Parallel Failure Recovery Techniques in Cluster-based Media Servers

Joahyoung Lee, Heonguil Lee, and Inbum Jung*
Department of Computer, Information, and Communications
Kangwon National University, Korea

Jinnie4u@.kangwon.ac.kr, hglee@kangwon.ac.kr, ibjung@kangwon.ac.kr(corresponding author)

Abstract

For large scale VOD service, cluster servers are spotlighted to their high performance and low cost. A cluster server consists of a front end node and multiple backend nodes. Though the increase of backend nodes provides the more QoS streams, the possibility of failures in backend nodes is proportionally increased. The failure causes not only the stop of all streaming services but also the loss of current playing positions. In this paper, when a backend node becomes a failed state, recovery mechanisms are studied to support the unceasing streaming service. Without considering the characteristic of cluster-based servers and MPEG media, the basic RAID techniques causes the performance bottleneck in the internal network and also results in the inefficiency CPU usage of backend nodes. To address these problems, a new failure recovery mechanism is proposed based on the pipeline computing concept. The proposed method distributes the network traffics needed in the process of recovery and utilizes the available CPU computing power of backend nodes. In experiments, it provides the improved performance of cluster-based VOD servers as well as the continuous streaming media service in the failed state of a backend node.

Key words:
streaming media, Autonomic recovery, pipeline computing, cluster servers

1. Introduction

Recent advanced computer and communication technologies have provide economically feasible multimedia services such as VOD(Video-On-Demand), digital library and e-learning. In these multimedia services, since the video data are given a great deal of weight due to the preference of clients to lively motion expressions, the VOD service is the most prominent multimedia application [1, 2]. In contrary to traditional file servers, VOD servers are subject to real-time constraints while storing, retrieving and delivering the movie data into the network. Since the ceasing and jittering streaming videos are meaningless for VOD clients, the streaming media should be supplied for each client within designated QoS(Quality Of Service) criterion. To support the QoS, servers must be able to continuously deliver video data at a constant interval to VOD clients. And also, even in the failure of server components, the streaming service should be re-continued within the human acceptable MTTR (Mean Time To Repair) value [3, 4].

Recently cluster server architecture has been exploited in the areas of Internet Web, database, game and VOD servers [9]. It has an advantage of the ratio of performance to cost and is easily extended from the general PC equipments. The cluster server architecture usually consists of a front-end node and multiple backend nodes. It is traditional decentralized systems to treat specific application fields. When this architecture model is used as VOD servers, the video data are distributed into several backend nodes. The performance of storage devices could be achieved accordingly as the number of backend nodes increase. However, even if the cluster server can be scaled by just adding new backend nodes, the probability of the failure of nodes also increases in proportion to the number of backend nodes. The fault of nodes causes not only the stop of all streaming service but also the loss of the being serviced positions of all playing movies. In the VOD service, since QoS streams are guaranteed to all clients even in the failure of nodes, the recovery mechanisms are necessary for dealing with a realistic VOD service. In this paper, the autonomous recovery mechanisms in cluster-based VOD servers are studied to support QoS streams while a backend node becomes a failure state.

To study the failure events during the actual VOD service, we implement a cluster-based VOD servers composed of general PCs and adopts parallel processing for MPEG media to support a large number of clients. From the implemented VOD server, we evaluate a legacy recovery system composed of the advantages of RAID-3 and RAID-4 algorithms. These RAID levels are known as providing very high speed data transfer rate suitable for the video streaming. However, this recovery system causes the performance bottleneck on the input network

*Corresponding Author
This research was financially supported by the Ministry of Coomerce, Industry and Energy(MOCIE) and Korea Industrial Technology Foundation(KOTEF) through the Human Resource Training Project for Regional Innovation

of the recovery node and shows the inefficiency CPU usage in backend nodes. To address these issues, we propose a new failure recovery system based on pipeline computing over all survived backend nodes. The proposed system autonomously distributes not only the workloads composed of exclusive-OR operations but also the network traffics across all backend nodes. Since all survived backend nodes are participated in the total recovery operations, the proposed method provides the improved performance of cluster-based VOD servers as well as the continuous streaming media service even in the failure state of a backend node.

