Modeling Hard-Disk Power Consumption

John Zedlewski[*] Sumeet Sobti[*] Nitin Garg[*] Fengzhou Zheng[*]
Arvind Krishnamurthy[†] Randolph Wang[*]

Abstract

Excessive power consumption is a major barrier to the market acceptance of hard disks in mobile electronic devices. Studying and reducing power consumption, however, often involves running time-intensive disk traces on real hardware with specialized power-monitoring equipment. This paper presents Dempsey, a disk simulation environment that includes accurate modeling of disk power consumption. It includes tools to automatically extract performance and power consumption parameters from a given disk drive, without needing detailed specifications from the manufacturer. The tools use stimulus-based measurements to extract these parameters. Dempsey is experimentally validated for two mobile hard disks, namely, the 1 GB IBM Microdrive and the 5 GB Toshiba Type II PC Card HDD. In the worst observed case, Dempsey's estimate of power consumption differs from the measured consumption by 7.5%. This demonstrates that disk power consumption can be simulated both efficiently and accurately.

1 Introduction

Many mobile electronic devices, such as MP3 players, digital cameras and personal digital assistants, exhibit an almost insatiable demand for storage capacity. These devices have traditionally relied on compact flash memory, which is reasonably fast and power-efficient, but also expensive and capacity-limited. Recent advances in magnetic disk technology have made possible the development of high capacity, small form-factor disk drives that are compatible with traditional mobile interfaces, such as PCMCIA and CF+ Type II slots.

[*]Department of Computer Science, Princeton University. Email: zedlwski@princeton.edu, {sobti, nitin, zheng, rywang}@cs.princeton.edu.

[†]Department of Computer Science, Yale University. Email: arvind@cs.yale.edu.

This work is supported in part by IBM. Wang is supported by NSF Career Award CCR-9984790 and Krishnamurthy is supported by NSF Career Award CCR-9985304.

Unfortunately, one important hurdle continues to block the widespread acceptance of these miniature hard disks in mobile devices: *power consumption.* Recent studies have demonstrated that a small form-factor disk, such as the IBM Microdrive, may consume 5-10 times more power than its flash memory counterparts [26]. In the context of a notebook computer with a powerful lithium-ion battery, these levels of energy consumption are quite bearable, and the storage subsystem in such computers is rarely responsible for more than 10-30% of the overall power drain [4]. In an MP3 player running on AAA batteries, on the other hand, every Joule is critical. Thus, effective power management of hard disks, especially the mobile ones, is becoming increasingly important.

Research in effective disk power management can be a frustrating process, as meaningful disk traces take days to run, and researchers need access to expensive power-monitoring equipment. In this scenario, simulation quite naturally seems like an attractive approach to follow. Simulation software has already been demonstrated to be able to model disk performance, both efficiently and accurately [6, 19, 20]. In this paper, we present Dempsey (Disk Energy Modeling and Performance Simulation Environment), a tool that seeks to bring a similar level of accuracy and convenience to energy-aware storage system designs.

Dempsey extends the well-tested DiskSim simulator [6] to model *power* consumption in addition to *performance* characteristics of hard disks. It includes a set of tools to extract the necessary power consumption parameters from a given disk. Dempsey uses stimulus-based measurements to derive all the required performance and power consumption parameters. Thus, no detailed specifications from the manufacturer are necessary. This enables it to handle disks with the IDE interface, which in general lacks commands to determine specific internal parameters of the disk. This ability to handle IDE disks is significant, since IDE is the dominant standard for mobile disks today.

Dempsey is experimentally validated for the

1 GB IBM Microdrive and the 5 GB Toshiba Type II PC Card HDD using a variety of synthetic and real-world traces. For the IBM Microdrive, Dempsey's estimate of power consumption differs from the measured consumption by 7.5% in the worst observed case. In the average case, however, the error is only 1.8%. The corresponding errors for the Toshiba HDD are 6.9% and 3.6% respectively. On a modern desktop machine, Dempsey is able to simulate traces at a rate of more than 8000 disk-requests per second.

Section 2 provides an overview of relevant existing work in the fields of disk performance modeling and disk power management. Details of Dempsey's design and implementation, including the performance and power modeling components, are given in Section 3. In Section 4, experimental results are presented, which validate the simulator for the two representative disks. Section 5 presents the conclusions.

2 Related Work

Dempsey attempts to connect two research fields: *disk performance modeling* and *disk power management.* We survey some related work in these two fields.

2.1 Disk Performance Modeling

Ruemmler and Wilkes present a thorough introduction to disk performance modeling [19], and convincingly demonstrate the need for sophisticated disk-simulation techniques. They suggest the use of the "demerit figure" as a measure of a simulator's accuracy. The demerit is defined as the root mean square of the horizontal difference between the simulated and real response-time distribution curves for a given trace. They show how demerit figures of as low as 3% can be achieved by simulating the disk behavior in extreme detail. Similar simulation techniques are used in [12, 15, 22].

DiskSim [6], developed by Ganger, Worthington and Patt, is a general-purpose simulator that goes beyond the techniques of Ruemmler and Wilkes. The DiskSim software requires a large set of parameters to characterize a disk drive. These parameters include nearly a hundred behavioral details and overhead timings, in addition to detailed disk geometry information and a large table of seek times. Much of this complexity stems from DiskSim's goal of simulating the widest possible range of disks. The source code for DiskSim is publicly available, and it has been used as the basis for Dempsey's implementation.

Several recent projects have attempted to automate the process of extracting these large number of disk performance parameters. Many of these rely on a combination of two kinds of techniques [24], namely interrogative extraction and empirical extraction. *Interrogative extraction* makes use of low-level commands in the disk interface to extract information about geometry and other static properties of the disk. Such techniques have successfully been used with many SCSI disk drives [20, 24]. Interrogative extraction, however, is not sufficient for several reasons. First, a given disk drive may not support all interrogative commands. Second, the information returned by such commands may be inaccurate, in which case it is only usable as a hint. *Empirical extraction* observes the behavior of a given disk on a set of carefully-chosen synthetic workloads, and extracts the parameter values from these observations. Since empirical extraction does not depend on specific commands in the disk interface, it is more generally applicable [2, 21].

Dempsey mainly uses empirical extraction techniques to extract the relevant performance and power parameters from a given disk. This makes Dempsey relatively general-purpose and suitable for disks with the IDE interface. The extraction techniques for the performance parameters are quite similar to those described in existing work, like DIXTrac [20]. To these, we add techniques for extracting power parameters.

2.2 Disk Power Management

Most research in disk power management has focused on the behavior of the disk during periods of inactivity, i.e., idle periods. Specifically, the question is when the disk should be put to sleep to minimize power consumption with little impact on performance. Many papers have analyzed the impact of aggressively spinning down disks when the time since last I/O request exceeds some threshold [4, 14, 23]. Algorithms for dynamically varying the spin-down threshold in response to changing user behavior and priorities have also been proposed [3, 7, 10, 13]. IBM's storage systems division has developed an adaptive power management algorithm [1] called ABLE (Adaptive Battery Life Extender). ABLE has been incorporated in IBM 2.5-inch Travelstar drives and IBM Microdrives.

