2009 Fifth International Joint Conference on INC, IMS and IDC

A Novel Cost-Effective Disk Scrubbing Scheme

Junping Liu†, Ke Zhou†, Liping Pang†, Zhikun Wang†,Yuhui Deng††, Dan Feng†
(† Wuhan National Laboratory for Optoelectronics
School of Compute Science and Technology
Huazhong University of Science and Technology, Wuhan, 430074, China;
†† EMC Research China, Beijing)
E-mail: liujunping@smail.hust.edu.cn; k.Zhou@mail.hust.edu.cn

Abstract—a **distinct benefit of disk scanning or scrubbing operation is identifying the potential failure sectors as early as possible, thus providing high reliability. Obviously, the higher the scrubbing frequency is, the higher the system reliability is. However, it may take a few hours for a scanning process to check the whole disk. In other words, the scrubbing process may result in a downtime or a lower system performance. Furthermore, the scrubbing process consumes energy. In order to reduce the impact of disk scrubbing on disk performance and energy consumption, system designers choose to scan the disk in a low frequency, which results in a lower reliability. Additionally, conventional disk scrubbing schemes assume that the disk failure rate is constant, while the recent researches[2][6] show the disk failure rate is more complex. In this paper, we present a novel scrubbing scheme to solve the above challenges. In the scheme, an optimum scrubbing cycle is decided by keeping a balance between data loss cost, scrubbing cost, and disk failure rate. Our research shows that the scrubbing scheme is applicable for storage with low-capacity disk and inexpensive data.**

Key words: Disk Scrubbing, Disk Scanning, Storage, RAID

I. INTRODUCTION

System reliability is a major challenge in system design. Avoiding system downtime and the cost of the actual downtime make up more than 40% of the total cost of ownership for modern IT systems. A large-scale system could involve hundreds of thousands of components, failures in the system are not exceptions any more[6].

Data is priceless for enterprises. Therefore, the storage systems which are employed to store the data must guarantee the system reliability. However, disk failures happen frequently in modern data centers in which there are thousands or even tens of thousands of disks. When reliability becomes one of the most key characteristics of storage systems, users cannot tolerate those failures. To improve storage reliability, RAID is used in many applications to store huge amounts of data and protect data from disk failures[1]. Whereas, recent research shows that RAID is not foolproof and storage subsystem failures are not independent. After one failure, the probability of additional failures (of the same type) is higher[2]. Moreover, the latent sector errors in disks are also not independent of each other[3]. Furthermore, according to the research of Panasas, the probability of block failure is approximately 30 times to the probability of disk failure. Thus, there may be an extreme scenario in storage system as follows: when we try to rebuild failed data, we discover that the disk containing the block has failed or that a sector in the block can not be read, and then we access the other blocks in the redundancy group in order to recover the failed data, finally, we find that many of the blocks have failed too. In such scenario, we are not able to reconstruct the data which are now lost. It is the reconstruction that unmasks failures in the system.

Techniques such as disk scrubbing[1][25][26] and intra-disk redundancy [27] have been proposed to avoid the above scenario and maintain high reliability. A widely used scheme is periodically detecting these failures by reading the disks. The detecting can check whether the disk has failed or not, and verify whether the data stored in the disk are correct or not. If there are sector errors, a repair process is launched to rebuild those failed data. This operation is called "scrubbing" or "scanning"[4].

In general, storage system designers use rules-of-thumb to determine the appropriate solutions to scan or scrub disks. These solutions can easily lead to an expensive, suboptimal system approach that fails to meet the users' requirements. Because every application is unique, the solution that works well for one application may not be applicable for another one. Users want solutions that meet their storage requirements to be in a cost-effective manner.

Based on the above discussion, designing a storage system requires to strike a good balance between the costs of the solution and the benefits that the solution can offer. If the scrubbing processes access disks frequently, it will consume more energy and result in a lower system performance. If the disks are accessed less frequently, the probability of block failure is higher and the reliability of system is lower. Therefore, a proper disk scrubbing

978-0-7695-3769-6/09 $26.00 © 2009 IEEE
DOI 10.1109/NCM.2009.26

scheme accounts not only for the needs of the reliability of system but also for the user's anticipation of performance and cost.

