Predictive Reduction of Power and Latency (PuRPLe)[*]

Matthew Craven Ahmed Amer
Department of Computer Science
University of Pittsburgh
mcraven@cs.pitt.edu amer@cs.pitt.edu

Abstract

Increasing efforts have been aimed towards the management of power as a critical system resource, and the disk can consume approximately a third of the power required for a typical laptop computer. Mechanisms to manage disk power have included spin-down policies and APIs to modify access workloads to be more power-friendly. In this work we present a measurement study of disk power consumption, focusing on the potential impact of successfully optimizing disk layout or predicting future disk accesses with predictive read-ahead. We demonstrate how such strategies can allow the reduction of disk power consumption at least as well as traditional disk spin-down schemes, while avoiding the serious performance degradation that can occur from excessive spin-downs. Experimental results showed that a successful predictive disk management policy could reduce disk power consumption by over 80%, while maintaining the responsiveness of a continuously running disk. In contrast, an aggressive spin-down policy that does not attempt to optimize layout or predictively read-ahead data, would achieve the same results at the expense of increasing average delays by 2 to 4 times. Another contribution of this work involves the accuracy of the measurements, which were conducted at a level precise enough to distinguish the power consumption of drive electronics, spindle-motors, and disk arm movement.

1. Introduction

The disk can consume up to a third of the power used by a typical mobile computer [20], and yet the great majority of research into disk optimization has focused on decreasing service delays caused by the storage subsystem. In this work we demonstrate the potential for predictive data management to satisfy these traditionally contradictory performance goals; reducing disk latencies while simultaneously reducing disk power consumption.

Modern disks have several different modes of operation, that can basically be divided into either active (full power consumption) or idle (low power consumption), and the traditional mechanisms for reducing disk power consumption have focused on controlling timing of a disk's spinning-down to the idle state. The complexity of this decision is brought about by the fact that accelerating a disk back up to speed consumes more power for a brief period of time, than simply leaving the disk running for a short period. The amount of power that is needed to bring a disk up to speed varies from disk to disk, and yet can be expressed relatively uniformly using a measure called the "spin-down cost" of a disk. This is the amount of time a disk can be left in running mode before it consumes as much power as would be needed to spin the disk back up. Spinning the disk down for an idle period (the time between disk requests) that exceeds this length of time is profitable, but spinning a disk down for an idle period that is shorter causes the disk to consume more power than if it had been kept in the full power consumption mode. This is in addition to the increased service delays caused by waiting for the disk to accelerate back up to speed.

The simplest mechanism to control disk spin-down is to have a fixed timeout value, and when the system encounters an idle period that exceeds this period, it places the disk in a low power state. This approach is very effective, but is sensitive to the nature of the workload and the length of the time-out value. If this time-out value is too short, then the system would be employing an spin-down policy that is too aggressive, resulting in too many delays and increased power usage. If the timeout value is too long, then the disk remains running (at full power consumption) for longer than is absolutely necessary and energy is wasted in a similar manner. Gener-

[*]Supported in part by the National Science Foundation award ANI-0325353.

ally aggressive spin-down policies can save more power at the expense of increased average service delays.

Prior studies of disk spin-down algorithms have focused on the power savings of adapting the spin-down timeout value, or for modifying the workload through energy-aware APIs. For these studies, the power consumption of the disk is the primary concern, and it is common to expect the delays to be insignificant, or a secondary issue. This is an acceptable strategy due to the difficulty of modeling delays when a workload has been modified due to disk spin-downs. In this work we explicitly consider both the power consumption effects and timing effects of possible predictive optimizations to storage. We do this by using real-world traces replayed against different real disks, and observing the physical power consumption of these hard drives. We then modify these workloads in such a way as to simulate the effects of predictive read-ahead or placement of varying success. In this manner we are able to demonstrate how these particular optimizations can simultaneously satisfy the goals of reducing disk energy usage and disk service delays.

