On the Impact of Disk Scrubbing on Energy Savings

Guanying Wang†, Ali R. Butt†, and Chris Gniady‡
†Virginia Tech *‡University of Arizona*
{wanggy,butta}@cs.vt.edu *gniady@cs.arizona.edu*

Abstract

The increasing use of computers for saving valuable data imposes stringent reliability constraints on storage systems. Reliability improvement via use of redundancy is a common practice. As the disk capacity improves, advanced techniques such as disk scrubbing are being employed to proactively fix latent sector errors. These techniques utilize the disk idle time for reliability improvement. However, the idle time is a key to dynamic energy management that detects such idle periods and turns-off the disks to save energy. In this paper, we are concerned with the distribution of the disk idle periods between reliability and energy management tasks. For this purpose, we define a new metric, energy-reliability product (ERP), to capture the effect of one technique on the other. Our initial investigation using trace-driven simulations of typical enterprise applications shows that the ERP is a suitable metric for identifying efficient idle period utilization. Thus, ERP can facilitate development of systems that provide both reliability and energy managements.

1 Problem Statement

Users count on computer systems to store data reliably. With improvements in storage media densities that increase disk capacities, the chances of data loss are becoming higher. Also, modern enterprises utilize a large number of devices, which further increases the probability that some disk failures will occur. Recent works [21, 19] have identified many reasons for data loss such as crash failures where disks become completely inaccessible, and latent sector errors where portions of stored data are corrupted but the disks remain accessible. As users increasingly store valuable, irreplaceable data on computers, the reliability requirements are becoming more stringent.

Employing a large number of storage devices not only leads to an increase in the number of failures in the system but also results in higher energy consumption. Recent research has shown that the use of energy conservation techniques has both financial and environmental impacts [3, 9, 18], and that disks consume significant energy [25]. Consequently, many organizations employ active energy management. Therefore, organizations are faced with two challenges of providing high reliability and reducing energy consumption of the storage systems. However, addressing these challenges can be conflicting and entail careful consideration for optimum system reliability, performance, and energy efficiency.

One of the commonly used mechanisms for improving data reliability is RAID [17] that provides protection against total disk failures. However, most recent disk failures are due to portions (sectors) of a disk going bad [19]. Such sectors may not be read and discovered to be corrupted in time when they can be fixed, leading to eventual data loss. To fend against such intra-disk failures, a number of techniques such as disk scrubbing [1, 22] are being employed. These reliability improving techniques periodically examine the contents of the disks for latent sector failures, so that proactive action can be taken to avoid data loss and exposing the failures to the end-user. To minimize performance impact, these techniques typically run as background jobs that utilize the periods when the disk is idle to perform data integrity checks.

The use of disk idle periods for improving reliability, e.g., for scrubbing, conflicts with energy saving mechanisms that want to turn the disks off during these periods [10, 16] and thus preclude scrubbing. This raises the question whether the two techniques can co-exist. In this paper, we examine this question and attempt to understand the effect of reliability improving techniques on disk energy savings. For this purpose, we use a trace-driven simulation study to investigate the design trade-offs and their implications.

We explore the similarities between energy-performance optimization and energy-reliability optimization to study energy efficiency in the presence of reliability improvement techniques. Energy management and performance are conflicting optimizations, since shutting the disk down will introduce additional delays potentially degrading the performance. To provide a balanced design, researchers rely on energy-delay product (EDP) [11]. Lowest EDP indicates both low energy consumption and low delay, therefore maximizing overall system efficiency. Similarly as EDP is used to capture the effect of energy savings on performance, we propose the concept of energy-reliability product (ERP) to capture the combined performance of energy saving and reliability improving approaches. This metric will provide a unified mechanism for evaluating both energy efficiency and data reliability in the system. The challenge lies in quantifying reliability and understanding the meaning of the energy and reliability product. There are many possible alternatives, e.g., user-specified expected reliability or energy savings level. However, we argue that ERP is necessary in evaluating different combinations of energy

management and reliability scenarios. We need a metric that will allow us to quantitatively compare two different designs, just as EDP is helping in design comparison in energy/performance community. We believe that ERP is that metric and we show its applicability using some intuitive examples.

2 Background

Extensive research studies have been performed individually on energy management and reliability, however, research effort that takes both areas into consideration has been lacking. In the following, we briefly review background work in each individual area.

