Transactions on Computer Science and Technology
June 2014, Volume 3, Issue 2, PP.25-34

An Efficient File Assignment Strategy for Hybrid Parallel Storage System with Energy & Reliability Constraints

Xupeng Wang, Wei Jiang[#], Hang Lei, Xia Zhang

School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu 610000, China

[#]Email: weijiang@uestc.edu.cn

Abstract

In this work, we are interested in the file assignment problem in a distributed file system. We adopt a hybrid parallel storage system consisting of hard and flash disks, and then address the problem of minimizing the system's mean response time by determining the distribution of the files in the system. In addition, energy efficiency and system reliability are all taken into consideration and regarded as the system constraints. Due to the complexity of the problem, we propose our Two Stage File Assignment algorithm (TSFA) to find an optimized solution with predefined constraints. The efficiency of our algorithm is verified by extensive experiments.

Keywords: Parallel I/O System; Flash Disk; Energy Conservation; System Reliability; FAP

1 INTRODUCTION

In recent years, digital data produced by users and applications has experienced an explosive growth. For example, image-intensive applications, such as video, hypertext and multimedia, generate an incredible amount of data every day. Accompanied by the emerging Big Data, prompt response to access request is consistently required to be further improved by end-users. Parallel storage systems like RAID (Redundant Array of Inexpensive Disks) [1] have been widely used to support a wide range of data-intensive applications, which distribute data across multiple disks and access requests are serviced in a parallel way.

Compared with hard disks, flash-memory based solid state disks have remarkable superiorities in energy consumption, access latency, data transfer rate, density and shock resistance. Its use as a storage alternative has seen a great success in the area of mobile computing devices [2], and been expended into personal computer and enterprise server markets. The main concern on current flash disks is their considerably high price and inadequacy in write cycles. Therefore, it is wise and practical to integrate flash disks with hard disks to form a hybrid parallel storage system to fully exploit their complementary merits [2][3][4].

Data should be properly assigned to disks of the system before being accessed, which is typically referred to as File Assignment Problem (FAP). A lot of researches have been done in this literature [6]-[11] aiming at quick response. Generally, the algorithms can be divided into two categories: static and dynamic file assignment strategy. To be specific, the former requires a complete knowledge of workload characteristics as a prior, while the latter generates file assignment schemes on-line by adapting to varying workload patterns. However, all these researches did not take into consideration the case of disks with different data transfer rates, which simplifies the problem.

Energy efficiency is another fundamental requirement in the use of parallel storage system [12]-[14]. Optimizing energy consumption has a great impact on the cost of backup power-generation and cooling equipment, because a large proportion of cost is incurred by energy consumption and cooling. Since placing files on different disks leads to different energy consumptions, how to design an energy efficient file assignment algorithm is a great challenge.

In this paper, we identify the problem of file assignment in hybrid parallel storage system. Specifically, we want to

propose a file assignment strategy with the objective of minimizing system's mean response time while satisfying energy requirement. The main contributions of this paper are in two aspects. Firstly, we formally formulate the file assignment problem in hybrid parallel storage system. Secondly, we propose an efficient algorithm called Two Stage File Assignment (TSFA) which guarantees the prompt response and energy-conservation requirements.

The rest of the paper is organized as follows: Section 2 formally depicts the system model. Section 3 formulates the file assignment problem to be addressed in this paper. Section 4 and section 5 present our proposed algorithm and the experiment results respectively. Section 6 concludes the paper with a summary and future work.

2 SYSTEM MODEL

2.1 System Architecture

FIG.1 depicts the system architecture explicitly. The file assignment algorithm is applied to the parallel storage system to distribute data properly before being accessed. In this paper, it is implemented in the File Allocation Module. After completion of file allocation, substantial access requests generated by users arrive, which will first be served by the Scheduler in the First Come First Served order. Afterwards, the Translation Module directs the access requests to their targeting disks, with the aid of the Mapping Table produced by the File Allocation Module. In addition, the system maintains one Local Queue at each node, where access requests are kept in their arriving order. Compared with hybrid disks, the time spent on these modules is so minimal that can be ignored.

