2006

eRAID: Conserving Energy in High Performance RAID Systems with Conventional Disks

Dong Li
University of Nebraska - Lincoln, li@cse.unl.edu

Jun Wang
University of Nebraska - Lincoln, wang@cse.unl.edu

University of Nebraska–Lincoln, Computer Science and Engineering
Technical Report TR-UNL-CSE-2006-0001
Issued Jan. 9, 2006

eRAID: Conserving Energy in High-Performance RAID Systems with Conventional Disks

Dong Li and Jun Wang
Computer Science and Engineering
University of Nebraska–Lincoln
Lincoln, NE 66588-0115
{li,wang}@cse.unl.edu

ABSTRACT

Recently energy consumption becomes an ever critical concern for both low-end and high-end storage server and data centers. A majority of existing energy conservation solutions resort to multi-speed disks. However, current server systems are still built with conventional disks.

In this paper, we propose an energy saving policy, eRAID, for conventional disk based RAID-1 systems. eRAID saves energy by spinning down partial or entire mirror disk group with predictable performance degradation. The heart work of eRAID is to develop an accurate dynamic performance control (including disk power management) scheme. To guarantee service quality, the dynamic performance control works for two performance measures — response time and throughput. In addition, a time-series analysis model (ARMA) is used to forecast workload features while queueing network models are adopted to do performance prediction. Experimental results show that eRAID can save up to 32% energy without violating predefined performance degradation constraints.

1. INTRODUCTION

In today's high-performance data-intensive computing systems, the energy consumed by the disk-based storage subsystems may easily surpass the energy consumed by the rest of the computing system [1, 2, 3]. Seeking energy-saving solutions in these environment has attracted more and more research interests. Due to the short I/O request inter-arrival time, disk spin down/up policy for energy saving of single disk is considered infeasible in array-based server environments [4, 5]. A majority of existing energy conservation solutions resort to multi-rotation-rate disks [3, 6, 7, 8, 9]. However, current server systems are still built with conventional disks. It is important to provide new solutions for conventional disks.

The most fundamental method to save energy from a conventional disk is to turn it to standby state when there is no request arriving. Since server side workloads are usually too intensive to supply long idle period for disks to spin down to save energy, these long idle periods have to be artificially created. Without intervening the execution of applications, a possible way to create long idle periods for disks is to unbalance disk workloads.

Currently, there are two approaches to unbalance workloads for conventional disks toward energy-saving: *relocating data* and *redirecting request*. Two representative work for the former approach are Massive Array of Idle Disks (MAID) [10] and Popular Data Concentration (PDC) [2]. The common feature of both policies is to intentionally unbalance disk loads by migrating data across disks according to the changing of data access patterns. However, Pinheiro *et al.* [2] showed that MAID and PDC can conserve energy for conventional disks only when the load on the server is extremely low, though MAID and PDC perform much better for multi-speed disks. That is, except for extremely light loads, it is still difficult to create long idle periods for conventional disks only by migrating data according to the dynamic changing of data access pattern.

The basic idea of the second approach *redirecting request* is to intensionally bypass a data target (e.g. a disk) by redirecting requests to other data target(s). The precondition is that the alternative data target(s) can provide the same information to users. With internal redundant information, RAID systems satisfy this precondition well. We explored how to use this approach in RAID systems in literature [7] with multi-speed disks other than conventional disks. The performance impact of redirecting requests is not deeply studied. To take advantage of such inherent redundancy for energy conservation, a good solution on dynamic performance control and power management is a must. Researchers from Illinois [8] are among the first to develop several control algorithms to ensure performance-guarantee for a multi-speed disk based storage system. But their performance prediction scheme for disks is still at an early stage without taking into account many critical factors, such as the workload dynamics, time-criticality, disk queueing delay and disk array organization.

In this paper, we propose an energy saving policy named eRAID deploying request redirecting for conventional disk based RAID-1 systems. The idea of eRAID is to save energy by adaptively spinning down partial or entire mirror disk group according to the fluctuating workload intensity. To realize the goal, we develop an accurate online performance control scheme for both synchronous and asynchronous workloads. To satisfy service quality, the dynamic performance control works for two performance measures — response time and throughput. We develop a power model for conventional disk based storage array systems. We extend existing queueing network models to design an online performance predictor. Additionally, we employ a time-series analysis model (ARMA) to forecast workload features to ensure high load prediction accuracy.

Experimental results show that eRAID can save up to 32% energy without violating predefined performance degradation constraints. Compared with previous server side energy saving policies [2, 3, 6, 7, 8, 9, 10], eRAID has the following salient advantages: (1) it is a soft solution so that it does not require hardware updates for current storage systems, such as adding extra hardware (e.g. cache disks) or replacing all their conventional disks with multi-speed disks; (2) it is independent with host system and can be easily deployed to standard RAID systems without any change of existing disk array configurations such as data layout, or introducing any data migration overhead; (3) it does tradeoff between energy-saving and performance degradation by taking both performance metrics, average response time and throughput, into consideration.

The rest of this paper is organized as follows. Section 2 describes our motivation. The design of eRAID are introduced in Section 3. Section 4 is the experimental evaluation. Section 5 briefly discusses the extending of eRAID for RAID-5. Related work is discussed in Section 6. Finally, we give the conclusion remarks in Section 7.

2. MOTIVATION

2.1 Limitations of Current Solutions

There are three major problems with current server-side disk energy management policies.

(1) No good solution for conventional disks based server systems: As we mentioned in the Introduction, most current energy saving policies for server side storage systems resort to multi-speed disks [3, 6, 7, 8, 9]. Although there are solutions, such as Massive Array of Idle Disks (MAID) [10] and Popular Data Concentration (PDC) [2], which can be applied to both conventional disks and multi-speed disks, in most cases they may only work well for multi-speed disks.

(2) Single performance measure: Doing trade-off between energy-saving and performance degradation is one of the most important task for disk energy management solutions. The traditional measures of performance of I/O system are throughput(bandwidth) and response time(latency) [11]. According to our knowledge, all the current disk energy management algorithms only take one of them rather than both into consideration for the tradeoff. However, energy-saving policy may degrade both throughput and response time. Here, for average response time, "degradation" means the stretching of response latency. While for throughput, "degradation" means the reduction of service bandwidth. The degradations of these two performance measures are not always proportional. Under different system utilizations, a little longer average response time may result in different amount of throughput reductions. For the same reason, throughput degradation itself can not tell how much the response time is stretched.

(3) No differentiation for workload time criticality: According to time criticality, I/O loads can be roughly divided into two classes, synchronous loads and asynchronous loads. For synchronous load, requests have dependency relationship and must be served according to some order. For asynchronous loads, the arrival of I/O request is not subject to the completion of previous requests. Since synchronous and asynchronous loads have different time criticality, their reactions to performance degradation of storage system are also different. For example: If we treat the I/O system as a black box, the request arrival rate is equal to the request departure rate when the system is stable [11]. Suppose the mean response time is stretched because of using energy saving policy while the system is still in stable state, the throughput may not be affected for asynchronous loads. However, for synchronous load, the request arrival rate may be lowered because the system takes more time to server a request. As a result, the system throughput may be decreased.

2.2 Three Observations

Three observations motivate us to develop an energy-saving policy for conventional disk based RAID-1 systems.

(1) With the help of redundant information, it is possible to generate long idle periods for disks in data-intensive environments. RAID-1, also called mirroring or shadowing, adopts twice as many disks as a non-redundant disk array to maintain 100% redundancy [12]. Read requests can be served by either of the two copies and write requests must be reflected to both copies. Modern RAID systems usually adopt a non-volatile RAM (NVRAM in brief) write-back cache to improve system performance, so the data update can be saved in NVRAM before they are flushed to disks. By redirecting read requests to the primary disks and deferring write update by caching, the idle periods of mirror disks can be arbitrarily stretched. In the rest of this paper, we interchangeably use NVRAM and controller cache.

(2) Serve disks usually have spare service capacity, and the system loads are roughly predictable. Gurumurthi *et al.* [6] show that server disks are idle in most time, and even under heavy workloads, disk utilization in server RAID systems is still less than 35% [13, 7]. A disk can be viewed as a single queueing node system.

When system is in low utilization, the system performance degrades slowly with the increasing of workload intensity. Therefore, redirecting requests of mirror disks to primary disks during light loads does not necessarily degrade much performance. Moveover, the system workloads of data-intensive environments, such as data center, are relatively stable [9] or repeat a similar pattern periodically. The predictable load pattern makes it feasible to redirect requests during lightly loaded hours.

(3) Queueing model is a widely adopted method to model RAID systems for performance analysis and prediction [14, 15, 16, 17, 18]. Queueing model can provide performance measures of both throughput and response time. Synchronous and asynchronous loads can be modeled by open and closed queueing models respectively [15, 18]. By using the modeling method, researches can understand the limitation and performance bottleneck of their disk array functions and optimizations [18].

Based on above observations, we attempt to solve the open problem of current server-side disk energy conservation, and provide a solution for conventional disk based RAID system, named eRAID.