The rest of this paper is organized as follows. Section 2 describes related work. Section 3 explains our cluster-based VOD servers and the management of video blocks. In section 4, a new recovery strategy by the pipeline computing is proposed to utilize the resources of backend nodes. Section 5 describes experimental environments. In Section 6, the results of performance evaluation are shown. Section 7concludes the paper.

2. Related Work

Many researches were undertaken for VOD systems to provide a stable service to more clients under the various client requirements and the limited resources [6, 7, 8]. For commercially successful VOD service, in the case where the partial failure state occurs, the QoS streams could be substantially provided to clients within the limited MTTR value. The human acceptable MTTR value is a mandatory condition. It is one of the important QoS metrics for the outstanding VOD service. There has been much research in the area of fault tolerance for file, database and web servers. However, the streaming media have its intrinsic characteristic such as real time specifications. There have not been enough researches to guarantee the QoS streams without ceasing and jittering even in the partial failure state of VOD servers.

Based on the mirror concept, several researches were performed for recovering the failed storage systems [9, 10]. The Tiger video server was implemented on the mirror based storage system for VOD service [11]. The RMD(Rotational Mirrored Declustering) techniques are suggested to recover the failed disks or individual nodes[12]. However, these mirror based approaches have the inefficient usage of disk storages and also aggravate the burden of the recovery node.

The RAID(Redundant Array of Inexpensive Disks) mechanisms are usually exploited to recover the failed disks or parallel nodes in clustered servers. In particular, the RAID-3, 4, 5 mechanisms are based on the parity based recovery algorithm [10, 13, 14]. In the RAID-3, the data block is striped and written on the data disks. Due to the fine striped unit, the numerous numbers of disk assess are necessary for retrieving the large video blocks. It may degrade the data transfer rate from disks. In the RAID-4, each entire block is written onto a disk. Since this method provides the coarse striped unit, the disk performance can be improved by retrieving once larger blocks every disk access. Both RAID-3 and RAID-4 has its own parity disks used in the recovery operations. In the RAID-5, the parity blocks in the same rank are spread out in distributed disks. After a backend node fails, all remaining nodes perform the total recovery operations individually. The parity blocks are distributed into all disks so that the network traffics between nodes highly increase to exchange those blocks. Under this working environment of RAID-5, since it is difficult to gauge the actual load of backend nodes in real time, the steady QoS streams are not guaranteed to VOD clients.

Recently, the cluster server architecture has been utilized for various areas due to its low cost and high performance [5]. In particular, the VOD service has its intrinsic property to provide the streaming media to all clients in real time environment. Even if the failure of disks or backend nodes happens, the irregular ceasing and jittering streaming media should be solved within the human acceptable time period [3, 4]. However, until now, the recovery mechanism has not been studied deeply in the cluster-based VOD server. In particular, for the more commercialized VOD service, the recovery system should be studied on the characteristics of streaming media.

For the reliable VOD service, our research was focused on both improving the performance of the recovery system and achieving the better MTTR value in the failure state.

3. Cluster-based VOD Server

3.1 Architecture of VODCA

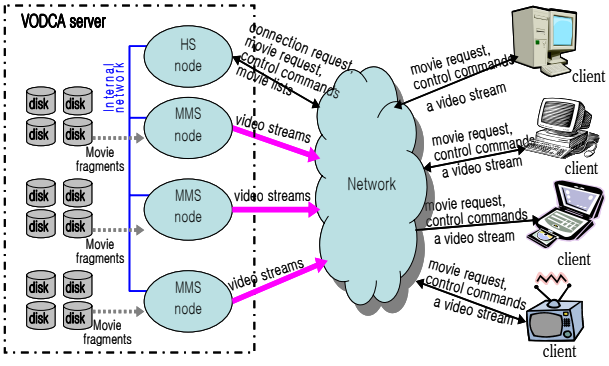

Figure 1: Architecture of VODCA server

For large scale VOD services, we implemented a cluster-based VOD server called as VODCA (Video On Demand on Clustering Architecture)[18]. The VODCA consist of a front-end node named as HS(Head-end Server) and several backend nodes known as MMS(Media Management Server). Figure 1 shows the architecture of our VODCA server and various VOD clients. Clients are working together with HS and MMS nodes. Throughout the internal network path between a HS node and MMS nodes, they exchange the working states and internal commands each other.