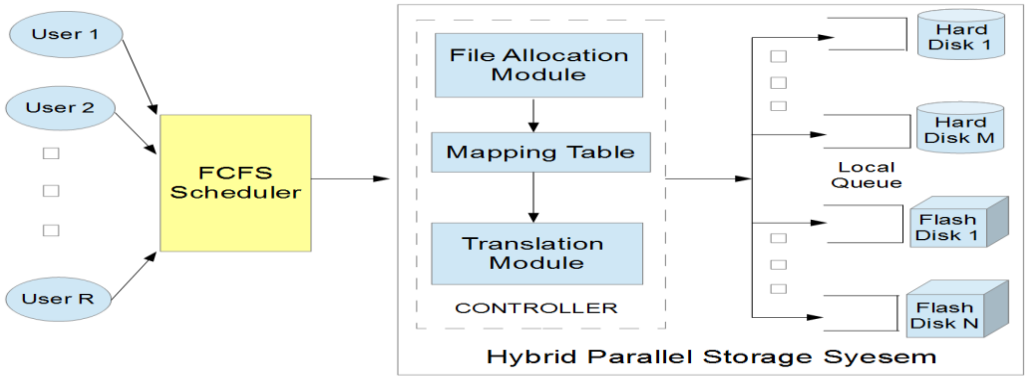


FIG. 1 SYSTEM ARCHITECTURE

In this paper, we adopt a hybrid storage system comprising M hard disks and N flash disks, whose characteristics are assumed to be the same respectively. It can be modelled as $DA=\{hd_1,...,hd_m, fd_1,..., fd_n\}$. Each flash disk is formulated as $fd=(r_f, w_f, ac_f, id_f)$: r_f is its read rate measured in Mbyte/second; w_f is write rate in Mbyte/second; ac_f and id_f represent active and idle energy consumption rate in Watts respectively. Similarly, tr_h demonstrates hard disk's ability of data transfer for both read and write in Mbyte/second. ac_h and id_h are energy consumption rates in Watts, corresponding to active and idle mode. In addition, we let *La* denote the sum of seeking and rotation latency of a hard disk, while the overhead is so minimal for flash disks as to be ignored in this paper.

The set of files to be distributed among the system are represented by $F=\{f(1),..., f(l)\}$. We assume that each disk is sufficient to accommodate all the files assigned to it. After assignment, files on hard disks are denoted by $F_h=\{f_h(1),..., f_h(m)\}$, and those on flash disks are $F_f=\{f_f(1),..., f_f(n)\}$. For the sake of simplicity, we assume that each file must be allocated in its entirety to one disk. In other words, we exclude the case of file partition or replication. In addition, all the files are isolated and accesses to each file can be view as independent.

Intensive access requests, consistently sent out by users, are modelled as $R=\{r(1),...,r(u)\}$, while each request can be represented by $r(k)=\{f(i),\lambda_i,t(i)\}$. In this paper, access request to file f_i is thought to be a Poisson process with a mean access rate λ_i. In addition, we assume service time of access request is fixed after files being assigned to disks. This assumption can be accepted as valid in that for most file systems or WWW servers the access to a file

tends to result in a sequential scan of the entire file, and for large files when the file access unit accounts for a large fraction, the seeking and rotation delays are negligible in comparison with the data transfer time.

2.2 Mean Response Time

For data-intensive server class applications, mean response time is the most important measurement that reflects system's ability of serving client requests. Given two file characteristics, λ_i and t_i, we combine them to get a metric that demonstrates the file's pressure on the system, called heat h_i and defined as $h_i = \lambda_i * t_i$. Let *I(k)* stand for the set of files assigned to disk d_k. Its utilization ρ_k, or the pressure given by *I(k)*, can be derived from $\rho_k = \sum_{i \in I_k} h_i$.

We model each disk as a single M/G/1 queue. Then, the mean response time of all the requests at disk d_k is given as

$$E(r_k) = E(s_k) + \frac{\Lambda_k * E(s_k^{\,2})}{2 * (1 - \rho_k)} \tag{1}$$

where $E(s_k)$ and $E(s_k^2)$ represent the mean and mean-square service time respectively; Λ_k is the aggregate access rate defined as $\Lambda_k = \sum_{i \in I_k} \lambda_i$.

Furthermore, the probability of access to file f_i at disk d_k is defined as $p_i^{(k)} = \lambda_i / \Lambda_k$.

Then, $E(s_k)$ and $E(s_k^2)$ can be computed as follows

$$E(s_k) = \sum_{i \in I_k} p_i^{(k)} * s_k = \frac{1}{\Lambda_k} * \sum_{i \in I_k} \lambda_i * s_i \tag{2}$$

$$E(s_k^{\,2}) = \sum_{i \in I_k} p_i^{(k)} * s_k^{\,2} = \frac{1}{\Lambda_k} * \sum_{i \in I_k} \lambda_i * s_i^{\,2} \tag{3}$$

The mean response time of access requests to disk d_k can be simplified as

$$E(r_k) = \frac{\rho_k}{\Lambda_k} + \frac{\Lambda_k * E(s_k^{\,2})}{2 * (1 - \rho_k)} = \frac{\rho_k}{\Lambda_k} + \frac{\sum_{i \in I_k} \lambda_i * s_i^{\,2}}{2 * (1 - \rho_k)} \tag{4}$$

