Computer Communication & Collaboration (Vol. 3, Issue 4, 2015)
ISSN 2292-1028(Print) 2292-1036(Online)
Submitted on Aug. 31, 2015
(DOIC: 2292-1036-2015-04-001-59)

HyBuM: Energy-Efficient Hybrid Mobile Storage Systems using Solid States and Buffer Disks

Mais Nijim (corresponding author), Ashraf yaseen

Faculty of Computer Science, Texas A&M University - Kingsville

700 University Blvd., Kingsville, TX, United States

E-mail: {mais.nijim},{ashraf.yaseen}@tamuk.edu

Waseem AlAqqad

Communication and Networking ENgineering, Prince Sultan University

Riyadh 12435, Saudi Arabia

E-mail: walaaqqad@psu.edu.sa

Abstract: High performance and energy-efficient storage systems are essential for mobile data-intensive applications such as remote surgery and mobile data center. Existing mobile storage systems consist of an array of independent small form factor hard disks connected to a host by a storage interface in a mobile computing environment. Although hard disks are cost-effective and can provide huge capacity and high-throughput, they have some intrinsic limitations such as long access latencies, high annual disk replacement rates, fragile physical characteristics, and energy-inefficiency. Compared with hard disk drives, solid-state disks (SSD) are much more energy-efficient, and can offer much faster access times. A major concern on current solid-state disk is its relatively higher price. In this paper, we developed hybrid energy efficient mobile disk architecture SBUD that integrates an array of solid-state disks with buffer disks and an array of mobile disks. The most recently used data will be cached in the solid state disks, the second most popular data will be stored in the buffer disks, and the least used data sets will be stored in the mobile disk array. Experimental results demonstratively show that SBUD provides significant energy saving for mobile storage systems such as laptops, mobile phones, and PDA compared non-hybrid architecture.

Keywords: Buffer Disks, Solid State Disks, Mobile Disks

1 Introduction

The technology has witnessed a propagation of mobile computing platforms. Examples of such platforms, which vary widely in terms of capability and functionality, include laptops (e.g., notebooks, tablets, etc.), pocket PCs, personal digital assistants (PDAs), cell phones, wireless single-board computers, sensor nodes, etc. Following the general trend in the consumer electronics market, the cost of these devices has been steadily decreasing while their capacity (i.e., processing, storage, communication) has been steadily increasing. However, the fact that they are typically powered by non-continuous energy sources imposes serious limitations to these devices' utility from the end user's point-of-view. Reducing energy consumption of large-scale computing platforms has become an increasingly hot research topic in the realm of high-performance computing. Green computing has recently been targeted by government agencies; efficiency requirements have been outlined by the U.S. Environmental Protection Agency (a.k.a., EPA) [1]. Large-scale parallel disk systems inevitably lead to a huge amount of energy due to scaling issues in data centers, which unsurprisingly consume power anywhere between 75 W/ft2 to 200 W/ft2. Such a high energy consumption rate is likely to continuously increase up to 200-300 W/ft2 in the near future [2]. Large-scale computing systems not only have a large economical impact on governments, companies, and research institutes, but also produce environmental impacts. Data from EPA indicates that generating 1 kWh of electricity in the United States results in an average of 1.55 pounds (lb) of carbon dioxide (CO2) emissions. With large-scale clusters requiring up to 40TWh of energy per year at a cost of over $4B, it is reasonable to conclude that energy-efficient clusters can have huge economical and environmental impacts [3].

Energy consumption in mobile computing has been an area of intense research spanning many fields such as mobile data centers. Mobile Data Centers are considered to be a substitute to conventional stationary data centers that are enclosed in buildings. They could be built on self-contained trucks, airplanes, UPS and satellite Internet Links [4]. Typical applications for mobile data centers include but not limited to live video broadcast [5], homeland security, and natural disaster recovery. In mobile data centers applications, high mobility and a fast large-volume data processing capability are effectively demanded. One way to save power is to spin down disks when they are sitting idle. However, spinning down disks is effective only if the disks can remain in standby for long time periods. Conventional wisdom of reducing disk traffic is to employ a buffer cache. The fundamental idea behind this research is motivated by buffer caches that allow data blocks to be accessed at memory speed if the blocks are residing in the caches. Read and write requests to data stored in the buffer caches do not need to issue disk operations. By eliminating unnecessary disk accesses, the buffer cache can play a major role in saving power by minimizing the number of disk spin ups and downs. In this study, we intend to seamlessly integrate parallel disk systems with flash drives, in which popular data blocks can be prefetched and cached to reduce disk spin up and down times.

Solid-state disk (SSD) is a form of non-volatile storage device that has gained popularity in the past few years. Data is stored in semiconductor memory that is approximately as fast as DRAM with the added advantage of no needing any refreshing to maintain the data. Besides, solid state disk has the non-volatility feature, holding data even when power is turned off. Solid state disks exhibit a number of advantages that make them ideal for mobile disks. They are physically robust with high vibration-tolerance and shock-resistance [5, 6]. They also consume much less energy than mechanical hard disks [7]. Moreover, they offer much fast read access times due to lack of moving. Although solid-state disk is as fast as memory with respect to reads, write performance of solid state disk is relatively low. Other drawbacks of solid-state drives include high cost and limited number of writes cycles [7]. As such, using solid state drives to build energy efficient parallel storage systems is not only highly expensive but also has negative impacts on data reliability. An ideal solution to this problem is to incorporate solid state drives into traditional mobile storage systems equipped with mobile hard drives.