Reliability Improvement Techniques: Failures and errors in storage systems can eventually lead to data loss. To avoid this, all modern systems employ some form of reliability improving techniques, ranging from simple inter- and intra-disk replication to large-scale RAID [17] systems with intra-disk redundancy. The goal is to increase Mean Time To Data Loss (MTTDL), a metric used for measuring reliability.

RAID [17] stores error-coded data across multiple disks to recover from entire disk failures. However, with latent sector errors in modern high-capacity disks [19], the redundancy provided by RAID can be compromised due to data corruption. This is especially bad in the case of infrequently-accessed data, which can become corrupted due to intra-disk errors. The errors remain undetected until some disk crashes, and RAID fails to recover the data.

To avoid data loss due to intra-disk errors, disk scrubbing [1, 22] is employed in conjunction with RAID to discover latent errors in time. Scrubbing is done periodically with a fixed interval between two consecutive cycles, referred to as the scrubbing period. Scrubbing reduces the probability of data loss by reading the whole disk periodically and detecting errors that will not otherwise be found by standard data accesses. A detected error can then be fixed using RAID.

In contrast to scrubbing, intra-disk redundancy [15] keeps multiple copies of data on a disk to enable recovery from latent sector errors, and IRAW [20] keeps the data in a disk cache till it has been written and verified. These approaches can achieve at least as high a reliability as scrubbing, but may impose performance overheads.

Energy Management Techniques:

Energy management in typical enterprises range from encouraging users to power down their computers once they leave work, to dynamic approaches that detect when devices are not in use and move them into low-power modes. The most popular approach shuts down the device after a period of inactivity [7, 13] as shown in Figure 1. Subsequent disk I/O will require a disk spin-up that consumes energy. For this reason, a *break-even time* is defined as the smallest time interval for which a disk must stay off to justify the extra energy spent in spinning the disk up and down. Also, spin-ups can result in multi-second delays in response, and reduce performance and energy efficiency. To remedy this, many dynamic predictors that shut down the device much earlier than the timeout mechanisms have been investigated [4, 14]. To improve accuracy, energy management can be relegated to programmers, since they have a better idea of what the application, and potentially the users, are doing at a given time [8, 24]. Finally, Power-Aware RAID (PARAID) [23] takes into account the power consumption of RAID systems. However, PARAID does not consider intra-disk reliability improvement techniques and their energy impact, and is orthogonal to the presented work.

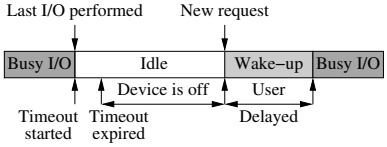


Figure 1: Anatomy of an idle period.

3 ERP: Reliability vs. Energy Efficiency

We face two alternatives for utilizing disk idle time. On one hand, the idle period provides opportunities for spinning down the disk and saving energy, on the other hand it can provide time for techniques to improve reliability without performance penalties. While advanced techniques exists for both, we initially focus on how basic energy saving, e.g., timeout based shutdown, and reliability improving approaches, e.g., disk scrubbing (Section 2), can co-exist. The goal of this paper is to evaluate basic mechanisms for idle time allocation between energy management and reliability mechanisms to achieve a balance between energy consumption and reliability.

Figure 2 shows an example time distribution for some workload. The time spent in performing I/O requests is independent of the scrubbing or energy saving mechanisms. Energy consumed for satisfying I/O requests is dictated by the application behavior in the system. During busy periods the reliability of the disk drive is not affected and corresponds to the basic disk reliability [15]. However, when the disk is idle, part of the idle time can be used to improve reliability and part to save energy. The key observation here is that if more of the idle time is allotted to disk scrubbing, the expected MTTDL will increase, which in turn decreases the probability of failures and thus improves reliability. But, less time will be available for spinning the disk down and additional energy will be consumed to perform scrubbing, thus decreasing energy efficiency. Spending more energy

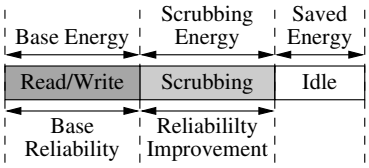


Figure 2: Disk activity in the system.

will increase reliability; however the goal is to minimize energy consumption while maximizing the reliability.

