Interplay of Energy and Performance for Disk Arrays Running Transaction Processing Workloads

Sudhanva Gurumurthi[†] Jianyong Zhang[†] Anand Sivasubramaniam[†] Mahmut Kandemir[†]
Hubertus Franke[‡] N. Vijaykrishnan[†] Mary Jane Irwin[†]

Technical Report CSE-02-014
October, 2002

[†] Department of Computer Science and Engineering
The Pennsylvania State University, University Park, PA 16802
{gurumurt,jzhang,anand,kandemir,vijay,mji}@cse.psu.edu

[‡] IBM Research Division
Thomas J. Watson Research Center
Yorktown Heights, NY 10598
frankeh@us.ibm.com

Abstract

The growth of business enterprises and the emergence of the Internet as a medium for data processing has led to a proliferation of applications that are server-centric. The power dissipation of such servers has a major consequence not only on the costs and environmental concerns of power generation and delivery, but also on their reliability and on the design of cooling and packaging mechanisms for these systems. This paper examines the energy and performance ramifications in the design of disk arrays which consume a major portion of the power in transaction processing environments.

Using traces of TPC-C and TPC-H running on commercial servers, we conduct in-depth simulations of energy and performance behavior of disk arrays with different RAID configurations. Our results demonstrate that conventional disk power optimizations that have been previously proposed and evaluated for single disk systems (laptops/workstations) are not very effective in server environments, even if we can design disks than have extremely fast spinup/spindown latencies and predict the idle periods accurately. On the other hand, tuning RAID parameters (RAID type, number of disks, stripe size etc.) has more impact on the power and performance behavior of these systems, sometimes having opposite effects on these two criteria.

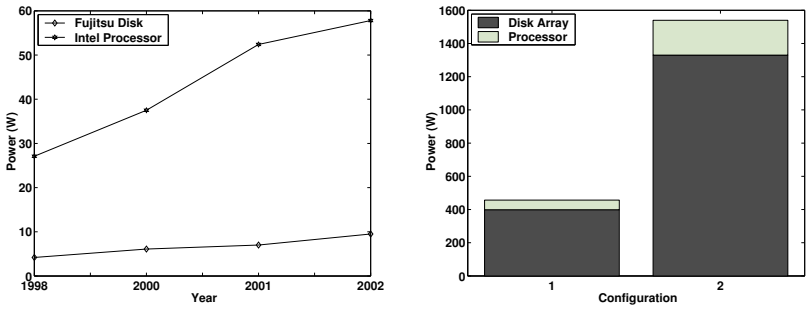
Keywords: Simulation, Workload Characterization, RAID, Transaction Processing, Power Optimization

1 Introduction

The growth of business enterprises and the emergence of the Internet as a medium for data processing has resulted in a proliferation of applications that are server-centric. Both small-scale businesses and large corporations employ servers for managing inventory and accounts, performing decision-support operations, and hosting web-sites. Traditionally, the main targets for optimization in these environments have been performance (either response time to a single user or improving overall throughput), reliability, and availability. However, power consumption is increasingly becoming a major concern in these environments [7, 4, 10]. Optimizing for power has been understood to be important for extending battery life in embedded/mobile systems. It is only recently that the importance

of power optimization in server environments has gained interest because of the cost of power delivery, cost of cooling the system components, and the impact of high operating temperatures on the stability and reliability of the components.

It has been observed that data-centers can consume several Mega-watts of power [21, 7, 28]. A related paper [6] observes that power densities of data centers could grow to over 100 Watts per square foot, and future centers could add 5 GW of demand (which is around 10% of the current generating capacity for California). The capacity of new data centers for 2005 could require approximately 40 TWh (around $4B) per year. This power is not only essential to keep the processors up and running, but is also drained by the I/O peripherals - particularly the disk subsystem - which in these server environments are employed in abundance to provide high storage capacities and bandwidth. Specifically, most of these servers use some type of RAID disk configuration to provide I/O parallelism. While this parallelism benefits performance, the additional disks drain power at the same time, and it is not clear whether the performance benefits justify the power that they consume. It is important to study these trade-offs when designing I/O subsystems for the data centers, and to our knowledge this is the first paper to do so.



(a) Single CPU and Disk (b) Current Server Configurations

Figure 1: Relative Proportion of Processor and Disk Power in High-End Systems

Figure 1 (a) shows the growth of power consumption of a representative (Intel family) processor and a representative (Fujitsu family) disk over the past four years. These power values are taken from standard specification sheets [11, 20]. The average processor power and the disk idle power (the disk is spinning but not serving any request) are used to draw this graph. For the disks, the form-factor was held constant at 3.5-inch and the capacity at 18.4 GB except for the 1998 case, where the the largest capacity listed was 10.24 GB. It may appear to the reader that the absolute power drawn by the processor is more than the disk, and the difference is also growing with time. However, it should be noted that these are single disk values, with the disk not serving any request. In a real server environment, several such disks (put together as RAID) are employed for I/O parallelism. Figure 1 (b) shows the power dissipation split between the processor and the disk subsystem on two real server environments. The parameters for these environments are taken from the executive summaries in the Transaction Processing Council (TPC) website [30, 31] - the first (denoted 1 in Figure) being an Intel Xeon Processor clocked at 2.2 GHz with 47 disks and the second (denoted 2 in Figure) a 4-way Intel Xeon SMP clocked at 1.6 GHz with 147 disks. Transaction processing workloads are typically constrained by the I/O bandwidth, and vendors tend to string together a large number of disks even for a small improvement in response time. As can be observed, the disk power overwhelms that consumed by the processor. It should also be noted that this assumes disks are idle (but spinning), and with higher I/O accesses as is the case in transaction processing workloads, the disk power is expected to be even higher.

Disk power management has been considered in previous research [9, 22, 16, 15, 24, 14] as an important issue in single disk environments (primarily laptops), where the goal has been to conserve battery power by spinning down the disks during periods of idleness. There are several differences between those studies and the issues under consideration in this paper. First, we are examining systems that employ a large number of disks (RAID configurations) for server systems, rather than single disk systems. Second, the workloads in server environ-

ments are significantly different from those on traditional workstation/laptop environments. We have to deal with several users/transactions at the same time, with workloads being much more I/O intensive. The server workloads typically have a continuous request-stream that needs to be serviced, instead of the relatively intermittent activity that is characteristic of the more interactive desktop and laptop environments. This makes it difficult to obtain sufficiently long idle-periods to profitably employ conventional disk power-management schemes which use mode control. Further, response time is a much more critical issue in web-servers and database-servers in the corporate world (compared to desktop workloads), and we need to be careful when there is the issue of degrading performance to gain power savings. In addition, the I/O subsystems in server environments offer several more parameters for tuning (RAID configuration, number of disks, striping unit, etc.) as opposed to single disk systems. Typically, these have been tuned for performance, and it is not clear whether those values are power efficient as well. Finally, server disks are also physically different from their laptop and desktop counterparts. They are typically heavier and have much larger spinup and spindown times and are more prone to breakdown when subjected to these mechanical stresses [3] (reliability and availability are of paramount importance in these environments). All these differences in system environments and workload behavior warrant a rethinking of the way power-management needs to be done for these systems.

Transaction processing workloads are amongst the most common and I/O intensive, of the commercial applications. We specifically focus on the TPC-C and TPC-H workloads in this paper [34]. These workloads are extensively used in the commercial world to benchmark hardware and software systems. TPC-C is an On-Line Transaction Processing (OLTP) benchmark, that uses queries to update and lookup data warehouses (such as those in Walmart, etc.). TPC-H, in contrast, involves longer-lived queries that analyze the data for decision-making (On-Line Analysis Processing - OLAP).

We carry out a detailed and systematic study of the performance and energy consumption across the RAID design-space (particularly RAID-4, RAID-5 and RAID-10), and examine the interplay between power and performance for these workloads. This is conducted with a trace-driven simulation using the DiskSim [13] simulation infrastructure. DiskSim provides a detailed disk-timing model that has shown to be accurate [12], and we have augmented this infrastructure for power measurements.

We first show that traditional disk power management schemes proposed in desktop/laptop environments are not very successful in these server workloads, even if we can design the disks to spinup and spindown very fast and predict the idle periods accurately. Consequently, the options for disk power management require the investigation and tuning of the different parameters, which this paper undertakes. We demonstrate that tuning the RAID configuration, number of disks, and stripe size has more impact from the power optimization angle. The values of system parameters for best performance are not necessarily those that consume the least power, and vice-versa.

It should be noted that it is possible to have large idle periods during intervals of *light load* (at nights, weekends, etc.). During those times, any power saving technique (even simple heuristics that wait a few minutes before powering down the disks/system) would suffice. Our focus in this work is more at periods of heavier load which is what TPC-C and TPC-H are intended to capture (and which is when more power would be dissipated and cooling becomes more important).

The rest of the paper is organized as follows. Section 2 presents the related work. Section 3 provides a brief overview of the RAID configurations used in this paper. Section 4 describes the workloads and metrics used in the evaluation, along with details of the simulated hardware. Section 5 presents the results of the study and Section 6 summarizes the contributions of this work.

