Markov Model Based Disk Power Management for Data Intensive Workloads

Rajat Garg,* Seung Woo Son,† Mahmut Kandemir,* Padma Raghavan,* Ramya Prabhakar*

*Department of CSE
Pennsylvania State University
University Park, PA 16802
Email: {rgarg, kandemir, raghavan, rap244}@cse.psu.edu

†Mathematics and Computer Science Division
Argonne National Laboratory
Argonne, IL 60439
Email: sson@mcs.anl.gov

Abstract—**In order to meet the increasing demands of present and upcoming data-intensive computer applications, there has been a major shift in the disk subsystem, which now consists of more disks with higher storage capacities and higher rotational speeds. These have made the disk subsystem a major consumer of power, making disk power management an important issue. People have considered the option of spinning down the disk during periods of idleness or serving the requests at lower rotational speeds when performance is not an issue. Accurately predicting future disk idle periods is crucial to such schemes. This paper presents a novel disk-idleness prediction mechanism based on Markov models and explains how this mechanism can be used in conjunction with a three-speed disk. Our experimental evaluation using a diverse set of workloads indicates that (i) prediction accuracies achieved by the proposed scheme are very good (87.5% on average); (ii) it generates significant energy savings over the traditional power-saving method of spinning down the disk when idle (35.5% on average); (iii) it performs better than a previously proposed multi-speed disk management scheme (19% on average); and (iv) the performance penalty is negligible (less than 1% on average). Overall, our implementation and experimental evaluation using both synthetic disk traces and traces extracted from real applications demonstrate the feasibility of a Markov-model-based approach to saving disk power.**

I. Introduction

It is well understood that reducing the energy requirements of portable devices is important to prolong battery life. But when it comes to large storage systems, making them bigger and increasingly powerful has been the priority, in order to attain the demanded availability and performance. Processors have become extremely powerful, making them more data hungry, and so have the data storage needs, leading to a tremendous growth in the energy consumption of present data centers [3]. In a typical data center, storage system contributes to more than 25% of total power consumption [31]. Apart from the energy consumed for disk operations, cooling costs are also a major concern for this high-density equipment [10]. In fact, the costs have already become the second largest contributor to data center total cost of ownership (TCO) [11]. High density racks and blade servers help reduce total power consumption, but their power density levels exceed the limits of many facilities.

Increasing the number of disks, apart from increasing the total storage space, also helps improve the performance, as data distributed across the disks can now be accessed simultaneously [4]. The reason for the rise in energy consumption is the way disks operate. Disks are made to service the requests at their maximum speeds. Normally, they continue spinning at their maximum rotational speed even if they are *not* servicing any request and hence contribute to the wastage of energy. A direct approach to reducing this energy wastage is to shut down all those components that are not doing any useful work at the moment. Much research has been done to obtain gains from this approach. Two important issues arise in this context:

- How accurately can we predict the occurrence of idle times?
- What would be the energy/performance tradeoffs if we decide to shut down (spin down in the context of disks)?

Recently, techniques that employ multi-speed disks [20] have also proposed and evaluated. With such techniques, when there is a slack (allowable increase in latency), the disk is rotated at a lower speed (compared to the maximum speed available), instead of being completely spun down. The choice of speed is based on the length of the available slack. This approach has been shown to be more applicable to high-performance scientific and data-intensive workloads where disk idle periods are typically small but numerous [8], [29]. While the main problem with spinning-down techniques is that they may not be applicable to short idle times; the problem with multi-speed disks is the large performance penalty incurred if disk idle and active periods are not predicted accurately. Focusing on a three-speed disk, in this paper we propose and experimentally evaluate a novel Markov chain [27] based disk power reduction scheme. Our main contributions can be summarized as follows:

- A Markov model to help disk power management. The rationale behind using a Markov model is that disk access patterns exhibit a repetitive behavior and can therefore be captured by using such a model. First, building a Markov model for a given disk system is presented, followed by the mechanism for making use of this model.
- A three-speed disk model. The need to have such a disk is discussed in detail and its benefits are assessed.
- A prediction scheme. We introduce a scheme that uses the information from the Markov model of the disk system to predict future states of the system (in terms of active and idle periods of disks).
- A runtime approach. This approach uses the Markov model, the three-speed disk model, and the prediction scheme for achieving disk energy savings. The approach decides what needs to be done and when.

Our experiments with various workloads, which include both synthetic traces and traces extracted from real applications, indicate that the Markov model effectively captures the behavior of the disk system. The success of our proposed scheme can be attributed to being able to predict the future states of the system. Since, our approach is *proactive*, meaning the idle periods are predicted in advance, the opportunities to save power are rarely missed (on mispredictions) and also are fully utilized, spinning down to the lowest power mode with little impact on performance. The use of the three-speed disk helps make the most of long idle times by entering the standby mode, additionally giving the flexibility to save energy even when idle times (spin-down to a lower speed) are short.

The rest of this paper is organized as follows. Section II discusses the related work. In Section III, we give a brief introduction to Markov modeling and discuss how it is used by our scheme. Section IV discusses the schemes that can be employed for predicting the next state of the disk system. Section V introduces the concept of a three-speed disk. In Section VI, the algorithm for our disk power management scheme is described in detail. Section VII provides the experimental setup and results, followed by our concluding remarks in Section VIII.

