Energy Smart Management of Scientific Data

Ekow Otoo, Dron Rotem, Shih-Chiang Tsao
Lawrence Berkeley National Laboratory
University of California
Berkeley, CA 94720
{ejotoo, d_rotem, weafon}@lbl.gov

Abstract

Scientific data centers comprised of high-powered computing equipment and large capacity disk storage systems consume considerable amount of energy. Dynamic power management techniques (DPM) are commonly used for saving energy in disk systems. These involve powering down disks that exhibit long idle periods and placing them in standby mode. A file request from a disk in standby mode will incur both energy and performance penalties as it takes energy (and time) to spin up the disk before it can serve a file. For this reason, DPM has to make decisions as to when to transition the disk into standby mode such that the energy saved is greater than the energy needed to spin it up again and the performance penalty is tolerable. The length of the idle period until the DPM decides to power down a disk is called idleness threshold.

In this paper, we study both analytically and experimentally dynamic power management techniques that save energy subject to performance constraints on file access costs. Based on observed workloads of scientific applications and disk characteristics, we provide a methodology for determining file assignment to disks and computing idleness thresholds that result in significant improvements to the energy saved by existing DPM solutions while meeting response time constraints. We validate our methods with simulations that use traces taken from scientific applications.

keywords: Disk storage, Power management, File allocation, Scientific workload, Performance guaranttee

1 Introduction

The rapid growth in highly data-intensive scientific research has fueled an explosion in computing facilities and demand for electricity to power them. Several analysts are now predicting that energy costs will eventually outstrip the cost of hardware in data centers [2]. As a result, reducing energy costs at data centers has become the focus of multiple research efforts which are aimed at devising architectural strategies for energy efficient computing systems. Examples of projects that are currently underway include the GreenLight project at UC San Diego, DiskEnergy at Microsoft, GREEN-NET Project in INRIA and the Green Grid Consortium.

There are multiple components that contribute to the power consumption in a data center such as servers, storage, cooling, networks etc. However, recent papers estimate that about 25 -35 percent of the energy consumption at data centers is attributed to disk storage systems [9]. This percentage of disk storage power consumption will continue to increase, as faster and higher capacity disks are deployed with increasing energy costs and also as data intensive applications demand reliable on-line access to data resources.

Reducing the energy consumption of the disk storage system has been addressed in many recent research works. Research efforts are directed at several levels such as *physical device* level , *systems* level and dynamic power management (DPM) algorithms. At the physical device level, disk manufacturers are developing new energy efficient disks [23] and hybrid disks (i.e., disks with integrated flash memory caches).

At the system level, a number of integrated storage solutions such as MAID [5], PARAID [25], PERGAMUM [24] and SEA [26] have emerged all of which are based on the general principle of spinning down and spinning up disks. Disks configured either as RAID sets or as independent disks, are configured with idle time-out periods, also called *idleness threshold*, after which they are automatically spun down into a standby mode. A read or write I/O request targeted to a standby disk causes the disk to spin-up in order to service it. This of course comes at the expense of a longer response time to file access requests as well as a penalty in terms of energy costs.

Dynamic power management (DPM) algorithms have been proposed to determine online when the disk should be transitioned to a lower power dissipation state while experiencing an idle period. Analytical solutions to this online problem have been evaluated in terms of their competitive ratio. This ratio is used to compare the energy cost of an online DPM algorithm to the energy cost of an optimal offline solution which knows the arrival sequence of disk access requests in advance. It is well known [13] that for a two state system where the disk can be in either standby or in idle mode there is a tight bound of 2 for the competitive ratio of any deterministic algorithm. This ratio is achieved by setting the idleness threshold, T_τ, to $\frac{\beta}{P_\tau}$ where β is the energy penalty (in joules) for having to serve a request while the disk is in standby mode, (i.e., spinning the disk down and then spinning it up in order to serve a request) and P_τ is the rate of energy consumption of the disk (in watts) in the idle mode. We call this value the *competitive idleness threshold*.

In this paper we focus mainly on read requests, we assume that write requests can be handled efficiently by using any one of the energy-friendly approaches presented in the literature. For example, in [24] it is recommended that files will be written into an already spinning disk if sufficient space is found on it or write it into any other disk (using best-fit or first-fit policy) where sufficient space can be found. The written file may be re-allocated to a better location later during a reorganization process. Another recently proposed strategy for energy saving for writes is called Write Off-Loading [18]. This technique allows write requests on spun-down disks to be temporarily redirected reliably to persistent storage elsewhere in the data center.

1.1 Contributions of this paper

In this paper, we quantify the effects of disk power management on response time based on request workloads and disk characteristics. To the best of our knowledge, with the exception of the work in [28], very little work has been done on modeling and analyzing the effects of power management on the response time for file access requests using realistic workloads and disk characteristics. In addition, the trade-off associated with using more or less disks on power consumption and response times has not been studied.

More specifically, our goal is to produce useful tools that can help in determining when power saving policies should be used at all as well as optimal idleness thresholds and bounds on the number of required disks needed in order to provide response time guarantees. Our main contributions are:

- We develop an analytic model of file requests served by a disk equipped with power saving mechanisms
- Based on this model, we develop a procedure called SmartIdle that computes several important parameters such as request arrival rate ”break-even” point that determines when power saving should be applied, how many disks must be used to support response time constraints, and optimal idleness thresholds
- We validate this procedure by applying it to two real life scientific application traces and show significant improvement over common existing DPM strategies that use *competitive idleness threshold* value to power down the disk.

The remainder of the paper is organized as follows. More details about related relevant work are provided in Section 2. In Section 3 our analytical model is described. In Section 4 we present our procedure for determining system parameters for maximizing energy savings while meeting performance requirements. In

Section 5 we present our simulation model and results and in Section 6 an application of our model on two scientific workloads is presented. Finally in Section 7 we present our conclusions and directions for future work.

2 Related Work

Conserving energy in large scale computing has been recently explored in [8, 19]. Colarelli and Grunwald [5] proposed *MAID* for near-line access to data in a massively large disk storage environment. They show, using simulation studies, that a MAID system is a viable alternative and capable of considerable energy savings over constantly spinning disks. A related system was implemented and commercialized by COPAN systems [7, 8]. This system, which is intended for a general data center, is not focused on scientific applications and is not adaptively reconfigurable based on workloads.

The theory of Dynamic Power Management of disks has drawn a lot of attention recently from the theoretical computer science community (see [13] for an extensive overview of this work). Most of this work considers a single disk only and attempts to find an optimal idle waiting period (also called idleness threshold time) after which a disk should be moved to a state which consumes less power. More specifically, the problem discussed in these research works is based on the assumption that the disk can be transitioned among n power consumption states where the i^{th} state consumes less power than the j^{th} state for $i < j$. The disk can serve file requests only when it is in the highest power state (the n^{th} state) which is also called the active state. The system must pay a penalty β_i if a request arrives when the disk is in the i^{th} state, the penalty is proportional to the power needed to spin up from state i to the active state n. The penalty is decreasing with the state number, i.e., $\beta_j < \beta_i, for j > i, and \beta_n = 0$.

The problem is that of devising online algorithms for selecting optimal threshold times, based on idle periods between request arrivals, to transition the disk from one power state to another. The most common case transitions between two states namely, idle state (full power) and standby or sleep state (zero power). The quality of these algorithms is measured by their competitive ratio which compares their power consumption to that of an optimal offline algorithm that can see the entire request sequence in advance before selecting state transition times. As mentioned before, for a two state system there is a tight bound of 2 for the competitive ratio of any deterministic algorithm.