2 Related Work

Disk power management has been extensively studied in the context of single disk systems, particularly for the mobile/laptop environment. Many current disks offer different power modes of operation (active - when the disk is servicing a request, idle - when it is spinning and not serving a request, and one or more low power modes

which consume less energy than idle where the disk may not spin). Managing the energy consumption of the disk consists of two steps, namely, detecting suitable idle periods and then spinning down the disk to a low power mode whenever it is predicted that the action would save energy. Detection of idle periods usually involves tracking some kind of history to make predictions on how long the next idle period would last. If this period is long enough (to outweigh spindown/spinup costs), the disk is explicitly spun down to the low power mode. When an I/O request comes to a disk in the spundown state, the disk first needs to be spun up to service this request (incurring additional exit latencies and power costs in the process). One could pro-actively spin up the disk ahead of the next request if predictions can be made accurately, but many prior studies have not done this. Many idle time predictors use a time-threshold to find out the duration of the next idle period. A fixed threshold is used in [22], wherein if the idle period lasts over 2 seconds, the disk is spun down, and spun back up only when the next request arrives. The threshold could itself be varied adaptively over the execution of the program [9, 16]. A detailed study of idle-time predictors and their effectiveness in disk power management has been conducted in [14]. Lu et al. [23] provide an experimental comparison of several disk power management schemes proposed in literature on a single disk platform.

If we move to high-end server class systems, previous work on power management has mainly focussed on clusters, employing techniques such as shutting off entire server nodes [7], dynamic voltage scaling [4], or a combination of both [10]. There has also been work to reduce the power consumption by balancing the load between different nodes of a cluster [28]. Shutting off nodes is possible if the computational load can be re-distributed or there exist mirrors for the data. The application of voltage scaling can reduce the CPU energy consumption, and has indeed been shown to reduce energy up to 36% for web-server workloads [4]. Investigation of power optimization for SMP servers has looked into optimizing cache and bus energy [25].

Our focus in this paper is on power optimizations for the I/O (disk) subsystem, particularly in the context of transaction processing workloads. In web server workloads (where there are fewer writes), duplication of files on different nodes offers the ability to shut down entire nodes of the web serving cluster (both CPU and its disks) completely. Further, a web page (file) can be serviced by a single node (disks at that node), and the parallelism on a cluster is more to handle several requests at the same time rather than parallelizing each request. On the other hand, database engines use disks for parallelism by striping the data across them, and mirroring is done (if at all) on a much smaller scale. Each database query typically involves several disks for its data processing. Even in data centers which use a cluster for transaction processing, there is usually a Storage Area Network that is accessible by all the nodes, on which the disks reside. In such cases, our work can be applied to manage the disks, while the previous cluster management techniques can be applied to the CPUs. Finally, transaction processing workloads can have very different characteristics from web server workloads [26] - locality between requests is expected to be higher in web servers compared to transaction processing workloads, potentially returning pages from the cache.

There have been some studies that have attempted power management in the context of disk arrays. [35] looked into minimizing the disk energy consumption of a laptop disk by replacing it with an array of smaller form-factor disks. The argument was that the power consumed by a disk is a direct function of its size. However, this study does not look at striping the data across these disks for I/O parallelism, and is thus not applicable to server environments. In [8], the authors have proposed replacing a slow and power-hungry tape-backup system with an array of disks that are kept in the spundown state as long as possible. This work targets archival and backup systems, where idleness of the disks is much easier to exploit, and writes overwhelm the reads.

To our knowledge, power management for high performance I/O subsystems with database workloads is largely unexplored. Our work can be applied either to SMP systems (which constitute the bulk of transaction processing servers today) that are directly attached to RAIDs, or to cluster environments which interface to these parallel disks via a Storage Area Network.

3 RAID Overview

Redundant Array of Independent/Inexpensive Disks (RAID) [27] employs a bunch of disks to serve a request in parallel, while providing the view of a single device to the request. If there are n disks, with each disk having a capacity of B blocks, then the RAID address space can be visualized as a linear address-space from 0 to $nB - 1$. The unit of data distribution across the disks is called the *striping-unit* or just *stripe*. A stripe consists of a set of consecutive blocks of user data. Since there are multiple disks present, the reliability of the array can go down. Consequently, RAID configurations use either parity or mirroring for error detection and recovery. There are several RAID configurations based on how the data is striped and how the redundancy is maintained. We consider RAID levels 4, 5, and 10 in this paper (the latter two are among the more popular ones in use today).

In a RAID-4 array of n disks, $n-1$ disks store data, and the remaining disk stores parity. Logical stripe i thus falls on disk i mod $(n-1)$, i.e. the data stripes are assigned to the disks storing data in a cyclic fashion. The parity disk stores the parity information for stripes 1 through $n-1$, n through $2(n-1)$, and so on. When a read for a logical block is presented, the array controller will figure out the disk(s) and their blocks that are involved, and issue the requests to them. At the same time, the parity blocks for those will also need to be obtained to confirm the validity of the data. In a write operation, apart from involving the data disks, the parity may have to be both read and written (because it may need to be re-calculated). Usually, writes thus become a problem (especially small ones).

RAID-5 works the same way as RAID-4, with the difference being that there is no one disk dedicated for parity. Instead, each disk takes turns serving to hold the parity for a given set of stripes (this is usually called rotating parity - e.g. disk n serves as parity for data stripes 0 through $n-1$, disk 1 serves as parity for data stripes n through $2(n-1)$, disk 2 serves as parity for data stripes $2n-1$ through $3(n-1)$, and so on). Consequently, this can avoid the bottleneck of a single parity disk (especially for high small-write traffic), evening out the load across the array.

RAID-10, which has gained popularity in recent times, employs a combination of data *mirroring* (duplication of data on two different disks) and striping. Its specifications are rather loose, and we explain the scheme used in our experiments. The disk array is split into two mirrors (each with an equal number of disks). Within each mirror, we simply stripe the data across the disks in a cyclic fashion (without any parity). A write will need to update both mirrors. In the read implementation, we send the request to the mirror which has the shortest request queue at that instant [13]. Ties are broken by picking the mirror with the shortest seek time. RAID-10 provides greater reliability since it can tolerate multiple disk failures.

4 Experimental Setup

Before we get to the results of this study, we go over the transaction processing workloads that we use and give details on the platform that is used for the simulation.

4.1 Workloads

As explained earlier, this paper focuses mainly on transaction processing workloads that are representative of the loads in several high end databases/data-warehouses. These workloads use database engines to store, process and analyze large volumes of data that are critical in several commercial environments. Many of these are also back-end servers for a web-based interface that is used to cater to the needs of several hundreds/thousands of users, who need low response times while sustaining high system throughput.

We specifically use two important transaction processing workloads identified by the Transaction Processing Council (TPC) [34]. While ideally one would like to run simulations with the workloads and database engines in their entirety in direct-execution mode, this would take an inordinate amount of time to collect data points with the

numerous parameters that we vary in this study. Consequently, we have used device level traces from executions on actual server platforms to drive our simulations.

- **TPC-C Benchmark:** TPC-C is an On-Line Transaction Processing (OLTP) benchmark. It simulates a set of users who perform transactions such as placing orders, checking the status of an order etc, that can be useful in managing the data processing needs of supply-chains in an enterprise or even across enterprises. Transactions in this benchmark are typically short, and involve both read and update operations. For more details on this benchmark, the reader is directed to [29]. The tracing was performed for a 20-warehouse configuration with 8 clients and consists of *6.15 million I/O references spread over 12.06 hours*. The traced system was a 2-way Dell PowerEdge SMP machine with Pentium-III 1.13 GHz processors with 4 10K rpm disks running IBM's EEE DB-2 [17] on the Linux operating system.
- **TPC-H Benchmark:** This is an On-Line Analytical Processing (OLAP) benchmark and is used to capture decision-support transactions on a database [32]. There are 22 queries in this workload, and these queries typically read the relational tables to perform analysis for decision-support. The trace that is used in this study was collected on an IBM Netfinity SMP server with 8 700 Mhz Pentium III processors and 15 IBM Ultrastar 10K rpm disks, also running EEE DB-2 on Linux, and consists of *18 million I/O references spread over a time-period of 45.9 hours*.

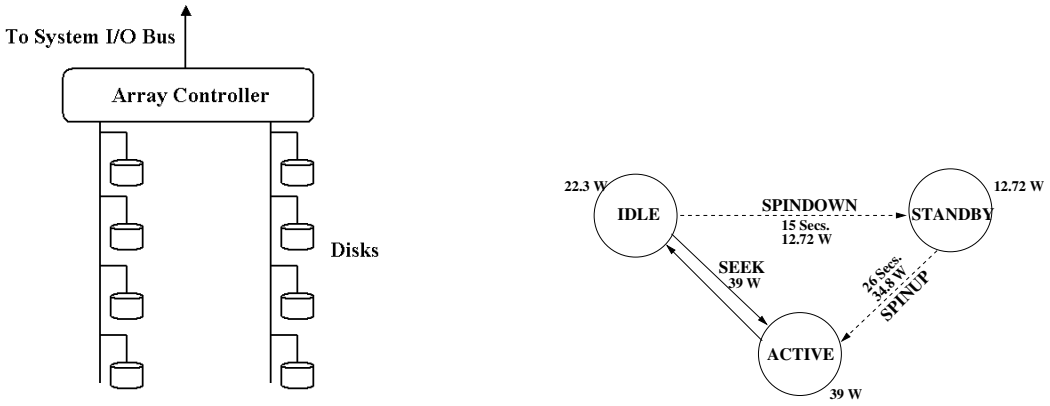
The above traces are those used in our default configuration, and we have also studied the impact of the dataset size in our experiments. We would like to mention that the overall trends hold across datasets, and the detailed results are omitted here in the interest of space. The traces have been collected at the device level and give the timestamp, type of request (read/write), logical block number and number of blocks. We map the logical block numbers to the disk parameters based on the RAID configuration.

