Energy-Efficient File Placement Techniques for Heterogeneous Mobile Storage Systems

Young-Jin Kim
School of Computer Science & Engineering
Seoul National University
Seoul 151-742, KOREA
youngjk@davinci.snu.ac.kr

Kwon-Taek Kwon
School of Computer Science & Engineering
Seoul National University
Seoul 151-742, KOREA
kkt@davinci.snu.ac.kr

Jihong Kim
School of Computer Science & Engineering
Seoul National University
Seoul 151-742, KOREA
jihong@davinci.snu.ac.kr

ABSTRACT

While hard disk drives are the most common secondary storage devices, their high power consumption and low shock-resistance limit them as an ideal mobile storage solution. On the other hand, flash memory devices overcome the main problems of hard disk drives, but they are still more expensive in the cost per bit over hard disk drives and can only support a limited number of erase cycles. In this paper, we show that combining the merits of a hard disk and a flash memory device can produce an energy-efficient secondary storage solution for mobile platforms. We propose an energy-efficient file placement technique for such heterogeneous storage systems. The proposed technique adapts an existing data concentration technique by separating read and write I/O requests. Experimental results show that the proposed technique reduces the energy consumption by up to 74.5% when the combination of a 1.8″ disk and a flash memory is used instead of a single 2.5″ disk, at the cost of small increase in the average response time.

Categories and Subject Descriptors

D.4.2 [**OPERATING SYSTEMS**]: Storage Management – *secondary storage.*

General Terms

Experimentation, measurement

Keywords

Energy conservation, heterogeneous mobile storage systems, file placement, separating I/O operations.

1. INTRODUCTION

The ownership and use of mobile computing systems such as PDAs, PMPs, MP3 players and video recorders are increasing annually and fueling the demand for mobile storage devices, which now provide data capacity and performance comparable to server or desktop PC storage. Hard disk drives with a small form-factor (2.5″ or less), which target mobile storage systems, are under continual development. At present, the demand for server and desktop PC disks is higher than that for smaller disks, but the two markets may soon become comparable [1].

Hard disks are significant power consumers in mobile computers and take several seconds to spin up and down. Disks also have low shock resistance due to their mechanical parts. Flash memory is semiconductor-based and does not have these drawbacks. In particular, it has a much lower power consumption. Recently developed NAND flash memory can also access data faster than some mobile hard disks [2]. But flash memory is more expensive than disks and requires blocks to be erased before data can be overwritten on them. Moreover, the lifetime of the memory is limited to about one hundred thousand erasure cycles. Nevertheless, Samsung has adopted this solution on their laptop computers, but the cost of the flash memory device (in the form of a flash disk) amounts to $30 per gigabyte [3]. Although the high cost and capacity limitations currently restrict wide adoption of this solution [2], technology improvements will reduce the price of flash memory devices, making this solution become increasingly feasible.

The different characteristics of flash memory and HDDs have motivated a lot of research on energy-efficient mobile storage systems which combine disks and flash memory. Many researchers have proposed the use of flash memory as a non-volatile cache [4, 5, 6, 7, 2, 8], storing data blocks which are likely to be accessed in the near future and thus allowing the disk to spin down for longer periods. However, if only a subset of the data on the disk is available on the flash memory then the disk may have to be woken up quite frequently due to cache misses or flushing. And frequent writing to the flash may seriously shorten its lifetime. Even so, developments in flash memory can be expected to make this use more attractive in future, and a heterogeneous solution may be expected to remain more cost-effective than the use of flash memory alone for some time to come.

In this paper, we show that a combination of a hard disk and a flash memory device can provide an energy-efficient secondary storage solution for mobile computing systems. We will replace a hard disk with another disk of smaller form factor and a flash memory device and we will investigate how data can be distributed between the devices to reduce energy consumption by separating read and write I/O requests and redirecting them to one of the two devices. To evaluate this approach, we developed a trace-based simulator, which has

Table 1. Characteristics of a laptop disk, a small form-factor disk, and a NAND flash memory

Device		Hard disk		NAND flash
		Travelstar 80GN (2.5″)	MK4004 GAH (1.8″)	K9K1208U
Latency (512B)	Read	19.1(ms)	22.1(ms)	35.8(us)
	Write	19.1(ms)	22.1(ms)	288(us)
Throughput (MB/s)	Read	43.75	16.6	14.3
	Write	43.75	16.6	1.78
Power (mW)	Active	2300	1400	33
	Idle	950	400	0.13
	Standby	250	200	N/A

allowed us to profile an existing data concentration technique and then to extend it, using traces obtained while executing a real workload scenario on an evaluation board.