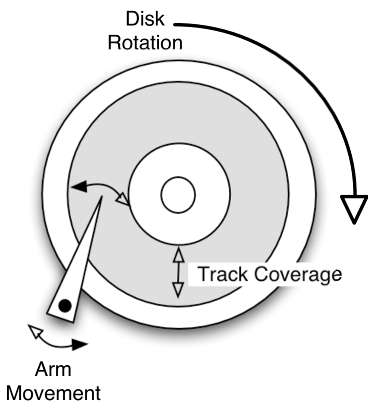
2. Reducing Disk Power and Latency

Predictive management of devices can be used to optimize various performance goals. For example, by prefetching data before it is explicitly requested we are able to reduce latency. This is possible due to the predictable nature of data requests, and the increased burstiness of the workload may also have a direct influence on device power consumption. If the new access behavior results in greater opportunities to spin-down the disk, then we would reduce the power consumed by the disk's rotation. While this approach may appear very promising, it is only one way of using predictive information to improve disk performance. Another alternative is to use predictive information to improve the behavior of a prefetching cache, reducing demand misses. Yet another alternative is to use access predictions to group related data and files to allow their collocation in close physical proximity on disk. This last approach of collocating related data, if successfully applied, has the greatest potential for reducing latencies while simultaneously reducing energy consumption, but we will show below how any of these predictive techniques has the potential to offer such benefits to varying degrees. Simply prefetching data that is likely to be requested is beneficial because it allows the disk to perform more operations in a shorter period of time, and hopefully spend a subsequently longer period of time idle (increasing the likelihood of a beneficial spin-down).

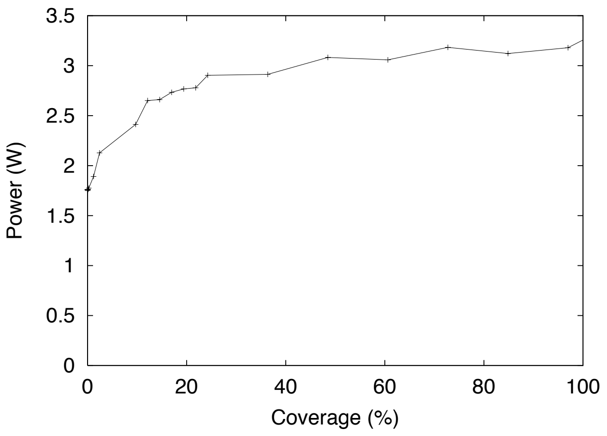
The reason we feel collocation of related data has greater potential than prefetching alone lies in the disk arm of modern hard drives. In addition to the spinning of the disk, another source of mechanical power consumption in a disk is the movement of the disk arm, which can be reduced by effective collocation of related data. In Figure 1(a) we see the two sources of mechanical movement in a disk drive: the disk arm, and the rotation of the disk platter(s). Both these processes consume electrical power, and are sources of mechanical delays. In particular, the movement of the disk arm to position the read-write head above the appropriate track is a significant component of disk access latency. While spinning down the disk may save a significant amount of power, it introduces potentially lengthy access delays by requiring the disk to be brought up to speed before data can be read or written. In contrast, reducing the required movement of the disk arm will reduce access latency, and may also save disk power. To verify the potential savings of successfully restricting disk arm movement (through predictive grouping or otherwise) we measured the power consumption of disk drives provided with a very specific workload designed to request random blocks across an increasing range of disk locations. Figure 1(b) shows the average power consumption (in Watts) of the mechanical drive mechanisms of a WD12100 disk drive under this synthetic workload. From this figure we can see an increase in average power consumption as the range of blocks (and subsequently average disk arm movement) increases. With practically no disk arm movement power consumption was observed at around 1.7 Watts. This value jumps to almost 3 Watts when the disk arm starts making movements that average approximately 27% to 30% of the maximum block range. These results would suggest that reducing disk arm movement may be one further-approach to reducing overall disk power consumption, by as much as 40% in this example, with none of the latency costs associated with disk spin-downs. This is especially encouraging when it is realized that collocating related data primarily results in a reduction of total I/O operations, which benefits both approaches to disk power reduction. Reduced random I/Os can reduce random disk arm movements, while simultaneously offering greater opportunities to completely spin-down the disk.

