AAI Performance and Congestion Management Studies: Year 1 Technical Report

Victor S. Frost
Joseph B. Evans
Douglas Niehaus
David W. Petr
Luiz A.P. DaSilva
Georgios Y. Lazarou
Hongbo Zhu
Roel J.T. Jonkman

TISL Technical Report TISL-10980-11

January 1996

Project Sponsor:
Sprint Corporation
Under prime contract to
Defense Advanced research Projects Agency
Contract number DABT 63-94-C-0068

TISL

Telecommunications and Information Sciences Laboratory
The University of Kansas Center for Research, Inc.
2291 Irving Hill Road Lawrence, Kansas 66045

AAI Performance and Congestion Management Studies:
Year 1 Technical Report

Victor S. Frost,
Director, Telecommunications and Information Sciences Laboratory,
and Professor, Electrical Engineering and Computer Science
Joseph B. Evans, Associate Professor, EECS
Douglas Niehaus, Assistant Professor, EECS
David W. Petr, Associate Professor, EECS
Luiz A.P. DaSilva, Graduate Research Assistant
Roel J.T. Jonkman, Graduate Research Assistant
Georgios Y. Lazarou, Graduate Research Assistant
Hongbo Zhu, Graduate Research Assistant

Telecommunications and Information Sciences Laboratory
Electrical Engineering and Computer Science Department
University of Kansas
2291 Irving Hill Road
Lawrence, Kansas 66045
Phone: (913) 864-4833
Fax: (913) 864-7789
E-mail: frost@tisl.ukans.edu

January 1996

TISL Technical Report TISL-10980-11

Sponsored by:
Sprint Corp.
under prime contract to
Advanced Research Projects Agency

Contract DABT63-94-C-0068

Table of Contents

List of Figures iii
List of Tables iv
Abstract v
1 Introduction 1
2 Year 1 Accomplishments 2
2.1 AAI Network Performance and Troubleshooting 2
2.2 Network Measurement Tools 4
2.2.1 Developed Network-Wide Performance Measurement Tool: NetSpec 4
2.2.2 Developed ATM Flow Monitoring and Statistics Archiving Tools 6
2.2.2.1 Point Measurements 6
2.2.2.2 Automatic ATM Flow Monitoring, Statistics Archiving, and Traffic Visualization Tools 9
2.3 AAI Network Measurements 9
2.3.1 Developed a Set of AAI Performance Testing Scenarios 9
2.3.2 Measured Performance of Multiple AAI Sites Using NetSpec 12
2.3.2.1 Scenario 1 13
2.3.2.2 Scenario 2 15
2.3.2.3 Scenario 3 16
2.3.3 ATM WAN File Transfer Performance Experiment 18
2.3.4 EDC Traffic Measurements 22
2.3.5 SC'95 Measurements 25
2.4 AAI Simulation 29
2.4.1 Developed Simulation Models for Components of the AAI 30
2.4.1.1 TCP 30
2.4.1.2 Cell-Level Pacing 30
2.4.1.3 ATM Layer Model 30
2.4.1.4 Basic ATM Switch Model 31
2.4.1.5 SONET Link Model 31
2.4.1.6 Traffic Model 31
2.4.2 Validation of AAI Simulation Models 31
2.4.2.1 System Parameters 31
2.4.2.2 Predicting the Effect of Rate Mismatches 32
2.4.2.3 Predicting the Performance of the AAI 33
2.4.3 Lessons Learned 34
2.5 Congestion Control 35
2.5.1 Initial ATM WAN Flow-Control Experiment 35
2.5.2 Initial Framework for the Implementation of End-to-End Rate-Based Congestion Control Within AAI 37
3 Major Technical Challenges 38
3.1 Network Performance Measurements and Troubleshooting 38
3.1.1 Dealing With a Large Heterogeneous Network 38
3.1.2 Monitoring Tools Lacking 38
3.1.3 Standard Software Was Not Designed for High Bandwidth-Delay Product Networks 38

3.2 Network Measurement Tools ... 38
3.2.1 Network benchmarking ... 38
3.2.2 Multilayer measurements require changes to operating system kernels ... 38
3.2.3 Need to support multiple operating systems ... 39
3.3 AAI Simulation ... 39
3.4 Congestion Control ... 39
4 Lessons Learned ... 40
4.1 Network Performance Measurements and Troubleshooting ... 40
4.2 TCP/IP over ATM ... 40
4.3 Network Measurement Tools ... 41
4.4 AAI Network Measurements ... 41
4.5 AAI Simulation ... 41
4.6 Congestion Control ... 41
5 Ongoing Problems ... 43
5.1 Connectivity ... 43
5.2 Access to remote hosts ... 43
5.3 Multiple organizations ... 43
5.4 Access to driver code ... 43
5.5 Lack of OC-3c-level performance from SPARC-20-class machines ... 43
6 Remaining Issues for Year 2 ... 44
6.1 Analyze the SC'95 data ... 44
6.2 Complete next version of NetSpec ... 44
6.3 Analyze the EDC data ... 44
6.4 Complete AAI-wide measurements using a combination of ARL MBONE machines and AAI OC-3-capable hosts ... 44
6.5 Analyze the continuous flow of switch statistics ... 44
7 Year 2 Research Agenda ... 45
7.1 Network Performance Evaluation ... 45
7.2 Data Analysis and Development of Application Profiles ... 45
7.3 Implementation and Evaluation of a Congestion Control Framework ... 45
7.4 A Preview of the Year 3 Research Agenda ... 46
8 References ... 47
9 Related Publications ... 48

List of Figures

Figure 1: Results from KU-NASA GSFC throughput experiments 3
Figure 2: Results from KU-NASA GSFC throughput experiments 4
Figure 3: AAI host connection status 8
Figure 4: AAI switch VCC connection status 8
Figure 5: WWW user interaction for throughput vs. time plot 10
Figure 6a: Sample throughput — fine time scale 10
Figure 6b: Sample throughput — medium time scale 11
Figure 6c: Sample throughput — coarse time scale 11
Figure 7: NetSpec measurement Scenario 1 13
Figure 8: NetSpec script for blast mode under Scenario 1 14
Figure 9: NetSpec script for CBR mode testing under Scenario 1 14
Figure 10: NetSpec script for random mode testing under Scenario 1 15
Figure 11: NetSpec measurement Scenario 2 16
Figure 12: NetSpec measurement Scenario 3 17
Figure 13: FTP experimental network 19
Figure 14: EDC - GFSC traffic measurement configuration 22
Figure 15a: Five-day traffic flow EDC port A1, VP=0, VC=201 23
Figure 15b: Five-hour traffic flow EDC port A1, VP=0, VC=201 23
Figure 16a: Five-day traffic flow NASA port B1, VP=0, VC=201 24
Figure 16b: Five-hour traffic flow NASA port B1, VP=0, VC=201 24
Figure 17a: Five-day traffic flow EDC port D1 25
Figure 17b: Five-hour traffic flow EDC port D1 25
Figure 18: Configuration of the AAI SC'95 measurements 27
Figure 19a: Example of SC'95 traffic flows 29
Figure 19b: Example of SC'95 traffic flows 29
Figure 20: Experimental configuration for TCP/ATM performance under rate mismatch 32
Figure 21: Simulation model for TCP/ATM performance under rate mismatch conditions 32
Figure 22: Experimental setup of a single connection in the AAI network 33
Figure 23: Simulation model of a single connection in the AAI network 34
Summary of Lessons Learned 40
Figure 24: Year 2 task time line 46

List of Tables

Table 1: AAI host status 7
Table 2a: Connectivity status 7
Table 2b: Switch port status 7
Table 2c: Status of selected PVCs 8
Table 2d: AAI switch status 8
Table 3: Blasting mode throughput testing under Scenario 1 14
Table 4: CBR mode throughput testing under Scenario 1 15
Table 5: Random mode throughput testing under Scenario 1 15
Table 6: Blasting mode testing under Scenario 2 16
Table 7: CBR mode testing under Scenario 2 16
Table 8: Random mode testing under Scenario 2 16
Table 9: Blasting mode testing under Scenario 3 17
Table 10: CBR mode testing under Scenario 3 17
Table 11: Random mode testing under Scenario 3 17
Table 12: Round-trip time and maximum throughput results. An MTU size of9180 bytes was used. 20
Table 13: FTP throughput results with small windows. An MTU size of 4470 bytes was used. 21
Table 14: FTP throughput results with large windows 21
Table 15: AAI SC'95 measurement inventory 28
Table 16: Simulation system parameters 31
Table 17: Throughput for TCP/ATM with rate mismatch 33
Table 18: Throughput of TCP over ATM using AAI network 34
Table 19: Level of model fidelity 35
Table 20: Individual and aggregate throughput for two contending TCP traffic streams 36
Table 21: Individual and aggregate throughput for two contending UDP traffic streams 37
Table 22: Individual and aggregate throughput for a TCP stream flowing from C to A and a UDP stream from B to A 37

Abstract

Wide-area Asynchronous Transfer Mode (ATM) networks are a new and evolving technology. As public services become available, new methods for performance measurements must be developed and applied in testbeds to obtain a better understanding of how advanced applications can take advantage of these high-speed wide-area networks (WANs). A three-year research program is aimed at the measurement of the AAI ATM WAN network performance and the use of developed measurement capabilities to determine the efficacy of congestion control and call admission. This report describes the first-year results on the development of network measurement tools, AAI network performance measurements and troubleshooting, AAI network traffic measurements, the development and validation of AAI simulation models, and initial experiments with congestion control for the AAI.

1 Introduction

The ACTS ATM internetwork (AAI) is an early example of a high-speed national-scale ATM system. The thrust of the AAI effort at the University of Kansas (KU) is the measurement of AAI network performance and the use of those measurement capabilities to determine the efficacy of congestion control and call admission. Our goal is to determine the performance of such a network under stress, as well as to profile the user applications, i.e., characterize the traffic. Issues associated with control and management of realistic networks are being addressed. It is anticipated that these efforts will yield: 1) the performance of a national-scale ATM system; 2) the properties of traffic on such networks; and 3) whether techniques developed and tested on 'small' systems scale up to national networks.

A three-year research agenda with four major tasks was mapped out to achieve the above goals.

KU's first task is measuring the performance of the AAI. An experiment plan has been developed to ensure that the measurements obtained accurately characterize the performance limits of the AAI and that they address the fundamental issues of the project. Techniques to predict the performance of the AAI are being developed to aid in explaining the operation of the network. A cycle of taking measurements, tuning the network and making predictions, taking more measurements, etc. is in process. The second task uses AAI measurements to profile the traffic generated by various applications on the AAI. Evaluation of congestion control techniques for AAI is the third task. Several congestion control techniques are being considered, e.g., i-out-of-m cell pacing, link-by-link flow control, and scheduling mechanisms for guaranteed bandwidth allocation. This task addresses two major issues: 1) how well existing techniques scale to larger networks; and 2) how well subnetworks with different congestion control techniques can interoperate. The final task is the evaluation of call admission control (CAC) on the AAI.

During the first year of the AAI program we have focused effort in five areas: 1) initial network performance measurements and troubleshooting; 2) network measurement tools; 3) network traffic measurements; 4) AAI simulation development and validation; and 5) congestion control for AAI.

In the following section the specific first-year accomplishments will be presented in detail. The major technical challenges that were overcome to achieve those accomplishments will be discussed in Section 3. General lessons that were learned from the first year's efforts will be highlighted in Section 4, while on-going problems will be presented in Section 5. The remaining technical issues to be addressed in Year 2 will be presented in Section 6. The conclusion will be the research agenda for Year 2.

2 Year 1 Accomplishments

2.1 AAI Network Performance and Troubleshooting

As the AAI became operational, TCP throughput measurements between the connected sites was poor, typically on the order of 5 Mb/s or less. The first steps taken to improve the performance were to ensure that the AAI hosts were using large TCP windows and that some form of cell pacing or peak rate limiting was used to overcome OC-3c-to-DS3 rate mismatches. These actions did not improve the system performance. Next, a set of coordinated experiments to determine the AAI bottleneck were conducted. These included observations of cell loss at the AAI hosts, AAI Point-of-Presence (PoP) ATM switches, and within the Sprint ATM network cloud. No cell losses were observed. However, packet losses were observed at the AAI hosts. This led to the belief that the FORE ATM host network interface cards (NIC) were not properly reporting losses.

To confirm this hypothesis, a set of experiments between DEC Alpha workstations at KU and NASA Goddard Space Flight Center (GSFC) were conducted with DEC (as opposed to FORE) ATM interfaces and with virtual circuits routed from GSFC over ATDnet, the AAI, and MAGIC to KU. (These experiments were performed with the help of Javad Boroumand, Suresh Bhogavilli, and Gary Veum at NASA GSFC.) The theoretical maximum throughput for ATM over DS3 was approached in these experiments. The specific results are shown in Figure 1.

Shortly after the KU-GSFC experiments were completed, a new version of the FORE NIC device driver software became available. To determine whether this software would improve the throughput performance, another set of experiments was designed. These experiments were conducted between KU and the Applied Research Laboratory at the University of Texas (ARL-UT). (These experiments were performed with the help of Lenny Tropiano from ARL-UT.) Again the throughput limit for ATM over DS3 was approached, as shown in Figure 2.

Thus, as of the middle of July 1995, TCP-level performance ceased to be a significant problem on the AAI.

In order to effectively evaluate the performance of the AAI, platforms (AAI performance hosts) for experimentation and measurements at AAI sites are required. In many cases, such hosts need to be reconfigured to support a variety of scenarios. The reconfiguration process may involve changing such things as protocol parameters, network interface drivers, and network configurations, all of which may affect usability of the network. In addition, our experience in MAGIC has been that network experiments and debugging may lead to occasional system crashes and disabled network interfaces. KU proposed and partially executed a plan for the deployment of AAI performance hosts.

Initially two SPARC-20 workstations were acquired. Evaluation of these platforms revealed that they would only support DS3-level AAI network testing and thus would not be suitable for stressing the AAI as it evolved to OC-3. The maximum measured TCP-level throughput was about 89 Mb/s. We have thus proposed that OC-3-capable hosts be obtained.