The simulators employed in these projects are simpler. Greenawalt uses an analytical model that assumes that requests arrive according to a Poisson distribution [8]. Helmbold et. al. [10] model power in terms of seconds of activity, rather than using Joules. This simplification relies on two implicit assumptions. One is that a disk has only two distinct power levels: active and idle. The second is that an active disk always consumes power at the same rate. Douglis et. al. [4] use a disk simulator that uses a fixed, average response time for all requests, except for those that lie within a small neighborhood of the previous request. This is quite similar to the model that Reummler and Wilkes show to have a demerit of 35% [19].

These earlier simplified simulations are valuable in studying disk spin-up and spin-down policies. One common assumption shared by these studies is a given disk I/O access pattern generated by a given file system. Researchers, however, have recently begun to investigate how to influence the disk I/O access pattern to reduce energy consumption, sometimes by changing the file system itself. Zheng et. al. [26], for example, analyze the effect of various file system attributes, like data layout policy, burstiness, background data reorganization algorithms, etc., on disk energy consumption. Even when the user issues the same sequence of system calls, the disk I/O requests issued by a log-structured file system, for example, can be very different from those generated by an update-in-place file system, in their locality and burstiness characteristics. A simulator that lacks a level of sophistication that is comparable to that seen in DiskSim, among other disadvantages, may err in its timing estimate of individual I/O requests, which may translate into inaccuracies in its energy estimate. Papathanasiou and Scott [16] explore file-system level techniques for increasing the burstiness of disk accesses. Heath et. al. [9] and Weissel et. al. [23] attempt to achieve similar goals by making applications more "energy-aware". We believe that an accurate and efficient disk power modeling tool like Dempsey can be very useful in projects of this kind. We also expect Dempsey to be useful for systems like ECOSystem [25] where accurate on-line estimates of energy consumption are needed outside a laboratory setting.

3 Dempsey

Dempsey extends the DiskSim simulator with a component to model disk power consumption. The power modeling component is relatively simple, adding fewer than 200 non-comment source lines of code to the existing DiskSim software. Given data files describing the performance and power characteristics of a disk, the simulator can take an input trace file and quickly return an estimate (in Joules) of the energy that would be consumed by executing the given trace on the specified disk. The simulator also produces the standard DiskSim output, which describes performance and response time characteristics.

Dempsey includes tools that automatically extract the required performance and power parameters of a given disk. In all, these tools contain about 2500 source lines of code in C++, Python and bash shell scripts. These tools use the *empirical extraction* technique to extract disk parameters. Specifically, the behavior of the disk on a set of carefully-selected synthetic workloads is observed, and parameter values are extracted from these observations. The performance characterization tools are similar, in nature, to those used in the DIXTrac project [20], so they will be described only in brief in Section 3.2. The power modeling component, however, is described in detail in Section 3.3.

3.1 Measurement Infrastructure

To compute the parameters required for power simulation, the characterization tools need to measure actual power consumption for a variety of synthetic traces. For this, the tools need to interface with a multimeter or a voltmeter. Clearly, it would be preferable to eliminate this need for special equipment, but other observable indicators of power consumption, such as battery life, etc., simply do not provide the speed and accuracy necessary for a detailed understanding of power usage. It is also important, however, to note that the *simulation* component of Dempsey does not require special equipment. Once a disk's power consumption has been characterized, researchers can use the resulting power parameters to model the disk drive without needing to carry out any tests with special equipment. This section describes Dempsey's hardware and software infrastructure.

3.1.1 Hardware

To gather power statistics, we have fashioned a PC card sleeve as pictured in Figure 1, with a schematic diagram in Figure 2. The power measurement sleeve consists of a PC card extender attached to a shunt resistor in series with the card's power supply. Following the general approach taken in [5, 11], we measure the voltage across the shunt resistor. The card extender connects to a PCMCIA-compatible disk drive, such as an IBM Microdrive. The card

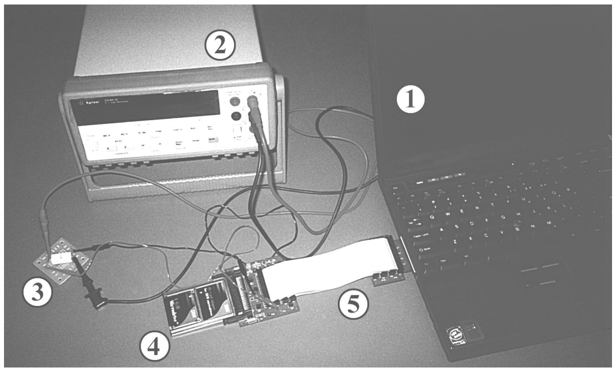

Figure 1: Power measurement apparatus. (1) Linux-based PC with a PCMCIA slot, (2) Digital multimeter, (3) Shunt resistor, (4) IBM Microdrive housed in a PC card adapter, (5) PC card extender.

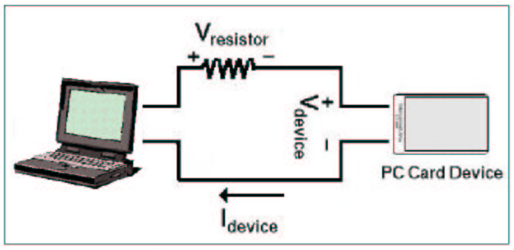

Figure 2: Schematic diagram of the power measurement apparatus.

extender is inserted into a PCMCIA slot of a Linux-based PC. A digital multimeter is used to measure the shunt resistor voltage, and integrate it over a sampling interval to compute the average resistor voltage for that interval. These measurements are logged via a serial link to the PC over the course of an experiment. After independently determining the ohmage of the shunt resistor, we deduce the average current delivered to the disk drive via Ohm's law $I_{Drive} = V_{Resistor}/R_{Resistor}$. This leads to the average power consumed by the drive over the sampling interval using $P_{Drive} = V_{Drive}I_{Drive}$, where V_{Drive} is the voltage of the power supply to the disk drive. The total energy consumed by the drive over the sampling interval is, then, the product of P_{Drive} and the length of the sampling interval.

In all of our measurements, we configure the multimeter to produce 3 samples per second, although the multimeter itself is capable of operating at much higher sampling rates. In other words, the time interval over which average voltage is measured across the shunt resistor is approximately 333 ms. This is a choice in favor of overall accuracy. The multimeter, in producing each sample, needs to spend some time performing computation, and the voltage change during the computation time is basically ignored. So, fast sampling rates, though good for showing peak values, are generally bad for overall accuracy. All of our experiments run several seconds or more. Thus, a relatively long sampling time interval is acceptable in this context.

