The InTENsity PowerWall:
A Case Study for a Shared File System Testing Framework

Alex Elder, Thomas M. Ruwart, Benjamin D. Allen,
Angela Bartow, Sarah E. Anderson, David H. Porter
Laboratory for Computational Science and Engineering
University of Minnesota
Minneapolis, MN 55455
{elder,tmr,benjamin,bartow,sea,dhp}@lcse.umn.edu
tel +1-612-626-0059
fax +1-612-626-0030

Abstract

The InTENsity PowerWall is a display system used for high-resolution visualization of very large volumetric data sets. The display is linked to two separate computing environments consisting of more than a dozen computer systems. Linking these systems is a common shared storage subsystem that allows a great deal of flexibility in the way visualization data can be generated and displayed. These visualization applications demand very high bandwidth performance from the storage subsystem and associated file system.

The InTENsity PowerWall system presents a real-world application environment in which to apply a distributed performance testing framework under development at the Laboratory for Computational Science and Engineering at the University of Minnesota. This testing framework allows us to perform focused, coordinated performance testing of the hardware and software components of storage area networks and shared file systems.[2] We use this framework to evaluate various performance characteristics of the PowerWall system's storage area network. We describe our testing approach and some of the results of our testing, and conclude by describing the direction of our future work in this area.

1 Introduction

The InTENsity PowerWall is a very high-resolution display system built in the summer of 1999 at the Laboratory for Computational Science and Engineering (LCSE) at the University of Minnesota. It represents the second generation of PowerWall technology, pioneered at the LCSE in the mid-1990's. The new PowerWall is comprised of five large flat display screens oriented radially around a central viewing area, with each screen displaying two rear-projected XVGA (1280x1024 pixel) panels. The high resolution of the InTENsity PowerWall allows for detailed visualization of very large data sets. It is also designed to allow for display of full, wall-sized images at rates in excess of 20 frames per second. Figure 1 depicts the InTENsity PowerWall screen and projector layout.

Currently the two major applications for the PowerWall system are generation and presentation of wall content. In "movie generation" mode, the power of either computing environment can be harnessed to render movies for display on the wall. The rendering software is also able to distribute its work across machines. Locating the data sets from

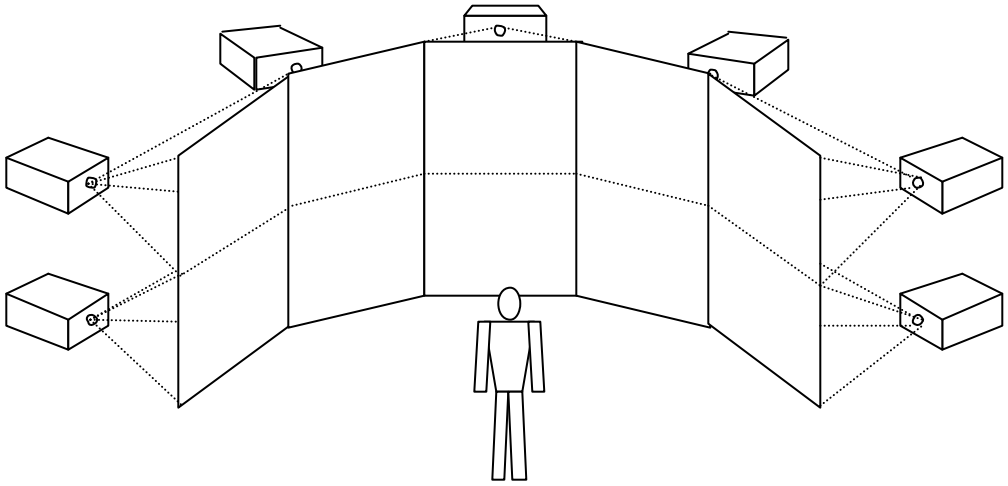


Figure 1. The InTENsity PowerWall at the LCSE

which these movies are derived (as well as the movie files themselves) on shared storage is desirable in order to avoid unnecessary data movement.