We can draw an analogy to energy-performance optimizations, where energy savings are increased at a cost of decrease in performance. The popular metric to combine both performance impact as well as energy savings is expressed as an energy-delay product. Consequently, researches are faced with the task of minimizing a single value, the corresponding energy-delay product [11], to improve overall system efficiency. To provide a similar metric for evaluating energy and reliability, we propose the energy-reliability product (ERP) to capture the impact of energy and reliability improving techniques on each other. We focus on the amount of additional energy consumed due to scrubbing and the increase in reliability it generates. We do not include the base energy consumption or reliability since it is independent of energy management or reliability mechanisms. Energy-reliability product can also be expressed in terms of energy savings generated in the system and the increase in reliability due to scrubbing.

Since reliability cannot be simply measured, we define the ERP in terms of MTTDL as follows:

$$ERP = Energy\ savings * Reliability\ improvement$$
$$= \Delta Energy * \Delta MTTDL, \quad (1)$$

where $\Delta Energy$ is the energy saved, and $\Delta MTTDL$ is the corresponding improvement in reliability. Recently, detailed models [15, 6] have been developed that show how MTTDL is affected by varying scrubbing period, when disk scrubbing is employed. Based on these models, under typical workload and scrubbing periods, the MTTDL increases with decreasing scrubbing period. Thus, the above equation can be simplified to:

$$\Rightarrow ERP \propto \Delta Energy * 1/Scrubbing\ Period \quad (2)$$

This leaves us with the objective to maximize ERP through idle-time allocation that achieves the best combination of energy savings and reliability.

Finally, the simple disk model that we have used can be extended to handle advanced multi-speed disks or sophisticated controls such as varying disk speeds [12]. Also, we explore the energy-reliability relationship in terms of distribution of disk idle times. A more traditional relationship between the two also exists, i.e., more power dissipation can lead to thermal stress on the disk causing it to fail. We assume that the systems are properly cooled and remain within acceptable working thermal limits, thus we do not capture this.

4 Methodology

The goal of this paper is to understand the impact of maximizing ERP on both energy and reliability in a typical workstation in an enterprise environment where both high reliability and energy savings are important. Consequently, we use trace-driven simulations of typical enterprise applications to study the effects of disk scrubbing and energy management on each other. Detailed traces of user-interactive sessions for each application were obtained by a `strace`-based tracing tool [5, 2] over a number of days. Table 1 shows the specifications of the disk that we used in our simulation. The WD2500JD has a spin-up time of about 9 seconds from a sleep state, which is common in high-speed commodity disks.

State	
Read/Write Power	10.6W
Seek Power	13.25W
Idle Power	10W
Standby Power	1.8W
Spin-up Energy	148.5J
Shutdown Energy	6.4J
State Transition	
Spin-up time	9 sec.
Shutdown time	4 sec.

Table 1: Western Digital WD2500JD specifications.

Table 2 shows six desktop applications that are popular in enterprise environments: *Mozilla* web browser, *Mplayer* music player, *Impress* presentation software, *Writer* word processor, *Calc* spreadsheet, and *Xemacs* text editor. The table also shows trace length and the details of I/O activity. I/Os satisfied by the buffer cache are not counted, since they do not cause disk activity.