3.1.2 Software

A typical power measurement experiment has two main tasks: (1) to execute the disk trace, and (2) to record average voltage measurements from the serial port. In the Dempsey setup, these two tasks are performed by two concurrent threads. It is important to ensure that these threads are scheduled fairly during the measurement. To minimize inaccuracy due to unfairness in relative scheduling of the threads, Dempsey schedules both threads as real-time processes, with the port reader having a higher priority than the execution thread. Additionally, the experiments for this paper are run on a Linux system that uses a kernel patched for low-latency and pre-emptible operation. An alternative would have been to use two separate computers, one to execute the trace and the other to record the measurements. The current setup, however, gives sufficient accuracy that we do not use a second computer.

To transfer data to the drive, the execution thread uses the Linux raw device interface (*/dev/raw*), which bypasses the operating system's buffer cache and allows the software to read from or write to arbitrary sectors. The disk also has an internal cache that can interfere with attempts to measure performance. The Microdrive, for example, has both a read cache and a write cache, and it allows a user to disable the write cache, but not the read cache. Unless otherwise noted, the write cache is enabled during the experiments in this paper.

3.2 Performance Modeling

Dempsey uses the DiskSim software to model the execution of a given trace on a given disk. DiskSim models the execution in extreme detail, including modeling different stages of the execution, namely, *seeking*, *rotation*, *data transfer* and *idle periods*. Dempsey adds code to model energy consumption during each of these stages.

To simulate the execution, DiskSim requires specific values for a large number of parameters that characterize disk geometry and layout, mechanical timings and cache behavior, among other things. Dempsey includes performance characterization tools to automatically extract these parameters from a given disk.

Typically, the first step in such characterization is the extraction of detailed information about disk geometry and physical layout of data blocks on the disk. For this, many previous efforts, including DIXTrac [20], have relied on low-level commands in the disk interface that reveal the mapping from logical block addresses (LBAs) to physical locations on the disk. The SCSI interface, for example, has the "Translate" option in its "SEND DIAGNOSTIC" and "RECEIVE DIAGNOSTIC" commands to translate a given LBA to the corresponding physical location.

Dempsey chooses not to rely on such translation commands, since not all disks support such commands. Also, even if a disk does support such commands, the information returned may not be entirely correct and reliable. For example, the IBM Microdrive, which supports the IDE interface, does include a "Translate Sector" command to translate a given LBA to the corresponding cylinder/head/sector (CHS) combination. Unfortunately, this CHS address does not represent the true physical location of the corresponding sector on the disk. Dempsey uses empirical extraction techniques to extract the geometry information, which is then used in the extraction of all other relevant parameters through empirical extraction techniques similar to those used in [20].

3.2.1 Extracting Disk Layout

The most important feature of the mapping between LBAs and their corresponding physical locations on a given disk drive is the drive's *zoning* strategy. Zoning refers to the technique of dividing a drive's surface into several groups (or zones) of tracks, such that all tracks within a given zone have an equal number of sectors. Tracks in different zones may have different numbers of sectors. This allows more sectors on the outermost tracks, which are longer than the innermost tracks. Because LBA to physical-location mappings are generally sequential, an accurate description of a disk's zones is nearly equivalent to a description of a drive's layout, provided that the drive's geometry has not been substantially affected by defects and that the number of heads is known.

Finding tracks. A fundamental building block of the zone discovery algorithm is the `Same_Track`(L) function. `Same_Track` returns a range of LBAs that includes all LBAs on the same track as LBA L. Let `Seek_Time`$(L_1,\ L_2)$ denote the time taken for a seek from LBA L_1 to LBA L_2. (A function to compute `Seek_Time` can be implemented using the SEEK command directly, so that it bypasses issues of caching and rotational latency.) The `Same_Track` function begins by computing `Seek_Time`$(L,\ L)$. This represents the time necessary to perform a zero-distance seek, essentially equivalent to the bus and command processing overhead. Any seek of non-zero distance will take substantially longer. The `Same_Track` function can then compute `Seek_Time`$(L,\ L_k)$ for a series of L_k values. If the `Seek_Time` value is more than twice of the minimal seek time (this factor of two is an arbitrary threshold that proves to work well in practice), then it is inferred that L_k does not lie on the same track as L. `Same_Track` can then use binary search to discover the upper and lower boundaries of the desired track.

From tracks to zones. Note that using the `Same_Track` function, it is trivial to determine the number of sectors on the track on which a given LBA lies. Let `Num_Sectors`(L) denote the number of sectors on the track on which L lies. Remember that a zone is defined to be a set of consecutive tracks, all of which have the same number of sectors. Let `Same_Zone`(L) be a function that returns the range of LBAs that lie in the same zone as LBA L. `Same_Zone` can easily be implemented using the `Num_Sectors` function and binary search.

Starting with the disk's first LBA and repeatedly applying `Same_Zone` function until the disk's final LBA is reached, Dempsey is able to generate a complete map of the disk's zones.

Disk heads. The boundaries, which record the number of blocks per track and the starting and ending LBAs of each zone, are still not enough information to map an LBA to a physical cylinder/head/sector address. On a disk with multiple heads, each cylinder logically contains tracks spread across several different data surfaces. Therefore, if a zone has S_z total sectors and S_t sectors per track, the zone contains $\frac{S_z}{(S_t \times H)}$ cylinders, where H is the number of heads on the disk. Clearly, then, Dempsey must know the number of physical heads on the disk in order to understand its geometry.

The current version of Dempsey relies on the manufacturer's specifications to determine the number of disk heads in a drive, although techniques for experimentally determining this number are also known [20].

3.2.2 Extracting Other Parameters

Other performance parameters, like the seek curve, the rotation speed and the cache parameters, are extracted using techniques similar to those used in the DIXTrac project. Therefore, these techniques are not described here in any detail.

As noted in [20], some disk drives that incorporate *adaptive caching techniques* (where the organization of the on-disk cache changes in response to changing disk access patterns) are not very accurately modeled by DiskSim. Both the IBM Microdrive and the Toshiba HDD are observed to have an adaptive cache. Thus, following [20], Dempsey uses average values for many cache parameters in the parameter files.

3.3 Power Modeling

Traditionally, most approaches to power modeling of hard disks have been very *coarse-grained*. For example, many previous attempts have modeled disk to be in one of two states at any given time: *active* and *sleep*, and power consumption in each of these states has been assumed to be constant. On the other hand, Dempsey's approach is fairly *fine-grained*. Dempsey attempts to accurately estimate the energy consumed by each of several sub-components of a disk request. Each disk request consists of several stages: a seek to the correct cylinder, a period of wait until the disk rotates to the correct position, an actual data transfer, and, possibly, a period of idleness before the next disk request is handled. In this section, we describe how Dempsey computes the values of various power parameters required to model the energy consumption for each of these stages, namely, (1) seeking, (2) rotation, (3) reading, (4) writing, and (5) idle-periods. Table 1 summarizes the power parameters for the two drives. We also describe how Dempsey uses these parameters to arrive at its estimates for the energy consumption for these stages during the execution of a disk trace.

