Distributed Large Data-Object Environments: End-to-End Performance Analysis of High Speed Distributed Storage Systems in Wide Area ATM Networks[1]

Brian Tierney (bltierney@lbl.gov),
William Johnston(wejohnston@lbl.gov),
Jason Lee, Gary Hoo, and Mary Thompson

Imaging and Distributed Computing Group
(http://www-itg.lbl.gov)
Lawrence Berkeley National Laboratory
Berkeley, CA 94720

1. **This work is jointly supported by ARPA - CSTO, and by the U. S. Dept. of Energy, Energy Research Division, Mathematical, Information, and Computational Sciences office, under contract DE-AC03-76SF00098 with the University of California.**

Overview

- The Problem: Finding performance bottlenecks in large distributed environments
- Large "Data-Objects" Environment
- The Magic Testbed
- DPSS
- Monitoring Method: NetLogger/LogTracer
- Performance Analysis
- Conclusions

Motivation and Approach

- **When building high-speed network-based distributed services, we often observe unexpectedly low network throughput and/or high latency - the reasons for which are usually not obvious**

  The bottlenecks can (and have been) in any of the components:

  - **the applications**
  - **the operating systems**
  - **the device drivers, the network adapters on either the sending or receiving host (or both)**
  - **the network switches and routers, and so on**

Motivation and Approach

- **It is difficult to track down performance problem because of the complex interaction between the many distributed system components resulting in problems in one place being most apparent somewhere else.**
- **Tools such at *ttcp* and *netperf* are somewhat useful but don't model real distributed applications, which are complex, bursty, and have more than one connection in and/or out of a given host at one time**

Motivation and Approach

- We have developed a methodology and a set of tools for realistically monitoring the behavior of all elements of the application-to-application communications path to determine exactly what is happening within this complex system. We have:
  - Instrumented our applications to do timestamping and logging at every critical point throughout the data path
  - Modified some Unix network and OS monitoring tools to log "interesting" events using a common log format
  - Incorporated information from SNMP queries of the network components
- The goal is to identify what must be done to produce predictable, high-speed components that can be used as building blocks for high-performance applications, rather than having to "tune" the system from top-to-bottom.

Large Data-Object Environments

We are developing a strategy for using high-speed networks as enablers for storage systems whose components are distributed around wide area networks.

The goal is to dramatically increase the location independence for access to "large data-objects".

These objects are typically tens of MBytes to tens of Gbytes, and are used by a diverse and geographically distributed community.

Examples include:

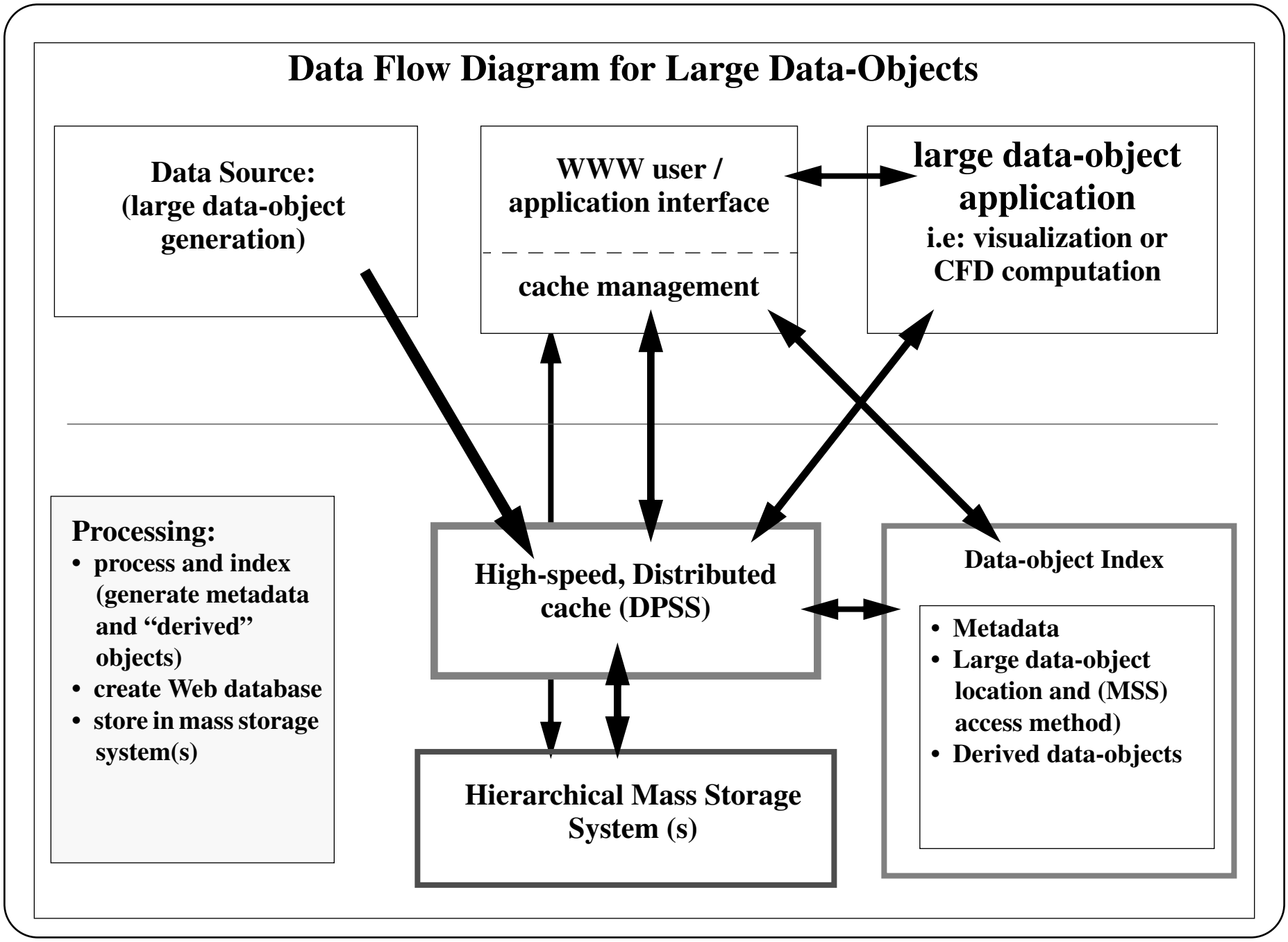
- **physics and nuclear science high energy particle accelerators and detector systems**
- **large electron microscopes**
- **ultra-high brilliance X-ray sources**
- **health care imaging data (MRI, CT, Video Angiography, etc.)**

Large Data-Object Environments

Elements of a distributed large data-object architecture:

- **data collection and the instrument-network interface**
- **on-line storage that is distributed throughout the network (for both performance and reliability)**
- **processing elements - also distributed throughout the network - for various sorts of data analysis**
- **data management that provides for the automatic cataloguing (metadata generation) of the data being stored**
- **data access interfaces, including application-data interfaces**
- **(transparent) tertiary storage ("mass storage") management**
- **user access to all relevant aspects (application, data, metadata, data management)**
- **transparent security that provides access control for all of the systems components based on the resource-owner's policy**

Large Data-Object Environments



The MAGIC Testbed

- This research done as part of our work in the DARPA-funded MAGIC Gigabit Network Testbed
- The MAGIC testbed consists of a large-scale, high-speed, ATM network. It is a heterogeneous collection of:
  - ATM switches and computing platforms
  - IP over ATM implementations
  - "middleware" (distributed services), etc.,
  - All tuned to enable applications to operate at high speed
- TerraVision (a virtual reality-like, terrain navigation application) and the Distributed-Parallel Storage System (which provides fast access to very large data sets) together are the model, high performance distributed application in MAGIC.

