A Systematic Scheme to Resolve QoS Dissatisfaction for Storage Cluster

Young Jin Nam, Chanik Park[†]

School of Computer and Information Technology
Daegu University
Kyungbuk, Republic of Korea
yjnam@daegu.ac.kr
[†]Department of Computer Science and Engineering/PIRL
Pohang University of Science and Technology
Kyungbuk, Republic of Korea
cipark@postech.ac.kr

Abstract. This paper addresses the types of QoS dissatisfaction caused by imbalance of the initial I/O workload pattern and storage performance across multiple storage servers in a storage cluster. It next proposes a systematic scheme to resolve the QoS problem that periodically monitors the QoS satisfaction level, analyzes the causes of the QoS problem, and performs data migration based on the analysis result. Finally, it verifies the effectiveness of the proposed scheme under a simulation environment under the different types of QoS dissatisfaction.

1 Introduction

A storage cluster typically consists of storage clients, virtual disks, and storage servers attached to a high-speed SAN. Each storage client distributes and accesses its data across multiple storage servers through a storage virtualization layer called a virtual disk. Storage clients represent various types of I/O applications that demand an underlying storage service, such as traditional file systems, cluster/SAN file systems, database applications, etc. Virtual disks, each of which is assigned to at least one storage client, represent logical volumes that map user data onto physically dispersed storage servers. Storage servers represent SAN-attached disk arrays or JBODs (Just Bunch of Disks). It mainly processes I/O requests arrived from virtual disks in a certain manner.

Large-scale storage systems like a storage cluster increase the chances that storage clients (or virtual disks) share the same storage server. Each storage client may require a different storage service, called storage Quality of Service (QoS); that is, each storage client requires receiving a guaranteed storage service, independently of the status of the I/O services in other storage clients. Unfortunately, the storage itself does not contain any feature of providing the storage QoS. Embedding QoS feature into a storage system needs to define storage QoS specifications [1], design a storage server to meet a given storage QoS specifications (requirements) [2, 3], and enforce the storage QoS requirements for each I/O request from different virtual disks (storage clients) [4]. Huang in [2] has proposed a QoS architecture called StoneHenge for a storage cluster that assures given QoS requirements of I/O performance.

The initial I/O workload patterns and storage performance that have been used for designing virtual disks are subject to change due to numerous reasons [7]. This implies that a virtual disk may not meet its QoS requirement due to the changes in the initial storage design information. In the case of a single storage server, the types of the changes include the increased I/O traffic and the degraded storage performance. The changes are typically resolved by redesigning the virtual disk with the changed information. The previous automatic storage design tools of Minerva [1] and Hippodrome [3] employed an iterative design loop to resolve QoS dissatisfaction on a single storage server. In the case of the storage cluster, the design tools need to be combined with the virtual disk mapping schemes proposed in StoneHenge [2].

Under a storage cluster, extra types of changes exist that are related to imbalance of the initial I/O workload and storage performance across storage servers that comprise a virtual disk. The imbalance of the initial I/O workloads is closely related to the variations of I/O traffic intensity across the storage servers within a virtual disk. Investigating an actual I/O workload gathered from `cello` [5] during 04/18–04/21 revealed that the storage system might experience QoS deterioration during 04/19–04/20 with the striped mapping and during the entire days with the linear mapping, assuming that the I/O requests are initially distributed over the storage servers in a uniform manner. Next, the imbalance of the storage performance can occur due to many reasons, such as loss of internal disks within a storage server, application re-installation or copy/remove operations, changes in I/O traffic of competing virtual disks that share the same storage server, etc.

2 The Proposed Scheme

The proposed scheme consists of a (storage) cluster-wide QoS monitor, a data migration planner, and data migration agents. Let us start by defining a QoS requirement.