Our goal is to save a substantial amount of energy in mobile storage systems by using a smaller form-factor disk and a flash memory device instead of one larger disk and distributing I/O requests across the two devices. To avoid excessive wear of the flash memory device, and to mitigate its low write throughput, we migrate frequently-read data to the flash memory device and frequently-written data to the hard disk.

As a first step towards a solution, we will consider using two smaller form-factor disks instead of a large disk, which is similar to previous approaches [9, 10]. Then we look at replacing one of the smaller disks with a NAND flash memory device. We will assume that the NAND flash memory can be connected to a mobile system through an appropriate software flash translation layer (FTL), and that the NAND flash has the same capacity as a small form-factor hard disk. We believe that this is the first attempt to combine a hard disk and a flash memory device at the level of secondary storage to reduce energy consumption. It is essential to consider file placement and I/O redirection to maximize the energy saving. We will leave the design of an appropriate FTL for future work.

The rest of this paper is organized as follows: Section 2 describes our motivation in terms of power and reliability in mobile storage systems and Section 3 explains energy-efficient file placement techniques that exploit multiple homogeneous or heterogeneous devices, including an existing method and our technique. Section 4 describes our simulator and presents simulation results. Section 5 describes related work and Section 6 concludes the paper.

2. MOTIVATION: POWER AND RELIABILITY

Table 1 shows the latency, throughput and power parameters of two different form-factor hard disks and those of NAND flash memory [11, 12, 13]. For the disks, latency represents the sum of the average seek time and the average rotation delay, while the write latency of the NAND flash memory includes both the write delay and the erasure delay. Since the NAND flash memory exhibits much lower power consumption and latency than either of the disks, the potential benefit of replacing a disk with a NAND flash device is clear.

Let us assume that we are using a 2.5″ hard disk (the Travelstar 80GN in Table 1) on a laptop computer for ten minutes. And suppose that the disk is processing I/O requests for 15% of the total execution time and is idle for the remaining 85%. For the moment we will assume that all the I/O operations are reads. From the parameters in Table 1, we see that the total energy consumption over ten minutes will be 691J. If we replace the 2.5″ disk with two 1.8″ disks (MK4004GAHs in Table 1) and assume that the same workload is shared equally between them, each disk will stay in the active state for 8.68% of the total execution time, since the response time of a 1.8″ disk is 15.7% slower than that of a 2.5″ disk. The total energy consumption of the two 1.8″ disks will be 584J, which represents an energy saving of 15.5%.

Now we replace one of the two 1.8″ hard disks with a NAND flash memory device of equal capacity. The figures in Table 1 indicate that the NAND flash has a read latency which is 533 times less than that of the 2.5″ disk. Therefore we can expect the 1.8″ disk and the flash memory to stay in the active state for 8.68% and 0.014% of the total execution time, respectively. The total energy consumption is reduced to 292J, saving 57.7% of the energy when compared with a single 2.5″ disk and 50.0% when compared with two 1.8″ disks.

But what happens if 50% of the I/O accesses are writes? (This access pattern seems to occur often in mobile computing environments [14].) If we distribute the I/O uniformly across the two devices the energy saving should be comparable to the figures given above. But we need to make sure that the number of erasure cycles within the flash memory does not become excessive. If some blocks would be overly erased before write operations and reach the limit of erasure cycles, the overall lifetime of the flash memory device could be reduced fast and reliability of the flash storage system could be impaired.

A simple heuristic would be to place all the data that is accessed by read operations on the flash memory device, and the remaining data on the 1.8″ hard disk. This placement would save a substantial amount of the energy consumption while a longer lifetime for the flash memory device is expected. However, in practice, we cannot know in advance whether data should be placed on the flash memory device or the hard disk. By monitoring I/O operations as they occur and migrating frequently-read data to the flash device, we can achieve the benefit of the perfect pre-allocation.

3. ENERGY-EFFICIENT FILE PLACEMENT

We will now review an existing method of skewing frequently-accessed data, called popular data concentration (PDC). We will go on to describe an extended technique, based on PDC, which skews the load according to the pattern of I/O requests (i.e. read or write) on to one of two heterogeneous devices (which may be a hard disk or a flash memory device), thus allowing the other device to have more idle time so as to conserve energy.