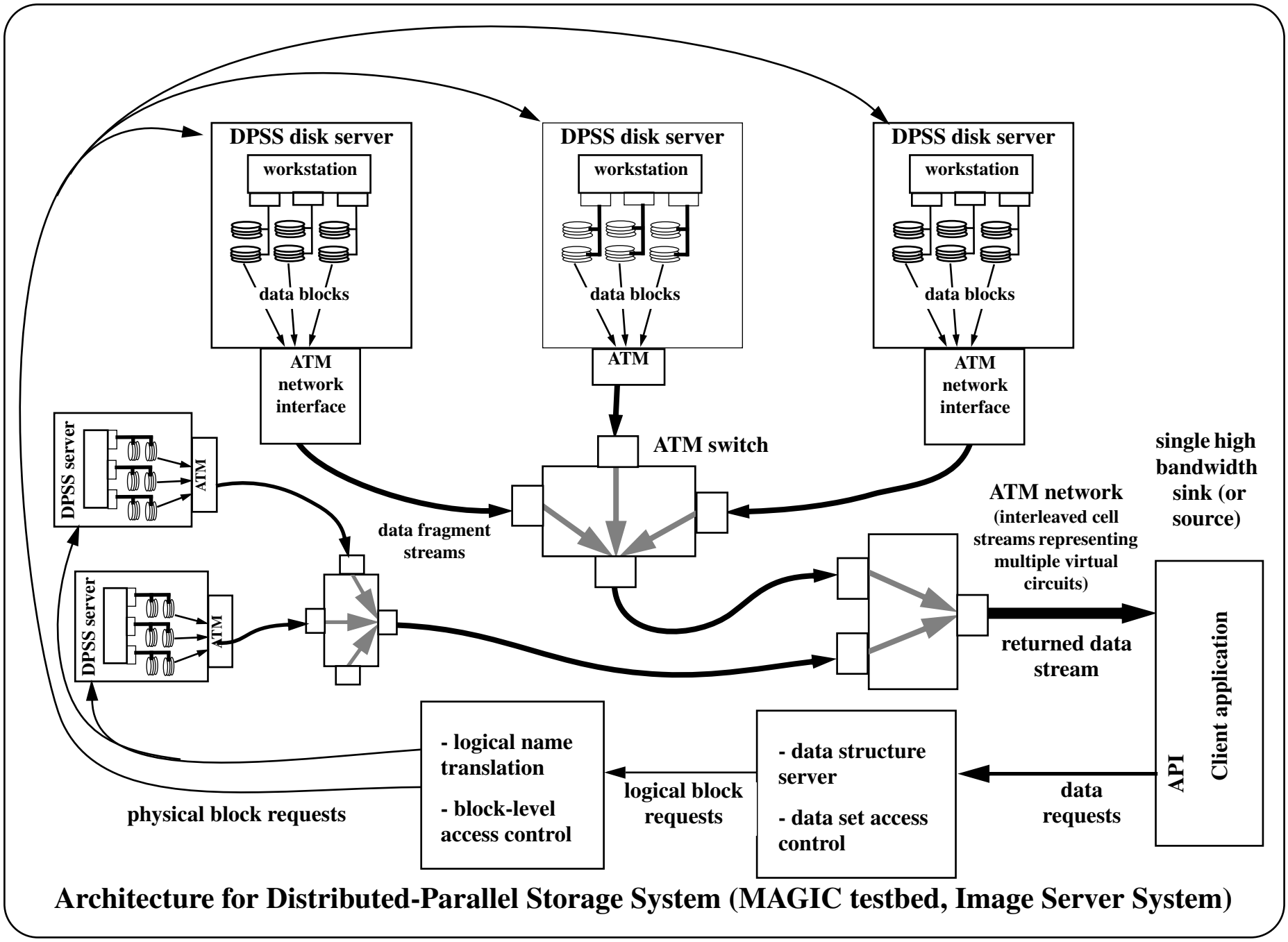
MAGIC Testbed

- **The performance monitoring work is a cooperative effort of the MAGIC participants:**
  - **Earth Resources Observation System Data Center, U.S. Geological Survey (EDC)**
  - **Lawrence Berkeley National Laboratory, U.S. Department of Energy (LBNL)**
  - **Minnesota Supercomputer Center, Inc. (MSCI)**
  - **SRI International (SRI)**
  - **University of Kansas (KU)**
  - **CNRI/MITRE Corporation (project coordination)**
  - **Sprint**
  - **U S WEST Communications, Inc.**

DPSS Architecture, Implementation, and Use

- The Distributed-Parallel Storage System (DPSS, aka ISS) is a dynamically configurable network-striped disk array designed to supply high speed data streams to other processes in the network
- DPSS uses parallel operation of distributed servers to supply image streams fast enough to enable various multi-user, "real-time", virtual reality-like applications in an Internet / ATM environment
- At the application level, the DPSS is a persistent cache of named objects, at the storage level it is a logical block server (It is not a reliable tertiary storage system.)
- DPSS supports both reading and writing

DPSS



Architecture for Distributed-Parallel Storage System (MAGIC testbed, Image Server System)

DPSS

- **Typical DPSS implementation**
  - **4 - 5 UNIX workstations (e.g. Sun SPARCStation, DEC Alpha, SGI, etc.)**
  - **4 - 6 fast-SCSI disks on multiple 2 - 3 SCSI host adaptors**
  - **an ATM network interface**

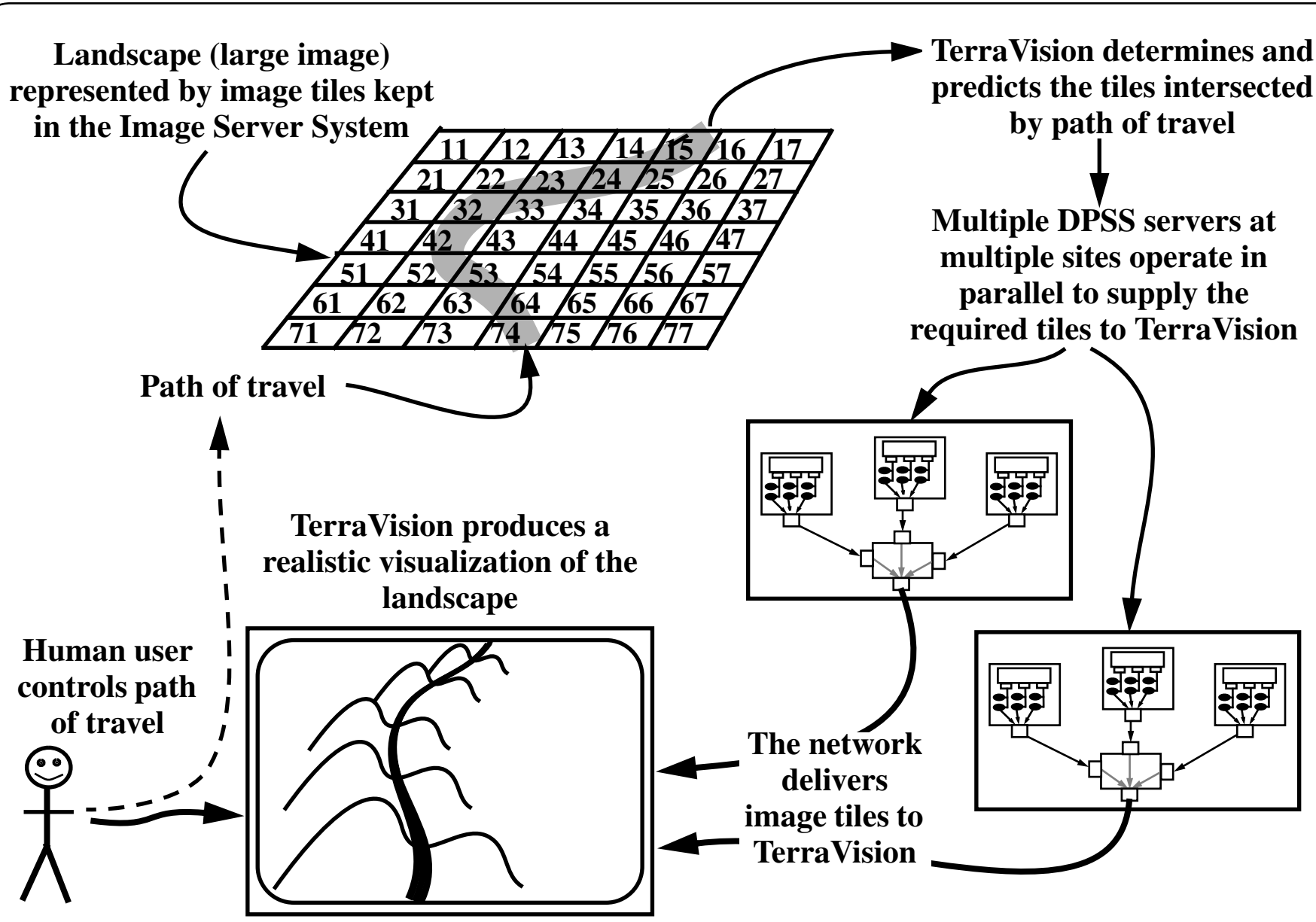
- **This configuration can deliver an aggregated data stream to an application at about 400 Mbits/s (50 MBy/s) using these relatively low-cost, "off the shelf" components by exploiting the parallelism provided by approximately five hosts, twenty disks, ten SCSI host adaptors, and five network interfaces.**

- **Transport is by either TCP or UDP/RTP, but performance testing is currently focused on TCP transport.**

