Network-Aware Program-Counter-Based Disk Energy Management

Igor Crk and Chris Gniady

Abstract Reducing energy consumption is critical to prolonging the battery life in portable devices. With rising energy costs and increases in energy consumption by devices in stationary systems energy expenses of large corporations can annually reach into millions of dollars. Subsequently, energy management has also become important for desktop machines in large scale organizations. While energy management techniques for portable devices can be utilized in stationary systems, they do not consider network resources readily available to stationary workstations. We propose a network-aware energy management mechanism that provides a low-cost solution that can significantly reduce energy consumption in the entire system while maintaining responsiveness of local interactive workloads. The key component of the system is a novel program-context-based bandwidth predictor that accurately predicts application's bandwidth demand for file server/client interaction. Our dynamic mechanisms reduce the decision delay before the disk is spun-up, reduce the number of erroneous spin-ups in local workstations, decrease the network bandwidth, and reduce the energy consumption of individual drives.

1 Introduction

Dynamic energy reduction techniques have been extensively studied with the aim of extending battery life and prolonging the operation of portable devices. Considering that end-systems in large-scale enterprise environments are already dependent, to some degree, on centralized storage, we see an opportunity for reducing energy consumption in both portable and desktop systems that rely on this resource. We propose a bandwidth demand predictor and design an end-system energy manage-

Igor Crk
University of Arizona, Tucson AZ 85721 e-mail: icrk@cs.arizona.edu

Chris Gniady
University of Arizona, Tucson AZ 85721 e-mail: gniady@cs.arizona.edu

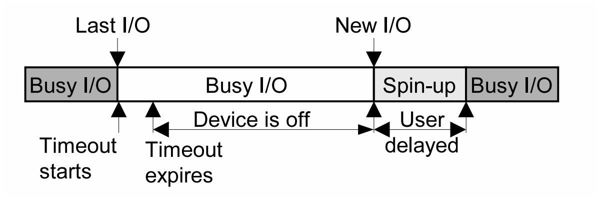


Fig. 1 Anatomy of an idle period.

ment mechanism that can be successfully used within large enterprise networks. The techniques we develop here are applicable to many system components, but our focus is on the hard drive, since it is responsible for a significant portion of overall system energy. The critical issues are keeping the hard drive spinning only when necessary, relying on the network for common data access, while at the same time avoiding network congestion.

We propose the Program Context Bandwidth Predictor (PCBP) for dynamic I/O bandwidth prediction that meets this need. The PCBP idea is supported by recent research regarding I/O context correlation [8, 9]. In this paper, we argue that there is a strong correlation between the application context and the I/O bandwidth demand generated in by the context. Furthermore, we exploit this correlation to accurately predict bandwidth demands from the application and design a network-aware power management system which dynamically adjusts the level of reliance on network storage based on bandwidth availability and predicted future bandwidth demand. The proposed mechanism is compared to the previously proposed prediction mechanism in Self-Tuning Power Management (STPM) [14]. Our results show that PCBP mechanisms achieve higher accuracy, better responsiveness, and more efficient network bandwidth utilization while improving energy efficiency.

2 Background

Manufacturers recommend spinning down the power hungry hard disks after some period of idleness [6, 11]. Fig. 1 shows the anatomy of an idle period. After the last request, a timer is started and the device is shut down once the timer expires unless there is additional disk activity occuring during the timeout period. The disk remains powered down until a new request arrives. However, accelerating the platters requires more energy than keeping them spinning while the disk is idle. Therefore, the time during which the device is off has to be long enough to offset the extra energy needed for the shutdown and spin-up sequence. This time is commonly referred to as the *breakeven time*, and is usually on the order of a few seconds. Eliminating shutdowns that not only waste energy but also significantly delay user requests is critical to conserving energy and reducing interactive delays.

Timeout-based mechanisms are simple to implement but waste energy while waiting for a timeout to expire. Consequently, dynamic predictors that shut down the device much earlier than the timeout mechanisms have been proposed to address the energy consumption of the timeout period [5, 12, 18]. Stochastic modeling techniques have also been applied to modeling the idle periods in applications and shut down the disk based on the resulting models [3, 4, 15, 17]. Additional heavy weight approaches have relied on application developers to insert hints about upcoming activity [7, 10, 13, 19] or have suggested compile-time optimizations for I/O reshaping and disk access synchronization.