II. Related Work

There has been an extensive body of work on power minimization in the context of both low-end embedded or portable devices and high-performance machines. Because of space concerns, however, in this section we restrict ourselves to the work performed on disk power minimization. Much of this work so far has made use of the usual two-speed (full-speed and standby) disks with prediction strategies to initiate disk spin-up and spin-down during idle times [17], [6]. Some of these schemes predict the next occurrence of the idle time and trigger a spin-down in advance (proactive), whereas others wait a certain length of time before entering the low power mode. Most of the prediction schemes use information gathered from the system resources to understand the workload behavior. For example, the number of the requests in the request queue [8] or the ghost buffer [1] (which records replaced memory pages as if they are stored in additional physical memory) can be used to guide the spin-up and spin-down policies. However, they provide a limited view of the workload evolution, and hence, are not very effective in predicting future idle times. There also exist schemes that try to increase the disk idle periods, thereby making spin-downs and spin-ups more profitable [35], [24], [2], [25], either by making use of I/O prefetching and caching or resizing the storage cache. These strategies could in fact work along with our scheme to improve the current savings. More recently, the concept of DRPM [8] was introduced, which tries to provide more flexibility of operation, making it possible to exploit even small idle periods without significantly hurting performance.

Once the major consumers of power were identified [19], researchers started working to conserve energy in network servers [3], [25] and in systems employing disk arrays [14], [34], [5], [31]. Colarelli and Grunwald [5] introduce MAID, where additional always-on cache disks are employed for a storage archiving scenario that will be useful only if the workload has enough data reuse [32]. While Carrera et al. [3] evaluate some of the disk power management schemes and point out the importance of using multi-speed disks for saving power in I/O intensive workloads, Zhu et al. [34] propose a technique that brings together all power-saving strategies under a combined scheme called Hibernator, by making use of data migration and multi-speed disks. Data migration helps in creating idle periods across some disks, but overhead is involved in determining the new data location. Also, the prediction scheme that determines the optimal speed of operation for a disk in [34] is coarse grained, thus missing some potential opportunities for saving power.

Markov modeling [27] has been used in the past for predicting the I/O access patterns [22], [18] to guide caching [15] and prefetching [12] policies in order to improve performance. There have been attempts [28], [26], [23] to use Markov model for disk power management. However, these efforts considered entirely different schemes and execution environments from our model. Specifically, most of them have focused on optimizing for portable devices or single-disk systems, and they consider it a policy optimization problem. Also, they work at the granularity of requests, whereas we use a sampling time window for power management. To the best of our knowledge, this paper is the first study that employs Markov modeling in the proposed format for reducing power consumption of high-performance disks used in data-intensive computing.

III. Markov Model for Disk Idleness Prediction

We model the disk state transitions using Markov modeling. A Markov model for a system can be completely specified by the total number of states n and the transition probability matrix P [13]. The number of potential states for a N-disk system is given by 2^N (here n=2^N). This is because a disk is either busy (*ON*, represented by 1) servicing a request, or idle (*OFF*, represented by 0). Given the present state and all past states, if the future state of the system depends only on the present state, the system is said to have the *Markov property*. The transition probability matrix is a square matrix of size $n \times n$, where n is the number of states in the system. Values contained in the matrix are probabilities, where P_{ij} (located in i^{th} row and j^{th} column) is the probability of transitioning from state i to state j.

In the context of disk power minimization, one can build a transition probability matrix by *sampling* the state of the disk system at regular intervals (states representing disk being accessed or not accessed). We sample all the disks at runtime, noting whether the disk was accessed during the last sampling period. If it was, then the bit is set for the corresponding disk; otherwise it is reset. Even if a disk access starts toward the end of the sampling period (thus leaving the system in a state of transition at the sampling point), we conservatively assume that the disk was *ON* during the whole sampling period. However, this assumption will not be made while calculating the energy for the base case. We represent the state of the system as a *bit vector*. For an eight-disk system, it will be an eight-bit vector represented as $D_1D_2D_3D_4D_5D_6D_7D_8$ (D_i stands for the i^{th} disk in the disk subsystem) and an example state would be 11001111, which indicates that except disks D_3 and D_4, all others were accessed. The transition probability matrix is built and updated during runtime with the help of these samples. There is a *warm-up period* (explained in Section VI), during which the workload characteristics are monitored to help mature the matrix, making it suitable for making predictions on future states (*ON/OFF*) of disks in the system. The probability matrix is updated at regular intervals by including the most recent set of samples. Because of this regular update on the probability matrix, our scheme is able to keep the up-to-date state of changing or mixed workloads. Note that both sampling frequency for the disk subsystem and the update frequency for the probability matrix have to be chosen carefully. We later study in Section VI how crucial is the value of the sampling period.

IV. Prediction Schemes

Transition probability matrix by itself is of no use as far as power reduction is concerned. There is need for a prediction algorithm that predicts the next state for the system by using the information maintained by the probability matrix. We can evaluate the accuracy of a given prediction algorithm by comparing the percentage of matches between the *actual* and *predicted states*. Below, we describe four prediction schemes evaluated in this work. These schemes are *1-step lookahead schemes,* meaning that only the state that directly follows the present state is predicted and none that may happen after this predicted state. We note that predicting the next state from the current state requires indexing into an appropriate row of the probability matrix. This row is determined by the current actual state of the system. Remember that the row and column number of the matrix represent the states and the matrix itself consists of transition probabilities.