3. Experiments and Results

To test the potential benefits or costs of predictive disk management we ran a set of trace-based benchmarks designed to test the impact of varying degrees of success in predicting accesses and optimizing lay-



(a) *Disk Mechanical Movement*



(b) *Disk Power Consumption*

Figure 1. Reducing disk power consumption by reducing arm movements.

out. We found that such predictive management had the potential to match or exceed the power saving of traditional disk spin-down schemes, while avoiding the increased time delays associated with aggressive spin-down policies.

3.1. Measuring Detailed Energy Usage

For our experiments we used a collection of IDE hard-drives that ranged in capacity from 2GB to 80GB. We selected drives that had separate 12 and 5 Volt power lines (with the exception of the 20GB Fujitsu drive, that had a single 5 Volt power line). For the drives with separate power lines, we were able to isolate the energy usage for the drive mechanics from that of the drive electronics. Power was measured by sampling the voltage drop across a 0.1 Ohm resistor in series with the 12 Volt line, and a 0.05 Ohm resistor with the 5 Volt line. Samples were collected using a DAQ system collecting 20,000 samples per second for each experiment. The results presented below are from the sum of the mechanical components of the drive (as these are sensitive to changes in the layout policy, and affect the spin-down cost of the drive) as well as the electronics. Placing the electronics components in an idle or sleep mode does not have a similar added cost for returning to active mode, so for the electronics any power savings is mainly proportional to the reduction in the spin-down timeout for the drive. The high frequency of the measurement samples, and the ability to isolate drive motor power sources, allowed us to isolate the contribution of disk arm movement to a disk's energy usage as we illustrated in Figure 1(b). Below we present power and average service-delay figures for the following IDE drives: a 20GB Fujitsu and an 80GB Maxtor.

3.2. Reducing Delays and Energy Use

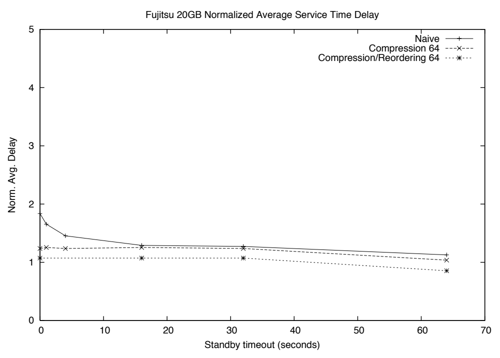
To measure the changes in average delay and energy usage of a drive we replayed a pre-recorded trace of disk-level read requests against a drive's raw interface. All operations were issued with relative timing kept as consistent as possible with the original trace. If an operation is not complete before the completion of the preceding request, it is issued as soon possible, but is delayed until it's original issue time if the preceding request had already completed. This approach allowed us to replay a consistent, yet realistic workload against drives with potentially varying performance characteristics, while remaining as faithful as possible to the original timing of the request stream. We believe this is superior to utilizing a synthetic benchmark that may not accurately reflect a realistic request stream, as they are notoriously difficult to synthesize with accuracy and believability [7]. For a test workload we used disk-level traces of workstation I/O requests from the DTB traces collected by [23], and we measured the effects of successfully applying predictive read-ahead and predictive data layout by modifying this workload appropriately.