There are also several results showing that with randomized online algorithms, the best competitive ratio achievable improves to $e/(e-1) \approx 1.58$ [14]. Response time penalty is not considered in these works. Another approach to DPM [12], attempts to learn the request arrival sequence probability based on previous history and then generates a probability-based DPM strategy that minimizes the expected power dissipation. It is known that power management schemes have an effect on the response time of the system. In [21] an upper bound on the additional latency of the sys- tem introduced by power management strategies is established.

More recently, it has been suggested that energy efficiency issues should become a first-class performance goal for query processing in large data base management systems. Several research papers deal with energy efficiency in DBMS using several benchmarks. Examples include the JouleSort [22]and SPECPower benchmarks which measure energy efficiency of entire systems that perform data management tasks.

In [20], the authors develop a power consumption model based on data from the TPC-C benchmark. In [17], the authors provide a framework for trading off performance and power consumption of storage systems based on a graduated, distribution-based QoS model. This work deals with workload profiling partitioning and scheduling to reduce energy consumption. In [10] energy-efficiency optimizations within database systems are discussed. The experiments in [10] use a decision support workload (TPC-H) which scans an entire table and applies a predicate to it. In [15] techniques for reducing power consumption in DBMS are introduced. One such technique, called QED, uses well known query aggregation methods to leverage common components of queries in a workload to reduce accesses to the storage system. The technique involves some performance penalties as it is done by delaying some queries in order to increase

such leveraging opportunities. Other energy conservation techniques proposed are addressed in [3, 18, 19, 24, 25, 28].

3 Model

3.1 Definitions and Notations

In the section, we apply the M/G/1 queuing model, similar to the approach in [16, 26, 28], to estimate the power cost and response time for a disk with a specific exponential arrival rate of file access and idleness threshold. Table 1 displays the notations and parameters used in the model. The values for the parameters of disk, e.g. T_d, T_u, P_u, and P_d, are given based on the specification in [23].

Table 1: Notations and Disk Parameter values

Name	***Notation***	***Default Value***
D_{PS}	disk with power-saving mode	
D_{NPS}	disk without power-saving mode	
G_{PS}	Energy cost in one cycle by D_{PS} (J)	
T_{PS}	The length of one cycle for D_{PS} (s)	
P_{PS}	Power cost of D_{PS} (W)	E_{PS}/T_{PS}
P_{NPS}	Power cost of D_{NPS} (W)	
P	P_{PS}/P_{NPS}	
T_τ	Idleness Threshold (s)	$10 \sim 500$
T_d	Time to spin down a disk (s)	10
T_u	Time to spin up a disk (s)	15
P_d	Power to spin down a disk (W)	9.3
P_u	Power to spin up a disk (W)	24
P_τ	Power in idle mode (W)	9.3
P_a	Power in active mode (W)	13.0
P_{sby}	Power in standby mode (W)	0.8
G_{du}	Energy to spin down and up a disk (J)	$P_d \times T_d + P_u \times T_u$
T_a	Length of a busy period entered from an idle state (s)	
$f_{T_a}(x)$	Length of a busy period entered from an x-second warm-up state, consisting of partial T_d and the whole T_u (s)	
λ	arrival rate of file access (1/s)	$0.1 \sim 0.001$
ρ	traffic intensity for the disk	$\lambda \times E[S]$
$E[S]$	Mean service time of a file (s)	7.56s
$E[S^2]$		178.05s

3.2 Power Cost

In this section $E[Y]$ denotes the expected value of the variable Y. In the following section we present an anlytical model for estimating the power costs for D_{PS} and D_{NPS} within one cycle of power mode transitions of a disk, where one cycle represents the time from the end of a busy period to that of the next busy period. Since one cycle of D_{NPS} must consist of an idle period and a busy period, we can express the mean value of

P_{NPS} as

$$E[P_{NPS}] = \frac{P_\tau \times 1/\lambda + P_a T_a}{1/\lambda + T_a}.$$

However, for D_{PS} there are three different patterns in one cycle, as shown in Figure 1 where t represents the time from the end of the last busy period to the arrival of the next request. Recall that under an M/G/1 model, if the arrival rate is λ, the time between two busy periods, t, is an exponential distribution with mean = $1/\lambda$. T_a denotes the length of a busy period entered from an idle state. We know that the mean of T_a under an M/G/1 model can be expressed as

$$E[T_a] = E[S]/(1-\rho);$$

where S denotes the service time and ρ is the traffic intensity. Also, $f_{T_a}(X)$ represents the length of a busy period entered from an x-second spinning-up state, consisting of partial T_d and the whole of T_u. Under the M/G/1 model with setup time, the mean of $f_{T_a}(X)$ can be written as

$$E[f_{T_a}(x)] = (1+\lambda \times x)E[T_a];$$

where x is the setup time [9], i.e., equivalent to the spin-up time in this work.

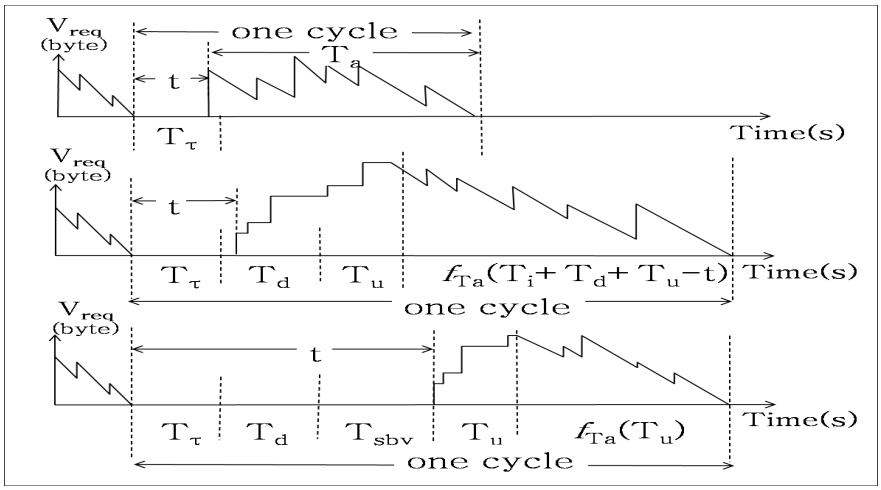


Figure 1: The three possible patterns in a cycle for DPS. The vertical axis, V_{req} represents the total volume, in bytes, of unserviced requests

Since each case has different occurrence probability in a period of one cycle, $E[P_{PS}]$ can be expressed as

$$E[P_{PS}] = \frac{E[G_{PS}]}{E[T_{PS}]} = \frac{\sum_{i=1}^{3} E[G_{PS}^i]P^i}{\sum_{i=1}^{3} E[T_{PS}^i]P^i};$$

where $E[G_{PS}^i]$, $E[T_{PS}^i]$, and P^i are the mean energy cost, mean time period and probability of a request arrival in Case i, respectively. The following three Cases occur depending on the arrival of a request. The details of power cost and response times calculations are given in Appendix A.

Case 1, $t < T_\tau$**:** This case indicates that a request arrives while the disk is idle but before the idle period reaches the Idleness threshold value for it to begin spinning-down.