- *ORing* (conservative): After indexing into the correct row, OR all the states (recall that state is represented as a bit vector) with transition probabilities greater than a certain probability (0.05 for our case) to get the next state prediction. The rationale behind this scheme is to never predict an idleness if the probability of the disk being *ON* in the next state is greater than some minimum. This scheme tends to produce an *ON* prediction most of the time, not usually giving a performance penalty but providing little power saving opportunities.
- *Most-probable* (aggressive): After indexing into the correct row, predict the next state based on the highest transition probability from the current state. Since we are just selecting the maximum value in the row, it does not necessarily have to be a large value. For example, it may be 0.05 and still be the maximum if other values in the row are all individually less than 0.05 (but they all add up to 0.95). As a result, this scheme might predict an *OFF* even on a value of 0.05. This scheme does produce good energy savings, but it may also lead to a performance penalty, resulting in spin-downs even when not desirable.
- *Last-state* (does not use the probability matrix): The next predicted state will be the last known state of the system. This is the value we used in all other schemes for indexing into the appropriate row (the current actual state). The success of this scheme is based on the assumption that the system possesses some inertia and hence will continue to remain in its present state for some time. The duration of this period is the crucial factor in the success or failure of the scheme. When the sampling period is kept small, the scheme is bound to give good results. We included this scheme in our evaluations to provide us with a baseline. We note that this scheme also inherently makes use of the Markov property by considering only the last state for future predictions.
- *Summing* (the scheme defended in this paper): In this scheme, after indexing into the correct row, we sum all probabilities leading to a 0 (*OFF* state). This is done for each disk separately to obtain its next state. If probability of transitioning to 0 (denoted by P_0) is greater than certain threshold, then we decide to turn the disk *OFF* else it is kept *ON*. Note that this scheme is slightly modified when used for disks with more than three levels, such that the threshold value changes to a range defined for each speed level.

An arbitrary row chosen from the transition probability matrix of our system is shown in Figure 1. This row contains eight entries (for a three-disk system, the number of possible states is 2^3), each entry being a probability for a three-disk system ($D_1D_2D_3$). States are represented as a bit vector with the leftmost bit for the first disk (D_1). Figures 1(a) and 1(b) show the results obtained using the ORing scheme and the Most-probable scheme, respectively. Figure 1(c), on the other hand shows how the Last-state scheme predicts the next state for D_1. Figure 1(d) depicts the computation of P_0 (probability of transitioning to 0), which if greater than, for example, 0.7, will give an *OFF* prediction. How we decide this threshold value is discussed later in Section VI. This scheme (Summing) is expected to give good energy savings without hurting the performance. Results of prediction accuracies obtained with these schemes are discussed in Section VII-B.

V. Three-Speed Disk

In this section, we describe *three-speed disk* that will be used for evaluating our power management scheme. The conventional two-speed disk either runs at the maximum speed or stays in the standby (spin-down) mode where it does not spin at all. The constraint of operating in one of these two modes does not give the flexibility of transitioning to a lower-power mode when the duration of the idle time is less than the break-even time.[1] Since we sample the disk system without looking at the actual start times of idle periods, we might miss some of these idle time opportunities. We also use a prediction scheme to guess the idle times that were captured during our sampling. Therefore, using a conventional (two-speed) disk would not give us much opportunity to save disk energy most of the time. Thus, the motivation for using a three-speed disk is to have the ability to capitalize on all the idle time opportunities that we are able to predict and to have enough flexibility to save energy even when the disk idle times are not long enough for the two-speed disk. We note that, when we refer to saving energy, minimizing the performance penalty automatically goes along with it. The flexibility with the three-speed disk comes from the intermediate level of operation, where we spin the disk at half of its maximum speed. A request when serviced at the intermediate speed almost doubles the service time but reduces the energy consumption by a factor of four [8]. The specifications of the three-speed disk along with the disk model we employ are provided in Figure 2. State transition times and energies are based on the linear power model given in [8], and the disk specifications have been extended for an IBM hard-disk [9]. We also note that disks with such multi-speed capabilities, such as Western Digital Caviar GP [20] and Sony multi-mode disk [21], are now

[1] Break-even time is the minimum amount of idle time for which spinning down a disk brings some energy benefits without increasing original execution latency [16].

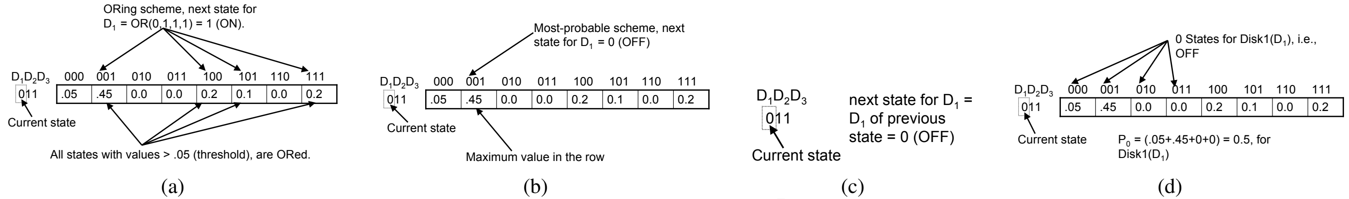

ORing scheme, next state for
D_1 = OR(0,1,1,1) = 1 (ON).
$D_1D_2D_3$
000
001
010
011
100
101
110
111
011
.05
.45
0.0
0.0
0.2
0.1
0.0
0.2
Current state
All states with values > .05 (threshold), are ORed.
(a)
Most-probable scheme, next
state for D_1 = 0 (OFF)
Current state
Maximum value in the row
(b)
next state for D_1 =
D_1 of previous
state = 0 (OFF)
Current state
(c)
0 States for Disk1(D_1), i.e.,
OFF
Current state
P_0 = (.05+.45+0+0) = 0.5, for
Disk1(D_1)
(d)

Fig. 1. Example showing the outcome of predictions with different schemes specific to D_1: (a) ORing, (b) Most-probable, (c) Last-state, and (d) Summing. Note that our defended scheme (Summing) is different from the ORing and Most-probable schemes, and might as well transition the system to a state which has zero probability in the probability matrix.

TABLE I
P_0 CORRESPONDING TO DISK SPEED LEVELS IN A FIVE-SPEED DISK.

Speed (RPM)	P_0 Range
15000	$0.00 < P_0 \leq 0.30$
11000	$0.30 < P_0 \leq 0.50$
7000	$0.50 < P_0 \leq 0.70$
3000	$0.70 < P_0 \leq 0.85$
0	$0.85 < P_0 \leq 1.0$

commercially available in the market, though they are not server-class disks.