In order to solve this problem, we propose to use a frequency-cost function instead of the deterministic scrubbing [1] cycle or scrubbing frequency to keep a balance between the cost and the reliability. The purpose of this function is to guide storage system disk scrubbing by leveraging the feasibility of cost. The frequency-cost function provides an optimal cost-effective storage solution in terms of the user's cost/benefit trade-off manner. To our best knowledge, no one has used such frequent-cost function to designing the scrubbing scheme in storage system.

The rest of this paper is organized as follows. Related work is discussed in Section 2. We briefly introduce disk failure model and scrubbing-cost model in Section 3. Section 4 discusses how to get the optimal disk scrubbing frequency and the impact of scrubbing parameter on scrubbing frequency and the application field of scrubbing scheme. Section 5 summarizes our conclusions and presents future work.

II. RELATED WORK

Scrubbing technique was first proposed by Saleh[7]. This technique was used in the memory fault detect. Kari[4] adopted this technique to detect failed disk sectors and implemented three disk scrubbing algorithms.

S.M.A.R.T. Linux[23] is a bootable floppy distribution containing tool for monitoring IDE/SCSI hard disks by using Self-Monitoring, Analysis and Reporting Technology.

These disk scrubbing solutions mentioned above failed in considering the trade-offs between the scrubbing frequency, performance, reliability and cost. Schwarz et al.[1] proposed a scrubbing scheme which offers an optimal scrubbing interval, but the scheme didn't take into account the balance between the scrubbing frequency and system cost.

Bachmat and Schindler[9] analyzed how the low priority disk drive tasks such as disk scrubbing impact the performance. There has also been significant research reported in the literature regarding designing cost-effective disaster recovery solutions, trading off solution costs with expected penalties for data loss and downtime[9][10][11]. Those researches have effectively used utility to trade off the costs of data protection mechanisms against the penalties when data are lost, thus creating minimum (overall) cost solutions for disaster recovery. This result lends support to the notion of using busyness costs as the basis for evaluating storage solutions.

Strunk et al.[5] proposed using utility functions, instead of fixed requirements, as a way for the system administrators to communicate these underlying costs and benefits to an automated provisioning tool.

This tool provides a simplistic evaluation of low-level configuration parameters or overall system costs respectively. Neither class of tools can propose an optimum storage system solution to meet user-specified goals.

III. COST-EFFECTIVE SCRUBBING MODEL

A. Disk Failure Model

Paper[6] shows that the disk failure rates are not constant. Disk drives experience a high failure rate in the early-failure period. The failure rate drops in the first year, also known as infant mortality period. From the second year to the fifth year or even the seventh year, the failure rate will remain relatively constant, and this period is a useful life period. The annual failure rate at this period is approximately 8.8%. At the end of the disks lifetime, the failure rate will rise again, we call this period as a wear out period. In this period, the disk failure rate is very high and we should replace those disks which are experiencing this period. We talk about this failure model as bathtub model.

B. Assumptions

Since systems vary widely, we cannot derive generic prescriptions. In this section, we present some general assumptions that can provide guidance for choosing an optimal scrubbing scheme.

1. The probability distribution function of disk failure moment is F(t), probability density is f(t), and the disk lifetime is T, thus F(T)= $\int_0^T f(t)dt = 1$.

2. The data loss cost that disk run with failure is directly proportional to the time, and ratio coefficient Lc is the loss cost of a time unit. The ratio coefficient Lc can be described by the following formula as discussed in [5]: $Lc = \mathrm{K} \cdot AFR$. K denotes the data price of data which are stored in disk or storage system. In different enterprises, the data price K is different. The parameter K is \$100M in terms of [5] and the AFR is the annual failure rate, i.e. the disk failure rate per year. Generally, disk failure rate per hour is denoted by r(t). The disk annual failure rate r(t) is defined as[6]: $\mathrm{r(t)} = \dfrac{f(t)}{1-F(t)}$. Supposing that there are 8766 hours per year, then AFR=8766*r(t). We redefine the Lc=KL*r(t), and KL =8766*K.