In this paper, we developed a large and cost effective hybrid mobile disk architecture where an array of solid state disks is incorporated into an array buffer disks and an array of mobile disks. The ultimate goal of this research is to achieve high energy efficiency architecture for mobile storage systems. Several techniques proposed to save energy in mobile disk systems including adaptive predictive workloads [8], power aware cache management [9], and prefetching techniques [10][11].

However, the research on energy efficient mobile disks with solid state is still in its infancy. Therefore, we make an effort to reduce energy dissipation in hybrid mobile disk systems while maintaining high I/O performance.

The rest of the paper is organized as follows. Section 2 summarizes related work. Section 3 presents the hybrid disk architecture or SBUD. In section 4 we describe the power management algorithm in SBUD. Section 5 describes the energy consumption model to facilitate the development of energy efficient mobile disk systems. Section 6 describes disk request processing mechanisms in SBUD. In Section 7 we present the simulation results and the experimental results. Finally, section 8 concludes the paper.

2 Related Work

2.1 Energy Saving for Mobile Disks

Most of the previous research regarding conserving energy focuses on single mobile disk system [12] such as laptop and mobile devices to extend the battery life. Recently, several techniques proposed to conserve energy in storage systems include dynamic power management schemes [13], power aware cache management strategies [14], power aware perfecting schemes [15], software-directed power management techniques [16], redundancy techniques [16], multi-speed settings [17], selection of a timeout threshold which could be a fixed time period [18] or adaptively adjusted run time [19,20] where these schemes adaptively monitor the disk I/O operations without extending disk idle time which will limit their energy saving potential, another approach is customizing system or application software for saving mobile disk energy [21]. Our study and the above research are different in that we aim to improve energy efficiency for parallel disk systems with flash drives and hard disks.

2.2 I/O Buffer Management

Buffer management has been used to boost performance of parallel disk systems [22,23,24]. Previous studies showed that the data buffers significantly reduce the number of disk accesses in parallel disk systems [25]. More importantly, it is observed from the previous studies that traffic of small writes becomes a performance bottleneck of disk systems, especially when RAM sizes for data buffers are increased rapidly [25]. It is expected that small writes dominate energy dissipation in parallel disk systems that support data intensive applications like remote sensing applications.

2.3 Memory Caches

Main memory caches based on volatile memory have long been used to reduce disk traffic in order to improve response time and throughput. More recently, the researchers have explored the idea of using non-volatile memory to reduce write traffic [26]. Marsh et al. developed an architecture that used flash memory as a second level cache to conserve energy as well as to improve performance.

3 Hybrid Mobile Disk Architecture

Figure 1 depicts a hybrid architecture or (SBUD for short) which contains *n* solid state disks with size S GB, *m* buffer disks, *D* data disks, and an energy-aware solid state/buffer disk controller. Note that the values of *n, D* and *m*, which are configuration on the fly, are independent of each other. A RAM buffer with a size ranging from several megabytes to gigabytes is incorporated to further improve I/O performance in SBUD. The solid state disks/buffer disk controller coordinates multiple modules, including power management, data partitioning, disk request processing, and perfecting schemes.

The solid state disks perform as a non-volatile cache to boost up the I/O performance and to improve the energy efficiency by absorbing the disk traffic fluctuations. The solid state disks respond to read and write requests. A read miss on the solid state disks causes a hit at the buffer disks. To boost up the performance, the disk request will be prefetched from the buffer disk to the solid state disks before it is requested. Write requests are served by the solid state disks first. If the solid state disks are full, the write request will be directed to the buffer disks using the least recent used policy.

A prefetching scheme is designed to bring data into buffer disks or solid state disks before its use. Apart from the prefetching scheme, we developed a write strategy to energy-efficiently handle writes using solid state disk and buffer disks. The write I/O load imposed on buffer disks is well balanced by equally distributed write request to all the active buffer disks to make the utilization of all the buffer disks identical. To improve I/O performance of buffer disks handling write requests, we chose to use a log file system that allows data to be written sequentially buffered in buffer disks to minimize disk seek times and rotational delays. We developed a buffer disk manager that is responsible for the following activities. First, the disk manager aims to minimize the number of active buffer disks while maintaining reasonably quick response time for disk requests. Second, the manager must deal with the read and write requests redirected from the solid state disk in an energy-efficient way. Third, the manager has to energy-efficiently move data among the solid state disk, buffer disks, and data disks.

