Journal of Electronics and Informatics (2019)
Vol.01/ No. 01
Pages: 1-11
http://www.irojournals.com/iroei/
DOI: https://doi.org/10.36548/jei.2019.1.001

EFFICIENT MANAGEMENT OF PROBE-BASED NANO STORAGE DEVICES

Dr. Hyokyung Bahn,
Department of Computer Engineering,
Ewha University, Seoul, Korea,
Email id: hyokyung.bahn@gmail.com

Abstract - Probe-based nano storage is a new storage medium that has several desirable features such as nano-scale density, concurrent accessing, and low-power consumption. Thus, it is expected to be adopted in a wide range of domains such as medical devices and cloud servers. However, as the internal architecture of probe-based nano storage is different from hard disk drives, new management policies are necessary. In particular, probe-based nano storage has thousands of concurrent working heads, which should be managed efficiently for maximizing performances. In this article, we analyze three important considerations in managing probe-based nano storage devices and propose scheduling policies by taking into account them. The proposed policies aim to maximize the throughput of storage but also consider the variation of waiting time. Performance evaluation results show that the proposed policies perform better than existing policies with respect to the average waiting time and the variation of the waiting time.

Keywords: Probe-based Nano Storage, Throughput, Scheduling, Management, Storage.

1. INTRODUCTION

Probe-based nano storage is foreseen as a promising storage medium that may be adopted for a wide range of future storage systems ranging from medical storage for embedded devices to high capacity cloud servers [1, 2]. Like hard disk drives, probe-based nano storage also needs management policies such as request scheduling. In this article, we analyze three important considerations in managing probe-based nano storage devices and propose scheduling policies by considering the analysis results. The proposed policies aim to maximize the throughput of storage but also consider the fairness issues to avoid starvation problems.

Because the volume of data to be managed becomes excessively large in the age of the fourth industrial revolution, storage capacity requirements are growing rapidly and emerging technologies and new applications that aim to realize this storage requirement also appears continuously. At the cloud part of the storage need, the size of hard disk drives grows more than 50% each year and the price of the products is likely to be going down rapidly. In another

ISSN: 2582-3825 (online)

Journal of Electronics and Informatics (2019)
Vol.01/ No. 01
Pages: 1-11
http://www.irojournals.com/iroei/
DOI: https://doi.org/10.36548/jei.2019.1.001

side of the storage need, as handheld smart devices become an inevitable part in our daily lives, the need for large storage in small consumer electronics is also increasing. NAND flash storage and solid state drives are introduced to satisfy such requirements. Probe-based nano storage is another candidate to answer demands on both a cloud server and a small handheld device.

There are several good features of the probe-based nano storage devices. Firstly, probe-based nano storage can be manufactured as a very small scale but its capacity is large enough. As depicted in Fig. 1, probe-based nano storage can store over 5 gigabytes although the range is as small as 1cm2. With a similar size of a hard disk drive, the capacity of probe-based nano storage is much larger. Due to this reason, probe-based nano storage can be adopted in a wide spectrum of domains including mobile handheld devices as well as datacenters.

Secondly, probe-based nano storage exhibits small access latency and its variance is also small. Unlike NAND flash storage, the access latency of probe-based nano storage is several hundred microseconds for both read and write operations. Note that the access latency of NAND flash storage is varied significantly according to operations to be performed. Specifically, read operations in NAND flash are fast, whereas write operations are 8 to 10 times slower than read operations. Hard disk drives have the access latency of tens of milliseconds because of its physical characteristics.

Thirdly, probe-based nano storage has some desirable features like energy efficiency, shock resistance, high bandwidth, and low price in comparison with hard disk drive or NAND flash storage. This allows probe-based nano storage an appropriate storage for cloud servers as well as small embedded systems.

In spite of such benefits, managing probe-based nano storage is difficult because it has some unique physical characteristics when compared with hard disk drives or NAND flash storage1-4. That is, probe-based nano storage has thousands of heads that can work concurrently. Moreover, the magnetic media of the probe-based nano storage is a square structure and does not rotate like hard disk drives. Consequently, management policies to control probe-based nano storage need to be studied [2, 3].

Like hard disk drives, scheduling policies are important to control the probe-based nano storage. Unfortunately, as the physical characteristics of the internal architecture are not identical to hard disk drives, a new scheduling policy is necessary for the probe-based nano storage devices. In particular, thousands of concurrent working heads make the scheduling policy an even more complicated problem if we consider the performance that can be obtained from the concurrency.