3.1 Popular Data Concentration

PDC was proposed by Pinheiro *et al.* [15] to deal with the highly skewed file access frequencies exhibited by the workloads of some network servers. In general, the frequency of file accesses by a web server has been shown to conform to a Zipf distribution with a high coefficient. Zipf's law predicts that the frequency of access, or popularity, τ of a file is inversely proportional to a power of its rank *r,* which we can simply write $\tau = 1/r^{\alpha}$. Workloads with a high value of α are said to exhibit skewed popularity. The idea of PDC is to concentrate the most popular (i.e. most frequently accessed) disk data by migrating it to a subset of the disks, so that the other disks can be sent to a low-power mode to conserve energy. PDC redistributes data across the disk array according to its popularity, so that the first disk stores the most popular data, the second disk stores the next most popular data, and so on. The least popular data and data that is apparently never accessed will be stored on the last few disks. Files are migrated to the target disk until it is full or the expected load approaches its maximum bandwidth.

Using PDC can save significant energy consumption in storage systems consisting of multiple disks. However, if the frequency of file access varies significantly with time, PDC may cause a lot of file migrations, which will increase energy use, in particular by disturbing idle disks. This also happens when new files are created, because they will be stored on the disk with the least popular data, which has to be woken up. These drawbacks suggest that the applicability of PDC to real mobile system workloads may be limited. PDC has been shown to perform well with two-speed disks, but these are not generally available in the current market.

Despite these reservations, the basic idea of saving energy using I/O concentration through data migration can be applied to mobile storage systems with multiple small form-factor disks. In this paper, we adapt PDC to a mobile system with a pair of 1.8″ disks, and suggest that energy savings will also be available for a larger number of disks.

3.2 Pattern-based PDC

PDC concentrates popular data without considering whether I/O accesses are reads or writes, and so this scheme may require a large number of migrations until a target hard disk is full or reaches its maximum bandwidth. Consider a disk that is nominally inactive. Suppose that some of the files on this disk have recently been accessed by read operations till now, while other files on the same disk have been accessed by write operations. PDC will try to migrate all these files to an active disk without regard to the types of the I/O accesses, and this involves moving a lot of data. But if we classify the I/O into reads and writes and move only the data corresponding to one sort of access, we can reduce the amount of migration that is necessary. (If the total amount of data associated with reads is less than that associated with writes, then transferring the data that is being read will be more profitable.)

We therefore propose a scheme called PB-PDC (pattern-based PDC), which augments the PDC technique by moving frequently-accessed read and write data to separate sets of disks. Thus, while the disks containing data which is accessed in one way (read or write) are being accessed frequently, the disks storing data accessed in the other way can be sent to a low-power mode to conserve energy. Some disks may be located on the border between the read and write subsets. In this case priority is given to the movement of frequently-accessed write data.

We can apply PB-PDC to a heterogeneous storage system consisting of a hard disk and a flash memory device. Because a flash memory device has low write throughput and limited erasure cycles, PB-PDC moves the popular write data to the hard disk and the popular read data to the flash memory device. Since PB-PDC is designed for workloads with mixed access patterns we expect it to perform better than PDC in this context.

PB-PDC maintains read and write counters for each file and monitors the frequency with which each file is written and read. If the access to the file is read (write) the read (write) counter will be increased by one. The read/write counters together with file information such as file id, file size and disk (or flash) id are managed, as in PDC, by multi-queues arranged in descending order.

If two hard disks are used, then PB-PDC periodically migrates the popular read data to one disk and the popular write data to the other. If a hard disk and a flash memory device are employed then the popular read data will be moved to the flash memory and the write data to the hard disk. When there is a conflict between popular read and write data on the same device the write data will have priority.

At the start of each migration period, PB-PDC decreases the counter values by half. This is because the pattern of read or write accesses may vary over time and the more recent access frequencies should be given higher weight. Though PDC does not have this mechanism, we also applied it to PDC in simulation.

4. EXPERIMENTAL RESULTS

We will now describe our simulation environment, and then use it to evaluate file placement techniques on mobile systems with two homogeneous or heterogeneous devices in terms of energy and response time.

In these simulations we will assume that a perfect software FTL resides over the device driver layer, converting disk I/O accesses into flash I/O accesses and levelling the wear on blocks in the flash memory device. Our scheme is to run applications on a mobile system with a single hard disk, extract single-disk traces, and then use to simulate multi-disk I/O accesses.