With the assumption that you can accurately predict a significant amount of related data, prefetching allows you to issue a larger number of I/O operations in response to a single request, which in turn will hopefully avoid the need to fetch these related disk blocks later on. This would effectively mean that we have *compressed* a certain group of I/O requests in time, effectively issuing them in immediate succession, and hopefully producing a lengthier subsequent idle period. The intuition from doing this is that the latency for subsequent requests is now reduced, and the lengthier subsequent idle period increases the likelihood of a short spin-down timeout being beneficial. This behavior is the reason for our first modification to the workload, where we replay the trace, but issue a group of requests immediately whenever possible, within an interval of fixed size. These requests are issued at the start of interval N, and the subsequent group of requests are issued at the original time for the first request in interval N+1. Increasing the size of this group of time-compressed requests is equivalent to achieving increasing degrees of success in issuing read-ahead requests for data. In our figures, we refer to a workload modified with an interval of N seconds as "Compression N."

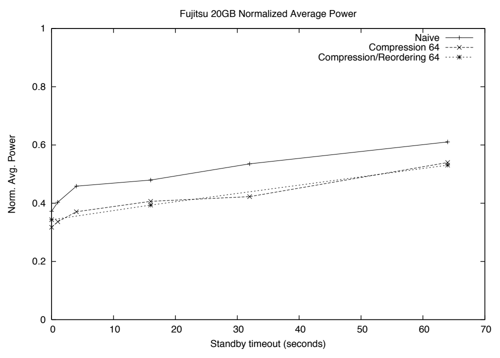
The second modification to the workloads reflects the effects of successfully optimizing a disk's layout in response to predicted file relationships. In this case requests are not simply time-compressed and issued in batches, but their relative locations on disk are updated to reduce disk arm movement. To model this behavior in a conservative manner, we modified the requests in an interval of size N by issuing them in bursts, with a further modification to place these block requests in order. This is the same effect as if data had been placed to reduce disk arm movement. For this form of modification, and a group of size N within an interval of N seconds, we refer to the workload in the figures below as "Compression/Reordering N."

Figures 2 and 3 show normalized delay and power figures for the three disks. The results were normalized against the average response and the energy usage of the same drive running continuously without a spin-down timeout. For delay, a value of 1 would indicate that there was no performance degradation for the workload when spin-downs are employed. Similarly for energy usage a value of 1 would indicate that the spin-downs have had no impact on energy usage for the given workload, but values greater than 1 would indicate an *increase* in energy consumption. Lower values are better for all figures.

In all these figures we see that increasing the spin-down timeout has a negative effect on energy consumption regardless of workload, while the predictive workloads exhibit marginally improved behavior beyond the simple timeout. In terms of energy consumption, all workloads across all drives benefited from a spin-down timeout, with shorter timeouts generally offering more energy savings. The modified workloads representing predictive layout (Compression with Reordering) tended to perform better than time-compression (successful prefetching alone).



(a) *Delay*



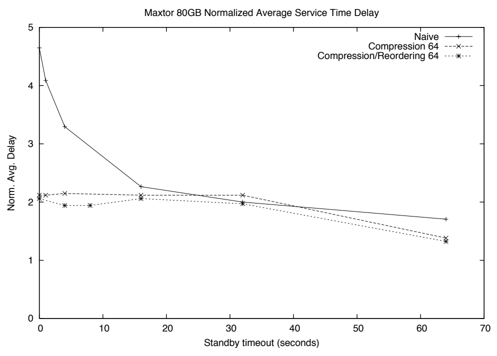
(b) *Power*

Figure 2. The effect of applying 64-operation compression and/or reordering of requests on power and delay for the Fujitsu 20GB drive.