With the file assignment strategy $I = \{I_1, I_2, ..., I_k\}$, system's mean response time can therefore be given as

$$E(r) = \sum_{k=1}^{m} \frac{\Lambda_k}{\Lambda} * E(r_k) = \frac{1}{\Lambda} * \sum_{k=1}^{m} \rho_k + \frac{1}{2 * \Lambda} * \sum_{k=1}^{m} \frac{\Lambda k}{1 - \rho k} * \sum_{i \in I_k} \lambda_i * s_i^{\,2} \tag{5}$$

2.3 Energy Gain

In this paper, we consider a two-level energy consumption model for both flash and hard disks: standby and access mode. To be specific, energy consumption can be divided into standby and access consumption, where the former refers to the energy consumed when the disk is idle, and the latter represents the energy consumption caused by read and write operations. For intensive server-level workloads, the short time intervals between requests at one disk make it effortless for disks to spin up and down to save energy. So this assumption is always recognized to be valid.

Since flash disks have a significant advantage in energy conservation over hard disks, their energy consumptions differ greatly with the same file assigned to them. Given a file *f(i)*, we define its energy cost as follows

$$Cost_{hd}(i) = P(i) * t_{WH}(i) * ac_h + (1 - P(i)) * t_{RH}(i) * ac_h \tag{6}$$

$$Cost_{fd}(i) = P(i) * t_{WF}(i) * ac_f + (1 - P(i)) * t_{RF}(i) * ac_f \tag{7}$$

Here, we assume for *f(i)* the percentage of write requests *P(i)* is fixed, which can be derived from $P(i) = \lambda_w(i) / \lambda(i)$.

Therefore, the energy gain of file f_i can be defined as

$$Egain(i) = Cost_{hd} - Cost_{fd} \tag{8}$$

where *Egain(i)* demonstrates the energy conservation achieved by allocating file f_i to flash disks instead of hard disks.

2.4 Reliability Loss

Although flash disks have comparative advantages in high performance, low energy cost and shock resistance, they do bear the shortage of insufficient write endurance. According to the disk characteristics listed in Table 1, total number of write cycles of the flash disk is limited, which is claimed to be 146,000 and denoted by WC,. In addition, the usage-based warranty or duration years of the flash disks noted as DY are set to be 5 years. Then, we define WCPS, which stands for write cycles per second and is the measurement of flash disk's endurance

$$WCPS = \left(WC \diagup DY\right) \diagup (365 * 24 * 60 * 60) \tag{9}$$

In this paper, the value is approximately 0.001(1/second). Given a flash disk *fd(i)* with all of its requests $R_{hd}(i) = \{r_{hd}(1), ..., r_{hd}(u)\}$, the reliability loss is defined as follows:

$$RL(i) = \begin{cases} 0, if \sum_{K=0}^{U} \lambda_w(k) \le WCPS \\ \sum_{k=0}^{u} \lambda_w(k) - WCPS, else \end{cases} \tag{10}$$

TABLE 1 DISK CHARACTERISTICS

	Hard Disk	**Flash Disk**
Model Number	Seagate ST4000NM0023	Seagate ST480FP0021
Disk Number	4	3
Read(Mbyte/second)	175	750
Write(Mbyte/second)	175	500
Latency(ms)	4.16	~0
Active Power(Watts)	11.27	4.05
Idle Power(Watts)	6.73	3
Guarantee Period(year)	10	5

3 PROBLEM FORMULATION

3.1 Problem Identifying

In this section, we formally formulate the file assignment problem to be addressed in this paper. There is a hybrid storage system denoted by $D = \{hd_1, ..., hd_m, fd_1, ..., fd_n\}$ and a set of files to be assigned to the system represented by $F = \{f_1, ..., f_j, ..., f_k\}$. We model a way of file assignment as $I(i) = \{I_{hd1}, ..., I_{hdm}, I_{fd1}, ..., I_{fdn}\}$, where I_{hdi} and I_{fdi} stand for the set of files on hard disk hd_i and the flash disk fd_i respectively. We aim to handle this problem by figuring out an optimized file assignment strategy *I(i)* that satisfies certain performance requirements.

3.2 Objective

For data-intensive server-class applications, the mean response time is recognized as the most important performance. Therefore, to obtain an optimized solution *I* to this problem, our design objective can be formulated as

$$E(r) = \sum_{i=1}^{m} \frac{\Lambda_i}{\Lambda} * E_{hd}(r_i) + \sum_{j=1}^{n} \frac{\Lambda_j}{\Lambda} * E_{fd}(r_j) \tag{11}$$