Case 2, $T_\tau \leq t < T_\tau + T_d$**:** Here an request arrives when the disk has been long enough past its idleness threshold; it is in the process of spinning down but has not completely spun-down.

Case 3, $(T_\tau + T_d) \leq t$**:** In this case the request arrives after the disk has completely spun-down.

The mean response times can similarly be estimated for disks operating in power saving mode D_{PS}. We can calculate the mean sojourn time θ of a request, i.e., its response time, by calculating the mean $E[\theta]$ for each case. The details of these calculations are given in Appendix A.

3.3 Numerical Results

In this section, we illustrate our methods using the disks and workload characteristics in Tables 1 and 2. Similar figures can be obtained using the analytical model developed. Figure 2 plots the relationship between $E[P_{PS}]/E[P_{NPS}]$ and λ for $T_\tau = 0, 10, 53,$ and160. When $\lambda < 0.029$, P_{PS}/P_{NPS} would be smaller than 1, i.e., the power-saving mechanism is efficient since it is below the threshold of λ. Figure 3 plots the corresponding values for response times including the case for $T_\tau = \infty$. We note that $T_\tau = 53$ is the *competitive idleness threshold* in our case obtained by $G_{du}/(P_\tau - P_{sby})$.

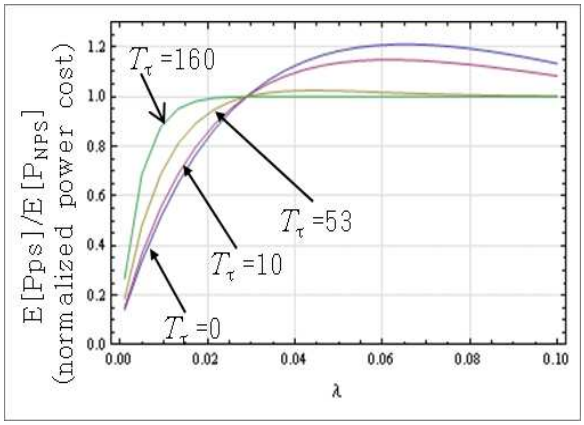


Figure 2: Relationship between the ratio of $E[P_{PS}]/E[P_{NPS}]$ and arrival rate λ for $T_\tau = 0, 10, 53$ and 160s.

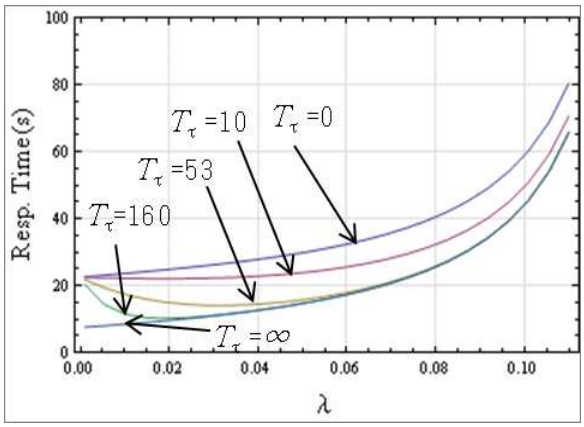


Figure 3: Graphs of response time vs. arrival rate λ, for $T_\tau = 0, 10, 53, 160$ and ∞ sec.

From Figure 2, we note that when $\lambda > 0.029$, the D_{PS} disk have a normalized power cost larger than 1. That is, when the arrival rate of files in a power-saving (D_{PS}) disk is larger than 0.029, then its power saving features should be turned off to avoid incurring more power cost than a non-power-saving (NPS) disk.

4 Procedure for Selecting Parameters of Disk Storage Configuration

4.1 Procedure

Figure 3 describes the relationship between the arrival rate of requests to one disk and their corresponding expected response time. In Figure 4 the curves $\theta_{PS}(\lambda, 0)$ and $\theta_{PS}(\lambda, \infty)$ represent the mean response times of disks hit with request arrivals at rate λ for $T_\tau = 0$ and $T_\tau = \infty$ respectively. The entire space covered by Figure 4 is divided into 5 areas based on the following rules. Let θ' denote the required constraint, by the user, on the response time. We will use this figure to describe our procedure called *SmartIdle* presented as Procedure 1. The procedure will determine the necessary number of active disks and the idleness threshold for these disks. Let R denote the total arrival rate of requests to the system, and suppose the minimum number of disks required to hold the entire set of files is N. The procedure first computes the arrival rate for a single disk, $\lambda = R/N$. Given a point X with coordinates $\langle \lambda, \theta' \rangle$ which represents a combination of request arrival rate and the required response time in Figure 4, we will show how to calculate the actual number of active disks and the idleness threshold based on the area that contains this point. First, if $\lambda \geq 0.029$, since power-saving mechanism is inefficient according to Figure 2, the procedure will suggest spinning disks for

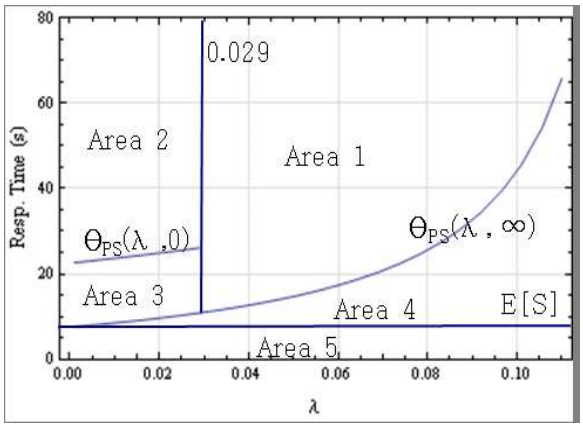


Figure 4: Five areas of the procedure

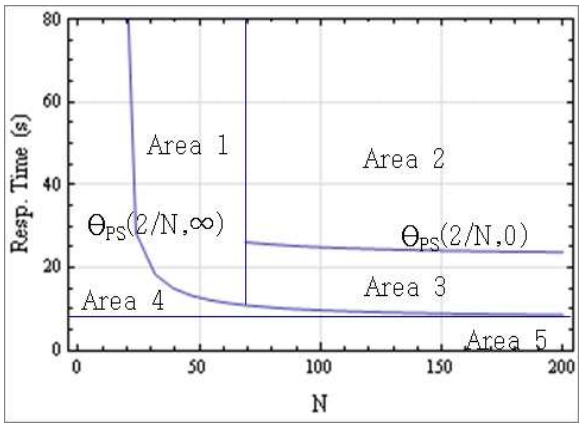


Figure 5: An alternative plot of 4, when $R = 2$ per sec

the whole time, i.e., set *i* the idleness threshold to ∞. Thus, for any θ' larger than $\theta_{PS}(\lambda,\infty)$ and $\lambda > 0.029$ (i.e., Area 1), we know N disks are enough to meet such constraints because they can offer $\theta_{PS}(\lambda,\infty)$ of response time, which is smaller than θ'. Note that, in this case using less than N disks, in order to save power is not feasible because N, is the minimum of disks necessary for storing the entire data.