3. eRAID

The main idea of eRAID is to spin down partial or entire mirror group to standby state to save energy; when these disks are in standby state, read requests are served by data copies in the primary disks. Write requests to the standby disks are deferred in controller cache or active disks, and then flushed to the standby disks after they are spun up. In this paper, we only study deferring writes by NVRAM. Since NVRAM is battery-backed, eRAID does not impact the reliability of RAID systems. It needs to be noted that the method using redundancy information to spin down disk to save energy can also be used for other RAID organizations, such as RAID-5. We proposed how to use this idea in RAID-5 with both theatrical analysis and experimental evaluation. Interested reader can refer our technical report [19]. Because of space limit, we focus on RAID-1 in this design, and only discuss how to using queueing network modeling to do performance prediction for RAID-5 in Section 5.

To spin down disks to save energy, a balance needs to be made to provide enough active disks to meet performance based service quality. eRAID deploys a time-window based performance control scheme (details in Section 3.3) to do the tradeoff between energy-saving and performance (response time and throughput) degradations. To predict the performance of storage system, we need to first forecast the load features in next time window (e.g., request inter-arrival rate). And then, we need to use the load parameters as input, feed them into a performance prediction model to make the prediction.

There are many techniques for capturing temporal behavior of workloads. To find which one is the best is beyond the scope of this paper. In our current implementation, we use ARMA (Autoregressive Moving Average), a widely adopted time serial analysis model, to predict workload features (e.g. request number in next hour) based on the load characteristics of historical information. Different queueing network models are used to precisely model the RAID-1 storage system with specific characteristics. The predicted workload feature serves as one of the input of the queueing network models.

At the beginning of each time-window, eRAID does performance/energy prediction to find out how many disks can be spun down to save energy, or how many disks should be spun up in order not to violate predefined constraints. This problem can be formalized as the follows.

$$\text{Object: maximize } S_E = \frac{E_{base} - E_{eRAID}}{E_{base}}$$

$$\text{Subject to} \begin{cases} \text{System Throughput Constraint:} \\ D_T = \frac{T_{eRAID} - T_{base}}{T_{base}} \leq Limit_T \\ \text{Response Time Constraint:} \\ D_X = \frac{X_{base} - X_{eRAID}}{X_{base}} \leq Limit_X \end{cases}$$

Here E, T and X denote energy consumption, mean response time and mean throughput. "base" represents the baseline system that employs no energy-efficiency policy. $Limit_T$ and $Limit_X$ are user-defined performance degradation parameters (e.g. 10%) for mean response time and mean throughput respectively. Next, we show how to address this problem.

3.1 Solving for Energy Saving S_E

We develop a power model for multi-disk systems to estimate the energy saving S_E. Since we mainly focus on the disk energy consumption, other components (e.g. CPU and cache) in RAID systems are not examined here. A single disk power model considering the spin down/up policy has been extensively studied. Recently, several energy consumption models are proposed for multi-speed disks [4, 6, 9, 20]. However, little research work has been conducted on modeling energy saving of a multi-disk system with spin down/up energy-efficiency policy. This is because that, disk spin down/up policy is considered an impractical method to save energy, due to the short idle periods typically encountered in server workloads. However, with the help of redundancy in RAIDs and deferring writes in NVRAM, we could use the spin up/down method to save energy in server-side environments.

3.1.1 *Formalizing* S_E

Given a time-window with length T and an N-disk RAID-1 system, assume that R requests are served in T and the average service time is t. Then the system utilization $\rho_1 = \frac{R*t}{NT}$. Let P_a, P_i, P_s and P_w denote the power when a disk is respectively active, idle, standby and doing state switching. Let T_s denote the time of disk in standby state, and T_w denote the time of disk doing state switching. For write load, the energy consumption of coherence depends on the write operation of workloads and is hard to estimate. To get the upper bound of energy saving, we do not consider coherence energy consumption here. Then the energy consumption of original disk array is $E_{base} = E_{active} + E_{idle} = P_a NT\rho_1 + P_i NT(1 - \rho_1)$.

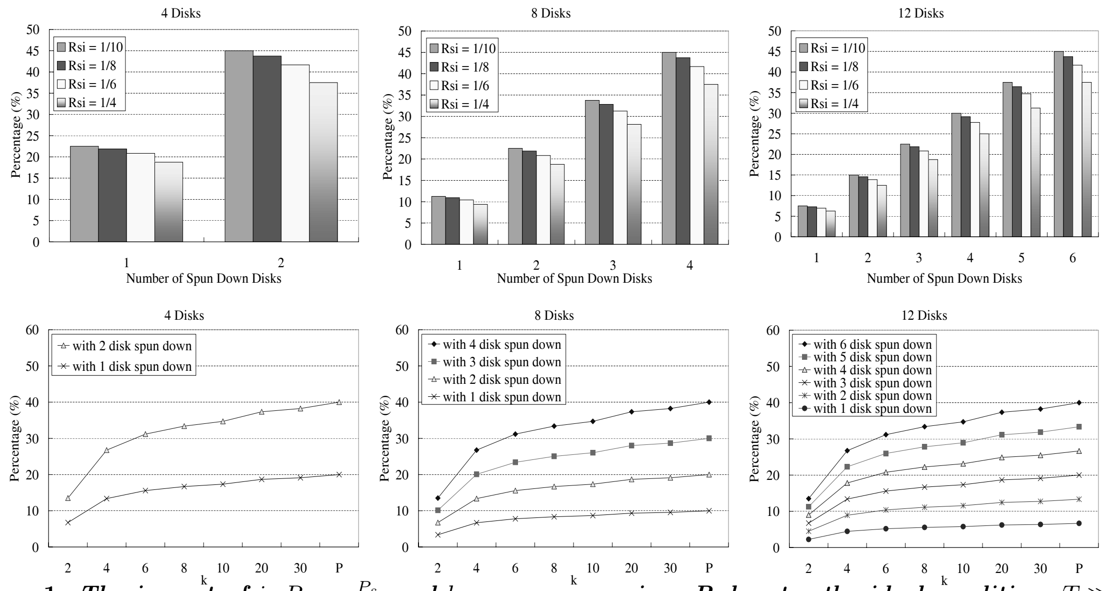


Figure 1: ***The impact of*** i, $R_{si} = \frac{P_s}{P_i}$ ***and*** k ***on energy saving. P denotes the ideal condition,*** $T \gg T_w$.

For asynchronous loads, the request arrival rate is independent of the number of active disks. While for synchronous loads, the request arrival rate could be reduced because less active disks provide smaller system throughput. That is, in synchronous loads, the served request in T may be less than R. We use ρ_2 ($\leq \rho_1$) denote the new utilization. Suppose i disks are spun down to standby state at the beginning of T, and spun up to active state at the end of T. Then, we have the energy consumption of the i disks is $i * (P_sT_s + P_wT_w)$. The energy consumption of the $(N - i)$ disks is $P_aNT\rho_2 + P_i[(N - i)T - NT\rho_2]$. Finally the new energy consumption can be described as: $E_{eRAID} = i * (P_sT_s + P_wT_w) + P_aNT\rho_2 + P_i[(N - i)T - NT\rho_2]$.

The percentage of energy saving, S_E, in the period T can be expressed as:

$$\begin{aligned} S_E &= (E_{base} - E_{eRAID})/E_{base} \\ &= \frac{(P_a - P_i)(\rho_1 - \rho_2) + \frac{i}{N}(P_i - \frac{P_sT_s + P_wT_w}{T})}{P_i + (P_a - P_i)\rho_1} \end{aligned}$$

Here ρ_1 and ρ_2 can be resolved by a performance prediction technique in Section 3.2.

3.1.2 Theoretical Analysis of S_E

From the equation of S_E, it can be seen that, the larger the value of $(\rho_1 - \rho_2)$ is, the more energy energy saving can be achieved. For asynchronous loads, $(\rho_1 - \rho_2)$ is zero. With the same ρ_1, the energy saving of asynchronous loads can be looked as the bottom line of synchronous loads. In this section, we only analyze the energy saving of asynchronous loads. We take IBM Ultrastar 36Z15 as the representative server SCSI disk in this analysis. Since disk utilization in server-side RAID systems is usually less than 35% [13], $P_i \gg (P_a - P_i)\rho_1$ is held in most practical cases. The energy saving equation can be simplified to be $S_E \approx \frac{i}{N}(1 - \frac{P_sT_s + P_wT_w}{P_iT})$.

We analyze the energy saving in two steps. In the first step, we assume $T \gg T_w$ to get the up bound of energy saving. According to this assumption, $T_s = T - T_w \approx T$. Then energy saving equation can be further simplified to be $S_E \approx \frac{i}{N} * (1 - \frac{P_s}{P_i})$. Now we can see that the energy saving is affected by two factors i and $\frac{P_s}{P_i}$. Defining $R_{si} = \frac{P_s}{P_i}$, we draw the top three diagrams in Figure 1 to show impacted of i and R_{si} to the energy saving. From these three diagrams we can see that, the energy saving is increased with the decreasing of R_{si} or increasing of i. With the same number of spun-down disks, R_{si} does not have significant impact to the energy savings. The number of spun down disks is the dominate factor for energy savings.