The "movie playback" mode requires a great deal of bandwidth from the storage subsystem into the memory of the system driving the displays. For movies to appear reasonably fluid they must be played at a rate of at least 15 frames per second, and preferably greater. These frame rates are made possible by distributing the task of movie display across ten machines. Given the PowerWall's 6400x2048 resolution, a single frame using 4-byte pixels is over 50 MB of data. Thus, over one gigabyte per second aggregate bandwidth is needed to sustain a full-resolution InTENsity PowerWall movie at 20 frames per second.

The InTENsity PowerWall can be driven by either of two distinct computing environments. The first consists of a pair of high-end SGI computers: an Onyx and an Onyx 2, each with two Infinite Reality™ graphics engines. These systems are used primarily for continued support of our existing PowerWall software and hardware base thereby easing the transition to the new InTENsity PowerWall format. Each of these systems has multiple Fibre Channel interfaces providing high bandwidth access to the storage subsystem. The second computing environment consists of a cluster of ten 4-processor SGI Visual PC 540 workstations. Each workstation sends video output to one of the ten panels on the wall. Two Fibre Channel interfaces on each workstation provide access to the storage subsystem.

A fairly complex storage area network was required to meet the performance and connectivity requirements of the InTENsity PowerWall. The result of our design is a storage system that is both capable and flexible. It is also an excellent environment in which to test performance characteristics of emerging storage area network hardware and software technologies. Our existing applications naturally stress storage systems in a number of ways. Yet we believe that application level testing like this is not sufficient to understand the complexity that comes into play to yield a given level of storage system performance. Instead, a more focused and closely controlled test environment is needed.

We will describe in this paper the framework we have developed for performing just this kind of controlled testing in a storage area network environment.

2 The LCSE Storage Area Network

We have constructed a storage area network that connects the systems involved with driving the PowerWall with a common set of disks via a Fibre Channel fabric. We designed this storage area network (SAN) with two main goals:

- Maximizing bandwidth available between hosts and disks
- Maximizing connectivity between hosts and disks

The hosts on the SAN consist of the two "large" Silicon Graphics ONYX systems and 12 "small" Intel-based Windows NT machines. All the hosts are connected to over 5 terabytes of disk storage through a fabric of four 16-port Fibre Channel switches (see Figure 2).

One of the large hosts is an 8-processor Silicon Graphics Onyx 2 which has four 2-port Prisa Fibre Channel adapters connected to the fabric. The second large host is a 4-processor Silicon Graphics Onyx with a total of four Fibre Channel ports connected to the fabric. Each of these machines has two Infinite Reality™ graphics engines for rendering images and/or for displaying movies on the InTENsity PowerWall.