4.1 Simulation Environment

We implemented a multi-device power and performance simulator that uses real workloads from an evaluation board. We also built a workload generator which simulates application patterns that are common on mobile systems and generates I/O traces for a disk.

Simulator. Our simulator models the static file placement and dynamic file migration which would occur when the workload represented by the traces runs on a mobile platform with multiple virtual devices. The static file placement is the distribution of files across the devices before any I/O occurs. During this process a multi-device allocation map is built which tells us how the files are distributed across the disks. Dynamic file migration is triggered whenever the energy-conserving techniques decide to move files between disks.

The important parts of the simulator, shown in Figure 1, are the four (shaded) components: 1) a file allocator which simulates initial file placement and dynamic file allocation on the storage devices using the I/O traces from the workload generator; 2) a file map manager which simulates the movement of files across

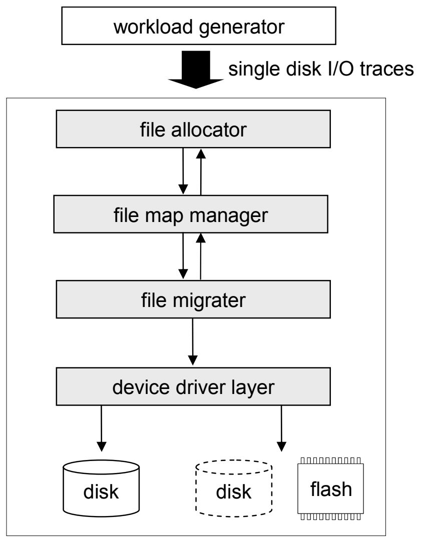

Figure 1. Simulator architecture

multiple devices and updates the multi-device allocation map; 3) a file migrater which monitors I/O accesses and invokes file migration using the algorithms we have described; 4) a device driver layer which simulates I/O scheduling operations and estimates each device's performance and energy use based on the pattern of I/O accesses. The simulator currently supports two devices and each device may be a hard disk or a flash memory. However, to obtain results relevant to this paper one device is always a hard disk.

The file migrater determines how the I/O accesses should be distributed across the devices and delivers this information to the file map manager, file allocator, and the device driver layer. The file allocator and file map manager perform file allocation and record file placement information. Following the directions of the file migrater, the device driver layer performs virtual I/O operations for each device and estimates its performance and energy consumption using a performance model, a power model, and a power control policy. For hard disks we use traditional threshold-based power management (TPM) as the power control policy, and we assume that a flash memory can be put into a low-power mode immediately there is idle time.

Workload generator. This module models workload patterns for specific mobile platforms and generates corresponding I/O traces. We assume that applications repeat predefined execution scenarios [6] and the types of applications are limited to file transfer, email, file search, and sleep. *File transfer* transmits and receives files to and from a network. *Email* reads mail messages from a mail box on a disk and transfers them to another mail box. *File search* reads files stored on disks and searches for specified strings. *Sleep* models periods during which no I/O requests are generated. To represent a simplified usage scenario of the target applications in a PDA, a typical mobile computing system, we set an execution order of them as follows: file transfer, email, file search, and sleep.

The workload generator creates I/O requests for a single 1.8″ disk (the MK4004GAH), executing applications repeatedly to follow the above scenario. I/O requests are delivered to the disk through the page cache, file system, and device driver within the Linux kernel 2.4. During these procedures, I/O requests which pass through the page cache and arrive at the disk are recorded on I/O traces. These traces contain the following information: a timestamp, a file identifier, the accessed block's offset within a file, the I/O data size, and the size of the accessed files.

Performance and energy model. The total request response time of each disk is estimated as the sum of a queue delay, a disk delay, and a service time per request.

The average request response time is obtained by dividing the total response time by *N*, which is the total number of requests in the I/O traces. I/O requests waiting in the request queue before processing may be merged into a large I/O request, at the device driver level, which causes the queue delay. But if requests stay long in the queue the response time will increase, so our simulator limits the maximum value of the queue delay to 60ms. That is, each request can stay in the request queue for at most 60ms, waiting for being concatenated with another request.

The disk delay is the delay which occurs due to a spin-up time when a disk, which has been in the inactive state, actually access the requested blocks. The spin-up time is the transition time either from the idle to active state or from the standby to active state.

The service time consists of a seek time, a transfer time, and a rotation delay. We used a simple seek time model based on a linear approximation obtained by measuring the delays recorded while varying the distance between two blocks that are accessed consecutively. The rotation delay is the average value (i.e. half a rotation). The total energy consumption of each disk is calculated as the sum of the energy used in each state and the energy consumed during transition periods.