In the second step, we remove the assumption $T \gg T_w$. To study the impact of the length of T to the energy saving, we set $T = kT_w, k > 1$. Now we get $S_E \approx \frac{i}{N}(1 - \frac{P_s(k-1) + P_w}{P_ik}) = \frac{i}{N}(\frac{77}{102} - \frac{110}{102k})$. The bottom three diagrams of Figure 1 show the impact of T to energy saving. From these three diagrams we can see that, with the increase of k, the energy-saving is approaching that of ideal condition P ($T \gg T_w$). For a fixed k (e.g. 10), the gap between the energy-saving with $T = kT_w$ and that of P becomes larger when there are more disks spun down. For example, when k is six for the 8-disk RAID-1 with one disk spun down, the difference of energy saving is 2.2% less than that of ideal condition. While this small gap is stretched to be almost 9% when four disks are spun down to save energy. To reduce the gap to be less than 3% with four spun-down disks, k should be larger than 20.

3.2 Solving for Performance Degradations D_T and D_X

We use queueing models to calculate the performance degradation factors D_T and D_X. Unlike the research works that use queueing model to analyze RAID system performance in normal mode, our task is to study a special case: how the system performance is impacted after some mirror disks are spun down. Our method can be described as follows: (1) use queueing models to model RAID-1 and get performance predictions; (2) examine how the input parameters are changed after spinning down some mirror disks; (3) calculate new performance predictions; (4) compare original and new predictions for decision-making.

Read and write operation in RAID-1 systems have different features, and synchronous and asynchronous loads have different time criticality. To isolate the difference, we examine the performance impact of spinning down mirror disks under synchronous read (SR), asynchronous read (AR), asynchronous write (SW) and asynchronous write (AW) loads individually.

In this paper, we extend two queueing models introduced by Varki *et al.* in literature [18] to model RAID-1 system with synchronous read and synchronous write loads. Based on both models, we develop another two queueing models for asynchronous read and asynchronous write loads. The accuracy of these queueing network model is shown in Section 4.3. Our queueing models are built based on the main features of a real RAID system, HP SureStore E Disk Array FC60 [21]. It needs to be noted that, some array controller optimizations, such as access coalescing, load balancing and prefetching, are not incorporated in the models used in this paper. We will study them in our future work.

3.2.1 Read load

The left diagram of Figure 2 shows the model of a RAID-1 system with synchronous read load. It is assumed there are M processes in this system while each process generating an I/O stream (array read requests). Array read requests are first submitted to controller cache. In case of cache misses they are directed to the disks. After the requests are finished, they are returned to the processes. Only after the previous request is returned does a process issue another request following a further process-delay time.

All the processes are modeled as a delay server. The average performance measures of this closed queueing network can be computed by the Mean Value Analysis (MVA) technique [22]. For asynchronous loads, the RAID system is modeled as an open queue network shown at the right diagram of Figure 2. Since the load balance policy is not considered here, the disk array is treated as a bunch of M/M/1 queue nodes. Many researchers use M/G/1 to model the mixed read/write load for disk system. Here we use M/M/1 to model the disk with pure read or write requests. The advantage of assuming exponential distribution is the efficiency and simplicity of the corresponding performance technique [18]. Our experiments show that the error caused by this assumption is in an affordable level (details in Section 4.3). Table 5 shows the model input parameters. The disks are named from subsystem 1 to subsystem N. The RAID controller cache is named as subsystem 0. Note that $P_0 = 1$ and $\sum_{i=1}^{N} P_i = 1 - cache\_hit\_rate$.

Table 1: Read Load Model Input Parameters

Parameters	Description
Common Parameters ($0 \le i \le N$)	
N	number of all disks in RAID-1
μ_i	service rate of subsystem i
P_i	access probability of subsystem i
Synchronous Read Model	
M	the number of processes
O_p	mean process delay
Asynchronous Read Model	
λ	mean request arrival rate

After some mirror disks are spun down to save energy, two input parameters of the queueing models could be affected: disk access probability and disk service time. When i mirror disks are spun down, the load of the mirror disks will be directed to their primary disks. Access probabilities of other disks are not affected. Disk service time depends on a large number of factors, such as disk specification, disk scheduling policy and workload features. To find the relationship between disk service time and disk queue length, we directly measure the service time of three physical disks under a wide spectrum of workloads. We find that, when the disk queue length is less than three[1], the change of disk service time (for both read and write operations) has minor impact to the calculation of performance degradation. Therefore, we do not consider the change of disk service time in the prediction.

Based on above discussion, we can compute the performance measures for both the base system and eRAID. For ease of presentation, given an N-disk RAID-1, we define disk 1 to N/2 to be mirror disks and disk (N/2+1) to N to be primary disks. Let disk n and disk v be a mirror disk and its corresponding primary disk, with v=N/2-n+1. For synchronous read load, we use $T_i(M)$ and $T'_i(M)$ to denote the mean response time of subsystem i, with M processes, in the baseline system and eRAID respectively. $T_i(M)$ and $T'_i(M)$ can be easily computed by MVA technique [22, 23]. The final mean response time of the disk array and mean throughput of the baseline system are:

$$\begin{aligned} T_{base}(SR) &= \sum_{n=0}^{N} [T_n(M)P_n], \\ X_{base}(SR) &= \frac{M}{T_{base}(SR) + O_p}. \end{aligned}$$

[1]We only spin down mirror disks when the system utilization is low. Even when the disk utilization is 35%, which means the queue length is $\rho/(1-\rho) = 35\%/(1-35\%) \approx 0.54$ according to queueing theory, doubling the disk load would increase the queueing length to $70\%/(1-70\%) \approx 2.33$.

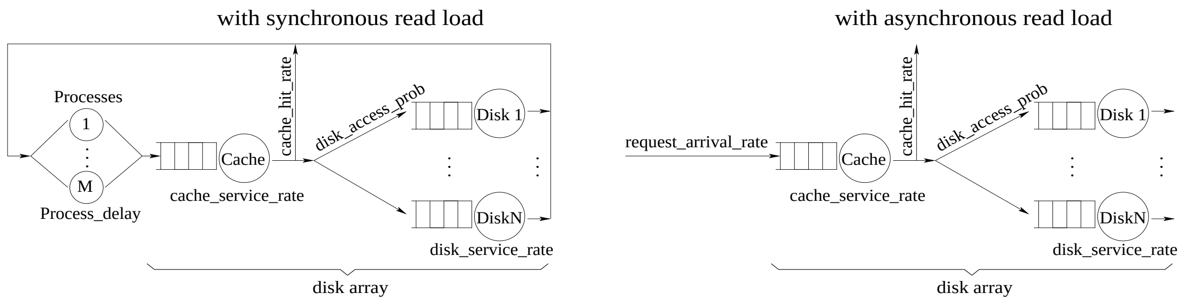


Figure 2: *Queueing network model of RAID-1 with read workloads*

The performance measures of eRAID are:

$$\begin{aligned} T_{eRAID}(SR) &= T_0(M) + \sum_{n=i+1}^{N-i} [T'_n(M)P_n] \\ &\quad + \sum_{n=N-i+1}^{N} [T'_n(M)(P_n + P_v)], \\ X_{eRAID}(SR) &= \frac{M}{T_{eRAID}(SR) + O_p}. \end{aligned}$$

For asynchronous read loads, as discussed in Section 2, system throughput is not affected if there is no disks in unstable state after spinning down the mirror disks. The mean response time is computed as mean cache response time plus mean disk response time. That is,

$$T_{base}(AR) = \frac{1}{\mu_0 - \lambda} + \sum_{n=1}^{N} \frac{P_n}{\mu_n - \lambda P_n} = \sum_{n=0}^{N} \frac{P_n}{\mu_n - \lambda P_n}.$$

After spinning down mirror disks 1 to i, the mean response time of eRAID is

$$\begin{aligned} T_{eRAID}(AR) &= \frac{1}{\mu_0 - \lambda} + \sum_{n=i+1}^{N-i} \frac{P_n}{\mu_n - \lambda P_n} \\ &\quad + \sum_{n=N-i+1}^{N} \frac{P_n + P_v}{\mu_n - \lambda(P_n + P_v)}. \end{aligned}$$

Then we discuss how to get M and O_p in real systems. In real system, none of these two parameters can be accurately detected on storage subsystem without the cooperation of host system OS or applications. Here we show how to use approximate method to get these two values. From the model of synchronous read load, we have $\frac{M}{T_{array}+O_p} = X_{array}$. Here $T_{array}$and X_{array} are mean response time and mean throughput of RAID systems. T_{array} and X_{array} can be easily measured. If anyone of M and O_p is known, the other one can be calculated by using above equation. With historical statistic information, ARMA can also be used to predict two values for next time-window, average request number (V_1) in disk array and the number (V_2) of processes that will issue requests. Since time-window length should be much longer than request service time, V_1 must be less than V_2. V_1 can be taken as the lower-bound of M because there may have more requests hidden in the thinking time of the processes. While V_2 can be looked as the upper-bound of M because some processes may finish their I/O tasks and then exit in the middle of the time-window. Therefore we can approximate M by $(V_1 + V_2)/2$. Then O_p can be easily computed.