This article analyzes three characteristics that need to be considered in the design of scheduling policies for probe-based nano storage devices; concurrency, seek latency, and fairness. With respect to these three characteristics, we simulate several types of scheduling policies for probe-based nano storage and depict the efficiency of the policies.

ISSN: 2582-3825 (online)

Journal of Electronics and Informatics (2019)
Vol.01/ No. 01
Pages: 1-11
http://www.irojournals.com/iroei/
DOI: https://doi.org/10.36548/jei.2019.1.001

Our experimental studies depict that concurrency-based fairness-aware scheduling policies outperform the legacy SPTF (Shortest Positioning Time First) or SSTF (Shortest Seek Time First) policies in terms of the average waiting time when the workload traffic is excessive.

The rest of the article is organized as follows. Section 2 describes the internal architecture of probe-based nano storage devices. Section 3 describes the concurrency-based scheduling policies. Then, Section 4 depicts the performance evaluation results. Section 5 briefly summarizes the related work. Finally, in Section 6, we conclude this article.

2. ARCHITECTURE OF PROBE-BASED NANO STORAGE

A probe-based nano storage device is composed of the magnetic media that has groups of regions and their corresponding heads to access data. To access data on a given (x, y) position, probe-based nano storage needs seek latency for moving to an appropriate location like hard disk drives. However, the heads of probe-based nano storage do not move and magnetic media itself goes to the position of the corresponding heads in order to access data. The movement of the media in the directions of x and y axes is independent and proceeds concurrently. Hence, the seek latency Lat(x, y) for a specific (x, y) position can be estimated as follows.

$$Lat(x, y) = \max (Seek_x, Seek_y) \quad (1)$$

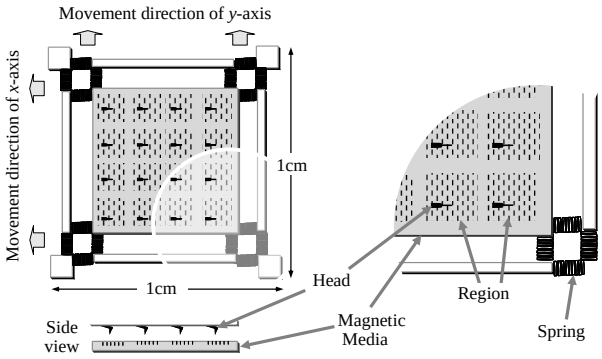

Fig. 1. Internal architecture of a probe-based nano storage device. A large number of regions exist on the magnetic media and each region has the corresponding head for read/write operations. The magnetic media moves along the x-axis and the y-axis independently.

ISSN: 2582-3825 (online)

Journal of Electronics and Informatics (2019)
Vol.01/ No. 01
Pages: 1-11
http://www.irojournals.com/iroei/
DOI: https://doi.org/10.36548/jei.2019.1.001

where Seekx and Seeky are the seek latency with respect to the x and y axes, respectively. Note that Seekx is larger than Seeky in most cases as some settlement latency needs to be included in Seekx. Settlement latency here is the latency necessary for the oscillations of the magnetic media to damp out. This latency is dependent on the construction of the magnetic media and the stiffness of the spring that sustains the magnetic media [4]. When data is accessed after seeking, the magnetic media moves along the direction of the y axis, and thus it only needs the constant speed in the direction of the y axis. Thus, oscillation in the direction of the x axis is significant due to the possibility of off-track interference, whereas the oscillation with respect to the direction of the y axis is only related to the transfer rate of accessing data [6].

3. SCHEDULING POLICIES FOR PROBE-BASED NANO STORAGE

Scheduling policies of probe-based nano storage need to take into account 3 important issues. The first is that the scheduling policy should take into account the concurrent accesses of heads. As thousands of heads can access the media at the same time, providing preferences to those positions that have multiple pending requests will be more efficient. Secondly, the scheduling policy needs to take into account the seek latency of moving the media. In particular, reducing the seek latency by decreasing the moving distance is necessary in order to service the requests in the scattered (x, y) locations effectively. Third, the scheduling policy needs to consider the variation of each request's waiting time. Although improving the average waiting time should be the primary goal, fairness is another important issue as waiting time of each request needs to be bounded. The former two issues are related to the scheduling efficiency, whereas the last issue is related to fairness. We will discuss the efficiency issue first in Section 3.1, and then the fairness of the scheduling in Section 3.2. Finally, the two issues will be merged in Section 3.3.