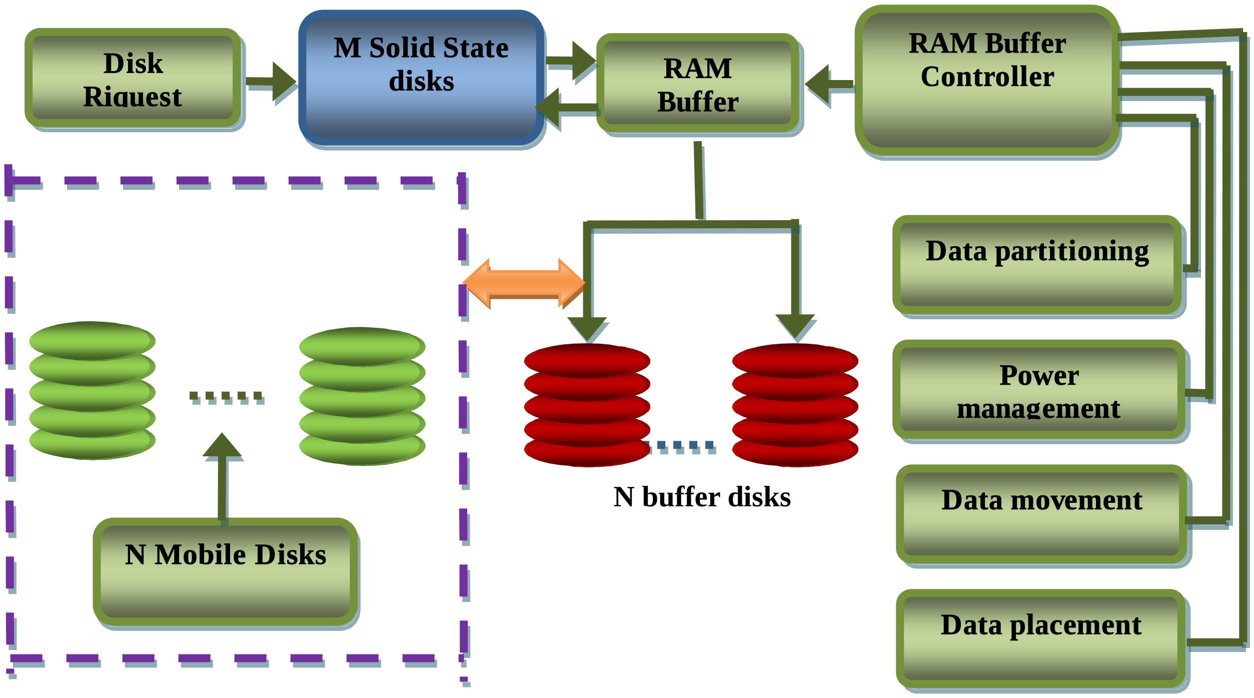

Figure 1. The Architecture for large capacity and cost effect Hybrid Mobile Disk Array

We developed a buffer disk manager that is responsible for the following activities. First, the disk manager aims to minimize the number of active buffer disks while maintaining reasonably quick response time for disk requests. Second, the manager must deal with the read and write requests redirected from the solid state disk in an energy efficient way. Third, the manager has to energy-efficiently move data among the solid state disk, buffer disks, and data disks. Third, the manager has to energy-efficiently move data among the solid state disk, buffer disks, and data disks.

4 Power Management Algorithm

The crucial goal of the proposed mechanism is to conserve energy for mobile disk systems as much as possible without scarifying the performance of the system. To reduce energy consumption, modern disks use multiple power modes that include active, idle, standby, and shut down modes. In active mode, the platters are spinning and the head is seeking or the head is actively reading or writing a sector. In idle mode, a disk is spinning at its full speed but no disk activity is taking place. Therefore, staying in idle mode when there is no disk request provides the best possible access time since the disk can immediately service request, but it consumes the most energy. To simplify discussion, we don't differentiate between the active mode and idle mode since in both modes the disk is operating at its full power. In the standby mode, the disk consumes less energy, but in order to service a disk request, the disk will incur significant energy and time overhead to spin up.

To improve I/O performance of buffer disks handing write requests, we chose to use a log file system that allows data to be written sequentially buffered in buffer disks to minimize disk seek times and rotational delays. We developed a buffer disk manager that is responsible for the following activities. First, the disk manager aims to minimize the number of active buffer disks while maintaining reasonably quick response time for disk requests. Second, the manager must deal with the read and write requests redirected from the solid state disk in an energy efficient way. Third, the manager has to energy-efficiently move data among the solid state disk, buffer disks, and data disks. Third, the manager has to energy-efficiently move data among the solid state disk, buffer disks, and data disks. When the dirty data is flushed to the buffer disk, the buffer disk controller will be always trying to keep as more data disks in sleeping mode. Once a data disk is wakening up, it will be keeping busy for a while because a large trunk of data coming from RAM buffer directly or from buffer disks will be written to it.

In order to fully utilize the gap of energy consumption rate under different mode in the SBUD architecture, the solid state disks and the buffer disk controller keeps as more data disks as possible in sleeping mode. For the write request, the data will be written to one of the solid state disks, if the corresponding disk of the solid state disk partition is in active mode, the data block will be written to the disk. Otherwise, the data block will be kept in the solid state disk and the dirty block will be flushed to the buffer disk. As a result, the data disk will stay in sleeping mode, which will help in saving energy. The controller will set up a time threshold for the weakened up data disks. If the idle time exceeded the threshold, the data disk will be turned back to sleep mode to save power. By using this strategy, we can conserve energy without scarifying the performance of the mobile disk systems.