QoS Requirement: The QoS requirement from a virtual disk i (briefly VD_i), denoted by Q_i can be represented as $Q_i = (SZ_i, IOPS_i^{targ}, RT_i^{targ})$, where SZ_i represents an average I/O request size, $IOPS_i^{targ}$ represents a target IOPS, and RT_i^{targ} represents a target response time. Under a storage cluster environment, multiple storage servers should assure a given QoS requirement of Q_i from a virtual disk in a cooperative manner. If each storage server of a virtual disk is designed to guarantee the given QoS requirement, the virtual disk can meet the given QoS requirement with an extremely high probability. However, this design approach suffers from an excessive use of storage resources. A better design approach demands to have greater knowledge of I/O workload patterns and storage performance over the virtual disk. As a result, it divides the servicing of the target IOPS into each storage server, thereby reducing the storage resources in use. Note, however, that the given target response time should remain unchanged at each storage server. Assuming that VD_i with Q_i distributes its data across N homogeneous storage servers, the given QoS requirement at each storage N storage servers, the given QoS requirement at each storage server j denoted by $Q_{(i,j)}$ is written as $Q_{(i,j)} = (SZ_{(i,j)}, IOPS_{(i,j)}^{targ}, RT_{(i,j)}^{targ})$, where $1 \leq j \leq N$. In addition, $SZ_{(i,j)}$, $IOPS_{(i,j)}^{targ}$, and $RT_{(i,j)}^{targ})$ should meet the following relationships: $SZ_{(i,j)} = SZ_i$, $IOPS_{(i,j)}^{targ} \leq IOPS_i^{targ}$, $IOPS_i^{targ} \leq \sum_{j=1}^{N} IOPS_{(i,j)}^{targ}$, and $RT_{(i,j)}^{targ} = RT_i^{targ}$. In case that perfectly balanced I/O workloads are issued to the storage servers within a virtual disk, we can minimize the usage of the storage resources for VD_i [2], where $IOPS_i^{targ} = \sum_{j=1}^{N} IOPS_{(i,j)}^{targ}$.

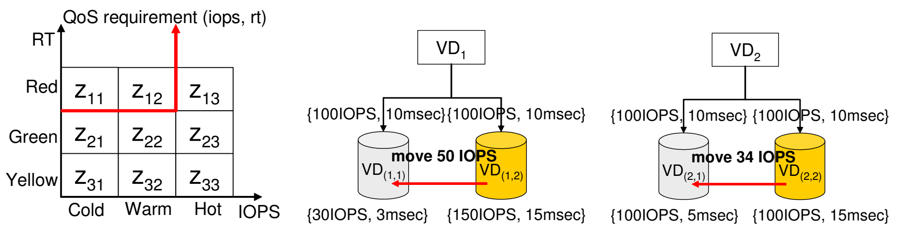
The Cluster-wide QoS Monitor: The cluster-wide QoS monitor (briefly QoS monitor) inspects the level of QoS satisfaction for each virtual disk and determines its state via a hierarchical QoS monitoring tree. The monitoring process is performed on a chunk of contiguous blocks at each storage server called BPAM. The BPAM stands for a base unit for performance monitoring and migration. The number of blocks under the control of each BPAM (shortly BPAM size) should not be too small to cause maintenance overhead and should not be too large to make it difficult to identify the cause of QoS dissatisfaction problem. Each BPAM includes the information of a virtual disk ID, a storage server ID, a start block address, a BPAM size, I/O requests per second (IOPS), response times (RT), and the target RT miss ratio (QoS requirement). The QoS monitor summarizes all the BPAMs of the same sub-virtual disk into the GPAM(Group of BPAMs) structure. Recall that a virtual disk consists of a set of sub-virtual disks.

A GPAM state can be classified into one of the nine combinations of the RT state and the IOPS state called GPAM-state chart, as shown in Figure 1(a). The RT states and IOPS states can be defined as follows. To begin, denote with $GPAM_{(i,j)}$ the j-th GPAM of VD_i. The QoS requirement of $GPAM_{(i,j)}$ is represented by $Q_{(i,j)} = (SZ^{targ}_{(i,j)}, IOPS^{targ}_{(i,j)}, RT^{targ}_{(i,j)})$. Three RT states exist in the GPAM that include `red`, `green`, and `yellow`. The state of `red` represents that the current target RT miss ratio $> \mathcal{MR}_H$, `green` represents that $\mathcal{MR}_L <$ the current target RT miss ratio $\leq \mathcal{MR}_H$, and `yellow` represents that the current target RT miss ratio $\leq \mathcal{MR}_L$. The parameters of $\mathcal{MR}_H$ and $\mathcal{MR}_L$ can be configured, such that $0 \leq \mathcal{MR}_L \leq \mathcal{MR}_H \leq 1.0$. In addition, three IOPS states exist in the GPAM that include `cold`, `warm`, and `hot`. The state of `cold` means that $0 \leq IOPS^{cur}_{(i,j)} \leq \alpha_c IOPS^{targ}_{(i,j)}$, `warm` means that $\alpha_c IOPS^{targ}_{(i,j)} < IOPS^{cur}_{(i,j)} \leq IOPS^{targ}_{(i,j)}$, and `hot` means that $IOPS^{cur}_{(i,j)} > IOPS^{targ}_{(i,j)}$. The parameter of α_c can be configured in the range of $(0, 1.0)$ depending on the current administration policy. Denote each cell (or combination) by an indexed zone z_{ij}, where i and j respectively represent a RT state and a IOPS state. In addition, we define $\mathcal{Z}^{NE}_i$ and $\mathcal{Z}^{E}_i$ as a set of non-empty zones and a set of empty zones for VD_i, respectively.