Since flash disks have a much better performance than hard disks in data transfer rate, service time of a file differs greatly when accommodated by the two storage mediums. Based on the assumptions mentioned above, the average service time $s_{hd}(i)$ and $s_{fd}(i)$ of a file *f(i)* on hard disks and flash disks can be calculated as follows respectively

$$s_{hd}(i) = p(i) * t_{Whd}(i) + (1 - p(i)) * t_{Rhd}(i) \tag{12}$$

$$s_{fd}(i) = p(i) * t_{Wfd}(i) + (1 - p(i)) * t_{Rfd}(i) \tag{13}$$

The mean response time of each hard disk hd_k or flash disk fd_k is given correspondingly as

$$E_{hd}(rk) = \frac{\rho_k}{\Lambda_k} + \frac{\Lambda_k * E_{hd}(s_k^{\,2})}{2 * (1 - \rho_k)} = \frac{\rho_k}{\Lambda_k} + \frac{\sum_{i \in I_{hdk}} \lambda_i * s_{hd}^{\,2}(i)}{2 * (1 - \rho_k)} \tag{14}$$

$$E_{fd}(rk)=\frac{\rho_k}{\Lambda_k}+\frac{\Lambda_k * E_{fd}(s_k{}^2)}{2*(1-\rho_k)}=\frac{\rho_k}{\Lambda_k}+\frac{\sum_{i\in I_{fdk}}\lambda_i * s_{fd}{}^2(i)}{2*(1-\rho_k)} \quad (15)$$

Therefore, the parallel I/O system's mean response time can be formulated as

$$E(r)=\frac{1}{\Lambda}*\sum_{i=1}^{m}\rho_i+\frac{1}{2*\Lambda}*\sum_{i=1}^{m}\frac{\Lambda_i}{1-\rho_i}*\sum_{l\in I_{hd}(i)}\lambda_l * s_l{}^2+\frac{1}{\Lambda}*\sum_{j=1}^{m}\rho_j+\frac{1}{2*\Lambda}*\sum_{j=1}^{n}\frac{\Lambda_j}{1-\rho_j}*\sum_{l\in I_{fd}(j)}\lambda_l * s_l{}^2 \quad (16)$$

Once all the files are distributed across the distributed system, the system's mean response time is determined only by the way of file assignment. Since FAP (file assignment problem) is an NP-complete problem, our objective is to propose a heuristic solution $I=\{I_{hd1},...,I_{hdm},I_{fd1},...,I_{fdn}\}$ that aims to optimize system's mean response time *E(r)*.

3.3 Constraints

After the completion of file assignment, substantial requests are served in parallel. Different ways of file assignment lead to different system performances, such as energy consumption, reliability and access delay. Since the optimized response time of the system can be achieved by balancing the loads and minimizing variance of the service times [10], the objective should be reached considering energy, reliability, disk utilization and service time of file requests.

1) Energy

Given all the files existing on flash disks, $F_f=\{f_f(1),...,f_f(l)\}$, the sum of energy conservation achieved by introducing flash disks can be calculated by

$$Egain=\sum_{i=1}^{l}Egain(i) \quad (17)$$

Since energy is one of the primary concerns in the storage system, we should try to reduce the energy consumption as much as possible in the design of the file placement strategy, in other words, maximizing *Egain*.

2) System Reliability

Flash disks can endure only a limited number of write cycles, while that of hard disks is often regarded to be infinite. Since the system is made up of the two mediums, the flash disk's endurance has a great effect on the system's reliability. Therefore, for each flash disk fd_i, its reliability loss *RL(i)* has to be kept as zero. In other words, the sum of write access requests on fd_i cannot violate the upper bound of the write cycles

$$\sum_{i=0}^{u}\lambda_w(i)\le WCPS \quad (18)$$

3) Load Balancing

Load refers to the amount of work to be dealt with by each disk. Balancing the load contributes to the minimization of the average latency of the system [10]. We say that $I=\{I_1,I_2,...,\mathrm{I_m}\}$ is a perfectly balanced file assignment if all the disks' utilizations are ρ_0, where $\rho_0=\frac{1}{m}*\sum_{i=1}^{n}\lambda * s_i$.

Therefore, each disk should be assigned files with the load of ρ_0, so that the work is distributed more or less evenly.

4) Minimal Variance of Service Time

Most of the work concentrates on minimizing the disk utilization by balancing the system load across all disks, but neglects the influence of request service time. As a matter of fact, the performance of the distributed file system can be greatly improved by reducing the variance of service times at each disk in addition to balancing the load [10].

4 PROPOSED TECHNIQUE

In this section, we formally present our Two Step File Assignment algorithm. The main purpose of our algorithm is to assign a set of files to the hybrid parallel storage system properly and make sure that the system operates with the minimal energy consumption whilst satisfying the reliability constraint. The comprehensive procedure is demonstrated by the pseudo-code.