3. We assume that the failure time between two successive scrubbing is a uniform distribution. There are two reasons for this assumption. First, when the scrubbing frequency is high in a time unit, we may consider that the failure time between two successive scrubbing follows the same distribution as the probability distribution of disk failure moment. Second, the assumption can simplify the mathematical model.

4. The cost for scrubbing is Sc, the frequency of

scrubbing up to the moment T can be expressed as $\int_0^t n(\tau)d\tau$.

C. Modeling the Cost-Effective Disk Scrubbing

We define the time interval between two successive scrubbings as a scrubbing cycle. We refer to the scrubbing scheme as how to determine the scrubbing cycle. The disk failure time is a random variable that follows an exponential distribution or a Weibull distribution. Once a disk fails, with the assumption that disk will run with failures until the next scrubbing, and this may cause considerable data loss. Obviously, the longer the scrubbing cycle, the greater the loss cost. At the same time, the scanning process may consume power, bring a downtime, and affect the disk performance. The shorter the scrubbing cycle, the more frequent the scrubbing, the higher the cost of scrubbing. Therefore, according to the assumption of a random distribution of failure, the loss cost, and the scrubbing cost, we need a stochastic optimization model to determine the scrubbing cycle, thus minimizing the overall average cost.

Generally, scrubbing cycle is not necessarily constant and should be based on the probability distribution of the failure time. Scrubbing cycle will be shorter when the probability of failure is lower, and vice versa. The probability distribution of disk failure is a continuous probability distribution function. Therefore, the scrubbing cycle is the function of time t, denoted as s(t).. We assume it is a continuous variable. The scrubbing frequency in one time unit is the function of time t, indicated as n(t). Obviously, $n(t) = 1/s(t)$. Generally, the scrubbing cycle is much shorter than the disk uptime, thus we can consider n(t) as a continuous function.

Assuming a disk keeps running until reaching its lifespan, the objective function of an optimal model is the mathematical expectation of total cost in a running process. If a disk failure is found in the interval $[t, t+\Delta t]$, according to the assumption 3, the failure time in the cycle s(t) is a uniform distribution, so the mean of uniform distribution is equivalent to the average value of the distribution interval. The time running with failure is $s(t)/2 = 1/2n(t)$, loss cost is $\frac{K_L \cdot r(t)}{2n(t)}$, and scrubbing cost is $S_c * \int_0^t n(\tau)d\tau$, then, the total cost is $C(n(t)) = \frac{K_L \cdot r(t)}{2n(t)} + S_c * \int_0^t n(\tau)d\tau$.

According to the assumptions, we can get a mathematical expectation of the total cost in a running progress:

$$E(C(n(t))) = \int_0^T \left[\frac{K_L \cdot r(t)}{2n(t)} + S_c * \int_0^t n(\tau)d\tau \right] f(t)dt \quad (1)$$

Equation (1) is the frequency-cost function of scrubbing scheme. Even it is possible to quantify the costs and benefits of a specific disk scrubbing solution by leveraging equation (1), it is still a challenge for a system designer to acquire an optimal scrubbing solution due to external constraints. For example, an " optimal " scrubbing scheme for a specific scenario may not be applicable for another scenario due to the lower reliability. Using frequency-cost function, the user has the ability to scale down this solution to find the best option that fits within the budget. As this example illustrates, frequency-cost function presents an opportunity to make trade-offs even among non-optimal or in less than ideal situations. Therefore, we have a chance to make trade-offs between reliability and cost by using this function.

The frequency of scrubbing should have been relative to the time interval, and it is a positive integer. In this model, we consider it as a continuous function of the time T. In this way, we can turn an optimization problem into a functional extreme value problem by adopting mathematical analysis tools.

IV. OPTIMAL SCRUBBING INTERVAL AND ANALYSIS

A. Optimal Scrubbing Interval

Scrubbing strategy is affected by multiple disk metrics simultaneously. By using the frequency-cost function, it is possible to take a holistic view of the problem and make optimal cost-effective choices.