4.2 Simulation Environment

In this study, we use the DiskSim simulator [13], augmented with a disk power model, to study the performance and power implications of RAID configurations on the transaction processing workloads. DiskSim provides a large number of timing and configuration parameters for specifying disks and the controllers/buses for the I/O interface. The default parameters that we use in this study are given in Table 1. The RPM and disk cache size have been chosen to reflect what is popular today for servers. The power values are taken from the data sheets of the IBM Ultrastar 36ZX [19] disk, which is used in several servers. The reader should note that spinup/spindown operations are quite expensive and these values have also been obtained from the data sheets for the IBM Ultrastar 36ZX.

Parameter	Value
Number of Disks: 32 **Stripe Size: 16 KB**	
Capacity	33.6 GB
Rotation Speed	12000 rpm
Disk Cache Size	4 MB
Idle Power	22.3 W
Active (Read/Write) Power	39 W
Seek Power	39 W
Standby Power	12.72 W
Spinup Power	34.8 W
Spinup Time	26 secs.
Spindown Time	15 secs.
Disk-Arm Scheduling Algorithm	Elevator
Bus Type	Ultra-3 SCSI

Table 1: Default Disk Configuration Parameters. Many of these have been varied in our experiments.



(a) Disk Array Configuration (b) Disk Power Modes

Figure 2: RAID Configuration and Power Modes

Our simulated I/O subsystem architecture looks as shown in Figure 2 (a) for an 8-disk configuration. The disks in the RAID array are attached to an array-controller using 2 Ultra-3 SCSI buses. In our experiments, half the disks are on each bus (typically, each bus can sustain the bandwidth needs of up to 16 disks). The array controller stripes the data (as per the RAID configuration) across all these disks. It also has an on-board cache, which can potentially avoid some of the disk accesses. The array controller is in turn interfaced to the main system via the I/O bus, and the power issues for those parts are not studied here.

Figure 2 (b) shows the power mode transitions for each disk. In the experiments where no power management is undertaken, *Active* and *Idle* are the only modes of operation, with the former being used during actual media accesses, and the latter when not performing any operation (disk is only spinning). In the experiments with explicit mode control, we use an additional low power state - *Standby* - where the power consumption is much lower (see Table 1), and a cost is expended to bring the disk to Active state before servicing a request (transitions to and from this state are shown as dashed lines).

The parameters that we vary include the RAID configuration (4, 5 and 10), the number of disks, and the stripe size.

4.3 Metrics

In our evaluation, we use four metrics, namely, *total energy consumption over all the requests (E_{tot})*, *average energy consumption per I/O request (E)*, *response-time per I/O request (T)*, and *energy-response-time product (E $\times$ T)*. These can be defined as follows:

- The total energy consumption (E_{tot}) is the energy consumed by all the disks in the array from the beginning to the end of the trace. We monitor all the disk activity (states) and their duration from the start to the end of the simulation, and use this to calculate the overall energy consumption by the disks (integral of the power in each state over the duration in that state).

- The energy consumption per I/O request (E) is E_{tot} divided by the number of I/O requests. A previous study on energy management of server clusters also uses a similar metric (Joules per Operation) for capturing the impact of energy/power optimization for a given throughput [6].

- The response-time (T) is the average time between the request submission and the request completion. This directly has a bearing on the delivered system throughput.

- The product of the previous two ($E \times T$) measures the amount of energy or performance we can tradeoff for the other to have an overall beneficial effect. For instance, if we increase the number of disks in the array, and, get much more improvement in response time than the additional energy consumption, then we can consider this optimization to be a net winner and the product would quantitatively capture this effect. We understand that the energy-delay product requires a complete system characterization to really quantify how the savings of response time in one hardware component can affect the energy of other hardware components (and vice-versa). In this paper, we use this term more to qualify the *relative importance* of energy and response time.

5 Results

In the following subsections, we explore the possible benefits of traditional disk power management using spin-downs for these server workloads and show that there is not much scope for energy savings with this approach. Subsequently, we investigate different hardware parameters that can be tuned for power savings.

5.1 Effectiveness of Conventional Disk Power Management

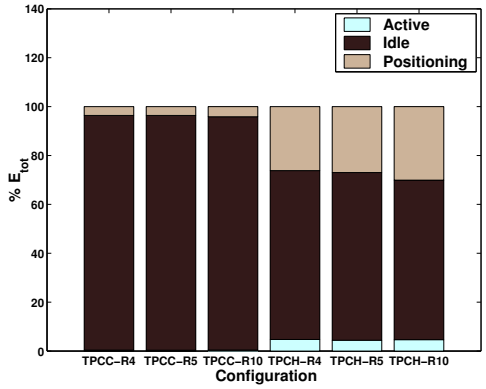


Figure 3: Breakdown of Total Energy Consumption in Different Disk Modes (R4, R5 and R10 refer to RAID-4, RAID-5 and RAID-10 respectively).

In order to optimize disk energy, we need to first examine the energy profile of an actual execution. Figure 3 shows the energy consumption breakdown for the three RAID configurations and two workloads, in terms of the energy consumed when in active mode, idle mode, and during head movements (positioning). It can be seen that contrary to expectations, the least amount of energy is actually spent in the active mode. Most of the energy is expended when the disk is idle, and to a lesser extent for positioning the head. This suggests that one should optimize the idle mode for energy and one possibility is to draw from the ideas of power mode transitions which tries to put the disk in standby mode when it is not performing any operation. This is explored in the next few experiments.

Optimizing disk power for prolonging battery life has been explored in earlier research with smaller form factor disks, by exploiting idle times (by being able to predict them) to spindown the disk. In the following results, we examine how applicable those techniques can be for the server environments under investigation. We first examine the predictability of idle times between disk activities. Next, we examine the duration of idle times to see how much scope is there for employing these techniques. Finally, we use an oracle predictor (that is accurate when predicting idle times both in terms of detecting when an idle period starts and what its duration would be) to see what is the maximum we can hope to gain by these techniques.

5.1.1 The Predictability of Idle-Periods

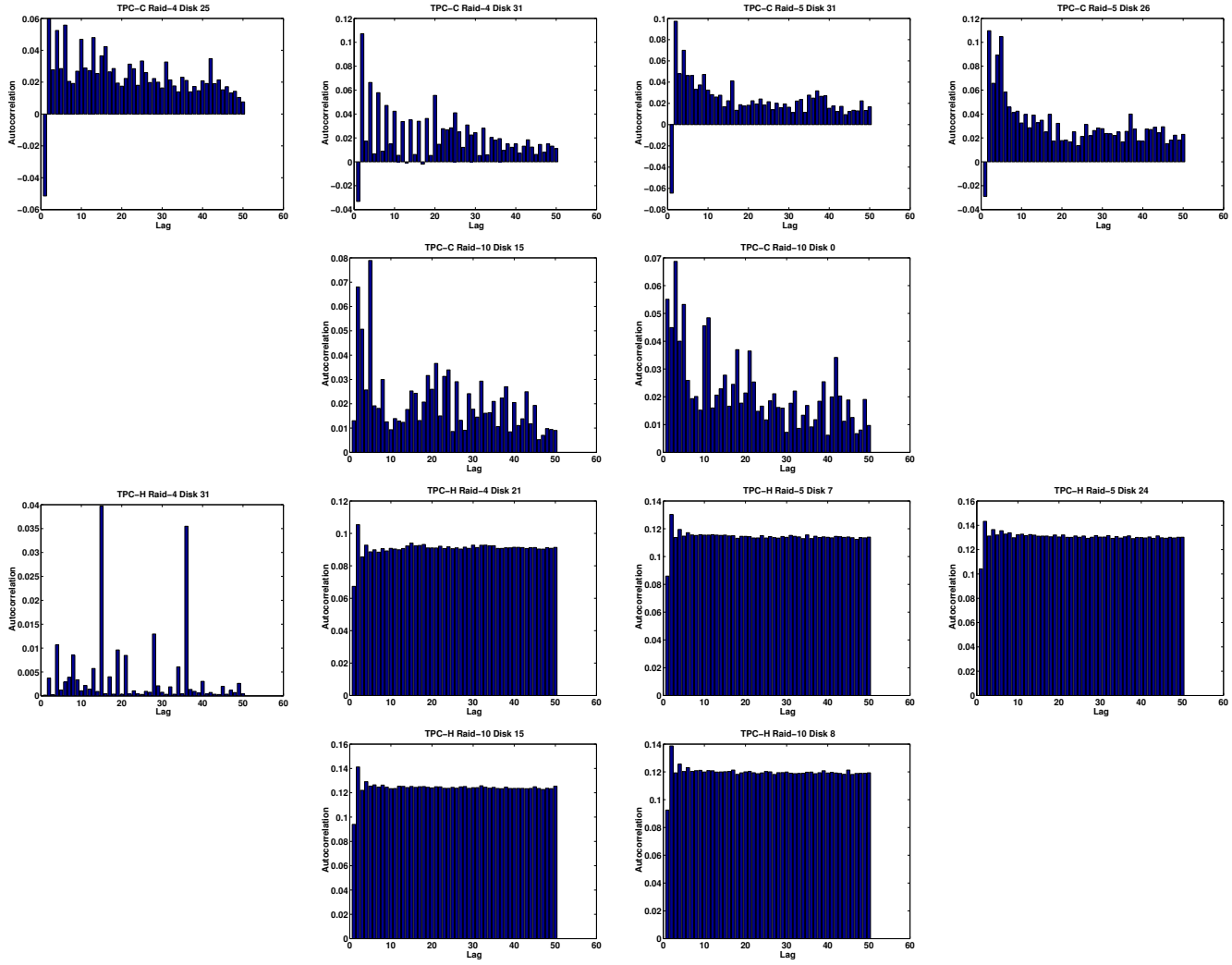


Figure 4: Autocorrelation of Idle Periods for 50 Lags. For each RAID configuration and for each workload, the first graph pertains to the disk that has the least number of distinct idle-periods and the second to the one with the most. Note that the values either degrade slowly (e.g. TPC-H RAID 5 Disk 7) or degrades sharply but the absolute values are quite low (e.g. TPC-C RAID 10 Disk 0)