VI. ALGORITHM

With the Markov model representing the disk state transitions and accompanied by a prediction scheme that helps predict the next state of the disk, there is a need to have an overall *control strategy* that can make high-level decisions for power management of an I/O subsystem consisting of the proposed three-speed disks. This requires making two important decisions:

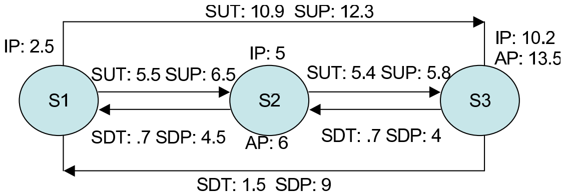
- When can a disk be spun down (should try to maximize the energy savings but it does not matter if we miss some opportunities)?
- When should a disk be spun up (should not miss to spin-up when required)?

Depending on how aggressively one makes these decisions, it can result in different energy savings and performance degradations. To make a decision for the next state, we look at the probability P_0 (probability of transitioning to a 0 state) for each disk. In order for this algorithm to work for an n-speed disk, one can set a threshold for each speed level. Essentially, as the value of P_0 decreases, the disk's operating speed should increase. We choose a threshold value of 0.7 for P_0 in our three-speed disk. In a multi-speed disk scenario, on the other hand, this threshold will be a range and not a value. But, the way we use our three-speed disk makes this modification feasible. As an example, Table 1 lists sample threshold values (as a range) for P_0 corresponding to each speed level in a five-speed disk.

We emphasize that increasing the number of operating speeds (e.g., moving from a three-speed disk to a five-speed disk) does not necessarily mean that we can save more energy. This can be seen in a manner similar to when TPM (traditional power management, which spins down the disk after a certain period of idleness) saves more energy than DRPM in the case of very long idle periods, since it can turn off the disk completely, whereas the DRPM scheme will spin down only to a nominal speed. Similarly, the three-speed disk provides enough flexibility to exploit small idle periods and also the ability to save maximum energy when possible. We note that as the idle periods grow smaller, opportunities to save power become meagre and risky. For a three-speed disk, one should not decide to spin up if spin-up time plus the request service time is more than the service time at the current disk speed. Also, One should spin down only if the idle energy consumed in the current state is more than the sum of the energy spent in spinning down and the idle energy in the lower speed state.

All the disks start off from a *normal state* where the three-speed disk is in its intermediate speed level. Once the transition probability matrix matures, we start making predictions about the future disk states. A prediction *ON* will spin up the disk by one level from its current state. A prediction *OFF* will necessarily spin down the disk to its lowest speed. The disk does not wait for a prediction to transition to the *normal state* if no disk requests were waiting. All these decisions help bring down the power consumption while minimizing the performance degradation. In the following paragraphs, we discuss how the values of various parameters employed in our approach affect the behavior of our proposed scheme.

- *Warm-up Period*: This is the period of time spent before building the initial transition probability matrix. This is an important step in getting started with making good predictions about disk accesses. The transition probability matrix built during the warm-up period will represent more of the transient behavior of disk accesses, but it eventually adapts itself to the changing workload during execution because of the regular updating of the matrix. Note that while updating the matrix, we give lower weight to the older values of the probability matrix. Deciding the right value for the warm-up period is a tradeoff between the accuracy of prediction (large value) vs the time of wait (small value) before the predictions begin. Instead of operating in either of these extremes, we can keep the warm-up period moderately short to obtain the best of energy savings and prediction accuracy. In our baseline implementation, we set it to the time taken to gather 50 samples, a value determined based on some preliminary experiments.
- *Threshold Probability*: This threshold value is used to decide which state our disk can transition to by comparing P_0 with this value. If we want to be aggressive and save more energy without caring much about the performance, then we can set it to a low value (e.g., 0.4). On the other hand, if we want to be conservative, then, say, 0.9 will be a good choice. It affects directly the prediction accuracy, which in turn can hurt both energy savings and performance. For our three-speed disk implementation, we chose this threshold to be 0.7, again based on some preliminary experiments.
- *Sampling Period*: The value of this parameter is crucial to the success of our prediction based scheme. It affects



(a)

IBM Ultrastar 36Z15	
Individual Disk Capacity	18.4 GB
Maximum Disk Rotation Speed (S3)	15000 RPM
Intermediate Speed Level (S2)	7000 RPM
Minimum Disk Speed (S1)	0 RPM

(b)

Fig. 2. Three-speed disk. (a) State model. SUT: Spin-up Time; SDT: Spin-down Time; SUP: Spin-up Power; SDP: Spin-down Power; AP: Active Power; and IP: Idle Power. Time is in seconds, and power is in watts. (b) Specification for the three-speed disk model.

the overhead involved in the scheme, the closeness with which the transition matrix represents the workload, and the energy savings achieved. If it is chosen to be very small, the frequency of state predictions and matrix updates increases. Depending on the disk state transition times and the energy consumed during transitions, a small sampling period may or may not be beneficial. On other hand, making this period too large can lead to missing some energy saving opportunities, specifically, when the idle time is greater than the break-even time but smaller than the sampling period (there was a short duration of disk access). The length of sampling-period used in our default implementation is 12.5 seconds for a simple (two-speed) disk, 7 seconds for a three-speed disk, and 4 seconds for a five-speed disk.

Two overheads are associated with our scheme: updating probability matrix and prediction. Since each prediction scheme uses a simple operation (e.g., bitwise-OR or summation), the prediction overhead is negligible. Updating the probability matrix might have some overheads depending on the size of matrix size. In an eight-disk system, the matrix size will be 256 (2^8) by 256. Since this operation can also be done by using simple loop and the update frequency is at least tens of seconds, we believe that the overhead associated with updating matrix is also negligible.