A virtual disk state can be determined based on the distribution of the GPAM states upon the GPAM-state chart. A virtual disk has more than one GPAM states, as it distributes data over multiple storage servers. Let us define three virtual disk states depending on whether a virtual disk satisfies a given QoS requirement and whether its storage resources are under-provisioned. The `well_designed` VD state represents that the virtual disk meets the given QoS requirement well with sufficient storage resources. The `imp_designed` VD state represents that the virtual disk does not guarantee the given QoS requirement even though it has sufficient storage resources. This undesirable VD state is attributed mainly by imbalance of the I/O workload pattern and storage performance with respect to their initial configurations. Finally, the `und_designed` VD state represents that the virtual disk does not guarantee the given QoS requirement because storage resources are provisioned insufficiently to the virtual disk. To resolve this QoS dissatisfaction, the virtual disk needs to allocate more storage resources if allowed. Next, the relationships between the virtual disk states and the distribution of the GPAM states can be given as follows: the `well_designed` VD includes $\{z_{11}, z_{12}, z_{13}\} \subset \mathcal{Z}^{E}_i$, or $\{z_{11}, z_{12}, z_{31}\} \subset \mathcal{Z}^{E}_i$ AND $\{z_{13}\} \subset \mathcal{Z}^{NE}_i$, the `imp_designed` VD includes $\{z_{11}, z_{12}\} \subset \mathcal{Z}^{E}_i$ AND $\{z_{13}, z_{31}\} \subset \mathcal{Z}^{NE}_i$, $\{z_{11}, z_{32(33)}\} \subset \mathcal{Z}^{NE}_i$, or $\{z_{12}, z_{32(33)}\} \subset \mathcal{Z}^{NE}_i$, and the `und_designed` VD includes $\{z_{11}\} \subset \mathcal{Z}^{NE}_i$ AND $\{z_{32}, z_{33}\} \subset \mathcal{Z}^{E}_i$, or $\{z_{12}\} \subset \mathcal{Z}^{NE}_i$ AND $\{z_{32}, z_{33}\} \subset \mathcal{Z}^{E}_i$. The `well_designed` state of VD_i represents the condition that the current target RT miss ratio is not greater than its higher bound of $\mathcal{MR_H}$ for each $G(i,j)$ with $IOPS^{cur}_{(i,j)} \leq IOPS(i,j)^{targ}$. The `imp_designed` state represents one of the following three conditions. The first condition is that a GPAM has a higher target RT miss ratio than $\mathcal{MR_H}$ with a higher average IOPS than its target IOPS, while another GPAM has a lower target RT miss ratio than $\mathcal{MR_L}$ with a lower average IOPS. It occurs if the initial I/O workload pattern becomes unbalanced, where I/O requests from the virtual disk are no longer distributed over its storage servers according to its target IOPS. The second and third conditions correspond to the situation where a GPAM has a higher target RT miss ratio than $\mathcal{MR_H}$ even with a lower IOPS, while another GPAM has a lower target RT miss ratio than $\mathcal{MR_L}$ with a target or even higher IOPS. These cases occur when the initial performance of the storage servers becomes unbalanced. The `und_designed` state corresponds to the condition where no GPAM exists with a lower target RT miss ratio with a target or higher IOPS, whereas a GPAM has a higher target RT miss ratio with a lower IOPS. Thus, the data migration for this state occurs only when the virtual disk is allowed to use extra storage resources for future extension.

