行政院國家科學委員會專題研究計畫成果報告

視訊伺服器之設計與製作(Ⅲ)

Design and Implementation of a Video Server (Ⅲ)

計畫編號：NSC 87-2213-E-009-085
執行期限：86 年 8 月 1 日至 87 年 7 月 31 日
主持人：張瑞川　　國立交通大學資訊科學學系

1. Chinese Abstract

許多視訊伺服器的研究都是只針對儲存系統來研究，因為磁碟頻寬被認為是視訊伺服器的瓶頸。事實上，其中重要的一個瓶頸是記憶體頻寬。在本報告中，我們詳述了我們的視訊伺服器的設計與製作，此視訊伺服器藉著增進有效記憶體頻寬來提升視訊伺服器的效能。此視訊伺服器執行於系統核心內，是一個 STREAMS 多工器，相較於一個等效但卻實作於使用者層次的視訊伺服器而言，效能提升了 57%。

關鍵詞：視訊伺服器、隨選視訊

Abstract

Many researches on VOD servers focus only on the storage subsystems, since it is believed that the disk bandwidth is the bottleneck of a VOD server. In fact, one of the bottlenecks of a VOD server is in the memory bandwidth. In this report, we describe the implementation of a VOD server called Nova, which improves the performance of the VOD server by improving the effective memory bandwidth. Nova runs within the kernel as a STREAMS multiplexor. The performance improvement, compared to an equivalent user level implementation, is 57%.

Keywords: Video Server, VOD

2. Motivation and Goal

Multimedia applications are becoming more and more popular as the progress of computing power, the increasing capacity of storage devices, the advances of compression technology for digitized audio/video data and still images, and the popularity of Internet. This report focuses on the research of video-on-demand (VOD) storage servers.

Since media data are usually huge and continuous in nature, traditional network file servers are not very suitable for storing media data. So many researches on VOD servers [1] , [5] , [6] , [8] , [12] design from the scratch for the requirements of media data. Some of them focus only on the storage subsystems, which use various techniques to squeeze the available bandwidth out of raw disks. In fact, in addition to the disk bandwidth, the bottleneck of a VOD server might also be the network bandwidth and the memory bandwidth. Many researches on I/O intensive applications also showed that the memory bandwidth is a scarce resource in the current generation of workstations [4] , [6] , [7] , [10] . So how to effectively utilize the memory bandwidth is our goal.

Our goal is to design and implement a VOD server, which utilizes memory bandwidth and disk bandwidth efficiently. In order to improve the effective memory bandwidth, the number of memory touches

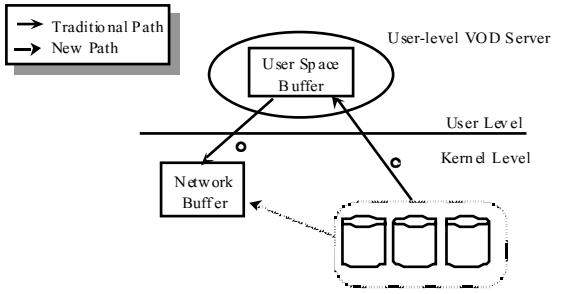

Figure 1: Data Transferring Path of a User-level VOD Server

occurring in the VOD applications must be reduced. Since the primary task of a VOD server is to pump data from disk to network, data read are not modified along the data-transferring path. Our idea is to pump data directly from disk to network, hoping that no memory copying is needed at all.

3. Results and Discussions

We have implemented a prototype of VOD server, called Nova. In order to reduce the redundant memory copy, Nova was implemented within the kernel to get rid of data copying needed for crossing protection boundary. The additional benefits of this in-kernel implementation are the avoidance of overheads incurred from context switches, system calls, and synchronization between threads, etc.

In order to pump data directly from disks to network, we make use of the STREAMS mechanism as the memory shortcut; that is, data are read directly from disks into network buffer as shown with the dotted line in Figure 1. The advantage of using STREMAS mechanism is that we don't have to modify the network subsystem.

By the kernel-level implementation and the STREAMS mechanism, no redundant memory copying is needed in Nova. To measure the performance gain of improving effective memory bandwidth by reducing the memory coping, an equivalent user level

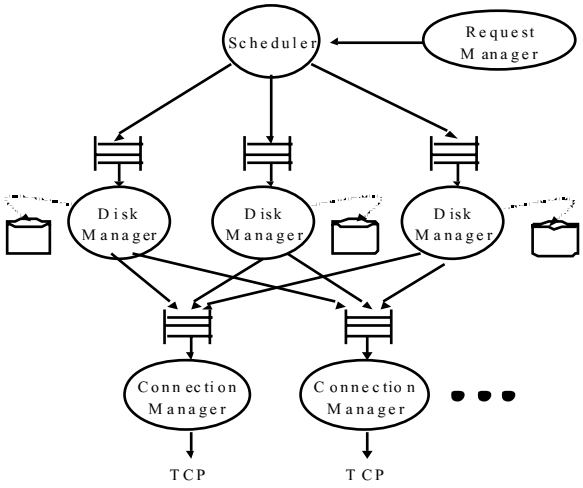

Figure 2: Architecture of Nova

VOD server has also been implemented for comparison. Performance evaluations showed that the streaming capacity is 33 streams for Nova and is only 21 streams for the user level VOD server. The performance improvement is 57%.