3.1. Maximizing Concurrency

An efficient scheduling policy in probe-based nano storage should take into account the concurrency of the heads. That is, requests on the same x and y positions in different regions can be serviced together. By considering this, a good scheduling policy can reduce the average of the seek latency by giving preferences to the location (x, y) that has more requests.

Suppose that Num(x, y) denotes the number of pending requests on the (x, y) position. Then, a concurrency-based scheduling policy can be incorporated into latency-based scheduling policies like SSTF (Shortest Seek Time First) or SPTF (Shortest Positioning Time First). Note that SSTF priorities pending requests based on their seek latency with respect to the x-axis only whereas SPTF considers the seek latency with respect to both the x-axis and the y-

ISSN: 2582-3825 (online)

Journal of Electronics and Informatics (2019)
Vol.01/ No. 01
Pages: 1-11
http://www.irojournals.com/iroei/
DOI: https://doi.org/10.36548/jei.2019.1.001

axis. Actually, in hard disk storage, the x-axis and the y-axis of probe-based nano storage correspond to the seek latency and the rotational latency, respectively [5, 10]. From now on, we will consider SPTF as the latency-based scheduling policy. Based on the aforementioned two characteristics, the P-SPTF policy is presented [9, 15]. Preferences in P-SPTF is decided as follows. Suppose that Preference(x, y) is the preference value to decide the next request to be served. Then, Preference(x, y) is denoted as

$$\text{Preference}(x, y) = \text{Num}(x, y) / \text{Lat}(x, y) \quad (2)$$

where Num(x, y) is the number of pending requests on the (x, y) location in all regions and Lat(x, y) is the seek latency from the current location to (x, y) position as is decided by SPTF. The request with the largest Preference(x, y) value is chosen as the next position to be serviced.

3.2. Minimizing Variation of Latency

Reducing the variation of latency is also important in the design of scheduling policies. Scheduling policies that only take into account the maximum concurrency or minimum seek latency is not appropriate with respect to minimizing variation of latency since it may do harm to fairness, causing excessive delay of some pending requests. In case of SPTF or P-SPTF, this can occur if a pending request has long distance from the current head location or few requests exist on the requested location. Although the request can be serviced as time progresses, the waiting time of that request may be much longer than the average cases. That is, the variation of the waiting time will be large.

To relieve this issue, we adopt an aging concept. We call a scheduling policy that consider the aging concept as well as the concurrency and distance issues CF-SPTF (Concurrency and Fairness aware SPTF). CF-SPTF considers the total waiting time of the requests on the same (x, y) position. CF-SPTF decides the next request to be serviced by considering the preference value like C-SPTF. The preference value Preference(x, y) of each position (x, y) is calculated by

$$\text{Preference}(x, y) = \square \text{ Wait}(x, y) / \text{Lat}(x, y) \quad (3)$$

where Lat(x, y) is the seek latency between the current head position and the requested (x, y) location and □ Wait(x, y) is the accumulation of the waiting time for requests on the same (x, y) location across all regions. □ Wait(x, y) considers the concept of the concurrency in C-SPTF as well as the aging concept.

ISSN: 2582-3825 (online)

Journal of Electronics and Informatics (2019)
Vol.01/ No. 01
Pages: 1-11
http://www.irojournals.com/iroei/
DOI: https://doi.org/10.36548/jei.2019.1.001

In reality, we can incorporate the aging concept into the scheduling policy in various ways to consider fairness. The basic philosophy is to visit all (x, y) locations in the region while traversing the media like the SCAN scheduling policy in hard disk drives.

The Z-SPTF (Zone-based SPTF) scheduling policy can be classified into this category [2, 8]. Z-SPTF partitions each region into some square zones, and determines the requests to be serviced within a zone by the SPTF policy whereas the C-SCAN (Circular SCAN) policy is utilized while deciding the next zone to be serviced. Z-SPTF solves the fairness issue of SPTF as it services requests far away from the head location based on the zone concept. Nevertheless, Z-SPTF does not consider the concurrency of the probe-based nano storage at all.

3.3. Considering Altogether

We have considered the three issues in the scheduling of probe-based nano storage. The first two issues are seek latency and concurrency, mostly related to efficiency and the third issue is related to fairness by reducing the variation of the waiting time.