3.3.1 General Approach

To compute the power parameters, Dempsey needs to measure the average power consumption for each of the above-mentioned disk stages. This is a significant challenge because any stage may complete in as little as half a millisecond, whereas the multimeter-based apparatus can, at best, take 75 samples per second. Thus, it is not possible to measure accurately the power consumption of a single stage in a single request, even if the software could somehow identify exactly when the disk transitions between different stages (which itself is a hard problem). Therefore, all of Dempsey's power characterization experiments rely on power measurements taken over the course of longer traces.

Stage	IBM Microdrive	Toshiba HDD
Seeking	637	1287
Rotation	594	1122
Reading	627	1185
Writing	756	1430
Stand-by	61	231

Table 1: Average power consumed (mW). The number for the "Seeking" stage is the average power consumed during a seek over one-third of the maximum seek distance.

Energy consumed during a single sampling interval can be computed using the reading from the multimeter, as described in Section 3.1.1. Total energy consumed by a trace is the sum of the energy consumptions of all the sampling intervals. In order to obtain a measure of the average power consumed by a specific disk stage S, Dempsey runs two traces that differ only in the amount of time spent in stage S. The following formula, then, gives the average power consumption for disk stage S.

$$\overline{P}_S = \frac{E_2 - E_1}{T_2 - T_1}$$

where E_i is the total energy consumed by trace i and T_i is the total time taken by trace i. Note that if the two traces differ only in the time they spend in stage S, then the numerator is the extra energy spent in stage S and the denominator is the extra time spent in stage S. We refer to this method of estimating average power consumption of an individual stage as the *Two-Trace method*.

3.3.2 Seeking

Dempsey generates a Seek-Power Profile to model the seeking stage. The profile lists the energy consumed by the disk for seeks of various distances, measured in number of cylinders crossed. Figure 3 presents the measured Seek-Power Profile for the IBM Microdrive.

Measuring seek power is relatively easier than measuring power consumption for other stages. For this purpose, Dempsey uses the SEEK command directly. To estimate the energy consumed by a seek of C cylinders, the Two-Trace method is used as

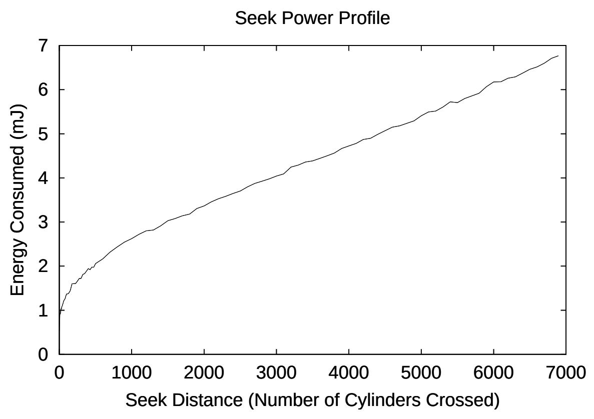

Figure 3: Seek-Power Profile for the IBM Microdrive.

follows. The first trace consists of only SEEK commands addressed to an LBA L chosen randomly at the beginning of the experiment. The second trace alternates evenly between SEEK commands addressed to L and those addressed to another block that is C cylinders away from L. Neither trace spends any time in the rotation, reading, writing or idle periods, so the traces only differ in the seeking stage.

3.3.3 Rotation

The rotation stage of a request occurs after the disk head has reached the correct track. The disk must then wait until its rotation places the desired sector beneath the active head. Although the disk is neither seeking nor transferring data during this interval, it is also not "idle" in the strict sense. During a true idle period, in which no requests are active, a disk may be able to shut down certain components in order to reduce power consumption. Thus, a disk may consume more power during a period of rotation than it does during a true idle period. For that reason, Dempsey includes tests that measure the power consumed while the disk waits for rotation to complete.

For this stage, Dempsey uses a single power parameter, which is a measure of average power consumption during rotation. Again, the Two-Trace method is used. A track T is selected randomly. Both traces issue the same number of single-block writes to T. The write-cache of the disk is disabled during this experiment. The first trace writes blocks on T sequentially, wrapping around when the last block on the track is reached. The second trace writes blocks on T with a larger *stride* between each pair of successive requests.

Each trace contains the same number of single-block write requests, ensuring that they spend equal amounts of time in the writing stage. Neither trace includes any seeking, reading or idle period. Thus, they differ only in their time spent in the rotation stage. Therefore, the Two-Trace method yields the correct power consumption for the rotation stage.

While the exact size of the large stride for the second trace is not particularly important, the stride must be large enough to ensure a substantial difference between the two traces. Dempsey uses a 50-block stride as an arbitrary choice.

3.3.4 Reading and Writing

Dempsey uses one power parameter each for the reading and writing stages. These parameters are a measure of average power consumption during these stages. Dempsey supports two separate methods to compute read and write power consumption. The decision of which method to use depends on whether the disk supports *zero-latency access* or not. When a disk *without* zero-latency access receives a large read or write request, it must wait for the disk to spin until it reaches the *first* sector of the request before it begins transferring data. A disk *with* zero-latency access (also called a read/write-on-arrival disk), on the other hand, can begin transferring data when the disk head is positioned above *any* of the sectors in the request.

To determine whether or not a disk supports zero-latency access, Dempsey issues a single-block read request at the first block of a track, followed by a request to read the entire track, beginning at the first block. In a disk without zero-latency access, this process will take two rotations, whereas in the other case it will take only slightly longer than one rotation. (This same technique is used by DIX-Trac [20].)

Measuring for disks without zero-latency access. To measure read power in a disk without zero-latency access, Dempsey again uses the Two-Trace method. A random sequence of distinct tracks is selected. The first trace consists of 1-block reads to the first block in each of the selected tracks. The second trace consists of whole-track reads to each of the selected tracks, where each read begins at the first block in the track. It is easy to see that both traces spend *almost* equal times in the seeking and rotation stages. Neither trace includes any idle time. Thus, the Two-Trace method yields a reasonable approximation of the power consumption parameter for the reading stage.

The power parameter for the writing stage is computed in the same manner, with the write cache of the disk disabled.

Measuring for disks with zero-latency access. It is easy to observe that the traces used above can not be used in this case, because here the two traces will differ significantly in the time they spend in the rotation stage.

To make it work for disks with zero-latency access, we need to change the first trace slightly. Here the first trace, instead of issuing 1-block read commands, issues SEEK commands to each of the tracks in the selected sequence. The second trace issues whole-track reads as in the previous case. It is easy to verify that these two traces, when executed on a zero-latency disk, will differ only in the reading stage. Thus, the Two-Trace method can be applied to obtain the average power consumption of the reading stage. Note that the traces that work in this case do not work for the previous case, because they will differ significantly in the time they spend in the rotation stage when executed on a disk without zero-latency access.

3.3.5 Idle Periods

The behavior of a hard disk during idle periods is substantially more complex than during the disk-stages described above. Typically, several power modes are defined where performance is traded-off for savings in power consumption. For example, many disks have four modes of operation: *active*, *idle*, *standby* and *sleep*. The active-mode is the only mode in which the disk can satisfy requests. The active-mode has the highest power consumption, followed by the idle-mode, the standby-mode and the sleep-mode in that order. The intention is to let the disk operate in one of the low power modes when there is no disk activity, but transitioning from one power mode to another usually incurs time and energy overheads. Thus, power management usually has implications for power and performance (response times, in particular).