DPSS

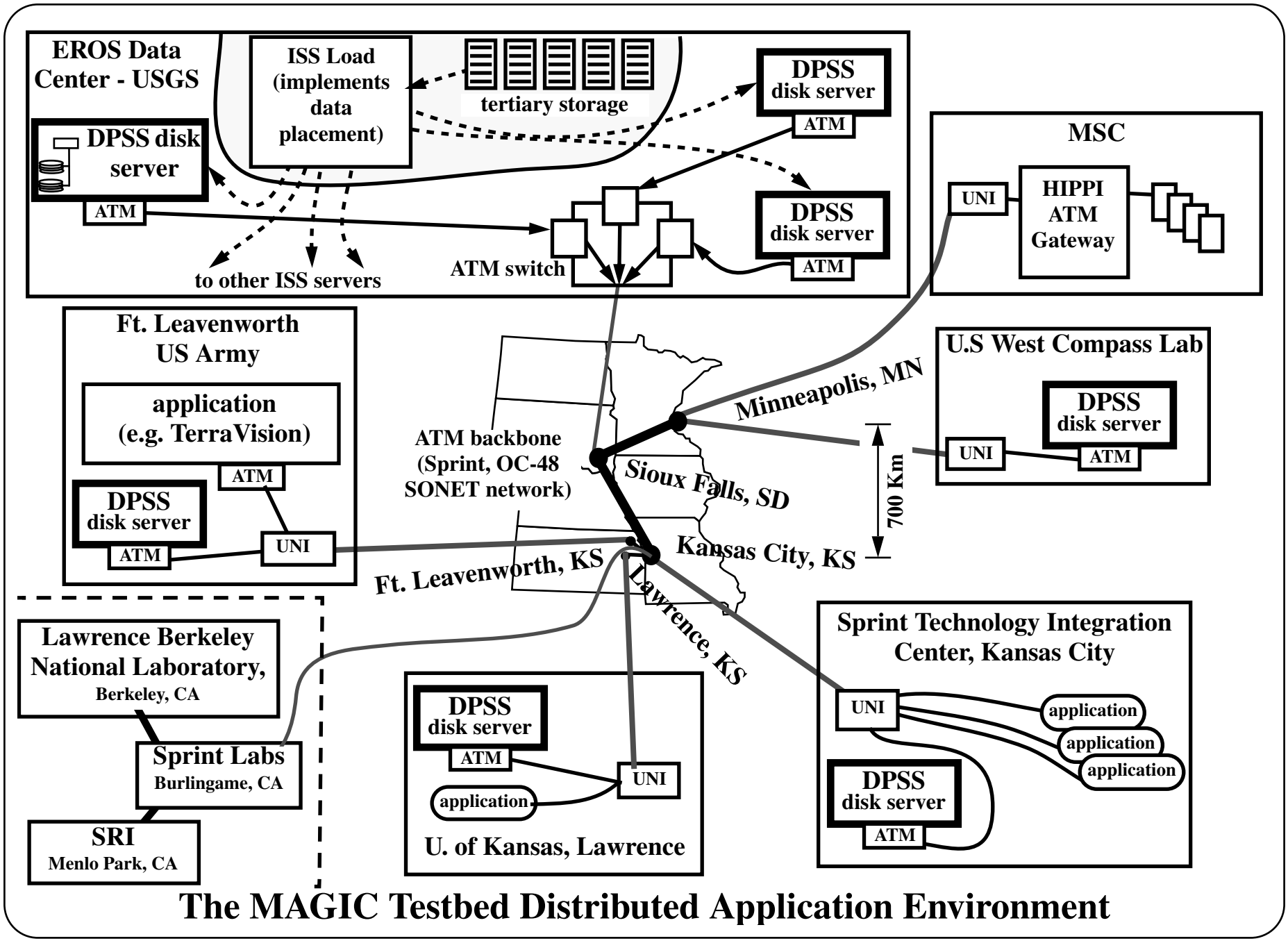
- The prototype high-speed application for the DPSS was TerraVision, developed at SRI
- TerraVision uses tiles images and digital elevation models to produce a 3D visualization of landscape.
  - The tiles are distributed across DPSS servers that are scattered around the MAGIC network.
  - Multiple DPSS servers operate in parallel to satisfy TerraVision data requests.

DPSS



TerraVision and the DPSS Cooperate to Visualize Landscape

DPSS

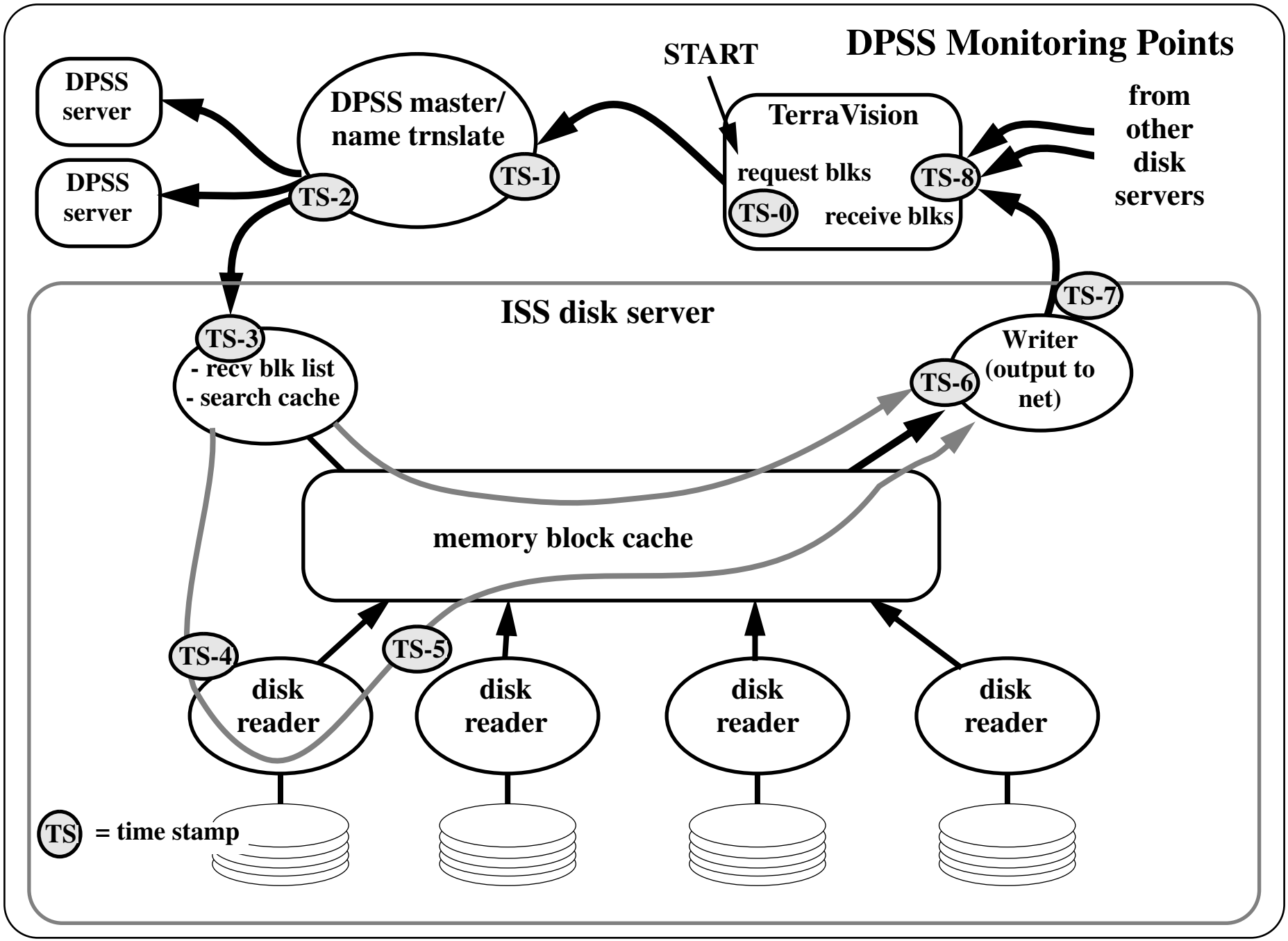


The MAGIC Testbed Distributed Application Environment

Performance Monitoring Approach and Experiments

- **Using NetLogger/LogTracer:**
  - **a methodology and set of tools to aid in create graphs that trace a data block through a distributed system**
- **Collect time stamps at all critical points in the data path. (see figure)**
- **Timing information is carried as a defined part of the data block structure.**
- **At least three, and usually more, systems are typically involved in the monitoring experiments.**

Monitoring


DPSS Monitoring Points
START
DPSS server
DPSS server
DPSS master/ name trnslate
TS-1
TS-2
TerraVision
request blks
TS-0
receive blks
TS-8
from other disk servers
ISS disk server
TS-3
- recv blk list
- search cache
TS-7
Writer (output to net)
TS-6
memory block cache
TS-4
TS-5
disk reader
disk reader
disk reader
disk reader
TS = time stamp

Monitoring

- **OS and network monitoring include:**
  - **TCP retransmits**
  - **CPU usage (user and system) and CPU interrupts**
  - **AAL 5 information**
  - **ATM switch buffer overflows**
  - **ATM hosts adapter buffer overflow**
- **Tools**
  - ***netstat* and *vmstat* modified to poll and report continuously**
    - **(Currently poll at 100 ms, so data is accurate to +/- 50 ms)**
  - **SNMP queries to switches and network interfaces**
    - **switch buffer overflows**
  - **(unfortunately SNMP time resolution is too coarse to be very useful)**

Monitoring

- **Common logging format:**

  keyword; hostname; seconds; nano-sec; data; data; data;.....;

  - **"keyword" - an identifier describing what is being logged - e.g. a reference to the program that is doing the logging (i.e):**
    DPSS_SERV_IN, VMSTAT_SYS_CALLS,
    NETSTAT_RETRANSSEGS, TV_RQ_TILE
  - **"data" elements (any number) are used to store information about the logged event - for example:**
    - **NETSTAT_RETRANSSEGS events have one data element: the number of TCP retransmits since the previous event**
    - **DPSS_START_WRITE events have data elements containing: the logical block name, the data set ID, a "user session" ID, and an internal DPSS block counter**

Monitoring

NETSTAT_OUTDATASEGS; 806706009; 652917; 4817378;
NETSTAT_RETRANSSEGS; 806706009; 652917; 16395;
NETSTAT_RETRANSBYTES; 806706009; 652917; 111841701;
NETSTAT_OUTDATASEGS; 806706009; 760199; 0;