Two Stage File Assignment Algorithm

Stage 1: Files Division

1: for *i= f(1)* to *f(l)* do

2: if $\lambda w_i = 0$ then $f_i \in fd - preferred$

3: else if $\lambda w_i \geq WCPS$ then $f_i \in hd - preferred$

4: else $f_i \in read - write$

5: compute priority of read-write files and then sort them into list I_{rw} in decent order

6: divide read-write files of list I_{rw} into *hd-preferred* and *fd-preferred*

7: for $i= f_{rw}(1)$ to $f_{rw}(l')$

8: if $reliabilityloss \geq \mathrm{n} * \mathrm{WCPS}$ then

9: $reliabilityloss = reliabilityloss + \lambda w_i$

10: $f_i \in fd - preferred$

11: else $f_i \in hd - preferred$

Stage 2: Files Distribution

12: sort *hd-preferred* and *fd-preferred* files into list I_{hd} and I_{fd} in descent order correspondingly according to service time t_i

13: computer the average disk utilization ρ_{hd} ρ_{fd} for hard disks and flash disks respectively

$$\rho_{hd} = \frac{1}{m}\sum_{i=1}^{l_{hd}} h_i \quad \rho_{fd} = \frac{1}{m}\sum_{i=1}^{l_{fd}} h_i$$

14: Assign to each hard disk the next contiguous segment of I_{hd} until its load, $load_j$, reaches the maximum allowed level ρ_{hd}

15: for j=1 to M do

16: While $\mathrm{load_j} \geq \rho_{hd}$ and $\mathrm{i} \geq n$ do

17: $I_j = I_j \cup \{i\}$

18: $load_j = load_j + h_i$

19: do the same to flash disks as steps 14-18

Generally speaking, our Two Stage File Assignment algorithm is composed of two stages: Files Division and Files distribution. The first stage tries to divide the entire set of files into a flash-preferred subset and a hard-preferred subset based on the file's write access rate, aiming to exploit advantages of flash disks and at the same time ensure reliability of the system. Then, each subset of files is judiciously distributed onto their favourite disks in stage 2.

Initially, we divide all the files into three sets by comparing their write access rate λ_{wi} with WCPS. There is no doubt that read-exclusive files and write-excessive files should be assigned to flash disks and hard disks respectively. In our Two Stage File Assignment algorithm, each read-write file is given a priority for assignment to flash disks. We keep on assigning files in their descent order of priority as long as their accumulation of write cycles is within flash disks' endurance. The others are classified as the hd-preferred files. The priority, taken into consideration file size, access rate and energy gain, can be derived from

$$priority_i = R_i * \frac{1}{P(i) * s(i)} \tag{19}$$

In stage 2, firstly, all the files belonging to flash disks and hard disks are assigned to lists I_{hd} and I_{fd} in descending order of service time, while the disks are selected for allocation in random order. Each disk is assigned the next contiguous segment from I_{hd} or If_d such that the load is distributed among the disks more or less evenly.

5 EXPERIMENTAL EVALUATION

In this section, we demonstrate the advantage of flash disk over hard disk intuitively by comparing performances of the distributed file system operating on hybrid storage architecture and hard disks only. Further, we present a comprehensive evaluation of the proposed Two Step File Assignment algorithm and make a comparison with the well-known PB-PDC algorithm.

5.1 Experimental Setup and Workload Characteristics

We have developed an execution-driven simulator that models the hybrid storage system consisting of six hard disks and four flash disks. The detailed characteristics of the two kinds of disks are listed in table 1. Compared with the time spent on disk accesses, delay on the scheduler and the internet is so minimal as to be neglected. As a result, the lifetime of each request is thought to be merely made up of the delay on the disk queue and its service time.

All the experiments are performed with synthetic workloads and we try to distribute 5,000 files among the system. For the sake of simplicity, each file is assumed to exist in one single disk without partition or replication. In addition, each file access experiences a sequential read or write of the entire file. Therefore, its service time can be regarded as fixed after assignment. According to the observations in real system traces, the majority of accesses typically concentrate on files with large sizes, while small files are relatively unpopular. In order to demonstrate the effectiveness of our algorithm in reality, the distributions of file sizes and file accesses across the files are set to be inversely correlated, and exhibit a Zipfian distribution with a skew parameter $\theta = \log(\frac{X}{100} - \frac{Y}{100})$, which means X percent of accesses are directed to Y percent of files. We also model the percentage of write accesses to each file as a normal distribution. Besides, the access to each file is supposed to be a normal distribution with the mean access rate λ_i. Therefore, the interval time of accesses to file fi is exponentially distributed with a fixed mean $1/\lambda_i$.

5.2 Experimental Results

We compare our TSFA in hybrid storage system with the algorithm of PB-PDC and the hard disk only system, and focus on two main performance metrics of distributed file system: mean response time and energy consumption. This collection of experiments demonstrates convincingly that our TSFA consistently generates excellent performances.

1) Impact of Aggregate Access Rate

Aggregate access rate measures the workload's pressure on the distributed file system. This series of experiments are designed to disclose the impact of the aggregate access rate on the algorithms, and examine their abilities to deal with the variable. We randomly generate file sets of different aggregate access rates ranging from 10(1/second) to 220(1/second).