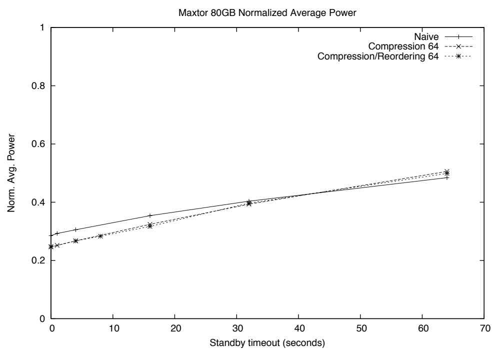
The real impact from successful predictive management can be seen when comparing the results for service delays using the most aggressive (shorter) spin-down timeouts. Without the benefits of increased burstiness due to predictive read-ahead or collocation, the naive time-out policy will degrade to over four times the average delay of a running disk. On the other hand, both modified workloads tended to remain close to a normalized average delay value of 1 (an ideal), while gaining all the energy-saving benefits of the shorter time-outs.

3.3. Practical Effectiveness

The results presented above focused on compression and reordering groups of size 64. This particular number was selected because it fell in the midrange of the observed results for all values tested. It is definitely possible that predictive groupings of data may be success-

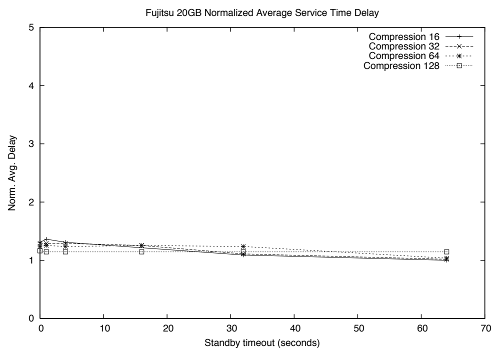


(a) *Delay*

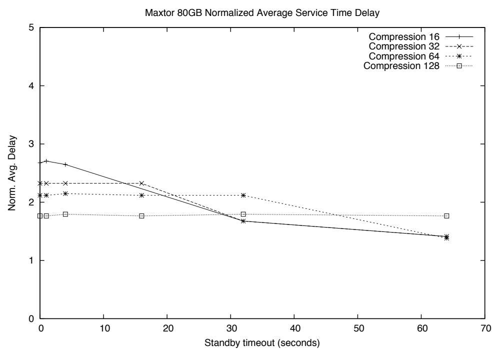


(b) *Power*

Figure 3. The effect of applying 64-operation compression and/or reordering of requests on power and delay for the Maxtor 80GB drive.



(a) *Fujitsu 20*



(b) *Maxtor 80*

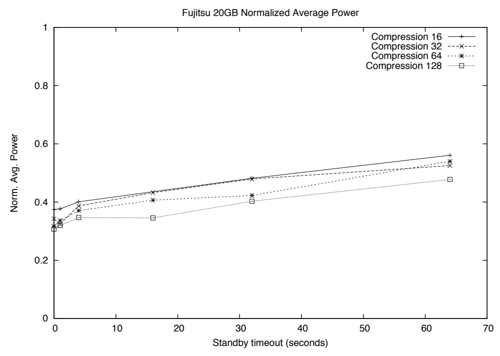
Figure 4. Normalized average service time delay for both drives, with request arrival time compression over increasing standby timeout. Data are normalized against the unmodified trace, with no standby timeout.

ful for larger or smaller groups of disk-level operations, and so we ran our tests for compression and reordering values ranging from 16 to 128, and while performance was best for the greatest success, the variation among the different values (which represent varying degrees of success for predictive optimizations) showed only minor variations in energy consumption or average delays. The behavior we observed above for groups of size 64 is seen across all values as we can see in Figure 4 for delays, and Figure 5 for power.

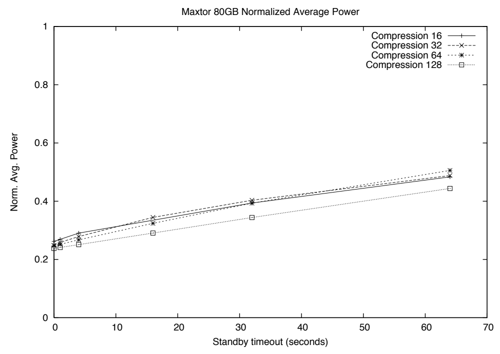
While it is often assumed that in a mobile environment, power savings can be achieved at the expense of disk responsiveness, this is not true for server environments. In such cases, the ability to save power without incurring degradation in storage responsiveness is a great advantage. To be able to reduce the energy consumption in such response-oriented environments would help reduce the direct and indirect energy costs associated with data centers and server farms. This is an increasingly critical issue, particularly with estimates of energy usage on the order of $2 billion in US data centers [3].