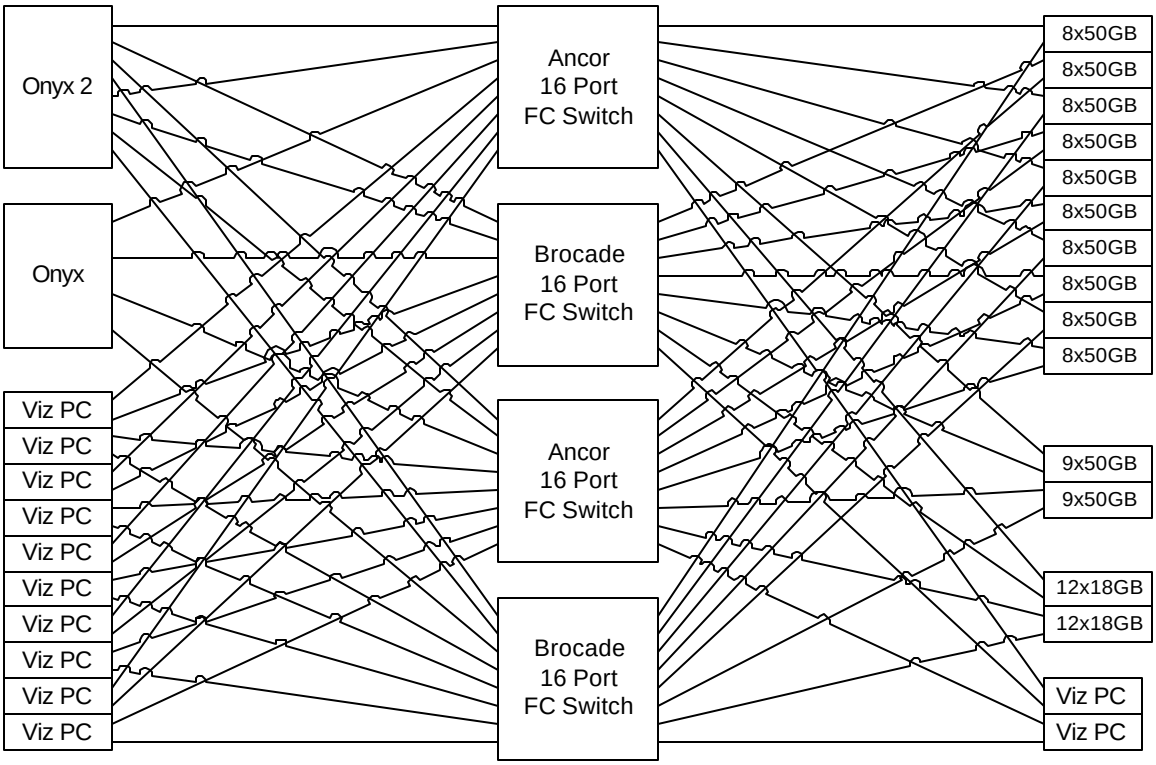


Figure 2. The LCSE storage area network supporting the InTENsity PowerWall

The small hosts are all 4-processor SGI Visual PC model 540 workstations (VizPC's). The graphics outputs of ten VizPC's are used to drive the ten panels of the InTENsity PowerWall. The remaining two VizPC's are used for development and for control of the

wall. These machines also expand the computing capability of the cluster, and serve as redundant spares in the event of a failure of one of the wall-driving machines. Each of the VizPC's has a single 2-port Qlogic Fibre Channel host-bus adapter; all of these Fibre Channel ports are connected to the fabric.

Four 16-port Fibre Channel switches implement the fabric. Two of the switches are Ancor MK-II's, and the other two are Brocade Silkworm's. Wherever a Fibre Channel host bus adapter or a drive enclosure has two ports to the fabric, one is connected to an Ancor switch and the other connects to a Brocade switch. Our configuration allows all hosts access to all disks through at least one path through the fabric.

The storage portion of our storage area network is made up of two generations of Seagate Fibre Channel disks enclosed in three types of drive enclosure. Our newest drives, Barracuda 50's, make up the majority of our storage. Ten JBOD enclosures made by JMR hold 80 of these disks. An additional 18 of these drives are enclosed in two Ciprico JBOD boxes. Finally we have two 12-drive MTI JBOD's filled with 18GB Barracuda drives.

3 PowerWall Applications

As mentioned earlier, two current PowerWall applications drive the high performance requirements of its shared storage subsystem. The first application is the generation of movies for display on the InTENsity PowerWall. The second application is the display of the generated movies. Each of these has fairly well-behaved I/O characteristics.

3.1 Movie Generation

Movie generation is a process of rendering movie frames from a very large, time-varying 3-Dimensional data set. Each data set contains many instances of a single volume of data, each instance representing the state of the volume as it evolves over time. The view of the volume is determined through an interactive process, whereby low-resolution image frames are generated and reviewed until a desired viewing location and angle are found. The resulting view information is then saved as a "key frame." A sequence of these key frames define a "flight path" though the data set in space and time. View information for the remaining frames of a movie are defined by interpolation between key frames.