Modeling a disk's power management scheme can be decomposed into two tasks. First, the energy and performance overheads associated with mode transitions need to be measured. Second, the power simulator requires a model of when the mode transitions occur.

To measure the cost of mode transitions, Dempsey issues a series of traces to generate an *Idle-Period Energy Profile*. Each trace has a large number of I/O operations that are separated by an idle period of constant length. By varying the length of idle periods across traces, Dempsey generates a series of tuples of the form (t, E, T). A tuple (t, E, T) records the fact that an idle period of length t consumes energy E, and introduces delay T in the response time for the following request. The Idle-Period Energy Profile can be used to determine accurately the modes used by the power management scheme, the overheads incurred during mode transitions, and the power consumed by the disk in different modes.

The task of developing a model for when the disk transitions between modes is more complex. Traditionally, disks have used fixed waiting thresholds to transition from higher power to lower power modes. For such disks, the Idle-Period Energy Profile is sufficient to predict when mode transitions would occur. The fixed thresholds for mode transitions can be inferred from the Idle-Period Energy Profile, and Dempsey provides a default mechanism for determining such fixed thresholds.

Newer disks, however, are moving toward more sophisticated mechanisms for managing their operation. For example, the IBM Microdrive employs the Adaptive Battery Life Extender (ABLE) technology, which, in addition to defining many more power modes, *adaptively* manages transitions between those modes. ABLE continuously monitors disk-request pattern and maintains statistics on the recent history of disk requests. Instead of fixed waiting thresholds, ABLE's decisions on when and how to make power-mode transitions are determined by its predictions about the current request-burst, the current level of internal disk-activity (prefetching, write-behind, etc.), the desired performance-level and the energy costs associated with the transitions.

Ideally, one would like to use the exact ABLE algorithms when modeling the IBM Microdrive in Dempsey, but since these algorithms are not available publicly, we use the fixed-threshold model with the following straightforward implementation. The estimate for the energy consumed during a given idle period of length t is computed by looking up the appropriate tuple in the energy profile. If no tuple is found for length t, then interpolation is used to arrive at an estimate. The time overhead is fed back into the performance modeling component of Dempsey so as to keep the response-time estimates accurate.

3.4 Simulation

Dempsey inherits the performance simulation module from DiskSim, which models the disk in extreme detail. To obtain an estimate of total energy consumption for a given trace, the simulator

Component	Description
Laptop	IBM ThinkPad T20 750 MHz P3, 128 MB RAM Linux Operating System
Multimeter	Agilent 34401A
Shunt Resistor	0.47 Ohm
PC Card Extender	Sycard PCCextend

Table 2: Various hardware components of the experimental setup.

	IBM Microdrive	Toshiba HDD
Model	DSCM-11000	MK5002MPL
Capacity (GB)	1	5
Seek Time (ms)		
Track-to-Track	3	3
Avg. Seek	13	15
Max. Seek	20	26
Rotation (RPM)	3600	3990

Table 3: Detailed characteristics of the IBM Microdrive and the Toshiba HDD.

simply computes estimates for the seeking, rotation, reading, writing and idle-period stages, and adds them up. The Seek-Power Profile is used to arrive at the estimate for the seeking stage. For the rotation, reading and writing stages, the corresponding power parameter is multiplied with the estimated time spent in the stage to arrive at the energy estimate for that stage. The energy spent during the idle periods is estimated as described in Section 3.3.5 above.

4 Experimental Validation

This section presents some experimental results that contribute toward validating the Dempsey simulator for the IBM Microdrive and the Toshiba HDD. We also compare Dempsey against several other alternatives which model the disk power consumption in less detail than Dempsey. We use both synthetic and real-world traces to perform the evaluation. Detailed specifications of various hardware components of the experimental setup and the mobile disks are provided in Tables 2 and 3.

4.1 Synthetic Traces

A disk trace is a sequence of disk requests, each of which has four components: request arrival time, starting block address, size of request and type of request (read/write). This suggests four natural dimensions along which a synthetic trace can be characterized.

- **Delay.** This refers to the probability distribution from which delay intervals between successive requests are chosen. We consider 4 different distributions. (1) *Fixed*: delay interval of a fixed size is always chosen. (2) *Standard*: delay interval is chosen uniformly at random in the range 1-80 ms (this is the distribution used in the DIXTrac validation test [20].) (3) *Long*: here a standard delay is chosen with 0.9 probability and a random delay between 1-8 seconds is chosen with 0.1 probability. (4) *Very Long*: here a random delay is chosen between 1-200 ms with 0.98 probability and between 5-20 seconds with 0.02 probability.

- **Access Pattern.** This determines the sequence of addresses accessed in the trace. Three kinds of access patterns are considered here. (1) *Sequential*: the blocks are sequentially accessed on the disk. (2) *Random*: block addresses in the trace are randomly chosen from the range of valid disk addresses. (3) *Cache Test*: this test is used in [20] to test the accuracy of the disk-cache modeling. In this pattern, 20% of the requests are sequential, 30% are local (i.e., within 250 blocks of their predecessor in either direction), and 50% are completely random.

- **Transfer Size.** This parameter specifies the number of blocks to transfer with each request. The traces used here have two kinds of transfer sizes: (1) *Fixed*, and (2) *Standard*: where size of each request is chosen uniformly at random between 16-24 sectors (i.e., 8-12KB.)

- **Request Type.** This refers to the distribution of read and write requests in the trace. Besides purely read and write traces, the evaluation uses randomly-generated *mixed* traces with read-write ratio of 2:1.

For each synthetic trace, it is necessary to specify a value for each of the four parameters described above. We use eight representative synthetic traces in our experiments. Table 4 summarizes the synthetic traces used. Unspecified parameters in the trace descriptions have the default values given in Table 5. Trace I, with 400 ms delay interval between successive requests, tests the simulator's ability to model medium-length idle periods in which the drives do not enter the low-power modes. Trace

Trace	Description
I	Delay: Fixed at 400 ms
II	Transfer Size: Fixed at 1 sector
III	All Parameters: Standard
IV	Delay: Long
V	Delay: Very Long
VI	Access Pattern: Sequential
VII	Access Pattern: Cache Test
VIII	Delay: Fixed at 100 ms, Access Pattern: Sequential, Transfer Size: Fixed at 128 sectors, Request Type: Only writes

Table 4: Summary of the synthetic traces used.

Parameter	Default Value
Delay	Standard (random in 1-80 ms)
Access Pattern	Random
Transfer Size	Standard (random in 8-12KB)
Request Type	Mixed (read-write ratio of 2:1)

Table 5: Default values for the synthetic trace parameters.

II, with single-sector requests, tests the simulator's accuracy for small requests. Trace III uses the standard parameters (i.e., default values in Table 5) from the DIXTrac traces to test the simulator's handling of mid-sized transfers. Traces IV and V include a number of long idle periods when the drives should enter the low-power modes. Traces VI and VII include non-random access patterns. Trace VIII contains large sequential writes, representing workloads that are typical of log-structured file systems [17].