4. Related Work

Our work is influenced by the contributions of earlier researchers in several areas. The assumptions regarding predictive optimizations are based on prior art in storage optimization and access prediction, while our consideration of disk power reduction draws from earlier works in that area.



(a) *Fujitsu 20*



(b) *Maxtor 80*

Figure 5. Average power for both drives, with request arrival time compression, given an increasing standby timeout value. These values are normalized against the original trace, with no standby timeout enabled.

4.1. Optimized Data Grouping & Placement

Grouping has previously been applied for data placement. The earliest such works attempted used frequency-based estimates of access likelihood to optimize the placement of popular data. Attempts to optimally place files on disk were originally done manually, placing frequently accessed files closer to the center of the disk. The need to automate this process was addressed by the work of Staelin and Garcia-Molina [26, 27]. This work dealt with optimal placement, but offered models based on the assumption that file access events are independent. These approaches made no attempt to capture dynamic relationships between files. The Berkeley Fast File System (FFS) includes attempts to group related data, *e.g.* file data and metadata, into cylinder tracks on disk [21, 25]. Prior work by Akyürek and Salem replicated similar "hot" data blocks to a common area on disk to improve disk performance [1].

Dynamic groups attempt to exploit inter-file relationships, but require explicit application hints to determine group membership [28]. Earlier work on the automatic detection of working sets includes the work of [29]. Examples of automated file grouping include C-FFS (collocating FFS), which bases grouping on a directory-membership heuristic [8], and Hummingbird [24] which utilizes the underlying structure of web files.

4.2. Access Prediction

The use of data grouping for predictive caching has been proposed in the form of the *aggregating cache* [2]. The use of a last successor model for file prediction, and more elaborate techniques based on pattern matching, were first presented by Lei and Duchamp [18]. Later work compared the predictive performance of the last successor model to earlier graph-based schemes, and more effective schemes based on context modeling and data compression [16, 11, 17, 4].

4.3. Disk Power Management

Wilkes first suggested the use of predictive techniques to dynamically adjust disk spin-downs for improved power conservation [31]. Douglis, Krishnan, and Marsh demonstrated that perfect non-invasive spin-downs could reduce disk power consumption by up to 60%, while an on-line algorithm could reduce power consumption by 53% compared to the manufacturer's recommended five minute time-out [5]. Disk spin-down decisions were analytically modeled as a rent-to-buy problem by Krishnan *et al* [15]. The exploitation and prediction of disk idle periods was further investigated by later works [9, 10]. An adaptive scheme based on a machine learning algorithm achieved the greatest power savings of these techniques [12, 13]. This particular algorithm used was the "share" machine learning algorithm [14], a variant of the weighted majority voting algorithms [19], but as with all prior techniques it assumes that an I/O workload is not actively changed.

The use of prediction as a means to modify an I/O workload in the hopes of reducing power consumption has been proposed by prior works [6, 20]. These suggestions focused on the ability of prefetching data to allow for increased idle-time periods, which in turn would hopefully allow greater opportunities for spin-downs.

Similarly, more recent work attempts to actively modify the workload and increase workload burstiness to increase opportunities for disk spin-down [30, 22]. All these prior works share the common feature that they do not consider any technique other than disk spin-down as a mechanism for reducing power consumption. While effective, this approach adds latency costs when a disk needs to be brought back to an active state. The predictive approaches we've considered to reduce disk activity incur no such penalties, and can even reduce a disk's activity while it remains active (*e.g.*, using predictive grouping and placement to minimize disk arm movement).