Once the frames along the movie path have been defined the final rendering process is initiated. This process involves rendering full resolution images for display on all ten panels of the wall for each frame in the movie path. Figure 3 depicts a path that rotates around the volume three times before repeating itself.

The data being rendered tends to be on the order of several gigabytes for an instance of a single volume and terabytes for an entire data set. As such, it is currently not possible to render the entire data set using in-core rendering techniques. The data set is therefore organized as a fixed-size hierarchy of sub-volumes. The size of these sub-volumes is a tunable parameter that can be matched to the characteristics of the system doing the rendering; typically they're in the range of 1 MB to 16 MB apiece. This organization allows for efficient data access by the rendering process, which has been designed to run in parallel across many processors and computers in a clustered-computing environment.

The output of each rendering step is a single panel-sized image; at four bytes per pixel, this comes out to 5 MB of data output per step. By using a shared storage resource for the original data set and the resulting movie frame storage, separate host systems can perform any rendering step without concern for excess data movement.

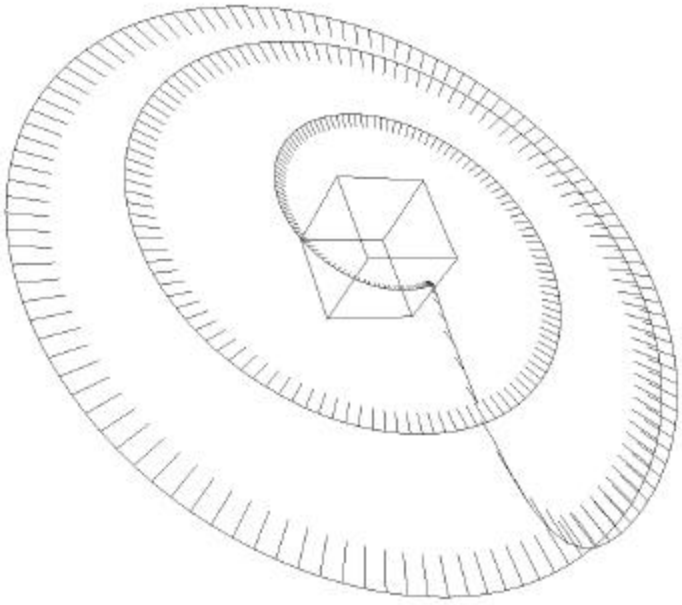


Figure 3. A movie path; each tick represents a frame's view of the volume

3.2 Movie Playback

This application consists of transferring up to ten independent streams of movie frames from the disk subsystem to the display. The streams are synchronized so that corresponding frames on separate screens are shown together. The movie player allows the display frame rate to be adjusted, subject to the limits imposed by the host systems' ability to keep pace with the data transfer rates required. The movie player makes use of read-ahead and asynchronous I/O techniques to maximize the effective data transfer rate and hence, the frame rate.

4 Testing Approach

There are different aspects of performance (such as bandwidth, requests per second, request completion time) that may be of interest for a given situation. But it is inadequate to express any performance metric as a single number, such as 1000 transactions or 20MB per second. Rather, the performance of a single disk (for example) should be expressed as a function of some other variable, such as request size or media position. This is because these other variables can have significant impact on the value of the metric being measured. Furthermore, being able to review a series of measurements in a time-correlated manner is useful in understanding where and when various performance anomalies occur in a storage subsystem. This is especially important in a shared-access environment where a single computer system does not have the ability to exclusively storage subsystem access.

To perform the evaluation of the various system components we have leveraged our existing tools and experience in testing raw storage system performance. The xdd program is a utility developed at the LCSE to assist in determining performance characteristics of the storage devices, both individually and in groups (i.e. logical volumes). The xdd program offers a very fine level of control and produces highly reproducible results, making it more suitable than some other available benchmarking programs for our purposes.