Prediction of disk requests based on prior history has been a topic of previous research [33]. One commonly used technique is "autocorrelation analysis", wherein data with good correlations are conducive for fitting with ARIMA models [5] as a time-series. Essentially, an autocorrelation at lag i is computed between the observation pairs (idle time periods) $y(t)$ and $y(t+i)$ [14]. The resulting values are plotted as a graph for different lags. A sequence that lends itself to easier prediction models is characterized by a graph which has a very high value for small lags and then steeply falls to low values for larger lags (i.e. recent history has more say on the prediction making it easier to construct a model based on these). Note that observations can be negatively correlated as well. Further, there could be some repetitive sequences which can cause spikes in the graph, resulting in deviations from monotonic behavior. The reader is referred to [5] for further explanation on such time-series models.

We have conducted such an autocorrelation analysis of the idle periods of disks for 50 lags (lag 0 is not plotted) and the resulting graphs are shown in Figure 4 for the two workloads and all three RAID configurations. For each experiment, we show the graphs for 2 disks: the disk with the minimum number of distinct idle periods, and the

disk with the maximum number of distinct idle periods.

Overall, we observe that we do not have good correlation of idle periods. Either the values degrade slowly or degrade sharply but the absolute values are quite low. We show in Table 2, the mean (μ) and standard deviation (σ) across all the disks for the first five lags. As we can observe, except in a few cases, the mean does not cross 0.12 and the standard deviation is not very high either. All these results suggest that it is difficult to get good predictions of idle times based on previous (recent) history. These results are in contrast with those for normal workstation workloads, which have been shown to have higher correlations [14].

Benchmark	RAID Level	Lag 1		Lag 2		Lag 3		Lag 4		Lag 5	
		μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
TPC-C	**RAID-4**	-0.064	0.013	0.070	0.020	0.037	0.014	0.045	0.016	0.031	0.009
	RAID-5	-0.044	0.016	0.087	0.019	0.058	0.015	0.057	0.013	0.050	0.016
	RAID-10	0.014	0.020	0.076	0.019	0.057	0.015	0.043	0.014	0.049	0.015
TPC-H	**RAID-4**	0.066	0.012	0.101	0.017	0.083	0.015	0.090	0.014	0.085	0.015
	RAID-5	0.085	0.011	0.130	0.009	0.115	0.011	0.120	0.010	0.116	0.010
	RAID-10	0.092	0.005	0.139	0.005	0.118	0.005	0.125	0.005	0.121	0.005

Table 2: Autocorrelation Statistics of All Disks Over 5 Lags. For each lag, μ and σ denote the mean and standard-deviation of the autocorrelation at the given lag respectively.

Note that, though it is difficult to obtain good predictability of the idle periods using time-series analysis, which relies on the recent past for making predictions, it is possible that good prediction accuracy could be obtained by other means. For example, if the idle periods could be fitted to a probability distribution, it may be possible to predict the duration of an idle period with higher probability. However, as we shall show in section 5.1.3, even with perfect prediction, conventional disk power management does not provide much savings in the energy consumption.

5.1.2 The Duration of Idle-Periods

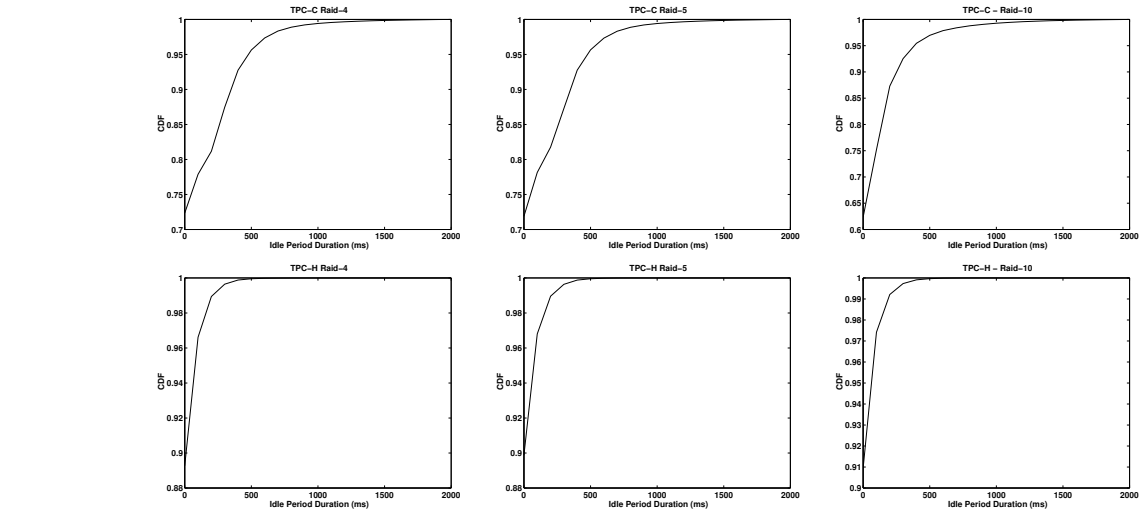
One could argue that even if we are not able to accurately predict the duration of the next idle period, it would suffice if we can estimate that it is larger than a certain value as far as power management is concerned. In order to ascertain the number of idle-periods that could potentially be exploited for power management, we plotted the idle periods of the disks as a Cumulative Density Function (CDF). Figures 5 (a) (cumulative over all disks) and 5 (b) (for the disks with minimum and maximum distinct idle times) present this data.

We observe that, whether it be the overall results or that for an individual disk, idle times are extremely short. In fact, the configurations for TPC-H do not show any visible idle times greater than even 1 second. TPC-C, on the other hand, shows some idle times larger than 2 seconds, but this fraction is still quite small (less than 1% in most cases).

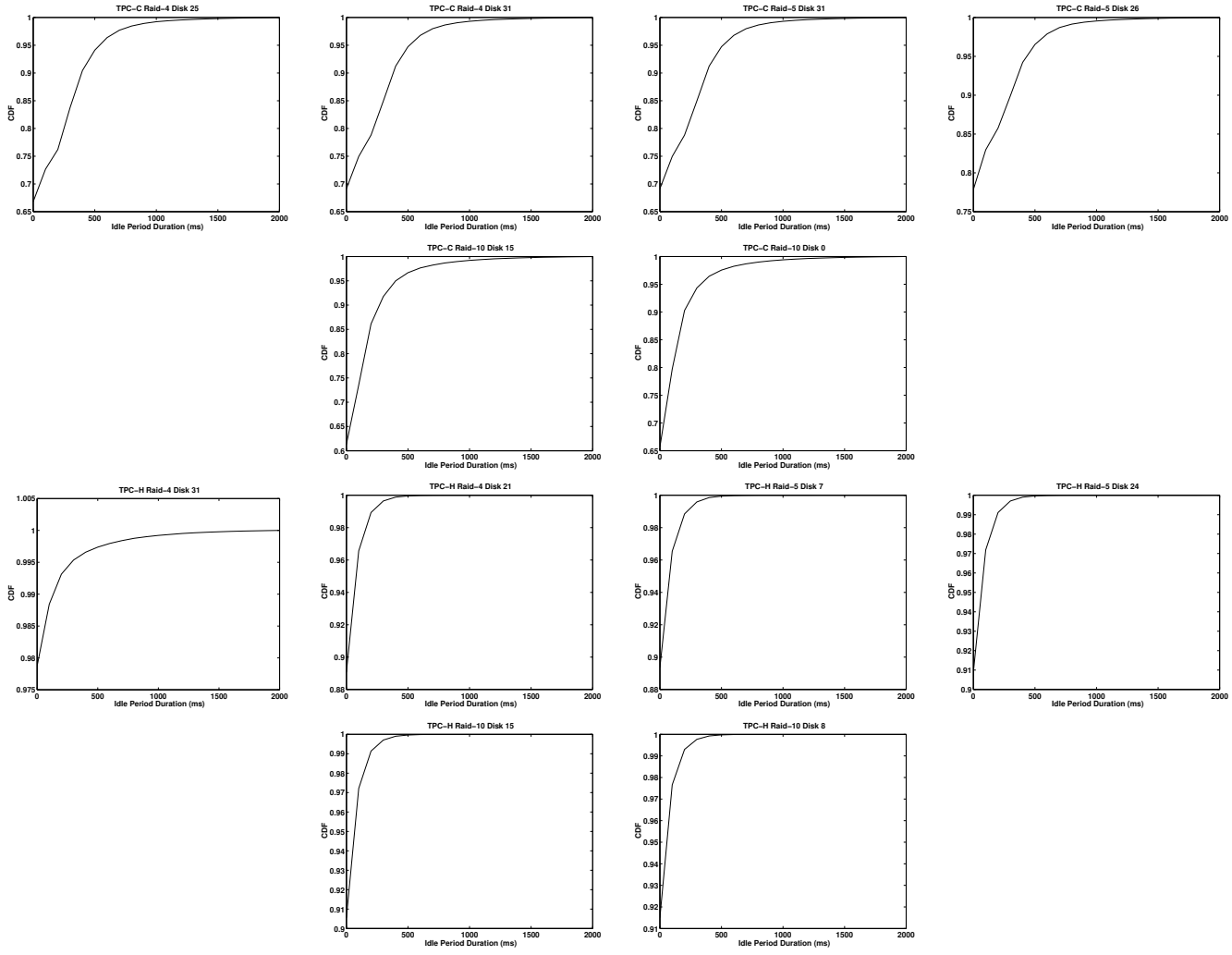
These results indicate that there is not much to be gained with traditional power management techniques, regardless of the predictability of the idle times, if spinup/spindown times are in the ranges indicated in Table 1.