To reduce energy consumption for mobile disk system, modern disks use multiple power modes that includes active, idle, and standby mode. The basic power model for the mobile disk system is the summation of all power states multiplied by the time that each power state was active. The states used are start-up, idle, and read/write/seek. Read, write, and seek are put together because they shared the same power consumption.

Let *Ti* be the time required to enter and exit the inactive state. The power consumption of a disk when entering and exiting the inactive state is *Pi*. Therefore, energy *Ei* consumed by the disk when it enters and exits the inactive state is expressed as T_i P_i. Let *Tactive* be the time interval when the mobile disk is in the active state. The power consumption rate of the mobile disk when it is in active state is denoted by P_{active}. Thus, the energy consumption of the disk when it is in the active state can be expressed as $E_{active} = P_{active}.T_{active}$. Let T_{idle} be the time interval when the mobile disk is in idle state, the power consumption rate of the mobile disk when it is in idle state is represented by $E_{idle} = P_{idle}.T_{idle}$. Thus, the total energy consumed by the mobile disks is calculates as the following:

$$\begin{aligned} E_{total} &= E_{solid} + E_{tr} + E_{active} + E_{idle} \\ &= E_{solid} + P_{tr}.T_{tr} + P_{active}.T_{active} + P_{idle}.T_{idle} \end{aligned} \qquad (1)$$

where E_{solid} is the energy consumed by the solid state disks. We use Samsung SSDs, which consume 60% less power than the hard disk drives. The E_{solid} is 1.3 Watt when it is in the active read state, 2.6 Watt when it is in the active write state, and 0.7 Watt when it is idle.

Let T_{ai} and T_{ia} denote the times a disk spends in entering and exiting the inactive state, and let P_{ai} and P_{ia} be the power consumption rates when the disk enters the inactive and active state. N_{ai} and N_{ia} are the number of times the disk enters and exits the inactive state. Thus, the transition time $T_{transition}$ is computed as follows:

$$T_{transition} = N_{ai}T_{ai} + N_{ia}T_{ia} \qquad (2)$$

The power transmission is computed by:

$$P_{transition} = P_{ai} + P_{ia} \qquad (3)$$

$$E_{tr} = (T_{ai} / (T_{ai} + T_{ia}))P_{ai} + (T_{ia} / (T_{ai} + T_{ia}))P_{ia} \quad (4)$$

The time interval T_{active} when the mobile disk is in active state is the sum of serving times of disk requests submitted to the mobile disk array.

$$T_{active} = \sum_{i=1}^{n} T_{service}(i) \qquad (5)$$

where *n* is the total number of requests submitted to the system, and $T_{service}(i)$ is the serving time of the *ith* disk request and is calculated by

$$T_{service}(i) = T_{seek}(i) + T_{rotation}(i) + T_{trans}(i) \qquad (6)$$

where T_{seek} is the amount of time spent seeking the desired cylinder, $T_{rotation}$ is the rotational delay and T_{trans} is the amount of time spent actually reading from or writing to disk.
Now the energy saved by our management policy is quantified as,

$$E_{save} = (T_{active} + T_{idle} + T_{tr})P_{active} - E_{total}$$

$$\begin{aligned} &= (T_{active} + T_{idle} + T_{tr})P_{active} - E_{solid} + (T_{active}P_{active} + T_{idle}P_{idle} + T_{tr}P_{tr}) \\ &= E_{solid} + (P_{active} - P_{idle})T_{idle}(P_{active} - P_{tr})T_{tr} \end{aligned} \qquad (7)$$

Where E_{solid} is the energy consumed in the solid state disk.

The transition power consumption is not considered in this study, for this model it is important to decide the power consumption for each state and the power consumption in solid state disk. These values can be obtained based on physical tests.

5 The HyBum Algorithm

Handling read requests is kind of simple and straightforward. Read requests first arrives to the solid state disk. If the data block is resided in the SSD, then the data is immediately sent back to the requester. If the requested data is not in the SSD, a copy of the data will be written to the SSD assuming that this data block is going to be used frequently. The data will be retrieved from the corresponding data disk if it is in active mode, otherwise, the read requests will be clustered together in the SSD waiting for the corresponding data disk to be in active mode. When the SSD is full, dirty blocks will be flushed to the buffer disk. On the other hand, the buffer disk clusters the miss read requests together. By clustering the read requests, the data disks will be able to stay in the sleep mode for longer periods of time.

Modern parallel disk system usually implements write-back caching. In this case, unlike read, a write request is completed once the data is written to the SSD. If the corresponding disk is in active state, the data block will be written directly to the data disk. Otherwise, the write request will be kept in SSD and dirty data are flushed to buffer disk according to a cache replacement policy. In this study we use a least recent used policy (LRU).