To reduce the burden of hint insertion on the programmer, automatic generation of application hints was proposed in Program Counter Access Predictor (PCAP) [9]. PCAP exploits the observation that I/O activity is caused by unique call sites within the applications. Depending on user interactions with the application, I/O sequences are generated by different call sites. Therefore, the activity described by a call site can be correlated to an idle period that follows a sequence of I/O operations. The program context of the idle period provides more information to the predictor, resulting in high accuracy and good energy savings.

In addressing both spin-up delays and energy efficiency, BlueFS [14] proposes that data should be fetched from alternate sources to mask the disk latency. The recent use of flash memory in hybrid disk drives by Samsung [16] provides similar energy savings. BlueFS uses ghost hints [1, 2] to predict when the disk should be spun-up. Ghost hints are issued to the disk for each item fetched from the network, and after receiving a few such hints in a short time, the disk is spun-up and takes over serving requests. A ghost hint discloses the potential time and energy that would have been saved if the disk was in the ideal power mode, i.e. the platters are spinning and the disk is ready to serve data. A running total of benefits is maintained and increased every time a hint is received. The total is also decremented by the energy cost of remaining in idle mode since the last ghost hint was issued. The disk is spun-up when the total exceeds a threshold. Throughout the spin-up period, data is fetched from the network, making disk power management transparent to the user. Furthermore, requests with low I/O activity are fetched entirely from the server, avoiding disk spin-ups altogether. By eliminating disk spin-ups we can significantly reduce both the client's power consumption and the wear-and-tear of the disk's mechanical components.

However, BlueFS's ghost hinting mechanisms suffer from drawbacks similar to those of the timeout-based shutdown mechanisms. First, the delay of spinning up the disk can result in longer delays for high bandwidth I/O requests. Second, some requests that result in a disk spin-up according to ghost hinting mechanisms may be satisfied before the disk spins up, resulting in no disk activity following the spin-up. Therefore, a dynamic predictor that can exploit a large history of I/O events and provide immediate spin-up predictions has the potential to improve the accuracy and timeliness of predictions.

Authors of PCAP have shown that using call sites to describe the behavior of different parts of the application can lead to a very accurate shutdown predictor. Different parts of the application exhibit different I/O behavior, therefore uniquely correlating application parts to the resulting I/O behavior should improve spin-up predictions, as well. This paper shows that we can accurately correlate the I/O bandwidth demand to the call sites to improve disk spin-up predictions.

3 Design

The key idea behind PCBP is that *there is a strong correlation between the program context that initiated the sequence of I/O operations and the resulting I/O activity*. We exploit this correlation to control the disk activity of desktop systems in a large enterprise environment, under the assumption that network storage is available and a relatively small portion of available bandwidth can be apportioned to support I/O for local applications.

3.1 Program context correlation

We define a busy period as the I/O activity between two idle periods that are longer than the breakeven time. To uniquely describe the program context, PCBP calculates the signature of the call site that initiated the I/O by traversing the application call stack and recording the program counter of each function call on the stack [9]. PCBP collects and records a single signature for each busy period, minimizing the overhead of the system call execution. Once the signature is recorded, PCBP begins collecting statistics about the current busy period.

PCBP stores the number of bytes requested (*busy-bytes*) during the busy period and the length of the busy period (*busy-time*) with each signature. PCBP updates and maintains the *busy-bytes* and *busy-time* variables per application, as shown in Fig. 2. Each time a new I/O request arrives, PCBP calculates the length of the idle period between the completion of the last I/O request and the beginning of the current one.

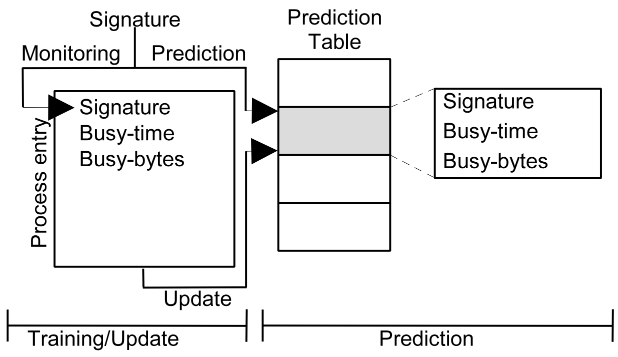


Fig. 2 Prediction and training structures.