3.2.2 *Write load*

Modern disk arrays usually implement write-back caching in the array controller. In this case, unlike reads, a write request is completed once the data is written to cache. FC-60 uses a two-threshold write-back policy [18, 21], destage_threshold (e.g 20% of cache size) and max_dirty_blocks (e.g cache size). Destaging operations starts if the number of saved dirty blocks is larger than destage_threshold. The maximum dirty blocks that can be held in the cache is determined by max_dirty_blocks. For write workloads, the RAID system can be modeled as a Markov birth-death M/M/1/K process [18], with $K = \frac{max\_dirty\_blocks - destage\_threshold}{average\_request\_size} + 1$. Practically, K is the maximum number of requests that can be held in cache. Thus, the disk array can be modeled by the cache alone with input parameters listed in Table 2.

Table 2: Write Load Model Input Parameters

Para.	Description
P_{miss}	array cache write miss probability
K	maximum queue length
d_λ	arrival rate of dirty blocks to cache
d_μ	rate at which dirty blocks are written from cache to disks
μ	mean disk service rate
	Synchronous Write Model
χ	maximum array throughput

Since all data must be written to both mirror and primary disks, an N-disk RAID-1 can serve N/2 requests at the same time. Thus the service rate d_μ is $\frac{N}{2\mu}$ [18]. In the synchronous write model, χ denotes the maximum array throughput, which is the array's throughput when the disk array has an infinite cache, and *chi* be computed by MVA technique. The arrival rate of dirty blocks is different for synchronous and asynchronous write loads. For a synchronous write load, d_λ can be approximated by $\chi * P_{miss}$, while for asynchronous loads, $d_\lambda = \lambda * P_{miss}$ where λ can be directly measured.

In current design of eRAID, we use NVRAM to save inconsistent data. By spinning down i disks, the service rate d_μ is $\frac{N-2i}{2\mu}$. Another two input parameters may be affected when some of the mirror disks are spun down: K and P_{miss}. Since deferring the write of dirty blocks of standby disks in the cache for a longer period may increase the destage_threshold, the queue length K may be decreased. For the other parameter P_{miss}, through experiments we found eRAID made little impact on P_{miss}. Two reasons to explain this. Firstly, the data access locality is relatively low in controller cache compared with other higher level cache such as file system cache. Secondly, eRAID restricts the accumulated dirty blocks of standby disks within the second half of LRU table to reduce the cache pollution problem.

For synchronous write load, the mean throughput of the disk array is computed by $\chi * (1 - P_{max\_dirty})$. Here P_{max_dirty} is the steady state probability when the cache has number of maximum dirty blocks. According to M/M/1/K queueing model [22], $P_{max\_dirty} = \frac{(1-a)*a^K}{1-a^{K+1}}$ with $a = \frac{d_\lambda}{d_\mu} = \frac{2\mu\chi P_{miss}}{N}$. That is,

$$X_{base}(SW) = \chi * (1 - \frac{(1-a)*a^K}{1-a^{K+1}}),\ a = \frac{2\mu\chi P_{miss}}{N}.$$

The mean response time can be computed using Little's Law on the entire system:

$$T_{base}(SW) = \frac{M}{X_{base}(SW)} - O_p.$$

$X_{eRAID}(AW)$ and $T_{eRAID}(AW)$ can be computed in the same way as computing $X_{eRAID}(SW)$ and $T_{eRAID}(SW)$ but replacing K by K' and $a = \frac{2\mu\lambda P_{miss}}{N-2i}$. Here $K' = \frac{max\_dirty\_blocks - new\_destage\_threshold}{average\_request\_size} + 1$.

For asynchronous load, the mean response time is $\frac{\overline{k}}{d_\lambda}$ according to M/M/1/K model. Here $\overline{k}$ is the average queue length and $\overline{k} = \frac{a}{1-a} - \frac{(K+1)a^{K+1}}{1-a^{K+1}}$ with $a = \frac{d_\lambda}{d_\mu} = \frac{2\mu\lambda P_{miss}}{N} < 1$. That is,

$$T_{base}(AW) = \frac{a}{1-a} - \frac{(K+1)a^{K+1}}{1-a^{K+1}},\ a = \frac{2\mu\lambda P_{miss}}{N}.$$

In the same way, $T_{eRAID}(AW)$ can be computed via replacing K by K' and $a = \frac{2\mu\lambda P_{miss}}{N-2i}$.

3.3 Control Algorithm

The control algorithm is used to automatically adjust the number of spun-down disks to trade off energy-efficiency and performance degradation. eRAID uses a time-window based control scheme with the forecasting of energy-saving and performance degradations. If the time-window length is set too long, the performance control algorithm may be inefficient and blunt to the unexpected changes of request traffic. At the same time, the time-window length can not be too short in order to avoid frequent state switch. Chen *et al.* [24] pointed queuing model based control is more viable at large time granularity. Through experiments we also find that the inaccuracies of predictions cause much performance degradation with small time-window (e.g $\leq$ 0.5 hour). In our current implementation, we set one hour as the default time-window length. The sensitivity study about the impact of time-window length is shown in Section 4.5.

As mentioned by Zhu *et al.* [9], frequently starting and stopping disks affects disk drive longevity. Usually disks are designed to sustain a certain number of start/stop cycles. For example, IBM Ultrastar 36Z15 can handle a minimum of 50,000 start/stop cycles [25]. This disk would reach its warrantable service time (three years) if disk state switching is less than 45 time a day. Therefore, we set 45 time per day to be the maximum start/stop frequency for each disk. A time window length is set to be long enough to enforce this constraint.

Based on the discussion of previous sections, we can do the predictions for both energy saving and performance degradations. Then next step is to solve the multi-constraint problem. This multi-constraint (S_E, D_T and D_X) problem can be represented as a multidimensional 0-1 knapsack problem with the dimension being two (response time limit and throughput limit). The multidimensional 0-1 knapsack problem is a classified NP-hard optimization problem [26]. Solving this problem involves much computing overhead. Fortunately, by examining the nature of disk array system, we can simplify the solution according two observations:

- First, as proved in Section 3.1, the energy saving S_E is dominated by the number of spun-down disks rather than which disks to spin down. In this case, our goal is to spin down as many disks as possible without violating the two performance constraints.
- Second, according to queueing model, the server (disk) with lower utilization tends to give less performance degradation when its access probability is doubled. That is, given performance degradation constraints, we may achieve the maximum number of spun-down disks by spinning down lightest loaded mirror disks.

Therefore, we first order all the mirror disks from the lightest loaded to the heaviest loaded according to the predicted load of each disk. Then, starting from the lightest loaded disk, we try to spin down as more mirror disks as possible without violating the constraints. This solution is described as Algorithm 1. Our experiments show that this algorithm gives optimal solution in most cases. Compared with using close-to-optimal algorithms [26] to solve the multidimensional knapsack problem, Algorithm 1 needs much less computing overhead.

However, in real world systems, prediction error of performance degradation is unavoidable. Here we define the *prediction error* (a ratio) to be denoted by

$$\sigma = \frac{|real\_degradation - predicted\_degradation|}{real\_degradation}.$$

This error comes from the load predictor and accuracy of queueing network models. Since both load predictor's error and the accuracy of queueing network models can be measured by verifying them with real workload and systems, we assume the average error is a known parameter and it is updated periodically (e.g. every day). In order not to violate the limits, we reset $Limit_T$ to be $Limit_T \times (1-\sigma)$ and $Limit_X$ to be $Limit_X \times (1-\sigma)$. Obviously, if σ is larger than one, eRAID can not spin down any disk to save energy. Experiment results show that σ is less than 15% in current design. More details about the prediction error are shown in Section 4.3.

```
Algorithm 1: find mirror disks to spin down
1: Put active mirror disks into list S at any order;
2: Sort all the mirror disks from the least loaded to
       the heaviest loaded;
3: $for(i = 1; i < N/2; i++)\{$
4:    Predict performance degradations of spinning
       down first $i$ disks of S;
5:    If any one of the constraints is violated, then the
       first $(i-1)$ disks in S are disks to spin down;
6: }
```

```
Algorithm 2: conservative control algorithm
1: $Limit_T = Limit_T \times (1-\sigma)$;
2: $Limit_X = Limit_X \times (1-\sigma)$;
3: Put the disks found by Algorithm 1 into set T;
4: Spin down disks in T;
5: Spin up standby disks that are not in T;
```

The conservative control algorithm is summarized as Algorithm 2, which is called at the beginning of each time-window. From Algorithm 1&2, we can see eRAID always tries to keep the performance degradation less than constraints in each time-window, so we call it a conservative control.