5.1.3 Limits of Traditional Disk Power Management

Figure 6 (a) shows the maximum energy savings that we can hope to get for these workloads and RAID configurations without any degradation in response times for the spindown/spinup values shown in Table 1 for the server disk under consideration. We are assuming an oracle-predictor which has perfect knowledge of the next idle period for this calculation. For TPC-C, performing traditional disk power management actually hurts the overall energy consumption, as the durations of the long idle periods are not sufficient enough to overcome the energy cost of spinning up the disk. Even in the best case (RAID-4 for TPC-H), the percentage energy savings is quite negligible (less than 0.5%).

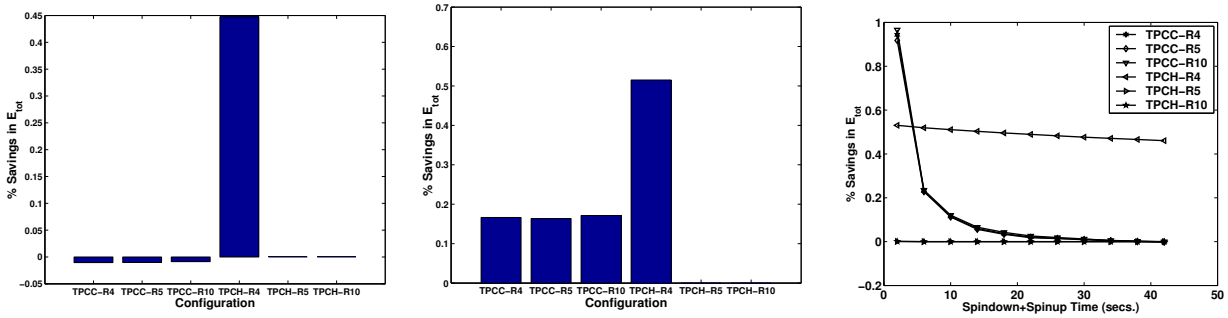


(a) For all Disks



(b) For Disks with the Minimum and Maximum Number of Idle Periods

Figure 5: Cumulative Density Function (CDF) of Idle Periods. This captures what fraction of the total idle times are less than a given value on the x-axis



(a) (b) (c)

Figure 6: Percentage Savings in the Total Energy Consumption with Disk Spindowns using a perfect idle time prediction oracle. The savings are given for (a) current server class disk (spindown+spinup = 41 secs), (b) an aggressive spinup + spindown value (9 secs, for a state-of-the-art IBM Travelstar disk used in Laptops), and (c) for different spinup + spindown values.

We have also investigated how this situation would change if we had a laptop-type disk, whose spindown/spinup times are typically much lesser than those of server disks. Even when we assume values of *4.5 seconds* for spindown and spinup (which is in the range of state-of-the-art laptop disks [18]), the energy savings for these workloads are still quite small as is shown in Figure 6 (b). Figure 6 (c) plots the energy savings that can possibly be obtained for different values of spinup+spindown latencies. As can be seen, even if these latencies become smaller than 2 seconds, we get less than 1% improvement in the energy consumption. It is not clear if we can get these numbers down to those values for server class disks without any degradation in performance. Even if we do, these may need very powerful spindle motors, which can in-turn increase the energy consumption.

Despite the high idle power that was shown in Figure 3, we find that idle periods themselves are not very long. The contribution to the idle power is more due to the number of idle periods than the duration of each. This also suggests that it would be fruitful if we can develop techniques to coalesce the idle periods somehow (batching requests, etc.) to better exploit power mode control techniques.

5.2 Tuning the RAID Array for Power and Performance

Since power mode control does not appear very productive when we do not have the flexibility of extending response times, it is then interesting to examine what factors within the I/O architecture can influence its design for power-performance trade-offs. In the following discussion, we look at three important parameters - the RAID configuration (RAID 4, 5 and 10), the number of disks across which the data is striped, and the stripe size. Traditional studies have looked at these parameters only from the performance angle.

5.2.1 Impact of Varying the Number of Disks

Benchmark	RAID Level	Number of Disks									
		30		32		34		36		38	
		Avg. SD	# Seeks	Avg. SD	# Seeks	Avg. SD	# Seeks	Avg. SD	# Seeks	Avg. SD	# Seeks
TPC-C	RAID-4	0.316	359108	0.319	3586994	0.326	3603281	0.322	3595662	0.323	3608751
	RAID-5	0.309	3499088	0.308	3496772	0.317	3520690	0.311	3517927	0.315	3538549
	RAID-10	0.999	4555182	1.011	4544314	1.042	4545012	0.986	4545181	1.039	4547390
TPC-H	RAID-4	266.686	82276911	273.184	81302660	307.488	76718922	294.403	69802024	295.256	70659910
	RAID-5	268.230	82257075	303.149	76466114	282.606	81590519	286.810	74076442	306.818	69966099
	RAID-10	540.029	73547703	527.901	74058987	538.656	74996155	496.434	74209205	514.636	73830877

Table 3: Average Seek Distance (SD) and Number of Seeks for each workload for a given number of disks.

