliminating TCP/IP, since their function already exists in a flow-controlled ATM network. In addition, our modified OTTO driver allows PVM library routines to directly access kernel memory, thereby reducing the number of data copies necessary for a given message transfer. We compared the round-trip latencies measured both with and without the modifications using a simple ping-pong PVM application. Results of these measurements demonstrated that our implementation can offer significant performance gains, especially when the PVM messages are small.

This approach opens up a broad range of possibilities in the area of parallel distributed computing, where the low cost/performance ratio could make this method competitive with today's massively parallel computers in many situations. Using this cost effective cluster-computing environment, we will continue to evaluate the performance characteristics for a wide range of parallel models. In particular, we would like to apply these same optimizations to a cluster of Pentium based personal computers with Free BSD Unix operating systems and DEC's ATM PCI adapter cards.

References

[1] D. E. Comer, *Internetworking with TCP/IP: Client-Server Programming and Applications,* pp. 240-255, Prentice-Hall, 1993.

[2] V. S. Sunderam, *PVM: A frame work for parallel distributed computing,* Concurrency: Practice and Experience, 2(4), pp. 315-339, December 1990.

[3] M. de Prycker, *Asynchronous Transfer Mode - Solution for Broadband ISDN,* Ellis Horwood Limited, pp. 97-124, 1992.

[4] T. E. Anderson, S. S. Owicki, J. B. Saxe, C. P. Thacker, *High Speed Switch Scheduling for Local Area Network,* DEC Internal Publication, 1993.

[5] J. Scott and B. Simcoe, *Digital Flow Control,* ATM Forum Contribution 93-778 (July 1993).

[6] C. A. Thekkath, *Cluster Computing on the AN2 ATM Network,* DEC Internal Publication, July 1995.

[7] R. Onvural, *Asynchronous Transfer Mode Networks: Performance Issues,* Artech House, Inc., pp.208-209 (1993).

[8] B. Simcoe, *Generic Fairness Configuration,* ATM Forum Contribution 94-0557, May 1994.

[9] Clapp, G., and Zeug, M., *Components of OSI: Asynchronous Transfer Mode (ATM) and ATM Adaptation Layers,* Connexions: The Interoperability Report, Vol.6, No. 4, pp.22-28 (April 1992).

[10] D. E. Comer, *Internetworking with TCP/IP: Principals , Protocols,, and Architecture,* pp. 129-150, Prentice-Hall, 1988.

[11] M. Laubach, *Classical IP and ARP over ATM,* RFC 1577, January 1994.

[12] Martin, D., *Network Simulator User's Manual* (September 22, 1988).

UNLIMITED RELEASE

INITIAL DISTRIBUTION:

9003 D. L. Crawford, 8900
9011 R. E. Palmer, 8901
9011 P. W. Dean, 8910
9011 H. Y. Chen (10)
9012 J. E. Costa, 8920
9012 S. C. Gray, 8930
9019 B. A. Maxwell, 8940
9011 D. B. Hall, 8970
9001 T. O. Hunter, 8000
Attn: J. B. Wright, 2200
E. E. Ives, 5200
M. E. John, 8100
L. A. West, 8200
W. J. McLean, 8300
R. C. Wayne, 8400
P. N. Smith, 8500
T. M. Dyer, 8700
P. E. Brewer, 8800
1436 LDRD Office
9021 Technical Communications Department, 8815, for OSTI (10)
9021 Technical Communications Department, 8815/Technical Library, MS 0899, 13414
0899 Technical Library, 13414 (4)
9018 Central Technical Files, 8940 (3)