Second, if $\lambda < 0.029$ but $\theta' > \theta_{PS}(\lambda,0)$ i.e., the point X falls in Area 2 of Figure 4. We know that such a constraint can be satisfied even when the disk is spun down if there are no requests pending for service. Thus, N disks are enough and their idleness threshold will be set to 0 to save the most power. Using more than N disks is not useful because the constraint on response time is always satisfied. In the case that $\theta_{PS}(\lambda,\infty) < \theta' < \theta_{PS}(\lambda,0)$, (i.e. X lies in Area 3), it is necessary to carefully get an idleness threshold T_τ that satisfies $\theta_{PS}(R/N,T_\tau) = \theta'$.

Third, if the given $\theta' < \theta_{PS}(\lambda,\infty)$, (i.e., Area 4), we have that the arrival rate is too high for each disk to finish serving a file within θ' even when the power-saving mechanism is disabled to avoid the additional delay of spinning-up and spinning-down disks. In this case, more than N disks are needed to obtain a $\lambda = R/N$ to be sufficiently low to satisfy $\theta_{PS}(\lambda,\infty) = \theta'$. Finally, if the point X falls in Area 5, the given θ' is not feasible to be satisfied because it is smaller than the service time. The *Procedure* 1, describes the process of estimating the required number of disks and expected response times for configuring a system of disks given a usage workload and specific disks characteristics.

4.2 Illustration

The following gives an example on how to use the procedure. Assuming the arrival rate of requests is fixed at 2 per sec, we can redraw Figure 4 to show the relationship between N and the response time. This is now shown in Figure 5. Observe that Figure 5 can be considered a Y-Axis mirror image of Figure 4. For $N = 50$, if a response time within 20 seconds is desired, then Area 1 of Figure 5 should be used. This implies that the files should be stored on 50 disks which are kept constantly spinning. However, if an average response time less than 10 seconds is desired, then X falls within Area 4. In this case we compute N such that $\theta_{PS}(2/N,\infty) = 10$. The solution is given by $N = 80$. This means that files should be distributed over 80 instead of 50 disks that are constantly spinning.

Next, if $N = 100$ and the response time constraint is set at 15 sec, then the point falls in Area 3. To get the possible idleness thresholds, we examine, from Figure 6 those curves that cross the line of $\theta = 15$ when $N \geq 100$. Such a position is marked by the symbol X, where the two curves $\theta_{PS}(2/N,53)$ and $\theta_{PS}(2/N,80)$ cross the line. So we still use 100 disks to pack the files and set the idleness threshold at 53 seconds, i.e., the smaller of 53 and 80. This saves more power while meeting the 15-sec constraint. However, if the response time constraint is 40-sec, the point falls in Area 2 In this case we distribute files into the 100 disks again,

Procedure `SmartIdle(`$R, N, \theta', \lambda, i$`)`

Input:

R : The total arrival rate of files to the system.

N : The minimum number of disks to store the data.

θ' : The constraint on the response time.

λ : Arrival rate of file requests to a disk.

i : The idleness threshold.

Output:

P : The expected mean power cost

θ : The expected mean response time

$\theta_{PS}(\lambda, i)$ defines a function for the mean response time of disk hit with request arrival rate λ and an idleness threshold of i. ;

Set $\lambda \Leftarrow R/N$; $X \Leftarrow$ coordinates$\langle \lambda, \theta' \rangle$;

switch Area that X lies **do**

case Area 1

Pack files into N disks that are never spun-down;

Set $P \Leftarrow N * P_{NPS}(a)$; $\theta \Leftarrow \theta_{NPS}(\lambda)$ alternatively $\theta \Leftarrow \theta_{PS}(\lambda, \infty)$;

break ;

case Area 2

Pack files into N disks ;

Set $i \Leftarrow 0$; $P \Leftarrow N * P_{PS}(\lambda, 0)$; $\theta \Leftarrow \theta_{PS}(\lambda, 0)$;

case Area 3

Set idleness threshold i that satisfies $\theta_{PS}(R/N, i) = \theta'$;

Pack files into N disks ;

Set $P \Leftarrow N * P_{PS}(\lambda, i)$; $\theta \Leftarrow \theta_{PS}(\lambda, i)$;

case Area 4

Left shift the point X until it intersects the curve ;

Pack files into M disks such that M satisfies $\theta_{PS}(R/N, \infty) = \theta'$; `// These are disks that are never spun-down`

Set $P \Leftarrow M \times P_{NPS}(R/M)$; $\theta \Leftarrow \theta_{NPS}(R/M)$;

case Area 5

No solution can satisfy the specified constraints ;

but set the idleness threshold to zero. This means a disk is spun-down as soon as no requests are pending for service. This not only saves power but also provides 25 seconds of average response time.

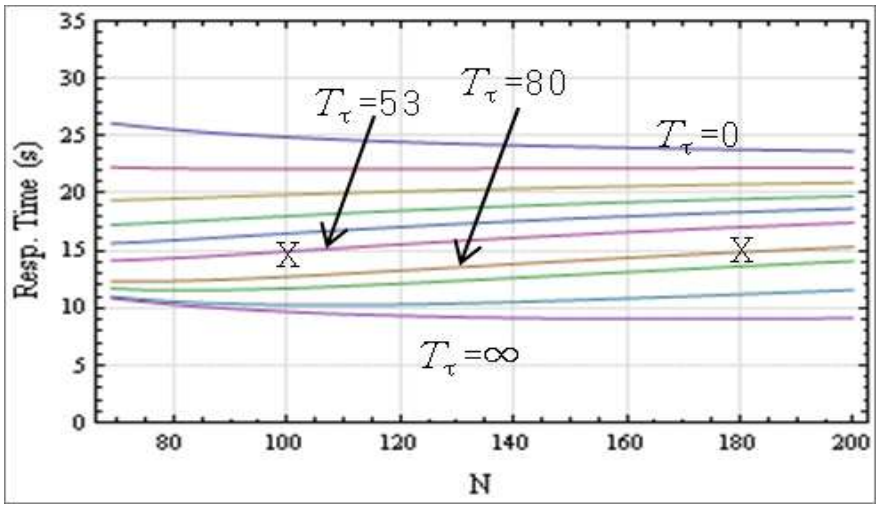


Figure 6: A enlarged area 3 for multiple curves of $\theta_{PS}(2/N, 1)$

5 The Simulation

We developed a simulation model to examine the model proposed in Section 3 and compared this with the SmartIdle procedure described in Section 4. The simulation environment was developed and tested using SimPy [11], as illustrated in Figure 7. The environment consists of a workload generator, a file dispatcher, and a group of hard disks.