VMSTAT_INTR; 806706009; 546568; 71612733;
VMSTAT_SYS_CALLS; 806706009; 546568; 2794466576;
VMSTAT_USER_TIME; 806706009; 546568; 2;
VMSTAT_SYS_TIME; 806706009; 546568; 5;
VMSTAT_INTR; 806706009; 646444; 3;

APP_SENT; 824951202; 824949; 34; 78; 45; 0; 6; 5; 198.207.141.6;
DPSS_MASTER_IN; 824951202; 832232; 34; 78; 45; 0; 6; 5; 198.207.141.6;
DPSS_MASTER_OUT; 824951202; 865724; 34; 78; 45; 0; 6; 5; 198.207.141.6;
DPSS_SERV_IN; 824951202; 877494; 34; 78; 45; 0; 6; 5; 198.207.141.6;
DPSS_START_READ; 824951202; 885279; 34; 78; 45; 0; 6; 5; 198.207.141.6;
DPSS_END_READ; 824951202; 909439; 34; 78; 45; 0; 6; 5; 198.207.141.6;
DPSS_START_WRITE; 824951202; 910743; 34; 78; 45; 0; 6; 5; 198.207.141.6; 49264
APP_RECEIVE; 824951210; 914210; 34; 78; 45; 0; 6; 5; 198.207.141.6;
APP_SENT; 824951202; 824949; 34; 76; 45; 0; 6; 3; 198.207.141.6;

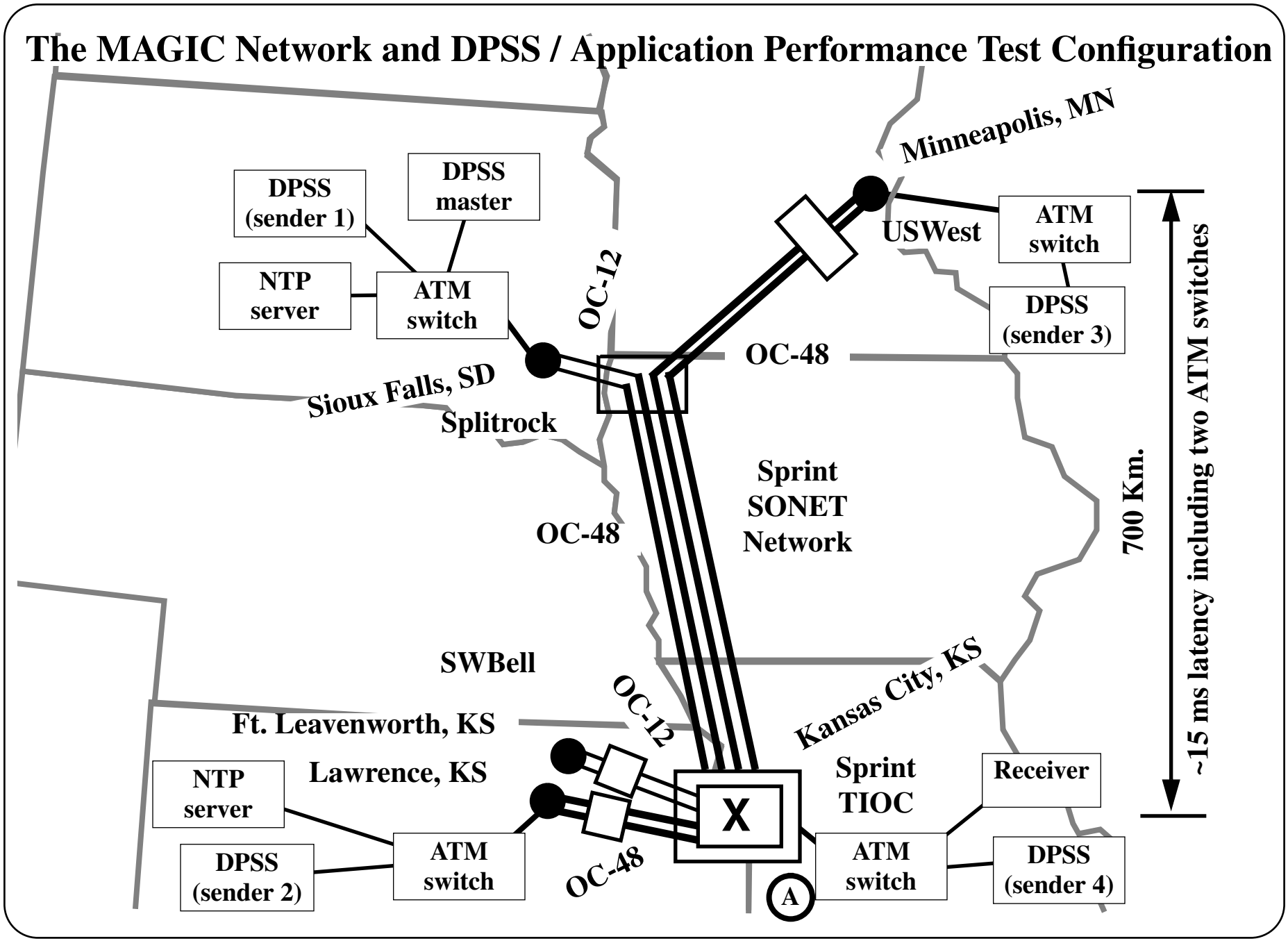
Monitoring

- **NTP is used to synchronize time stamps throughout MAGIC**
  - **All MAGIC hosts run *xntpd*, which synchronizes the clocks of each host both to time servers and to each other**
  - **The MAGIC backbone segments are used to distribute NTP data, allowing us to synchronize the clocks of all hosts to within about 250 microseconds of each other, but...**
    - **Many different sys admins (harder to synchronize than clocks)**
    - **The systems have to stay up for a significant length of time for the clocks to converge to 250 μs**

Monitoring Experiments

- **The monitoring experiments are carried out in the MAGIC environment:**
  - **A 700 Km diameter, star topology, configurable OC-48 SONET network**
  - **OC-3 and OC-12 ATM switches**
  - **Backbone and "leaf" switches**
  - **Several of directly attached hosts at each of six sites**

Monitoring Experiments



Monitoring Experiments

- **Network "tuning"**
  - **Initial studies by KU and MSCI established the optimal TCP parameters, and the cell pacing for the DEC Alphas (which, at the time, were the only senders capable of overrunning the switches).**
- **Server to Application Throughput (per server)**

DPSS Server Configuration	Burst (Mbits/sec)		Average (Mbits/sec)	
	TCP	UDP	TCP	UDP
Sun SS10-41; fddi	44	43	26	27
Sun SS10-42; fddi	62	56	37	40
Sun SS20-62; fddi	72	94	54	66
Sun SS20-62; atm	104	125	65	70
DEC Alpha 3000/600; atm	134	-	72	90