In our experiments, we also vary the default warm-up period, threshold, and sampling period values and conduct a sensitivity analysis.

VII. Experimental Evaluation

In this section, we first introduce our experimental setup (Section VII-A) and then present the results from our experiments (Section VII-B).

A. Setup

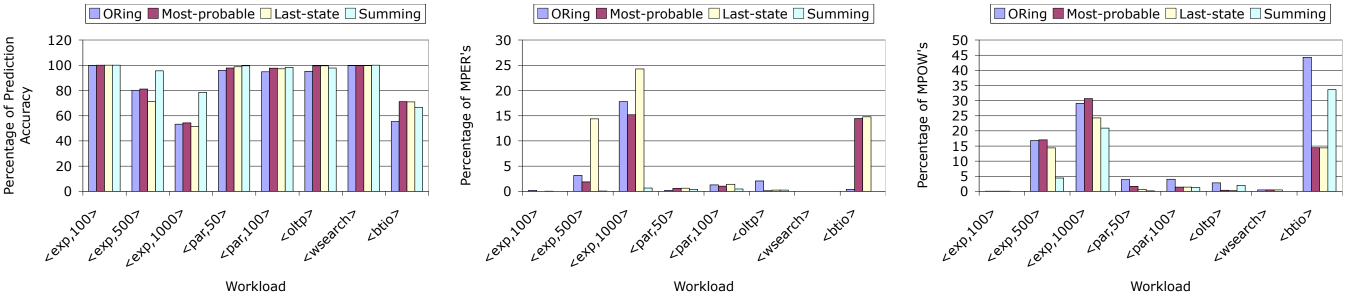
DiskSim [7] was used to simulate the behavior of our disk subsystem and to measure the benefits brought by our scheme. DiskSim is an accurate, highly configurable disk system simulator to support research into various aspects of storage systems. DiskSim is a trace-driven simulator, and we performed one simulation per each workload. Our simulated system has 8 disks; the specifications for the disk were provided earlier in Figure 2. We augmented DiskSim to help us carry out the experiments for various prediction algorithms discussed above to analyze how good they work in saving energy. As the simulation runs, this augmented version of DiskSim checks the state of the disk system at regular intervals. This is referred to as *sampling the system.*

DiskSim provides a synthetic workload generator used to generate the workloads with desired characteristics. Some characteristics common to all workloads are given in Table II. We concentrated mainly on workloads with small inter-arrival times where TPM and other older techniques have failed to save energy efficiently (that is, the high-performance workloads that exhibit short disk idle periods) and results for DRPM [8] could be compared. For the synthetic disk traces, we used two types of workloads:

- Type 1: Inter-arrival times were exponentially distributed, and
- Type 2: Inter-arrival times followed the Pareto distribution.

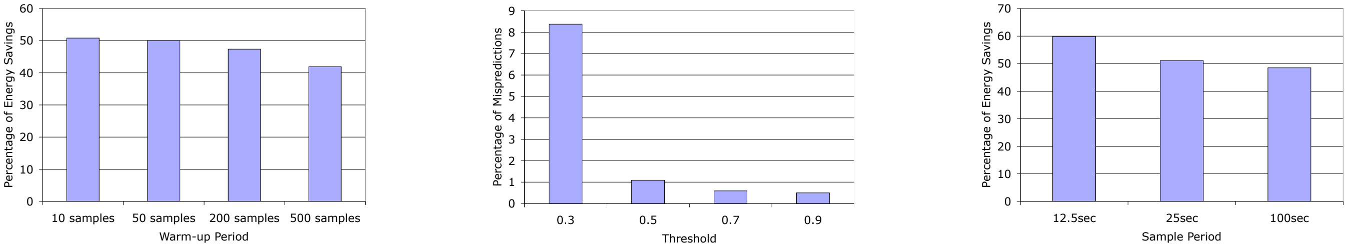
Type 1 workload is represented as $< exp, t >$, where t is the mean inter-arrival time in milliseconds. This type of workload models a purely Poisson process, with arrival traffic showing some kind of regularity. Type 2 workload is represented in a similar fashion as $< par, t >$, with t having the same meaning as before. This workload offers more burstiness in the traffic behavior, meaning that there exists a group of requests clustered close to each other at some places. We used synthetic workloads to show that our scheme is well adapted to different type of inter-arrival times. As far as disk power management is concerned, inter-arrival times matter most because they will eventually affect the length of disk idle periods. Thus, these two types of workloads do offer a good experimental testspace. The original version of DiskSim does not support generation of Pareto workloads. Thus, as a part of our work, it was also augmented to generate such a workload. With these two types of workloads, we vary the mean arrival times of requests, which affects the length of the idle periods (the higher the value of t, the greater the idleness). Table 2 summarizes some default characteristics of the synthetic workloads for the request distribution across the disks.

In addition to our experiments with these synthetic traces, we performed experiments with traces extracted from real applications. These applications are *parallel* in that the number of clients issuing the requests for our 8-disk system are more than one. More specifically, the number of clients range from one to sixteen. One of the workloads is a trace from an online transaction processing application (OLTP); the other trace is gathered from a popular Web search engine. OLTP traces [30] are characterized by frequent insert/updates. The web search trace we use [30] captures the I/O traces of a system that processes web search queries. Both of these traces are obtained from a publicly available repository [30]. The I/O accesses exhibited by these applications are small, numerous, and concurrent. The results with the OLTP trace are indicated with $< oltp >$, whereas those with the search engine trace



(a) (b) (c)

Fig. 3. (a) Prediction accuracies with different schemes. (b) Contribution of mispredictions leading to performance loss (MPER). (c) Contribution of mispredictions leading to power loss (MPOW).