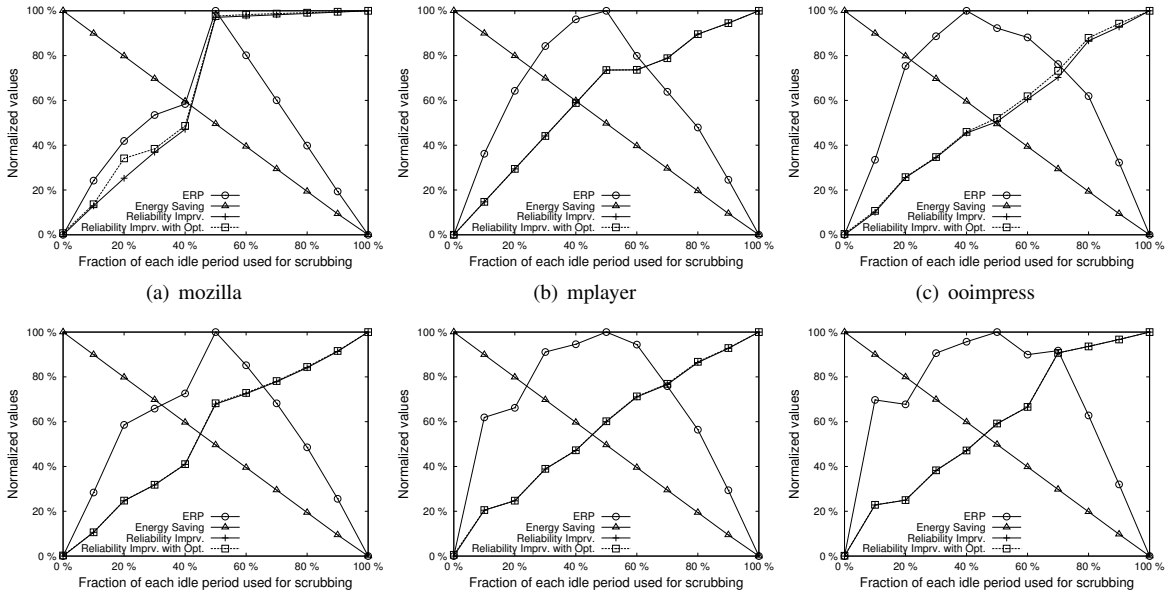
Appl.	Trace Length [hr]	Number of Reads	Number of Writes	Referenced [MB] Reads	Referenced [MB] Writes
mozilla	45.97	13005	2483	66.4	19.4
mplayer	3.03	7980	0	32.3	0
impress	66.76	13907	1453	92.5	40.1
writer	54.19	7019	137	43.8	1.2
calc	53.93	5907	93	36.2	0.4
xemacs	92.04	23404	1062	162.8	9.4

Table 2: Traces collected for the studied applications.

5 Results

For each of the studied application, we simulate the energy consumed and the time it would take for a scrubbing cycle to complete using the disk model of Table 1. Next, we use the scrubbing period to estimate the improvement in reliability, and finally use equation (2) to determine the ERP for each measurement. For each application, we dedicate an increasing portion of an idle period to scrubbing (0% to 100%) and plot the resulting changes in reliability and energy, as well as the ERP. The reason for using such simple mechanisms is to understand the meaning of the value conveyed by ERP. This allows us to reason the impact, and observe whether ERP follows our intuition about energy savings and reliability improvements.

Figure 3 shows the normalized (w.r.t. maximum) energy savings, the corresponding improvement in reliability and the ERP, for each application. We also used an optimization (shown using dashed line) where all idle periods that are smaller than the break-even time are entirely used for scrubbing. This does not affect the energy savings, but improves reliability. Observe that the peak ERP points to a distribution of idle periods that can provide an efficient combination of energy savings and reliability.



(a) mozilla (b) mplayer (c) ooimpress

(d) oowriter (e) oocalc (f) xemacs

Figure 3: Normalized energy savings and associated improvements in reliability under varying fractions of idle periods.

Appl.	Two-phase Allocation		Scrubbing only in small idle periods		Alternate Allocation	
	Energy savings	Reliab. Imprv.	Energy savings	Reliab. Imprv.	Energy savings	Reliab. Imprv.
mozilla	96.2%	5.1%	99.9%	0.7%	54.3%	35.5%
mplayer	42.6%	84.4%	99.9%	0.0%	99.9%	0.0%
impress	97.5%	2.6%	99.9%	0.4%	41.6%	71.3%
writer	96.8%	3.2%	99.9%	0.3%	46.8%	60.9%
calc	96.8%	4.1%	99.9%	0.6%	30.2%	43.8%
xemacs	98.1%	2.4%	99.9%	0.1%	70.9%	74.5%

Table 3: Energy savings and improvement in reliability achieved under different idle-period allocations.

Next, we study four alternate schemes for allocating idle periods. In the first scheme, we adopt a two-phase approach. In the first phase, we dedicate all idle periods for scrubbing. This phase continues until the entire disk is scrubbed once. Then the second phase starts, where all idle periods are used exclusively for energy savings and no scrubbing is done. In this case, the scrubbing period equals the trace length. The 'Two-phase' column of Table 3 shows the energy consumed and the associated reliability improvement for each of the studied application. The numbers are normalized against those of the previous experiments. The results show that although this scheme saves energy, the reliability improvement is very small, with an ERP of less than 24.8% averaged across all applications. This approach is not advisable for two reasons: (i) the scrubbing cycle is significantly large and the resulting MTTDL very small; and (ii) in the energy-saving phase, all idle periods that are smaller than the break-even time are wasted, as they neither provide energy-savings nor contribute to improving reliability.