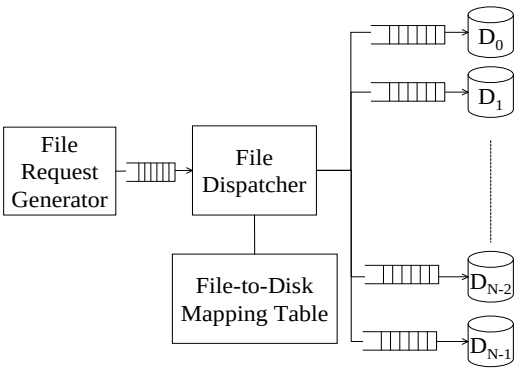


Figure 7: The configuration of disks in the simulation

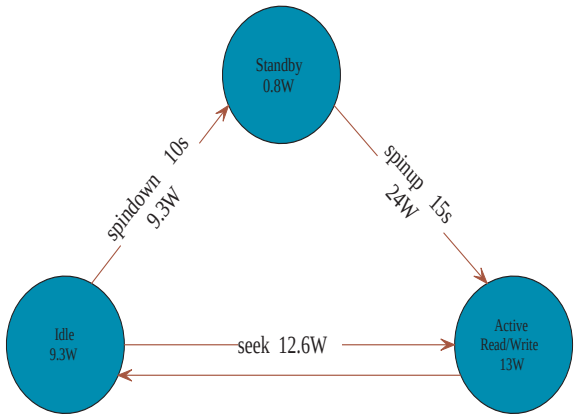


Figure 8: Power consumption of disks in different modes

5.1 Hard Disk Characteristics

Table 2 shows the characteristics of hard disk used in the simulation. With the specifications taken from [23] and [27] we built our own hard disk simulation modules. A hard disk is spun down and set into standby mode (see Figure 8) after it has been idle for a fixed period which is called *idleness threshold* [6, 19]. We do not

use the recently revised DiskSim simulator [4], that is commonly used in the literature for our simulations because, it still provides only old and small disk models, e.g., 1998's 8GBytes disks, and the number of events needed to handle a file request is highly correlated with file sizes making DiskSim too slow for a realistic data center simulation that involves disks, each of the order of 500 GBytes and tens of thousands of files requiring terabytes/petabytes of total data storage.

Table 2: Characteristics of the Hard Disk

Description	***Value***	***Description***	***Value***
Disk model	Seagate ST3500630AS	Standby power	0.8 Watts
Standard interface	SATA	Active power	13 Watts
Rotational speed	7200 rpm	Seek power	12.6 Watts
Avg. seek time	8.5 msecs	Spin up power	24 Watts
Avg. rotation time	4.16 msecs	Spin down power	9.3 Watts
Disk size	500GB	Spin up time	15 secs
Disk load (Transfer rate)	72 MBytes/sec	Spin down time	10 secs
Idle power	9.3 Watts		

5.2 Workload Generator

The workload generator supports two different ways to produce file requests. First, the generator can produce requests based on a log of file accesses to a storage system. We extract the distribution of file file sizes and the arrival time of each request from the real workload. Second, the generator can follow a Poisson process to produce requests at a rate R to get files specified in a given list. The sizes of the files in the list are generated based on a Zipf distribution whose probability distribution is given by

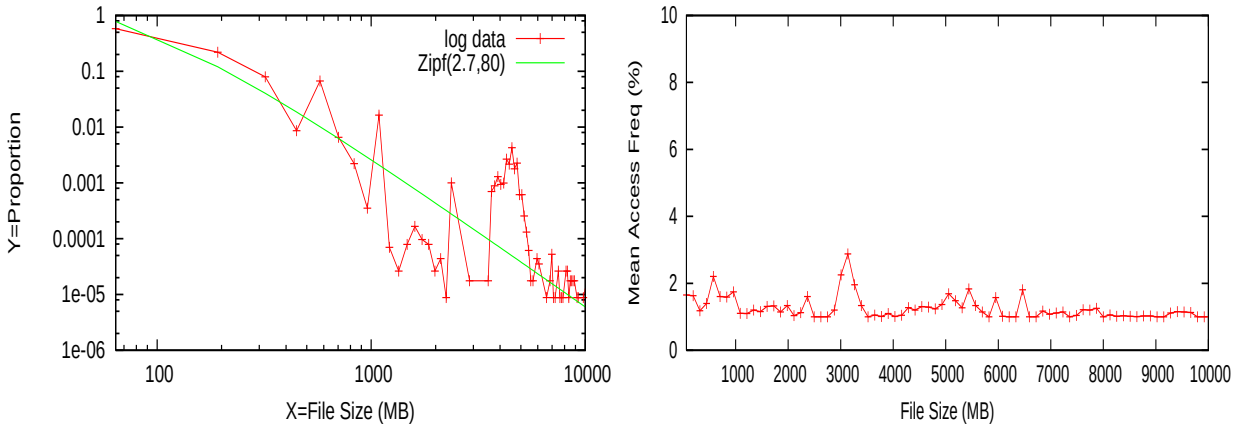
$$P(x) = \frac{x^{-K}}{\zeta(N,K)}; \quad \text{where} \quad \zeta(N,K) = \sum_{i=1}^{N} i^{-K}. \tag{1}$$

Also, the generator can control the frequency of requests to each file. To determine reasonable parameters for the Zipf distribution that are close to the actual data accesses, we logged the file requests to the NERSC's High Performance Storage System (HPSS) for 30 days (between May 31 and June 29, 2008). There were 88,631 files accessed in the 115,832 read requests. The mean size of the files requested was 544 MB. This requires 7.56 sec to service a file if these files were to be accessed from a disk storage systems with disk transmission rate of 72MBps. The minimum space required for storing all the requested files is 95 disks. Next we classified the 88,631 files into 80 bins based on their sizes, where the width of each bin is 128MB. We then compute the proportion of the number of files in each bin compared with the total number of files. Figure 9(a) plots these proportions for the 80 bins. Each point $z\langle X,Y\rangle$ in Figure 9(a), represents the proportion Y of files with sizes in the interval $(X-64, X+64]$ in MBytes. As we can see this distribution is closely related to the Zipf distribution because the proportion decreases almost linearly in the log-log scale of the axes. Figure 9(b) shows the relationship between the sizes of files and their corresponding access frequencies. In this analysis of accesses to NERSC datasets, the access frequencies of files are independent of the sizes. We can therefore assume that each file has the same access frequency f.

5.3 File Dispatcher and Mapping Table

Once a request is generated, the file dispatcher forwards it to the corresponding disk based on the file-to-disk mapping table. Files are randomly mapped to a specific number of disks. The number of disks and idleness thresholds are determined by the procedure proposed in Section 4. For the purpose of comparing the power

consumptions of the DPM strategy, we also generated a mapping table that maps files randomly to all disks and fix the idleness threshold at 0, 53, 160 secs or ∞, i.e., without enabling the power saving features of the disk. The time to map a file to disk by the dispatcher is ignored since it is negligible.



(a) Log-log plot of workload distribution

(b) Plot of access frequency of workload

Figure 9: Analysis of workload from NERSC

6 Experimental Results

6.1 Evaluation of the Model

We first evaluate the correctness of our analytical model proposed in Section 3. Only one disk is used in this scenario. The service time of requests has the same distribution and parameters (see Table 2), as those assumed in the analysis. The simulation ends when it has served 30000 requests. Figures 10 and 11 show the normalized power cost of the disk and the response time of requests under different values of arrival rate and idleness threshold respectively. By respectively comparing the two figures with Figures 2 and 3, we can validate our analysis for power cost and response time.