(a) (b) (c)

Fig. 4. Effect of changing the important parameters: (a) warm-up period, (b) threshold, and (c) sample period.

TABLE II
DEFAULT SYSTEM PARAMETERS.

Parameters	Values
Request Number	100000
Number of Disks	8
Disk Size	18 GB
Sequential Access Probability	0.1
Local Access Probability	0.2
Read Access Probability	0.6
Maximum Local Distance	100 blocks

are represented by using $< wsearch >$. We also tested our scheme with a trace from a scientific application called BTIO, which is a disk-based version of a flow-solver program from the NAS Parallel Benchmarks [33]. The main operation in the code is periodic writes performed by all processors to a multidimensional array stored in a file. This trace is represented as $< btio >$. The number of clients for this type of workload was kept as sixteen. Note that the energy-saving opportunities in all these traces depend on the length of idle periods between various accesses. Specifically, the workload from the search engine was found to contain less than 2 percent overall I/O system idle time. Our experiments were carried out with these diverse (synthetic plus real) workloads to obtain statistics for the following.

- Total energy consumed by disk system when no optimization is performed (E_{tot})
- Percentage of energy savings with different power management schemes (Sav)
- Performance penalty
- Accuracy of various prediction schemes
- Effect of changing the important parameters employed in our scheme

We also conducted experiments with a five-speed disk based execution scenario in order to evaluate the effect of increasing the number of speed levels in a disk. The energy savings produced with the five-speed disk are compared against those achieved with the three-speed disk and TPM. Note that all the energy saving results presented here consider the savings across all disks in our 8-disk system. The energy spent in transitioning the disk to a different state was considered in all our calculations. In the context of this work, the performance penalty of a disk system is defined as the percentage increase in the execution time for the given workload. More specifically, if the last request of the workload was serviced at time T when no energy optimization was applied and now with the optimizations it gets serviced at time $(T + x)$, the percentage performance penalty is calculated as $(x/T) * 100$. The results presented below include *all the overheads* incurred by our scheme.

B. Results

We conducted experiments to test and validate the three-speed disk model along with the prediction schemes and verify the usefulness of Markov modeling. First, the prediction algorithms described earlier were evaluated for their prediction accuracies. Specifically, we tested each prediction scheme on all the workload types we have. Figure 3(a) shows the prediction accuracies of the four schemes discussed earlier. The average prediction accuracies (when all workloads are considered) are 86.0%, 84.2%, 87.6%, and 92.0% for the Last-state, ORing, Most-probable, and Summing schemes, respectively. Since inaccurate prediction of disk idleness can be determined from a performance perspective, we consider 90% or higher as a good accuracy, and our prediction accuracies are in this range. The total mispredictions (TMPs) can be broken down into two types:

- Mispredictions leading to performance loss (MPER), and
- Mispredictions leading to energy loss (MPOW).

MPER happens when one predicts a spin-down for the disk but the disk was actually accessed and hence we incur spin-

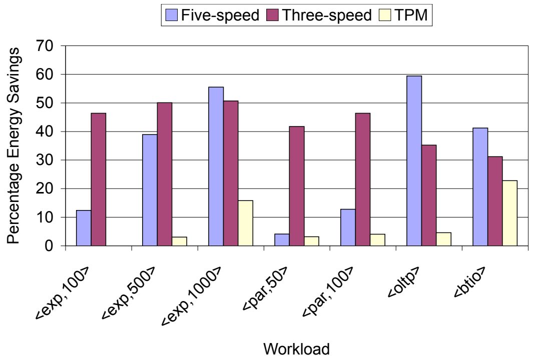


Fig. 5. Comparison of energy savings achieved by five- and three-speed disk based systems relative to the base case, namely, TPM.

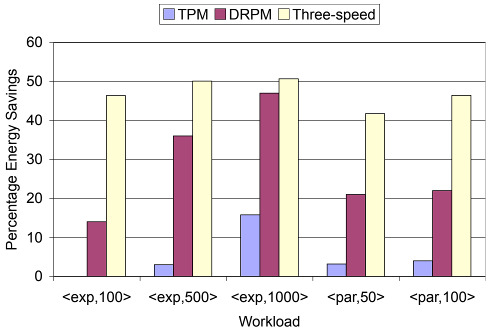


Fig. 6. Comparison of energy savings with different schemes including results with DRPM. Results for DRPM were obtained from [12], where no tests were performed with real traces.

TABLE III
PERCENTAGE OF PERFORMANCE PENALTY.

Workload	Penalty (TPM)	Penalty (Three-Speed Disk)
$< exp, 100 >$	0.0	0.0
$< exp, 500 >$	0.03	0.0
$< exp, 1000 >$	0.015	0.0
$< par, 50 >$	0.0	0.0
$< par, 100 >$	0.0	0.0
$< oltp >$	1.42	1.76
$< btio >$	0.0	0.0

up delays. In comparison, *MPOW* happens when the disk is predicted *ON* but it was never accessed during that period, and consequently, an opportunity to spin-down was missed. We see from Figures 3(b) and 3(c) that the ORing technique gives more mispredictions leading to energy loss, whereas the Last-state technique gives more mispredictions leading to performance loss. Overall, our defended prediction scheme (Summing) performs better in all respects. Although all these schemes do provide a good percentage of correct predictions, the Summing scheme has significantly lower *MPER* value. It is also clear from these results that all the prediction schemes tend to become less accurate as the sampling period increases, specifically the Last-state scheme. In cases where even a slight performance degradation is intolerable, one should try to minimize the percentage of the *MPER* even if, in doing so, we increase the contribution of *MPOW*. Note that a higher *MPOW* value only means that we missed some energy saving opportunities, but a higher $MPER$ value may be intolerable in a high-performance computing environment.