Once the dirty data are flushed to the RAM buffer as shown in section 3, the buffer disk controller responsibility is in two fold. First, the controller will check the size of the write requests. The write requests are divided into small write and large write requests. If the request is large write for example 10MB or more, the request will be sent directly to the corresponding data disk. Otherwise, the controller will send the write request to the RAM buffer that buffers small write requests together and form a log of write requests that will be written to the data disk later. Our focus for this study is on small write requests. Second, the controller will test the state of all buffer disks. If the buffer disk is not busy with writing a previous log, the data will be written to the buffer disk to ensure that a reliable copy resides on one of the buffer disks. Operations which could write the same block data into different buffer disks is forbidden if one legal copy of this block still exists in any buffer disk.

6 Experimental Results

In this section, energy efficiency alongside with the performance of HyBum architecture for mobile storage system was evaluated in the Matlab environment and compared with the case where SBUD was not applied to the disk storage system. In the evaluation and comparison, the following assumptions were made:

1) One half of the total capacity of each SSD disk is dedicated to the read and the other is dedicated to the write operations.
2) There can be three types of data requests: read, write and ‘No Data’ request.
3) No two types of data requests occur simultaneously.
4) If any HDD does not have any activity for 10 seconds, it goes to the standby mode.
5) If any HDD passes 10 seconds in standby mode, it will wake up.
6) Total time is quantitatively equal to total number of data requests and the rate of data request is one data unit per second.
7) All data unit size for reading and small writing is of 1 Mega Byte each.
8) Each SSD disk has only one corresponding HDD.

9) DM-PAS is applied for data prefetching algorithm.
10) The prefetched data are distributed among the SSD disks in a rotating manner.
11) When writing small data, if the space allocated for writing is full, 2 data units are flashed to the drum memory.
12) If any SSD disk is full, 2 data units are flashed to the drum memory.
13) Probability of ‘No Data’ request and probability of writing or reading request is same, that is 50%.
14) Both writing and reading request have the same probability, which is 25% of the total data requests.
15) Of the total writing requests, large and small writing request have the same probability.

Table 1 Simulation Parameters

Parameter	Value
Power consumption of HDD when active	1186 mW
Power consumption of HDD when standby	231 mW
Power consumption during transition period	891 mW
Power consumption of SSD	730 mW

6.1 Impact of the number of data requests

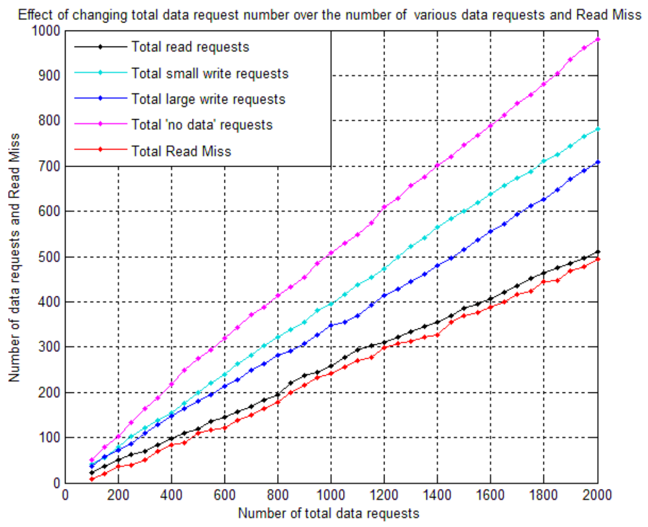

Figure 2. Effect of changing the total number of data requests

In this experiment, the total number of data requests was changed from 100 to 2000 by a step of 50. From figure 2 it can be observed that, as the number of total data requests increases, the number of Read Miss also increases. It is because, as the number of capacity of each SSD disk and the number of SSD disks are limited, with the increase of total data requests, more requests for the data from the HDDs are made. As the probability of making ‘No Data’ request is 50%, the line representing the number of ‘No Data’ requests is almost diagonal. However, although the number of Read Miss increases, total amount of energy saving also increases with the increment of total data request number, because the system would have to keep the HDDs active for a longer period of time if the SBUD strategy were not implemented.