The Data Migration Planner: Previous research mainly stressed the problem of scheduling each migration from its original location to its new one to minimize the total migration time [7]. Little research exist to create an efficient data migration plan, for example, to maximize the number of clients that can be served by the parallel disks or to automatically improve storage I/O performance [8]. However, no such previous research has directly addressed and handled the QoS dissatisfaction problem under a storage cluster.

Let us start by defining a few notations. Denote with $GPAM_{(i,j)}$ the j-th GPAM for VD_i, where $1 \leq j \leq N$. Denote with $BPAM_{(i,j,k)}$ the k-th BPAM for $GPAM_{i,j}$. Denote with $IOPS^{cur}_{(i,j,k)}$ the current IOPS for $BPAM_{(i,j,k)}$. Denote with $IOPS^{targ}_{(i,j)}$ and $IOPS^{cur}_{(i,j)}$ respectively the target IOPS for $GPAM_{(i,j)}$ and the current IOPS for $GPAM_{(i,j)}$, *i.e.*, the weighted-average IOPS of all $IOPS^{cur}_{(i,j,k)}$. Assume that VD_i distributes its data across N storage servers. Denote with $VD_{(i,j)}$ the j-th sub-virtual disk of VD_i. Planning an optimal data migration is infeasible in practice, because future I/O access patterns are not foreseeable. Thus, it leads us to devise a heuristic algorithm based on past cluster-wide QoS monitoring information. Our proposed data migration planner operates based on the two key ideas, the "minimal IOPS balancing(MIB)" and the "actual current IOPS normalized to the actual storage performance(ACI)." Figure 1(b) shows an example for the MIB, where VD_1 with Q_1 = (200IOPS, 10msec) is initially mapped onto the two homogeneous storage servers, $VD_{(1,1)}$ with $Q_{(1,1)}$ = (100IOPS, 10msec) and $VD_{(1,2)}$ with $Q_{(1,2)}$ = (100IOPS, 10msec). The QoS monitor detects $GPAM_{(1,1)}$ = (30IOPS, 3msec) with no target RT miss ratio and $GPAM_{(1,2)}$ = (150IOPS, 15msec) with 0.5 target RT miss ratio, where $\mathcal{MR}_H = 0.3$. Next, it determines that the $GPAM_{(1,1)}$ state is in z_{31} and the $GPAM_{(1,2)}$ state is in z_{13} and concludes that VD_1 is in the `imp_designed` state. Finally, according to the MIB, data blocks (BPAMs) equivalent to 50 IOPS of $VD_{(1,2)}$ migrates to $VD_{(1,1)}$, instead of 70 IOPS. Notice that the MIB minimally balances the IOPS across the storage servers to resolve the current QoS dissatisfaction. Figure 1(c) shows an example of the ACI, where VD_2 is configured exactly the same as VD_1. The QoS monitor detects that $GPAM_{(2,1)}$ = (100IOPS, 5msec) with zero target RT miss ratio and $GPAM_{(2,2)}$ = (100IOPS, 15msec) with 0.3 target RT miss ratio. The QoS monitor determines that the VD_2 is in the `imp_designed` state due to changes in the underlying storage performance. The actual target IOPS denoted by $\hat{IOPS}^{targ}_{(i,j)}$ for $VD_{(i,j)}$ can be computed



(a) GPAM-stat chart (b) MIB (c) ACI

Fig. 1. The GPAM-state chart, and the examples of the minimal IOPS balancing(MIB) and the actual current IOPS normalized to the actual storage performance(ACI)

from its target IOPS as follows:

$$\hat{IOPS}^{targ}_{(i,j)} = IOPS^{targ}_{(i,j)}(RT^{targ}_{(i,j)}/RT^{cur}_{(i,j)}). \quad (1)$$

Then, the actual target IOPS and RT for the storage servers become (200IOPS, 10msec) and (66IOPS, 10msec), respectively. Based on these, $VD_{(2,2)}$ migrates 34 IOPS to $VD_{(2,1)}$. For a given P IOPS to migrate from $VD_{(i,j)}$ to $VD_{(i,k)}$, we choose the first M BPAMs with the highest average IOPS, such that the sum of their average IOPS is equal to P IOPS. This design approach works well when the I/O workload pattern has a high spatial locality. Otherwise, more than one data migration is likely to occur by detecting an unbalanced condition repeatedly.