The HS node not only receives clients' requests but also manages MMS nodes to support QoS with the admission control. When new MPEG movies are enrolled, the HS splits them and distributes them into each MMS node. The MMS nodes transmit their stored movie fragments to clients under the supervision of the HS node. Each MMS node sends the present working status to the HS node periodically. This message operates as a heartbeat protocol between MMS nodes and a HS node.

3.2 Stripping of Video Blocks

The cluster server easily provides the parallel computing environment based on the high speed network among composed backend nodes and independent working spaces of each server. To apply parallel processing for MPEG movies, we stripe the movie files according to the defined granularity policy. After striping, the movie file is partitioned into many fragments and they are distributed into backend nodes with their header information. To exploit MPEG media characteristics in parallel processing, we use GOP size as a striping unit. Since each GOP has approximately equal running time in MPEG streams, the MPEG movies are split into GOPs and distributed into each node with their sequence number and size.

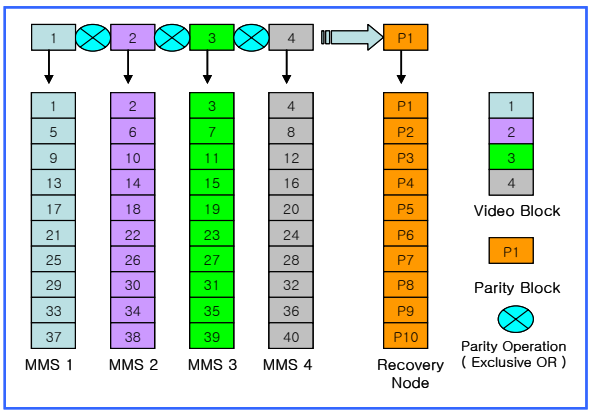

Figure 2: Striped video blocks and parity blocks

The RAID mechanism is usually used to recover the inaccessible data in the failed disks. Various kinds of RAID levels are existed in commercial areas. In our research, RAID-3, 4 levels are used for the basic recovery system in our VODCA server. These levels are suitable for the video streaming service by supporting very high speed data transfer rates [13, 14].

In the RAID-4 algorithm, the parity block for same rank blocks should be generated on writes and recorded on the parity disk. Figure 2 shows the distributed video blocks in each MMS node and the parity blocks stored in a recovery node. For example, the parity block P1 is generated by the exclusive-OR operation to video blocks 1, 2, 3, 4 and stored into the recovery node.

The RAID-4 reduces the number of disk assess but it causes the problem in the process of aggregating the video blocks to recover the failure state. To address this problem, we use the RAID-3 algorithm when the video blocks are transferred and recovered. As mentioned above, since the RAID-3 supports the small stripe units during the recovery process, the large video blocks retrieved from the disk of MMS nodes are partitioned as small data units and they are transferred into the recovery node. From these small data, the recovery node gradually performs exclusive-OR operation to rebuild the failed video blocks.