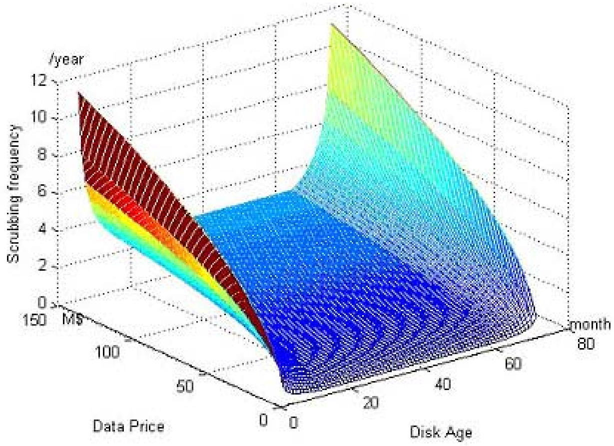
It is important that these choices are a balance of competing factors (mainly cost and reliability in this case). Taking this trade-off to an extreme, such as choosing a very high reliability solution regardless of cost, results in poor utility, and vice versa.

Implicit in the coarse description of disk scrubbing economics above equation (1) is the existence of a criterion function that depends on scrubbing frequency, that is, it means reliability and cost. Thus, an optimization process for minimizing the criterion function is required. Now we begin to find the optimal scrubbing solutions through the knowledge about calculus of variations as follows.

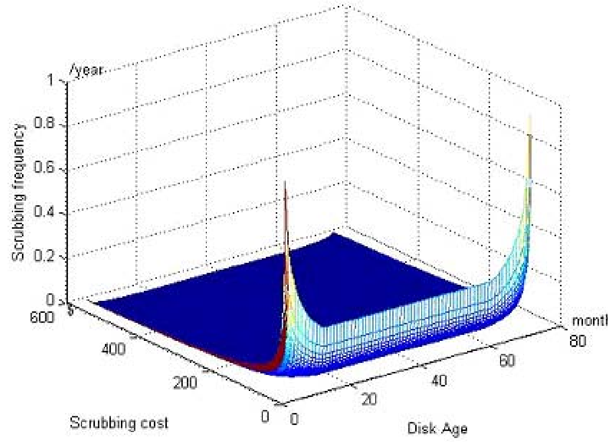
E(C(n(t))) is the functional of n(t) and it is our objective function. Our objective is to find a n(t) to minimize the C(n(t)). We can get this n(t) through math knowledge. n(t) must satisfy the following equation :

$$n(t) = \sqrt{\frac{K_L}{2S_c} r(t)} \quad (2)$$

The formula gives us a reference when a system administrator decides the scrubbing frequency. The scrubbing frequency is varying with the r(t). This result is not applicable for the repair operation performed by



(a)



(b)

Figure 1. (a) When scrubbing cost is constant, the impact of loss cost on scrubbing frequency with the change of disk age. (b) When loss cost is constant, the impact of scrubbing cost on scrubbing frequency with the change of disk age.

TABLE 1: THE SCRUBBING FREQUENCY ON DIFFERENT DATA PRICE AND DISK FAILURE RATE

r(t) \ Data Price ($)	0.01M	0.1M	1M	10M	100M	1000M
1.7%	0.07	0.23	0.73	2.31	7.30	23.10
2.0%	0.09	0.27	0.86	2.72	8.59	27.17
2.5%	0.11	0.34	1.07	3.40	10.74	33.97
3.0%	0.13	0.41	1.29	4.08	12.89	40.76
3.5%	0.15	0.48	1.50	4.76	15.04	47.56
4.0%	0.17	0.54	1.72	5.43	17.19	54.35
4.5%	0.19	0.61	1.93	6.11	19.34	61.14
5.0%	0.21	0.68	2.15	6.79	21.48	67.94
5.5%	0.24	0.75	2.36	7.47	23.63	74.73
6.0%	0.26	0.82	2.58	8.15	25.78	81.52
6.5%	0.28	0.88	2.79	8.83	27.93	88.32
7.0%	0.30	0.95	3.01	9.51	30.08	95.11
7.5%	0.32	1.02	3.22	10.19	32.23	101.90
8.0%	0.34	1.09	3.44	10.87	34.37	108.70
8.5%	0.37	1.15	3.65	11.55	36.52	115.49

administrator. However, we can get the disk failure rate at different disk age from disk specification (Disk failure rate at different disk age are given in some disk specifications), and we can run a process to monitor the disk age and compute the scrubbing frequency through the above formula.