Our approach to performance testing attempts to take into account the all components of the system under test. Where possible, we perform tests that specifically exercise one component to understand its contribution to the overall system performance. This reflects our philosophy that the performance behavior of a complex system can only be understood after first understanding the performance of its components. Our testing attempts to evaluate simple cases, gradually making them more complex. As anomalies in behavior are noted, we consider all components in attempting to determine the cause. Based on this, our approach to understanding the performance of the storage area network was to do a series of single host tests, then move on to more complex tests involving multiple hosts accessing shared storage area network resources concurrently.

For our storage area network, the kinds of components that can impact performance include:

- Components internal to a computer system. These include architectural features which place limits on performance. They also include limits imposed by operating system software.
- Components at the system/storage interface. This covers host bus adapters (HBA's) and their drivers, which are typically developed somewhat independently of any particular type of hardware or operating system.
- Components making up the fabric. This includes the hardware (switches, hubs, media) that make up the communication channel as well as the way in which those components are interconnected.
- Storage devices. Each storage device type has characteristics such as speed and as on-device cache size that can have sometimes surprising effects on performance.

When multiple hosts join to share access to common storage on a storage area network, the interactions become much more complex than the single host case. One host's activities involving the fabric or one or more storage devices can have large and unpredictable impact on the performance. So in addition to the above, we are interested in:

- The shared file system software. We consider this separate from the operating system because in the storage area network case this component is distributed among a number of host systems.

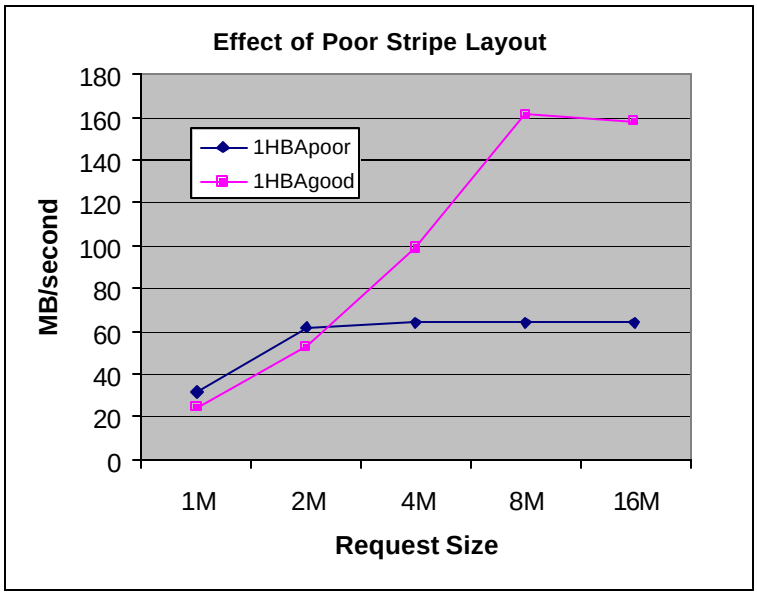
With a firmer understanding of the behavior of the components of our storage area network, we can get a better grasp on interactions that can complicate the performance picture. We have a much better basis for explaining storage system performance.

5 Single Host Testing

The first step in this performance evaluation process is to characterize the performance of a single host system connected to a logical volume through the fabric in isolation (i.e. through the fabric without any other traffic).

This establishes a baseline for further tests. These and all other tests described herein were performed using a single CVFS volume comprised of sixteen 50 GB drives, eight drives in each of two Ciprico JBOD's. Each of these JBOD's has two channel connectors, for the A and B ports of the drives within the enclosure. We connect both channels to the fabric. In addition, both of the Fibre Channel ports on each SAN host are connected to the fabric.

We observed immediately that the performance we were getting from the file system was not close to what we had expected. After some analysis we determined that the way the striping of the volume had been automatically laid out was less than ideal. Laying out the volume the way we had intended yielded a considerable improvement in some of the performance numbers. (Note: All performance results listed here are for read operations.)