For flash memory, we assume a similar performance and energy model, except that the cost of transitions to and from low-power mode is assumed to be negligible.

Simulation setup. Detailed power and performance parameters for the different form-factor disks are given in Table 2, and follow the values given elsewhere [11, 12, 13]. And the parameters of the flash memory device are given in Table 1. However, for our simulations the capacities of the 1.8″ disk and the flash memory are bounded to 400MB, and the capacity of the 2.5″ disk is bounded to 800MB.

We used a PXA255 embedded evaluation board with a 1.8″ Toshiba MK4004GAH disk running the Linux 2.4 kernel to execute the workload generator and extract I/O traces. To generate I/O traces of mobile workloads, the PDA scenario, which was described previously, was repeated for 81 minutes. The size of initial files was 408MB and the total data size of I/O requests except migrations was 315MB. And measurement showed that the request rate was about 10 requests/second and the proportion of write requests reached 5%.

We compared the energy-efficiency and performance of the five schemes listed in Table 3. SINGLE uses a single 2.5″ hard disk and has no file migration. PDC-I and PDC-II are applied to a mobile system with using two 1.8″ disks and a 1.8″ hard disk and a flash memory device, respectively. And the same storage composition is applied to PB-PDC-I and PB-PDC-II. In all the simulations we assumed that files are initially located randomly but uniformly across the device(s). For PDC-I, PDC-II, PB-

Table 2. Parameters of the hard disks

Device		Hard disk	
		2.5″	**1.8″**
Capacity (MB)		800	400
Rotation Speed (RPM)		4200	4200
Avg. rotation time (ms)		7.1	7.1
Avg. seek time (ms)		12	15
Power (W)	**Active**	2.3	1.4
	Idle	0.95	0.4
	Standby	0.25	0.12
Active to idle energy/time (J/s)		1.15/0.5	0.7/0.5
Idle to active energy/time (J/s)		1.15/0.5	0.7/0.5
Active to standby energy/time (J/s)		2.94/2.3	2.05/3.1
Standby to active energy/time (J/s)		5.00/1.6	1.84/1.7
DPM threshold: idle/standby (s/s)		1/3.379	1/5.979
Seek time model: $a_1\|b_i-b_j\| +a_0$ (a_1, a_0)		(2.9^{-10}, 0.0072)	(3.6^{-10}, 0.0090)

Table 3. The five file placement schemes used in simulation

Scheme name	**Description**
SINGLE	a 2.5″ disk and no file migration
PDC-I	two 1.8″ disks and PDC algorithm
PDC-II	a 1.8″ disk, a flash memory, and PDC algorithm
PB-PDC-I	two 1.8″ disks and PB-PDC algorithm
PB-PDC-II	a 1.8″ disk, a flash memory, and PB-PDC algorithm

PDC-I, and PB-PDC-II, we assume that file migration occurs every ten minutes. We compared the energy savings and average request response times of all the other schemes against SINGLE, which is a baseline scheme representing an upper bound on energy consumption and a reference point for average response time.

4.2 Simulation Results

Figure 2 shows the energy consumption of the five schemes for the PDA trace. As expected, SINGLE exhibits the highest energy consumption of 8075J. PDC-I achieves a 25.3% energy saving over SINGLE, consuming 6033J because the PDC algorithm moves frequently-accessed files to disk 0 (predefined as the active disk) and skews I/O operations on to it. This gives disk 1 more chances to enter its standby state, and Figure 3 shows that this indeed happens.

PDC-II consumes 4151J, which is 48% less than SINGLE and 31% less than PDC-I. This is mainly because of the low power consumption of flash memory, which is only 6J and is thus almost invisible in Figure 2.

PB-PDC-I saves as much energy consumption as PDC-I but Table 4 shows that its average response time is 38.3% better.