For write load, eRAID always tries to replace dirty blocks belonging to active disks until the accumulated dirty data of standby disks is larger than the new destage_threhold. eRAID should make the new destage_threhold be able to cache dirty blocks of standby disks for a reasonable long period. The length of this period can be decided by studying the relationship between disk spin-down period and energy-saving. For example, for IBM Ultrastar 36Z15, Figure 1 shows relatively large energy saving can be achieved when the disk spun-down period is longer than six times of the sum of disk spin-down and spin-up period. Therefore, the calculation of the new destage_threshold can be derived from: $\frac{destage\_threshold}{data\_write\_rate\_of\_standbydisks} > 6T_W$. The standby disks are activated when the size of their accumulated dirty blocks in the controller cache are larger than the new destage_threshold. After all the dirty data are flushed back to these disks, eRAID spins down them again if these disks' start/stop frequency is less than 45 times per day.

4. EXPERIMENT EVALUATION

4.1 Experiment Setup

We evaluate our policy with both synchronous and asynchronous loads by using trace-driven simulations. We choose IBM Ultrastar 36Z15 [25] as the disk power model. This disk is commonly used in data-intensive environments and some research works [9]. The important parameters [4] are shown in Table 3.

Table 3: Disk Model

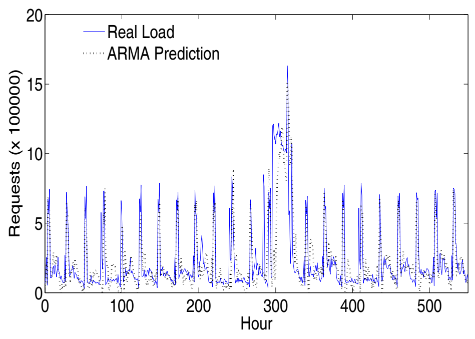
Parameter	Value
Individual Disk Capacity	18.4 GB
Disk Rotation Speed	15000 RPM
Active	13.5 W
Seek Power	13.5 W
Standby Power	2.5 W
Idle Power	10.2 W
Spinup Power	13.5 W
Spinup Time	10.9 secs.
Spindown Time	1.5 secs.
Average Rotation Delay	2 ms
Internal Transfer Rate	55 MB/sec

Eight traces are used in the evaluation. We use two traces to evaluate eRAID for each of AR, AW, SR and SW loads. For asynchronous loads, we select two real world traces to replay I/O traffic on large storage server systems. One is extracted from the Cello99 trace suite that was collected from the HP "cello" server over the period from January 14 - December 31, 1999. During that collection time, cello was a K570 class machine (4 CPUs) running HP-UX 10.20, with about 2 GB of main memory[2]. To make Cello99 trace intensive enough for current hardware conditions, we merge three day's (4/12-4/14/1999) load into one day. The other trace is TPC-C20, which was collected by running TPC-C database benchmarks with 20 warehouses on a storage server with a Redhat Linux 7.1 OS[3]. We extract 14-hour trace for TPC-C20 for the evaluation. The characteristics of the real traces are listed in Table 4 and the request number in each hour of these two traces are shown in the right diagram of Figure 3. We separate the read and write requests of these two traces for AR and AW loads respectively.

Since real traces do not provide enough information about the synchronous relation among requests, we generate two synthetic synchronous loads based on the extracted parameters from Cello99. With the assumption that 30% of all requests are synchronous requests, we get 14 and 23 as the representative process numbers for the loads in the morning and the afternoon respectively. The 14-process trace is named S14 while S23 for the 23-process traces. To make S14 and S23 to be roughly same intensive, we set the average process delay to be 40 ms and 80 ms. The access sequentiality and locality are 0.22 and 0.36, and average request size is also the same as that of Cello99. The request inter-arrival time follows exponential distribution.

In this implementation, time serial analysis model ARMA (Autoregressive Moving Average) is used to predict workload characteristics. One example is given by the left diagram in Figure 3 that shows the predicted request

[2]http://tesla.hpl.hp.com/public_software
[3]http://traces.byu.edu/new/Tools



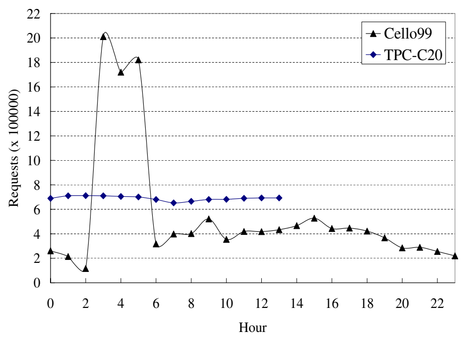


Figure 3: Trace load characteristics

Table 4: Trace Features

Traces	Cello99	TPC-C20
Avg. Inter arrival time	6.57 ms	5.21 ms
Read Ratio	65.52%	75.05%
Avg. Request size	12.73 KB	40.66 KB

number and real request number of each hour in the real-world workload Cello99. We use the last seven day's load to predict next one day's load for Cello99. While for TPC-C20 that has a more stable I/O traffic, using the last two hours' loads to predict next one hour's load is good enough.

We configure a RAID-1 system based on Hewlett-Packard SureStore E FC-60 disk array [21] in Disksim, a popular storage system simulator. The RAID-1 system has eight disks and two mirrored array controllers. Each controller has 512MB NVRAM. Then we augment Disksim with disk power model and our energy-efficient policy. LRU is chosen as the default controller replacement algorithm. For eRAID, we defer the data update of spun-down disks only in the second half of the controller cache as suggested in literature [27].

4.2 Evaluation Results

Figure 4 shows the overall simulation results of AR, AW, SR and SW loads for three scenarios. From Figure 4 we can see that, our performance control algorithm is effective for all the four kinds of workloads and performance degradation constraints,$Limit_T$ and $Limit_X$, are not violated in all the scenarios. In CASEI with tight constraints, the maximum energy saving, 22.0%, is achieved in the AR load of Cello99. Since TPC-C20 loads are more intensive than Cello99 loads, less energy savings are obtained from TPC-C20 loads, only 17.0% for its AR load and 7.1% for AW load. Because of the constraints of both response time and throughput, less energy are saved for synchronous load than for asynchronous loads. The maximum energy of synchronous loads, 15.2%, is achieved in read load of S23. For any one of synchronous and asynchronous loads, more energy is conserved in read loads than write loads. This is because that, when mirror disks are spun down in RAID-1 with write loads, the decrease of system service rate degrades more than that with read loads. For example, with same disk access probability, when i mirror disks are spun down, system service rate is changed from N to $N-i$ in AR, while it is changed from $\frac{N}{2}$ to $\frac{N-2i}{2}$ in AW loads. The degradation of system service rates are $\frac{N-(N-i)}{N} = \frac{i}{N}$ and $\frac{\frac{N}{2}-\frac{N-2i}{2}}{\frac{N}{2}} = \frac{2i}{N}$ for AR and AW loads respectively.

CaseII loosens the throughput degradation limit from 5% in CASEI to 10%. This does not impact the energy savings of asynchronous loads, but eRAID achieves more (2.3∼5.9%) energy saving with the synchronous loads. CASEIII loosens the constraints to a further step through increasing the limit of response time degradation from 20% to 50%. This change makes both synchronous and asynchronous loads see more energy savings. Compared with the energy savings in CASEII, the energy savings of synchronous loads in CASEIII have an increase of 4.5∼9.0%, while 5.0∼11.4% for asynchronous loads. Finally, the maximum energy saving with the three cases is 32.4% that is achieved in AR load of Cello99.

For all loads, the degradation of throughput is less than that of response time. With synchronous loads, the degradation of response time is from 7.0∼18.5%, while the throughput degradation is from 2.3∼8.0%. In all cases, S23 sees less throughput degradations then S14 does. With providing the same I/O intensity, the load having more processes (with longer process delay) trends to have less throughput degradation. Asynchronous load, which has no throughput degradation, can be viewed as an extreme case with very large process number and very long process delay. For asynchronous loads, the maximum response time degradation is observed in read load of TPC-C20.

In Figure 6, we take Cello99 as an example to show the details of energy saving, response time degradation and average number of spun-down disks in each hour in CASEIII. With read load, eRAID can spin down three to four disks in most time, while less than three disks in most time for AW load. In the heaviest loaded period, from 3AM to 5AM, no disk is spun down.