3.3 Recovery System on RAID 3,4

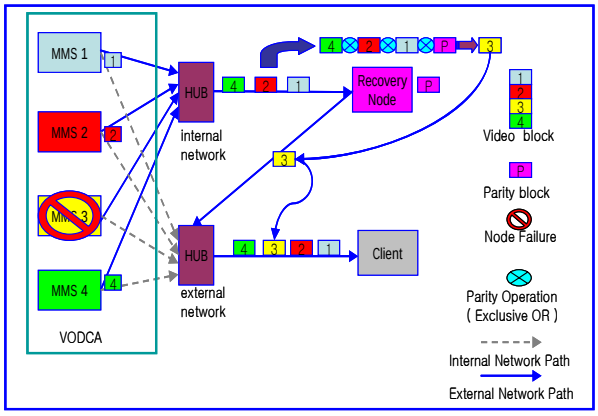

Figure 3: Architecture and video block flows in RS-BRM

Figure 3 shows the architecture of the recovery system operated on basic RAID-4, 3 mechanisms [13,14]. We denote this recovery model as RS-BRM (*Recover System based on Basic RAID Mechanisms*) and adopt on our VODCA as a basic recovery system. As shown in this Figure, two network paths are existed. One is an external network for connecting between the MMS nodes and the VOD clients. Another network is an internal network

path installed between all MMS nodes and a recovery node.

When all MMS nodes are operated normally, MMS nodes transmit their stored video blocks to clients directly through the external network path. On the other hand, when a MMS node fails to run, the survived MMS nodes are sent the video blocks to both the clients and the recovery node. The recovery node regenerates the failed video blocks with the video blocks received from the MMS nodes and the parity blocks stored in its own disks. Since both MMS nodes and the recovery node use their internal network path for these recovery operations, the external network bandwidth just takes charge of streaming services to the VOD clients. For example, as shown in Figure 3, when the MMS 3 node fails, the recovery node regenerates the video block 3 by executing the exclusive-OR operation with the received video blocks 1, 2, 4 and its own parity block. Since the regenerated video block 3 is sent to the corresponding client via the external network path, the streaming media service can be continued.

However, the performance of RS-BRM suffers from the bottleneck of input network on the recovery node because all video blocks stored in the survived nodes should be transmitted into a recovery node at the same time.

4. Pipeline Computing to Recover Failed Video Blocks

4.1 Architecture

To address the internal network bottleneck of the RS-BRM, we propose a new recovery system based on the concept of pipeline computing. It is denoted as RS-PCM(***R***ecovery ***S***ystem based on ***P***ipeline ***C***omputing ***M***echanism). The proposed method distributes the network traffics needed in the process of recovery into all survived MMS nodes and fully utilizes the available CPU time of MMS nodes.

In the parity based RAID algorithms, the exclusive-OR operations for video blocks take a great part of computing loads. To rebuild the video blocks stored in the failure MMS node, several exclusive-OR stages are necessary sequentially. For each stage, the two blocks are needed to execute the exclusive-OR operation at a time. However, the final result is independent of the order in which pairs are processed. Therefore, even if the exclusive operations for all video block pairs are executed in out of order in sequence, the same results are achieved. Based on this characteristic, we distribute the exclusive-OR stages into MMS nodes. It can solve the network congestion problem on the input network port of recovery node. In addition, the available CPU time of MMS nodes can be utilized by distributing the exclusive-OR operation into each MMS node.

Figure 4 shows the architecture of RS-PCM and the flow of video blocks in the VODCA server. As shown in this figure, the RS-PCM not only distributes the network traffics for recovery processes but also spreads the exclusive-OR operations over all survived MMS nodes. In RS-PCM, when a MMS node fails, all survived MMS nodes do not send their video blocks to the recovery node directly. On the other hand, each MMS node transmits the original video block or it own exclusive-OR result block to their neighbor MMS node. A MMS node performs its own fraction of exclusive-OR operation with both a video block retrieved from its local disk and the other block received from its neighbor MMS node.

In RS-PCM, the blocks received from its neighbor MMS node may be an original video block stored in the disk or the result of exclusive-OR operation executed by the neighbor MMS node. The results are sent to the neighbor MMS node successively such as the pipeline process in the instruction level [15].