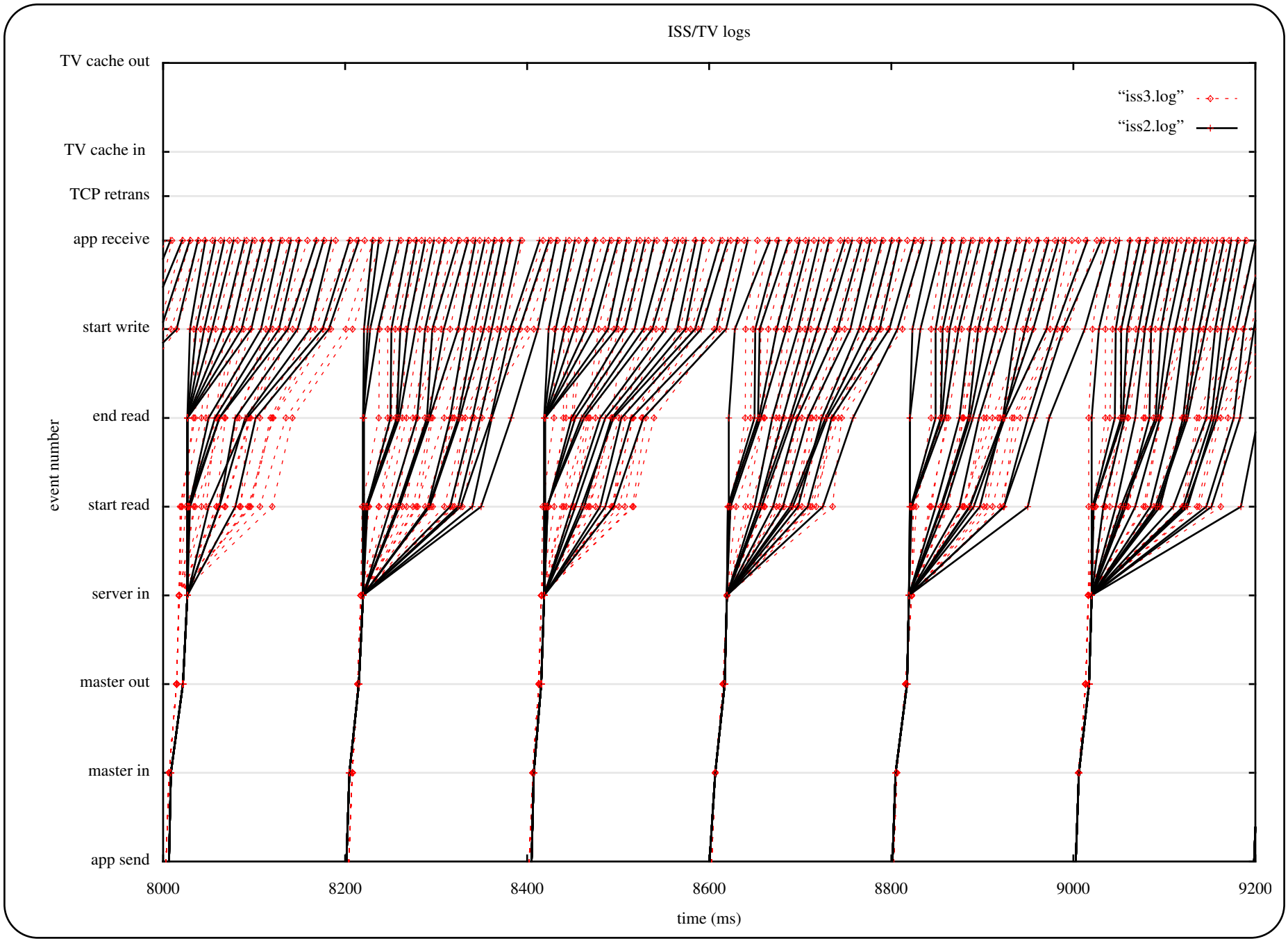
Monitoring Experiments

- **End-to-End performance experiments**
  - **TerraVision (the "real" application) is run interactively**
  - **Traces of data requests are collected**
  - **tv_sim is both a parameter and trace driven simulator (in trace mode, the application can be - and usually is - "speeded up")**

End-to-end, Top-to-Bottom Analysis

- **Control test: 2 servers over ATM LAN**
  - **Each line represents the history of a data block as it moves through the end-to-end path.**
  - **Data requests are sent from the application every 200 ms (the nearly vertical lines starting at *app_send* monitor point).**
  - **Initial single lines fan out as the request lists are resolved into individual data blocks (*server_in*).**

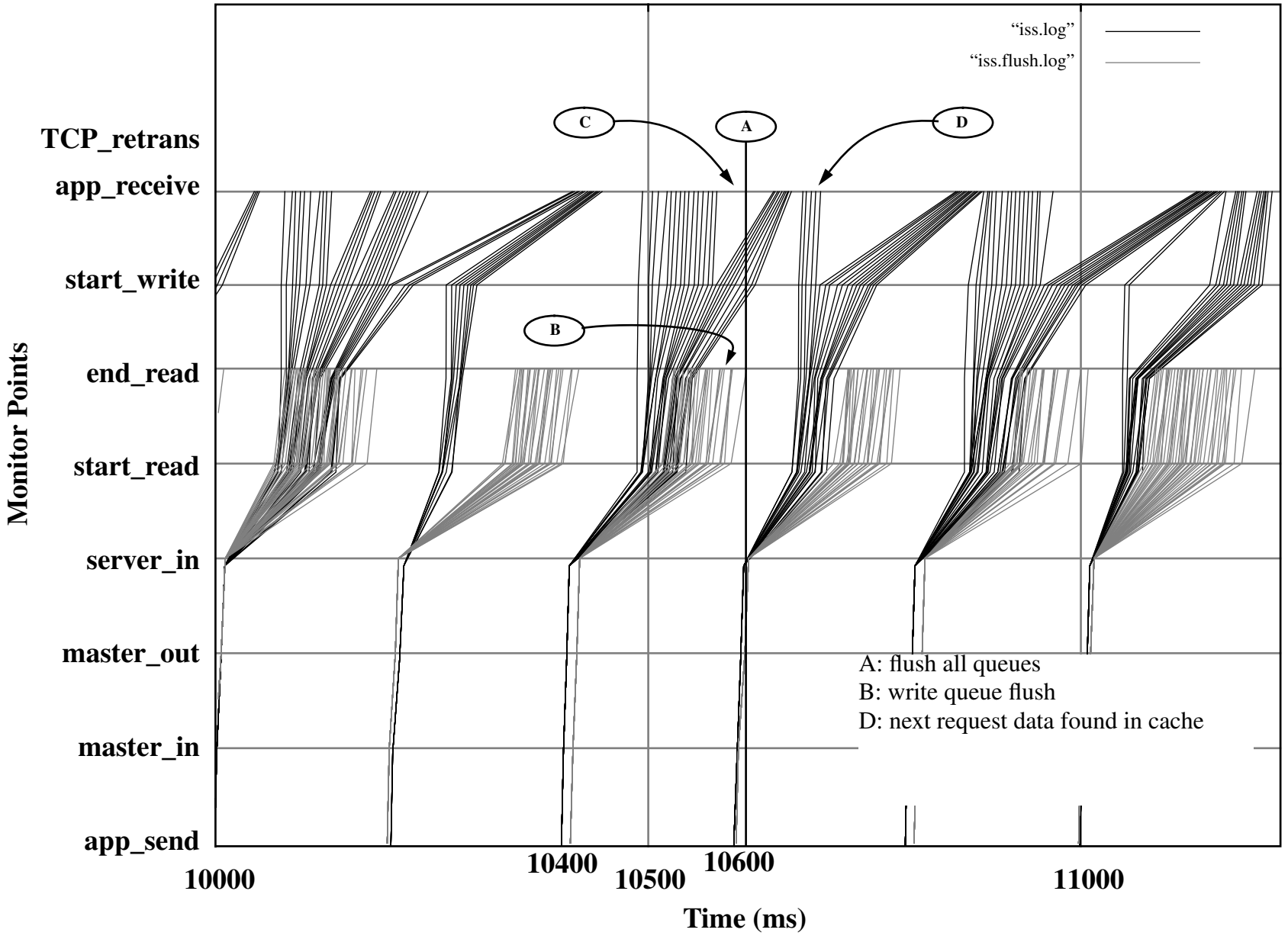
End-to-End, Top-to-Bottom Analysis



End-to-End, Top-to-Bottom Analysis

- **The following figure illustrates:**
  - **At Ⓐ flush of all queues due to next arriving block request list.**
  - **At Ⓑ write queue flush because a transmission delay (at Ⓒ) prevented this collection of blocks from being transmitted before the next list arrived.**
  - **At Ⓓ the blocks that were flushed from the write queue are retained in the memory cache, and the next request data found this group in cache.**