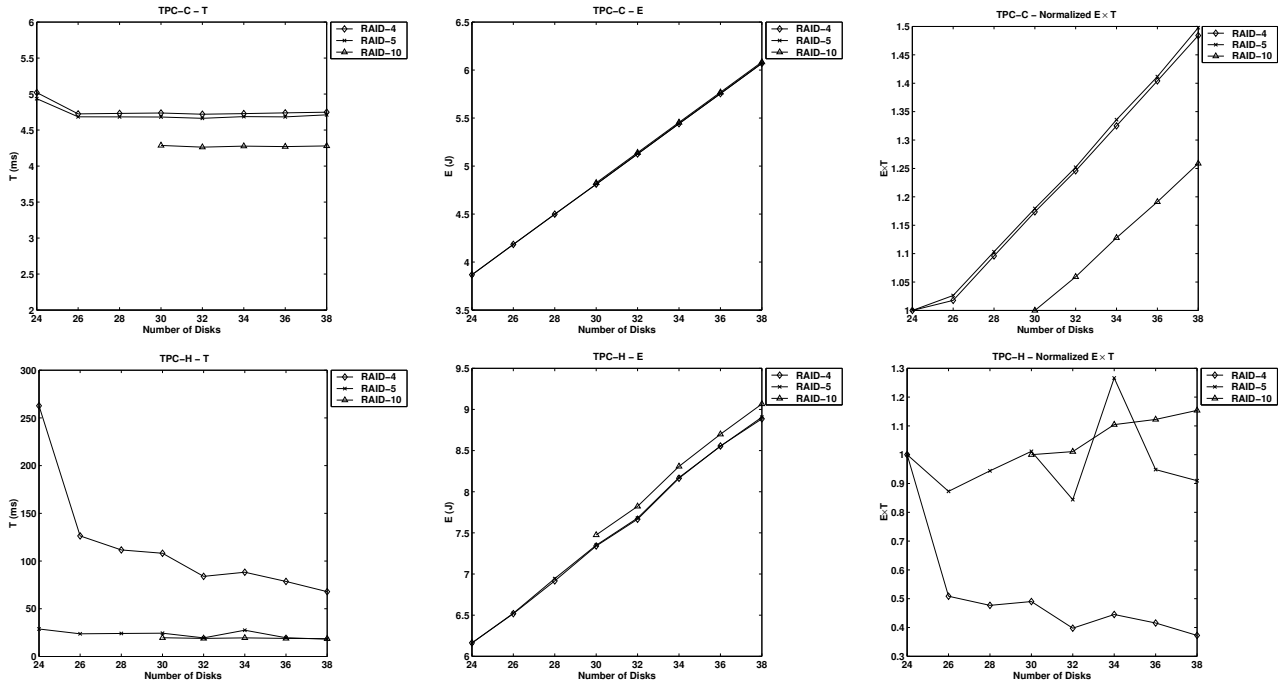


Figure 7: Impact of the Number of Disks in the Array

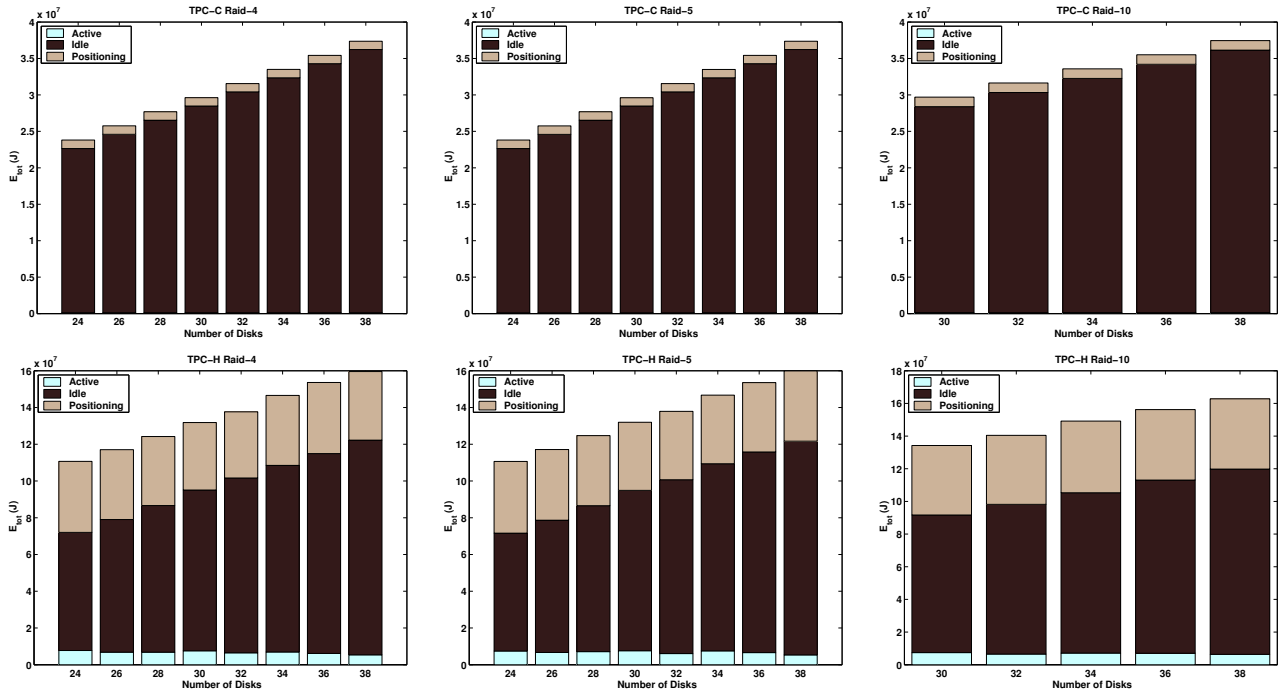


Figure 8: Impact of the Number of Disks in the Array - Breakdown of Total Energy Consumption (E_{tot})

Benchmark	RAID Level	Number of Disks				
		30	32	34	36	38
TPC-C	RAID-4	1.010 (D) 1.049 (P)	1.010 (D) 1.047 (P)	1.010 (D) 1.047 (P)	1.010 (D) 1.046 (P)	1.010 (D) 1.046 (P)
	RAID-5	1.012	1.011	1.012	1.012	1.012
	RAID-10	1.0000 (M1) 1.0000 (M2)	1.0000 (M1) 1.0000 (M2)	1.0000 (M1) 1.0000 (M2)	1.0000 (M1) 1.0000 (M2)	1.0000 (M1) 1.0000 (M2)
TPC-H	RAID-4	1.400 (D) 75.142 (P)	1.263 (D) 56.627 (P)	1.027 (D) 63.248 (P)	1.011 (D) 56.157 (P)	1.011 (D) 46.961 (P)
	RAID-5	1.258	1.024	1.413	1.030	1.014
	RAID-10	1.010 (M1) 1.010 (M2)	1.004 (M1) 1.004 (M2)	1.014 (M1) 1.014 (M2)	1.005 (M1) 1.005 (M2)	1.002 (M1) 1.002 (M2)

Table 4: Instantaneous Queue Lengths of All Configuratons. For RAID-4, the average queue length of all the data disks (D) and that of the parity disk (P) are given. For RAID-5, the given value is the average queue length of all the disks in the array. For RAID-10, the average queue length of each mirror, M1 and M2, is presented.

Figure 7 shows the T, E, and $E \times T$ (as defined in section 4.3) as a function of the number of disks that are used in the three RAID configurations for the two workloads. Please note that the third graph is normalized with respect to the leftmost point for each line. The energy-response time product has been normalized this way since we are more interested in the trend of a single line than a comparison across the lines. Figure 8 shows the total energy consumption (across all disks) broken down into the active, idle and positioning components. Due the size of the I/O space addressed by the workloads, we chose configurations for RAID 10 starting from 30 disks. We make the following observations from these graphs.

When we examine the performance results, we notice little difference between the three RAID configurations for the TPC-C workload. Though the TPC-C workload has a high amount of write-traffic (14.56% of the total number of requests issued), even for the RAID-4 configuration, beyond 26 disks, there was little variation in the response time. On the other hand in TPC-H, that has a lesser percentage of write requests (8.76%), the RAID-4 configuration showed greater performance sensitivity when the number of disks were increased compared to the other two RAID configurations. This is due to a combination of two factors, namely, the inter-arrival time of the requests and the size of the data accessed per write request. It was found that the average inter-arrival time between requests for RAID-4 TPC-C was 119.78 ms whereas it was 59.01 ms for TPC-H. Further, for TPC-C, 98.17% of the writes spanned atmost one stripe-unit whereas in TPC-H, 99.97% of the writes were over 2 stripe-units. These two factors caused a greater amount of pressure to be put on the parity disk. When the number of disks increases, there is some improvement across the three configurations (due to increase in parallelism), but this improvement is marginal, except for RAID-4 TPC-H. It should be noted that there are some variations when increasing the disks because of several performance trade-offs. The benefit is the improvement in bandwidth with parallelism, and the downside is the additional overheads that are involved (latency for requests to more disks, and the SCSI bus contention). But these variations are not very significant across the range of disks studied here, and the main point to note from these results is that there is not much improvement in response time beyond a certain number of disks.

On the other hand, the energy consumption keeps rising with the number of disks that are used for the striping. If we look at the detailed energy profile in Figure 8, we observe that most of the energy is in the idle component (as mentioned earlier). When the number of disks is increased, the positioning component does not change much, but the idle component keeps increasing linearly, impacting the overall energy consumption trend as well.

Comparing the two workloads, we find that the active energy (though a small component in both) is more noticeable in TPC-H compared to TPC-C. This is because the former does much more data transfers than the latter. In terms of head positioning power, again TPC-H has a larger fraction of the overall budget, because the number of seeks and average seek distances are higher in this workload (as shown in Table 3). We feel this happens because TPC-H queries can manipulate several tables at the same time, while TPC-C queries are more localized.

One interesting observation that can be seen in both the energy results in Figure 7 and Figure 8 is that the

total and the breakdown are comparable across the three RAID configurations for a given number of disks. To investigate this further, in Table 4, we show the average queue length to different groups of disks for each RAID configuration: (i) for the normal data disks and the parity disk separately in RAID-4, (ii) average over all the disks for RAID-5, and (iii) for each of the two mirrors in RAID-10. We see that the load on the disks (except for the parity disk in RAID-4 which is known to be a bottleneck) across the configurations is comparable, regardless of the number of disks in the range chosen. Since the idle energy is directly related to the load on the disks, and the loads are comparable, the overall energy (which is dominated by the idle energy) and its breakdown are more or less similar across the configurations.

If we are only interested in performance, one can keep increasing the number of disks. This is a common trend in the commercial world where vendors publish TPC results with large disk configurations (even though the improvements may be marginal). On the other hand, power dissipation gets worse with the number of disks. The relative influence of the two factors depends on the nature of the workload. The energy growth is in fact much more influential of these two factors for TPC-C and for RAID-10 in TPC-H , and is the determining factor in the energy-response time product graph. However, the performance gains of using more disks plays a more dominant role in the product for RAID-4 and RAID-5 TPC-H.