If the idle period is shorter than the breakeven time, it is considered part of the busy period. An idle period longer than the breakeven time indicates the beginning of a new busy period.

The prediction table is organized as a hash table indexed by the call site signature. The table is shared among processes to allow table reuse across multiple executions of the same application as well as concurrent executions of the same application. Table reuse reduces training for future invocations of an application, and can be easily retained in the kernel across multiple executions of the application due to its small size. Table reuse was successfully exploited in previous program context based predictors [9, 8]. Additionally, the table can be stored locally or remotely in order to eliminate training following a reboot.

3.2 Disk spin-up prediction

PCBP predicts I/O activity that will follow a call site preceding the first I/O of a busy period. Continuing to monitor I/O activity during the busy period ensures proper handling of mispredictions. PCBP's predictions of upcoming I/O traffic and busy period duration are used to determine whether the disk should be spun up or the file server should satisfy the request instead. The file server is assumed to contain all user files and application files and sufficient memory to cache them.

To spin up, PCBP first considers the possibility of bursty traffic. In this case, if a request can be satisfied from the file server in the time it takes to spin up the disk in the local workstation, the disk remains down. In making the decision, PCBP considers the predicted length of the busy period. If the predicted length is shorter than the time it takes to spin up the disk, PCBP considers the amount of I/O traffic that is to be fetched during the busy period. If the I/O requests can be satisfied from the server, given a particular server load, in the time it would take to spin the disk up, the disk is not spun up and the requests are serviced by the file server. Otherwise, PCBP spins the disk up immediately. While the disk is spinning up, I/O requests are serviced by the file server. Once the disk is ready, it takes over and services the remaining requests.

The second step in spin-up prediction considers steady traffic with low bandwidth demand. In this case, a low bandwidth prediction in the primary PCBP predictor supresses the issuing of ghost hints, leaving the backup mechanism to make predictions only when no primary PCBP prediction is available. A spin-up occuring for low-bandwidth activity periods is considered a miss, since the requested data can be served by the server with little to no performance impact. A beneficial spin-up reduces the delay associated with serving data from the server when the available bandwidth is saturated. To prevent excessive reliance on network storage during training and mispredictions, PCBP uses the BlueFS ghost hinting mechanism as a backup predictor.

PCBP mechanisms use the same criteria for a spin-up decision in the case of both a single and multiple processes issuing I/O requests. A prediction for each

Table 1 Applications and execution details

Applications	Number of Executions	Global Busy Periods	Local Busy Periods	Total Signatures
mozilla	30	361	997	222
mplayer	10	10	10	1
impress	29	180	632	110
writer	29	137	415	61
calc	26	150	591	67
xemacs	31	100	127	14

process issuing I/O requests is sent to the Global Spin-up Coordinator (GSC), which considers the combination of all currently predicted I/O demands for each process in deciding whether to spin the disk up. As long as the sum of I/O activity arriving from processes can be satisfied by the given file server bandwidth, the disk remains in the off state. If GSC detects that the bandwidth demand will most likely drop below the available bandwidth in the time it would take to spin the disk up, the disk is not spun up. In addition, GSC monitors the traffic from all applications and compares it to the total predicted traffic. If there is no prediction available, such as during training, or if following a prediction to solely utilize the network for upcoming requests the amount of I/O observed exceeds the amount of I/O that was predicted, the backup predictor is activated.

PCBP relies on BlueFS mechanisms to synchronize the local file system to the file server. User generated data is transferred to the file server periodically, allowing for extended buffering in memory. In the case of machine failure, user data on the local machine can be restored to its correct state from the server. Further, the mirroring of user and application data from all workstations does not cause a major storage overhead, due to the relative homogeneity of enterprise computing environments. As shown by the BlueFS system study [14], the amount of user data that has to be synchronized with the server is quite small and should not result in excessive communication overheads or energy consumption.