Experiments:

mauchly.ukans.aai.net to cobra.nasa.atd.net

- TCP window size: 208 kB
- MTU size: 9180 bytes
- write buffer size: 64 kB
- RTT: 36.3 ms
- encapsulation: LLC
- cell-level pacing: 34 Mb/s
- throughput: 33.77 Mb/sec
- packet errors: none observed with cell pacing at this level
- packet errors observed with pacing above this level

cobra.nasa.atd.net to mauchly.ukans.aai.net

- TCP window size: 208 kB
- MTU size: 9180 bytes
- write buffer size: 64 kB
- RTT: 36.3 ms
- encapsulation: LLC
- cell-level pacing: none
- throughput: 27.29 Mb/sec
- packet errors: occasional packet errors observed (10 per 30 seconds) with no pacing

Workstation Information:

mauchly.ukans.aai.net

- Location: University of Kansas
- DEC AXP 3000/400, OSF/1 v3.0
- DEC OTTO (ATMworks 750) OC-3c ATM interface

cobra.nasa.atd.net

- Location: NASA Goddard Space Flight Center
- DEC AXP 3000/400, OSF/1 v3.2
- DEC OTTO (ATMworks 750) OC-3c ATM interface

Figure 1: Results from KU-NASA GSFC throughput experiments

However, to facilitate the on-going measurement work, one of the two SPARC-20 workstations has been deployed to ARL-UT. This workstation is on the AAI, and favorable performance tests have been performed. This host played an important role in our ability to collect AAI traffic measurements for SuperComputer'95 (SC'95) as well as other AAI-wide experiments. The results will be discussed in Section 2.3. The other SPARC-20 workstation has been sent to EROS Data Center (EDC) and used in AAI-wide experiments.

```
Experiments:

mauchly.ukans.aai.net to psicorp.utexas.aai.net
        TCP window size: 256 kB
        MTU size: 9180 bytes
        write buffer size: 64 kB
        cell-level pacing: 34 Mb/s
        throughput: 33.34 Mb/sec
        packet errors: none observed with cell pacing at this level
        packet errors observed with pacing above this level

psicorp.utexas.aai.net to mauchly.ukans.aai.net
        TCP window size: 256 kB
        MTU size: 9180 bytes
        write buffer size: 64 kB
        cell-level pacing: none
        throughput: 33.80 Mb/sec
        packet errors: none observed in this direction

Workstation Information:

mauchly.ukans.aai.net
        Location: University of Kansas
        DEC AXP 3000/400, OSF/1 v3.0
        DEC OTTO (ATMworks 750) OC-3c ATM interface

psicorp.utexas.aai.net
        Location: Applied Research Laboratory at the University of Texas
        SGI IRIX 5.3
        Fore OC-3c ATM interface (esa-200 hw=1.6.0 fw=2.3.0)
        ForeThought_3.0.1b (1.28) driver
```

Figure 2: Results from KU-NASA GSFC throughput experiments

2.2 Network Measurement Tools

To develop an understanding of the performance characteristics of the AAI, it was necessary to develop new measurement tools. For this work these tools fall into two categories: tools to generate specific network traffic and collect end-to-end performance; and tools to collect ATM cell-level flows through switches. This section discusses both types of tools.

2.2.1 Developed Network-Wide Performance Measurement Tool: NetSpec

NetSpec is a tool designed to provide convenient and sophisticated support for experiments aimed at evaluating the function and performance of networks. NetSpec provides a far wider range of test types and scenarios than current methods (e.g. ttcp or NetPerf). Accurately characterizing network behavior and performance requires multiple network elements transmitting and receiving a variety of traffic types. Current tools only provide a "full blast" data stream from point A to point B. A far more realistic testing scenario with several connections running at various and variable data rates provides a more realistic evaluation of network capacity and performance. NetSpec was created to enable an investigator, in an automated and reproducible way, to specify a non-trivial network load scenario and then collect a specified set of functional and performance data. NetSpec makes this possible by providing a scripting language in which the investigator specifies an experiment. This description is then used by the NetSpec software to set up, conduct, and report data from the experiment. The execution of the experiment is under software control, and is far more reproducible than those conducted by hand.

NetSpec was designed to overcome the limitations we perceived in the other tools commonly used for network performance testing. Such tools include NetPerf, nettest, and ttcp. NetPerf is an excellent tool, as it supports a variety of protocols, but we found it inadequate for our purposes because it did not perform all of the test types we desired and because it did not provide support for large-scale (multiple connection), automatic, distributed, and highly reproducible experiments. Nettest is a multicast test tool whose functionality resembles ttcp adapted to a multicast environment. NetSpec does not yet provide a multicast environment; that is scheduled for version 3.0. TTCP, while it is among the most commonly used tools for network performance testing, has some significant drawbacks and limitations. First, many implementations are inefficient and include circuit setup and tear down time in the calculation of throughput. Further, most implementations are limited to transmitting information as fast as possible, which is only one kind of network traffic. Finally, the interface is limited, and it is difficult to extend the tool to permit creation of a larger-scale experiment involving multiple connections using ttcp components that can be run reproducibly.

The current version of NetSpec consists of a controller, a test daemon, and a reporter daemon. The controller can control multiple connections, allowing specific repeatable network congestion patterns to be established, as specified in a source file describing an experiment in NetSpec experiment description language (EDL).

The NetSpec EDL provides a rich vocabulary for describing network performance experiments as sets of connections between arbitrary machines. A connection consists of two test daemons and a reporter daemon. The controller controls the daemons by means of a simple binary protocol. All the daemons can run either standalone or under inetd, which is the standard UNIX Internet daemon system. NetSpec experiments can be scheduled for specific durations in time or for the transmission of a specific number of bits. Currently the TCP-level traffic patterns supported are:

- full speed (as fast as the source can transmit to the network);
- Constant Bit Rate, CBR (transmission of a periodic pattern of bursts); and
- random (transmission of a random pattern of bursts).

The full-speed test works like ttcp, which writes as quickly as data can be sent into the network, and similarly receives it as fast as possible at the destination. The CBR test writes data to the network in a paced manner, e.g., one specifies the period in which a certain amount of data gets sent to the network. Since the control of the pacing is done at the application layer (just as a user would generate), the network traffic may not look like CBR traffic at the lower layers. Any differences would be caused by the operating system, which adds variation as data is propagated through the various layers of the network stack. The random test is similar to the CBR test, but instead of the inter-transmission time being constant, it is random following a specified distribution. Currently, NetSpec supports two protocols, UDP and TCP, as these are the most widely used Internet protocols for data transfer. UDP is an unreliable connectionless protocol, while TCP is a reliable connection-oriented protocol.

Even though the current version of NetSpec only collects throughput statistics, the architecture is designed to provide a variety of measurements under future versions, including but not limited to:

- throughput;
- delays, one way and round trip;
- interarrival times of the buffers;
- CPU load in percentage;
- total CPU time spent in system calls;
- total CPU time spent in user code;
- total number of system calls;
- number of units transferred (buffers, packets, pdu's, cells, etc.);
- total time spent executing test; and
- interface-specific statistics.

The current version of NetSpec, 2.0, is currently supported on Sun SPARC SunOS 4.1.3 U1, Sun SPARC Solaris 2.4, SGI Onyx IRIX 5.3, and Digital Alpha OSF1 V3.0. See "http://www.tisl.ukans.edu/netspec/" for more information on NetSpec and to obtain the software.

The application of NetSpec to the evaluation of the AAI will be discussed in Section 2.3.

2.2.2 Developed ATM Flow Monitoring and Statistics Archiving Tools

2.2.2.1 Point Measurements

A set of interactive World Wide Web (WWW) tools was developed to test the status of the AAI on a host, switch, and VC basis. SNMP provided the basis for collecting most of the information. These existing tools provided single-point-in-time views of the AAI and did not provide a general archiving capability. For example, Table 1 shows the result of checking the AAI host status using existing tools.

Monitoring the status of switches within the AAI using the developed WWW tools provided information like that shown in Tables 2a, 2b, 2c, and 2d. Aspects of this tool were extended for use in measuring SC'95 traffic. The columns in these tables typically represent information provided by the standard FORE switch monitoring software, e.g.

VPs/max is the number of VPs in use and the maximum in use at one time.

A tool was also constructed to check the status of connections between specific hosts. Figure 3 is a specific example.

Host	Status	Delay	Last Seen
204.235.71.2	responding	1 msec	Mon., May 15, 1995, 10:19:24
204.235.65.2	responding	172 msec	Mon., May 15, 1995, 10:19:24
eckert-atm.ukans.aai.net	responding	0 msec	Mon., May 15, 1995, 10:19:24
mauchly-atm.ukans.aai.net	responding	0 msec	Mon., May 15, 1995, 10:19:24
204.235.71.130	unreachable	-	
204.235.66.2	unreachable	-	
stacer-atm.ukans.aai.net	unreachable	-	
204.235.67.2	unreachable	-	
hopper-atm.ukans.aai.net	unreachable	-	Mon., May 15, 1995, 9:22:26
204.235.68.2	unreachable	-	

Table 1: AAI host status

```
Status of punx.tioc.magic.net

Ping Results

punx.tioc.magic.net is alive

Fore SPANS Configuration
```

Table 2a: Connectivity status

ASX-200 switch up 10 days, 6:26, 17 ports (9 active), software 3.0.1, hardware 1.0

Port	Name	Uptime	VPs/max	VCs/max	Kb/s	Free	Max	Total Mb
A1	merlin-atm.edc	246:26	4/4	24/24	0	0	155000	29015
A2	gandalf-atm.msc	246:26	3/3	30/30	0	0	155000	20718
A3	cardini-atm.bcb	246:26	1/1	8/8	0	0	155000	16253
B1	blackstone-atm	246:26	1/1	13/13	0	0	100000	6672
B2	slydini-atm	246:26	1/1	13/13	0	0	100000	3388
B3	houdini	82:29	2/2	16/16	0	0	100000	136170
B4	houdini.tioc.	82:29	2/2	20/20	0	0	100000	24660
D1	134.207.18.71	8:51	14/14	33/33	0	0	155000	117087
CTL	punx-atm.tioc.m	246:26	1/1	47/47	0	0	80000	37316

Table 2b: Switch port status

Information similar to that in Table 2a is returned from such queries. Such information may also be obtained on a switch basis as shown in Figure 4.

It was hoped that these WWW tools would aid in the monitoring and status updating of the AAI. However, it was soon learned that these tools were not as useful as expected because the environment lacked stable connectivity. These tools may be accessed (with appropriate permissions) from "http://www.tisl.ukans.edu/Projects/AAI/status/".

Port	Type	Mb/s	State Time	VPs	VCs	BW	Cells	VPs	VCs	BW	Cells
A1	net	155 up	10d 06:27	4	24	155	68 M	4	24	155	78 M
A2	net	155 up	10d 06:27	3	30	155	48 M	3	30	155	57 M
A3	user	155 up	10d 06:27	1	8	155	38 M	1	13	155	135M
A4	user	155 down	10d 06:27	1	63	155	135 M	1	64	155	226 M
B1	user	100 up	10d 06:27	1	13	100	15 M	1	10	100	11 M
B2	user	100 up	10d 06:27	1	13	100	7 M	1	13	100	8 M
	user	100 up	3d 10:30	2	16	100	321 M	2	18	100	124 M
B4	user	100 up	3d 10:30	2	20	100	58 M	2	20	100	272 M
C1	user	45 down	10d 06:27	4	2	45	0	4	2	45	1 M
C2	user	45 down	10d 06:27	4	3	45	873946	4	3	45	2 M
C3	user	45 down	10d 06:27	3	3	45	531871	3	3	45	2 M
C4	user	45 down	10d 06:27	1	2	45	0	12	45	45	1 M
D1	net	155 up	08:52:08	14	33	155	276 M	14	35	155	66 M
D2	user	155 down	10d 06:27	1	2	155	0	1	3	155	1 M
D3	user	155 down	10d 06:27	1	2	155	0	1	3	155	1 M
D4	user	155 down	10d 06:27	1	2	155	0	1	2	155	1 M
CTL	user	80 up	10d 06:27	1	47	80	88 M	1	45	80	64 M

Table 2c: Status of selected PVCs

Source	Destination	In Port	In VPI	In VCI	Out Port	Out VPI	Out VCI	Mb/s	Uptime	BW	Cells
mauchly .ukans.aai.net	houdini .tioc.aai.net	A4	0	210	B4	0	210	0	10d 06:28	0	1 M
houdini .tioc.aai.net	mauchly .ukans.aai.net	B4	0	210	A4	0	210	0	10d 06:30	0	1 M

Table 2d: AAI switch status

Please enter the source address: mauchly.ukans.aai.net
(e.g., mauchly.ukans.aai.net)

Please enter the destination address: houdini.tioc.aai.net
(e.g., houdini.tioc.aai.net)

Note that checking the VCC status may take several minutes.

Figure 3: AAI host connection status

Please enter the name of the switch: merlin.edc.magic.net
(e.g., merlin.edc.magic.net)

Note that checking the status of all VCCs on the switch may take several minutes.

Figure 4: AAI switch VCC connection status

2.2.2.2 Automatic ATM Flow Monitoring, Statistics Archiving, and Traffic Visualization Tools

The development of the above tools indicated that there was a need for automatic ATM flow monitoring, statistics archiving, and traffic visualization tools. Such tools will facilitate capturing properties of AAI traffic. A set of WWW-based tools has been developed for this purpose. Again, SNMP is used to request the information from remote AAI switches; in this case, time-stamped cell counts are continuously collected from the connected AAI PoP switches. Information is sorted according to port number. Traffic being received by the switch can also be sorted based on VP/VC number (traffic leaving the switch, being transmitted, cannot be sorted based on VP/VC number because of limitations in the FORE MIB). From time-stamped cell counts, traffic flows, i.e., throughput vs. time, can be constructed. The WWW interface allows the user to construct a throughput vs. time plot for specific switch/port/VP/VC and a time scale, e.g., last 1 hour, 3 hours, or all available data. An example of the user interaction is given in Figure 5, and samples of the time plots are provided in Figures 6a, 6b, and 6c.

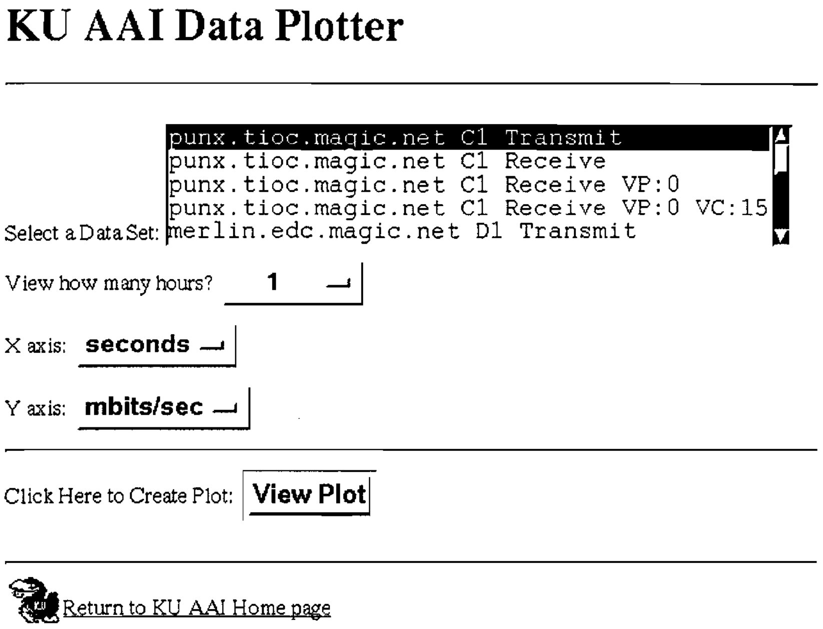
This set of automatic collection tools will become important in the next year as more of the AAI sites begin to routinely use the AAI for distributed processing. Data collected from this tool will provide the basis for quantifying the properties of AAI traffic. In Section 2.3.4, such throughput vs. time data will be presented for EDC-to-GSFC traffic during a period in December 1995.