There should be enough samples to build the transition probability matrix initially so it really does reflect the workload characteristics with a reasonable accuracy. Hence we decided to take at least 50 samples to capture the workload behavior. Obviously, the more samples we take, the better our knowledge of the workload. However, this also means we start the energy optimizations late. Figure 4(a) shows that the energy savings decrease when the warm-up period is increased. Figure 4(b) shows the effect of varying the threshold value on the percentage of mispredictions leading to performance loss (*MPER*). Decreasing the threshold value means that we aggressively turn disks *OFF* and therefore increase the chances of mispredictions, which is reflected in Figure 4(b). In Figure 4(c), on the other hand, the effect of increasing the length of the sampling period is shown. The energy savings decrease because we miss some idle time opportunities. We also tested the effectiveness of these prediction schemes using a five-speed disk. The results given in Figure 5 indicate that three-speed disk provides better energy savings in most cases. The response of a five-speed disk to a disk state prediction is more gradual than that of the three-speed disk. The reason is that the five-speed disk slows down the disk speed one step at a time unless a disk experiences big slowdown in the response time. Consequently, it takes more time to transition to a lowest power mode, in turn producing less savings. This one step approach is a bit less aggressive in lowering the disk speed, but it enables us to identify the system state at all times and ensures easy recovery on misprediction (there are forced spin-ups and spin-downs when the actual state is not equal to the current state of the system). Note that one has more flexibility with a five-speed disk when it comes to selecting a speed level, which can be helpful, as is the case for the savings on OLTP and BTIO workloads in Figure 5. Although we achieve better energy savings with a five-speed disk, it also leads to more performance penalty (not shown in results because of lack of space). This can be attributed to the increased overhead of transitioning across different speed levels.

In Figure 6, the energy savings obtained with the three-speed disk supported by our scheme are compared with TPM and DRPM savings. All energy savings are normalized with respect to the base case, where no power saving scheme is employed. The energy consumption evaluated and the power saving results consider the entire disk system. We regenerate the energy savings with DRPM (denoted as $DRPM_{perf}$ in [8]) where it can predict the idle times with full accuracy; consequently, there is no performance loss. Since we are using the same simulation tool for generating the same workload types, it makes sense to compare the results. We see from these results that our scheme provides more energy savings compared to TPM. It also does better than DRPM. The reason can be attributed to the ability of the disk to totally spin down (standby mode) whenever possible, and save energy even when the duration of idle periods is not sufficiently long (spin at an intermediate speed level). Although the opportunity to save energy with these workloads may look meagre, it is the result of using the predictive scheme along with the concept of a multi-speed disk that helps save energy. There is not much performance penalty from TPM as this scheme triggers a shutdown only when the disk has been idle for a long period of time. However, when we use prediction algorithms and perform spin-ups and spin-downs proactively, there is a chance of significant performance penalty. This can be a result of a mispredicted spin-down (MPER) when the disk is being actually accessed. Table 3 shows that, with our scheme, there is very small or no performance penalty with the used traces. Gurumurthi et al. [8] give performance degradation in terms of response times, but does not show the net effect on the total execution time.

VIII. Concluding Remarks

The main contribution of this paper is a novel Markov model based disk idleness prediction scheme that can be used for reducing disk power consumption when used with a three-speed disk. The paper explains in detail why the defended prediction mechanism is better than others and why it saves disk power. To evaluate the effectiveness of our approach, we implemented it using DiskSim and performed experiments with both synthetic traces and real application traces.

Our experimental results show that (i) the prediction accuracies of the proposed scheme are very good (87.5% on average); (ii) it generates significant energy savings over the traditional power saving method of spinning down the disk when idle (35.5% on average); (iii) it performs better than a previously proposed multi-speed disk management scheme (19% on average); and (iv) the performance penalty it brings is negligible (less than 1% on average). Overall, our implementation and experimental evaluation demonstrate the feasibility of a Markov model based approach to saving disk power. Our ongoing work involves integrating this scheme with existing disk power saving strategies and testing them under different workloads. We are also investigating whether high-level (application level) information supplied by programmers can be used for improving our power savings.

Acknowledgement

This work is supported in part by NSF grants #0406340, #0444158, #0621402, #0724599, #0821527, and #0833126 and by the Office of Advanced Scientific Computing Research, Office of Science, U.S. Department of Energy, under Contract DE-AC02-06CH11357.

References

[1] L. Cai and Y.-H. Lu. Joint Power Management of Memory and Disk. In *Proceedings of the Conference on Design, Automation and Test in Europe*, pages 86–91, 2005.

[2] L. Cai and Y.-H. Lu. Power Reduction of Multiple Disks Using Dynamic Cache Resizing and Speed Control. In *Proceedings of the International Symposium on Low Power Electronics and Design*, pages 186–190, 2006.

[3] E. Carrera, E. Pinheiro, and R. Bianchini. Conserving Disk Energy in Network Servers. In *Proceedings of the International Conference on Supercomputing*, pages 86–97, 2003.

[4] P. M. Chen, E. K. Lee, G. A. Gibson, R. H. Katz, and D. A. Patterson. RAID: High-Performance, Reliable Secondary Storage. *ACM Comput. Surv.*, 26(2):145–185, 1994.

[5] D. Colarelli and D. Grunwald. Massive Arrays of Idle Disks for Storage Archives. In *Proceedings of the ACM/IEEE Conference on Supercomputing*, pages 1–11, 2002.

[6] F. Douglis, P. Krishnan, and B. Bershad. Adaptive Disk Spin-Down Policies for Mobile Computers. In *Proceedings of the USENIX Symp. on Mobile and Location-Independent Computing*, pages 121–137, 1995.