4.3 The Error of Performance Predictions

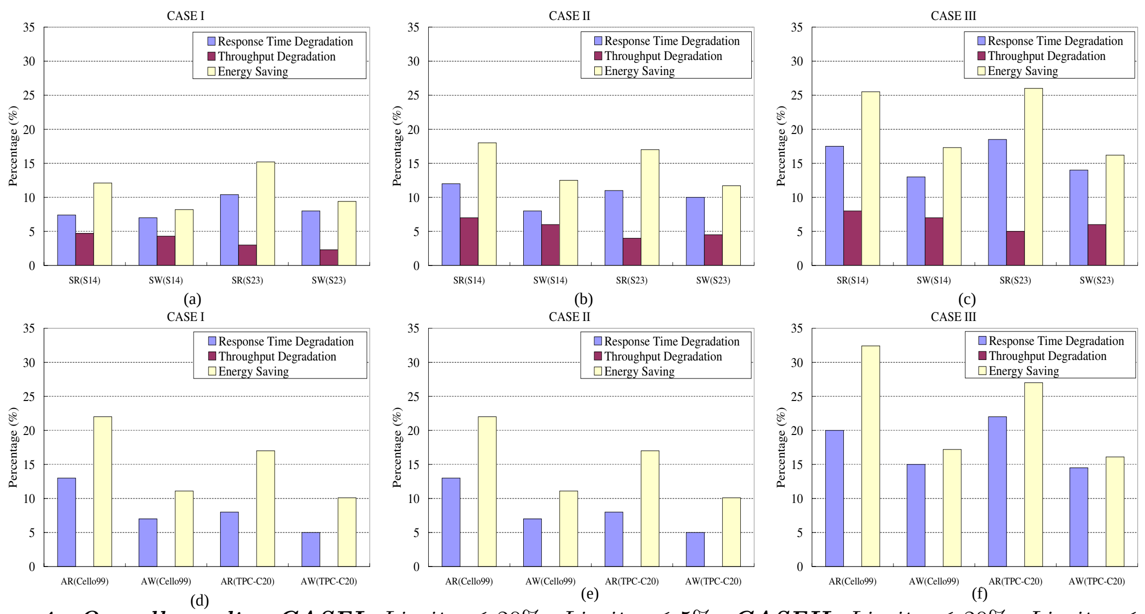


Figure 4: ***Overall results.*** ***CASEI:*** $Limit_T \leq 20\%$***,*** $Limit_X \leq 5\%$***;*** ***CASEII:*** $Limit_T \leq 20\%$***,*** $Limit_X \leq 10\%$***;*** ***CASEIII:*** $Limit_T \leq 50\%$***,*** $Limit_X \leq 10\%$***.***

Although providing generalized models to accurately predict both workload changing and system performance is not the task of this paper, we need to consider the prediction error into our control policy as mentioned in Section 3.3. In our current design we assume the prediction error of degradation is a given parameter. In order to get the real performance degradations, we run the simulations again without any disk spun down and collect performance measures of each time-window. By comparing these performance measures with those of eRAID, we can get the real performance degradations. From the real degradation and eRAID prediction for each time-window, we get the average prediction error of degradation.

Figure 5 shows the prediction error of degradation for different workloads. From the left diagram of Figure 5, it can be seen that the prediction errors for response time degradations are always larger than those for throughput degradations. In the right diagram of Figure 5, we show two kinds of results. One is the response time degradation with ARMA providing workload (request number) prediction, and the other one is the response time degradation with prefect traffic prediction, which is realized by inputting real workload features gotten by off-line trace analysis to eRAID. ARMA's traffic prediction contributes 1$\sim$3% to overall prediction errors. Because the request inter-arrival time of TPC-C20 loads less strictly follows exponential distribution, the degradation prediction errors of TPC-C20 loads are larger than those of Cello99 loads with perfect traffic prediction. However, I/O traffic of TPC-C20 is more stable than that of Cello99, so less degradation prediction errors are introduced by ARMA traffic prediction.

4.4 The Effect of RAID Size

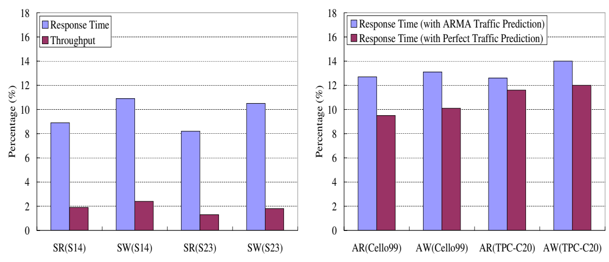


Figure 5: ***Prediction errors of perf. degradations (defined by*** $\frac{|real\_degradation - predicted\_degradation|}{real\_degradation}$***)***

To examine the effect of RAID size, we redo experiment for 6-disk RAID-1 and 10-disk RAID-1 with CASEIII as the constraint. We take traces S14 and Cello99 as the synchronous and asynchronous loads. To make relatively fair comparisons, we randomly remove 1/4 of all requests from Cello99 loads for 6-disk RAID-1, For 10-disk RAID-1, we extract requests from another day's trace and add them to Cello99 to let it have 1/4 more requests than the original trace. For S14, the average process delay is increased from 40 ms to 55 ms for 6-disk RAID-1. On the contrary we decrease the process delay to be 30 ms for 10-disk RAID-1. Since NVRAM is important for write loads, we also change the NVRAM size to 384 MB and 640 MB respectively for 6-disk and 10-disk RAID-1 systems.

Figure 7 shows the impact of the RAID size to performance and energy saving. All the results are normalized to the results of 8-disk RAID-1 system. From Figure 7 we can see that the energy saving and performance degradation of 6-disk RAID-1 system are always less than those of 8-disk RAID-1 system. While we get opposite observation for 10-disk RAID-1 system. From Figure 1 in Section 3.1 we know that, the upper-bound

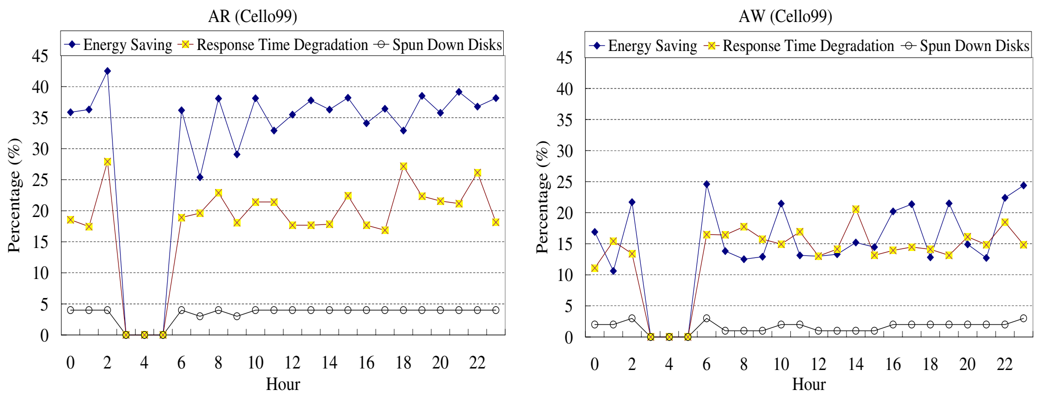


Figure 6: ***The experiment results in each hour of Cello99***

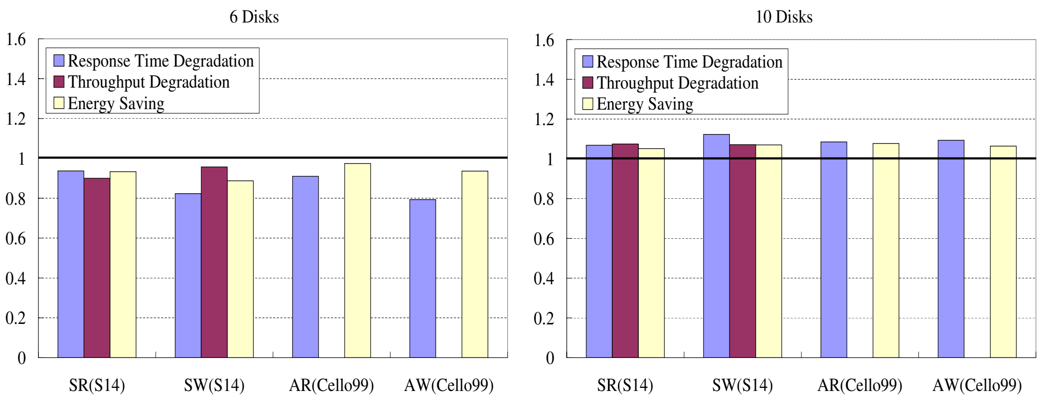


Figure 7: ***The impact of RAID size (results are normalized to 8-disk RAID-1)***

of energy-saving is around 40% for RAID-1 with IBM 36Z15 disks. With a larger size, the RAID-1 system has more options to fine tune the number of spun-down disks. In a 6-disk RAID-1 system, only three mirror disks, while in a 10-disk RAID-1 system, eRAID can select the number of spun-down disk from zero to five. With the ability to better tune the number of spun-down disks, eRAID saves more energy in 10-disk RAID-1 system than that in 8-disk RAID-1. At the same time, the performance degradations is also increased.

4.5 The Effect of Time-Window Length

Figure 8 shows the energy effects of time-window length on eRAID with four workloads, SR(14), SW(14), AR(Cello99) and AW(Cello99) for CASEIII. From Figure 8 we can see that, with the increase of time-window length, energy-savings of all the loads tend to be relatively stable. However, energy-savings of Cello99 loads drop quickly with the decreasing of time-windows length. This is because both ARMA and queueing models generate larger prediction errors with fine granularity (smaller time-window length). In order to guarantee the performance degradation is less then predefined limits, conservative control tightens the constraints by lower the limits ($Limit_T <= Limit_T \times (1-\sigma)$, $Limit_X <= Limit_X \times (1-\sigma)$). Therefore, the faster the σ increases, the quickly energy-savings decreases.