4 Methodology

We evaluate the performance of PCBP and compare it to ghost hinting mechanisms in BlueFS using a trace-based simulator. Detailed traces of user-interactive sessions for each of a set of applications were obtained by a modified `strace` utility over a number of days. The modified `strace` utility allows us to obtain the PC, PID, access type, time, file descriptor, file name, and amount of data that is fetched for each I/O operation. The applications themselves are popular interactive productivity and entertainment applications familiar to Linux users.

Table 1 shows the details of the chosen applications' traces. As previously discussed, global busy periods are composed of one or more local busy periods. Per-process predictions are performed using the PC signatures at the beginning of each

Table 2 Energy consumption specifications, **a** for the WD2500JD hard disk drive (left), and **b** the CISCO 350 wireless network card (right)

State	Power
R/W	10.6W
Seek	13.25W
Idle	10W
Standby	1.7W
Transition	**Energy**
Spin-up	148.5J
Shutdown	6.4J
Transition	**Delay**
Spin-up	9 sec.
Shutdown	4 sec.
Breakeven	19 sec.

State	Power
Low	.39W
High	1.41W
Transition	**Energy**
Ready	.510J
Shutdown	.530J
Transition	**Delay**
Ready	.40 sec.
Shutdown	.41 sec.

local busy period. GSC uses these predictions to generate a prediction for each global busy period.

Energy consumption and savings are calculated based on application behavior and the amount of time spent in a particular state. Table 2 shows the energy consumption profiles of a Western Digital Caviar 250.0GB Internal Hard Drive Model WD2500JD. and the Cisco 350 Network Interface Card.

We implemented the ghost hinting mechanisms according to the description in [14, 2], placing equal weight on response time and energy efficiency. A simple timeout-based shutdown predictor set to breakeven time provides the common basis for comparison of accuracy of the two mechanisms and does not introduce another dimension for evaluation.

The evaluation assumes that the available network bandwidth to the server is constrained to one thousandth of the total bandwidth of a 1 Gb network, chosen experimentally as the point after which further constraints on bandwidth availability don't result in significant energy savings for the local disk. An overabundance of available network bandwidth would keep the disk mostly in the off state, so by constricting the available bandwidth we illustrate that energy savings are possible when the available bandwidth is low.

5 Evaluation

5.1 Prediction Accuracy

Figure 3 compares the ability of Ghost Hints (GH) and PCBP to correctly predict the I/O activity during a busy period. The GH mechanism consists of ghost hints alone, therefore Fig. 3 shows only hits and misses for GH. PCBP mechanisms use ghost hints as a backup predictor during training and mispredictions. We define a hit as

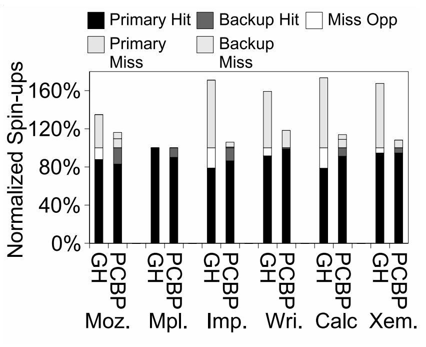


Fig. 3 Comparison spin-up predictor accuracies, normalized to the number of spin-ups

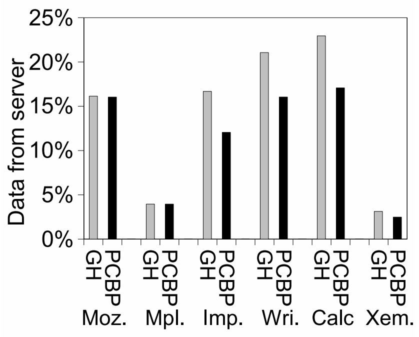


Fig. 4 Percent of data served by the server

a disk spin-up followed by bandwidth demand that is higher than what is available from the server. Spinning up the disk wastes energy when few or no I/O requests are served following spin-up, hence these events are defined as misses.

We find that on average PCBP mechanisms make 17% more correct disk spin-ups than GH, with an average of 76% fewer misses than GH. The primary predictor is responsible for a majority of the coverage, attaining on average 88% of the hits. To avoid a possibility of inheriting the large misprediction rate of GH, we use a 2-bit saturating counter as a confidence estimator to guide PCBP in using the backup predictor. The resulting misprediction rate of the backup predictor is on average 18%.