B. The Analysis of Formula (2)

Figure 1 (a) confirms that the scrubbing frequency varies with the disk failure rate associated with the disk age. We suppose that the disk failure rate follows the bathtub curve discussed in section 3. We vary the loss cost between 0$ to 150M$. Like the bathtub curve, the 3D surface is also a bathtub surface. In the bathtub curve, the curve has a sharp decline when the disk age is in infant mortality. Figure 1 (a) shows that the scrubbing frequency n(t) increases in the disk infant mortality period with the increased data price. In the useful life period, the failure rate r(t) is a constant, but the n(t) is varying with the increase of cost loss. Obviously, the scrubbing frequency is increasing in the wear out period because the failure rate is increasing. The rate of increase in scrubbing frequency is not the same to the rate of failure, it is constrained by the loss cost. Additionally, in wear out period, the scrubbing frequency should not be the system administrator's key topic, the key topic should be how and when to replace the disks which are in wear out period.

When the loss cost is a constant, the impact of scrubbing cost on scrubbing frequency is shown by

Figure 1 (b). In the early of infant mortality, if the scrubbing cost is low, then we can scrub the disk at a higher frequency. With the increase of the scrubbing cost, the scrubbing frequency should be decreased. In the wear out period, we should replace the disks which are extremely unreliable. Figure 1 (b) shows that the scrubbing scheme is not suitable for the storage system consisting of high-capacity disks. The reason is listed as follows: First, the high-capacity disks result in long scrubbing time and more scrubbing cost. On the other hand, according to the researches[28][29], the average use of disk space is approximately 50%. There is a significant waste of resource to scrubbing free space.

Usually, Sc consists of the loss incurred by the performance degradation and the additional scrubbing power consumption. In this paper, we ignore the loss generated by the performance degradation. There are two reasons for doing so. First, the scrubbing operations are low priority tasks and the impact of scrubbing on performance is pimping. Second, it is difficult to weigh the impact of scrubbing on the system brought by the performance degradation. Thus, we adopt the scrubbing power consumption as the scrubbing cost. Generally, the power price is 7.5-8 ¢ per kw·h, and the disk bandwidth is approximately 60MB/S-70MB/S, so it takes approximately 120000/60*60*60=5/9 hour (On the assumption that the disk read speed is 60MB/S) to scrub a 120G disk. The power of a 120G disk is approximately 13W. Supposing that there are 1000 disks in a data center, the cost (power consumption) S_c for scrubbing all the disks is $13 \times 0.075 \times 1000 \times 5/9=1625/3$\$(Supposing the power price is 7.5¢ per kw·h), the scrubbing frequency n(t) can be computed by equation (13).

AFR is the annual failure rate of disks. Proverbially, the AFR is varied from 1.7% to 8.6%[24], when the scrubbing cost is constant. According to equation (13), the corresponding n(t) shown in Table 1 varies for 0.07 to 115 per year based on different data price and disk failure rate. The results mean that the optimal times for scrubbing all disks per year are decided by AFR and data price.

In Schwarz's paper[1], the optimal scrubbing is 3 times per year. His scheme supposes that the disk scrubbing is used by storage system such as MAID in which the disks often have a long idle period. In those storage systems, disk will power on or power off to scrubbing. However, this scheme may be not the optimal for a storage system in which the data is accessed frequently. In Schwarz's paper, it must consider the class of the raid when system administrator decides the scrubbing frequency. There is no uniform expression when we want to get scrubbing frequency. According to paper [3] and [26], the scrubbing process scans the entire disk at least once every two weeks. In other words, a realistic scrubbing frequency is at least 24 times per year. Table 1 illustrates that the scrubbing scheme is applicable to the storage system in which the preserved data is mezzo. Once the data is very expensive, the storage system must ceaselessly scan all the disks because overfull scrubbing results in worse system performance[9] and reliability[1]. So the storage system with extremely expensive data should adopt the other reliability mechanism. Apparently, if the data are inexpensive, it is low utility to provide excessively reliability.