2.3 AAI Network Measurements

A goal of the AAI research is to conduct systematic studies on AAI performance, applications, and traffic flows. This section will begin with a review of a set of AAI performance testing scenarios. The results from an initial set of such scenarios will be presented next. In the following section, the quantitative performance of a simple network application, in this case ftp over the AAI, will be discussed. During December 1995, EDC began to utilize the AAI for communications with GSFC; an example of the EDC traffic flows will be presented. Also in December 1995, several sets of throughput vs. time data were collected in association with the SuperComputer'95 Conference (SC'95). A discussion of the AAI resources used to collect this data, along with several examples, will be presented. Extensive analysis and conclusions for this data are not included here because much of this data has only recently been collected. Such analysis remains for Year 2.

2.3.1 Developed a Set of AAI Performance Testing Scenarios

This section can be seen as a plan for the performance experiments to be conducted on the AAI network. As a result of these experiments, we expect to quantify the characteristics of the AAI as well as identify potential bottlenecks that may affect network performance. As a final result, in the future we will suggest admission, flow, and congestion control techniques that may be used to improve this performance. The scope of this section is to identify the scenarios (topologies, duration, etc.) in which the experiments will be performed. Details concerning these scenarios can be found in [1]. Basically there were five configurations developed in [1].



KU AAI Data Plotter

Select a Data Set:
```
punx.tioc.magic.net C1 Transmit
punx.tioc.magic.net C1 Receive
punx.tioc.magic.net C1 Receive VP:0
punx.tioc.magic.net C1 Receive VP:0 VC:15
merlin.edc.magic.net D1 Transmit
```

View how many hours? 1

X axis: seconds

Y axis: mbits/sec

Click Here to Create Plot: View Plot

Return to KU AAI Home page

Figure 5: WWW user interaction for throughput vs. time plot

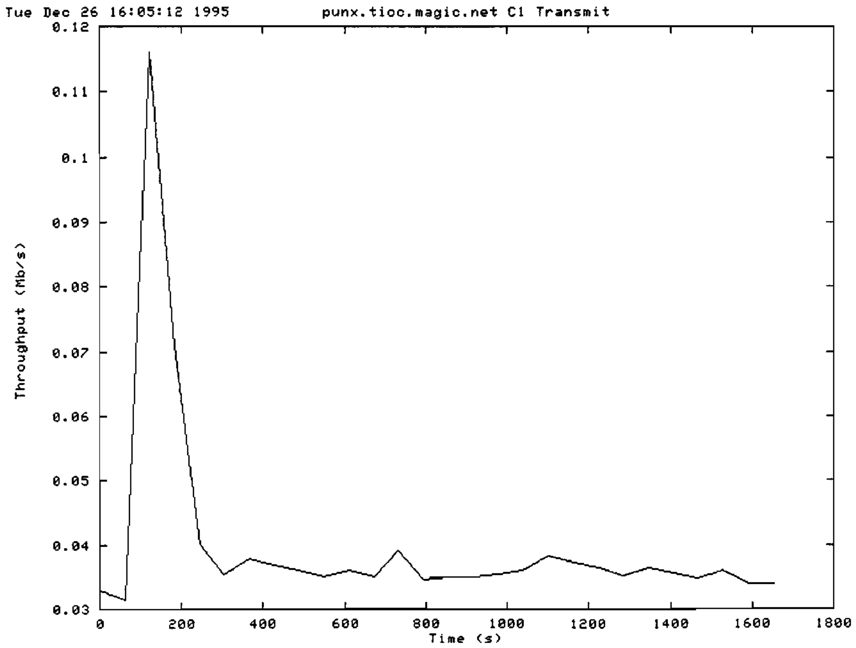


Figure 6a: Sample throughput — fine time scale

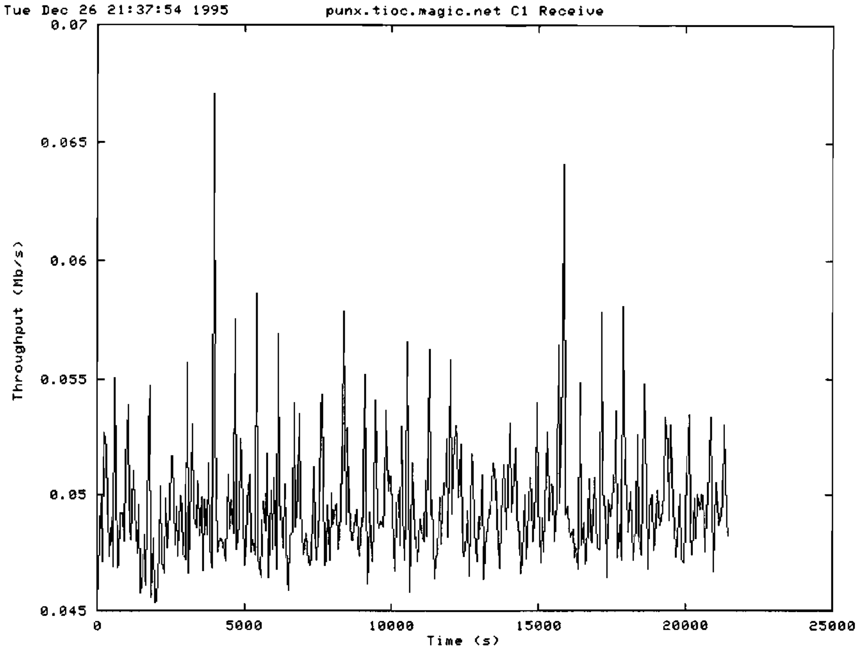


Figure 6b: Sample throughput — medium time scale

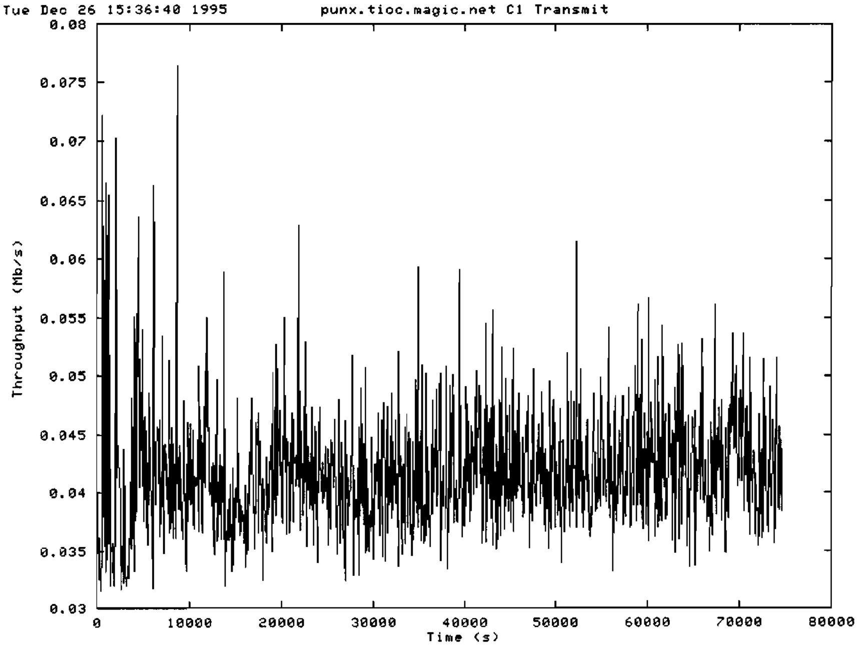


Figure 6c: Sample throughput — coarse time scale

Point-to-point full-duplex configuration

There are 28 different possible pairs of point-to-point configurations in the AAI network. See Figure 7 as an example. Performance testing will be done point-to-point separately in an under-load state to determine the maximum attainable rates and other parameters.

Three-site pair full-duplex configuration
Traffic will be set up simultaneously between 3 pairs of measurement sites in the underload state. There are a total of 28 such combinations. See Figure 11 as an example.

Switch-by-switch two-way full-duplex configuration
Traffic will be set up between one given site and all other sites connected to the same AAI core switch. In this case, all traffic must go through only one switch. Therefore, there are five possible combinations. In the current AAI topology, this scenario is only possible for the Chicago switch, since for the California switch it would be equivalent to the point-to-point full duplex configuration, and there is only one main site directly connected to the Maryland switch. See Figure 12 as an example.

Two-way full-duplex configuration
Traffic will be set up between one site and every other site in the network in the underload state. In this case connections for all links are set up simultaneously.

All-site full-duplex mesh configuration
In this scenario, traffic will be sent simultaneously on all measurement sites. Each measurement site sends traffic to every other site, resulting in a mesh configuration. Notice this configuration is equivalent to performing all combinations of point-to-point testing simultaneously. Under this configuration, traffic will travel through all three switches.

Initial results from measurements for some of these scenarios will be discussed in the next section.

2.3.2 Measured Performance of Multiple AAI Sites Using NetSpec

The following measurements serve both as a demonstration of the use of NetSpec and a preliminary execution of a part of the performance testing plan presented in Section 2.3.1. As explained before, the NetSpec measurement tool is able to generate traffic in three different modes:

- blasting mode;
- CBR (constant bit rate) mode; and
- random mode.

The burst size is determined by a message size and a number of messages to be sent. For example, if the message size is 65536 bytes, and the number of messages per burst to be sent is 10, then the burst size is 655360 bytes; furthermore, if the time per burst is 1 second, then the traffic source is paced to send a burst of 655360 bytes every second.

The random mode traffic source sends a specified burst of data with a uniformly distributed time per burst. Similar to the CBR mode, the burst size is determined by a message size and a number of messages per burst to be sent. The time per burst is uniformly distributed (i.e. the probability density function is uniform) between two specified values: minPeriod and maxPeriod. Three measurement scenarios will be discussed in this section.

2.3.2.1 Scenario 1

The first scenario can be seen as a point-to-point configuration described in Section 2.3.1. Two sites are involved in testing Scenario 1. As shown in Figure 7, workstation A is at EDC, and workstation B is at ARL-UT

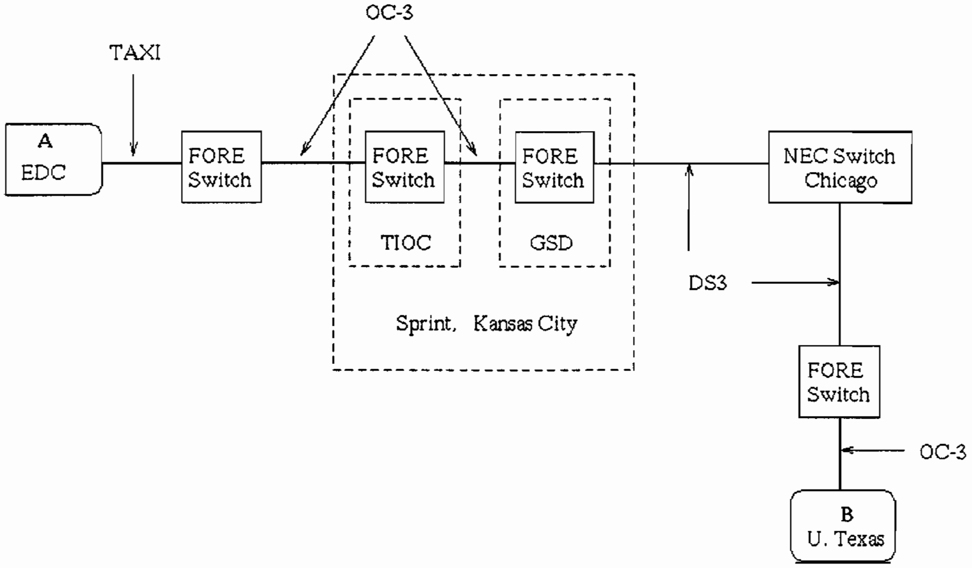


Figure 7: NetSpec measurement Scenario 1

In the first Scenario 1 experiment, workstations A and B are sending to each other simultaneously (duplex connection). The corresponding NetSpec script is shown in Figure 8. The testing results are shown in Table 3.

Note that the poor performance results from the lack of pacing and small window size in this experiment.

The CBR mode of NetSpec was used in the second Scenario 1 experiment, where workstation B sends data to workstation A (simplex connection). Note that the CBR mode approximates TCP- (or UDP-) level pacing. In this experiment, the goal is to determine the maximum throughput using UDP-level pacing. A duplex connection is not suitable for this experiment because workstations may be "overloaded" due to simultaneous transmission and reception.

Figure 9 shows the corresponding NetSpec script, and Table 4 shows the testing results. Note that when pacing is used at the UDP level, the throughput performance can approach the DS3 link rate despite the link mismatch situation. In this particular measurement, the message size is 18432 bytes, and the number of messages per burst to be sent is 5; this results in a burst size of 91260 bytes. Also, the time per burst is 20 ms. Therefore the pacing rate is 36.864 Mbps.

```
experiment{
parallel{
 connection{
   xmcntl = utexas.perf.ukans.aai.net;
   rcvcntl = 204.235.71.130;
   xmt = utexas.perf.ukans.aai.net (xmtbuf = 51200);
   rcv = 204.235.71.130 (rcvbuf = 51200);
   rep = utexas.perf.ukans.aai.net;
   protocol = tcp;
   test = full(messagesize = 65536);
   delay = 0 s;
   duration = 100 s;
  }
  connection{
   xmcntl = 204.235.71.130;
   rcvcntl = utexas.perf.ukans.aai.net;
   xmt = 204.235.71.130; (xmtbuf = 51200);
   rcv = utexas.perf.ukans.aai.net; (rcvbuf = 51200);
   rep = 204.235.71.130;
   protocol = tcp;
   test = full(messagesize = 65536);
   delay = 0 s;
   duration = 100 s;
  }
}
}
```

Figure 8: NetSpec script for blast mode under Scenario 1

	Tx (Mbps)	Rx (Mbps)
A-B	7.949	7.931
B-A	8.901	8.848

Table 3: Blasting mode throughput testing under Scenario 1

```
experiment {
  connection {
   xmtcntl = utexas.perf.ukans.aai.net;
   rcvcntl = 204.235.71.130;
   xmt = utexas.perf.ukans.aai.net (xmtbuf=51200);
   rcv = 204.235.71.130 (rcvbuf = 51200);
   rep = utexas.perf.ukans.aai.net;
   protocol = udp;
   test = cbr(messagesize = 18432:5, period=20000);
   delay = 0 s;
   duration = 100 s;
  }
}
```

Figure 9: NetSpec script for CBR mode testing under Scenario 1

	Tx (Mbps)	Rx (Mbps)
B-A	35.449	34.000

Table 4: CBR mode throughput testing under Scenario 1

To illustrate the use of random traffic, an experiment was conducted where workstation B sent data to workstation A (simplex connection). Figure 10 shows the corresponding NetSpec script, and Table 5 shows the testing results.

```
experiment{
 connection{

  xmtcntl = utexas.perf.ukans.aai.net;
  rcvcntl = 204.235.71.130;
  xmt = utexas.perf.ukans.aai.net (xmtbuf=51200);
  rcv = 204.235.71.130 (rcvbuf = 51200);
  rep = utexas.perf.ukans.aai.net;
  protocol = udp;
  test = random(messagesize = 8192:9, minPeriod=20000, maxPeriod=40000);
  delay = 0 s;
  duration = 100 s;

  }
}
```

Figure 10: NetSpec script for random mode testing under Scenario 1

	Tx (Mbps)	Rx (Mbps)
B-A	21.097	21.052

Table 5: Random mode throughput testing under Scenario 1

The above results are consistent with our expectations.