5.2 Server and Disk Utilization

Figure 4 shows the fraction of data served from the server for GH and PCBP. It is important to emphasize that the reliance on a network server for user data should not adversely affect the operation of the local system nor other systems accessing network resources. We observe that the amount of data fetched from the disk was greater when using PCBP mechanisms for all applications, excluding *Mplayer*. Conversely, we observe that using PCBP mechanisms, the load on the server is on average 3% less than with GH. We can conclude that using PCBP mechanisms, the server can either increase the number of machines it can support or provide a higher quality service to the same number of machines.

Figure 3 shows that PCBP mechanisms initiate fewer spin-ups while serving more data from the disk, as shown in Fig. 4. This implies that PCBP mechanisms are more efficient in serving data from the disk as shown in Fig. 5. Figure 5 shows the average amount of data served from the local drive with PCBP and GH per beneficial disk spin-up. On average, PCBP mechanisms result in serving 7% more data than GH following each disk spin-up. Higher efficiency in PCBP mechanisms re-

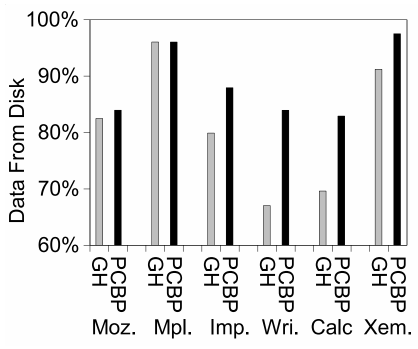


Fig. 5 Average amount of data fetched per beneficial disk spin-up

Applications	GH Ave. Delay	PCBP Ave. Delay
mozilla	10.46s	5.42s
mplayer	1.37s	0.01s
impress	6.17s	4.60s
writer	3.90s	2.26s
calc	5.48s	4.78s
xemacs	2.42s	1.15s

Fig. 6 Average delay in seconds before the disk has begun spinning up

sults from the PCBP spinning up the disk sooner than GH, therefore serving more data from the disk in a particular busy period. In all applications, the disk spin-ups were consistently more beneficial when using PCBP mechanisms.

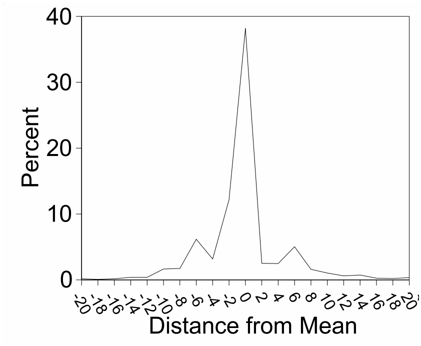
5.3 *Response Time*

Table 6 details the average delay time before the decision to spin the disk up is made by GH and PCBP. PCBP's timely prediction and infrequent reliance on the backup predictor ensure that the PCBP spin-up delay time is on average 47% shorter than that of GH mechanisms.

5.4 *Predictability of Process Activity*

The distribution of I/O activity for all collected signatures is shown in Fig. 7 as the distance from the mean. A distance of one represents the range of values equivalent to one twentieth of the average expected I/O activity for all traced applications. Each increment is equivalent to 12kB. We observe that 40% of all activity equals the mean for a signature each time that signature is observed. Further, more than 80% of activity is less than 120kB from the mean, while the average local-busy activity for all traced processes is 240kB. Irregularities in the distribution are due to the nature of interactive applications, for example, one signature may be used to record the activity associated with accessing multiple user-specified files of significantly different sizes.

Figure 8 shows that the aggregate time elapsed between requests in a single busy period of a process shows small variability, as did Fig. 7 for the number of bytes fetched. Distances from the mean are computed at microsecond accuracy. In this



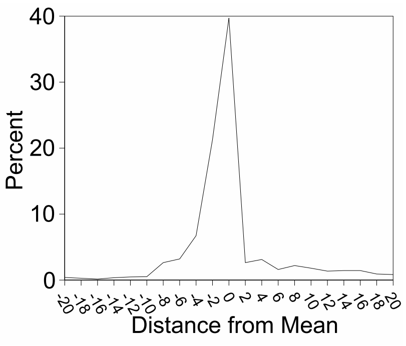


Fig. 7 Distribution of bytes fetched during I/O activity

Fig. 8 Distribution of I/O activity by duration

case, we find that over 85% of measured aggregate times fall within 10ms of the mean for a particular signature. The predictability of bytes fetched during I/O activity and its elapsed time allows us to make accurate decisions about future activity and its bandwidth requirements.