0	1	8	9	10	11	12	13
14	15	2	3	4	5	6	7

"Bad" stripe layout

0	2	4	6	8	10	12	14
1	3	5	7	9	11	13	15

"Good" stripe layout

Figure 4. Effect of stripe layout on performance

Next we did some experimenting with the host bus adapter. Our development system had two Fibre Channel HBA's installed, and this gave us an additional option on testing. It allowed testing for the one host case to be performed using either two ports from a single adapter, or one port from each of two adapters. We found that this too made a difference in the rate at which data could be read from disk (see figure 5a).

We did a some testing to examine the effect of varying file system striping parameters on overall file system performance. CVFS allows the file system logical block size to be changed at file system build time. It is also possible to define what they call the "stripe

breadth," meaning the number of file system blocks that are read/written to a given disk in a stripe before moving on to the next drive in the stripe. We tried a number of configurations of this: 32x32K, 16x32K, 8x32K, and 16x16K. The performance curves for these combinations are show n in figure 5b.

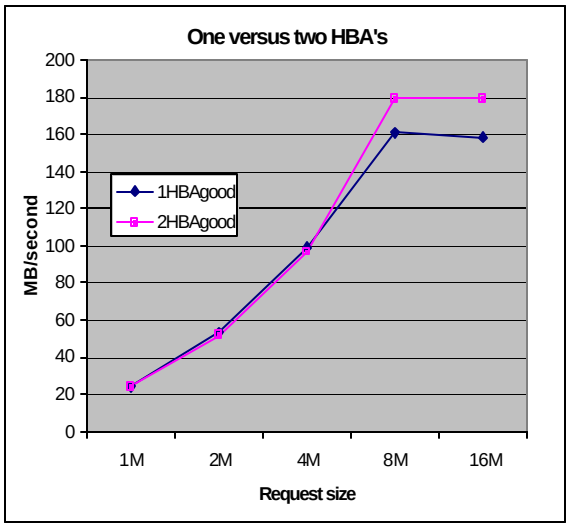


Figure 5a. Improved performance due to use of multiple HBA's

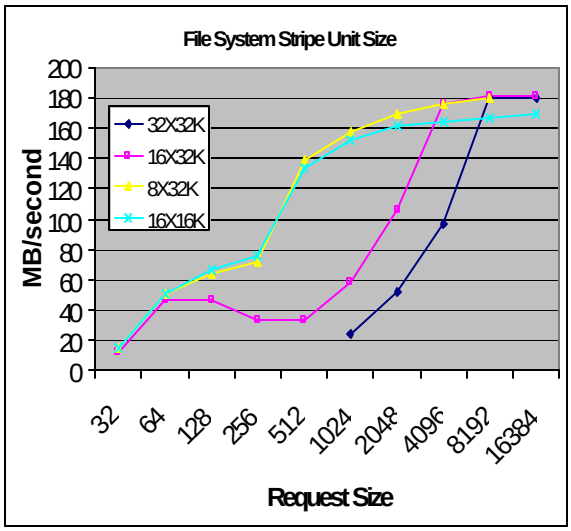


Figure 5b. Affect of different stripe breadths on read performance

This deserves a little further explanation. By reducing the effective stripe breadth the number of bytes transferred to and from a disk drive for a given operation was reduced. The effect of this was to increase performance for the mid-range request sizes (between 128KB and 2MB). The reason for this has to do with the size of the cache on the individual disk drives. Smaller transfers more readily fit in the cache, allowing it to be utilized more efficiently. Larger transfers, meanwhile, tend to overrun the disk cache, thereby losing performance benefit the cache might have offered.

6 Multiple Host Testing

Testing the shared file system software was more complex than testing the underlying storage subsystem and required the creation of a framework to coordinate testing on multiple systems concurrently. The two basic functions of this framework are:

- Accounting for the existence of multiple clocks
- Coordinating the initiation of tests to run concurrently on multiple hosts

Our performance testing utilities make use of precise time stamps to quantify and report storage performance characteristics. Each host accessing the shared storage has its own internal sense of time, and without a common reference clock it is impossible to interpret the relationship between tests run on separate hosts. Thus a consistent time base is needed in order to correlate test results generated by separate systems. We are also able to make use of a common clock to coordinate initiating tests on multiple hosts simultaneously.