We come up with the term "thrashing point" to depict the situation where one disk is constant in the active mode and the arrived request always has to wait for the terminal of the one ahead occupying the disk. With the gradual increase of the aggregate access rate, disks will sooner or later reach their "thrashing point" definitely. Afterwards, the mean response time will increase by more than one order of magnitude, while the energy consumption continues to be in its highest level.

We found that TSFA consistently provides the best performance both in mean response time and energy consumption. Fig. 2 explicitly shows that with the application of TSFA the system encountered two "thrashing points": 140(1/second) and 200(1/second). At the point of 140, hard disks reach the thrashing point for the first time while flash disks still have spare ability to cope with the increasing load. After 200(1/second), the utilizations of hard disks and flash disks are all utilized to the largest extent. When it comes to PB-PDC, the bottle- neck comes a little

earlier than TSFA at 100(1/second) and 180(1/second) correspondingly. As a result, compared with PB-PDC, TSFA continue to gain energy conservation until all the disks reach ”thrashing point” with 20 percent improvement of mean response time. This is because our algorithm fully exploits the advantage of flash disks and assigns part of the read-write files to them. Consequently, load of the file system is decreased and, more importantly, distributed among the disks in a more evenly way. As for the distributed file system on hard disks, it is always the worst case.

2) Impact of Skew Parameter

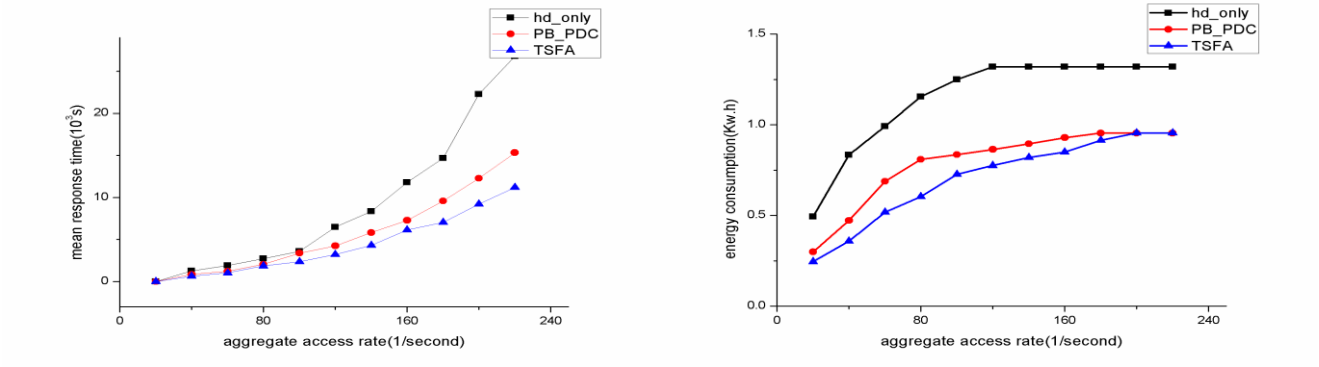
The skew parameter explicitly determines the distribution of access requests across the files. In this section, we verify the impact of skew parameter θ by setting it to 2.33(70/30) and 1.5(60/40).

As Fig. 2 and Fig. 3 show, the qualitative ranking of the three algorithms do not change for different skew parameters. However, with skew parameter decreasing from 2.33 to 1.5, all the “thrashing points” are shifted to a smaller aggregate access rates. In addition, the mean response times become a little longer and increase more sharply. Energy consumptions of the system reach their peaks in a more quickly way.

The decline of parameter skew from 70/30 to 60/40 leads to an even distribution of access requests in which files with different sizes and service times have little difference in access requests. On one hand, load of the system is comparatively enlarged. On the other hand, it will happen much more frequently that small file accesses have to wait for larger file accesses that were queued ahead of them.

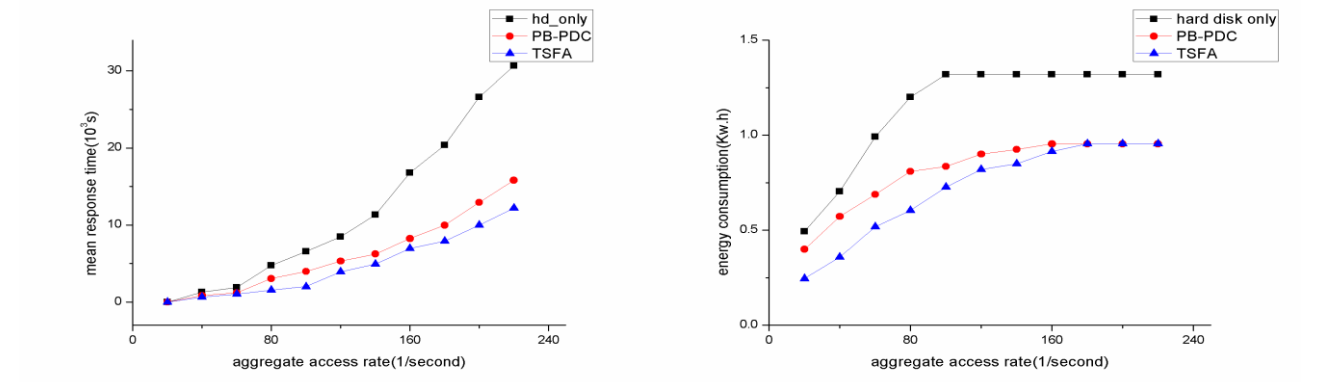
3) Impact of Write Access Proportion:

Flash disks have superior performances except for its relatively weak endurance. Since our algorithm is designed based on the characteristics of flash disks, files’ write access requests have a significant effect on its performance. The goal of this experiment is to figure out the influence of write access proportion on the algorithms. To be specific, we assume that the proportion of write access request across all the files exhibits a normal distribution with its mean percentage set to 40% 50% and 60%. In addition, the aggregate access rate is fixed at 100(1/second) and the file sizes are randomly distributed according to Zif’s law with skew degree 70/30.



(a) The impact of aggregate access rate on mean response time (b) The impact of aggregate access rate on energy consumption

FIG. 2 PERFORMANCE CORRESPONDING TO SKEW PARAMETER (70/30)

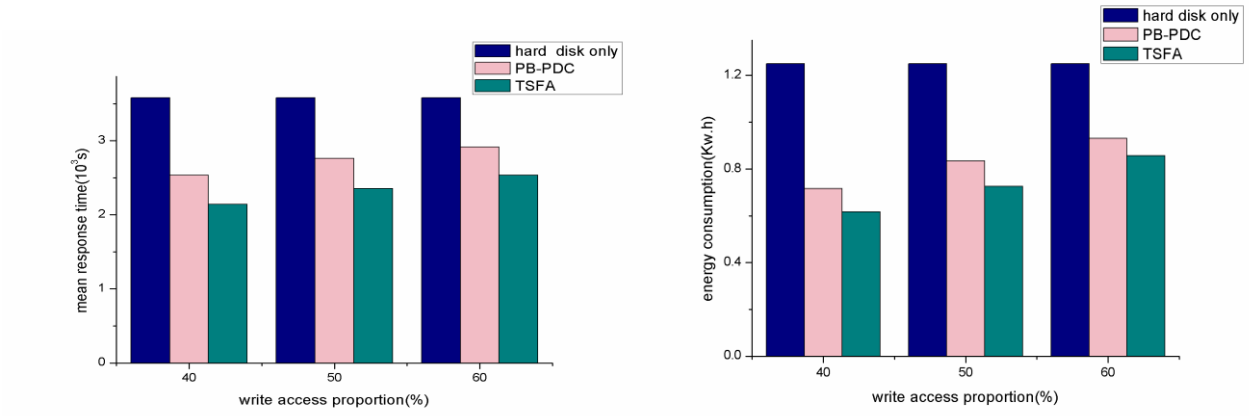


(a) Impact of Aggregate Access Rate on Mean Response Time (b) Impact of Aggregate Access Rate on Energy Consumption

FIG. 3 PERFORMANCE CORRESPONDING TO SKEW PARAMETER (60/40)

Fig. 4 demonstrates that the mean response time and energy consumption corresponding to our proposed TSFA continues to mount while the write access proportion increases from 40% to 60%. Similarly, PB-PDC responses in a intenser way. On the contrary, the two metrics of hard disk only system remain without any change.

The increase of write access proportion directly transfers a number of files from flash disks to hard disks in the algorithm of TB-TDC, while the number is relatively small in our TSFA in that files with smaller write access rate than WCPS are kept in flash disks.



(a) Impact of Write Access Proportion on Mean Response Time (b) Impact of Write Access Proportion on Energy Consumption

FIG. 4 PERFORMANCE CORRESPONDING TO WRITE ACCESS PROPORTION

6 CONCLUSIONS

We have presented a novel static file assignment algorithm, Two Stage File Assignment, to address the problem of distributing files across a hybrid storage system. To verify the effectiveness of our algorithm, we conducted a series of experiments based on synthetic workloads. Experiment results demonstrate convincingly that TSFA generates impressive performances over hard disk only distributed system and the famous PB-PDC algorithm.

In the future, our studies in this area should be performed in the following directions. Firstly, we will take the number of disks into consideration to perfect our algorithm. This may significantly improve the scalability of TSFA. Secondly, we will try to solve the issue of dynamic file assignment, where files' characteristics are not known ahead of time and further may change over time.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for their constructive comments and suggestions that have helped improved the quality of this manuscript.