3.1 Nova Architecture

Nova consists of four components: *request manager*, *scheduler*, *disk manager*, and *connection manager* as shown in Figure 2. The request manager waits for any request from clients. When connection request arrives, the request manager determines whether resources are available for this request. If it passes the admission control, the request manager then registers this client to the scheduler and creates a connection manager for this new connection. The connection manager is destroyed when the connection is closed.

Nova uses the server push model [11] , so it automatically reads data needed in the next cycle and sends the data read from the previous cycle to the clients periodically. This periodically is activated by the scheduler. The scheduler is awakened once per second. On awakened, it calculates the

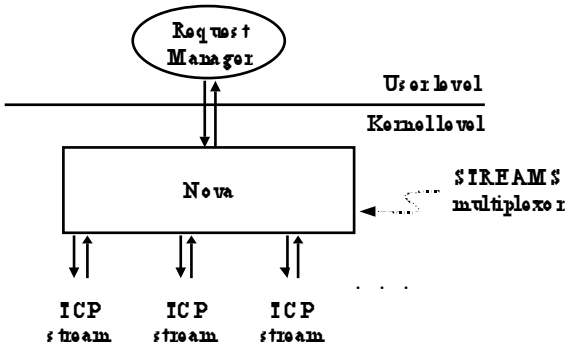

Figure 3: Implementation of Nova.

data blocks needed in the next cycle for all the registered clients and passes the blocks requests to the corresponding disk managers. The disk scheduling algorithm used in Nova is SCAN.

After data has been read from disk, the disk manager then passes the requests to the connection manager that uses the TCP protocol to send the media data to client.

3.2 Implementation

Nova is implemented as a STREAMS multiplexor on UnixWare™ 2.0, a descendant of SVR4/MP. A STREAMS multiplexor is a kind of STREAMS driver. Applications can link or unlink a stream under the multiplexor by using the ioctl() system call. As shown in Figure 3, all the components of Nova are implemented in the multiplexor except the request manager. When a new connection is established, a stream is constructed by the TCP protocol and is linked by Request manager under the Nova. The stream is unlinked from the multiplexor after a session is over.

In STREAMS mechanism, message passing is used in communication between STREAMS modules or drivers. A message contains three parts: message block descriptor, data block descriptor, and data buffer. So Nova must convert the data read from disks into messages and then send them down to the TCP/IP streams linked beneath it. Data are directly read from disk into the data buffer part of a message. Nova manages its own pool of free buffers. The data buffer used in a message is allocated from this pool and is freed to the pool when a message is sent and acknowledged by the client.

Data striping is used when storing media data in disks. Media data is chopped into fixed-sized blocks that are stored in data disks in round-robin order. Metadata is stored on a separate disk that uses UNIX file system.

4. Evaluations

This research has completely met the goals of this project. The research results have been published on [2][3]. The following describes the performance evaluation of our server.

We measure the performance of Nova in two different environments. In the first environment, there is only one client machine that is connected directly to the server. The client machine forks many processes and each process emulates a real world client. These processes do not decode and display the MPEG-I stream sent from the server. It simply reads the data and then drops it. In the second environment, we use six client machines that are connected to the server via fast ethernet switch. Three of the client machines run a software decoder to display MPEG-I stream in real-time. This decoder exploits the MMX instructions of Pentium to speed up the decoding and displaying process. The other client machines run the same program as in the first environment.

Six kinds of MPEG I media data are used in the experiments. Their bit rates range from $0.983*10^6$ to $2.20*10^6$ bits/sec. For the purpose of comparison, we also implement

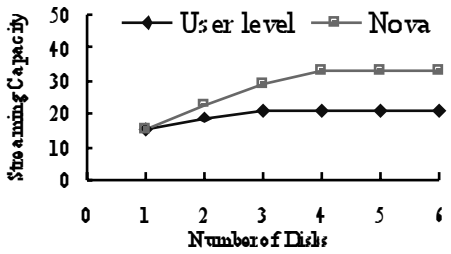

Figure 4: The Streaming Capacity in First Environment

an equivalent user-level server. The server components are implemented as threads within a single process.