Note that traditional scheduling policies used in hard disk drives such as SSTF and SPTF consider only the first issue, that is the seek latency. C-SPTF incorporates the concurrency issue into the seek latency whereas Z-SPTF incorporates the fairness issue into the seek latency. CF-SPTF meets all three requirements, seek latency, concurrency, and fairness.

Now, we will introduce another scheduling policies that consider seek latency, concurrency, and fairness. They are ZSCAN-C-SPTF and ZC-SPTF. Unlike CF-SPTF that considers the three requirements within a unified preference value, ZSCAN-C-SPTF and ZC-SPTF consider the requirements through hierarchy basis.

Specifically, ZSCAN-C-SPTF partitions a region by zones like Z-SPTF. A zone is firstly chosen by this policy and all requests within the zone is serviced. Then, another zone is selected. Zone selection is performed by the SCAN policy and request selection within a zone is performed by the C-SPTF policy. In case of the ZC-SPTF policy, C-SPTF is utilized for zone selection as well as the request selection within a zone.

4. PERFORMANCE EVALUATION

To evaluate the efficiency of various scheduling policies, simulation experiments have been conducted. A logical block size is set to 512 bytes and a sector size within a probe-based nano storage is set to 8 bytes similar to previous studies. Thus, a logical block is mapped to 64 sectors. Note that these 64 sectors within a logical block is mapped to

ISSN: 2582-3825 (online)

Journal of Electronics and Informatics (2019)
Vol.01/ No. 01
Pages: 1-11
http://www.irojournals.com/iroei/
DOI: https://doi.org/10.36548/jei.2019.1.001

different regions in order to allow concurrent accesses. Logical blocks of adjacent positions are mapped sequential to the y-axis direction in order to generate sequential accesses without repositioning.

Similar to previous studies, the inter-arrival time of each request conforms to an exponential distribution. The request size also conforms to the exponential distribution with the average value of 4 kilobytes, and the allocation of requests is performed uniformly to all positions within the device.

To see the effect of workload traffic changes, we vary the inter-arrival time from the original scale to the scaling factor of 30. Note that a scaling factor of 10 makes a workload ten times more intense than the original situation.

When the total data size is larger than the capacity of probe-based nano storage, multiple devices are used together, which also allows the simultaneous access of data on different devices.

Six scheduling policies, C-SPTF, CF-SPTF, SPTF, SSTF, ZC-SPTF and ZSCAN-C-SPTF, are simulated and compared. For zone based policies, we set the number of zones in each region to 75.

Fig. 2 depicts the average waiting time of six policies as the scaling factor is varied. Note that the results for SPTF, SSTF, and C-SPTF are similar to previous studies [9, 15].

As we see, concurrency based scheduling policies, C-SPTF, CF-SPTF, ZC-SPTF, and ZSCAN-C-SPTF exhibit better results than SPTF and SSTF excessively as the scaling factor increases. The reason is that concurrent based scheduling policies take into account the number of requests on the same (x, y) location and provide preferences in scheduling. Quick serving of locations with many pending requests leads to the improved waiting time. Through these experiments, we can observe that concurrency based scheduling policies are more scalable than other policies and are efficient specially when the workload traffic is very heavy. Another finding from this experiment is that the performances of concurrency based policies, C-SPTF, CF-SPTF, ZC-SPTF, and ZSCAN-C-SPTF, are very similar.

Fig. 3 depicts the variation of waiting time (σ^2/μ^2) as the scaling factor is varied. Note that □ is the standard deviation of waiting time and μ^2 is the average waiting time. A lower value of (σ^2/μ^2) implies that the waiting time of each request is not far from the average waiting time. This is an index used to evaluate the fairness [5]. By accumulating the waiting time of all requests on the same (x, y) position, CF-SPTF exhibits competitive results in comparison with other policies for a wide range of experimental conditions.

ISSN: 2582-3825 (online)

Journal of Electronics and Informatics (2019)
Vol.01/ No. 01
Pages: 1-11
http://www.irojournals.com/iroei/
DOI: https://doi.org/10.36548/jei.2019.1.001

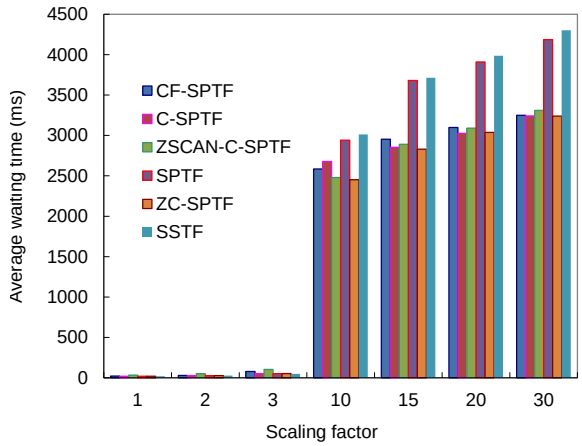

Fig. 2. Average waiting time of the six policies as the scaling factor is varied.