The **imp_designed** state can be caused by the changes in either the initial I/O workload pattern or the initial storage performance. The **imp_designed** state with the changed initial I/O workload pattern corresponds to one of two cases for the improperly-designed virtual disk VD_i, where $\{z_{11}, z_{12}\} \subset \mathcal{Z}^E_i$ and $\{z_{13}, z_{31(32)}\} \subset \mathcal{Z}^{NE}_i$. For this, the minimal amount of IOPS migrates from z_{13} to $z_{31(32)}$, resultingly the actual current IOPS of GPAM in z_{13} does not exceed its target IOPS. The **imp_designed** state with the changed initial storage performance corresponds to the following distribution of the GPAM states, where $\{z_{11}, z_{32(33)}\} \subset \mathcal{Z}^{NE}_i$ or $\{z_{12}, z_{32(33)}\} \subset \mathcal{Z}^{NE}_i$. We can compute an actual target IOPS based on the the observed RT and IOPS from Equation (1) that can meet the given target RT for VD_i. Next, we transform the observed IOPS into its actual current IOPS denoted by $\hat{IOPS}^{cur}_{(i,j)}$ on the basis of the actual target IOPS as follows:

$$\hat{IOPS}^{cur}_{(i,j)} = IOPS^{targ}_{(i,j)} + (IOPS^{cur}_{(i,j)} - \hat{IOPS}^{targ}_{(i,j)}). \quad (2)$$

The migration planner first computes an actual target IOPS for each storage server, according to Equation (1). The migration planner calculates the actual current IOPS at each GPAM, according to Equation (2). To sum, a minimal amount of IOPS needs to be moved from z_{11} (or z_{12}) to z_{32} and z_{33}, so that the actual current IOPS of the GPAM in z_{11} (or z_{12}) does not exceed its target IOPS.

The under-designed virtual disk is mainly attributed to the lack of storage resources to meet the given QoS requirement for the virtual disk. This virtual disk state has a distribution of GPAM states that is similar to that of the second case of improperly-designed virtual disks; that is, $\{z_{11}\} \subset \mathcal{Z}^{NE}_i$ and $\{z_{32}, z_{33}\} \subset \mathcal{Z}^E_i$, or $\{z_{12}\} \subset \mathcal{Z}^{NE}_i$

and $\{z_{32}, z_{33}\} \subset \mathcal{Z}_i^E$. While z_{11} or z_{12} has a GPAM, no GPAMs exist that will process a part of I/O requests for GPAMs in z_{11} or z_{12}. As a result, we need to migrate the IOPS in z_{11} or z_{12} to a new storage server that can be additionally used by the virtual disk. Unless the extra storage server is available, no migration plan will be made. Instead, static virtual disk reconfiguration will deal with this problem. Hereafter, we assume that at least a single new storage server is available to each virtual disk. Given multiple extra storage servers, we need to decide which storage server will be used for the virtual disk. The proposed scheme selects a storage server, where the ratio of the current IOPS to the target IOPS is the lowest among others. More detailed descriptions for each algorithm can be found in [4].

Data Migration Agents and Operational Parameters: A data migration plan is sent to the associated QoS servers to initiate actual data migration among the storage servers. Then, a data migration agent at each storage sever is in charge of executing the data migration plan. The proposed scheme can be configured by a set of policy-based operational parameters that include a QoS monitoring interval (T_I), QoS satisfaction level for each sub-virtual disk, $VD_{(i,j)}$ ($\mathcal{MR}_H$), sensitivity for determining a virtual disk state (T_m, U_m), BPAM size ($|BPAM|$), and marginal storage capacity ratio(MSR). Given (T_m, U_m), for example, a virtual disk can be determined as an improperly-designed VD state only if the QoS monitor detects the improperly-designed VD state U_m times over past I_m monitoring intervals, *i.e.,* an observation time window of $T_m = I_m T_I$ seconds. Configuring $MSR = 100\%$ implies that each sub-virtual disk reserves 100% of its storage capacity for data migration.