End-to-End, Top-to-Bottom Analysis



One server, ATM LAN, one SS-20 as server, tv_sim on DEC 3000/600

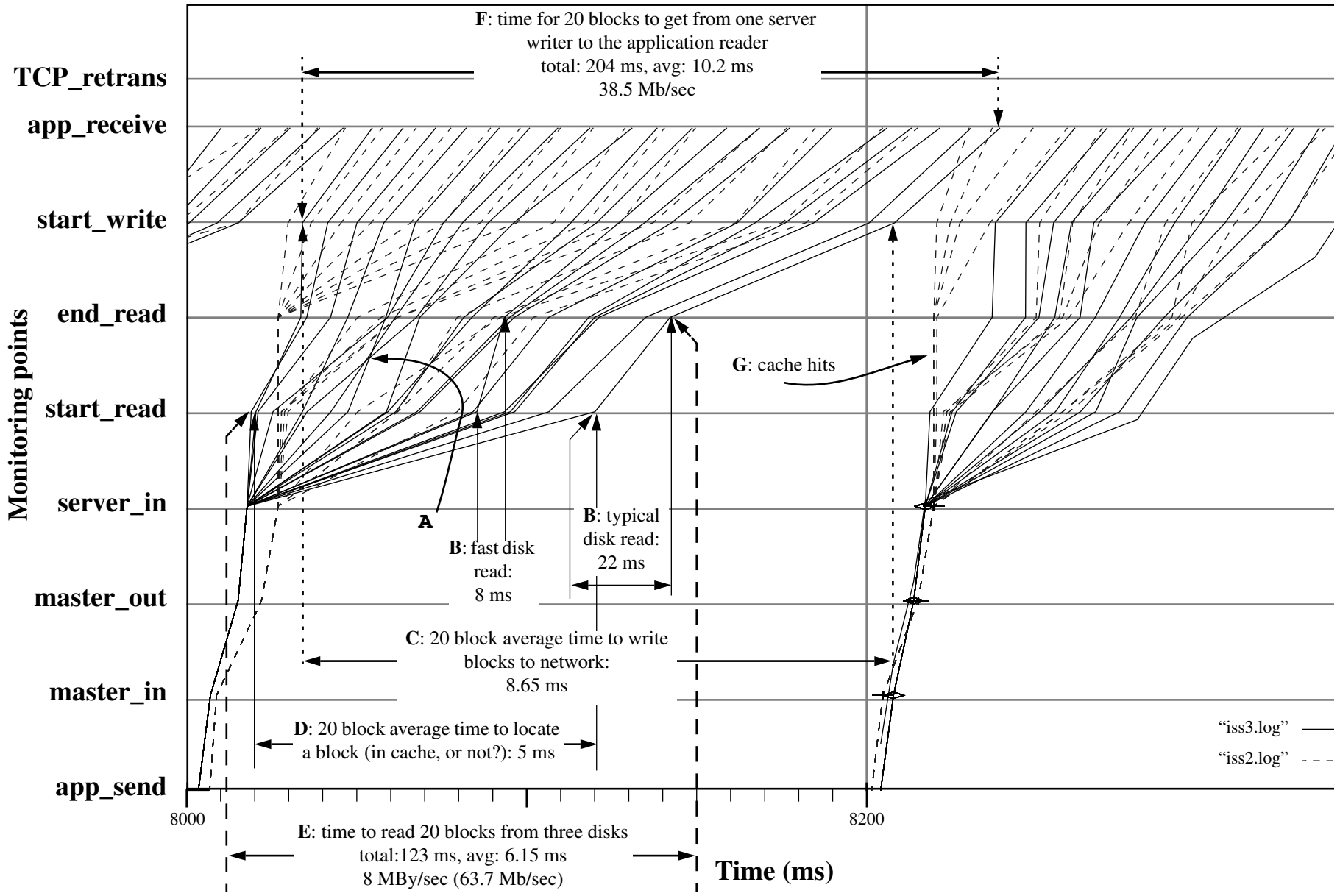
End-to-End, Top-to-Bottom Analysis

- **Notice that:**
  - **Some lines terminate at *end read*, a few also end at *start read***
  - **The reason for this is:**
    - **If a data request is not satisfied before the next request list arrives, it is flushed (discarded), from all the queues, but not from the memory cache.**
    - **The reason requests are flushed is based on the assumption that a new data block request is more important than any unsatisfied previous requests, so the DPSS discards all pending requests. The application needs to predict ahead. This approach ensures that the entire data pipeline stays full, and that disk server resources are never idle.**

End-to-End, Top-to-Bottom Analysis

- A detailed analysis of the data block life-lines shows:
  - A: if two lines cross in the area between *start read* and *end read*, this indicates a read from one disk was faster than a read from another disk
  - B: since all the disks are the same type, the variation in read times are probably due differences in disk seek times
  - C: average time to move data from the memory cache into the network interface is 8.65 ms
  - D: the average time to locate the block in memory cache or on disk is 5 ms
  - E: the average read rate from four disks is 8 MBy/sec
  - F: the average send rate (receiver limited, in this case) is 38.5 Mb/sec.
  - G: some requested data are found in the cache

End-to-End, Top-to-Bottom Analysis

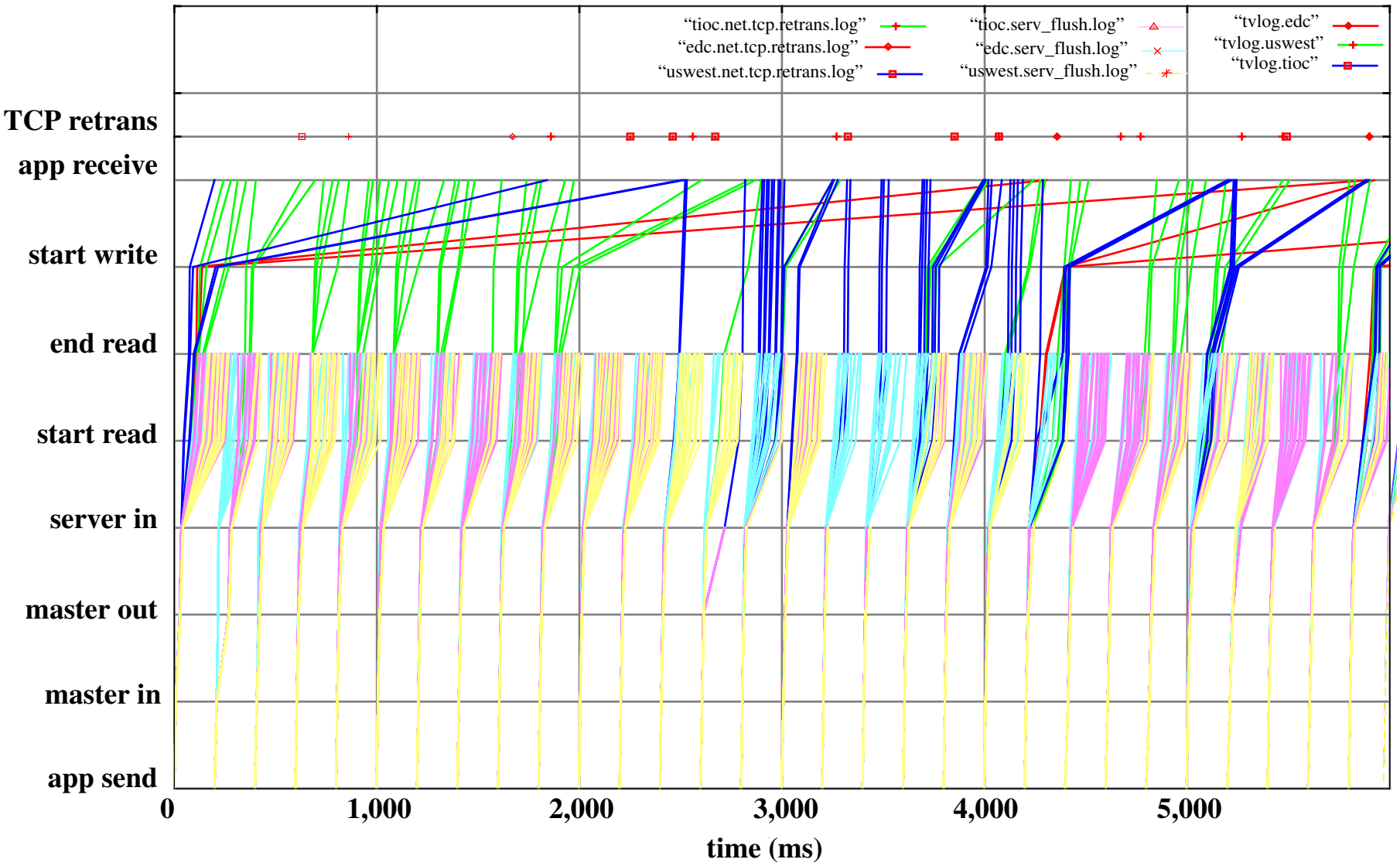


Two server, ATM LAN

End-to-End, Top-to-Bottom Analysis

- **Three server, WAN case (following figure)**
- **Things to notice:**
  - **Many TCP retransmissions, and some very long delays (up to 5500 ms) (Once a block is written to the TCP socket, the user level flushes have no effect, and TCP will re-send the block until transmission is successful, even though the data is likely no longer needed and is holding up newer data).**
  - **These long delays are almost always accompanied by one or more TCP retransmit events. (The elongated blocks at the top of the figure indicate the interval during which the retransmit took place).**