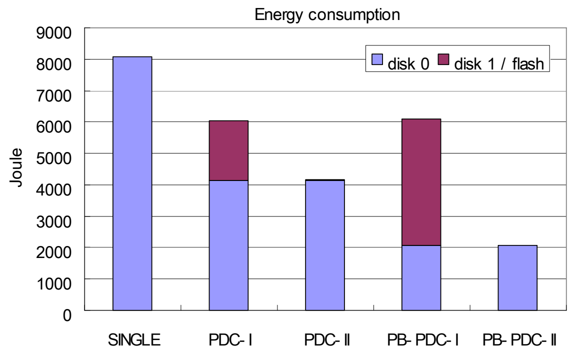

Figure 2. Energy consumption of the five file placement schemes

The bar graphs for disk 0 and disk 1 in Figure 2 for PDC-I almost seem to be reversed copies of those for PB-PDC-I. PDC-I moves all popular files regardless of the operation type to disk 0, while PB-PDC-I migrates popular write files to disk 0 and popular read files to disk 1; but the write ratio is only 5%, so disk 1 is now doing most of the work.

In comparison with PDC-I, PB-PDC-I substantially reduces the number of migrations between disks due to the efficient dynamic file placement achieved by separating I/O operations into reads and writes. Reduced migration makes PB-PDC-I considerably more energy-efficient than PDC-I. However, PB-PDC-I has distributed the data across the two disks by I/O type and there are less chances of putting a disk into the standby state, while PDC-I keeps almost one disk active and has more time to put the other disk into the standby state. This offsets the energy saving from decreased migrations and the total energy consumption of PB-PDC-I and PDC-I is remarkably similar, showing that the overhead of data migration cannot be neglected.

PB-PDC-II consumes only 2060J, an energy saving of 74.5% over SINGLE and the lowest figure of all the placement schemes. This is due to combining the use of flash memory with the separation of I/O types. By concentrating the much more frequent read operations on the flash memory, PB-PDC-II makes excellent use of its low read latency and saves 66% more energy than PB-PDC-I. PB-PDC-II saves 50% more energy than PDC-II, which uses the same devices. This energy saving can be directly attributed to the separation of I/O operations. In the simulation, PDC-II performed 6925 file migrations between the devices while PB-PDC-II only did 3325 file migrations.

We observe that the 50% more energy saved by PB-PDC-II than by PDC-II exceeds the differential of 31% between PDC-I and PDC-II. This indicates that data separation saves more energy than replacing a disk with a flash memory device in the heterogeneous storage system that we are modeling.

Furthermore, PB-PDC-II can achieve reduction of write/erase operations on the flash memory device at a high and coarse level by redirecting largely popular read accesses to it. We believe that if this coarse-level data placement would be combined with an appropriate FTL, which will assume the tasks of reclamation, wear-leveling, and finer-level placement of hot and cold data, then it would provide higher reliability of the overall heterogeneous storage system.

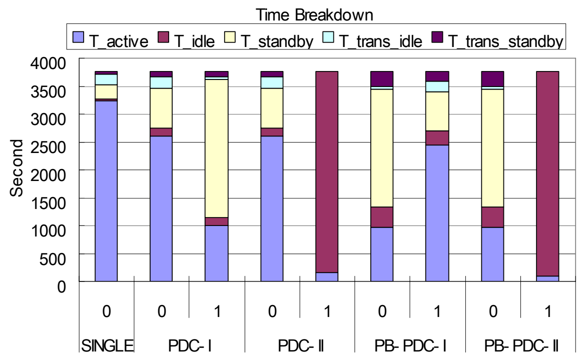

Figure 3. Time spent in different power modes by the five file placement schemes

In summary, the great energy savings of PDC-II over PDC-I (or PB-PDC-II over PB-PDC-I) show the advantage of introducing a flash memory device into mobile storage. And the substantial energy reduction of PB-PDC-II over PDC-II indicates that our data separating algorithm may overcome the drawback of the PDC algorithm, which was described in subsections 3.1 and 3.2.

A better understanding of the relative performance of the five schemes can be obtained by looking at the length of time that devices spend in each state, as shown in Figure 3. We can see that SINGLE seldom has enough idle time to enter a low-power mode and its active time is longer than that of any other scheme. Disk 1 of PDC-I has longer standby time than disk 0 of PB-PDC-I but disk 0 of PDC-I has a little more active time than disk 1 of PB-PDC-I. As previously described, the reason is because PB-PDC-I should access disk 0 for popular write operations and disk 1 for popular read operations by turns, which interrupts the deep sleep of each disk, and that PDC-I has more time to put the inactive disk (i.e. disk 1) into the standby state while keeping almost one disk active. The amount of time spent by PB-PDC-I in transition between the active and standby state reflects this effect.