The *streaming capacity* of the server is defined as the number of clients that can be serviced simultaneously without violating the continuous requirement, and is measured by having the client simultaneously issue *n* connection requests to the server and check if the server can successfully support that number of clients.

In the first environment, the server runs on top of UnixWare™ 2.0 on Pentium 133 with 32-Mbyte RAM, one AHA-3940 (a PCI SCSI adaptor which has two SCSI buses on it), and one Intel EtherExpress™ PRO/100 (a PCI FastEthernet adaptor). All the experiments are measured by connecting a dedicated Ethernet line between the client and the server.

Figure 4 shows the results of measurements in the first experiment. The number of disks in the system varies from 1 to 6. Both systems have the same streaming capacity when one disk is used. This is because the system bottleneck lies in the disk bandwidth. As the number of disk increases, the aggregate bandwidth of disks also increases, which improves streaming capacity. However, after the aggregate bandwidth of disks exceeds the memory bandwidth, memory bandwidth becomes the bottleneck of the system. Because the effective memory bandwidth of Nova is higher than the user-level implementation, the bottleneck shifts from disk to memory bandwidth when the number of disks is 4 in Nova whereas 3 in the user-level implementation. When the number of disks is 6, the streaming capacity is 33 streams for Nova and 21 streams for user-level implementation. Performance gap is 57%.

In the second environment, except the CPU and mainboard, the server runs with the same hardware equipment. The CPU is Cyrix PR166+ for the server. Three of client machines are Pentium MMX 200. The other client machines are Cyrix PR166+. The fast ethernet switch used is Cisco system's workgroup catalyst 3200. The streaming capacity is showed in Figure 5. When the number of disks is 6, the streaming capacity is 28 streams for Nova and 23 streams for user-level implementation.

5. References

[1] C. Y. Cheng, C. H. Wen, M. H. Lee, F. C. Wang, and Y. J. Oyang, "Effective Utilization of Disk Bandwidth for Supporting Interactive Video-on-Demand," IEEE Transactions on Consumer Electronics, Feb. 1996.

[2] W. Y. Chung and R. C. Chang, "An Implementation of the CMSS VOD Server," In 1997 Workshop on Consumer Electronics: Digital Video and Multimedia and NSC Annual Seminar on HDTV Research and Development, Oct. 1997.

[3] W. Y. Chung, M. L. Chiang, and R. C. Chang, "Reducing the Redundant Memory Copying in

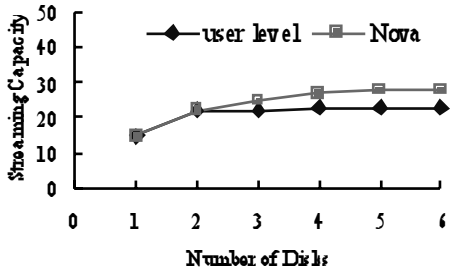

Figure 5: The Streaming Capacity in Second Environment

VOD Server," In 1998 IEEE International Symposium on Consumer Electronics (ISCE'98), Oct. 1998.

[4] C. Dalton, G. Watson, D. Banks, C. Calamvokis, A. Edwards, and J. Lumley, "Afterburner," IEEE Network, July 1993.

[5] R. L. Haskin and F. B. Schmuck, "The Tiger Shark File System," Proc. COMPCON'96, Feb 1996, pp. 226-231.

[6] A. Heybey, M. Sullivan, and P. England, "Calliope: A Distributed, Scalable Multimedia Server," USENIX Conference, Jan. 1996, pp. 75-86.

[7] BJ Murphy, S Zeadally, CJ Adams, "An Analysis of Process and Memory Models to Support High-Speed Networking in a UNIX Environment," Proc. USENIX Winter Conference, Jan. 1996, pp. 239-251.

[8] G. Neufeld, D. Makaroff, and N. Hutchinson, "Design of a Variable Bit Rate Continuous Media File Server for an ATM Network," IS&T/SPIE Multimedia Computing and Networking, Jan 1996.

[9] Oracle Corp., "Oracle Video Server™ System Tour for the LAN Environment," 1996.

[10] J. Pasquale, "I/O System Design for Intensive Multimedia I/O," IEEE Workshop on Workstation Operating Systems, April 1992, pp.29-33.

[11] S. S. Rao, H. M. Vin, and A. Tarafdar, "Comparative Evaluation of Sever-push and Client-pull Architectures for Multimedia Servers," 6th Networks and Operating Systems Support for Digital Video and Audio, April 1996.

[12] H. Tezuka, and T. Nakajima, "Simple Continuous Media Storage Server on Real-Time Mach," USENIX Conference, Jan. 1996, pp. 87-98.

[13] M. Vernick, C. Venkatramani, and T. C. Chiueh, "Adventures in Building the Stony Brook Video Server", The 4th ACM International Multimedia Conference, Nov. 1996, pp. 287-295.