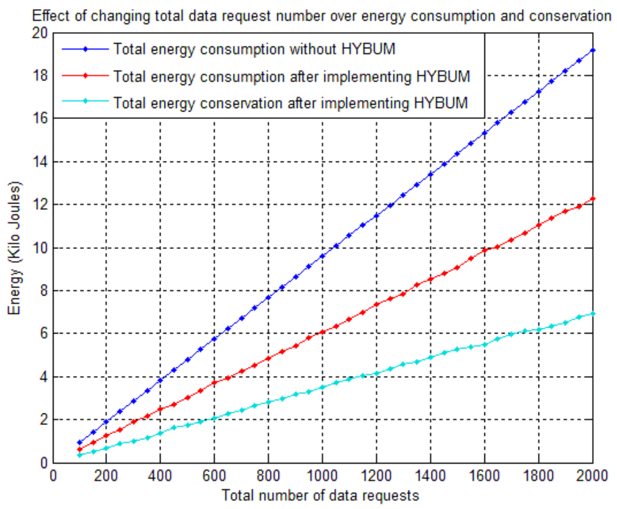

Figure 3. Effect of changing the total number of data requests

6.2 Impact of the SSD size

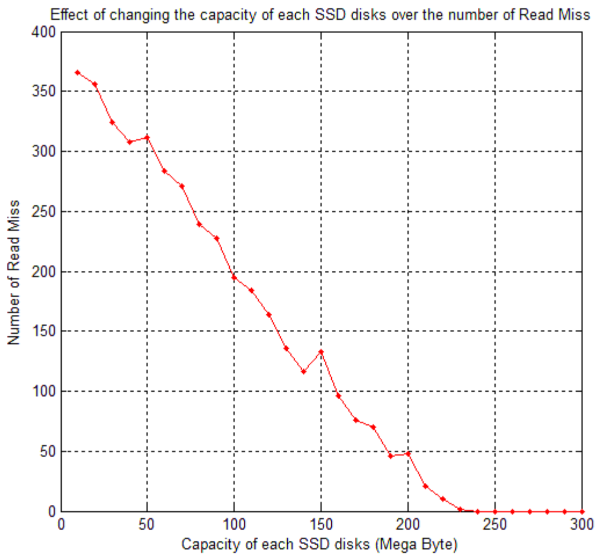

Figure 4. Effect of changing the capacity of each SSD disks.

In this experiment, the capacity of each SSD disk was incremented from 10 Mega Byte to 300 Mega Byte by a 10 Mega Byte step. As anybody can easily speculate, the number of Read Miss decreases as the capacity of each SSD disk increases because, with more capacity the SSD disks can accommodate more data requested by the users. Figure 4 shows the effect of changing the capacity of each SSD disks over the number of Read Miss.

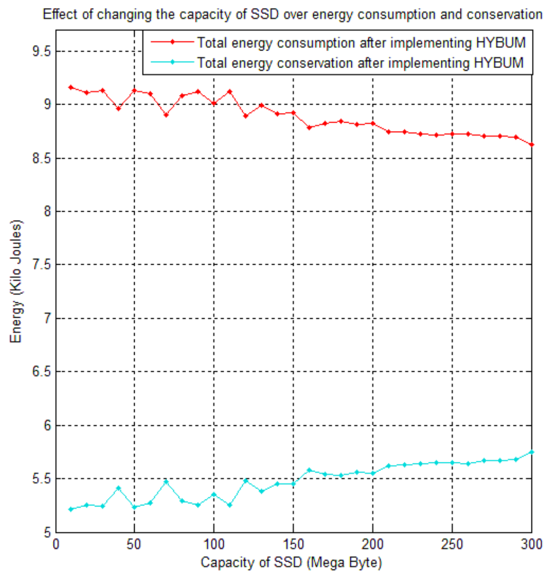

Figure 5. Effect of changing the capacity of SSD

From figure 5 we can see that, the energy conservation increases and energy consumption decreases as the capacity of each SSD disks increases. With more capacity of SSDs, the system can keep the HDDs in standby mode for more time.

6.3 Impact of the number of SSD

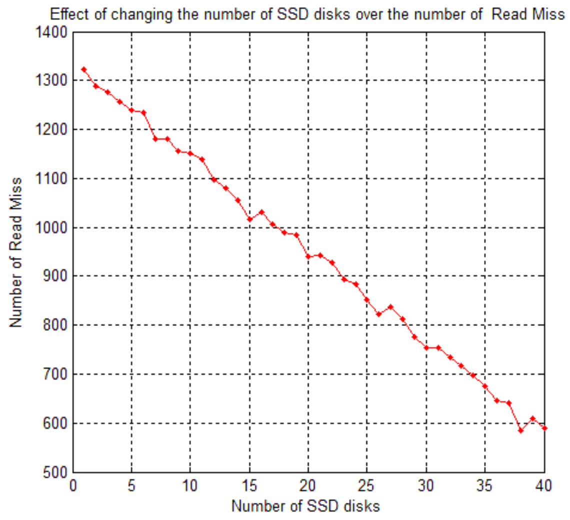

Figure 6. Effect of changing the total number of SSD

From figure 6 we can see that, the number of Read Miss decreases with the increment of the capacity of each SSD disk as the system can accommodate more data in the SSDs requested by the users.

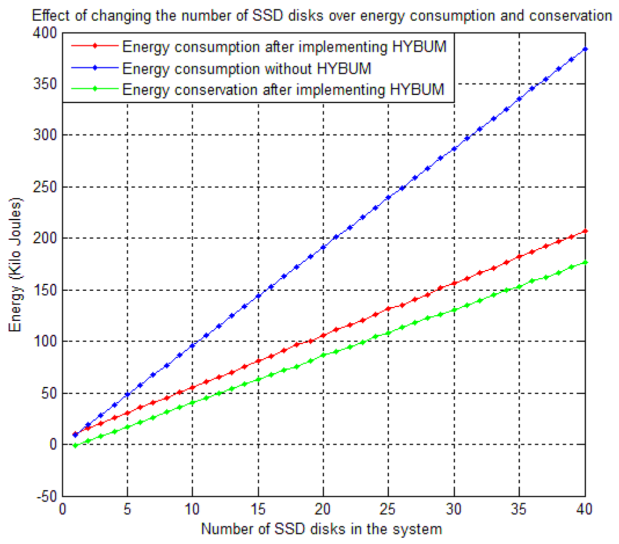

Figure7. Effect of changing the total number of SSD

From figure 7, we can see that energy conservation increases significantly with the increment of SSD disk number. With more number of SSD disks and consequently with more HDDs, the SBUD architecture can keep more HDDs in standby status which would not be possible for the system without the implementation of the strategy.