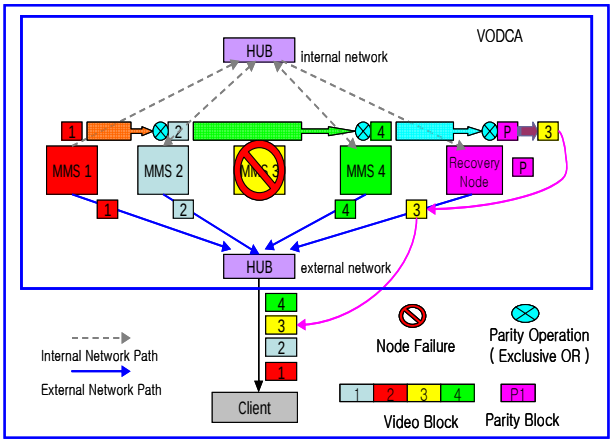

Figure 4: Architecture of RS-PCM

Finally, the recovery node performs the last exclusive-OR operation with its parity block and the aggregate result of all MMS nodes. As a result, the video block stored in the failed MMS node is rebuilt. After that, the regenerated video block is transmitted to clients through the external network path. For example, as shown in the Figure 4, when the MMS node 3 fails, the MMS 1 node sends the video block 1 to the MMS 2 node. The MMS 2 node performs the exclusive-OR operations with both the video block 1 and the block 2. After that, the result is sent to the MMS 4 node to perform the exclusive-OR operation with the video block 4. Finally, after the exclusive-OR operations for all survived video blocks are finished, the result is sent to the recovery node. The

recovery node regenerates the video block 3 throughout the exclusive-OR operation with the parity block.

4.2 Characteristics of the RS-PCM

The Figure 5 shows the recovery operations according to the pipeline concept of the RS-PCM. As shown in this Figure, the issuing of at least one retrieving or transmitting or executing exclusive-OR operation every cycle is like to the pipeline technique in the parallel processing of instructions [15]. This parallel processing for recovering the failed blocks induces a better performance. As shown in this Figure, the failed MMS 3 node has video block 3, 7, 11, 15, 19, 23. These blocks are regenerated in the recovery node every cycle according to the pipeline computing.

The recovery node in RS-PCM executes exclusive-OR operation just one time for each cycle and sends the result to the client. The RS-PCM distributes not only the computation load for exclusive-OR operations but also the network traffics into all MMS nodes. The input network traffic of recovery node is equal to the output traffic of one MMS node. Each MMS node has the same amount of network traffics as its output. The recovery node and MMS nodes could utilize the full capacity of the internal network path. If the n-1 MMS nodes are survived and output traffic is m, only the m network traffics exist on the input interface of the recovery node. Due to this characteristic, the recovery node does not suffer from the bottleneck phenomenon in the input network port.

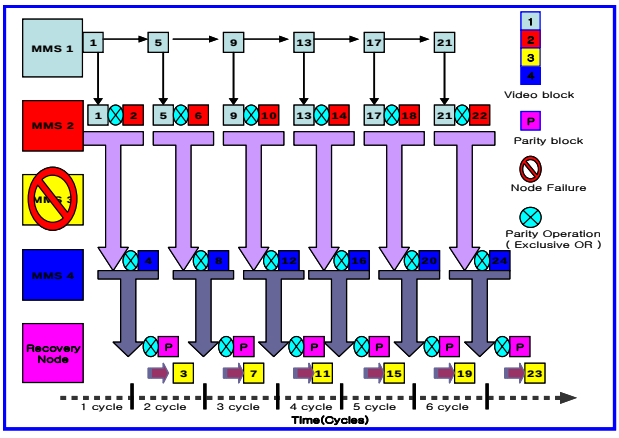

Figure 5: Recovery steps on pipeline concept

5. Experimental Environment

5.1 System Configuration

The VODCA server for our experiments consists of a HS node, 4 MMS nodes and a recovery node. Each node operates on the Linux operating system. The MMS nodes, HS node and clients are connected via a 100 Mbps Ethernet switch. All MMS nodes and the recovery node are also connected via the internal network path constructed by a 100 Mbps Ethernet switch.