5.2.2 Impact of Stripe Size

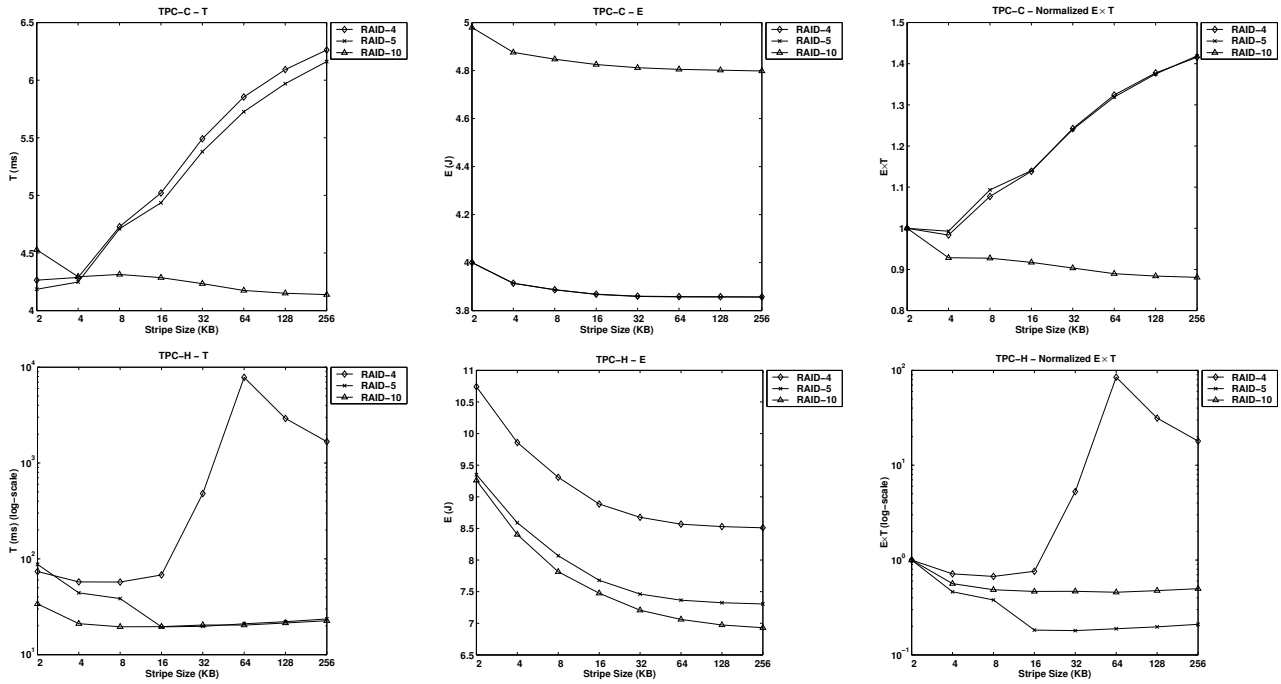


Figure 9: Impact of the Stripe Size

Figure 9 shows the impact of varying stripe sizes on the three RAID configurations for the two workloads. In these experiments, the number of disks used for each configuration was chosen based on what gave the best energy-response time product in Figure 7 (24 for RAID levels 4 and 5 in TPC-C and 30 for RAID-10; 38,32, and 30 for RAID levels 4,5, and 10 of TPC-H respectively). Figure 10 breaks down the energy consumption in these executions into the idle, active and positioning components.

It is to be expected that a higher stripe size will lead to less head positioning latencies/overhead, providing better scope for sequential accesses and possibly involve fewer disks to satisfy a request. However, this can adversely affect the degree of parallelism, which can hurt performance. We find the latter effect to be more significant in determining the performance. Between the two workloads, TPC-H requests are larger, and hence

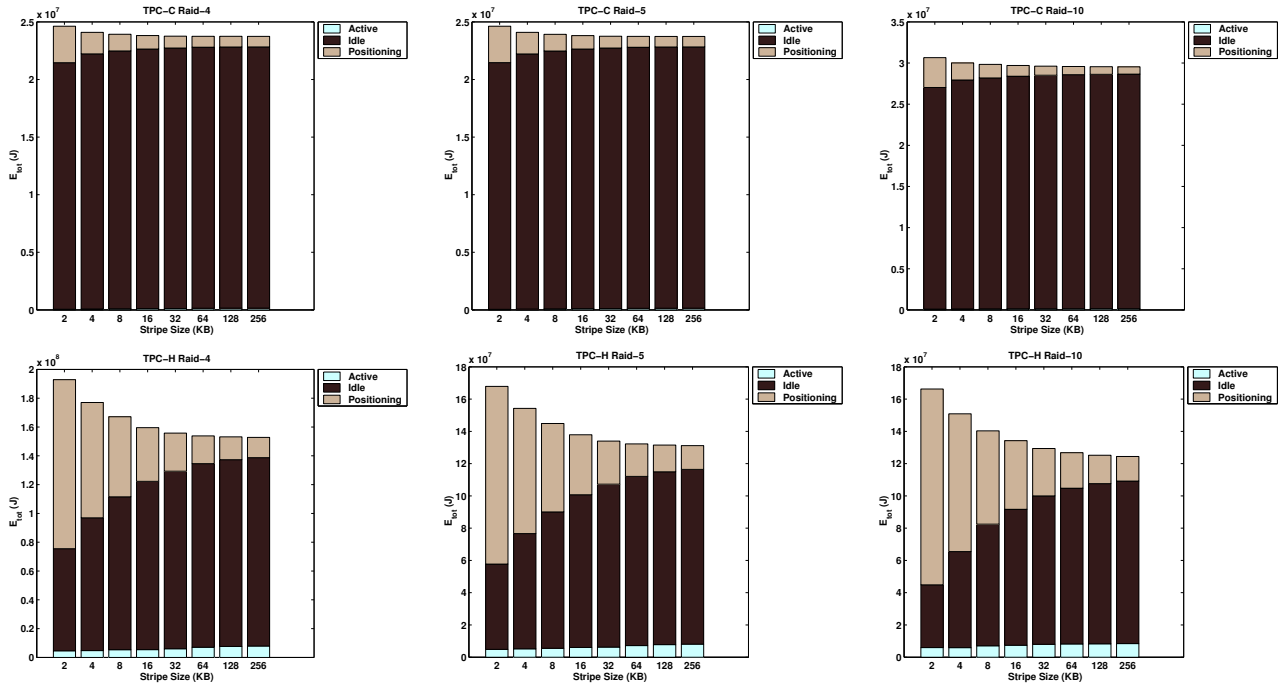


Figure 10: Impact of the Stripe Size - Breakdown of Total Energy Consumption (E_{tot})

the point where the adverse effects become more significant tend to shift to the right for TPC-H. Of the RAID configurations, RAID-4 is more susceptible to stripe size changes because the sequentiality problem is higher there, i.e. one disk (the parity) can turn out to become a bottleneck. This becomes worse for TPC-H, which exercises the parity disk to a larger extent due to reasons explained in the previous section, making RAID-4 performance much worse. RAID-5 and RAID-10 for TPC-H are much better (note that the y-axis for TPC-H response time and energy-response time product graphs are in log-scale to enhance the readability of the RAID-5 and RAID-10 lines) though the overall above explanations with regard to stripe size still hold.

Increasing stripe size can lead to fewer disks being involved per request, and higher sequential accesses per disk (reducing seek overheads). Consequently, the head positioning overheads drop, having a consequence on its energy decrease as is shown in the energy profile graphs of Figure 10. This drop in positioning energy causes the overall energy to decrease as well. For both the workloads, the decrease in the energy consumption is not significant beyond a certain point. This is because of two other issues: (i) the idle component for the disks not involved in the transfer goes up (as can be seen in the increase in idle energy), and (ii) the active component for the disks involved in the transfer goes up (the number of accesses per disk does not linearly drop with the increase in stripe size, making this number degrade slower than ideal, while the transfer energy per request grows with the stripe size). These two offset the drop in the positioning component. This effect is much more pronounced for TPC-C compared to TPC-H, since in the latter the positioning overhead is much higher, as mentioned in section 5.2.1. Finally, the reader should note that despite these overall energy changes, the percentage variations are in fact quite small since the scale of the energy graphs in Figure 9 is quite magnified.

The energy-response time product indicates that response time is a much more significant factor in determining stripe size than energy variations components when stripe size is increased.

Different criteria thus warrant a different stripe size. If performance is the only goal, a smaller stripe size of around 4KB appears to be a good choice. If energy is the only goal, then wider stripes of around 256K seem better (though the energy savings may not be significantly better than a smaller stripe). Overall, a stripe size of 4-16K seems a good choice from the energy-response time product perspective.

5.2.3 Implications

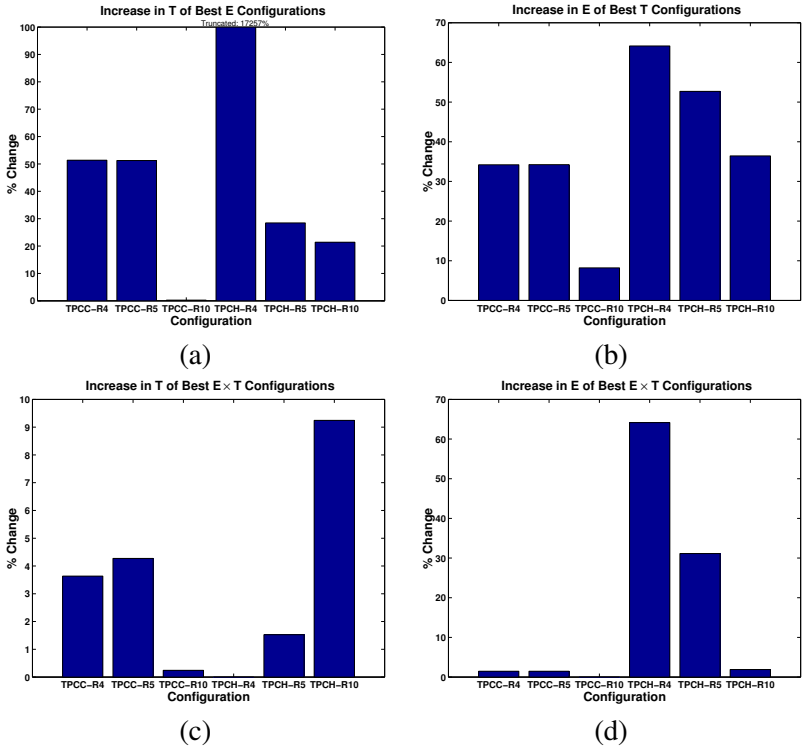


Figure 11: The Effect of Tuning with different Performance and Energy Criteria

Benchmark	RAID Level	Best T	Best E	Best E × T
TPC-C	RAID-4	32/4 KB	24/256 KB	24/4 KB
	RAID-5	32/4 KB	24/256 KB	24/4 KB
	RAID-10	32/4 KB	30/256 KB	30/256 KB
TPC-H	RAID-4	38/8 KB	24/256 KB	38/8 KB
	RAID-5	38/32 KB	24/256 KB	32/32 KB
	RAID-10	38/8 KB	30/256 KB	30/64 KB

Table 5: Optimal Configurations for the Workloads. For each configuration, the pair of values indicated give the number of disks used and the stripe-size employed.