V. CONCLUSIONS AND FUTURE WORK

Low-cost disk drives with higher capacity but lower reliability are more and more popular in data center. It leads to more frequent data loss coming from sector errors. The scrubbing scheme is adopted to keep storage system away from sector errors. It needs to balance two competing system metrics，i.e. cost and reliability，in order to choose a proper disk scrubbing scheme.

In this paper, we studied the impact of disk scrubbing frequency on system cost by using an analytical mathematical model. Our research has shown that scrubbing frequency is constrained by data loss cost, scrubbing cost, and disk failure rate，and the scrubbing scheme is applicable for storage with low-capacity disk and inexpensive data.

Scrubbing frequency could be influenced by other factors, such as system performance and system energy consumption. Therefore, our future work is to find a global optimum solution which takes into account all these factors. Additionally, we would investigate the disk failure log, conclude the disk failure distribute, and analyze the reliability of storage system using the theory of non-Markov process in reliability theory.

ACKNOWLEDGEMENTS

We would like to thank the anonymous reviewers for their constructive and helpful comments. This work is supported in part by National Basic Research 973 Program under Grant 2004CB318201, National Science Foundation of China (NSFC) under Grant 60703046 and 60873028, the Program for New Century Excellent Talents in University NCET-04-0693 and NCET-06-0650. the Program for Changjiang Scholars and Innovative Research Team in University(No.IRT-0725). 863 project 2008AA01A401

REFERENCES

[1] Thomas J. E. Schwarz, S.J. Qin Xin, Ethan L. Miller, Darrell D.E. Long. Disk Scrubbing Large Archival Storage Systems. In Proceedings of the 12th IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS 2004), October 2004.

[2] Weihang Jiang, Chongfeng Hu, Yuanyuan Zhou, and Arkady Kanevsky. Are Disks the Dominant Contributor for Storage Failures? A Comprehensive Study of Storage Subsystem Failure