In this section we present the results of our experimental study. In this section, we compare the SBUD with two strategies. The first strategy is called Solid State which measures the energy consumption when solid state disks are used. The second strategy is called MUD which measures the energy consumption when using mobile disks only. We have developed a simulator which models *n* mobile disks arrays which forms a one mobile disk array. We utilize the parameters of LaCie 301831 to emulate the mobile storage system. The simulation parameter is described in Table 2.

Table 2. Simulation Parameters

Description	**Value**
CPU	Intel Pentium 4 , 2.8 GHZ
Capacity	320 GB
Disk Model	Segate Cheetah 15K.4 (ST3146854LW)
Average seek time	12 ms
OS	Windows XP SP2
Interface	High speed USB 2.0
Transfer rate	420 Mbit/sec
Solid state disk model	Samsung 470 SATA solid state drive
Solid state size	64GB

6.4 Impact of the arrival rate

This experiment is focused on comparing the SBUD strategy against the two baseline architectures described above. Let us study the impacts of miss ratio on the normalized energy consumption ranging from 0 to 1. To achieve this goal, we gradually increased the miss ratio from 75% to 100%.

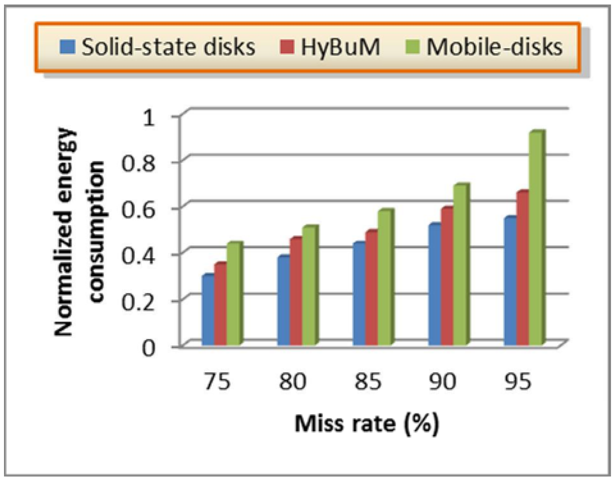

Figure 8. Energy consumption versus miss ratio

Figure 8 plots empirical results when there are five mobile disks in the mobile data center where the average size of disk requests is 300 MB. As the miss rate is increased, the energy consumption of the three strategies also increased. The solid state strategy consumes less energy than the other two alternatives strategy. Different from the mobile disk, the solid state disk is made of chips without any mechanical component, such as disk platters, which consumes a huge amount of energy. Moreover, the solid state disk does not need power to maintain its data. Thus, the energy consumption of the solid state disks is almost negligible compared with the mobile disk.

6.5 Impact of the Data Size

The focus of this experiment is on comparing the three strategies when the data size varies from 400KB to 800KB to examine the performance impact of the data size on SBUD. When the data size increases, the normalized energy consumption will increase as well.

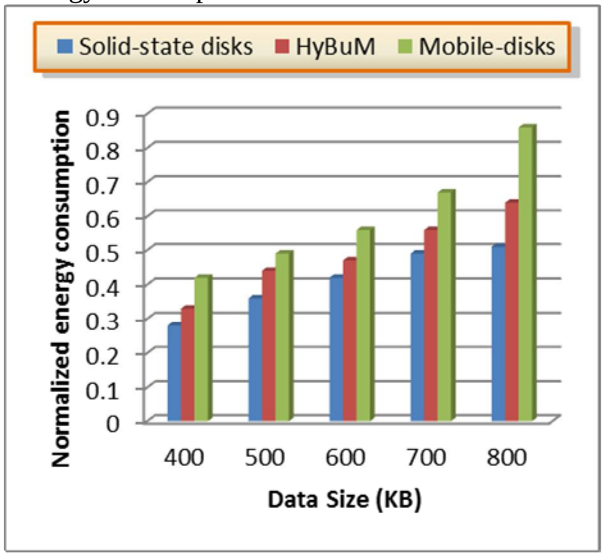

Figure 9. Energy consumption versus Data Size

References

[1] E.Jones, (2006-10-23). EPA Announces New Computer Efficiency *Requirements*.U.S. A.Retrieved on 2007-10-02.

[2] M. Kallahalla and P. J. Varman, "Improving parallel-disk buffer management using randomized writeback,"*Proc. Int'l Conf. Parallel Processing,* pp. 270-277, Aug. 1998.

[3] E. Carrera, E. Pinheiro, and R. Bianchini. "Conserving Disk Energy in Network Servers," *Proc. Int'l Conf. Supercomp.*, pp.86-97, 2003.