2.3.2.2 Scenario 2

The second scenario can be seen as a simplified switch-by-switch two-way configuration as described in Section 2.3.1. Three sites (Figure 11) are involved in the testing under this scenario. Simplex connections are used for all testing under Scenario 2.

The results of blasting mode testing under Scenario 2 are shown in Table 6. The relatively low throughput performance resulted from the link mismatches.

The CBR and random traffic were also used in Scenario 2. The following NetSpec parameters were used for both connections in this testing: message size is 65536 bytes, number of messages per burst to be sent is 10, and time per burst is 1 second. This set of parameters results in a pacing rate of 5.243 Mbps for both connections. The TCP protocol is used for both connections. The throughputs for CBR are shown in Table 7, and the random results are in Table 8. The following NetSpec parameters were used for both connections in the random mode: a message size of 8192 bytes, the number of messages per burst to be sent is 4, minPeriod is 20 ms, and maxPeriod is 40 ms.

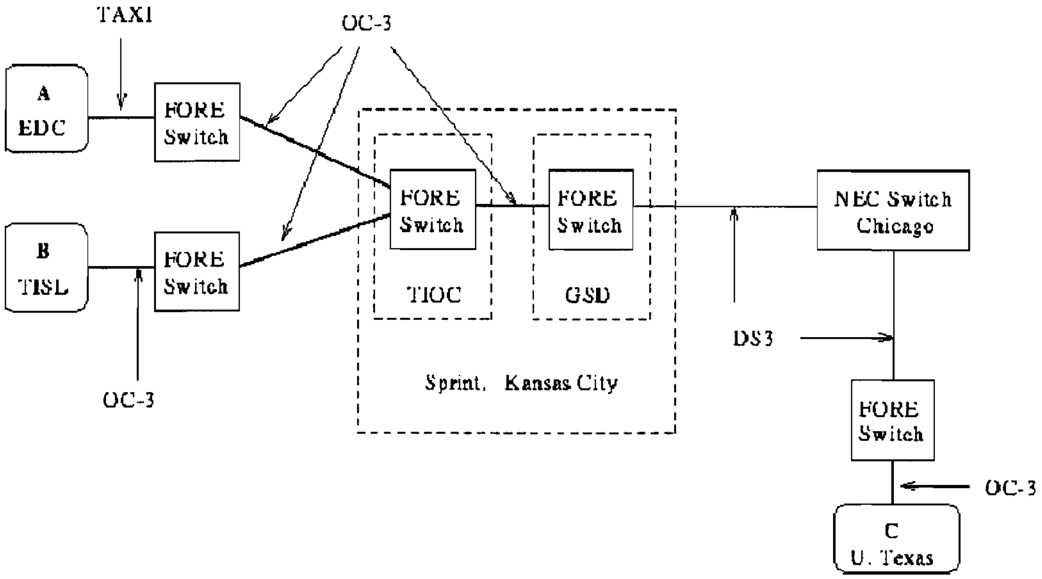


Figure 11: NetSpec measurement Scenario 2

	Tx (Mbps)	Rx (Mbps)
A-C	7.912	7.872
B-C	6.001	5.973

Table 6: Blasting mode testing under Scenario 2

	Tx (Mbps)	Rx (Mbps)
A-C	5.213	5.211
B-C	5.147	5.109

Table 7: CBR mode testing under Scenario 2

	Tx (Mbps)	Rx (Mbps)
A-C	7.693	6.972
B-C	9.401	8.150

Table 8: Random mode testing under Scenario 2

2.3.2.3 Scenario 3

As shown in Figure 12, four sites are involved in testing Scenario 3. Workstation A is at

EDC, workstation B is at TISL, workstation C (an SGI Onyx) is at the Sprint TIOC, and workstation D is at ARL-UT.

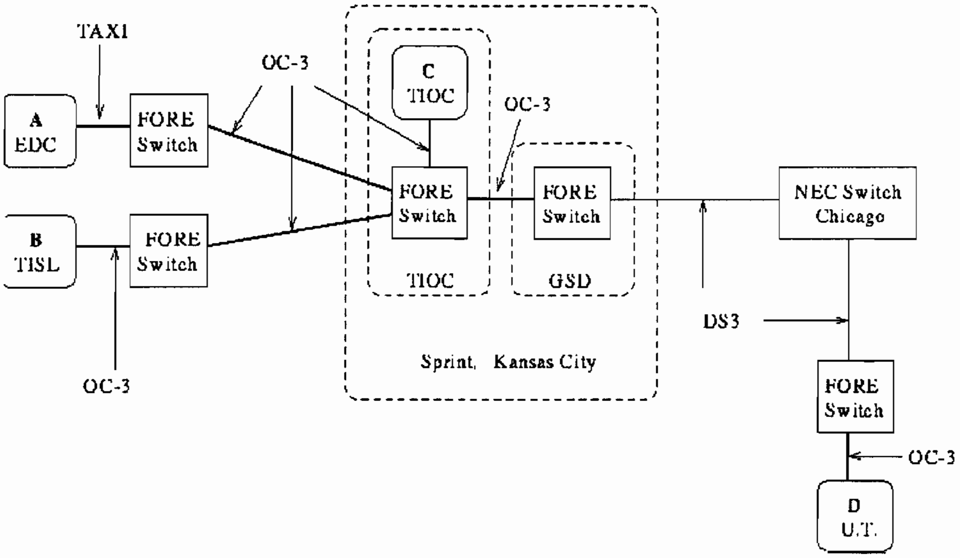


Figure 12: NetSpec measurement Scenario 3

Tables 9, 10, and 11 present the results for the blast, CBR, and random traffic respectively.

	Tx (Mbps)	Rx (Mbps)
A-C	25.403	25.370
B-D	7.793	7.706

Table 9: Blasting mode testing under Scenario 3

	Tx (Mbps)	Rx (Mbps)
A-C	29.319	29.287
B-D	29.366	29.204

Table 10: CBR mode testing under Scenario 3

	Tx (Mbps)	Rx (Mbps)
A-C	21.096	21.102
B-D	17.190	17.030

Table 11: Random mode testing under Scenario 3

The parameters used for CBR are as follows: message size is 8192 bytes, number of messages per burst to be sent is 9, and time per burst is 20 ms. This set of parameters results in a pacing rate of 29.49 Mbps. The parameters used for the random traffic for both connections are as follows: message size is 8192 bytes, number of messages per burst to be sent is 9, the minPeriod is 20 ms, and the maxPeriod is 40 ms. The UDP protocol is used for both connections for both CBR and random traffic. The results in Table 9 again show the effect of rate mismatches. Link B-D has an OC-3 to DS3 mismatch.

2.3.3 ATM WAN File Transfer Performance Experiment

The mid-1990s have witnessed the deployment of the first wide-area ATM networks, operating at speeds ranging from a few tens of megabits per second to a few hundreds of megabits per second. The increasingly widespread availability of such bandwidths enables, among a variety of other services, the transfer of large amounts of data in short periods of time. FTP (File Transfer Protocol) is a popular application for transferring data between two hosts. It provides for the establishment of two separate connections between the source and destination hosts. One of the connections is used for the login between the machines, and uses the Telnet protocol; the other connection is used for the actual transfer of data. Here, experiments were conducted with the AAI wide-area ATM network, and actual FTP transfers were employed to duplicate a real-world application. This section deals with the main limiting factors for FTP in such a scenario and ways to minimize these limitations.

Even though FTP is a mature and widely used protocol, its use in wide-area high-speed ATM networks presents a new set of issues. The main issue is one that faces any application utilizing the Transmission Control Protocol (TCP). FTP sits on top of TCP on the protocol stack, making use of TCP services such as reliable transmission. End-to-end reliability is implemented by TCP through a sliding window-based acknowledgment and retransmission scheme. The transmitting host is allowed to transmit no more than N messages before the first one is acknowledged. At any given moment, there are no more than W unacknowledged bytes within a TCP connection — we shall refer to W as the TCP window size. A significant limiting factor of throughput for TCP applications over high-speed large delay networks can be the TCP window size. The window size (in bytes) needed to ensure continuous transmission in the absence of packet errors can be calculated simply as the product of the link speed times the round-trip delay. In this fashion, for a link operating at about 34 Mbps (for ATM over DS3), as in the cases considered here, and a round-trip delay of 60 ms, a window size of 255,000 bytes would be needed. On the other hand, in the case of a coast-to-coast connection within the U.S., conventional TCP window sizes (i.e., 32 to 64 kilobytes) limit the throughput to about 8 to 9 Mbps. Once a proper window size is used, the next major limiting factor is the disk access speed. For instance, fast SCSI disks available at many sites operate at around 12 Mbps. Clearly, the use of fast, wide SCSI disks capable of operating in the tens of Mbps is needed to take full advantage of a high-speed connection.

The AAI network (augmented with a connection to the Jet Propulsion Laboratory) was used for these experiments. In this configuration, the system emulates a distributed national network with a site on each coast and two sites in the central region of the United States. The network is depicted in Figure 13.

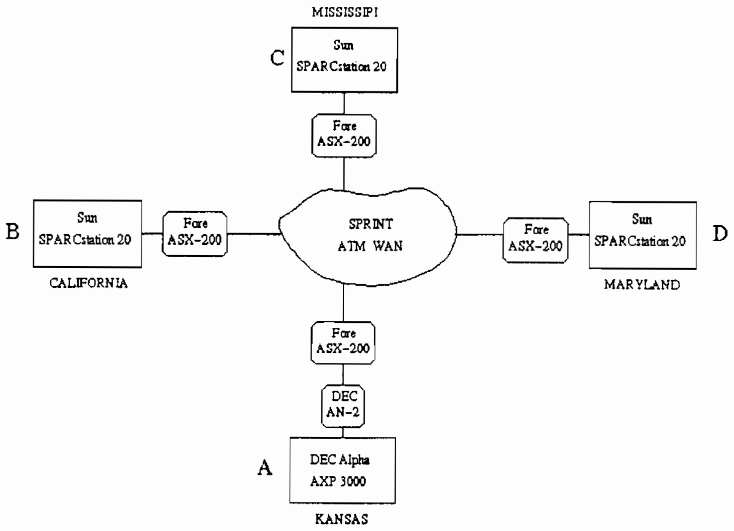


Figure 13: FTP experimental network

Line rates within the wide-area ATM network are DS3 rates (45 Mbps). Because some of the hosts are connected to the respective switches at a higher rate, traffic pacing was used to avoid rate mismatches. A set of Permanent Virtual Circuits (PVCs) was designed to connect these sites, forming a full mesh grid. In using FTP, the window size is determined by negotiation between the client (the host running the FTP source code) and the server (the host running the FTP daemon). Therefore, both the source and daemon codes must be modified to allow the use of large windows. For these experiments, the available FTP code running under Solaris 5.4 and OSF1 3.0 was modified. The modification occurs at the point where a socket is set up for the transfer of data. The definition of the window size to be used is one of the parameters for this socket.

A series of measurements was performed in order to determine the performance of the file transfers under different conditions. These measurements included the following.

- Round-trip delay — Round-trip time (RTT) was measured to determine the theoretical TCP window size needed to ensure continuous transmission. "Ping" was used to estimate this delay.

- Maximum throughput — The TTCP tool was used to measure maximum throughput.

- File transfer throughput — File transfer throughput was obtained directly from the FTP application. File sizes ranging from tens of MB to hundreds of MB were used, recognizing the fact that high volumes of data are expected to be transmitted over high-speed networks.

- Cell counts — ATM-layer measurements were taken at intermediate switches. These measurements were primarily aimed at identifying possible cell losses and/or retransmissions.

The round-trip delay in wide-area connections is controlled primarily by the circuit distance between the transmitting and receiving sites. Round-trip times measured in our experiments ranged from 36 ms to a maximum of 73 ms. Each PVC had consistent round-trip times over a number of tests. These results, summarized in Table 12, correlate very well with the actual physical route miles associated with each connection.

Source/Destination	RTT (msec)	Window Size (bytes)	Measured Throughput (Mbps)
B-A	61	31290	3.9
		262144	34.6
B-D	62	31290	4.2
		409600	23.8
A-D	36	31290	5.9
		352144	15.7
D-C	39	314570	30.84
B-C	73	524288	29.10
A-C	73	262144	27.01

Table 12: Round-trip time and maximum throughput results. An MTU size of 9180 bytes was used.

Table 12 summarizes the maximum achievable throughput (measured using TTCP) between two hosts for different TCP window sizes. These results are consistent with the window limiting the throughput, given by:

$$\text{Maximum Throughput} = \text{Window Size}/\text{RTT}$$

If we take as an example the first case shown in Table 12, we see that the throughput should be limited to 4.1 Mbps. This is in close agreement with our measured throughput (3.9 Mbps). On the other hand, as we increase the window size, the theoretical link rate is approached.

The additional overhead of FTP, and especially the additional disk reads and writes, will reduce the effective throughput to values lower than those reported by TTCP in Table 12 for the same connection.

A Maximum Transmission Unit (MTU) size of 4470 was utilized in these file transfer experiments. This value is of special significance due to the fact that a very large number of legacy LANs utilize FDDI connections, with transmission units of 4470 bytes. Table 13 summarizes the results for a number of transfers with different size files utilizing regular window sizes (32 to 64 KB).

These low values of throughput agree with what one would expect for high-speed data transfers using small TCP windows. We may also notice that the throughput seems to be uncorrelated with the size of the files being transmitted. The experiment was then repeated using the modified version of FTP with the results for the case with the largest RTT in Table 14.

Source/Destination}	File Size (million bytes)	Throughput (Mbps)
B-C	14	2.56
B-A	17	3.60
	55	3.63
D-B	75	2.88
D-A	75	4.61
D-C	75	3.39
A-D	333	2.56
A-B	333	1.76
A-C	333	3.28

Table 13: FTP throughput results with small windows. An MTU size of 4470 bytes was used.

Source/Destination	MTU Size (Bytes)	File Size (million bytes)	Throughput (Mbps)
B-D	4470	17	15.2
B-D	4470	55	12.8
D-B	9188	17	22.4

Table 14: FTP throughput results with large windows

By adapting FTP to use large windows, we were able to increase throughput by one order of magnitude. The flow control exercised by TCP is no longer the limiting factor; we believe that at this point, we have reached the limitations imposed by the disk access times. An experiment was performed in which three separate traffic transfers were done simultaneously (A to B, A to C, A to D). Since we are operating significantly below the network capacity, the three traffic streams did not cause congestion. Thus, no performance penalty was observed. The investigation of wide-area network performance under competing traffic streams is one of the areas of Year 2 research.