All applications included the system administrative tools of the HS node are developed on Qt, C and C++ libraries. Table 1 shows the hardware components for each MMS node in the VODCA system. Table 2 shows the detail specification of movies used in our experiments. They are MPEG-2 movies and have enough running time to evaluate their performance in our system.

Table 1: Specification of MMS nodes and a recover node

CPU	Intel Pentium 4, 3.0 GHz
Memory	1GByte DDR
Disk	Segate Baracuda ATA IV 40GB 7200RPM x 2
OS	RedHat 7.3 (Kernel 2.4.18)
Network	100 Mbps Fast Ethernet, 100Mbps Ethernet Switch with 24 ports

Table 2: Specification for experimental movies

Movie name	John Q	Ice Age
Frame size(H x V)	352 x 288	352 x 288
Frame rates(number/sec)	25	25
Bit rates(bps)	1,437.6	1,437.6
Running time(Minutes)	110	85
GOP size(Kbytes)	124.1	120.8

5.2 Load Generator and Performance Metrics

We use the *yardstick program* to measure the performance of our cluster-based VOD server [16]. The yardstick program consists of the *virtual load generator* and the *virtual client daemon*. The virtual load generator is located in the HS node and generates client requests based on the Poisson distribution with $\lambda = 0.25$[17]. These requests are sent to each MMS nodes. After that, all MMS nodes concurrently begin streaming media services for satisfying the clients' demand.

The virtual client daemon locates in test-bed PCs for clients. It plays the role of receiving movie data from MMS nodes. Based on MPEG-1, 2 specifications, we assume that a QoS stream requires 1.5 Mbps network bandwidth. To support this QoS criterion, the virtual

client daemon measures the time elapsed for receiving 1.5Mbits of data. If the elapsed time is below 1 second, the virtual client daemon remains in an idle state until 1 second period had passed. After exhausting this remaining time, the daemon wakes up again and begins to receive the next media data. This waiting process makes the virtual client daemon act as a real client.

6. Performance Evaluation

6.1 Network Traffics

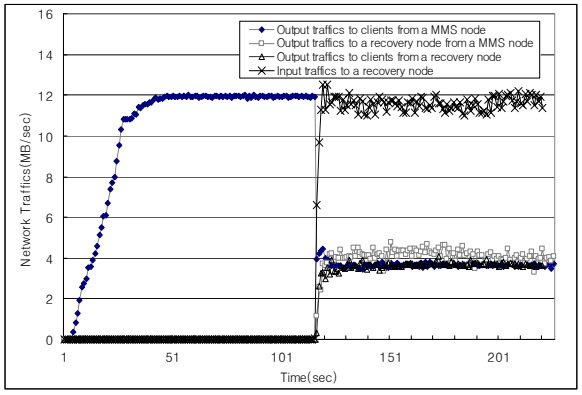

Figure 6: Network traffics in RS-BRM

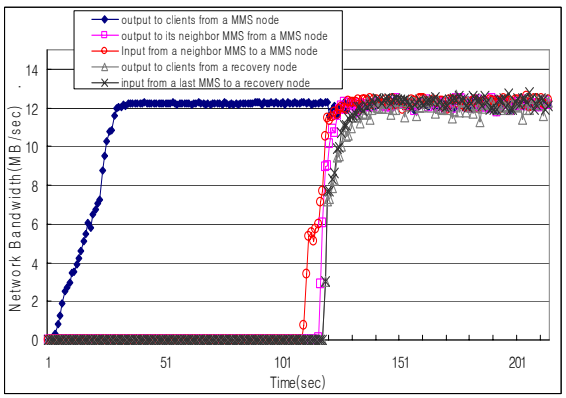

Figure 7: Network traffics in RS-PCM

The Figure 6 shows the network traffics in a MMS node and the recovery node when the 12MB/sec traffics are loaded. The output network traffic of 12MB/sec represents that 4 MMS nodes provide 256 streams(12M * 8 bit * 4 MMS nodes / 1.5 Mbps). The failure of a MMS node takes place at the 120 second of the time line. As shown in this Figure, after a failure occurs, the network traffics from a MMS node to clients are decreased to 4MB/sec rates from 12 MB/sec rates. However, as marked by the x legend, the amount of input traffics of the recovery node reaches about 12 MB/sec.