5.5 Energy Savings

Figure 9 compares the energy-delay products of GH and PCBP, normalized to the energy-delay of the on-demand spin-up mechanism. All predictors evaluated in this figure shut down the disk after the breakeven time has passed since the last request has been served. The energy consumption is calculated from a combination of power-cycle energy, consumed when the disk spins up or shuts down, idle energy, consumed when the disk is spinning idly, and active energy, consumed when the disk is actively seeking, reading or writing.

Overall, PCBP shows an average of 40% improvement in energy-delay product over the demand-based mechanism, due to its overall 52% improvement in energy consumption and timeliness of spin-ups in instances where bandwidth demand exceeds the allowable network limit. GH performs reasonably well with an average 30% improvement in energy-delay product over the demand-based mechanism. This corresponds to the observations seen in Figs. 4 and 5.

Figure 10 presents energy savings for one network-attached machine given the constrained available server bandwidth. We use a standard workstation as a file server with a measured peak bandwidth of 42 MB/s. This is shown in Fig. 10 as the maximum bandwidth available in our experments. When the available bandwidth is systematically restricted, a drop in energy savings occurs at 150 KB/s for most applications, suggesting that at this point the I/O generated by the attached machine has begun to saturate the allocated bandwidth.

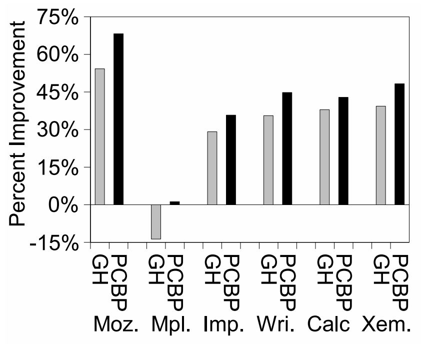


Fig. 9 Hard drive energy-delay product improvement normalized to demand-based disk spin-up

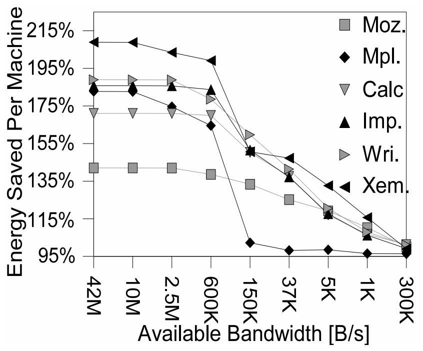


Fig. 10 Percent of energy saved per network-attached machine as available network bandwidth decreases, normalized to the demand-based predictor

6 Conclusion

Energy consumption plays a significant role not only in portable devices but also in large enterprises due to rising energy costs and increases in energy consumption by stationary systems. In this paper, we proposed a novel program-context-bandwidth predictor and utilize the predictor in designing energy management mechanisms for large scale enterprise environments.

The paper presented evidence of a strong correlation between program context and an application's bandwidth demand. The proposed PCBP mechanisms accurately correlate bandwidth demand during busy periods with the program context that initiated the busy period, and thus have the potential to make much more accurate predictions than previous schemes. Compared to ghost hinting mechanisms that predict I/O behavior based only on most recent history, the PCBP mechanism offers several advantages: (1) it can accurately predict the I/O activity of the new busy period during the first request that initiated the busy period; (2) it spins up the disk immediately upon predicting the high activity busy period, eliminating the prediction delays due to recent history collection; (3) it significantly reduces number of unnecessary spin-ups reducing impact of energy management on disk reliability.

Our evaluations using desktop applications commonly encountered in enterprise environments show that compared to ghost hinting mechanisms in BlueFS, PCBP mechanisms reduce unnecessary disk spin-ups on average by 76%. The predicted spin-ups are more efficient by serving on average 6.8% more data per spin-up than ghost hints. As a result, PCBP mechanisms improve the energy-delay product on average by an additional 10% for an overall improvement of 40% from the popular timeout-based mechanism. With respect to energy consumption, PCBP is only 4% from a perfect spin-up predictor.