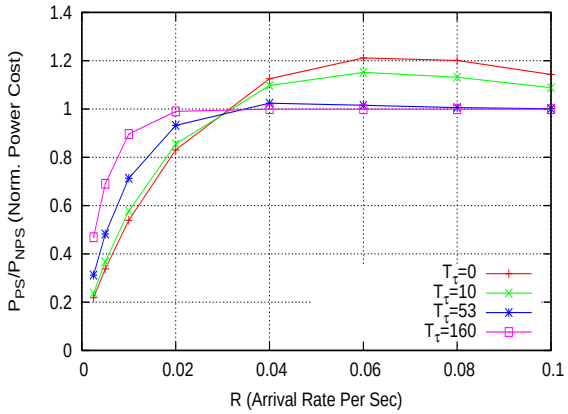


Figure 10: Power cost at different arrival rates

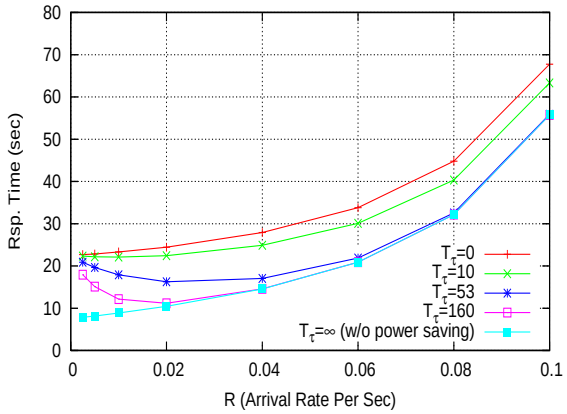


Figure 11: Response times at different arrival rates

6.2 Constraints on Response Time

Next, we examine whether the procedure *SmartIdle* determines the suitable number of active disks and idleness threshold to meet the response time constraints while still saving power compared with our analytical model. Suppose we consider using 100 disks and set $N = 40$ and try to satisfy a response time constraint of 20 seconds. For comparison, we also plotted the response times and power saving ratio when the disk idleness threshold is fixed at 53 seconds, which as mentioned before is the *competitive idleness threshold* in our case.

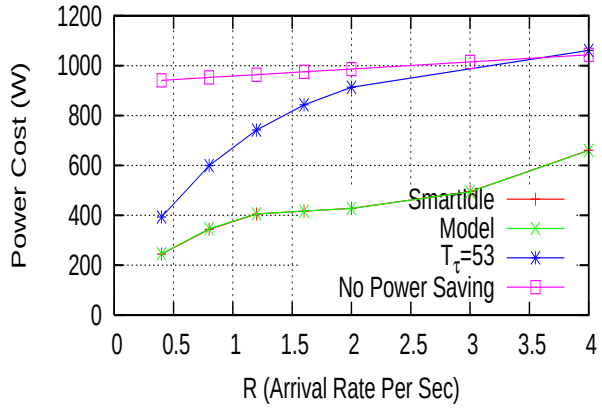


Figure 12: Power savings with constraint $\theta' = 20$ and $N = 40$.

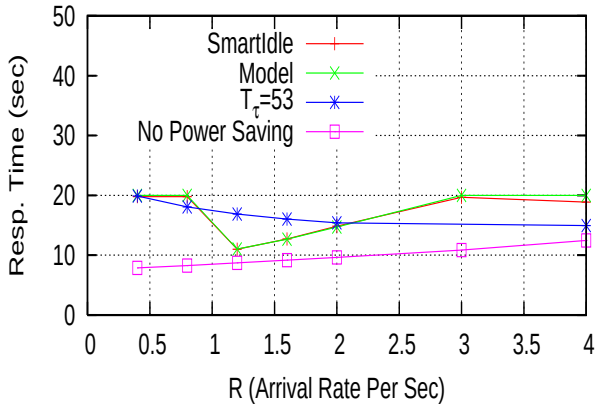


Figure 13: Response time with constraint $\theta' = 20$ and $N = 40$

Figures 12 and 13 show the power saving and response time plots for a response time constraint of 20-sec. The SmartIdle procedure design satisfies the response time constraint while saving 60% of power on average. The *SmartIdle* procedure results in a much shorter response time than the specified constraint for arrival rates ranging from 0.8 to 2.5 and yet saves more power than with fixed 53-sec idleness threshold. In this interval of arrival rates the point $\langle \lambda, \theta' \rangle$ always falls in Area 1 and the SmartIdle procedure suggests that the disks be kept spinning at all times, instead of spinning down after an idleness threshold. Spinning down a disk after a fixed idleness threshold is inefficient since it not only results in a longer response time but also incurs more power cost than simply spinning disk.

6.3 Using Trace Logs of Scientific Data Accesses

In this subsection we test whether the SmartIdle procedure can be used to derive the configuration of disks that satisfy the constraint of response time when apply the request arrival rate extracted from a real workload. Suppose we consider the use of 200 disks with the minimum number required set as $N = 96$. The arrival rate of requests is 0.044683. Figure 14 shows the ratio of power savings obtained from using the analytical model, the *SmartIdle* procedure and fixed idleness threshold of 53 secs. The initial idleness threshold used varies from 8 to 35 seconds. Figure 15 shows the corresponding response times of obtained in each design configuration. From Figure 15, we find that disks configuration obtained from applying the *SmartIdle* not only meets the constraints, but also provides response time far less than the initial response time constraints ranging from 8 to 25 sec. Further we see that from Figure 14 that we achieve more power savings than that the expected savings achievable from the analytical model.

In addition to the NERSC workload, we also tested our *SmartIdle* procedure with workloads from the BaBar project [1]. The BaBar project is a high energy physics experiment with over 600 world-wide collaborators from 75 institutions in 10 countries. The data for this experiment is stored at the Stanford Linear Accelerator Collider (SLAC) site. There are about 86,378 distinct files stored which will require at least

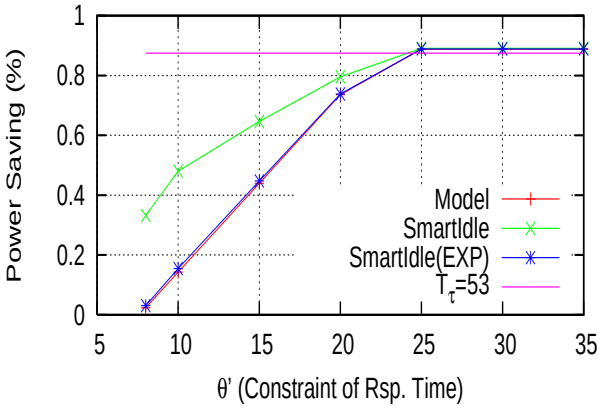


Figure 14: Power savings between SmartIdle and fixed idleness threshold values for NERSC data

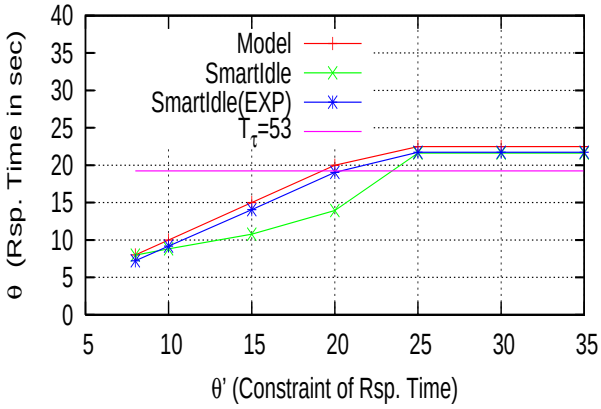


Figure 15: Response times of SmartIdle and fixed idleness threshold values for NERSC

123 disks of 500GB to store them. The trace log of file requests for Oct 1, 2004 was used in this study. It contained 93,172 read requests and involved 10,735 distinct files. The average arrival rate (per second) of the requests is 1.07838, which is much higher than that in the workload of NERSC. The mean size of files accessed by these requests is 1,235 MB, which requires about 16.5138 sec of mean service time, $E[S]$, and 332.438 sec for $E[S^2]$ when the disk transmission rate is 72MBps and a single 32GB LRU cache is deployed in front of all disks.