4.2 Real-world Traces

We use a portion of the 1992 Cello Trace (Disk-2) from HP Labs [18] in our study. The trace from June 12^{th}, 1992 is broken into 6 traces of length 4 hours each. These traces are named A-F in the tables below. Since the Microdrive used has 1 GB capacity, all accesses in these traces to blocks after the first Gigabyte are removed. The traces A-F contain about 130,000 disk requests in all, of which about 81,000 are writes. Table 6 gives detailed characteristics of these traces.

4.3 Alternative Power Models

We compare Dempsey against three less-sophisticated power models. Table 7 lists the values used for the parameters in these models.

Trace	Reads	Writes	Total	Size	Rate
A	529	7460	7989	7	54
B	14942	4433	19375	5	207
C	5590	15619	21209	6	258
D	22107	36077	58184	7	390
E	4351	11765	16116	7	284
F	1742	5689	7431	7	289
Total	49261	81043	130304		

Table 6: Characteristics of the real-world traces. Each trace is 4 hours long. Columns 2-4 describe the number of reads, writes and total number of operations respectively. The fifth column gives the average request size (KB). The last column lists the maximum number of requests in a second in each trace. Note, however, that the disks may not be able to handle requests at these rates, and some requests may be delayed.

Parameter	IBM Microdrive	Toshiba HDD
P_{active}	624 mW	1186 mW
$P_{active-idle}$	531 mW	891 mW
P_{sleep}	61 mW	231 mW
T_0	2 s	15 s

Table 7: Parameter values for the three alternative models.

4.3.1 The 2-Parameter Model

The 2-Parameter model is, in fact, a naive model, which makes two simplifying assumptions: (i) the disk consumes energy at a constant rate when it is actively satisfying disk requests, and (ii) when the disk finishes a disk request and finds that there are no more pending requests in the queue, it immediately enters the sleep-mode. The model uses the following formula to compute an estimate of total energy consumption for a given trace.

$$E_{total} = E_{active} + E_{idle} \tag{1}$$

E_{active} is estimated as $P_{active}T_{active}$, where P_{active} is the assumed power consumption when the disk is active (regardless of whether it is in the seeking, rotation, reading or writing stage), and T_{active} is the time spent by the disk while actively satisfying disk requests. E_{idle} is estimated as $P_{sleep}T_{idle}$, where P_{sleep} is the assumed power consumption in the sleep-mode, and T_{idle} is the length of the entire idle-period in the trace. P_{active} is derived by executing a random trace of 12 KB accesses (evenly mixed with reads and writes), and computing the average power consumed by the trace. P_{sleep} is measured by observing the disk in the sleep-mode after it has been left idle for a long time.

4.3.2 The 3-Parameter Model

The 3-Parameter model improves upon the 2-Parameter model by further refining the modeling of idle periods. As in the 2-Parameter model, equation (1) is used to estimate the total energy consumption, and E_{active} is estimated as $P_{active}T_{active}$. The behavior during the idle periods, however, is modeled as follows. The model assumes that the disk enters an intermediate power-mode, called the "active-idle" mode, as soon as it finishes a disk request and finds no more pending requests in the queue. The transition from the active-idle-mode to the sleep-mode, however, is governed by a fixed waiting threshold T_0. For an idle-period of length L with $L \leq T_0$, the energy consumption is estimated as $P_{active-idle}L$. For an idle-period of length L with $L > T_0$, the energy consumption is estimated as $P_{active-idle}T_0 + P_{sleep}(L - T_0)$. Note that the time or energy overheads of the mode transitions are not modeled. $P_{active-idle}$ is measured by observing the disk immediately after a disk request has finished.

4.3.3 The Coarse-Dempsey Model

The Coarse-Dempsey model is a hybrid between the 3-Parameter model and Dempsey. As before, it uses equation (1) to estimate total energy consumption. E_{active} is estimated as in the 3-Parameter model, whereas E_{idle} is estimated as in Dempsey. Therefore, this model uses the complete Idle-Period Energy Profile of the disk, which also models the time and energy overheads of mode transitions.

4.4 Experimental Results

Tables 8 and 9 present results from executing the synthetic and the real-world traces on the IBM Microdrive. The tables compare the estimates from the power models with the actual (measured) power consumption on these traces. The corresponding results for the Toshiba HDD are presented in Tables 10 and 11.

The foremost conclusion that can be drawn from these results is that Dempsey is able to model disk power consumption quite accurately. For the IBM Microdrive, Dempsey's worst observed error is an underestimate by 7.5%, while the mean error is only 1.8%. For the Toshiba HDD, the corresponding errors are 6.9% and 3.6% respectively.

The simple 2-Parameter model is grossly inadequate. It consistently underestimates the energy consumption because it wrongly assumes that the disk spends all of its non-active time in the sleep-mode with the lowest power consumption, and because it does not model the energy overheads of mode transitions.

For the Microdrive, the 3-Parameter model appears to achieve a vast improvement over the 2-Parameter model in many cases. Here, the worst observed error for the 3-Parameter model is only 10.0%. This highlights the importance of being able to accurately model disk behavior during idle periods. The Coarse-Dempsey model and Dempsey consistently achieve higher accuracy only by being able to model idle periods more accurately. Recall that the 3-Parameter model does not model the energy and time overheads associated with power-mode transitions, which the Coarse-Dempsey model and Dempsey do. This explains why estimates from the 3-Parameter model are consistently lower than those from the Coarse-Dempsey model and Dempsey.

For the Toshiba HDD, however, the 3-Parameter model appears to be less accurate. On Traces IV, V and A-F, which include relatively long idle-periods, the 3-Parameter model overestimates the energy consumption by a large amount. This shows that its modeling of disk behavior during idle-periods is not very accurate. The reason for this is that the Toshiba disk has several *intermediate* power-modes between the highest power active-mode and the lowest power sleep-mode. From the intermediate power-modes, the 3-Parameter model selects the one with the highest power consumption to derive the value for $P_{active-idle}$. Therefore, unless an idle period is very short, the 3-Parameter model stays in a mode with relatively high power consumption.

The only difference between the Coarse-Dempsey model and Dempsey is in the modeling of periods of disk activity. The Coarse-Dempsey model assumes that the disk uses energy at a constant rate when it is satisfying disk requests, regardless of whether it is in the seeking, rotation, reading or writing stage. Dempsey, on the other hand, attempts to separately estimate the energy consumed in each of these stages. As the results show, the two models perform almost equally well on all traces except Trace VIII. Trace VIII is a trace containing only large sequential writes interspersed by idle-periods of 100 ms. It is representative of workloads that may occur more frequently in a different file system (such as an LFS [17]) than those traced in Traces A-F. For this trace, Dempsey is observed to be more accurate than the Coarse-Dempsey model by as much as 5%. Additional experiments with a write-only workload that has little idle time show that the Coarse-Dempsey model can be off by amounts indicative of the power variances shown in Table 1.