[4] Mobile Emergency Datacenter, North Am. Access Technologies,http://www.naat.com/Disaster%20Recovermobiledatacenter htm, 2006.

[5] E. Carrera, E. Pinheiro, and R. Bianchini, "Conserving Disk Energy in Network Servers," *Proc. 17th Ann. Int'l Conf. Supercomputing,* pp. 86-97, 2003.

[6] L.P. Chang and T.W. Kuo, "An adaptive striping architecture for flash-memory storage," *Proc. IEEE Real- Time and Embedded Technology Systems of Embedded systems and Applications Symp.*, pp. 187-196, 2002.

[7] T. Kgil, D. Roberts, and T. Mudge, "Improving NAND Flash Based Disk Caches," *Proc 35th Int'l Symposium on Computer Architecture (ISCA)*, pp. 327-338, 2008.

[8] Rybczynski, J.P., Long, D.D.E., Amer, A.," Adapting Predictions and workloads for power management" 14th *IEEE International Symposium on Modeling Analysis and Simulation of Computer and Telecommunication Systems*, pp. 3-12, 2006.

[9] S. Gurumurthi, A. Sivasubramaniam, M. Kandemir, and H. Fanke, "DRPM: Dynamic Speed Control for Power Management in Server Class Disks," *Proc. Int'l Symp. of Computer Arch.*, pp. 169-179, June 2003.

[10] Feng Chen, Song Jiang, Xiaodong Zhang," SmartSaver:Turning Flash Drive into a Disk Energy Saver for Mobile Computers" *International Symposium on Low Power Electronics and Design*, pp. 412-417, 2006.

[11] Jaewoo Kim; Ahron Yang; Minseok Song;" Exploiting flash memory for reducing disk power consumption in portable media players" , *IEEE Transactions on Consumer Electronics*, pp. 1997-2004, 2009.

[12] A. Roth, A. Manzanares, K. Bellam, M. Nijim, and X. Qin, "Energy Conservation for Real-Time Disk Systems with I/O Burstiness," *roc.IEEE Int'l Workshop Next Generation Autonomous Storage and High Performance Computing*, St. Thomas, Virgin Islands, Aug. 2008.

[13] F. Douglis, P.Krishnan, and B. Marsh, "Thwarting the Power-Hunger Disk," *Proc. Winter USENIX Conf.*, pp.292-306, 1994.

[14] Q. Zhu, F.M David, C.F. Devaaraj, Z. Li, Y.Zhou, and P. Cao, "Reducing Energy Consumption Of Disk Storage Using Power Aware Cache Management," *Proc. High Performance Computer Framework*, 2004.

[15] S.W. Son and M. Kandemir, "Energy Aware data perfecting for multispeed disks,"Proc. *ACM International Conference on Computing Frontiers*, Ischia, Italy, May 2006.

[16] S.W. Son, M. Kandemir, and A. Choudhary, "Software-directed disk power management for scientific applications," *Proc. Int'l Symp. Parallel and Distr. Processing*, April 2005.

[17] S. Gurumurthi, A. Sivasubramaniam, M. Kandemir, and H. Fanke, “DRPM: Dynamic Speed Control for Power Management in Server Class Disks,” *Proc. Int’l Symp. of Computer Arch.*, pp. 169-179, June 2003.

[18] F. Douglis, P. Krishnan, and B. Marsh. “Thwarting the power-hungry disk”. *In Proc. of USENIX'94*, Jan. 1994.

[19] F. Douglis, P. Krishnan, and B. Bershad. “Adaptive disk spin-downi policies for mobile computers”. In *Computing Systems*, volume 8(4), pages 381-413, 1995.

[20] D. P. Helmbold, D. D. E. Long, and B. Sherrod, “A dynamic disk spin-down technique for mobile computing”, *In Proc. of the 2nd Annual International Conference on Mobile Computing and Networking*, 1996.

[21] A. E. Papathanasiou and M. L. Scott, “Energy efficient prefetching and caching”. In Proc. of USENIX'04, 2004.

[22] A. Manzanares, K. Bellam, and X. Qin, “A Prefetching Scheme for Energy Conservation in Parallel Disk Systems,” *Proc. NSF Next Generation Software Program Workshop,* April 2008.

[23] X.-J. Ruan, A. Manzanares, K. Bellam, X. Qin, “DARAW: A New WriteBuffer to Improve Parallel I/O Energy-Efficiency,” *Proc. the 24th Annual ACM Symposium on Applied Computing*, March 2009.

[24] J.-H Kim, S.-W. Eom, S.H. Noh, and Y.-H. Won, Stripping and buffer caching for software RAID file systems in workstation clusters,” *Proc. 19th IEEE Int’l Conf. Distributed Computing Systems*, pp. 544-551, 1999.

[25] Y. Hu and Q. Yang, “DCD-Disk Caching Disk: A New Approach for Boosting I/O Performance,” *Proc. Int’l Symp. Computer Arch.*, 1996.

[26] B.Marsh, F.Douglis, and P. Krishnan,” Flash Memory File Cashing for Mobile Computers” *Proc. the 27th Annual Hawaii International Conference on system sciences*, 1994.