[7] G. Ganger, B. Worthington, and Y. Patt. The DiskSim Simulation Environment Version 3.0 Reference Manual. http://www.pdl.cmu.edu/DiskSim/.

[8] S. Gurumurthi, A. Sivasubramaniam, M. Kandemir, and H. Franke. DRPM: Dynamic Speed Control for Power Management in Server Class Disks. In *Proceedings of the International Symposium on Computer Architecture*, pages 169–181, 2003.

[9] IBM. Ultrastar 36Z15 Hard Disk Drive. http://www.hgst.com/hdd/ultra/ul36z15.htm, 2003.

[10] Intel. Addressing Power and Thermal Challenges in the Datacenter. http://download.intel.com/design/servers/technologies/thermal.pdf.

[11] Intel. Increasing Data Center Density While Driving Down Power and Cooling Costs. http://www.intel.com/business/bss/infrastructure/enterprise/power_therm%al.pdf.

[12] D. Joseph and D. Grunwald. Prefetching Using Markov Predictors. *IEEE Trans. Comput.*, 48(2):121–133, 1999.

[13] G. Latouche and V. Ramaswami. *Introduction to Matrix Analytic Methods in Stochastic Modelling*. PH Distributions, 1999.

[14] D. Li and J. Wang. EERAID: Energy Efficient Redundant and Inexpensive Disk Array. In *Proceedings of the 11th Workshop on ACM SIGOPS European Workshop: beyond the PC*, page 29, 2004.

[15] K. Li, W. Qu, H. Shen, D. Wu, and T. Nanya. Two Cache Replacement Algorithms Based on Association Rules and Markov Models. In *Proceedings of the First International Conference on Semantics, Knowledge and Grid*, page 28, 2005.

[16] Y.-H. Lu, E.-Y. Chung, T. Simunic, L. Benini, and G. D. Micheli. Quantitative Comparison of Power Management Algorithms. In *Proceedings of the Conference on Design, Automation and test in Europe*, pages 20–26, 2000.

[17] Y.-H. Lu and G. de Micheli. Adaptive Hard Disk Power Management on Personal Computers. In *Proceedings of the Ninth Great Lakes Symposium on VLSI*, pages 50–53, 1999.

[18] T. M. Madhyastha and D. A. Reed. Input/Output Access Pattern Classification Using Hidden Markov Models. In *Proceedings of the 5th Workshop on I/O in Parallel and Distributed Systems*, pages 57–67, 1997.

[19] Maximum Institution Inc. Power, Heat, and Sledgehammer. http://www.max-t.com/downloads/whitepapersSledgehammerPowerHeat20411.pd%f, 2002.

[20] C. Mellor. Western Digital Launches Power-Efficient Disk Drives. http://www.techworld.com/green-it/news/index.cfm?newsid=10711&email.

[21] K. Okada, N. Kojima, and K. Yamashita. A Novel Drive Architecture of HDD: "Multimode Hard Discdrive". In *Proceedings of the International Conference on Consumer Electronics*, pages 92–93, 2000.

[22] J. Oly and D. A. Reed. Markov Model Prediction of I/O Requests for Scientific Applications. In *Proceedings of the International Conference on Supercomputing*, pages 147–155, 2002.

[23] G. Paleologo, L. Benini, A. Bogliolo, and G. Micheli. Policy Optimization for Dynamic Power Management. In *Proceedings of the Design Automation Conference*, pages 182–187, 1998.

[24] A. E. Papathanasiou and M. L. Scott. Energy Efficient Prefetching and Caching. In *Proceedings of the USENIX Annual Technical Conference*, pages 255–268, 2004.

[25] E. Pinheiro and R. Bianchini. Energy Conservation Techniques for Disk Array-Based Servers. In *Proceedings of the International Conference on Supercomputing*, pages 68–78, 2004.

[26] Q. Qiu and M. Pedram. Dynamic Power Management Based on Continuous-Time Markov Decision Processes. In *Proceedings of the Design Automation Conference*, pages 555–561, 1999.

[27] L. R. Rabiner. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

[28] T. Simunic, L. Benini, P. Glynn, and G. D. Micheli. Dynamic Power Management for Portable Systems. In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, pages 11–19, 2000.

[29] S. W. Son, M. Kandemir, and A. Choudhary. Software-Directed Disk Power Management for Scientific Applications. In *Proceedings of the International Parallel and Distributed Processing Symposium*, page 4.2, 2005.

[30] UMass Trace Repository. http://traces.cs.umass.edu.

[31] C. Weddle, M. Oldham, J. Qian, A.-I. A. Wang, P. Reiher, and G. Kuenning. PARAID: A Gear-Shifting Power-Aware RAID. In *Proceedings of the USENIX Conference on File and Storage Technologies*, pages 245–260, 2007.

[32] M. E. Wolf and M. S. Lam. A Data Locality Optimizing Algorithm. In *Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation*, pages 30–44, 1991.

[33] P. Wong and R. F. V. der Wijngaart. NAS Parallel Benchmarks I/O Version 2.4. Technical Report NAS-03-002, NASA Advanced Supercomputing Division, January 2003.

[34] Q. Zhu, Z. Chen, L. Tan, Y. Zhou, K. Keeton, and J. Wilkes. Hibernator: Helping Disk Arrays Sleep through the Winter. In *Proceedings of the ACM Symposium on Operating Systems Principles*, pages 177–190, 2005.

[35] Q. Zhu, F. M. David, C. F. Devaraj, Z. Li, Y. Zhou, and P. Cao. Reducing Energy Consumption of Disk Storage Using Power-Aware Cache Management. In *Proceedings of the International Symposium on High-Performance Computer Architecture*, pages 118–129, 2004.