In the second scheme, we use periods smaller than break-even time for scrubbing – avoiding loss in energy savings – and longer periods for spinning-off the disks. Table 3 also shows the result for this approach. Once again, although energy savings are maximized ($\approx 100\%$), the scrubbing period increases significantly, thus resulting in negligible ($\leq 0.7\%$) reliability improvement, and an $ERP \approx 0\%$.

In the third scheme, we alternate between scrubbing and spinning-down the disk on each idle period. All idle periods smaller than the break-even time are once again used only for scrubbing. The 'Alternate' column of Table 3 shows the results. Note that the reliability improvement in *mplayer* is negligible. This is due to bursty nature of the workload that has only a few large idle periods, which helps in saving energy but do not provide enough time for scrubbing. Overall, this scheme achieves good energy savings (average 47.7%) while improving reliability (average 57.3%). The average ERP of 76.4% (179.5% for *xemacs*) suggests that this is a viable approach compared to the first two.

Finally, we investigate the reliability impact of the commonly used timeout based approach for energy management. A timer with a fixed timeout interval is started whenever the disk is idle, and the disk is spun down when the timer expires. Scrubbing is only done during the timeout interval, i.e., the time between the disk becoming idle and being shut down. Figure 4 shows the resulting energy savings and reliability improvements for the studied applications under various timeout intervals. Note that for this case, the energy comparison is done against a simple timeout scheme without scrubbing. It is observed that such an approach, while providing almost 100% of the possible energy savings, does not provide enough opportunity for scrubbing, thus achieving little reliability improvement. The problem with simply using a longer timeout is that the resulting reliability improvement depends on the idle

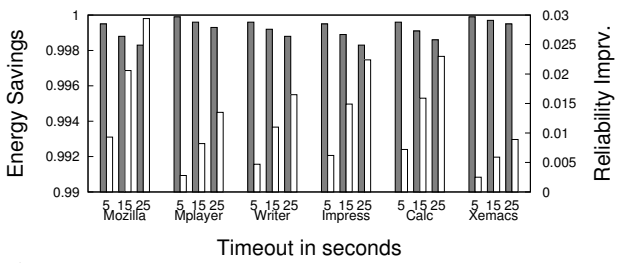


Figure 4: Normalized energy savings (gray) and improvement in reliability (white) achieved under a timeout based approach with different timeouts.

period distribution. For example, if there are a large number of idle periods, the longer timeout would provide desired higher reliability at a cost of energy savings. On the other hand, if we imagine a scenario where there is only one very long busy period followed by one long idle period, the one timeout interval spent on scrubbing is not sufficient for significantly improving reliability.

Discussion: We have observed how idle period distribution affects reliability and energy savings. It is clear that further study is needed to broadly understand the implication of ERP and its impact on designing both energy efficient and reliable systems. Once the implications are clear, further research is needed in more dynamic energy-reliability management techniques. Different workload characteristics of the applications affect the portion of time dedicated to either one of the energy or reliability improvement techniques. Therefore, adaptive techniques are needed, which dynamically monitor system characteristics and provide online energy-reliability balancing mechanisms. The analysis and identification of such use cases remain a focus of our ongoing research.

6 Conclusion

In this paper, we have explored the interactions between reliability and energy management in disks. We argue that similarly as the energy-delay product metric is used for capturing the effect of energy management on system performance, we should consider an energy-reliability product to evaluate combination of reliability and energy efficiency of storage systems. Using trace-driven simulations of several enterprise applications, we have shown that ERP can help identify efficient distribution of disk idle time to energy and reliability management. In our study, we have relied on simpler techniques both for energy and reliability management, but the evaluation techniques using ERP can guide the design of more advanced energy-reliability management.

References

[1] L. N. Bairavasundaram, G. R. Goodson, S. Pasupathy, and J. Schindler. An Analysis of Latent Sector Errors in Disk Drives. In *Proc. SIGMETRICS*, 2007.

[2] A. R. Butt, C. Gniady, and Y. C. Hu. The performance impact of kernel prefetching on buffer cache replacement algorithms. *IEEE Transactions on Computers*, 56(7):889–908, 2007.

[3] J. Chase, D. Anderson, P. Thackar, A. Vahdat, and R. Boyle. Managing energy and server resources in hosting centers. In *Proc. SOSP*, 2001.