The pattern of activity by disk 0 is the same for PDC-II as for PDC-I, but quite different for the other device. This is because the flash memory is effectively either active or idle, as shown in Table 1. The same characteristic differentiates PB-PDC-II from PB-PDC-I. However, with PDC-II the flash is active for twice as long as it is with PB-PDC-II. The extended activity of PDC-II can be attributed to the large number of data migrations, while the active time with PB-PDC-II is largely spent in reading popular data. In the case of the disk, accessing popular data, regardless of the type of I/O operation, is responsible for about 69.1% of the active time with PDC-II, while writing popular data and data migration take about 25.9% of the active time with PB-PDC-II. Although the write ratio in the trace is only 5%, the initial positions of the files may cause this pattern of activity by the disk with PB-PDC-II. This is because the number of data migrations can vary with the initial file positions. Separation of the data reduces the number of accesses to the disk, which results in 56% of the standby time of PB-PDC-II, which is almost three times longer than that of PDC-II.

Table 4 shows the average time taken to respond to a request, which can be taken as a measure of the performance of each scheme. Due to the inherent difference between the performance of 2.5″ and 1.8″ disks and the large number of file migrations, the average response time of PDC-I is 12 times slower than that of SINGLE. PDC-II shows a better response time because of the flash memory but still takes five times as larger as SINGLE due to the file migrations. PB-PDC-I requires fewer migrations than PDC-I and so its average response time is lower than that of PDC-I, but the performance is still poor due to the relatively slow 1.8″ disk and I/O congestion. PB-PDC-II has only 34.1ms more delay than SINGLE.

Table 4. Average request response time for each file placement scheme

Scheme	Avg. request response time (ms)
SINGLE	55.1
PDC-I	666.6
PDC-II	279.1
PB-PDC-I	411.1
PB-PDC-II	89.2

The 62% increase in response time of PB-PDC-II, by far the best of the heterogeneous schemes, might seem large but research indicates that interactive applications on mobile computers usually spend a lot of time waiting for the user's input. We believe that the response remains adequate to meet the quality of service generally required by users. And it would be possible to improve the overall response time by eliminating the request queue for the flash memory device, because we found that most requests to the flash memory were serviced without being merged, and the 60ms delay spent waiting for further requests was largely pointless.

5. RELATED WORK

There has been some research on saving energy by replacing a high-power, and high-performance hard disk with multiple disks of lower power and performance. Carrera *et al*. [9] and Papathanasiou *et al*. [10] investigated the possibility of saving energy by replacing high-speed server disks with arrays of smaller form-factor disks with almost the same aggregate I/O throughput, but their approaches do not address file placement problems. Pinheiro *et al*. [15] proposed a technique called popular data concentration (PDC) that dynamically migrates frequently-accessed data to a subset of the disks in an array. PDC skews the load towards a few of the disks, allowing the others to be sent to low-power modes. But PDC may cause a large number of file movements because it makes no distinction between different types of disk access.

These techniques are all targeted at server storage systems, whereas we are considering mobile storage systems. However, the basic idea of energy conservation using I/O concentration through static and dynamic file placement can be applied to mobile storage systems with multiple disks.

There has been quite a lot of research on saving energy in mobile storage systems by combining hard disks with flash memory in various configurations. March *et al*. [4], Bisson *et al*. [5], and Chen *et al*. [6] have all proposed using flash memory as a non-volatile cache, maintaining blocks which are likely to be accessed in the near future, and thus allowing a hard disk to spin down for longer time. Bisson *et al*. focused on the redirection of write requests to a flash memory device instead of a hard disk, while Chen *et al*. have recently studied a technique of partitioning the flash memory into a cache, a prefetch buffer, and a write buffer to save energy. Douglis *et al*. [14] examined three alternatives for mobile storage in terms of energy consumption and read/write performance: a hard disk, a flash disk, and a flash memory card. Samsung and Microsoft have recently developed the Hybrid Hard Disk Drive (H-HDD) which combines a hard disk and a NAND flash memory device for performance boosting, low power, and high reliability on mobile computers [2, 7, 8]. The H-HDD also uses the NAND flash memory device as a non-volatile cache.

Our work is distinct from the above research because it integrates flash memory with a hard disk as secondary storage. Moreover, our work includes an energy-efficient technique which augments the PDC method by separating I/O requests into reads and writes.

6. CONCLUSIONS

Hard disks and flash memory already have in their own distinct shares of the mobile storage markets. However, if the cost per bit of flash memory becomes comparable to that of hard disks in the near future, they will become direct competitors. In the mean time, we expect that mobile systems with heterogeneous multiple storage devices will soon be available.