Having conducted an investigation of the different parameters, we put these results in perspective in Figure 11 which shows the trade-offs between performance tuning and energy tuning. We show four graphs in this figure: (a) the percentage increase (over the best-performing version) in response time for the best-energy (per I/O request) version; (b) the percentage increase (over the best-energy version) in energy consumption for the best-performing version; (c) the percentage increase (over the best-performing version) in response time for the best energy-response time product version; and (d) the percentage increase (over the best-energy version) in energy consumption for the best energy-response time product version. Table 5 gives the configurations that generate the best E, T, and $E \times T$ values. Overall, we observe that performance and energy optimizations can lead to very different choices of system configurations. We would like to mention that overall trends presented in this paper are not very different even when we go for different dataset sizes.

6 Conclusions and Future Work

This paper has conducted an in-depth examination of the power and performance implications of disk arrays (RAIDs) that are used for transaction processing workloads. It has used real traces of TPC-C and TPC-H, and has simulated their execution on three different RAID configurations (RAID-4, RAID-5, and RAID-10) using DiskSim [13] which has been extended with power models. From this detailed examination, this paper makes the following contributions:

- We show that conventional power mode control schemes which have been extensively used in laptop/workstation environments do not show much benefit for these workloads if we do not have the luxury of stretching response times. Even though the idle power is quite high and so is the number of idle periods, the duration of each is rather small. This makes it difficult to offset the high spindown/spinup costs associated with server disks (even if we assume very optimistic costs for these). This is true even if we had a perfect oracle predictor of idle periods. Another problem with frequent spinup/spindown operations is the decrease in mean time between failures, which is an important consideration for server environments. This can turn out to be another reason against using explicit spindowns for server disk power optimization.

- On the other hand, with the current state of technology, tuning of RAID parameters has more to gain, and allows the scope for different optimizations - whether power or performance. Increasing the number of disks, though adversely impacts the energy consumption, may buy significant performance benefits depending on the nature of the workload. The choice of stripe size, on the other hand, is more determined by the performance angle than the energy consumption. We found that these parameters had a more determining impact on energy and response time than the choice of RAID configuration itself (particularly RAID-5 and RAID-10) since the load on the disks are comparable.

This research takes a step towards the eventual goal of being able to make good apriori, power-aware RAID design decisions in an automated manner, as suggested in [2, 1]. Our ongoing work is examining issues about extending idle times by possibly batching requests (studying the trade-offs between extending response times and saving energy), other server workloads (web servers and hosting centers), together with incorporating power models for the data transfers between the host and the I/O subsystem. Another avenue for research that we plan to explore is to design disk arrays that use a combination of disks with different performance and energy consumptions. There are several interesting issues related to reducing the energy consumption without significantly affecting performance by appropriately directing the requests to the different disks, ensuring good load-balance etc.

References

[1] E. Anderson, M. Kallahalla, S. Spence, and R. Swaminathan. Ergastulum: an approach to solving the workload and device configuration problem. Technical Report HPL-SSP-2001-05, HP Laboratories Storage Systems Program, 2001.

[2] E. Anderson, R. Swaminathan, A. Veitch, G.A. Alvarez, and J. Wilkes. Selecting RAID Levels for Disk Arrays. In *Proceedings of the Conference on File and Storage Technology (FAST)*, pages 189–201, January 2002.

[3] P. Bohrer, D. Cohn, E. Elnozahy, T. Keller, M. Kistler, C. Lefurgy, R. Rajamony, F. Rawson, and E.V. Hensbergen. Energy Conservation for Servers. In *IEEE Workshop on Power Management for Real-Time and Embedded Systems*, May 2001.

[4] P. Bohrer, E.N. Elnozahy, T. Keller, M. Kistler, C. Lefurgy, C. McDowell, and R. Rajamony. *The Case for Power Management in Web Servers*, chapter 1. Kluwer Academic Publications, 2002.

[5] G.E. Box and G.M. Jenkins. *Time Series Analysis Forecasting and Control*. Holden-Day, 2nd edition, 1976.

[6] J. Chase and R. Doyle. Balance of Power: Energy Management for Server Clusters. In *Proceedings of the 8th Workshop on Hot Topics in Operating Systems (HotOS)*, May 2001.

[7] J.S. Chase, D.C. Anderson, P.N. Thakur, A.M. Vahdat, and R.P. Doyle. Managing Energy and Server Resources in Hosting Centers. In *Proceedings of the 18th ACM Symposium on Operating Systems Principles (SOSP'01)*, pages 103–116, October 2001.

[8] D. Colarelli, D. Grunwald, and M. Neufeld. The Case for Massive Arrays of Idle Disks (MAID). In *USENIX Conference on File and Storage Technologies (FAST'02) Work-in-Progess Session*, January 2002.

[9] Fred Douglis and P. Krishnan. Adaptive Disk Spin-Down Policies for Mobile Computers. *Computing Systems*, 8(4):381–413, 1995.

[10] E.N. Elnozahy, M. Kistler, and R. Rajamony. Energy-Efficient Server Clusters. In *Proceedings of the Workshop on Power-Aware Computer Systems (PACS'02)*, pages 124–133, February 2002.

[11] Fujitsu. http://www.fujitsu.com.

[12] G.R. Ganger. *System-Oriented Evaluation of I/O Subsystem Performance*. PhD thesis, The University of Michigan, June 1995.

[13] G.R. Ganger, B.L. Worthington, and Y.N. Patt. *The DiskSim Simulation Environment Version 2.0 Reference Manual*. http://www.ece.cmu.edu/ ganger/disksim/.

[14] R. Golding, P. Bosch, and J. Wilkes. Idleness is not sloth. Technical Report HPL-96-140, HP Laboratories, October 1996.

[15] T. Heath, E. Pinheiro, and R. Bianchini. Application-Supported Device Management for Energy and Performance. In *Proceedings of the Workshop on Power-Aware Computer Systems (PACS'02)*, pages 114–123, February 2002.

[16] D. Helmbold, D. Long, T. Sconyers, and B. Sherrod. Adaptive Disk Spin-Down for Mobile Computers. *ACM/Baltzer Mobile Networks and Applications (MONET) Journal*, To Appear.

[17] IBM DB2. http://www-3.ibm.com/software/data/db2/.

[18] IBM Hard Disk Drive - Travelstar 40GNX. http://www.storage.ibm.com/ hdd/ travel/ tr40gnx.htm.

[19] IBM Hard Disk Drive - Ultrastar 36ZX. http://www.storage.ibm.com/ hdd/ ultra/ ul36zx.htm.

[20] Intel Corporation. http://www.intel.com.

[21] Jennifer Jones and Brian Fonseca. Energy Crisis Pinches Hosting Vendors. http://iwsun4.infoworld.com/articles/hn/xml /01/01/08/010108hnpower.xml.

[22] Kester Li, Roger Kumpf, Paul Horton, and Thomas E. Anderson. Quantitative Analysis of Disk Drive Power Management in Portable Computers. In *Proceedings of the USENIX Winter Conference*, pages 279–291, 1994.

[23] Y-H. Lu, E-Y. Chung, T. Simunic, L. Benini, and G.D. Micheli. Quantitative Comparison of Power Management Algorithms. In *Proceedings of the Design Automation and Test in Europe (DATE)*, March 2000.

[24] Y-H. Lu and G.D. Micheli. Adaptive Hard Disk Power Management on Personal Computers. In *Proceedings of the IEEE Great Lakes Symposium*, March 1999.

[25] A. Moshovos, G. Memik, B. Falsafi, and A.N. Choudhary. JETTY: Filtering Snoops for Reduced Energy Consumption in SMP Servers. In *Proceedings of the 7th International Symposium on High-Performance Computer Architecture (HPCA)*, January 2001.

[26] V.S. Pai. *Cache Management in Scalable Network Servers*. PhD thesis, Rice University, November 1999.

[27] D. Patterson, G. Gibson, and R. Katz. A Case for Redundant Arrays of Inexpensive Disks (RAID). In *Proceedings of ACM SIGMOD Conference on the Management of Data*, pages 109–116, 1988.

[28] E. Pinheiro, R. Bianchini, E. V. Carrera, and T. Heath. Load Balancing and Unbalancing for Power and Performance in Cluster-Based Systems. In *Proceedings of the Workshop on Compilers and Operating Systems for Low Power*, September 2001.

[29] TPC-C Benchmark V5. http://www.tpc.org/tpcc/.

[30] TPC-C Executive Summary - - Dell PowerEdge 4600/2.2/1P. http://www.tpc.org/tpcc/results/tpcc_result_detail.asp?id=102062603.

[31] TPC-C Executive Summary - Dell PowerEdge 6650/4/1.6GHz. http://www.tpc.org/tpcc/results/tpcc_result_detail.asp?id=102053101.

[32] TPC-H Benchmark. http://www.tpc.org/tpch/.

[33] N.N. Tran. *Automatic ARIMA Time Series Modeling and Forecasting for Adaptive Input/Output Prefetching*. PhD thesis, University of Illinois at Urbana-Champaign, 2002.

[34] Transaction Processing Performance Council. http://www.tpc.org/.

[35] R. Youssef. RAID for Mobile Computers. Master's thesis, Carnegie Mellon University Information Networking Institute, August 1995.