PCBP can be combined with PCAP [9] to fully integrate context based shutdown and spin-up prediction. Furthermore, it is possible to use PCBP to predict shutdown due to accurate prediction of user activity during the busy periods. We will investigate those possibilities in our future work.

References

1. Anand, M., Nightingale, E.B., Flinn, J.: Self-tuning wireless network power management. In: Proceedings of the 9th Annual International Conference on Mobile computing and Networking, pp. 176–189. ACM Press, New York, NY, USA (2003). DOI http://doi.acm.org/10.1145/938985.939004
2. Anand, M., Nightingale, E.B., Flinn, J.: Ghosts in the machine: Interfaces for better power management. In: Proceedings of the Second International Conference on Mobile Systems, Applications, and Services (MOBISYS'04) (2004). URL http://www.eecs.umich.edu/ anandm/mobisys.pdf
3. Benini, L., Bogliolo, A., Paleologo, G.A., Micheli, G.D.: Policy optimization for dynamic power management. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems **18**(6), 813–833 (1999)
4. Chung, E.Y., Benini, L., Bogliolo, A., Lu, Y.H., Micheli, G.D.: Dynamic power management for nonstationary service sequests. IEEE Transactions on Computers **51**(11), 1345–1361 (2002)
5. Chung, E.Y., Benini, L., Micheli, G.D.: Dynamic power management using adaptive learning tree. In: Proceedings of the International Conference on Computer-Aided Design, pp. 274–279 (1999)
6. Dell Computer Corp.: Dell System 320SLi User's Guide (1992)
7. Ellis, C.S.: The case for higher-level power management. In: Workshop on Hot Topics in Operating Systems, pp. 162–167. Rio Rico, AZ, USA (1999)
8. Gniady, C., Butt, A.R., Hu, Y.C.: Program counter based pattern classification in buffer caching. In: Proceedings of the 6th Symposium on Operating Systems Design and Implementation (2004)
9. Gniady, C., Hu, Y.C., , Lu, Y.H.: Program counter based techniques for dynamic power management. In: Proceedings of the 10th International Symposium on High Performance Computer Architecture (HPCA) (2004)
10. Heath, T., Pinheiro, E., Hom, J., Kremer, U., Bianchini, R.: Application transformations for energy and performance-aware device management. In: Proceedings of the 11th International Conference on Parallel Architectures and Compilation Techniques (2002)
11. Hewlett-Packard: Kittyhawk power management modes. Internal document (1993)
12. Hwang, C.H., Wu, A.C.: A predictive system shutdown method for energy saving of event driven computation. ACM Transactions on Design Automation of Electronic Systems **5**(2), 226–241 (2000)
13. Lu, Y.H., Micheli, G.D., Benini, L.: Requester-aware power reduction. In: Proceedings of the International Symposium on System Synthesis, pp. 18–24 (2000)
14. Nightingale, E.B., Flinn, J.: Energy efficiency and storage flexibility in the blue file system. In: Proceedings of the 6th Symposium on Operating Systems Design and Implementation (OSDI) (2004)

15. Qiu, Q., Pedram, M.: Dynamic power management based on continuous-time markov decision processes. In: Proceedings of the Design Automation Conference, pp. 555–561. New Orleans, LA, USA (1999)
16. Samsung: Samsung Teams with Microsoft to Develop First Hybrid Hard Drive with NAND Flash Memory (2005)
17. Simunic, T., Benini, L., Glynn, P., Micheli, G.D.: Dynamic power management for portable systems. In: Proceedings of the International Conference on Mobile Computing and Networking, pp. 11–19 (2000)
18. Srivastava, M.B., Chandrakasan, A.P., Brodersen, R.W.: Predictive system shutdown and other architecture techniques for energy efficient programmable computation. IEEE Transactions on VLSI Systems **4**(1), 42–55 (1996)
19. Weissel, A., Beutel, B., Bellosa, F.: Cooperative I/O—a novel I/O semantics for energy-aware applications. In: Proceedings of the Fifth Symposium on Operating System Design and Implementation (2002)