In this paper, we have shown that combining a hard disk and a flash memory device can provide an energy-efficient secondary storage solution for mobile platforms. We have extended an existing data concentration technique by concentrating read and write I/O requests separately, and shown that this approach can reduce energy consumption and the number of flash memory erasure cycles. Workload-based simulations show that our scheme reduces the energy consumption by up to 74.5% when a 1.8″ disk and a flash memory are used instead of a single 2.5″ disk, at the cost of 34.1ms increase in the average response time. Compared with the existing PDC scheme, our approach saves more than 50% of disk energy consumption and improves the average I/O response time by a factor of 7.4.

Our work can be extended in several directions. For example, in order to fully evaluate the proposed techniques, we need to explore the effect of different workloads in more detail, varying parameters such as read/write ratio, I/O request rate, and file popularity. We are also investigating the use of multiple (i.e. more than 2) storage devices for higher energy conservation while meeting performance constraints. It will be an interesting future work as well to build a specialized software FTL which can be customized to work with the proposed file placement technique, improving the reliability of heterogeneous mobile storage systems.

7. ACKNOWLEDGMENTS

This research was supported by the MIC (Ministry of Information and Communication), Korea, under the ITRC (Information Technology Research Center) support program supervised by the IITA (Institute of Information Technology Assessment). And the ICT at Seoul National University provided research facilities for this study.

8. REFERENCES

[1] L. D. Paulson, "Will hard drives finally stop shrinking?," *IEEE Computer*, vol. 38, no. 5, pp.14-16, 2005.

[2] Microsoft, ReadyDrive and Hybrid Disk. http://www.microsoft.com/whdc/device/storage/hybrid.mspx.

[3] http://www.engadget.com/2006/03/10/samsung-shows-off-flash-laptop-drive-at-cebit/.

[4] B. Marsh, F. Douglis, and P. Krishnan, "Flash memory file caching for mobile computers," in *Proc. of the 27th Hawaii International Conference on System Sciences*, Hawaii, USA, pp.451-460, Jan. 1994.

[5] T. Bisson and S. Brandt, "Reducing energy consumption with a non-volatile storage cache," in *Proc. of International Workshop on Software Support for Portable Storage (IWSSPS),* held in conjunction with *the IEEE Real-Time and Embedded Systems and Applications Symposium (RTAS 2005)*, San Francisco, California, March, 2005.

[6] F. Chen, S. Jiang, and X. Zhang, "SmartSaver: turning flash drive into a disk energy saver for mobile computers," in *Proc. of 11th ACM/IEEE International Symposium on Low Power Electronics and Design (ISLPED'06)*, Tegernsee, Germany, October 4-6, 2006, in press.

[7] http://www.samsung.com/Products/HardDiskDrive/news/HardDiskDrive_20050425_0000117556.htm.

[8] R. Panabaker, "Hybrid Hard Disk & ReadyDrive™ Technology: Improving Performance and Power for Windows Vista Mobile PCs," in *Proc. of Microsoft WinHEC 2006*. http://www.microsoft.com/whdc/winhec/pres06.mspx.

[9] E. V. Carrera, E. Pinheiro, and R. Bianchini, "Conserving disk energy in network servers," in *Proc. of the 17th International Conference on Supercomputing*, June 2003.

[10] A. Papathanasiou and M. Scott, "Power-efficient server-class performance from arrays of laptop disks," Technical Report 837, Department of Computer Science, University of Rochester, May 2004.

[11] Hitachi GST, Travelstar 80GN. http://www.hitachigst.com/tech/techlib.nsf/products/Travelstar_80GN.

[12] Toshiba, MK4004GAH. http://www3.toshiba.co.jp/storage/english/spec/hdd/mk4004gs.htm.

[13] H. G. Lee and N. Chang, "Low-energy heterogeneous non-volatile memory systems for mobile systems," *Journal of Low Power Electronics*, Vol. 1, Number 1, pp. 52-62, April, 2005.

[14] F. Douglis, F. Kaashoek, K. Li, R. Caceres, B.Marsh, and J. A. Tauber, "Storage alternatives for mobile computers," in *Proc. of the 1st Symposium on Operating Systems Design and Implementation (OSDI)*, 1994.

[15] E. Pinheiro and R. Bianchini, "Energy conservation techni-ques for disk array-based servers," in *Proc. of the 18th International Conference on Supercomputing (ICS'04)*, June 2004.