Compared to the uniformly distributed accesses observed in the NERSC workload (Figure 9(b)), the workload shows 48% of requests accessing files with size larger than 1.6GB. Further observation shows these requests target only 783 files that constitute 7.3% of the files involved in all requests.

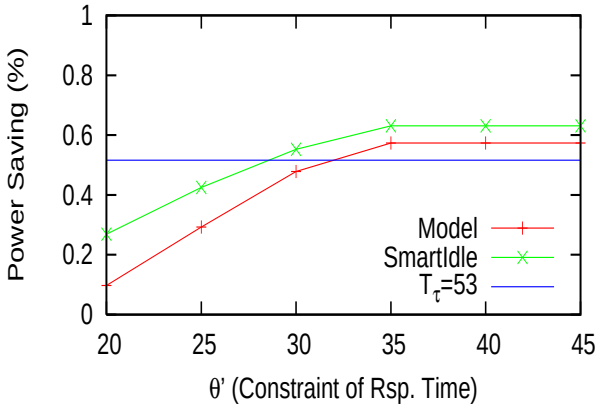


Figure 16: Power savings between SmartIdle and fixed idleness threshold values for BaBar

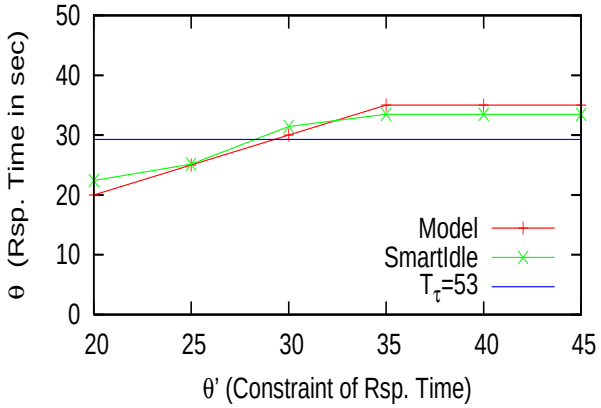


Figure 17: Response times of SmartIdle and fixed idleness threshold values for BaBar data

Figure 16 shows the ratio of power savings incurred while Figure 17 shows the response time of disks when their idleness threshold are configured by SmartIdle for constraints varying from 20 to 45 seconds. From the two figures, we again find that disk configuration derived from using *SmartIdle* not only meets the constraints of the response times, but further gives a greater saving than expected by the analytical model.

7 Conclusion and Future Work

In this paper we developed an analytic model to analyze the interaction of file access workload with a disk system that uses power saving mechanisms. The model allowed us to devise a procedure that allows design-

ers to accurately evaluate the trade-offs between energy consumption and response time. The procedure can be used to determine whether the required response times are achievable by the current system and what are the associated energy costs. The procedure also allows designers to tune the performance of the system by adding or subtracting disks as well as determining idleness thresholds. Using the procedure on simulated data as well as real life work logs showed significant improvement in energy costs over commonly used DPM strategies.

Additional work also needs to be done to make dynamic decisions about migrating files between disks if it is discovered that the arrival rates to disks deviate significantly from the initial estimates used as an input to the SmartIdle procedure. We also plan to investigate our techniques with more real life workloads that include various mixes of read and write requests. In addition, we will also investigate the effects of various caching strategies as we believe that cache size and cache replacement policies may significantly affect the trade-off between power consumption and response time.

Acknowledgment

This work is supported by the Director, Office of Laboratory Policy and Infrastructure Management of the U. S. Department of Energy under Contract No. DE-AC02-05CH11231. This research used resources of the National Energy Research Scientific Computing (NERSC), which is supported by the Office of Science of the U.S. Department of Energy.

References

[1] BaBar. The babar collaboration http://www.slac.stanford.edu/bfroot/.

[2] Luiz André Barroso. The price of performance. Queue, 3(7):48–53, 2005.

[3] T. Bisson, S. A. Brandt, and D. D. E. Long. A hybrid disk-aware spin-down algorithm with I/O subsystem support. In Proc. Perf., Comput., and Com. Conf., (IPCCC'07), pages 236–245, New Orleans, LA, May 2007.

[4] J. Bucy, J. Schindler, S. Schlosser, and G. Ganger. The disksim simulation environment.

[5] Dennis Colarelli and Dirk Grunwald. Massive arrays of idle disks for storage archives. In Supercomputing'02: Proc. ACM/IEEE Conference on Supercomputing, pages 1 – 11, Los Alamitos, CA, USA, 2002. IEEE Computer Society Press.

[6] Carlos Cunha, Azer Bestavros, and Mark Crovella. Characteristics of www client-based traces. Technical report, Boston University, Boston, MA, USA, 1995.

[7] A. Guha. A new approach to disk-based mass storage systems. In 12th NASA Goddard - 21st IEEE Conf. on Mass Storage Syst. and Tech., College Park, Maryland, 2004.

[8] A. Guha. Data archiving using enhanced maid (massive array of idle disks), May 15 – 18 2006.

[9] S. Gurumurthi, A. Sivasubramaniam, M. Kandemir, and H. Franke. Reducing disk power consumption in servers with drpm. Computer, 36(12):59–66, 2003.

[10] S. Harizopoulos, J. Meza M. A. Shah, and P. Ranganathan. The new holy grail of data management systems research. In CIDR'09: Proc. of the 4th Biennial Conf. on Innovative Data Syst. Research, New York, NY, USA, 2009. ACM.

[11] SimPy: SimPy Simulation Package in Python. http://simpy.sourceforge.net/archive.htm.

[12] S. Irani, R. Gupta, and S. Shukla. Competitive analysis of dynamic power management strategies for systems with multiple power savings states. In DATE '02: Proceedings of the conference on Design, automation and test in Europe, page 117, Washington, DC, USA, 2002. IEEE Computer Society.

[13] Sandy Irani, Gaurav Singh, Sandeep K. Shukla, and Rajesh K. Gupta. An overview of the competitive and adversarial approaches to designing dynamic power management strategies. IEEE Trans. VLSI Syst., 13(12):1349–1361, 2005.

[14] A. R. Karlin, M. S. Manasse, L. A. McGeoch, and S. Owicki. Competitive randomized algorithms for non-uniform problems. In SODA '90: Proceedings of the first annual ACM-SIAM symposium on Discrete algorithms, pages 301–309, Philadelphia, PA, USA, 1990. Society for Industrial and Applied Mathematics.

[15] W. Lang and J. M. Patel. Towards ecofriendly database management systems. In CIDR'09: Proc. of the 4th Biennial Conf. on Innovative Data Syst. Research, New York, NY, USA, 2009. ACM.

[16] Lin-Wen Lee, Peter Scheuermann, and Radek Vingralek. File assignment in parallel I/O systems with minimal variance of service time. IEEE Transactions on Computers, 49(2):127–140, February 2000.

[17] L. Lu and P. Varman. Workload decomposition for power efficient storage systems. In Proc. Workshop on Power Aware Computing and Systems (HotPower '08), San Diego, CA, USA, Dec. 2008.

[18] D. Narayanan, A. Donnelly, and Antony Rowstron. Write off-loading: Practical power management for enterprise storage. In Proc. 6th USENIX Conf. on File and Storage Technologies (FAST'2008), pages 253 – 267, San Jose, California, Feb. 2008.