Characteristics. The 6th USENIX Conference on File And Storage Technologies (FAST '08), 2008.
[3] Lakshmi N. Bairavasundaram, Garth R. Goodson, Shankar Pasupathy, Jiri Schindler. An Analysis of Latent Sector Errors in Disk Drives. ACM SIGMETRICS Int'l. Conference on Measurement and Modeling of Computer Systems. Jun. 2007.
[4] H. H. Kari. Latent Sector Faults and Reliability of Disk Arrays. PhD thesis, Helsinki University of Technology, 1997.
[5] John D. Strunk, Eno Thereska, Christos Faloutsos, Gregory R. Ganger. Using Utility to Provision Storage Systems. The 6th USENIX Conference on File And Storage Technologies (FAST '08), 2008.
[6] Bianca Schroeder, Garth A. Gibson. Disk failures in the real world: What does an MTTF of 1,000,000 hours mean to you? The 5th USENIX Conference on File And Storage Technologies (FAST '07), 2007.
[7] A. M. Saleh, J. J. Serrano, J. H. Patel, "Reliability of Scrubbing Recovery-Techniques for Memory Systems", IEEE Transactions on Reliability, vol. 39, no. 1, April 1990, pp. 114-122.
[8] T. Schwarz. Verification of parity data in large scale storage systems. In Parallel and Distributed Processing Techniques and Applications (PDPTA '04), Las Vegas, NV, June 2004.
[9] Eitan Bachmat, Jiri Schindler. Analysis of Methods for Scheduling Low Priority Disk Drive Tasks. ACM SIGMETRICS Int'l. Conference on Measurement and Modeling of Computer Systems. Jun. 2002.
[10] G. A. Alvarez, J. Wilkes, E. Borowsky, S. Go, T. H. Romer, R. Becker-Szendy, R. Golding,A. Merchant, Spasojevic, and A.Veitch. Minerva: An automated resource provisioning tool for large-scale storage systems. ACM Transactions on Computer Systems, 19(4):483 – 518, November 2001.
[11] K.Amiri and J. Wilkes. Automatic design of storage systems to meet availability requirements. Technical Report HPL – SSP – 96 – 17, Hewlett-Packard Laboratories, August 1996.
[12] E.Anderson,M.Kallahalla,S.Spence,R.Swaminathan, and Q. Wang. Ergastulum: Quickly finding near-optimal storage system designs. Technical Report HPL – SSP – 2001 – 05, Hewlett-Packard Laboratories, 2001.
[13] E. Anderson, M. Hobbs, K. Keeton, S. Spence, M. Uysal, and A. Veitch. Hippodrome: Running circles around storage administration. In Conference on File and Storage Technologies, pages 175 – 188. USENIX Association,2002.
[14] E. Anderson, S. Spence, R. Swaminathan, M. Kallahalla, and Q. Wang. Quickly finding near-optimal storage designs. ACM Transactions on Computer Systems, 23(4):337 – 374, November 2005.
[15] A. AuYoung, L. Grit, J. Wiener, and J. Wilkes. Service contracts and aggregate utility functions. In International Symposium on High-Performance Distributed Computing,pages 119 – 131. IEEE, 2006.
[16] J. O. Kephart and W. E. Walsh. An artificial intelligence perspective on autonomic computing policies. In International Workshop on Policies for Distributed Systems and Networks, pages 3 – 12. IEEE, 2004.
[17] M. P. Mesnier, M. Wachs, R. R. Sambasivan, A. Zheng, and G. R. Ganger. Modeling the relative fitness of storage. In Conference on Measurement and Modeling of Computer Systems, pages 37 – 48. ACM Press, 2007.
[18] W. E.Walsh, G. Tesauro, J. O. Kephart, and R. Das. Utility functions in autonomic systems. In International Conference on Autonomic Computing, pages 70 – 77. IEEE, 2004.
[19] J. Wolf. The Placement Optimization Program: A practical solution to the disk file assignment problem. Performance Evaluation Review, 17(1):1 – 9, May 1989.
[20] http://www.copansystems.com/products/disc_aerobics.php
[21] Xiaoyan Zhu,Jinhong Cao. The application of bathtub curve in the design and management of reliability. China Quality.7:25-27,2007.
[22] Kimberly Keeton, Cipriano Santos, Dirk Beyer, Jeffrey Chase, John Wilkes. Designing for disasters. The 2th USENIX Conference on File And Storage Technologies (FAST '04), 2004.
[23] http://smartlinux.sourceforge.net/
[24] Eduardo pinheiro,Wolf-Dietrich Weber and Luiz Andr´e Barroso. Failure Trends in a Large Disk Drive Population. The 5th USENIX Conference on File And Storage Technologies (FAST '07), 2007
[25] Mary Baker, Mehul Shah, David S.H. Rosenthal. A Fresh Look at the Reliability of Long-term Digital Storage.In Proceedings of the ACM SIGOPS/EuroSys European Conference on Computer Systems (EuroSys 2006),2006
[26] Ilias Iliadis, Robert Haas, Xiaoyu Hu, and Evangelos Eleftheriou. Disk Scrubbing Versus Intra-Disk Redundancy for High-Reliability RAID storage System. In Proceedings of the 2008 ACM SIGMETRICS international conference on Measurement and modeling of computer systems. 2008
[27] A. Dholakia, E. Eleftheriou, X.-Y. Hu, I. Iliadis, J. Menon, and K. Rao. A new intra-disk redundancy scheme for high-reliability RAID storage systems in the presence of unrecoverable errors. ACM Transactions on Storage (TOS). Vol. 4, No. 1. May. 2008, pp. 1-42.
[28] John Douceur and William Bolosky, A Large-Scale Study of File-System Contents, ACM SIGMETRICS Performance Review, 59–70, 1999.
[29] Hai Huang, Wanda Hung, and Kang G. Shin, FS2: Dynamic Data Replication in Free Disk Space for Improving Disk Performance and Energy Consumption, 18th ACM Symposium on Operating Systems Principles (SOSP), 2005.