The reason is that when the recovery node can not receive any more video blocks due to its input network bottleneck. Under the 12 MB/sec traffics per a MMS node, after a MMS node fails, the network traffics from remained 3 MMS nodes to the recovery node are reaching to 36MB/sec. However, as mention in the Table 1, the input network capacity of recovery node is limited to 12.5MB/sec(100Mbps). Thus, the input port of a recovery node suffers from the bottleneck phenomenon due to the overwhelmed video data. The clogged video blocks on the input network port of recovery node can not be used in the process for rebuilding the failed video blocks. Therefore, to balance with the processing speed of the recovery node, the MMS nodes also automatically decrease the amount of output traffics to their clients. As a result, 12 MB/sec traffics from a MMS node converge into 4 MB/sec traffics. The 4MB/sec traffics mean that only 85 clients can be supported(4M *8 bit * 4 MMS nodes / 1.5 Mbps).

The Figure 7 shows the network traffics in a MMS node and the recovery node when the 12MB/sec traffics are loaded. The failure of a MMS node takes place at the 120 second of the time line. As shown in this Figure, even if a failure occurs, the network traffics from a MMS node to clients are continuously sustained as 12 MB/sec rates. The 12MB/sec traffics mean that only 256 clients can be supported in the failed state(12M *8 bit * 4 MMS nodes / 1.5 Mbps). In the partial failed state, when compared to the RS-BRM, the RS-PCM provides 3 times unceasing streams in the same working environment.

The square legend mark in the Figure 7 represents the amount of output traffics toward the neighbor MMS node. In the RS-PCM, if the current MMS node is not the last MMS node, it transmits its own video blocks or the result blocks executed the exclusive-OR operation to its neighbor MMS. From the circle legend mark, we find that the amount of input traffics from the neighbor MMS node is almost equal to that of its own output traffics. The amount of input traffics from the last MMS node reaches to the 12MB/sec rates so that the recovery node also can rebuild the video blocks as much as 12 MB/sec rates. After that, the recovery node transmits the recovered video blocks to clients. From the triangle legend mark of the Figure 7, we can confirm that the output traffics of rebuilt blocks in the recovery node also get to the 12MB/sec rates.

As a result, since the RS-PCM distributes the network traffics among all MMS nodes and utilizes the available CPU resources of MMS nodes, it provides the more number of unceasing QoS streams in a partially failed cluster-based VOD server.

7. Conclusion

To study the recovery system in the actual VOD service, we implement a cluster-based VOD servers composed of general PCs and the internal network path. From the implemented VOD server, the RS-BRM was designed with the advantage of RAID-4 in disk retrieving speed and the advantage of RAID-3 in effective memory usage. However, in the RS-BRM, we found that the input network path of a recovery node is easily saturated with the video blocks transmitted from the survived MMS nodes. The delay of video blocks transmitted to the recovery node caused the reduction of the number of clients and the aggravation of the quality of video streams. In addition, the RS-BRM showed the inefficiency CPU usage of MMS nodes. In this method, Since the MMS nodes simply performed the retrieving and transmitting of their own video blocks, the average CPU utilization was measured below 10%.

To address these issues, we proposed the RS-PCM based on the pipeline computing over MMS nodes and a recovery node. In the RS-PCM, the recovery node rebuilt the video blocks stored in the failed MMS node and sends them to clients just one time for each cycle. This mechanism is similar to the pipeline process of instructions. The RS-PCM made efficient use of the available CPU resource of MMS nodes so that all survived MMS nodes were participated in the recovery procedures to rebuild the impaired video blocks. Based on this pipeline computing, the RS-PCM distributed not only the computation load for exclusive-OR operation but also the network traffics across all MMS nodes. From our experiment, we observed that the input network traffics of a recovery node were the same amount of output traffics induced by the last MMS node. Even in the failure state of a MMS node, the RS-PCM showed the improved performance by providing at least 3 times QoS streams when compared to the RS-BRM.