5. Conclusions and Future Work

We have presented measurement results for power and service delays of trace-based workloads replayed against several modern disk drives. We've paid particular attention to the mechanical components of power consumption, but have presented results reflecting the sum of all power sources to the drives. While we have not tested a particular predictive algorithm, we have considered the effects of algorithms as applied for placement or predictive read-ahead/prefetching with varying degrees of success. We achieved this by modeling the effects of such policies in modifying a workload, without assuming any specific algorithm. We found that for a wide range of success rates, it is possible to reap significant benefits in meeting the contradictory goals of reducing disk delays while simultaneously improving energy conservation in modern disk drives. Future work includes combining this effort with specific predictive algorithms. It also includes testing the effects of false predictions, as well as comparing against a wider set of workloads and classes of disks.

6. Acknowledgments

This research was supported in part by the National Science Foundation under award ANI-0325353. We would like to acknowledge Darrell D. E. Long and our colleagues in the Storage Systems Research Center (SSRC) at the University of California, Santa Cruz. We are also indebted to our colleagues in the University of Pittsburgh's Power-Aware Real-Time Systems (PARTS) group for their continuous support and assistance.

References

[1] S. Akyürek and K. Salem. Adaptive block rearrangement. *ACM Transactions on Computer Systems*, 13(2):89–121, May 1995.

[2] A. Amer, D. D. E. Long, and R. C. Burns. Group–Based Management of Distributed File Caches. In *Proceedings of the 22nd International Conference on Distributed Computing Systems (ICDCS '02)*, Vienna, Austria, July 2002. IEEE.

[3] J. S. Chase, D. C. Anderson, P. N. Thakar, A. Vahdat, and R. P. Doyle. Managing energy and server resources in hosting centres. In *Proceedings of the 18th ACM Symposium on Operating Systems Principles (SOSP '01)*, pages 103–116, Banff, Canada, Oct. 2001.

[4] K. M. Curewitz, P. Krishnan, and J. S. Vitter. Practical prefetching via data compression. In *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data (SIGMOD '93)*, pages 257–266, Washington, D. C., May 1993.

[5] F. Douglis, P. Krishnan, and B. Marsh. Thwarting the power-hungry disk. In *Proceedings of 1994 Winter USENIX Conference*, pages 293–306, Boston, MA, Jan. 1994.

[6] J. Flinn and M. Satyanarayanan. Energy-aware adaptation for mobile applications. In *Symposium on Operating Systems Principles (SOSP99)*, pages 48–63, 1999.

[7] G. R. Ganger. Generating representative synthetic traces: An unsolved problem. In *Proceedings of the 21st International Conference for the Resource Management and Performance and Performance Evaluation of Enterprise Computing Systems (CMG95)*, pages 1263–1269, Dec. 1995.

[8] G. R. Ganger and M. F. Kaashoek. Embedded inodes and explicit grouping: Exploiting disk bandwidth for small files. In *Proceedings of the 1997 USENIX Annual Technical Conference*, pages 1–17, Anaheim, CA, Jan. 1997.

[9] R. Golding, P. Bosch, C. Staelin, T. Sullivan, and J. Wilkes. Idleness is not sloth. In *Proceedings of the 1995 Usenix Technical Conference*, pages 201–12, New Orleans, LA, USA, Jan. 1995.

[10] R. Golding, P. Bosch, and J. Wilkes. Idleness is not sloth. Technical Report HPL-96-140, Hewlett-Packard Laboratories, Palo Alto, CA, 1996.

[11] J. Griffioen and R. Appleton. Reducing file system latency using a predictive approach. In *USENIX Summer Technical Conference*, pages 197–207, June 1994.

[12] D. P. Helmbold, D. D. Long, and B. Sherrod. A dynamic disk spin-down technique for mobile computing. In *Proceedings of the Second Annual ACM International Conference on Mobile Computing and Networking*. ACM/IEEE, Nov. 1996.