Table 12 lists the number of transitions into the

Trace	Actual	2-Parameter		3-Parameter		Coarse-Dempsey		Dempsey	
I	220.1	37.2	(−83.1%)	214.0	(−2.8%)	219.0	(−0.5%)	219.1	(−0.5%)
II	24.4	13.6	(−44.1%)	23.2	(−4.7%)	23.5	(−3.6%)	23.4	(−4.0%)
III	24.6	14.5	(−40.9%)	24.0	(−2.3%)	24.3	(−1.2%)	24.4	(−0.8%)
IV	171.2	42.4	(−75.2%)	160.3	(−6.4%)	170.8	(−0.3%)	170.8	(−0.2%)
V	89.7	31.9	(−64.4%)	84.5	(−5.8%)	87.6	(−2.3%)	87.7	(−2.2%)
VI	22.2	5.8	(−73.7%)	21.2	(−4.3%)	21.7	(−2.4%)	21.6	(−2.5%)
VII	23.8	10.8	(−54.8%)	22.3	(−6.1%)	22.7	(−4.7%)	22.7	(−4.8%)
VIII	8.0	2.3	(−71.3%)	7.2	(−10.0%)	7.2	(−10.0%)	7.4	(−7.5%)

Table 8: Measured and estimated total energy consumption (in Joules) for the synthetic traces executed on the IBM Microdrive. The column labeled "Actual" lists the measured energy consumption. The last four columns, respectively, present the energy consumption estimates from the 2-Parameter model, the 3-Parameter model, the Coarse-Dempsey model and Dempsey. The numbers in parentheses in the last four columns are the percentage difference between the estimated and the actual values.

Trace	Actual	2-Parameter		3-Parameter		Coarse-Dempsey		Dempsey	
A	2116.5	963.0	(−54.5%)	2036.9	(−3.8%)	2136.0	(+0.9%)	2137.9	(+1.0%)
B	1897.9	1030.6	(−45.7%)	1840.6	(−3.0%)	1903.0	(+0.3%)	1902.1	(+0.2%)
C	2462.7	1049.6	(−57.4%)	2370.6	(−3.7%)	2479.4	(+0.7%)	2480.9	(+0.7%)
D	3156.0	1255.9	(−60.2%)	3027.7	(−4.1%)	3152.5	(−0.1%)	3157.3	(+0.0%)
E	2114.3	1022.0	(−51.7%)	2031.4	(−3.9%)	2121.0	(+0.3%)	2122.6	(+0.4%)
F	1801.4	958.4	(−46.8%)	1742.5	(−3.3%)	1813.7	(+0.7%)	1814.4	(+0.7%)

Table 9: Measured and estimated total energy consumption (in Joules) for the real-world traces executed on the IBM Microdrive. The numbers listed have the same meaning as in Table 8.

Trace	Dempsey Simulation Time (s)
A	0.9
B	2.7
C	2.3
D	6.8
E	1.9
F	0.8

Table 13: Time (in seconds) taken by Dempsey to simulate the IBM Microdrive on the real-world traces, each of which is a 4-hour long trace.

sleep-mode predicted by Dempsey and the corresponding numbers from an actual execution of the real-world traces. The "Actual" number of transitions for a given trace is obtained by counting the number of instances in the execution where the response time exceeds a certain threshold. The worst observed error is 3.4% for the IBM Microdrive and 5% for the Toshiba HDD. This shows that Dempsey is able to model the drives' behavior during idle-periods with reasonable accuracy.

Simulation time for Dempsey to simulate the IBM Microdrive on the 4-hour traces are listed in Table 13. These times are measured by running Dempsey on a desktop machine with a 2 GHz Pentium processor. The traces take a total of less than 16 seconds to execute, indicating that Dempsey is able to process traces at the rate of more than 8000 disk-requests per second. Dempsey's memory usage is less than 2 MB. Thus, Dempsey has the potential of being used as an efficient and accurate disk-power modeling tool.

5 Conclusion

In this paper, Dempsey is demonstrated to be able to model hard-disk energy consumption quite efficiently and accurately. Dempsey includes tools that can extract the required parameters from a given disk automatically. This makes Dempsey relatively general-purpose.

Dempsey is experimentally validated for two mobile hard disks, namely, the 1 GB IBM Microdrive and the 5 GB Toshiba Type II PC Card HDD. Both synthetic and real-world traces are used for the evaluation. For the IBM Microdrive, Dempsey's worst observed error is an underestimate by 7.5%, while the mean error is only 1.8%. For the Toshiba HDD, the corresponding errors are 6.9% and 3.6% respectively.

Trace	Actual	2-Parameter		3-Parameter		Coarse-Dempsey		Dempsey	
I	368.0	108.6	(−70.5%)	360.1	(−2.1%)	350.5	(−4.8%)	351.1	(−4.6%)
II	40.4	24.2	(−40.1%)	39.6	(−2.0%)	39.9	(−1.2%)	40.3	(−0.2%)
III	41.4	25.6	(−38.2%)	40.5	(−2.2%)	40.8	(−1.4%)	41.3	(−0.2%)
IV	293.5	126.8	(−56.8%)	432.0	(+47.2%)	277.6	(−5.4%)	278.2	(−5.2%)
V	195.0	89.1	(−54.3%)	280.6	(+43.9%)	189.3	(−2.9%)	189.8	(−2.7%)
VI	36.9	14.2	(−61.5%)	35.3	(−4.3%)	35.8	(−3.0%)	35.6	(−3.5%)
VII	39.7	20.1	(−49.4%)	37.4	(−5.8%)	37.7	(−5.0%)	37.9	(−4.5%)
VIII	14.4	7.1	(−50.7%)	12.5	(−13.2%)	12.6	(−12.5%)	13.4	(−6.9%)

Table 10: Measured and estimated total energy consumption (in Joules) for the synthetic traces executed on the Toshiba HDD. The numbers listed have the same meaning as in Table 8.

Trace	Actual	2-Parameter		3-Parameter		Coarse-Dempsey		Dempsey	
A	5808.0	3338.7	(−42.5%)	8614.2	(+48.3%)	5507.9	(−5.2%)	5511.0	(−5.1%)
B	5206.0	3453.1	(−33.7%)	6884.8	(+32.3%)	4992.9	(−4.1%)	4994.6	(−4.1%)
C	6476.9	3556.0	(−45.1%)	9523.5	(+47.0%)	6114.3	(−5.6%)	6115.0	(−5.6%)
D	6992.9	3914.4	(−44.0%)	10059.3	(+43.8%)	6832.6	(−2.3%)	6834.9	(−2.3%)
E	6115.2	3516.5	(−42.5%)	9305.7	(+52.2%)	5934.4	(−3.0%)	5935.8	(−3.0%)
F	5544.0	3392.5	(−38.8%)	8297.2	(+49.6%)	5421.8	(−2.2%)	5421.9	(−2.2%)

Table 11: Measured and estimated total energy consumption (in Joules) for the real-world traces executed on the Toshiba HDD. The numbers listed have the same meaning as in Table 8.