6.1 Reference Clock

Each of the systems used for testing has a clock register that updates at a high frequency, allowing for very precise measurement of elapsed time. The resolution of this clock varies for different systems (ranging from 2 to 80 nanoseconds per tick or so), so clock values are converted to a common time unit (picoseconds) for the purpose of synchronization.[1]

We use a very simple clock model to establish a common time base. We assume that all clocks run at the same, constant rate. We therefore assume that conversion from a given machine's "local time" to the common "global time" involves only the addition of a constant to the local clock's value. With this simplified model we must only determine the value of the constant difference between pairs of clocks.

One machine is designated to keep the global sense of time. That machine provides a service with which others communicate to determine the offsets of their own clock from the global time. Each client initiates a request to the server to get the current global time. The difference between the time value returned and the client's local time is recorded as the basis for the offset. This offset is further adjusted to compensate for the propagation delay required to carry the time request and its response over the communication medium. This propagation delay bounds the error in the difference between our estimated and the actual offset between the two clocks. We perform this request/response protocol a number of times, and use the offset associated the minimum propagation delay as the final offset value.

6.2 Coordination of Concurrent Tests

With a common time base defined, it is possible to coordinate the initiation of tests on different host systems. We extended our existing testing software to determine the time offset for the machine under test. The program is provided an indication of a (global) time at which all tests are to begin. This global time is converted to a localized start time using the offset value. The program then polls the high-resolution clock repeatedly until the start time has arrived. At that point test execution begins. Test results generated by individual hosts are buffered during test execution, and saved to disk for later analysis.

6.3 Concurrent Host Test Results

We performed a series of tests using one, two, and four hosts concurrently reading from the same file on the shared file system. The graph in figure 6 shows the performance curves for the aggregate bandwidth achieved across all hosts for each of these tests.. Each host combination uses either two or four Fibre Channel ports connected to the disk subsystem. The results are given for one host using two ports, two hosts using two ports, two hosts using four ports, and four hosts using four ports. We observe that the performance curve for the single host case is the highest of the four up to a request size of 1 MB. This is because all the read operations are purely sequential, which makes ideal use of the caches on the disk drives themselves.

The graph of the two host, two port configuration shows the lowest performance. This is due the performance degradation caused by random I/O effects. Random I/O request

patterns reduce performance because of the expense of positioning drive heads; it also does not allow effective use of the disk caches for reads. The remaining two curves did better than this case because they had four channels to the disks, and were able to make use of the additional bandwidth to improve overall performance.