3 Performance Evaluations

Performance evaluations have been conducted on a storage simulator that consists of an I/O workload generator, a set of virtual disks (storage clients), a set of storage servers. The operational parameters are configured as T_I = 5sec, $\mathcal{MR}_H = 0.3$, $\mathcal{MR}_L = 0.1$, $\alpha_c = 0.3$, $U_m = 1$, $T_m = 5$, $|BPAM|$ = 2048blocks, and $MSR = 100\%$. The two performance metrics include the average response time and the target RT miss ratio for the I/O workload from each virtual disk. In our simulation, two virtual disks of VD_1 and VD_2 are mapped onto the storage cluster of SS_1 and SS_2, implying that each storage server is shared by the two virtual disks. The QoS the requirements of the virtual disks are Q_1=(4KB, 90IOPS, 70msec) and Q_2=(4KB, 90IOPS, 100msec). Thus, the QoS requirements of $VD_{(i,j)}$ for VD_1 and VD_2 are are defined as follows: $Q_{(1,1)}$= (4KB, 45 IOPS, 70msec), $Q_{(1,2)}$= (4KB, 45 IOPS, 70msec), $Q_{(2,1)}$= (4KB, 45 IOPS, 100msec), and $Q_{(2,2)}$= (4KB, 45 IOPS, 100msec).

Our simulation employs four different types of QoS dissatisfaction that include WS_1^{imp1}, WS_2^{imp1}, WS_1^{imp2}, and WS_1^{und}. The types of WS_1^{imp1} and WS_2^{imp1} represent that the most of I/O requests from VD_2 are issued to the SS_2; that is, WS_1^{imp1} and WS_2^{imp1} respectively send 100% and 90% of all the I/O requests of the VD_2 to SS_2. In the case of WS_1^{imp2}, the performance of SS_2 decreases, because seek times and rotational delays for processing I/O requests from VD_1 and VD_2 become higher. Our simulator emulates storage performance degradation by adjusting the equations and parameters to compute a seek time and a rotational delay as follows: the long and short seek times of VD_2 are respectively changed to $9.0 + 0.008d$ and $4.24 + 0.4sqrt(d)$ from the initial equations [6] of $8.0 + 0.008d$ and $3.24 + 0.4sqrt(d)$, and the average rotational delay is from 2.99msec to 4.28msec. In the case of WS_1^{und}, it is assumed that the VD_1 is initially mapped onto SS_1 and SS_2, and the VD_2 is initially mapped onto SS_2 and SS_3. Then, the response times of I/O requests at SS_3 will obviously become higher than its target response time with a high target RT miss ratio. By contrast, the target RT miss ratio at SS_2 remains slightly high, because the heavier I/O workload is given to SS_2 from VD_1. As a result, it needs to migrate an amount of data blocks to a new storage server that is allowed for extra use. In our experiment, the data migration planner will send data blocks from SS_3 to SS_1. Assuming that the extra storage resource for VD_2 is equivalent to the storage resource allocated to $VD_{(2,1)}$, the newly allocated storage server is configured with the same QoS requirement as in the other storage servers for the virtual disk. Table 1 summarizes the results of the experiments for the four types of QoS dissatisfaction. We add extra performance metrics of the number of migrated BPAMs and T_{settle}, where T_{settle} represents the elapsed time to complete data migration. In WS_1^{impl}, the proposed scheme(prop) can

Table 1. Result of RT variations of VD_1 and VD_2 with the four different types of QoS dissatisfaction: WS_1^{impl}, WS_2^{impl}, WS_1^{imp2}, and WS_1^{und}

		Avg. IOPS (IOPS)		Avg. resp. time (msec)		Target resp. time miss ratio		# of mig. BPAMs	T_{settle} (sec)
		no-mig	prop	no-mig	prop	no-mig	prop		
WS_1^{impl}	VD_1	85.9	88.7	53.7	23.8	0.25	0.01	14	74.2
	VD_2	74.3	87.6	227	31.3	0.94	0.03	28	311.8
WS_2^{impl}	VD_1	87.0	88.2	43.5	23.1	0.16	0.00	6	126.4
	VD_2	80.4	87.5	128	37.7	0.59	0.02	14	72.4
WS_1^{imp2}	VD_1	85.9	87.6	56.5	38.0	0.25	0.06	12	636.5
	VD_2	85.7	86.9	57.0	37.5	0.11	0.00	13	26.3
WS_1^{und}	VD_1	122.8	121.9	25.6	31.2	0.01	0.03	n/a	n/a
	VD_2	81.6	86.9	104.7	40.4	0.43	0.02	37	631.7