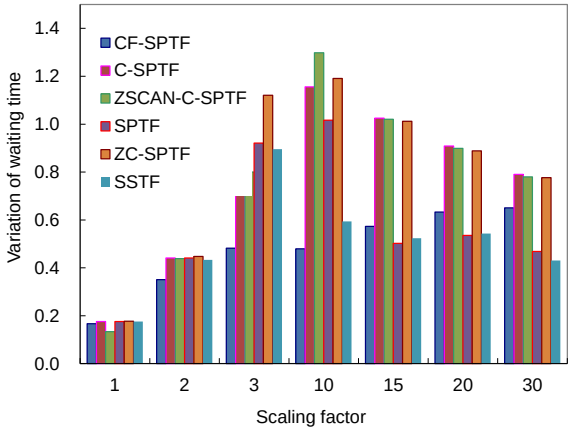

Fig. 3. Variation of waiting time as the scaling factor is varied.

Concurrency based policies with the zone concept like ZSCAN-C-SPTF and ZC-SPTF do not exhibit good results with respect to the variation of waiting time. As ZC-SPTF performs the scheduling of a zone based on the C-SPTF policy, the variation of waiting time becomes large. However, as ZSCAN-C-SPTF performs the scheduling of zones by using the SCAN policy, we did not expect the large variation of waiting time. We expect that partitioning of a region by a smaller zone size can improve the variation of waiting time in case of ZSCAN-C-SPTF. Nevertheless, a small zone may degrade the average waiting time as it weakens the concurrency effect.

ISSN: 2582-3825 (online)

Journal of Electronics and Informatics (2019)
Vol.01/ No. 01
Pages: 1-11
http://www.irojournals.com/iroei/
DOI: https://doi.org/10.36548/jei.2019.1.001

5. RELATED WORKS

Request scheduling is an important topic in storage systems [11-14]. There are several traditional scheduling policies such as the first-come first-served (FCFS) policy, the shortest seek time first (SSTF) policy, and the shortest positioning time first (SPTF) policy. As such policies were originally proposed for hard disk drives, studies that analyze the effectiveness of these policies on probe-based nano storage have been performed by allocating seek latency to the moving latency on the x-axis and rotational latency to the moving latency on the y-axis [7].

Internal structures of the probe-based nano storage have been considered in some scheduling policies [2, 3]. The minimum spanning tree (MST) based scheduling policy has been proposed for probe-based nano storage [3]. Since the shortest positioning time first policy does not exhibit good performances with respect to the average waiting time because it is a greedy algorithm, the MST policy makes a minimum spanning tree for the requests in the queue by considering the seek latency and decides the schedules through visiting the spanning tree. It was shown that the MST policy performs better than SPTF with respect to the average waiting time. The problem of the MST policy is that it requires overhead while generating and reconstructing the minimum spanning tree if a new request comes. Moreover, the MST policy has a fairness problem like SPTF, leading to excessive delay for some requests.

A zone-based shortest positioning time first (Z-SPTF) scheduling policy has been presented by taking into account the fairness problem of SPTF [2, 8]. Z-SPTF partitions the regions of the magnetic media to a group of zones by considering the seek latency equivalence. The C-SCAN (Circular Scan) policy is adopted for selecting zones for scheduling, whereas the Z-SPTF policy is adopted to select requests for scheduling within a zone. The performance of Z-SPTF is rather worse than SPTF with respect to the average waiting time. However, Z-SPTF exhibits better performances than SPTF with respect to the variance of waiting time, implying that the starvation problem has been reduced and the fairness is improved.

Concurrency based request scheduling policies for probe-based nano storage devices have also been presented [9, 15]. C-SPTF allocates a preference value to each (x, y) location and the location with the largest preference value is selected for scheduling. The rationale of C-SPTF is to allocate large preference value to locations that have a lot of requests. Consequently, a large number of pending requests can be serviced rapidly and simultaneously, improving the average waiting time significantly. CF-SPTF has also been presented to combine the aging concept into the C-SPTF policy, improving the fairness.