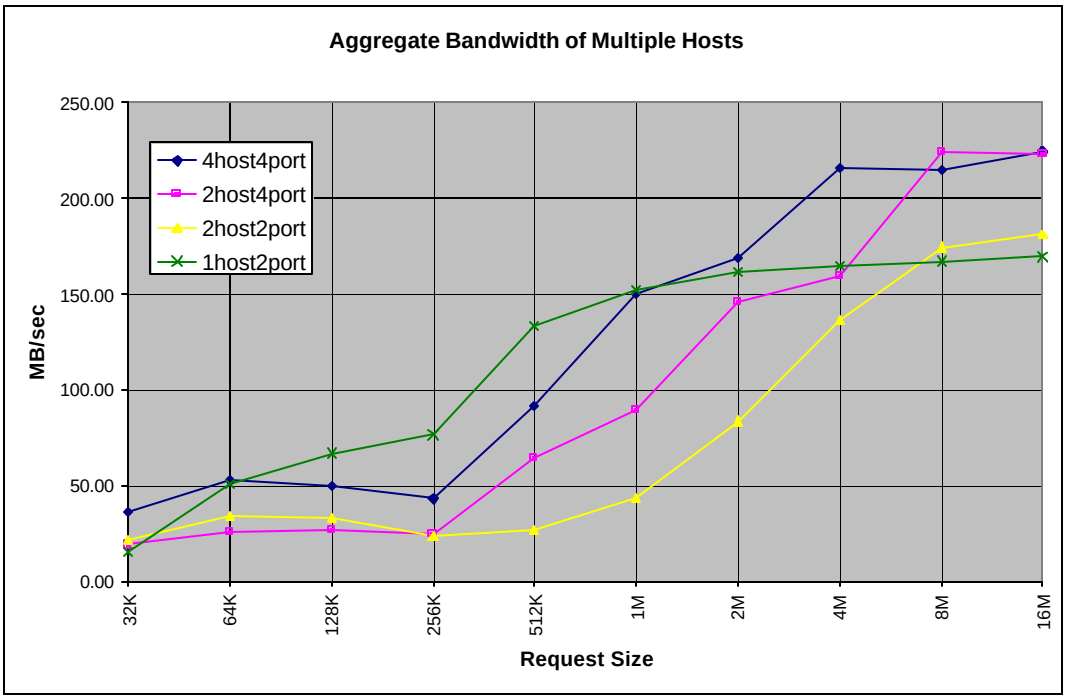


Figure 6. Aggregate bandwidth for multiple host tests.

7 Summary

The InTENsity PowerWall is a high resolution display system backed by a very high performance distributed computing system. The systems and storage area network that drive the PowerWall are a good test bed for evaluating and understanding the performance of shared storage technologies. We built a distributed testing framework that allows concurrent testing by multiple hosts of common hardware to help in evaluating such technologies. Our initial testing has demonstrated that achieving high performance, even in single host systems, is not as straightforward as might initially appear. Furthermore, achieving good, predictable performance in the face of the complexity of a shared storage environment will surely be a challenge. We believe there is much work to be done in this area.

8 Future Work

We have only scratched the surface on the topic of performance testing of shared file systems, and there are many obvious and relevant questions that spring to mind when considering this work. Generally speaking our future work will involve extending the testing framework to evaluate a much larger set of file system functionality. We also intend to continue beyond the limited testing whose results are presented here, and apply

the testing framework to include other file system environments. We will be evaluating other network technologies, such as VIA or Myrinet, to assess their effectiveness in improving inter-host communications as well as establishing a more accurate sense of a common time reference.

9 Conclusions

Generation and display of movies on the InTENsity PowerWall at the LCSE are I/O intensive applications that can take great advantage of the benefits offered by shared storage systems. They demand data rates from storage that are both maximal and consistent. The InTENsity system also provides an opportunity for experimentation with and evaluation of emerging shared storage technologies. We have extended our existing disk testing software to accommodate testing performance in a distributed environment attached to common storage. These extensions addressed issues of establishing a common time base and synchronizing the execution of tests on multiple systems. We found that this distributed testing framework served our needs well. It opens up a whole new range of possibility for further study.

References

[1] This work was supported in part by the National Science Foundation, under the NSF Cooperative Agreement No. CI-9619019, and by the Department of Energy through the ASCI Data Visualization Corridor Program under contract #W-7405-ENG-48.

[2] Thomas M. Ruwart, "Disk Subsystem Performance Evaluation: From Disk Drives to Storage Area Networks." NASA/IEEE Joint Storage Conference, March 2000.

[3] Sue B. Moon, Paul Skelly, and Down Towsley. Estimation and Removal of Clock Skew from Network Delay Measurements. Technical Report 98-43, Department of Computer Science, University of Massachusetts at Amherst, October 1998.

[1] We are currently using 100 base T Ethernet as the communication medium through which synchronization information is passed. Because of the latencies involved with this medium, picosecond clock granularity is probably overkill. Nevertheless, we designed the model to ensure it can accommodate better low-latency communication channels as well as the ever-increasing clock rates of computing equipment as they become available.