guarantee the given QoS requirements with almost 100% for VD_1 and VD_2 even in the presence of the unbalanced I/O workload pattern by migrating 14 BPAMs of VD_1 and 28 BPAMs of VD_2 from SS_2 to SS_1. While the settling time (T_{settle}) of VD_2 is observed to be 311.8 seconds, the QoS dissatisfaction problem is actually resolved in about 34.2 seconds. By contrast, VD_2 violates its target response time mostly without data migration(no-mig), and VD_1 has also a high target RT miss ratio. We have a similar result in WS_2^{impl}. However, notice that the number of the migrated BPAMs is slightly smaller, compared with WS_1^{impl}. In WS_1^{imp2}, 12 BPAMs of VD_1 and 13 BPAMs of VD_2 migrate from SS_2 to SS_1 by the migration planner and agents. In WS_1^{und}, the target RT miss ratio of VD_2 decreases to 0.02 from 0.43 by migrating 37 BPAMs for VD_2 among different storage servers.

4 Concluding Remarks

This paper addressed the types of QoS dissatisfaction caused by the imbalance of the initial I/O workload pattern and storage performance under a storage cluster environment and the proposed a systematic scheme to resolve the problem. The proposed scheme introduced a base unit of storage called BPAM for efficient performance monitoring and data migration processes. The proposed scheme detects any problem of QoS dissatisfaction for each virtual disk and then identifies the cause of the problem in a systematic manner. Subsequently, it resolves the problem by minimally balancing actual current IOPS normalized to the actual storage performance across multiple storage servers within a virtual disk. For this, the proposed scheme provides a cluster-wide QoS monitoring scheme for each virtual disk, a data migration planner to change improperly-designed and under-designed virtual disks into well-designed ones, and data migration agents at storage servers to perform actual data migration between storage servers. The simulation results conducted in our storage cluster simulator revealed that the proposed data migration scheme can effectively handle any QoS dissatisfaction in the presence of various changes in the initial I/O workload pattern and storage performance. In future, we need to devise a more intelligent data migration planner that concurrently takes into account the status of data migration in the other virtual disks.

Acknowledgments

This research was supported by the Daegu University Research Grant, No 20050346. The authors would like to thank the Ministry of Education of Korea for its support towards the Elec. and Computer Eng. Division at POSTECH through the BK21 program. This research has also been supported in part by HY-SDR IT Research Center, in part by the grant number R01-2003-000-10739-0 from the basic research program of the Korea Science and Engineering Foundation, in part by the regional technology innovation program of the Korea Institute of Industrial Technology Evaluation and Planning, and in part by the next generation PC program of the Korea ETRI.

References

1. G. Alvarez, *et al.*, "Minerva: An automated resource provisioning tool for large-scale storage systems," *ACM Transactions on Computer Systems*, vol. 19, pp. 483–518, November 2001.
2. L. Huang, "Stonehenge: A high performance virtualized network storage cluster with QoS guarantees," Tech. Rep., SUNY at Stony Brook, January 2002.
3. E. Anderson, M. Hobbs, K. Keeton, S. Spence, M. Uysal, and A. Veitch, "Hippodrome: Running rings around storage administration," in *Proceedings of Conference on File and Storage Technologies*, January 2002.
4. Y. Nam, *Dynamic Storage QoS Control for Storage Cluster and RAID Performance Enhancement Techniques.* Ph.D Dissertation, POSTECH, February 2004.
5. C. Ruemmler and J. Wilkes, "Unix disk access patters," in *Proceedings of Winter USENIX*, pp. 405–420, January 1993.
6. C. Ruemmler and J. Wilkes, "An introduction to disk drive modeling," *IEEE Computer*, vol. 27, pp. 17–29, March 1994.
7. Y. Kim, "Data migration to minimize the average completion time," in *Proceedings of the 14th Annual ACM-SIAM Symposium on Discrete Algorithms*, 2003.
8. F. Hidrobo and T. Cortes, "Automatic storage system based on automatic learning," in *Proceedings of the International Conference on High-Performance Computing*, 2004.