S14 loads are more insensitive to the time-window length and the change of energy-saving is within 5%. This is because it is a stable load and its request inter-arrival time is strict exponential distribution. The prediction error does not change much with the change of time-window length.

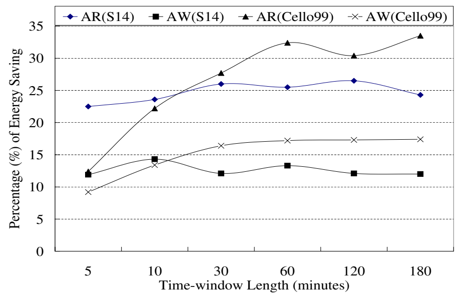


Figure 8: ***The impact of time-window length***

4.6 Limitations of eRAID

The limitation of eRAID mainly derives from the accuracy of queue network models and load predictor (i.e., ARMA). The more precisely the system can be modeled, the more accurate decisions eRAID can make. Further more, in current design, some input parameters (e.g. process number and mean process delay) are set by approximate methods. More accurate approximate methods are expected to be developed with the hints from OS or applications. The conservative control is easy to implement, which is its advantage. However, we believe the energy-saving potential of eRAID is not fully exploited. If more aggressive control policies are employed by eRAID, more energy-saving could be achieved.

5. EXTENDING eRAID FOR RAID-5

For RAID-5 systems, we still want to follow the basic approach we used in RAID-1 system: read request is served by using degraded mode (with XOR operation) and write request is deferred by caching. This idea is proposed in our technical report [19]. Here we only discuss how to using queueing network modeling to do performance prediction for RAID-5.

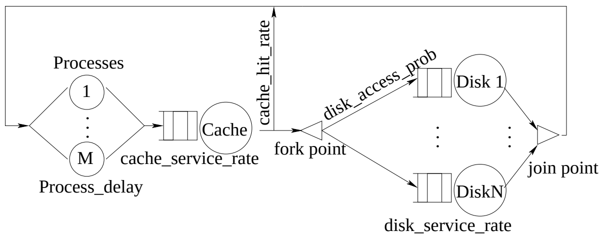


Figure 9: *Queueing network model of RAID-5 with synchronous read workloads*

There are two major differences between RAID-1 and RAID-5 systems.

- RAID-1 does not stripe data, while RAID-5 stripes data among all disks. Thus, an array read request in RAID-5 may be divided into more than one stripe requests for different disks. Only when all the disk requests are finished, the array request is served and returned to the user.
- In RAID-1 system, data update is flushed to two copies, that is, one write request involves two write operations. However, a write request in RAID-5 system may trigger four operations, two reads and two writes: read old data stripe, read old parity stripe, write new data stripe and write new parity stripe that is generated by exclusive-ORing the first three stripes.

Compared with RAID-1, RAID-5's unique features make it more challenging to build accurate models for it. Here we take the model proposed in literature [28] to model RAID-5 system with synchronous read load to get performance predictions. Figure 9 shows the system queueing network model. Table 5 shows the model input parameters.

The performance measures of this queueing network can be easily computed by using MVA technique. MVA technique iteratively solves the following equations with m = 1, 2, ..., M, and $Q_i(0) = 0$ for all subsystems: $R_i(m) = \frac{1}{\mu_i}[H_k+Q_i(m-1)]$, $X_i(m) = \frac{m}{\sum_{n=0}^{N}[R_n(m)P_n/P_i]+O_p}$, $Q_i(m) = R_i(m) * X_i(m)$. Here H_k is the kth harmonic number defined as $\sum_{i=1}^{k}\frac{1}{i}$. $R_i(m)$, $X_i(m)$ and $Q_i(m)$ are respectively response time, throughput and queue length of subsystem i when the system has m requests.

Table 5: Read Load Model Input Parameters

Para.	Description
	Common Parameters ($0 \le i \le N$)
N	number of all disks in RAID-5
μ_i	service rate of subsystem i
P_i	access probability of subsystem i
k	mean number of disks that a request is striped onto
M	the number of processes
O_p	mean process delay

Then we discuss the possible change of the input parameters after one disk are spun down. For read requests, spinning down one disk, all the read requests for this disk are served by degrade operation. This increases workload of the whole system, and thus the access probabilities of all the other disks. Suppose originally one array request is striped onto k ($\le N$) disk requests on average in an N-disk RAID-5 system. There are two cases:

1. k disk requests are all on active disks and all the requests can be served normally;
2. one of k disk requests are on standby disk and $N - 1 - (k - 1) = N - k$ new disk requests are needed to do XOR operation.

The probability of case 1 and 2 are respectively C_k^{n-1} and C_{k-1}^{n-1}. Now, one array read request is striped onto $k' = k * C_k^{n-1} + (N - k) * C_{k-1}^{n-1}$ disk requests. Based on above discussion, the final mean response time of disk array and mean throughput of the baseline system can be expressed as: $T_{base}(SR) = \sum_{n=0}^{N}[T_n(M)P_n]$ and $X_{base}(SR) = \frac{M}{T_{base}(SR)+O_p}$. By replacing H_k with H'_k, we can get the performance measures of eRAID.

6. RELATED WORK

In last decade, a lot of power-efficient solutions have been developed to find good thresholds to spin down disks for energy conservation for a single hard disk drive [29, 30, 31, 32]. Youssef examined the design issues of low-power, highly available disk arrays for mobile computers. He found that, by dynamically remapping the location of newly written data using a log-based allocation strategy and by deferring parity updates in an NV-RAM cache, the rate of drive spin-ups can be reduced by a factor of 2 [33]. Although aforementioned solutions work well for a single disk or mobile computers, they are not suitable for the server storage system because the idleness period in server loads is too short for disks to switch to different power-state modes.

Recently, some researchers have been striving for seeking novel power- and energy-efficient solutions for server disk storage systems. Gurumurthi *et al.* advocated the use of new disk architecture called multi-speed disk and developed the corresponding power management policy named DRPM to reduce energy consumption for server workloads [6]. The multiple-speed disk is able to quickly scale down its rotating speeds for energy conservation during the period of light load, and quickly speed up to service requests during the period of heavy load. Their experiments proved such a multi-speed disk can significantly slash power-consumption. Unfortunately the multi-speed disk is not available on the market. Carrera *et al.* studied some hybrid approaches by combining laptop disks and server disks together to conserve energy [4]. They conclude that only two-speed disk solution can effectively save energy varying from 14% to 23%. Recently Zhu *et al.* developed several power-aware storage cache management algorithms to indirectly save energy [3]. Colarelli *et al.* [10] used "cache disks" to cache active files/blocks, allowing other disks to spin down. Pinheiro and Bianchini presented a Popular Data Concentration (PDC) scheme to save energy for network servers. PDC aims to skew the load toward a few of all the disks, so that others can be transitioned to low-power modes by migrating frequently accessed data to a subset of the disks [2]. Unfortunately the above-mentioned techniques do not work at the RAID system level and thus ignore the detail among various disk array organizations. Huang *et al.* [34] propose a file system named FS2 which can dynamically places copies of data in file system's free blocks according to the disk access patterns. As a result, their approaches could not best obtain the energy savings. eRAID is able to directly optimize the disk access distribution for the best energy-efficiency by redundancy-aware request dispatches with predictable performance impairment.

There is a large body of work on analytical models of disk arrays [14, 15, 16, 17, 35, 36, 37, 38]. Menon and Kasson [39] present response-timemodels of disk arrays using RAID-5 layout, which include some simple cache effects but disk drives are not directly modeled as a part of these models. Uysal et al provide a modular, analytical through put model with a disk model [40] derived from Shriver *et al.* [41], as is the general idea of decomposing a storage system into components which transform the I/O stream passing through them. Recently Varki *et al.* [18] provide integrated performance models for RAID systems with considering both disks and array cache, and verify their model on a real RAID system. We extend their models for RAID-1 system for performance predictions. Zhu *et al.* [9] use M/G/1 queueing model to estimate the average response time for each disk. Their usage of queueing model is limited to a single disk. In eRAID, the queueing models take in account the controller cache and distinguish synchronous load from asynchronous load.

7. CONCLUSIONS AND FUTURE WORK

Most of current server-side disk energy saving solutions resort to the help of multi-speed disks. In this paper, we propose an energy saving policy named eRAID deploying request redirecting for conventional disk based RAID-1 systems. We evaluate our policy by trace-driven simulation. Eight traces are used in the evaluation. Experimental results show that eRAID can save up to 32% energy without violating predefined performance constraints.

In the future, we would like to extend the idea of eRAID to other RAID organizations (e.g. RAID-10) with the incorporating of advanced controller features such as access coalescing, load balancing and prefetching. Moreover, we would like to implement eRAID on a soft RAID system for more comprehensive study and evaluation.

8. REFERENCES

[1] "Power, heat, and sledgehammer." White paper, Maximum Throughput Inc., http://www.max-t.com/downloads/whitepapers/SledgehammerPowerHeat20411.pdf, 2002.