REFERENCES

[1] Chen P M, Lee E K, Gibson G A, et al. RAID: High-performance, reliable secondary storage[J]. ACM Computing Surveys (CSUR), 1994, 26(2): 145-185.Choi, Mihwa. "Contesting Imaginaires in Death Rituals during the Northern Song Dynasty." PhD diss., University of Chicago, 2008

[2] Kim Y J, Kwon K T, Kim J. Energy-efficient file placement techniques for heterogeneous mobile storage systems[C]//Proceedings of the 6th ACM & IEEE International conference on Embedded software. ACM, 2006: 171-177.Garc á M árquez, Gabriel. Love in the Time of Cholera. Translated by Edith Grossman. London: Cape, 1988

[3] Nijim M, Manzanares A, Ruan X, et al. HYBUD: an energy-efficient architecture for hybrid parallel disk systems[C]//Computer Communications and Networks, 2009. ICCCN 2009. Proceedings of 18th Internatonal Conference on. IEEE, 2009: 1-6.Kossinets, Gueorgi, and Duncan J. Watts. "Origins of Homophily in an Evolving Social Network." American Journal of Sociology 115 (2009): 405-50. Accessed February 28, 2010. doi:10.1086/599247

[4] Xie T, Madathil D. SAIL: self-adaptive file reallocation on hybrid disk arrays[M]//High Performance Computing-HiPC 2008. Springer Berlin Heidelberg, 2008: 529-540.Pollan, Michael et al., The Omnivore's Dilemma: A Natural History of Four Meals.

New York: Penguin, 2006

[5] Dowdy L W, Foster D V. Comparative models of the file assignment problem[J]. ACM Computing Surveys (CSUR), 1982, 14(2): 287-313.Stolberg, Sheryl Gay, and Robert Pear. “Wary Centrists Posing Challenge in Health Care Vote.” New York Times, February 27, 2012. Accessed February 28, 2012. http://www.nytimes.com/2010/02/28/us/politics/28health.html

[6] Copeland G, Alexander W, Boughter E, et al. Data placement in Bubba[M]. ACM, 1988.Weinstein, Joshua I. “The Market in Plato’s Republic.” Classical Philology. 104 (2009): 439-58

[7] Wolf J. The placement optimization program: a practical solution to the disk file assignment problem[M]. ACM, 1989

[8] Wah B W. File placement on distributed computer systems[J]. IEEE Computer, 1984, 17(1): 23-32

[9] Madathil D K, Thota R B, Paul P, et al. A static data placement strategy towards perfect load-balancing for distributed storage clusters[C]//Parallel and Distributed Processing, 2008. IPDPS 2008. IEEE International Symposium on. IEEE, 2008: 1-8

[10] Lee L W, Scheuermann P, Vingralek R. File assignment in parallel I/O systems with minimal variance of service time[J]. Computers, IEEE Transactions on, 2000, 49(2): 127-140

[11] Xie T, Sun Y. A file assignment strategy independent of workload characteristic assumptions[J]. ACM Transactions on Storage (TOS), 2009, 5(3): 10

[12] Zhu Q, David F M, Devaraj C F, et al. Reducing energy consumption of disk storage using power-aware cache management[C]//Software, IEE Proceedings-. IEEE, 2004: 118-118

[13] Carrera E V, Pinheiro E, Bianchini R. Conserving disk energy in network servers[C]//Proceedings of the 17th annual international conference on Supercomputing. ACM, 2003: 86-97

AUTHORS



[1]***Xupeng Wang*** was born in Shandong Province, China, in 1986. He received the B.S degree in computer science from the Chengdu University of Technology in 2010, and M.S degree from the University of Electronic and Technology of China in 2012, where he is currently working toward the Ph.D. degree. His research interests include embedded system and computer vision.



[2]***Wei Jiang*** was born in Sichuan Province, China, in 1981. He received the B.S and Ph.D. degree in computer science from the University of Electronic Science and Technology of China in 2003 and 2009 respectively. His main research interests include embedded system and reliability computing.

He is currently an Associate Professor at the School of Information and Software Engineering, University of Electronic Science and Technology of China. From 2011 to 2012, he was a visiting scholar in Link ping University and Technical University of Denmark.



[3]***Hang Lei*** was born in Sichuan Province, China in 1960. He received the B.S degree from Sichuan University in 1982, and the M.S and Ph.D. degree from the University of Electronic Science and Technology of China in 1988 and 1997 respectively. His main research interests include embedded system, real-time computing and software reliability.

He is currently a professor at the School of Information He is currently a professor at the School of Information and Software Engineering, University of Electronic Science and Technology of China. He is also the vice president of the school and head of the embedded & real-time computing group.



[4]***Xia Zhang*** was born in Sichuan Province, China in 1987. She is a master student in the University of Electronic Science and Technology of China. She received the B.S degree in 2009 from Sichuan University. She is currently involved in the research on the project of National Science Funding of China. Her research interests include security and energy-aware distributed real-time system.