End-to-End, Top-to-Bottom Analysis

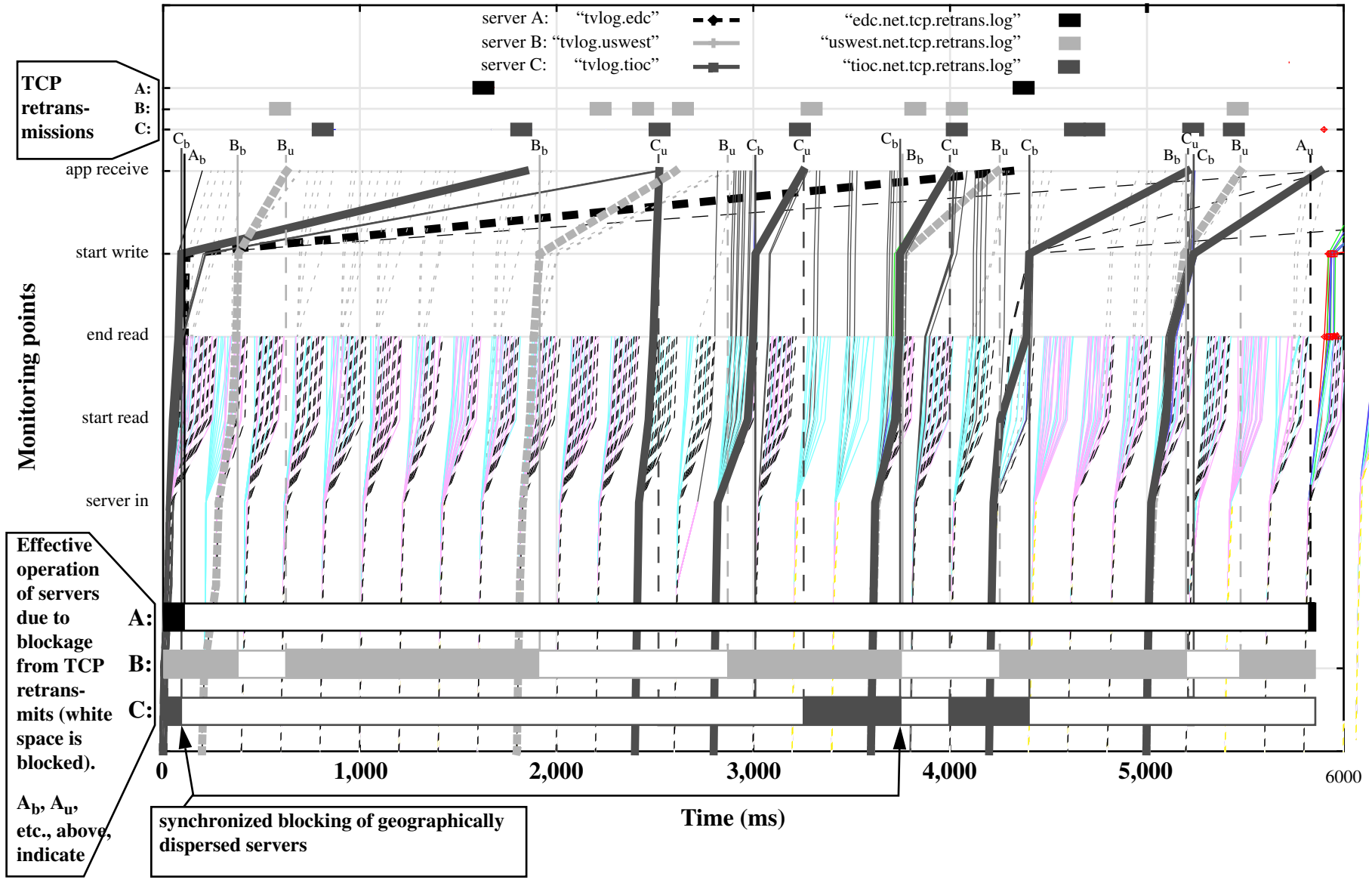


Three servers, ATM WAN, SS-10s as servers, tv_sim on SGI Onyx

End-to-End, Top-to-Bottom Analysis

- **The impact of this behavior on the server as a supplier of data is shown at the bottom of the next figure.**
  - **The horizontal bars indicate normal server operation by gray regions, and the blocked state by white regions.**
  - **These server operation profiles show almost unbelievably poor performance: With three servers operating (EDC, USW, TIOC) we obtained throughput of 0.240 Mbits/sec, 3.8 Mb/s, and 4.4 Mb/s respectively, to deliver 106 blocks in 5 seconds out of about 900 requested blocks.**
  - **This is in a network with minimum link speeds of 100 Mb/s and servers each with a verified data delivery capability of at least 45 Mb/s.**

End-to-End, Top-to-Bottom Analysis



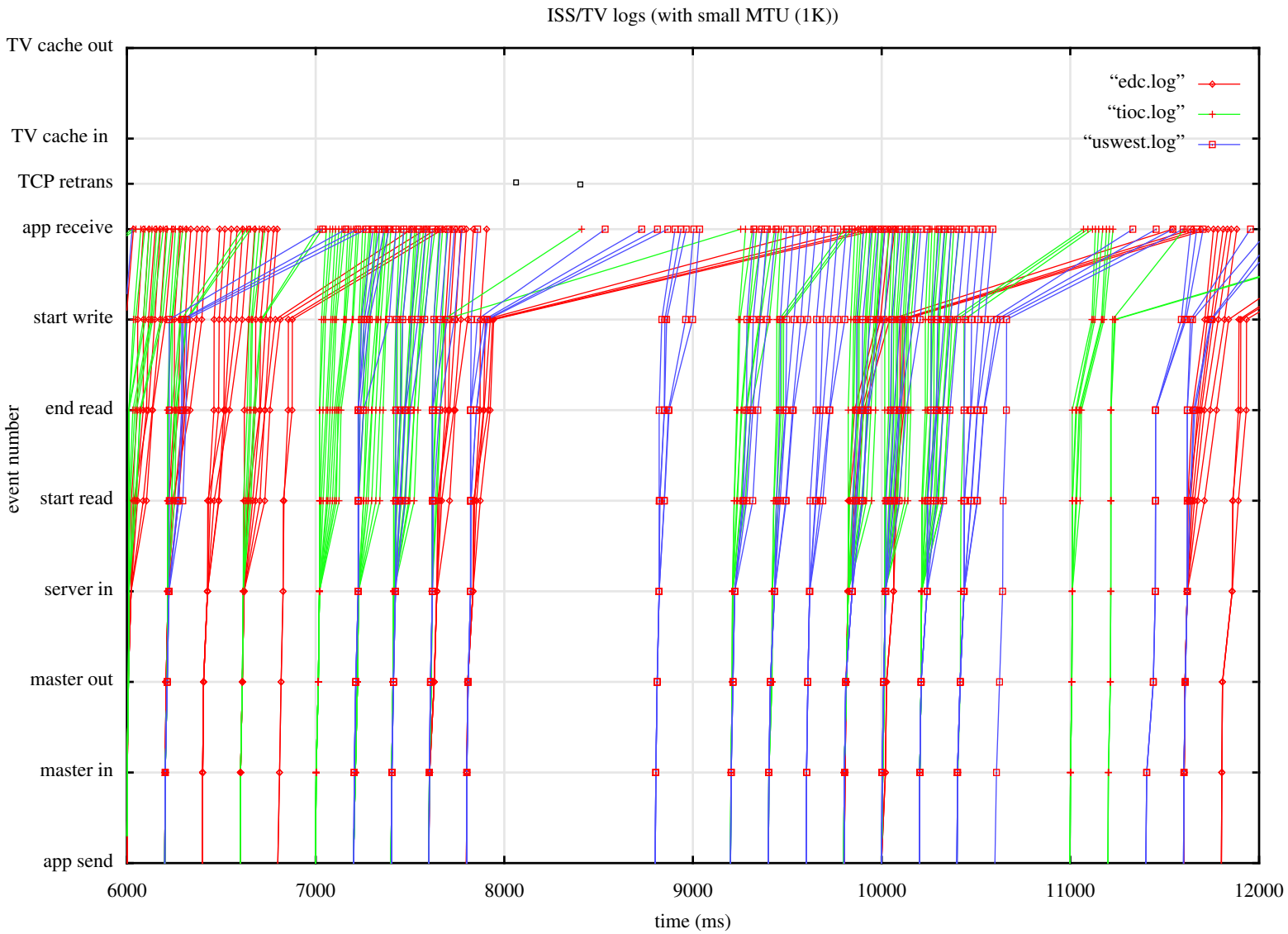
End-to-End, Top-to-Bottom Analysis

- **What we believe is happening is that TCP's ability to handle congestion is defeated by an unreasonable network configuration:**
  - **The final ATM switch where the three server streams come together has very small output buffers (about 13K Bytes).**
  - **The network MTU (minimum transmission unit) is 9180 Bytes (as is typical for ATM networks).**
  - **The TCP congestion window cannot get smaller than the MTU, and therefore TCP's throttle-back strategy is pretty well defeated.**
  - **Three sets of 9 KBy IP packets are converging on a link with less than 50% that amount of buffering available, resulting in most of the packets (roughly 65%) being destroyed by cell loss at the switch output port.**