6. CONCLUSION

ISSN: 2582-3825 (online)

Journal of Electronics and Informatics (2019)
Vol.01/ No. 01
Pages: 1-11
http://www.irojournals.com/iroei/
DOI: https://doi.org/10.36548/jei.2019.1.001

Probe-based nano storage is expected to be commercialized as a storage medium for future storage systems such as medical storage for embedded devices. This article argued that traditional scheduling polices cannot perform well for probe-based nano storage and analyzed requirements of scheduling policies for probe-based nano storage. Specifically, we considered seek latency, concurrency, and fairness issues in designing an efficient scheduling policy. We presented and compared six scheduling policies with respect to the average waiting time and the variation of the waiting time. Trace-driven simulations for a wide range of configurations have shown that concurrency based scheduling policies exhibit better performances than traditional policies like SPTF or SSTF with respect to the average waiting time.

References

[1] W. Koelmans, J. Engelen, and L. Abelmann, “Probe-based data storage,” Technical Report, https://arxiv.org/abs/1511.08755 (2015).

[2] B. Hong, S. Brandt, D. Long, E. Miller, K. Glocer, and Z. Peterson, “Zone-based Shortest Positioning Time First Scheduling for probe-based Storage Devices,” 11th IEEE/ACM MASCOTS Conf. (2003).

[3] H. Yu, D. Agrawal, and A. Abbadi, “Towards optimal I/O scheduling for probe-based storage,” 20th IEEE MSST Conf. (2003).

[4] J. Griffin, S. Schlosser, G. Ganger, and D. Nagle, “Modeling and performance of probe-based storage devices,” ACM SIGMETRICS Conf., 56-65 (2000).

[5] B. Worthington, G. Ganger, and Y. Patt, “Scheduling Policies for Modern Disk Drives,” ACM SIGMETRICS Conf., 241-251 (1994).

[6] S. Schlosser and G. Ganger, “probe-based storage devices and standard disk interfaces: A square peg in a round hole?” 3rd USENIX FAST Conf. (2004).

[7] J. Griffin, S. Schlosser, G. Ganger, and D. Nagle, “Operating system management of probe-based storage devices,” 4th USENIX Symp. Operating Systems Design and Implementation, 227-242 (2000).

[8] B. Hong, S. Brandt, D. Long, E. Miller, K. Glocer, and Z. Peterson, “Using probe-based Storage in Computer Systems – Device Modeling and Management,” ACM Transaction on Storage, 2, 2, 139-160 (2006).

[9] S. Lee, H. Bahn, and S. H. Noh, “Parallelism-aware Request Scheduling for probe-based Storage Devices,” 14th IEEE MASCOTS Conf. (2006).

[10] P. Denning, “Effects of scheduling on file memory operations,” AFIPS Spring Computer Conf., pp.9-21 (1967).

[11] S. Schlosser, J. Griffin, D. Nagle, and G. Ganger, “Designing computer systems with probe-based storage,” 9th Int’l Conf. Architectural Support for Programming Languages and Operating Systems (2000).

ISSN: 2582-3825 (online)

Journal of Electronics and Informatics (2019)
Vol.01/ No. 01
Pages: 1-11
http://www.irojournals.com/iroei/
DOI: https://doi.org/10.36548/jei.2019.1.001

[12] P. Vettiger, M. Despont, U. Drechsler, U. Dürig, W. Häberle, M. Lutwyche, H. Rothuizen, R. Stutz, R. Widmer, and G. Binnig, "The Millipede – More than one thousand tips for future AFM data storage," IBM Journal Research and Development, Vol.44, No.3, pp.323-340 (2000).

[13] R. Rangaswami, Z. Dimitrijevic, E. Chang, and K. Schauser, "probe-based disk buffer for streaming media servers," Int'l Conf. Data Engineering (2003).

[14] H. Yu, D. Agrawal, and A. Abbadi, "Tabular placement of relational data on probe-based storage devices," Int'l Conf. Very Large Databases (2003).

[15] H. Bahn, S. Lee, and S. H. Noh, "P/PA-SPTF: Parallelism-aware Request Scheduling Policies for probe-based Storage Devices," ACM Transactions on Storage, Vol.5, No.1 (2009).

ISSN: 2582-3825 (online)