Cell counts were collected at intermediate switches; these measurements serve the purpose of identifying whether a significant number of retransmissions occurred. Consider as an example the transmission of 14 million bytes, as in the case of the transmission from B to C (first case reported in Table 12). Since an ATM cell can carry up to 48 bytes of information, we would expect a minimum of 291,670 cells. The actual number of cells measured at an intermediate switch was 297,375; the experiment was repeated three times with the same result. Considering that some cells may be only partially filled, such a result seems to indicate the absence of significant cell losses within the network. All the connections behaved in the same fashion.

These experiments indicate that the high-speed transfer of files over a wide-area ATM network using FTP is achievable under two conditions: the protocol must use TCP window sizes compatible with the round-trip delay for the connection (typically, this will be in the range of hundreds of thousands of bytes); and all hosts involved must be equipped with disk drives whose access rates are higher than the transmission rate provided by the network.

These experiments were conducted with collaboration from Rick Lett from Sprint GSD, Daniel Bromberg from NASA Jet Propulsion Laboratory, Suresh Bhogavilli and Javad Boroumand from NASA Goddard Space Flight Center, and Kenneth Finnegan from Naval Research Laboratory. Also, Sprint's Broadband Operations Center (BBOC) provided valuable assistance monitoring these tests. (See [9] for additional details of this FTP network performance study.)

2.3.4 EDC Traffic Measurements

The ATM cell flow monitoring tools have been applied to collect the traffic between EDC and GFSC. This traffic is representative of the early EDC use of the AAI. The configuration in use by EDC is shown in Figure 14. This cell flow monitoring is automatic, and the data is continuing to be archived. The traffic transmitted from the EDC ISS host to the EDC ATM switch and then on to GFSC for a five-day period is shown in Figure 15a, and the approximate five-hour period with significant traffic is shown in Figure 15b. The corresponding flow received at port B1 on the GSFC ATM switch is shown in Figures 16a and 16b. (The traffic transmitted out of port A0 on the GFSC switch is identical to the B1 traffic.) All the traffic leaving the EDC switch from Port D1 toward the WAN is given in Figures 17a and 17b. Note the traffic transmitted from this port is not identical to the port A2 flows because there are other traffic flows from EDC to the ATM WAN. Traffic analysis remains for Year 2.

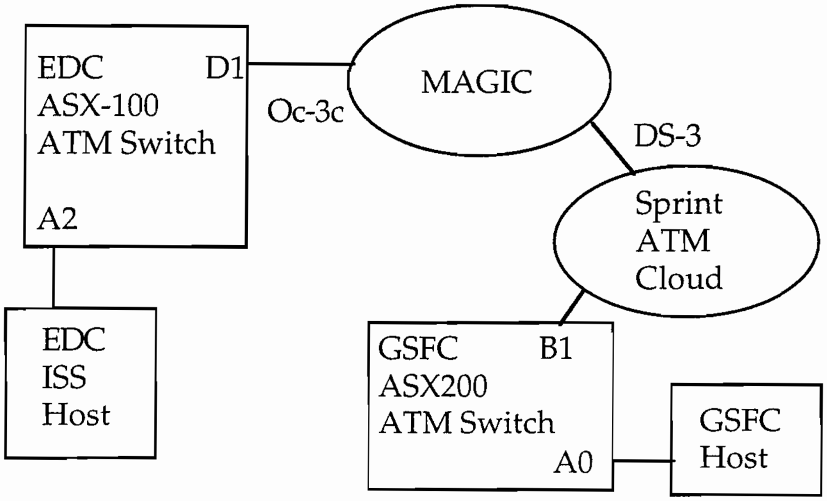


Figure 14: EDC - GFSC traffic measurement configuration

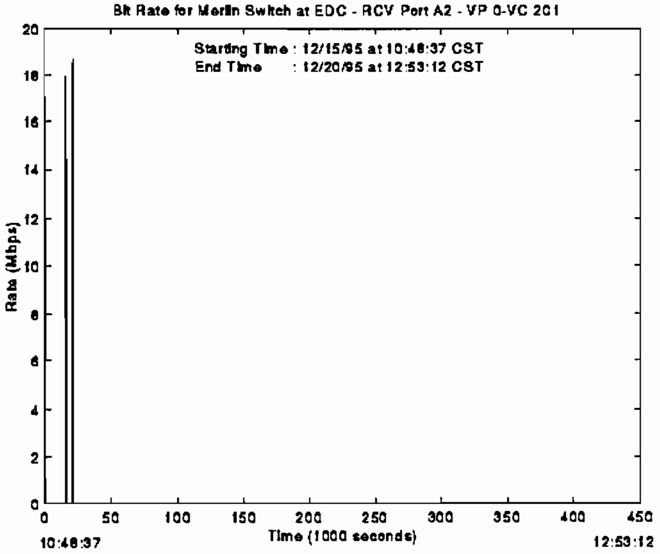


Figure 15a: Five-day traffic flow EDC port A1, VP=0, VC=201

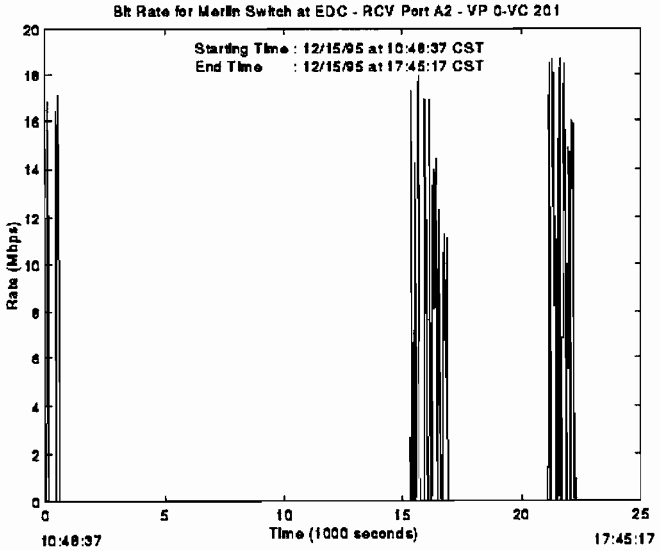


Figure 15b: Five-hour traffic flow EDC port A1, VP=0, VC=201

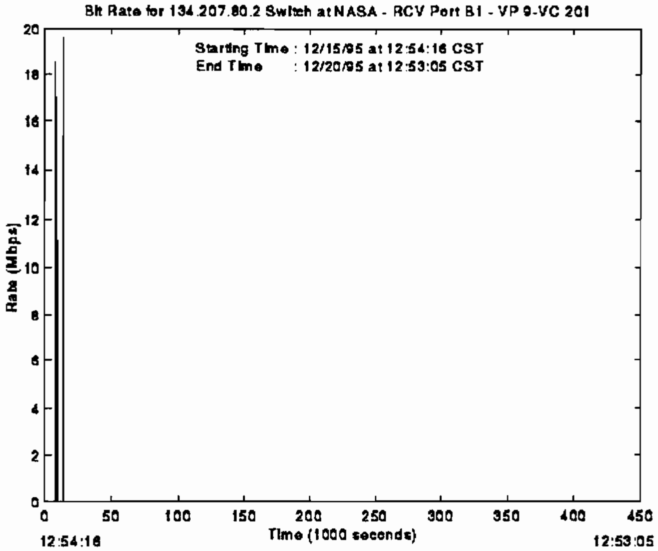


Figure 16a: Five-day traffic flow NASA port B1, VP=0, VC=201

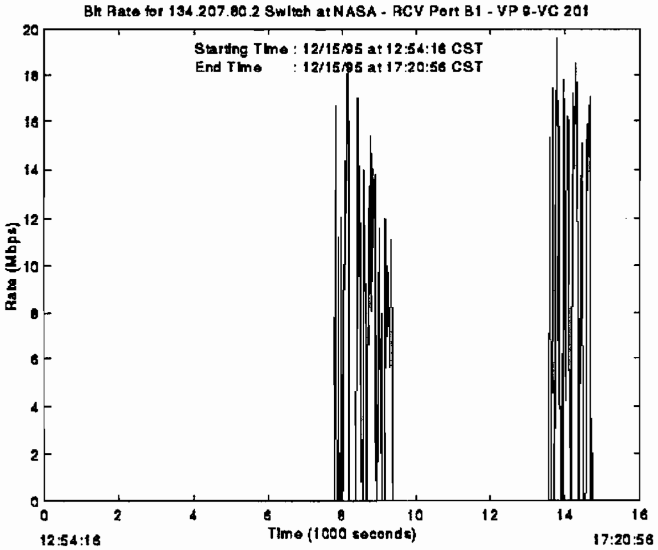


Figure 16b: Five-hour traffic flow NASA port B1, VP=0, VC=201

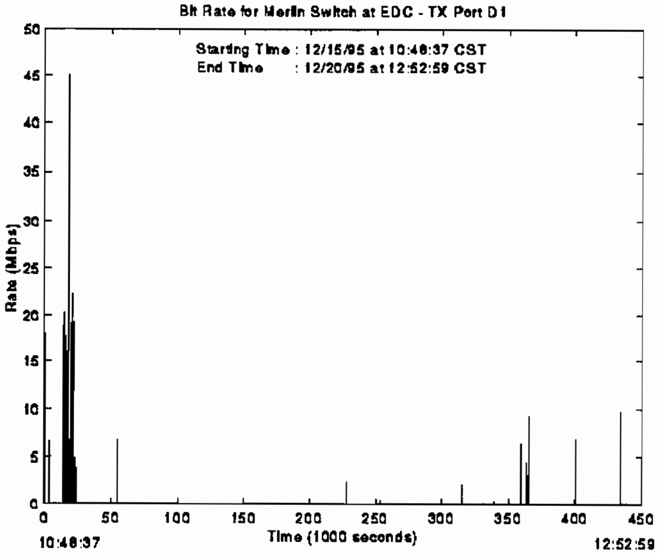


Figure 17a: Five-day traffic flow EDC port D1

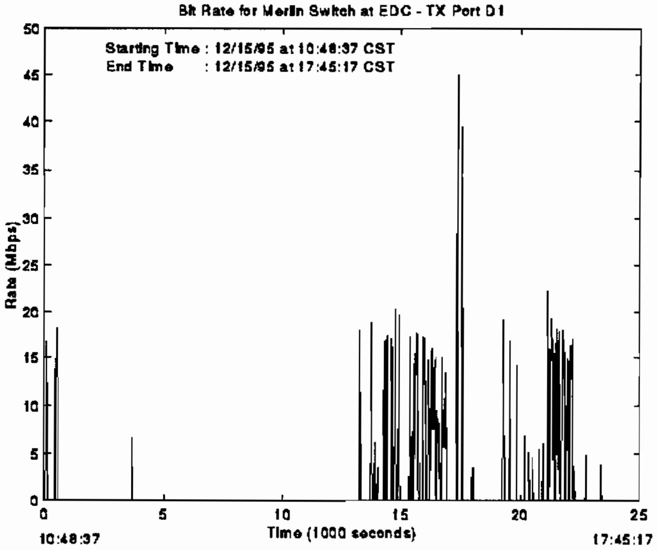


Figure 17b: Five-hour traffic flow EDC port D1

2.3.5 SC'95 Measurements

Several AAI sites participated in SuperComputer'95 (SC'95) WAN demonstrations during the week of December 4, 1995. A measurement program was conducted to attempt to characterize the AAI traffic flows for the SC'95 demonstrations and to identify any resulting network congestion. The measurement tools used in the above studies were tailored for SC'95 based on convention floor and site port/VP information as provided to us. Flows through a total of eight ATM switches were monitored. Six of these switches were at AAI sites while two were on the convention floor as shown in

Figure 18.

A specific inventory of the data collected is given in Table 15. Highlighted entries in Table 15 indicate the presence of significant bursts of traffic. Note no buffer overflows were observed on any of the monitored ports/VPs; this may be due to the lack of congestion or, as has been previously observed, incorrect reporting of overflows by the switches. A complete set of plots of the traffic flows represented in Table 15 were provided to Sprint shortly after SC'95. Also, due to a programming error, receive traffic was not separated by VP on the last day (12/7-12/8) of the demonstrations.

Examples of the type of data collected are given in Figure 19a. Traffic flows from Phillips toward SC'95 and the traffic flow into port B1 of the Sprint floor switch are presented.

An active TCP-level throughput experiment was also designed for SC'95 using NetSpec. A NetSpec experiment was constructed and tested locally to transmit a stair-step pattern (1, 2, 3, 4, 5 Mb/s for one minute each), repeated every half hour from a remote host to a host on the SC'95 floor. Unfortunately, a throughput of no greater than 1 Mb/s could not be obtained between the remote and floor hosts, even though two different floor hosts were used, a Sun SPARC-5 and an SGI Indigo. The NetSpec program was then changed to a 'full bast' test repeated every thirty minutes.

The collected SC'95 data offers an opportunity to study 'supercomputer'-like traffic flows. In addition to the data reported above, Sprint collected cell counts on a 15 min. time scale. Examining and correlating the fine scale (approximately every 15 sec) properties of the above data with the coarse scale (approximately every 15 min.) Sprint data will be of interest. Issues related to the self-similar nature of such data may be possible. It remains for Year 2 to conduct such studies.