[19] E. Pinheiro and R. Bianchini. Energy conservation techniques for disk array-based servers. In Proc. Int'l. Conf. on Supercomputing (ICS'04), Saint-Malo, France, June 26 2004.

[20] Meikel Poess and Raghunath Othayoth Nambiar. Energy cost, the key challenge of today's data centers: a power consumption analysis of tpc-c results. Proc. VLDB Endow., 1(2):1229–1240, 2008.

[21] Dinesh Ramanathan, Sandy Irani, and Rajesh Gupta. Latency effects of system level power management algorithms. In ICCAD '00: Proceedings of the 2000 IEEE/ACM international conference on Computer-aided design, pages 350–356, Piscataway, NJ, USA, 2000. IEEE Press.

[22] S. Rivoire, M. A. Shah, P. Ranganathan, and C. Kozyrakis. Joulesort: a balanced energy-efficiency benchmark. In SIGMOD '07: Proceedings of the 2007 ACM SIGMOD international conference on Management of data, pages 365–376, New York, NY, USA, 2007. ACM.

[23] Seagate. Seagate Barracuda 7200.10 Serial ATA Product Manual. Seagate Technology, Scotts Valley, CA, Dec 2007.

[24] M. W. Storer, K. M. Greenan, and E. L. Miller. Pergamum: Replacing tape with energy efficient, reliable, disk-based archival storage. In Proc. 6th USENIX Conf. on File and Storage Technologies (FAST'2008), pages 1 – 16, San Jose, California, Feb. 2008.

[25] C. Weddle, M. Oldham, J. Qian, and A. Wang. PARAID: A gear shifting power-aware RAID. ACM Trans. on Storage (TOS), 3(3):28 – 26, Oct. 2007.

[26] Tao Xie. Sea: A striping-based energy-aware strategy for data placement in RAID-structured storage system. IEEE Transactions on Computers, 57(6):748–769, 2008.

[27] J. Zedlewski, S. Sobti, N. Garg, F. Zheng, A. Krishnamurthy, and R. Wang. Modeling hard-disk power consumption. In FAST'03: Proc. 2nd USENIX Conf. on File and Storage Tech., pages 217–230, Berkeley, CA, USA, 2003. USENIX Association.

[28] Q. Zhu, Z. Chen, L. Tan, Y. Zhou, K. Keeton, and J. Wilkes. Hibernator: Helping disk arrays sleep through the winter. In SOSP'05: Proc. 20th ACM Symp. on Operating Syst. Principles, pages 177–190, NY, USA, 2005. ACM Press.

Appendix A: Derivation of Results for Analytical Model

Expressions for computing the power costs and the mean response times are given below.

Case 1, $t < T_\tau$ **:**

$P^1 = \text{Prob}\{t < T_\tau\}$; $E[G^1_{PS}] = P_\tau E[T_1] + P_a E[T_a]$; and $E[T^1_{PS}] = E[T_1] + E[T_a]$;
where E[T_1] is $E[t]$ in Case 1 and can be written as $E[T_1] = \int_0^{T_\tau} t\,p(t)dt / \int_0^{T_\tau} p(t)dt$ where $p(t)$ is the probability density function (pdf) of t.

Case 2, $T_\tau < t < T_\tau + T_d$**:**

$P^2 = \text{Prob}\{T_\tau < t < T_\tau + T_d\}$; $E[E^2_{PS}] = P_\tau T_\tau + G_{du} + P_a E[T_2]$; and $E[T^2_{PS}] = T_\tau + T_d + T_u + E[T_2]$;
where $E[T_2]$ is the busy period in Case 2 and can be written as

$$E[T_2] = \frac{\int_{T_\tau}^{T_\tau+T_d} E[f_{T_a}(T_d + T_u - (t - T_\tau))]p(t)dt}{\int_{T_\tau}^{T_\tau+T_d} p(t)dt}.$$

Case 3, $(T_\tau + T_d) < t$**:**

$P^3 = \text{Prob}\{T_\tau + T_d < t\}$; $E[E^3_{PS}] = P_\tau T_\tau + P_d T_d + P_{sby} E[T_3] + P_u T_u + P_a E[f_{T_a}(T_u)]$; and $E[T^3_{PS}] = T_\tau + T_d + E[T_3] + T_u + E[f_{T_a}(T_u)]$;
where T_3 is standby time, $E[\mathrm{T}_{sby}]$, in Case 3 and can be written

$$E[T_3] = \frac{\int_{T_\tau+T_d}^{\infty}(t - T_\tau - T_d)p(t)dt}{\int_{T_\tau+T_d}^{\infty} p(t)dt}.$$

Mean Response Time

Similarly, by the above model for D_{PS}, we can calculate the mean sojourn time θ of a request, i.e. its response time, by averaging the $E[\theta]$ of each case as

$$E[\theta_{PS}] = \sum_{i=1}^{3} E[\theta^i]P^i;$$

where P^i and $E[\theta^i]$ are the probability and the mean request sojourn time, respectively during the cycle of case i. Besides, since a D_{NPS} does not spin down, we can simply regard its T_τ as ∞ and then get $E[\theta_{NPS}] = E[\theta^1]$. Next, recall that the sojourn time in M/G/1 [9], is

$$E[\theta] = \frac{\rho}{1-\rho}\frac{E[S^2]}{2E[S]} + E[S]; \tag{2}$$

and in M/G/1 with setup time X, the mean sojourn time $E[\theta_x]$ is

$$E[\theta_X] = \frac{\rho}{1-\rho}\frac{E[S^2]}{2E[S]} + \frac{\lambda^{-1}}{\lambda^{-1}+E[X]} + \frac{E[X]}{\lambda^{-1}+E[X]}\frac{E[X^2]}{2E[X]} + E[S]. \tag{3}$$

Then, according to the above two equations, we get $E[\theta^1]$, $E[\theta^2]$ and $E[\theta^3]$ as follows:

Case 1, $t < T_\tau$**:**

Based on equation 2, we have $E[\theta^1] = (\rho/(1-\rho))(E[S^2]/2E[S]) + E[S]$

Case 2, $T_\tau < t < T_\tau + T_d$**:**

From Case 2 of Subsection 3.2, we have that the setup time X_2, of Case 2 is $X_2 = T_2 = T_\tau + T_d + T_u - t$. So we get

$$E[X_2] = \frac{\int_{T_\tau}^{T_\tau+T_d}(T_i + T_d + T_u - t)p(t)dt}{\int_{T_\tau}^{T_i+T_d} p(t)dt}. \tag{4}$$

$$E[X_2^2] = \frac{\int_{T_\tau}^{T_\tau+T_d}(T_\tau + T_d + T_u - t)^2 p(t)dt}{\int_{T_\tau}^{T_\tau+T_d} p(t)dt}. \tag{5}$$

Then, we can express $E[\theta_2]$ from 3 by substituting terms with 4 and 5.

Case 3, $T_\tau + T_d < t$**:**

Because the setup time in Case 3 is $X_3 = T_u$, we have

$$E[\theta^3] = \frac{\rho}{(1-\rho)}\frac{E[S^2]}{2E[S]} + \frac{\lambda^{-1}}{\lambda^{-1}+T_u} + \frac{T_u}{\lambda^{-1}+T_u}\frac{T_u}{2} + E[S].$$