[13] D. P. Helmbold, D. D. E. Long, T. L. Sconyers, and B. Sherrod. Adaptive disk spin-down for mobile computers. *ACM/Baltzer Mobile Networks and Applications (MONET)*, 5(4), 2000.

[14] M. Herbster and M. K. Warmuth. Tracking the best expert. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 286–94, Tahoe City, CA, 1995. Morgan Kaufmann.

[15] P. Krishnan, P. Long, and J. S. Vitter. Adaptive disk spin-down via optimal rent-to-buy in probabilistic environments. In *Proceedings of the Twelfth International*

Conference on Machine Learning (ML95), pages 322–330, Tahoe City, CA, July 1995. Morgan Kaufman.

[16] T. M. Kroeger and D. D. E. Long. The case for efficient file access pattern modeling. In *Proceedings of the Seventh Workshop on Hot Topics in Operating Systems (HotOS-VII)*, pages 14–9, Rio Rico, Arizona, Mar. 1999. IEEE.

[17] T. M. Kroeger and D. D. E. Long. Design and implementation of a predictive file prefetching algorithm. In *Proceedings of the 2001 USENIX Annual Technical Conference*, Boston, MA, June 2001.

[18] H. Lei and D. Duchamp. An analytical approach to file prefetching. In *Proceedings of the 1997 USENIX Annual Technical Conference*, pages 275–88, Anaheim, CA, Jan. 1997.

[19] N. Littlestone and M. K. Warmuth. The weighted majority algorithm. *Information and Computation*, 108(2):212–61, 1994.

[20] J. R. Lorch and A. J. Smith. Software strategies for portable computer energy management. *IEEE Personal Communications*, 5(3), June 1998.

[21] M. K. McKusick, W. N. Joy, S. J. Leffler, and R. S. Fabry. A fast file system for UNIX. *ACM Transactions on Computer Systems*, 2(3):181–97, Aug. 1984.

[22] A. E. Papathanasiou and M. L. Scott. Increasing file system burstiness for energy efficiency, Dec. 2002.

[23] A. Peacock. Dynamic detection of deterministic disk access patterns. Master's thesis, Department of Computer Science, Brigham Young University, Provo, Ut., Apr. 2001.

[24] E. Shriver, E. Gabber, L. Huang, and C. Stein. Storage management for web proxies. In *Proceedings of the 2001 USENIX Annual Technical Conference*, pages 203–16, Boston, MA, June 2001.

[25] K. A. Smith and M. Seltzer. A comparison of FFS disk allocation policies. In *Proceedings of the 1996 USENIX Technical Conference*, pages 15–25, San Diego, CA, Jan. 1996.

[26] C. Staelin and H. Garcia-Molina. File system design using large memories. In *Proceedings of the Fifth Jerusalem Conference on Information Technology (JCIT)*, pages 11–21. IEEE, Oct. 1990.

[27] C. Staelin and H. Garcia-Molina. Smart filesystems. In *Proceedings of the Winter 1991 USENIX conference*, pages 45–51, Jan. 1991.

[28] D. C. Steere. *Using Dynamic Sets to Reduce the Aggregate Latency of Data Access*. PhD thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, Jan. 1997.

[29] C. D. Tait and D. Duchamp. Detection and exploitation of file working sets. Technical Report CUCS-050-90, Computer Science Department, Columbia University, New York, NY 10027, 1990.

[30] A. Weissel, B. Beutel, and F. Bellosa. Cooperative I/O: A novel I/O semantics for energy-aware applications. In *Proceedings of the 2002 Symposium on Operating Systems Design and Implementation (OSDI)*, Boston, MA, Dec. 2002.

[31] J. Wilkes. Predictive power conservation. Technical Report HPL-CSP-92-5, Concurrent Systems Project, Hewlett-Packard Laboratories, Palo Alto, CA, Feb. 1992.