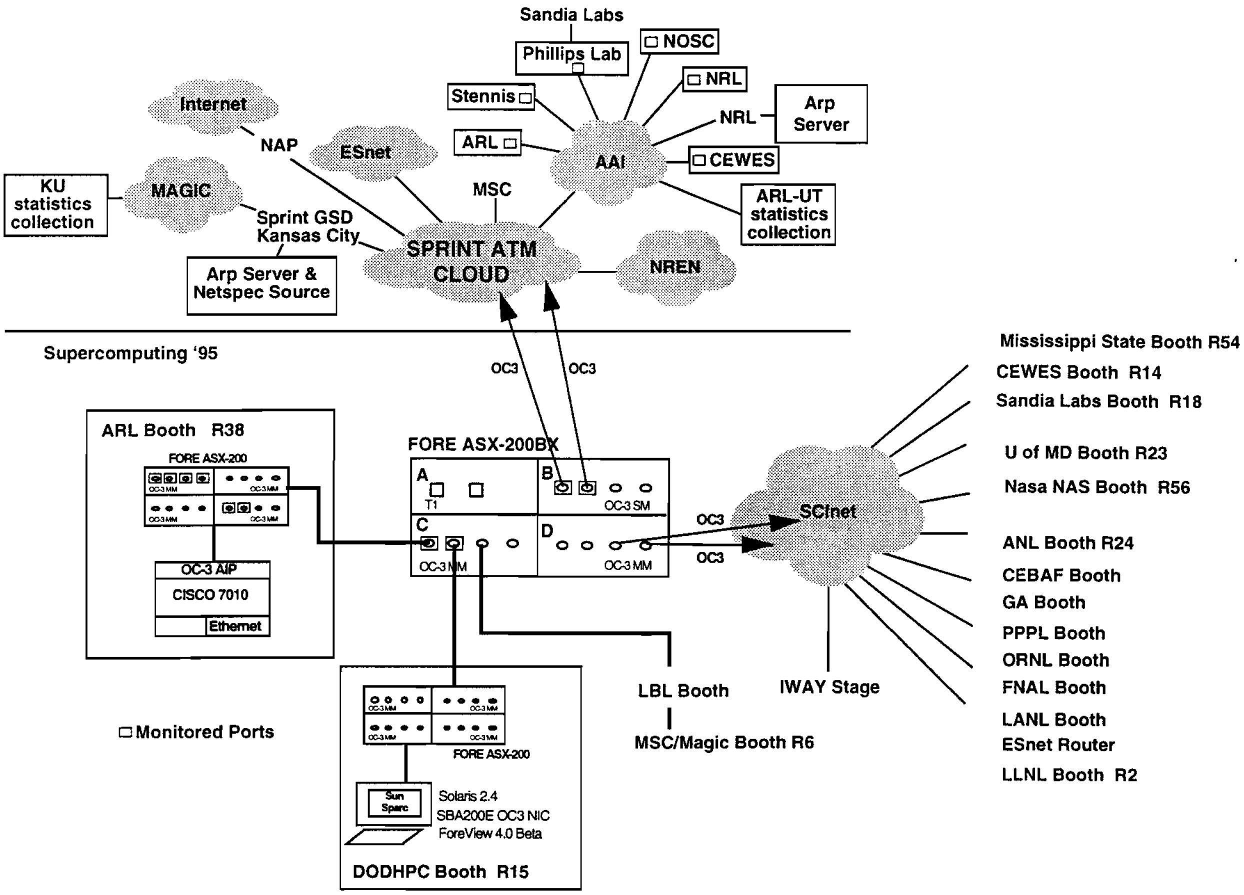


Figure 18: Configuration of the AAI SC'95 measurements
(Figure based on original provided by Laura Klein Sprint GSD)

Switch	T/R	Port	VP (only on receive)	Start Time (CST)	End Time (CST)	Significant burst
Sprint Floor	T	B1		12/4/95 14:48:34	12/8/95 11:09:06	Yes
Sprint Floor	T	B2		12/4/95 14:49:00	12/8/95 11:09:07	Yes
Sprint Floor	T	C1		12/07/95 14:49:23	12/08/95 11:09:08	
Sprint Floor	R	B1	All	12/07/95 13:05:51	12/08/95 11:09:19	Yes
Sprint Floor	R	B1	10 (Phillips)	12/04/95 13:27:35	12/07/95 13:40:01	Yes
Sprint Floor	R	B1	4 (CEWES)	12/04/95 13:27:25	12/07/95 13:39:56	
Sprint Floor	R	B1	2 (Lawrence Livermore)	12/04/95 13:27:22	12/04/95 15:09:07	
Sprint Floor	R	B1	8 (Stennis)	12/04/95 13:27:33	12/07/95 13:39:56	
Sprint Floor	R	B2	All	12/07/95 13:09:29	12/08/95 11:08:57	Yes
Sprint Floor	R	B2	10	12/04/95 13:27:53	12/07/95 13:39:58	
Sprint Floor	R	B2	11	12/04/95 13:28:01	12/07/95 13:40:01	
Sprint Floor	R	B2	4 (ARL)	12/04/95 13:27:43	12/07/95 13:39:58	
Sprint Floor	R	B2	8 (NRL-DC)	12/04/95 13:27:50	12/07/95 13:40:01	
Sprint Floor	R	C2	All	12/07/95 14:49:45	12/07/95 18:04:28	
ARL Floor	T	A1, A2, A3, A4, B3, B4, D1, D2		12/07/95 11:46	12/07/95 18:17	
ARL Floor	R	A1, A2, A3, A4, B3, B4, D1, D2		12/07/95 11:54	12/07/95 18:17	
Phillips	T	B1		12/04/95 14:46:22	12/06/95 20:55:22	Yes
CEWES	T	B1		12/04/95 14:46:05	12/08/95 11:03:35	
CEWES	R	B1	8	12/04/95 11:11:10	12/07/95 11:58:13	
CEWES	R	B1	All	12/07/95 11:58:43	12/08/95 11:03:36	
Stennis	T	B1		12/04/95 14:46:42	12/08/95 11:03:36	
Stennis	R	B1	12	12/05/95 09:05:26	12:06/95 17:10:58	
Stennis	R	B1	All	12/07/95 11:58:50	12/08/95 11:03:43	
NOSC	T	B1		12/04/95 14:46:05	12/08/95 11:03:42	Yes
NOSC	R	B1	2	12/04/95 11:11:03	12/04/95 17:37:52	
NOSC	R	B1	All	12/07/95 11:58:40	12/08/95 11:03:37	Yes
NRL	T	B1		12/05/95 15:18:13	12/08/95 11:08:59	Yes
NRL	R	B1	12	12/04/95 12:59:47	12/07/95 13:39:57	Yes
NRL	R	B1	All	12/07/95 13:12:05	12/08/95 11:09:12	
ARL	T	A1, C3, B3, B4 C1, C2		12/06/95 16:21:47	12/06/95 17:42:11	
ARL	T	B1		12/04/95 16:17:10	12/08/95 11:09:09	
ARL	R	A1	0	12/06/95 16:22:05	12/06/95 17:41:48	
ARL	R	A1	3	12/06/95 16:21:44	12/06/95 17:41:52	
ARL	R	B1	All	12/07/95 13:13:19	12/08/95 11:08:56	
ARL	R	A3	0	12/06/95 16:22:04	12/06/95 17:41:59	
ARL	R	B3	0	12/06/95 16:22:00	12/06/95 17:41:48	
ARL	R	B4	0	12/06/95 16:22:14	12/06/95 17:41:48	

Table 15: AAI SC'95 measurement inventory

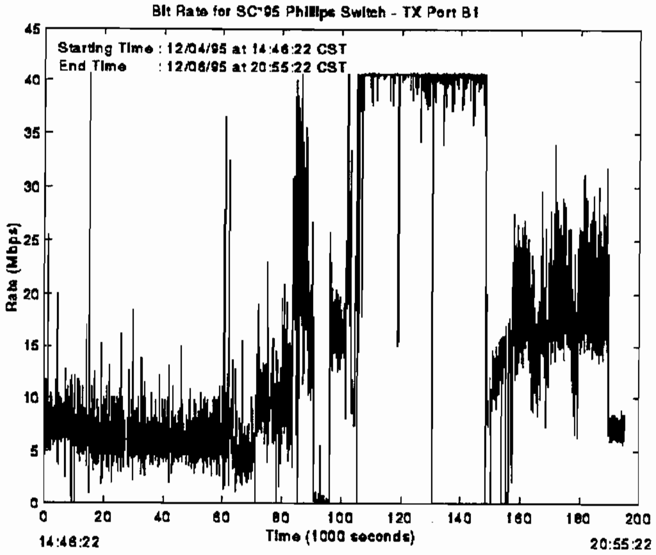


Figure 19a: Example of SC'95 traffic flows

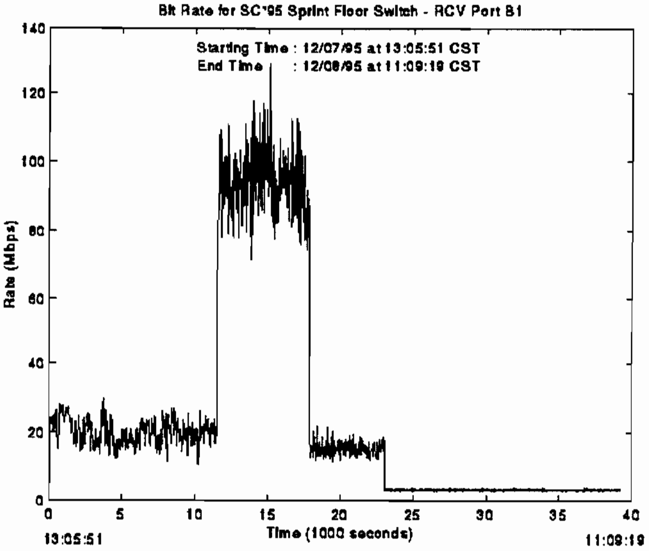


Figure 19b: Example of SC'95 traffic flows

2.4 AAI Simulation

Predicting the throughput of high-speed wide-area ATM networks (WANs) is a difficult task. Simulation models can enable the systematic evaluation of these systems when mathematical models are not available and experiments on the actual systems are impossible or impractical. In this section, validation simulation models of TCP/IP over the ATM WANs, including the AAI, are presented. These simulation results indicate that network congestion effects can be modeled and simulation models can predict the performance of complex high-speed ATM wide-area networks.

2.4.1 Developed Simulation Models for Components of the AAI

2.4.1.1 TCP

The simulation software environment used for all our simulations is BONeS Designer [2]. It is a software package for modeling and simulating event-driven systems. A system model can be constructed hierarchically and graphically using building blocks from the BONeS model library, or using primitives written in C or C++. A TCP BONeS primitive module was created for this study. The source code for this primitive was based on the MIT Network Simulator (NetSim) TCP module [3]. However, using the NetSim module, we were unable to match measurement with simulation results in the presence of network congestion because specific TCP timer mechanisms were not modeled. The NetSim TCP module is partially based on the Berkeley Standard Distribution (BSD) 4.3 Tahoe version. All TCP implementations that are 4.3 BSD-based include two timer functions: one is called every 200 ms (the fast timer) and the other every 500 ms (the slow timer) [4]. The fast timer is used with the delayed ACK timer, and the slow timer is mainly used with the retransmission timer. The NetSim TCP model did not include these timers and thus cannot accurately predict the performance of TCP over ATM under congestion. The modified TCP model developed here is based on the 4.3 BSD Reno version. It supports the following major mechanisms:

- fast and slow timers;
- slow start and congestion avoidance;
- fast retransmit and fast recovery; and
- window advertisement.

A simulation model of a host using TCP over ATM was developed. The TCP model generates the data buffers for transmission, and the ATM block accepts TCP packets and performs the SAR function to generate ATM cells. It also accepts cells from the network and reconstructs the TCP packets. An IP module is not used. IP provides an unreliable, connectionless datagram delivery service between hosts attached to a TCP/IP internetwork. In this study, the connectionless IP datagrams are carried only on connection-oriented ATM networks. Hence, the IP routing functionality is redundant.

2.4.1.2 Cell-Level Pacing

Cell-level pacing is the mechanism that reduces the source cell transmission rate. In early studies on the MAGIC network such pacing was essential for achieving adequate throughput. The Pacer functionality is modeled here by an infinite size FIFO queue and a server with service rate equal to the pacing rate.

2.4.1.3 ATM Layer Model

For this study, the host interfaces use the ATM adaptation layer AAL5 for mapping packets to ATM cell streams. The AAL5 Segmentation and Reassembly Processing is modeled by adding the time required for the AAL5 SAR sublayer to map the packets to ATM cell streams or to reconstruct cell streams to IP datagrams.

2.4.1.4 Basic ATM Switch Model

ATM switches were modeled as nonblocking output buffered systems.

2.4.1.5 SONET Link Model

The SONET links were modeled by reducing the link speed to account for the SONET overhead.

2.4.1.6 Traffic Model

To measure the end-to-end throughput at the TCP layer, a public domain software tool, ttcp, was used. This tool transfers TCP packets from local memory to memory on a remote host as fast as the operating system, interfaces, and network allow. In order to model the ttcp traffic, the data send buffer at the transmitting host was always kept full.

2.4.2 Validation of AAI Simulation Models

2.4.2.1 System Parameters

The common system parameters used in these simulations are listed in Table 16.

System Parameter	Value
TCP MTU Size	9180 Bytes
TCP Processing and OS Overhead Time	200-300 mu sec
TCP User Send Buffer Size	64 KBytes
Slow-Timer Period	0.5 s
Fast-Timer Period	0.2 s
Minimum RTO	1.0 s
AAL5 Segmentation and Reassembly Processing Time	0.2 mu sec
AAL5 Cell Payload Size	48 Bytes
Switch Processing Time	4 mu sec
Switch Output Buffer Size per VC	256 Cells
OC-3c Link Speed	155 Mb/s
TAXI Link Speed	100 Mb/s
DS3 Link Speed	45 Mb/s

Table 16: Simulation system parameters

The maximum size of the TCP segment is specified by the TCP MTU (maximum transmission unit) size parameter. TCP Processing and OS Overhead Time is the overall time needed by the TCP software to create a segment for transmission or process an incoming segment and for the operating system to handle all system calls and I/O operations during transmission or reception of a TCP segment. The TCP User module sends data buffers to the TCP module for transmission. The size of these data buffers is designated by the TCP User Send Buffer Size parameter. The retransmission timer is decremented every 0.5 seconds (Slow-Timer Period), and only when the timer reaches 0 is a retransmission performed. A delayed ACK is sent every time the 0.2-second delayed ACK timer (Fast-Timer) expires. The retransmission timer is bounded by TCP to be between 1 (Minimum Retransmission Time-Out) and 64 seconds [4]. For this study, the host interfaces use the ATM adaptation layer AAL5 for mapping packets to ATM cell streams. The AAL5 Segmentation and Reassembly Processing Time is the time required for the AAL5 SAR sublayer to map the packets to ATM cell streams or to reconstruct cell streams to packets. The values of the TCP Processing and OS Overhead Time, AAL5 Segmentation and Reassembly Processing Time, and Switch Processing Time parameters are based on experimental measurements [5, 6].

2.4.2.2 Predicting the Effect of Rate Mismatches

An experiment was conducted by Ewy and Evans [7] in order to study the performance of TCP over ATM under the case of rate mismatch. The configuration for this experiment is shown in Figure 20.

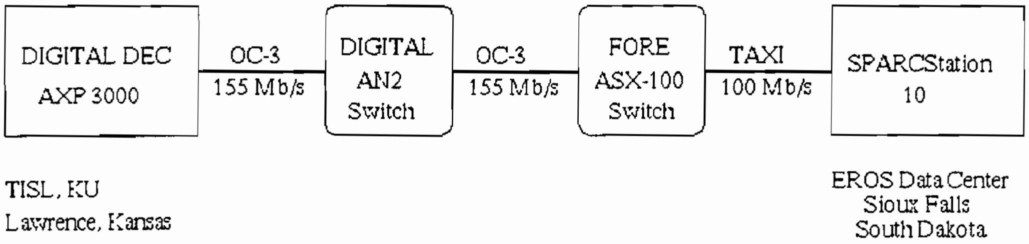


Figure 20: Experimental configuration for TCP/ATM performance under rate mismatch

A single host transmits to another host with a 155 Mb/s-to-100 Mb/s bandwidth constriction in the path. The simulation system used for this experiment is shown in Figure 21.