[4] E.-Y. Chung, L. Benini, and G. D. Micheli. Dynamic power management using adaptive learning tree. In *Proc. ICCAD*, 1999.

[5] I. Crk and C. Gniady. Context-aware mechanisms for reducing interactive delays of energy management in disks. In *Proc. USENIX ATC*, 2008.

[6] A. Dholakia, E. Eleftheriou, X.-Y. Hu, I. Iliadis, J. Menon, and K. Rao. A new intra-disk redundancy scheme for high-reliability raid storage systems in the presence of unrecoverable errors. *Trans. Storage*, 4(1):1–42, 2008.

[7] F. Douglis, P. Krishnan, and B. Bershad. Adaptive Disk Spin-down Policies for Mobile Computers. In *Proc. USENIX Symp. on Mobile and Location-Independent Computing*, 1995.

[8] C. S. Ellis. The Case for Higher-Level Power Management. In *Proc. ACM HotOS*, 1999.

[9] M. Elnozahy, M. Kistler, and R. Rajamony. Energy conservation policies for web servers. In *Proc. USITS*, 2003.

[10] C. Gniady, A. R. Butt, Y. C. Hu, and Y.-H. Lu. Program counter-based prediction techniques for dynamic power management. *IEEE Transactions on Computers*, 55(6):641–658, 2006.

[11] R. Gonzalez and M. Horowitz. Energy dissipation in general purpose microprocessors. *Solid-State Circuits, IEEE Journal of*, 31(9):1277–1284, Sep 1996.

[12] S. Gurumurthi, A. Sivasubramaniam, M. Kandemir, and H. Franke. Drpm: dynamic speed control for power management in server class disks. *SIGARCH Comput. Archit. News*, 31(2):169–181, 2003.

[13] D. P. Helmbold, D. D. E. Long, and B. Sherrod. A dynamic disk spin-down technique for mobile computing. In *Mobile Computing and Networking*, pages 130–142, 1996.

[14] C.-H. Hwang and A. C. Wu. A Predictive System Shutdown Method for Energy Saving of Event Driven Computation. *ACM Transactions on Design Automation of Electronic Systems*, 5(2):226–241, April 2000.

[15] I. Iliadis, R. Haas, X.-Y. Hu, and E. Eleftheriou. Disk scrubbing versus intra-disk redundancy for high-reliability raid storage systems. In *Proc. SIGMETRICS*, 2008.

[16] Y.-H. Lu, E.-Y. Chung, T. Simunic, L. Benini, and G. D. Micheli. Quantitative Comparison of Power Management Algorithms. In *Proc. DATE*, 2000.

[17] D. A. Patterson, G. Gibson, and R. H. Katz. A case for redundant arrays of inexpensive disks (raid). In *Proc. ACM SIGMOD*, 1988.

[18] E. Pinheiro, R. Bianchini, E. V. Carrera, and T. Heath. Load balancing and unbalancing for power and performance in cluster-based systems. In *Proc. Workshop on Compilers and Operating Systems for Low Power*, 2001.

[19] E. Pinheiro, W.-D. Weber, and L. A. Barroso. Failure trends in a large disk drive population. In *Proc. USENIX FAST*, 2007.

[20] A. Riska and E. Riedel. Idle read after write: Iraw. In *Proc. USENIX ATC*, 2008.

[21] B. Schroeder and G. A. Gibson. Disk failures in the real world: what does an mttf of 1,000,000 hours mean to you? In *Proc. USENIX FAST*, 2007.

[22] T. Schwarz, Q. Xin, E. Miller, D. Long, A. Hospodor, and S. Ng. Disk scrubbing in large archival storage systems. In *Proc. MASCOTS*, 2004.

[23] C. Weddle, M. Oldham, J. Qian, A.-I. A. Wang, P. Reiher, and G. Kuenning. Paraid: a gear-shifting power-aware raid. In *Proc. USENIX FAST*, 2007.

[24] A. Weissel, B. Beutel, and F. Bellosa. Cooperative I/O—a novel I/O semantics for energy-aware applications. In *Proc. OSDI*, 2002.

[25] Q. Zhu, Z. Chen, L. Tan, Y. Zhou, K. Keeton, and J. Wilkes. Hibernator: helping disk arrays sleep through the winter. In *Proc. SOSP*, 2005.