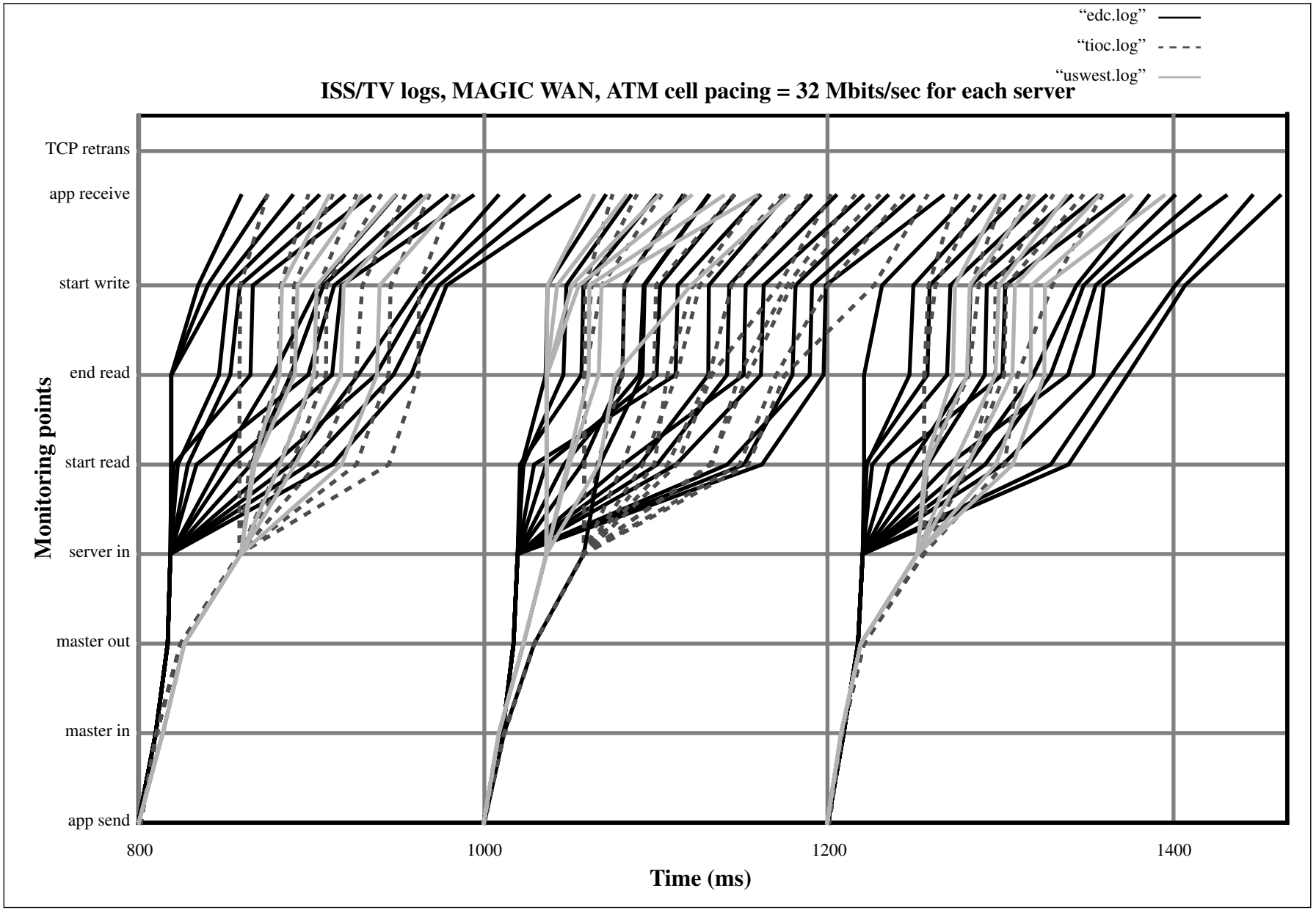
End-to-End, Top-to-Bottom Analysis

- **Solution 1: Reduce MTU Size**
  - **We can reduce the network MTU to a very small value (e.g. 1024 bytes). This helps somewhat. (see graph)**
- **Solution 2: Cell Pacing**
  - **We can also cell pace the disk servers to 1/N of the "broken link" bandwidth.**
  - **Cell pacing is bandwidth limiting at the level of ATM cells.**
  - **This helps a lot. Under these conditions, all three servers provided roughly equal throughput of 30 Mb/s, and the delivery of blocks was "well behaved". (see graph)**
- **Solution 3: Upgrade the switch**
  - **New switch just arrived: will test soon.**

End-to-End, Top-to-Bottom Analysis



End-to-End, Top-to-Bottom Analysis

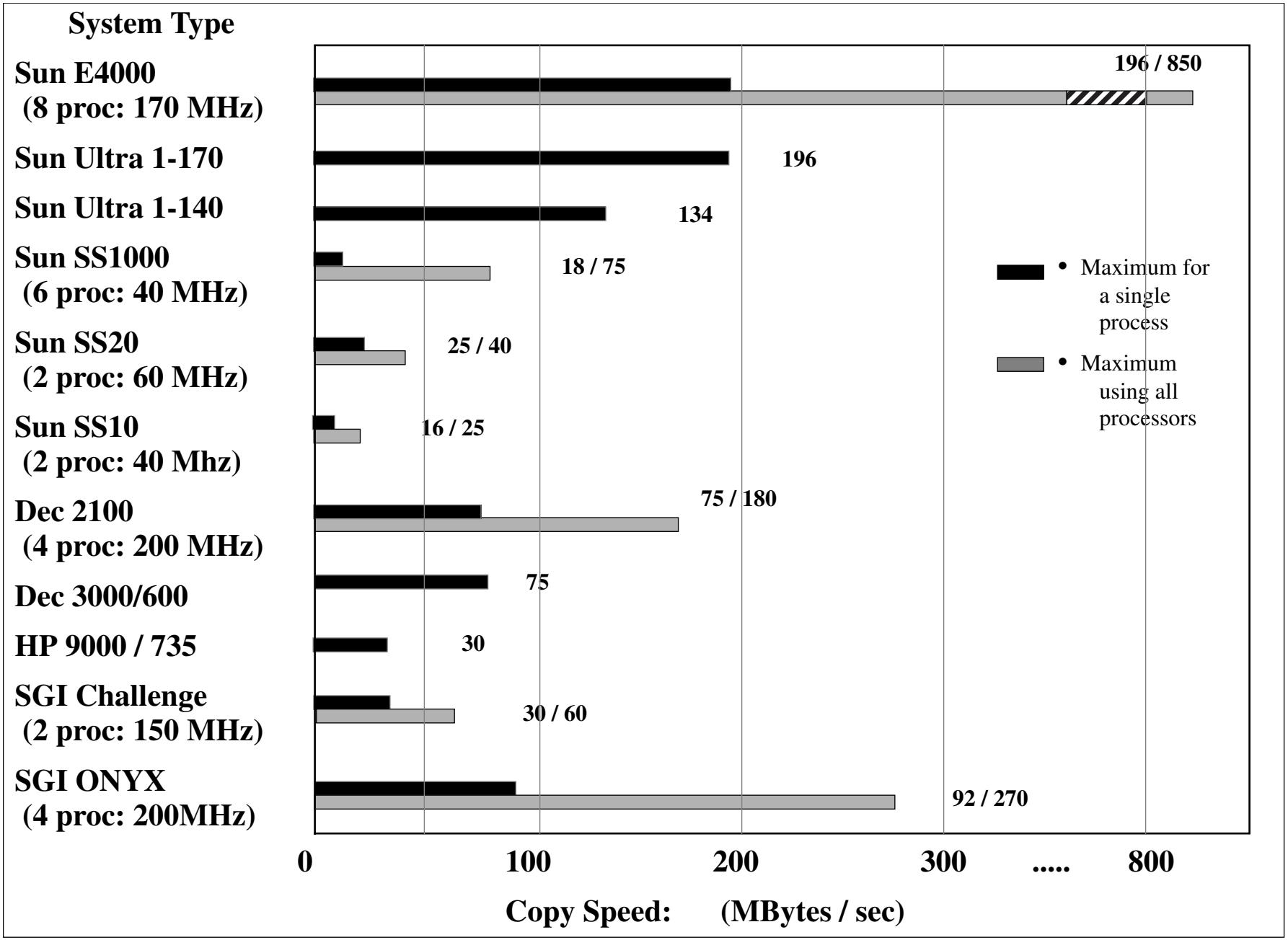


The Three Server ATM WAN Experiment with Cell Pacing

Memory Copy Speed

- In the previous generation of UNIX workstations, memory copy speed was the bottleneck
  - TCP is never better than 50% of the memory-to-memory copy speed (i.e.: a TCP stream in a SS10 will never exceed 18 Mbytes/sec, or 64 Mbits/sec)
- With the newest generation of hardware (i.e: Sun Ultra E4000), this is no longer the case
  - i.e.: a single TCP stream might now be as fast as 780 Mbits/sec, and total aggregate throughput over 8 streams might be a fast at 3.4 Gbits/sec (not yet verified)

End-to-End, Top-to-Bottom Analysis



Conclusions

- The Netlogger/LogTracer methodology is very helpful in finding problems in the end-to-end system.
- These type of graphs are very useful and informative.
- If you are building distributed applications:
  - timestamp all critical operations using a uniform log format
  - run NTP on all hosts
- If you are using ATM networks:
  - upgrade any old (pre mid-1995) ATM switch hardware!

References

- Cavanaugh, John, Timothy Salo, "Internetworking with ATM WANs", (available as http://www.msci.magic.net/docs/magic/ip-atm.ps)
- Evans, Joseph B., Victor S. Frost, Gary J. Minden, "TCP and ATM in Wide Area Networks", CNRI Gigabit Network Workshop '94. (http://www.magic.net/tcp/overview.html)
- Jonkman, Roelof J.T., "An Overview of NetSpec", Telecommunications & Information Sciences Laboratory, University of Kansas. (http://www.tisl.ukans.edu/Projects/AAI/products/netspec/)
- Lau, S, and Y. Leclerc, "TerraVision: a Terrain Visualization System,", Technical Note 540, SRI International, Menlo Park, CA, Mar. 1994. Also see: http://www.ai.sri.com/~magic/terravision.html
- Richer, I. and Fuller, B., "An Overview of the MAGIC Project," M93B0000173, The MITRE Corp., Bedford, MA, 1 Dec. 1993. (Available from http://www.magic.net/MAGIC_Summary.ps.)
- Tierney, B., Johnston, W., Chen, L.T., Herzog, H., Hoo, G., Jin, G., Lee, J., "Performance Analysis in High-Speed Wide Area ATM Networks: Top-to-bottom end-to-end Monitoring", IEEE Network, May/June 1996, LBL Report 38246. Available from http://www-itg.lbl.gov/ISS/papers.html.)
- Tierney, B., Johnston, W., Chen, L.T., Herzog, H., Hoo, G., Jin, G., Lee, J., "Using High Speed Networks to Enable Distributed Parallel Image Server Systems", Proceedings of Supercomputing '94, Nov. 1994, LBL-35437. Available from http://www-itg.lbl.gov/ISS/papers.html.)

- **For more information see:**

  http://www-itg.lbl.gov/DPSS/logging/

  Information on NetLogger/LogTracer Package

  http://www-itg.lbl.gov/DPSS

  Lots of information and Papers on the DPSS

  http://www-itg.lbl.gov/DPSS/talks

  These slides

  http://www.magic.net

  Magic Project Home Page

  Much of the material in this presentation is described in more detail in an article in the May issue of IEEE Network.