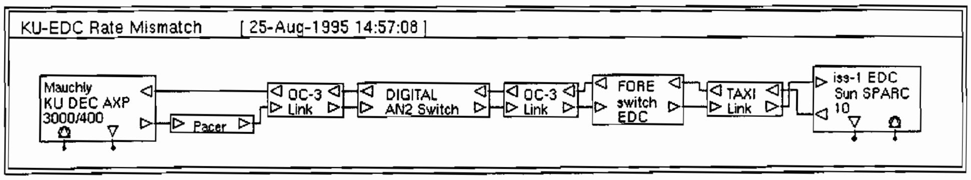


Figure 21: Simulation model for TCP/ATM performance under rate mismatch conditions

The TCP window size is set to 128 KB and the cell-level pacing is fixed at a rate of 70 Mb/s. Cell-level pacing is the mechanism that reduces the source cell transmission rate.

The results are shown in Table 17. This confirms that ATM WAN congestion effects can be accurately modeled.

	No Pacing	Pacing
Experimental Results	1.20 Mb/s	68.20 Mb/s
Simulation Results	1.16 Mb/s	68.60 Mb/s

Table 17: Throughput for TCP/ATM with rate mismatch

2.4.2.3 Predicting the Performance of the AAI

A simulation model was created for predicting the performance of the AAI Network. The simulation results over a single connection were then validated by measurement. The experimental configuration is shown in Figure 22.

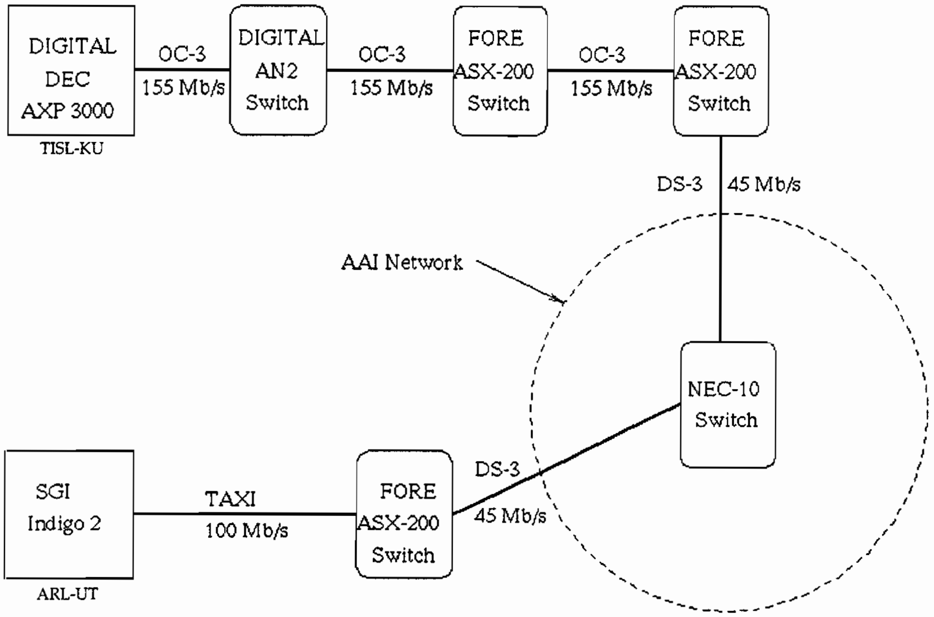


Figure 22: Experimental setup of a single connection in the AAI network

In this experiment, a Digital DEC 3000 AXP with an OTTO OC-3 interface transmits to an SGI Indigo 2 host with a 100 Mb/s TAXI interface. The round-trip time for this connection is about 36 ms. The simulation model for this experiment is shown in Figure 23.

Note in this system that the lowest link capacity is DS3. Hence, a cell-level pacer is used with a rate of 40 Mb/s since the maximum rate of a DS3 link after excluding the physical layer overhead is 40.7 Mb/s. We used 40 Mb/s at the pacer to avoid any possibility of cell loss due to network congestion. A TCP window size of 256 KB was used. The results are shown in Table 18. Once more, the results show that simulations can accurately predict the performance of complex high-speed ATM wide-area networks.

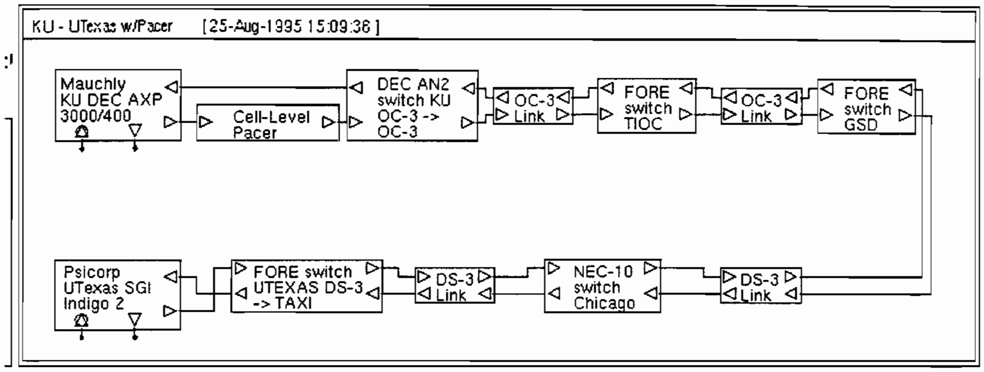

KU - UTexas w/Pacer [25-Aug-1995 15:09:36]
Mauchly KU DEC AXP 3000/400
Cell-Level Pacer
DEC AN2 switch KU OC-3 -> OC-3
OC-3 Link
FORE switch TIOC
OC-3 Link
FORE switch GSD
Psicorp UTexas SGI Indigo 2
FORE switch UTEXAS DS-3 -> TAXI
DS-3 Link
NEC-10 switch Chicago
DS-3 Link

Figure 23: Simulation model of a single connection in the AAI network

Experimental Results	34.2 Mb/s
Simulation Results	34.3 Mb/s

Table 18: Throughput of TCP over ATM using AAI network

2.4.3 Lessons Learned

TCP imposes heavy processing overhead when individual packets are retransmitted immediately after they time out and are acknowledged. To reduce this overhead, many TCP implementations use the slow and fast timer mechanisms to handle the retransmission and acknowledgment operations. However, the cost of doing that is a performance degradation. The long delay introduced by these timer mechanisms causes a significant reduction in the performance of TCP over ATM when there are cell losses due to network congestion. All workstations used for this study employ TCP implementations that use these timers. Therefore, it was necessary to include them in our TCP model in order to match our simulation results with measurements. For example, in our simulation studies, we first used the NetSim TCP model, which does not use these timers. The simulation result (without modeling these timers) for the experiment with no pacing was 24 Mb/s, which is 20 times greater than the experimental result. Our results indicate that simulations can be used to predict the performance of high-speed wide-area networks. However, to enable feasible simulations of such systems, the minimum level of model fidelity for each system element must be used. The simulation run time is often inversely proportional to the level of model fidelity. Hence, simulation run time can be reduced significantly if unnecessary model precision is avoided. Table 19 shows the level of model fidelity used in our simulation systems.

Detailed IP and SONET models are not required in our simulation systems. We avoided the need for a SONET model at the physical layer by reducing the OC-3 link speed from 155 Mb/s to 149 Mb/s. An IP model is not used because the IP routing functionality is not needed in ATM networks. Here, the ATM switch is modeled only by a single FIFO queue and a server. High precision for this switch model is unnecessary since a single connection with no cross-traffic was considered. Also, the details of the pacing algorithm were not modeled, since a simple FIFO was sufficient to capture its impact on performance. To obtain the results presented in this study, a high level of fidelity was needed for the TCP. By using the level of model fidelity shown below, we were able to reduce the simulation time significantly. The run time of the simulation system shown in Figure 21 on a SPARCstation-10 with 120 Mbyte of RAM is about 20 to 30 minutes for each second of real time. If accurate IP, SONET, and switch models were used, the simulation of these high-speed wide-area ATM networks would be difficult. The results presented here demonstrate that simulation can be used to capture and evaluate the interactions of ATM control with TCP packet flow and congestion control mechanisms. Additional information on this simulation study can be found in [7].

Model	Level of Fidelity
ATM	High
ATM Switch	Medium
IP	Low
Link	Medium
Pacer	Medium
SONET	Low
TCP	High
TCP User (Application Layer)	Low

Table 19: Level of model fidelity

2.5 Congestion Control

2.5.1 Initial ATM WAN Flow-Control Experiment

Even though congestion control will be a major focus of Year 2 research, to get initial experience with ATM WAN control issues, some preliminary experiments were conducted. Note that congestion control mechanisms, e.g. rate-based controls, are not currently implemented in most of the hosts and switches in AAI. However, the Digital AN2 switch and Digital NICs have credit-based control implemented. It was this functionality that was evaluated over the ATM WAN.

Credit-based congestion control consists of a sliding window algorithm. In order to transmit a cell on a given VC, an upstream node must have credits available. Periodically, a downstream node sends credit updates to the upstream node, indicating the availability of buffer space for receiving data on a VC [5]. The credit count is decremented for each transmitted cell. This procedure is followed independently for each link, and hence we sometimes refer to this algorithm as link-by-link congestion control. Once a node runs out of credits, it is no longer allowed to transmit any cells. In this fashion, when congestion occurs, cells will not be acknowledged and upstream nodes will eventually run out of credits.

In testing the performance of credit-based congestion control, two DEC 3000 workstations (Hosts A & B) located in the Telecommunications and Information Sciences Laboratory (TISL) at KU were used. They both access a DEC AN2 local-area ATM switch at OC-3 rates (155.52 Mbps). The third host (Host C) used here was a DEC 3000 workstation, located in the Sprint Technology Integration and Operation Center (TIOC), Kansas City, at a distance of approximately 100 km from the switch, and connected to it at OC-3 rates. Note that, although the nominal OC-3 rate is 155.52 Mbps, the theoretical maximum achievable TCP layer bandwidth is 135.102 Mbps. Experimental results for throughput may be even lower due to other processes running on the hosts (e.g., daemons). In all of these experiments, Host A is the receiving host, and B and C are transmitting hosts. A set of experiments was conducted to determine the appropriate number of credits to assign to each VC; here 319 credits/VC were used.

Tables 20, 21, and 22 summarize the results of a set of experiments testing the performance of credit-based congestion control. All individual throughput values presented in the tables are averages taken over a repetition of ten experiments. From Table 20, we can observe that when two courteous traffic streams compete for the same link, the aggregate throughput without congestion control is almost 15% lower than that with congestion control. The reason for this is that while TCP's backoff mechanism prevents severe throughput degradation, it cannot prevent cell losses from happening at the switch. From Table 21, it's found that without congestion control, neither of the two UDP traffic streams can get good throughput, and the aggregate throughput is only 9.8 Mbps at the receiver side; with congestion control, although there is still some cell loss in the transmitting hosts due to the non-courteous nature of UDP, the aggregate throughput is much better. From Table 22, it's found that without congestion control, UDP takes hold of most of the bandwidth, causing an unfair situation; with congestion control, fairness is achieved and aggregate throughput is also improved.

The results of these experiments are in preparation for the congestion control research remaining for Year 2. (See [8] for additional details on the congestion control study.)

	Congestion Control		No Congestion Control	
	Transmitter	Receiver	Transmitter	Receiver
C to A (TCP)	58.5 Mbps	58.5 Mbps	57.1 Mbps	57.0 Mbps
B to A (TCP)	56.2 Mbps	56.3 Mbps	40.4 Mbps	40.5 Mbps
Aggregate Throughput		114.8 Mbps		97.5 Mbps

Table 20: Individual and aggregate throughput for two contending TCP traffic streams

	Congestion Control		No Congestion Control	
	Transmitter	Receiver	Transmitter	Receiver
C to A (UDP)	68.6 Mbps	44.5 Mbps	129.5 Mbps	4.1 Mbps
B to A (UDP)	67.4 Mbps	44.2 Mbps	127.6 Mbps	5.7 Mbps
Aggregate Throughput		88.7 Mbps		9.8 Mbps

Table 21: Individual and aggregate throughput for two contending UDP traffic streams

	Congestion Control		No Congestion Control	
	Transmitter	Receiver	Transmitter	Receiver
C to A (TCP)		55.7 Mbps		9.5 Mbps
B to A (UDP)	69.7 Mbps	41.2 Mbps	120.4	78.5 Mbps
Aggregate Throughput		96.9 Mbps		88.0 Mbps

Table 22: Individual and aggregate throughput for a TCP stream flowing from C to A and a UDP stream from B to A

2.5.2 Initial Framework for the Implementation of End-to-End Rate-Based Congestion Control Within AAI

Currently, the AAI network does not have any ATM-layer congestion control mechanisms. The possibility of implementing or emulating a rate-based congestion control scheme as defined by the ATM Forum Traffic Management Specification within AAI has been explored. One approach to emulating the rate-based congestion control scheme is to employ a workstation as the controlling element associated with the bottleneck ATM switch. The workstation directly connects to one port of the ATM switch. Each ABR source contending for the same output port of that switch should establish a signaling VC with the controlling workstation. Before starting or terminating data cell transmission, an ABR source sends a resource management (RM) cell to the controlling workstation through the signaling channel, so that the controlling workstation always keeps track of the number of active ABR sources. The controlling workstation, in turn, computes a fair share of bandwidth among all active ABR VCs, and sends RM cells back to each ABR source telling the transmission rate allowed by the network. Each ABR source, upon receiving an RM cell back from the network, would immediately adjust its rate according to what is required by the network. If ATM-layer congestion control mechanisms continue to be absent from the AAI, implementation details of this approach will be explored in Year 2.

3 Major Technical Challenges

3.1 Network Performance Measurements and Troubleshooting

3.1.1 Dealing With a Large Heterogeneous Network

Conducting experiments over a national-scale network with a variety of host and switch types presents significant technical challenges. A global view and experience are required to correlate and analyze performance observations. The unique properties of a wide array of operating systems, host NICs, and switches had to be taken into account to draw correct conclusions.

3.1.2 Monitoring Tools Lacking

When this research effort started, there were no network-wide performance analysis tools that were suitable for the evaluation of national-scale ATM networks, like the AAI. Troubleshooting the throughput bottleneck on AAI required the use of hardware protocol analyzers, switch SNMP software, and specified NIC code, as well as TCP-level tools. Coordinating the use of such an array of measurement probes is a challenge.

3.1.3 Standard Software Was Not Designed for High Bandwidth-Delay Product Networks

Few of the host operating systems on AAI were able to deal with high bandwidth-delay product networks. Even though all the current hosts on AAI use some flavor of UNIX, each version of the networking software needed to be modified to get adequate performance over the AAI.

3.2 Network Measurement Tools

3.2.1 Network benchmarking

Unlike computers, there are no standard benchmarks for networks. General requirements, specification, and development of a new tool capable of supporting such benchmarking were required to meet this challenge. A number of issues exist, such as synchronization of traffic streams across possible wide-area networks, and operating system limitations such as interrupt rates. NetSpec software architecture for scalability and expandability needed to be addressed. Measurement software, NetSpec, was designed to have the potential to grow into a true network benchmarking tool.

3.2.2 Multilayer measurements require changes to operating system kernels

Observing network flows both within the protocol stack and across the network is desirable. To make measurements within the protocol stack requires modifications to operating system kernels. Such modification present technical challenges, including:

- obtaining source licenses;
- identifying correct monitoring points; and
- minimizing instrumentation effects.