[2] E. Pinheiro and R. Bianchini, "Energy conservation techniques for disk array-based servers," in *Proceedings of the 18th International Conference on Supercomputing*, pp. 68–78, June 26 - July 01 2004.

[3] 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 *Tenth International Symposium on High Performance Computer Architecture (HPCA-10)*, (Madrid, Spain), Feb. 14–18, 2004.

[4] E. V. Carrera, E. Pinheiro, and R. Bianchini, "Conserving disk energy in network servers," in *Proceedings of the 2003 International Conference on Supercomputing (ICS-03)*, (New York), pp. 86–97, ACM Press, June 23–26 2003.

[5] S. Gurumurthi, J. Zhang, A. Sivasubramaniam, M. Kandemir, H. Franke, N. Vijaykrishnan, and M. J. Irwin, "Interplay of energy and performance for disk arrays running transaction processing workloads," in *Performance Analysis of Systems and Software (ISPASS)*, pp. 123–132, Mar. 2003.

[6] S. Gurumurthi, A. Sivasubramaniam, M. Kandemir, and H. Franke, "DRPM: dynamic speed control for power management in server class disks," in *Proceedings of the 30th Annual International Symposium on Computer Architecture (ISCA-03)*, (New York), pp. 169–181, June 9–11 2003.

[7] D. Li and J. Wang, "EERAID: Energy-efficient redundant and inexpensive disk array," in *Proceedings of 11th ACM SIGOPS European Workshop*, (Leuven, Belgium.), September 20-22, 2004.

[8] X. Li, Z. Li, F. David, P. Zhou, Y. Zhou, S. Adve, and S. Kumar., "Performance-directed energy management for main memory and disks," in *Proceedings of the Eleventh International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS'04)*, October 2004.

[9] Q. Zhu, Z. Chen, L. Tan, Y. Zhou, K. Keeton, and J. Wilkes, "Hibernator: Helping disk arrays sleep through the winter," in *20th ACM Symposium on Operating Systems Principles (SOSP'05)*, (Brighton, United Kingdom), October 23-26 2005.

[10] D. Colarelli and D. Grunwald, "Massive arrays of idle disks for storage archives," in *Proceedings of Super Computing'2002 Conference CD*, (Baltimore, MD), pp. 312–317, IEEE/ACM SIGARCH, Nov. 2002.

[11] J. Hennessy and D. Patterson, *Computer Architecture: A Quantitative Approach, 3rd ed.* Palo Alto, CA 94303: Morgan Kaufmann Publishers Inc., 2002. ISBN 1-55860-596-7.

[12] P. M. Chen, E. L. Lee, G. A. Gibson, R. H. Katz, and D. A. Patterson, "RAID : High-performance, reliable secondary storage," *ACM Computing Surveys*, vol. 26, pp. 145–185, June 1994.

[13] S. Gurumurthi, A. Sivasubramaniam, M. Kandemir, and H. Franke, "Reducing disk power consumption in servers with drpm," *IEEE Computer Special Issue on Power-Aware and Temperature-Aware Computing*, vol. 36(12), 2003.

[14] J. B. Chen and B. N. Bershad, "The impact of operating system structure on memory system performance," in *14th ACM Symposium on Operating Systems Principles, Ashville, NC, USA, December 5–8, 1993* (ACM, ed.), vol. 27(5) of *Operating Systems Review*, (New York, NY 10036, USA), pp. 120–133, ACM Press, Dec. 1993.

[15] S. Chen and D. Towsley, "A performance evaluation of RAID architectures," *IEEE Transactions on Computers*, vol. 45, pp. 1116–1130, Oct. 1996.

[16] A. Merchant and P. S. Yu, "Analytic modeling of clustered RAID with mapping based on nearly random permutation," *IEEE Trans. Comput.*, vol. 45, no. 3, pp. 367–373, 1996.

[17] O. I. Pentakalos, D. A. Menasce, M. Halem, and Y. Yesha, "An approximate performance model of a unitree mass storage system," in *Fourteenth IEEE Symposium on Mass Storage Systems*, (Monterey, CA), pp. 210–224, IEEE, 1995. GSFC.

[18] E. Varki, A. Merchant, J. Z. Xu, and X. Z. Qiu, "Issues and challenges in the performance analysis of real disk arrays," *IEEE Transactions on Parallel and Distributed Systems*, vol. 15, pp. 559–574, June 2004.

[19] D. Li, H. Cai, X. Yao, and J. Wang, "Exploiting redundancy to construct energy-efficient, high-performance RAIDs," Tech. Rep. TR-05-07-04, Computer Science and Engineering Department, University of Nebraska Lincoln, 2005.

[20] Q. Zhu and Y. Zhou, "Power-aware storage cache management," *IEEE Transactions on Computers*, vol. 54, pp. 587 – 602, May 2005.

[21] Hewlett-Packard Company, "HP SureStore E Disk Array FC60 User's Guide," vol. Pub.No.A5277-90001, Dec. 2000.

[22] G. Bolch, S. Greiner, H. de Meer, and K. S. Trivedi, *Queueing Networks and Markov Chains: Modeling and Performance Evaluation with Computer Science Applications.* New York: John Wiley & Sons, Aug. 1998.

[23] E. Varki., "Response time analysis of parallel computer and storage systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 12, pp. 1146–1161, Nov. 2001.

[24] Y. Chen, A. Das, W. Qin, A. Sivasubramaniam, Q. Wang, and N. Gautam, "Managing server energy and operational costs in hosting centers," in *SIGMETRICS*, pp. 303–314, 2005.

[25] "IBM Hard Disk Drive - Ultrastar 36Z15." http://www.hitachigst.com/hdd/ultra/ul36z15.htm.

[26] M. R. Garey and D. S. Johnson, *Computer and Intractability: A Guide to the Theory of NP-Completeness.* W. H. Freeman, 1979.

[27] A. Varma and Q. Jacobson, "Destage algorithms for disk arrays with non-volatile caches," in *High Performance Mass Storage and Parallel I/O: Technologies and Applications* (H. Jin, T. Cortes, and R. Buyya, eds.), New York: IEEE/Wiley Press, 2001. chap. 10.

[28] E. Varki and S. X. Wang, "A performance model of disk array storage systems," in *The Computer Measurement Group's 2000 International Conference*, (Orlando, Florida), December 2000.

[29] F. Douglis, P. Krishnan, and B. Marsh, "Thwarting the power-hungry disk," in *Proceedings of USENIX 1994 Winter Technical Conference*, pp. 293–306, January 1994.

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

[31] D. P. Helmbold, D. D. E. Long, and B. Sherrod, "A dynamic disk spin-down technique for mobile computing," in *Proceedings of the 2nd annual international conference on Mobile computing and networking*, pp. 130–142, 1996.

[32] Y.-H. Lu and G. D. Micheli, "Adaptive hard disk power management on personal computers," in *Proceedings of the IEEE Great Lakes Symposium*, pp. 50–53, Mar. 1999.

[33] R. Youssef, "RAID for mobile computers," Master's thesis, Carnegie Mellon University Information Networking Institute, Aug. 1995. Available as INI-TR 1995-3.

[34] H. Huang, W. Hung, and K. G. Shin, "Fs2: Dynamic data replication in free disk space for improving disk performance and energy consumption," in *20th ACM Symposium on Operating Systems Principles (SOSP'05)*, (Brighton, United Kingdom), October 23-26 2005.

[35] D. Bitton and J. Gray, "Disk shadowing," in *Proceedings of the 14th International Conference on Very Large Data Bases*, pp. 331–338, 1988.

[36] M. Y. Kim and A. N. Tantawi, "Asynchronous disk interleaving: Approximating access delays," *IEEE Transactions on Computers*, vol. 40, pp. 801–810, July 1991.

[37] E. K. Lee and R. H. Katz, "An analytic performance model of disk arrays," in *Proceedings of the ACM Sigmetrics Conference on Measurement and Modeling of Computer Systems* (B. D. Gaither, ed.), vol. 21-1 of *Performance Evaluation Review*, (New York, NY, USA), pp. 98–109, ACM Press, May 1993.

[38] A. Thomasian and J. Menon, "RAID5 performance with distributed sparing," *IEEE Trans. Parallel and Distrib. Systems*, vol. 8, no. 6, pp. 640–657, 1997.

[39] J. Menon and J. Kasson, "Methods for improved update performance of disk arrays," in *Proc. of 25th Intl. Conf. on System Sciences*, vol. 1, pp. 74–83, January 1992.

[40] M. Uysal, G. A. Alvarez, and A. Merchant, "A modular, analytical throughput model for modern disk arrays," in *Ninth International Symposium in Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS'01)*, (Cincinnati, Ohio), pp. 183–192, August 2001.

[41] E. Shriver, A. Merchant, and J. Wilkes, "An analytical behavior model for disk drives with readahead caches and request reordering," in *Proc. of Intl. Conf. on Measurement and Modeling of Computer Systems (SIGMETRICS)*, pp. 182–91, June 1998.