References

[1] Dinkar Sitaram, Asit Dan, "Multimedia Servers: Applications, Environments, and Design," Morgan Kaufmann Publishers, 2000.

[2] http://www.mpeg.org

[3] Armando Fox, David Patterson, "Approaches to Recovery Oriented Computing," IEEE Internet Computing, Vol. 9, no. 2, pp.14-16, 2005.

[4] Dong Tang, Ji Zhu, Roy Andrada, "Automatic Generation of Availability Models in RAScard," IEEE International Conference of Dependable Systems and Networks, June 23-26, pp. 488~494, 2002.

[5] http://www.ieeetfcc.org

[6] Nabil J. Sarhan, Chita R. Das, "Caching and Scheduling in NAD-Based Multimedia Servers," IEEE Transactions on PARALLEL AND DISTRIBUTED SYSTEMS, Vol.15, No.10, pp.921~933, 2004.

[7] Sooyong Kang, Heon Y. Yeom, "Modeling the Caching Effect in Continuous Media Servers," Multimedia Tools and Applications, 23(3), pp 203-224, 2003.

[8] Prashant J. Shenoy, Pawan Goyal, Harrick M. Vin, "Data Storage and Retrieval for Video-on-Demand Servers," IEEE Fourth International Symposium on Multimedia Software Engineering (MSE'02),pp.240-245, December 2002.

[9] Jamel Gafsi, Ernst W. Biersack,"Modeling and Performance Comparison of Reliability Strategies for Distributed Video Servers,"IEEE Transactions on Parallel and Distributed Systems, Vol. 11, No. 4, pp.412~430, 2000.

[10] J. Gafsi and E.W. Biersack, "Data Striping and Reliablity Aspects in Distributed Video Servers," In Cluster Computing: Networks, Software Tools, and Applications, 2 (1): pp. 75~91, February 1999.

[11] W.J. Bolosky, R.P. Pitzgerald, J.H. Draves,"Distributed schedule management in the Tiger video fileserver,"Proceedings of the sixteenth ACM symposium on Operating systems principles, Saint Malo France, October 05-08. pp. 212~223, 1997.

[12] T. Chang, S. Shim, and D. Du, "The Designs of RAID with XOR Engines on Disks for Mass Storage Systems,"IEEE Mass Storage Conference, March 23-26, pp. 181~186, 1998.

[13] A. Merchant and P.S. Yu, "Analytic modeling and comparisons of striping strategies for replicated disk arrays,"IEEE Transactions on Computers, vol.44, Mar., pp.419~433, 1995.

[14] M. Holland, G.Gibson, and D. Siewiorek, "Architectures and algorithms for on-line failure recovery in redundant disk arrays,"Journal of Distributed and Parallel Databases, vol.2, pp. 295~335, 1994.

[15] David A. Patterson and John L. Hennessy, "Computer Organization & Design,"PP. 392~490, Morgan Kaufmann, 1998.

[16] Brian K. Schmidt, Monica S. Lam, J. Duane Northcutt, "The interactive performance of SLIM: a stateless, thin-client architecture," ACM SOSP'99, pp. 31~47, 1999.

[17] Jung-Min Choi, Seung-Won Lee, Ki-Dong Chung, "A Muticast Delivery Scheme for VCR Operations in a Large VOD System,"8th IEEE International Conference on Parallel and Distributed Systems, June 26-29, pp. 555~561, 2001.

[18] Dongmahn Seo, Joahyoung Lee, Inbum Jung, "Resource Consumption-Aware QoS in Cluster-based VOD Servers," Journal of Systems Architecture, Volume 53 , Issue 1, pp. 39-52, 2007