3.2.3 Need to support multiple operating systems

For network-wide tools to be of ubiquitous utility, they must work on a variety of hosts with different Operating Systems. These operating systems have differing network protocol implementations as well as platform-dependent low-level support functions. This presents obvious development and maintenance challenges, including:

- application level: NetSpec written for portability; platform specific aspects were minimized but still non-trivial;
- NIC driver interface variations;
- The data stream driver we have implemented under OSF provides a uniform interface for the collection of performance data from all levels of the operating system that have been instrumented using its facilities. This approach is simpler than that commonly used, which provides a separate interface, and application level tool, for each data collection point.

3.3 AAI Simulation

Simulation of large-scale ATM networks presents challenges in several dimensions. An understanding of the specific protocol implementation and system parameters is needed before the models can be used to predict network performance. Given this understanding, appropriate model fidelity must be used to limit the execution time of the models. The general simulation of large-scale ATM networks is still an open problem.

3.4 Congestion Control

It will be a significant technical challenge to emulate end-to-end rate-based congestion control in a network where such functionality is not supported in the hosts and switches.

4 Lessons Learned

4.1 Network Performance Measurements and Troubleshooting

It is possible to deliver high TCP throughput (34 Mb/s for ATM-DS3) over an ATM WAN. The technical challenges to obtain this performance can be overcome. As the lessons from AAI and similar efforts become known, ATM WAN technology will mature, and such throughputs will become commonplace. To achieve this level of performance, some form of pacing or flow control is needed for systems with rate mismatches and small switch buffers. ATM host software (e.g., Fore device drivers) is still somewhat immature and may have significant bugs, leading to lost cells and lower performance. The use of IP classless interdomain routing (CIDR) (supernetting, i.e. building an IP subnetwork using Class C blocks) is still immature. This had implications both for network management (a large number of host routes must be set up) and for network performance (communication between the Class C subnets that form the CIDR block may use small default MTUs unless configured carefully, leading to more memory copies and lower throughput). The use of multi-homed interfaces exacerbates this problem. **A stable network configuration with switched virtual circuits will become essential for the evaluation of large-scale ATM WANs.**

4.2 TCP/IP over ATM: Summary of Lessons Learned

Many performance tests of TCP/IP over ATM WANs have been executed across the MAGIC and AAI testbed networks over the past three years. Below is a short summary of the results of these tests and the lessons learned, based on our comprehensive experience with TCP/IP over ATM.

With DEC 3000/400 Alphas, now several years old, as well as newer 3000/600 machines, we have been able to fill OC-3c links (134 Mb/s at the application level) at distances of 1000 km. We were also able to fill an ATM DS3 at even greater distances (Maryland/Kansas) with these systems. The throughput was measured with ttcp and NetSpec. The DEC machines are very WAN-capable.

With special debug tools that perform direct reads/writes to the device (no TCP/IP), we can fill an OC-3c link using a DEC 5000/240 (old technology).

Approximately 110 Mb/s has been measured between dual-processor SPARC-20s running Solaris 2.4 with the TCP long windows patch in the lab. About 89 Mb/s between SPARC-20s running Solaris 2.4 with the TCP long windows patch has been observed, even over distances of 1000 km. These values have been obtained with ttcp and NetSpec.

With Pentiums (P100) running Linux, TCP/IP applications get about 50 Mb/s of throughput. With custom software developed at KU under a separate effort, KU researchers have obtained full OC-3c rates out of these systems.

Key factors to obtaining high WAN throughput include:

- TCP long windows support;
- avoiding or working around bandwidth mismatches that lead to cell loss;
- sending long PDUs; and
- lots of memory bandwidth.

4.3 Network Measurement Tools

For large networks, multiple network element measurement tools like NetSpec are required to get a system-wide perspective of network performance. However, weaknesses in the existing version of NetSpec have been identified. Measurement at multiple sites using a variety of tools is difficult but is possible with proper coordination. Effort was devoted to the development of WWW-based point measurements. These tools were not as useful as expected; it was difficult to obtain the proper context and then visualize the results. Continuous monitoring with a visualization capability was thus needed. The WWW is the right platform for these kinds of functions.

4.4 AAI Network Measurements

Standard software (e.g., applications such as FTP) was not designed for high bandwidth-delay product networks, and thus modification was required to get adequate throughput. It was demonstrated that with proper modification, applications like FTP can take full advantage of ATM WANs like the AAI. Vendors should be encouraged to add this sort of adaptability. SNMP can be used to gather useful traffic statistics from ATM switches, albeit with relatively coarse time granularity. Switch MIBs may provide inaccurate results. In particular the cell reject counters in the Fore MIB were found to be unreliable. The NetSpec measurement demon, which will be implemented in version 3.0, was created in part to provide access to finer grain data than is commonly available through SNMP. Since the measurement demon resides on the target system, it can use local APIs to obtain data. However, this clearly does not help for information available only through SNMP, and we should thus also urge manufacturers to provide finer grain performance data where appropriate.

4.5 AAI Simulation

Rarely are simulation performance predictions validated with measurements. This research has shown, for the cases considered, that it is possible to validate network simulation models with measurements. Refer to Section 2.4.3 for details. The simulation-based performance prediction tool developed here will provide the basis for the evaluation of congestion and call admission control techniques in the AAI.

4.6 Congestion Control

Early experimental results indicate that a credit-based congestion control scheme works effectively in minimizing aggregate throughput degradation during congestion and ensuring a fair allocation of the available bandwidth in both wide area and local area. These results provide a baseline for the congestion control research to be conducted in Year 2.

5 Ongoing Problems

5.1 Connectivity

The lack of a stable network caused problems for both troubleshooting the systems and obtaining performance measurements. Even with the recent use of UNI on AAI, this is still a problem. NetSpec experiments are currently possible between just a few sites, and SNMP statistics gathering from all switches is not yet possible.

5.2 Access to remote hosts

Access to remote resources to accomplish the measurement objectives is an on-going problem. Accounts on remote machines and cooperation in installing software are required for both network troubleshooting and performance testing.

5.3 Multiple organizations

Coordination of activities across multiple organizations is difficult. As the AAI is used for more applications, such coordination will become more important, so that the captured traffic flows can be correlated with specific applications.

5.4 Access to driver code

NIC-level performance tools have been developed in cases where there is access to the appropriate driver software. Such access is lacking for the FORE driver, thus limiting the ability to collect NIC-level performance measurements, including access to OS source for protocol stack for instrumentation.

5.5 Lack of OC-3c-level performance from SPARC-20-class machines

As AAI moves to OC-3 access, users should be aware that most SPARC-20-class machines cannot fully utilize the transmission facilities.

6 Remaining Issues for Year 2

6.1 Analyze the SC'95 data

The fine- and course-scale traffic flow data collected from SC'95 needs to be studied.

6.2 Complete next version of NetSpec

The versatility and generality of NetSpec will be enhanced, e.g., support for multicast, measuring performance data from several levels of a connection beyond the application layer, and describing experiments whose elements are not just simple connections will be addressed. These changes will require significant extensions to the description language and the structure of NetSpec.

6.3 Analyze the EDC data

Continuous monitoring of EDC data is underway; this data requires analysis.

6.4 Complete AAI-wide measurements using a combination of ARL MBONE machines and AAI OC-3-capable hosts

A plan has been proposed to obtain access to the ARL MBONE hosts and to acquire OC-3-capable hosts to support the evaluation of the AAI. This plan remains to be executed, as defined by the AAI performance testing scenarios (see Section 2.3.1).

6.5 Analyze the continuous flow of switch statistics

Continuous monitoring of traffic flows from other AAI sites is underway. This data requires analysis. Of special interest will be the study of the flows generated by the distributed processing application coming online on the AAI from Stennis.

7 Year 2 Research Agenda

7.1 Network Performance Evaluation

KU will continue to be responsible for measuring the performance of the AAI. As part of this effort, we will continue to identify system bottlenecks as well as suggest solutions to performance degradation problems. This work will be needed as AAI transitions from DS3 to OC-3. The infrastructure to support the performance evaluation will include the use of ARL MBONE machines for sites remaining on DS3, as well as the ARL-UT and EDC performance hosts deployed during Year 1. The current plan is to acquire four Digital Alpha workstations with ATM interfaces to be placed at the first AAI sites to be connected by OC-3 (ARL, CEWES, and NRaD) and at KU. The Digital Alpha workstations would be suitable for stressing the OC-3 components of the AAI. NetSpec will be used to determine many aspects of network performance, including whether QoS guarantees are maintained under statistical multiplexing in an ATM WAN. KU will continue to develop and refine techniques for measuring and predicting the performance of the AAI. We will continue the cycle of measurements, tuning the network and the prediction capability, followed by more measurements and predictions. The predictive capability will be based on the ATM network simulation models constructed during Year 1.

7.2 Data Analysis and Development of Application Profiles

The data collected from SC'95 and EDC in Year 1, as well as additional measurements from other AAI sites collected throughout Year 2, will be used to quantify network utilization as well as develop profiles of the traffic flows generated by various applications on the AAI. A major activity in Year 2 will be the collection of traffic profiles. In Year 2 we begin to evaluate the suitability of standard, as well as evolving, traffic modeling techniques to characterize these flows. Examples of such models include self-similar models, discrete spectral analysis of the rate correlation function, and transform expand sampling. Traffic modeling will continue into Year 3. A significant contribution to the high-speed networking community will be the measurement of traffic characteristics, such as those that will be obtained from the AAI, and their comparison to proposed models.

7.3 Implementation and Evaluation of a Congestion Control Framework

In Year 2 a framework for the implementation within AAI of a congestion control algorithm will be developed. KU will be responsible for the evaluation of these congestion control techniques for the AAI WAN using the measurement capabilities described above. This evaluation will continue into Year 3. Several congestion control techniques may be considered, e.g., TCP level pacing, i-out-of-m cell pacing, link-by-link flow control, and scheduling mechanisms for guaranteed bandwidth allocation. Thus, both traffic shaping and flow control techniques will be evaluated. This task will address two major questions: 1) how well existing techniques scale to large national-scale networks; and 2) how well subnetworks with different congestion control techniques can interoperate. A task time line for Year 2 activities is shown in Figure 24.

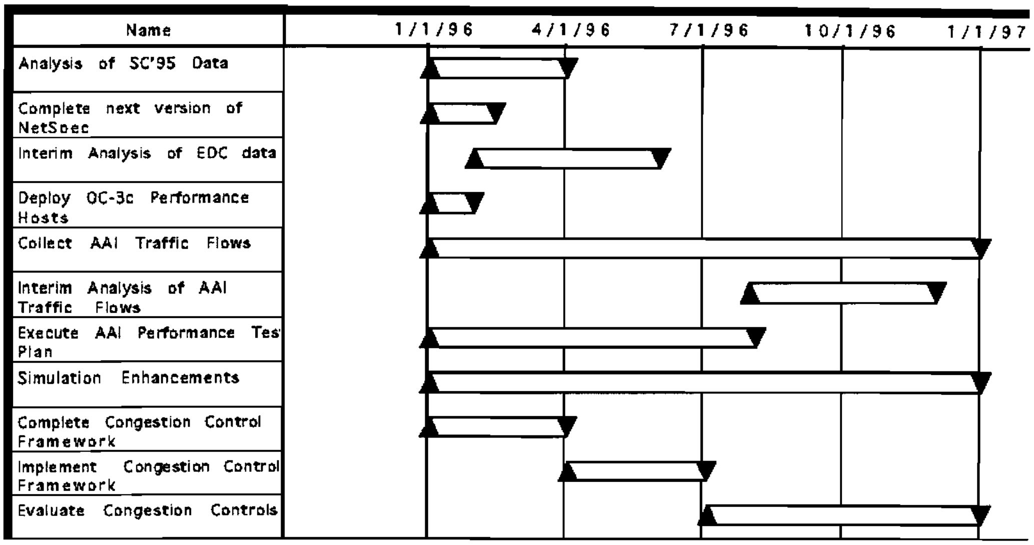


Figure 24: Year 2 task time line

7.4 A Preview of the Year 3 Research Agenda

The final year of the AAI research will focus on: 1) completion of the analysis of AAI traffic flow and the development of AAI-based traffic models; 2) the completion of the evaluation of congestion control techniques; and 3) evaluation of call admission control for the AAI. In Year 3, KU will be responsible for the evaluation of call admission control (CAC) on the AAI. We will use our measurement capabilities to determine the efficacy of existing CAC techniques, e.g., CAC techniques based on equivalent bandwidth. As part of this task, we will determine how well known CAC techniques scale to large ATM/IP internetworks and how well they perform in the face of unknown traffic profiles.

8 References

[1] "Performance and Traffic Measurement Scenarios for the AAI Network," Luiz DaSilva, Hongbo Zhu, Victor S. Frost, Joseph B. Evans, Douglas Niehaus, David W. Petr, TISL Technical Report 10980-3, March 1995.

[2] Systems & Networks, "BONeS DESIGNER 3.0 Modeling Guide," Lawrence, KS, 1995.

[3] D. Martin, "Network Simulator User's Manual," MIT, 1988.

[4] W.R. Stevens, "TCP/IP Illustrated," Volume 1, 2, Addison-Wesley, Reading, Massachusetts, 1994.

[5] T.E. Anderson, S.S. Owicki, C.P. Thacker, "High Speed Switch Scheduling for Local Area Network," DEC Internal Publication, 1993.

[6] H.Y. Chen, J.A. Hutchins, N. Testi, "TCP Performance over Wide Area Networks," SAND93-8243, September 1993.

[7] Georgios Y. Lazarou, Victor S. Frost, Joseph B. Evans, Douglas Niehaus, "Using Measurements to Validate Simulation Models of TCP/IP over High Speed ATM Wide Area Networks," submitted to ICC'96.

[8] Hongbo Zhu, Luiz A. DaSilva, Joseph B. Evans, Victor S. Frost, "Performance Evaluation of Congestion Control Mechanisms in ATM Networks," Computer Measurement Group Annual Conference (CMG'95), December 1995.

[9] Luiz A. DaSilva, Rick Lett, Victor S. Frost, "Performance Considerations In File Transfers Over Wide-Area ATM Networks," Telecommunications and Information Sciences Laboratory, TISL TR-10980-10, 1995.

9 Related Publications

Georgios Y. Lazarou, Victor S. Frost, Joseph B. Evans, Douglas Niehaus, "Using Measurements to Validate Simulation Models of TCP/IP over High Speed ATM Wide Area Networks," accepted for ICC'96.

Kunyan Liu, Hongbo Zhu, David W. Petr, Victor S. Frost, Cameron Braun, William L. Edwards, "Design and Analysis of a Bandwidth Management Framework for ATM-Based Broadband ISDN," accepted for ICC'96.

Hongbo Zhu, Luis A. DaSilva, Joseph B. Evans, Victor S. Frost, "Performance Evaluation of Congestion Control Mechanisms in ATM Networks," Computer Measurement Group Annual Conference (CMG'95), December 1995.