Dempsey uses a fine-grained approach to model the energy consumption of a disk. In particular, Dempsey attempts to accurately estimate the energy consumption of specific disk stages, namely, seeking, rotation, reading, writing and idle-periods. Dempsey is also compared against several other models, which model disk power consumption in less detail than Dempsey. The results show that accurate modeling of disk behavior during idle periods is critical to the accuracy of any power model. Accurate modeling of periods of activity is also shown to be important, although to a smaller extent.

Acknowledgments

We would like to thank our shepherd Erik Riedel and other FAST referees for their excellent suggestions and many pointers to existing related work. We would also like to thank Russ Joseph for putting together the PC Card sleeve that we could use for our measurements.

References

[1] Adaptive power management for mobile hard drives. Tech. rep., Storage Systems Division, IBM Corporation, April 1999. Available at: http://www.almaden.ibm.com/almaden/pbwhitepaper.pdf.

[2] Aboutabl, M., Agrawala, A. K., and Decotignie, J.-D. Temporally determinate disk access: An experimental approach. In *Measurement and Modeling of Computer Systems* (1998), pp. 280–281.

[3] Douglis, F., Krishnan, P., and Bershad, B. Adaptive disk spin-down policies for mobile computers. In *Proceedings of the Second USENIX Symposium on Mobile and Location Independent Computing* (April 1995), pp. 121–137.

[4] Douglis, F., Krishnan, P., and Marsh, B. Thwarting the power-hungry disk. In *Proceedings of the Winter USENIX Conference* (1994), pp. 292–306.

[5] Farkas, K. I., Flinn, J., Back, G., Grunwald, D., and Anderson, J.-A. M. Quantifying the energy consumption of a pocket computer and a java virtual machine. In *Proc. International Conference on Measurement and Modeling of Computer Systems* (2000), pp. 252–263.

[6] Ganger, G., Worthington, B., and Patt, Y. *The DiskSim Simulation Environment Version 2.0 Reference Manual*, December 1999. Available at: http://www.ece.cmu.edu/ ganger/disksim/.

[7] Golding, R. A., Bosch, P., Staelin, C., Sullivan, T., and Wilkes, J. Idleness is not sloth. In *Proceedings of the Winter USENIX Conference* (1995), pp. 201–212.

Trace	IBM Microdrive		Toshiba HDD	
	Actual	Dempsey	Actual	Dempsey
A	742	745 (+0.4%)	187	189 (+1.1%)
B	424	428 (+0.9%)	130	134 (+3.1%)
C	757	774 (+2.2%)	220	231 (+5.0%)
D	783	810 (+3.4%)	158	163 (+3.2%)
E	655	661 (+0.9%)	330	344 (+4.2%)
F	529	532 (+0.6%)	318	325 (+2.2%)

Table 12: Number of sleeps. The columns labeled "Actual" give the number of sleeps in a real execution of the trace, while the columns labeled "Dempsey" list the number of sleeps estimated by Dempsey. The numbers in parentheses are the percentage difference between the estimated and the actual values.

[8] Greenawalt, P. Modeling power management for hard disks. In *Proceedings of the Symposium on Modeling and Simulation of Computer Telecommunication Systems* (1994), pp. 62–66.

[9] Heath, T., Pinheiro, E., Hom, J., Kremer, U., and Bianchini, R. Application transformations for energy and performance-aware device management. In *Proceedings of the 11th International Conference on Parallel Architectures and Compilation Techniques* (Sept. 2002).

[10] Helmbold, D. P., Long, D. D. E., Sconyers, T. L., and Sherrod, B. Adaptive disk spin-down for mobile computers. *Mobile Networks and Applications 5*, 4 (2000), 285–297.

[11] Joseph, R., and Martonosi, M. Run-time power estimation in high-performance microprocessors. In *Proceedings of the International Symposium on Low-Power Electronics and Design* (Aug. 2001).

[12] Kotz, D., Toh, S. B., and Radhakrishnan, S. A detailed simulation model of the HP 97560 disk drive. Tech. Rep. PCS-TR94-220, Dept. of Computer Science, Dartmouth College, July 1994.

[13] Krishnan, P., Long, P. M., and Vitter, J. S. Adaptive disk spindown via optimal rent-to-buy in probabilistic environments. In *Proceedings of the Twelfth International Conference on Machine Learning* (1995), pp. 322–330.

[14] Li, K., Kumpf, R., Horton, P., and Anderson, T. E. A quantitative analysis of disk drive power management in portable computers. In *Proc. of Winter USENIX Conference* (January 1994), pp. 279–292.

[15] Muller, K., and Pasquale, J. A high performance multi-structured file system design. In *Proceedings of the 13th ACM Symposium on Operating System Principles* (1991), pp. 56–67.

[16] Papathanasiou, A. E., and Scott, M. L. Increasing disk burstiness for energy efficiency. Tech. Rep. 792, University of Rochester, November 2002.

[17] Rosenblum, M., and Ousterhout, J. K. The design and implementation of a log-structured file system. *ACM Transactions on Computer Systems 10*, 1 (1992), 26–52.

[18] Ruemmler, C., and Wilkes, J. UNIX disk access patterns. In *Proc. of the Winter USENIX Conference* (San Diego, CA, Jan. 1993), Usenix Association, pp. 405–420.

[19] Ruemmler, C., and Wilkes, J. An Introduction to Disk Drive Modeling. *IEEE Computer 27*, 3 (March 1994), 17–29.

[20] Schindler, J., and Ganger, G. Automated disk drive characterization. Tech. Rep. CMU-CS-99-176, School of Computer Science, Carnegie Mellon University, December 1999.

[21] Talagala, N., Arpaci-Dusseau, R. H., and Patterson, D. Microbenchmark-based extraction of local and global disk characteristics. Tech. Rep. CSD-99-1063, University of California, Berkeley, 1999.

[22] Thekkath, C. A., Wilkes, J., and Lazowska, E. D. Techniques for file system simulation. *Software - Practice and Experience 24*, 11 (1994), 981–999.

[23] Weissel, A., Beutel, B., and Bellosa, F. Cooperative I/O: A novel I/O semantics for energy-aware applications. In *Proceedings of the Fifth Symposium on Operating Systems Design and Implementation* (Dec. 2002), pp. 117–129.

[24] Worthington, B. L., Ganger, G. R., Patt, Y. N., and Wilkes, J. On-line extraction of SCSI disk drive parameters. In *Proceedings of the ACM Sigmetrics Conference* (1995), pp. 146–156.

[25] Zeng, H., Fan, X., Ellis, C., Lebeck, A., and Vahdat, A. ECOSystem: Managing energy as a first class operating system resource. In *Tenth International Conference on Architectural Support for Programming Languages and Operating Systems* (Oct. 2002).

[26] Zheng, F., Garg, N., Sobti, S., Zhang, C., Joseph, R., Krishnamurthy, A., and Wang, R. Considering the energy consumption of mobile storage alternatives. *Submitted for publication, 2002*.