Saving power without compromising disk drive reliability

Xenia Mountrouidou
College of William and Mary
Williamsburg, VA 23187, USA
Email: xmount@cs.wm.edu

Alma Riska
EMC Corporation
Cambridge, MA 02140, USA
Email: alma.riska@emc.com

Evgenia Smirni
College of William and Mary
Williamsburg, VA 23187, USA
Email: esmirni@cs.wm.edu

***Abstract*—We present a robust framework that aims at harvesting future idle intervals for power savings within strict constraints: first, it is imperative to contain the delays in service of IO requests that occur during power savings since the time to bring up the disk is not negligible and second, ensure that the power saving mechanism is triggered few times only, such that the disk wear out due to powering up and down does not compromise its lifetime. Extensive experimentation on a set of enterprise storage traces illustrates frameworks effectiveness.**

***Keywords*- disk drive wear out; power saving; performance guarantee; histogram; idleness**

I. Introduction

Storage systems in data centers host thousands of disk drives which are not all accessed simultaneously. As a result, one compelling approach for reducing power consumption in data centers is to spin down idle disk drives. This approach is routinely deployed in storage systems that serve as archival or backup systems [1] and is being exploited even in high-end computing environments [2].

However, spinning down disk drives to save energy in a high-end environment *transparently* to the end user and *reliably* to the disk drive's lifetime is a challenging open problem for a host of reasons. First, in interactive environments, requests that arrive while the drive is in a power saving mode may be delayed during recovery time, i.e., the period before the disk drive becomes physically ready to serve jobs again. Second, often idle times can be highly fragmented, i.e., while the overall drive utilization is low, idle periods that are long enough to be used effectively for power savings may be very few [3]. Third, every time a disk is powered up/down, the lifetime of the disk drive deteriorates, therefore there are strict limitations on the number of times that the disk is put in power savings mode to preserve disk reliability.

Provided that future disk requests are unknown, we address the above challenges by presenting a framework that uses as input user- or system-level constraints such as the number of allowable power ups/downs of a disk within a time period (strict constraint) and the user acceptable potential performance degradation of future IOs (soft constraint). While abiding to these constraints the framework provides a strategy on how to schedule power savings and estimates accurately the potential savings.

There is a wealth of literature focusing on conserving power in disk drives. Hibernator [4] is a framework that addresses power savings in a storage system setting, by redirecting workload to active disks that are dynamically deployed with different rotation speeds while meeting performance goals. This approach to save energy in a cluster does not consider the wear and tear of disks due to spin downs. WRITE offloading [2] extends idleness in a disk drive by offloading the WRITE traffic elsewhere in the storage system. Similarly, SRCMap [5] is a workload shaping technique that uses an energy vs. workload intensity proportionality model to determine which disks in the system will be used for power savings and which to serve IO. Both [2] and [5] use fixed idle waiting periods (in the order of minutes) to limit performance degradation, albeit no guarantees are given on performance degradation because of power savings. In [6] the authors formulate an optimization problem to minimize power consumption and reliability costs in data centers while

limiting the response time degradation to a target value. Different from all the above works, the framework proposed in this paper extends the idea in [3] by providing scheduling for power savings, i.e., "when" and for "how long" to power down a disk, with *both* performance and reliability user *predefined* guarantees.

This paper is organized as follows. Section II summarizes the power savings opportunities in storage systems. In Section III, we present the methodology proposed to identify and estimate the power savings opportunities in a system under a given workload. We validate the effectiveness of the approach in Section IV. Conclusions are given in Section V.

II. Power Saving Modes in Disk Drives

There are several levels of power consumption depending on the state of the disk's mechanical and electronic components. Each power consumption level or mode is characterized by the amount of *power* it consumes and the amount of *time* it takes to get out of the power saving mode and become ready to serve IOs. Table I presents a coarse description of the possible power saving modes focusing on the components that are slowed down or shut off, and the penalties associated with each power saving mode. The reported penalty values are within representative ranges published by two disk drive manufacturers [7], [8].

Operation Mode	Description	Power savings	Penalty (sec)
Level 1	Serving IOs	0%	0.0
Level 2	Active (but) idle	40%	0.0
Level 3	Unloaded heads	48%	0.5
Level 4	Slowed platters	60%	1
Level 5	Stopped platters	70%	8
Level 6	Shut down	95%	25

Table I
Characteristics of power saving modes.

The time it takes a disk to become active following a power saving mode obviates the need to account for the performance penalty before deciding on a disk operation mode for power savings. In storage systems it is common to not put the system automatically in a power saving mode when an idle interval is observed. Instead it waits for a time period in anticipation of future IO arrivals. This idle waiting time guides the system to use idle intervals that are sufficiently large (i.e., longer than the reactivation time) for power savings.

In addition to the performance penalty, there is a reliability penalty. The latter is not straight forward to quantify, because it is associated with the wear out of the disk drives during power ups (spin-up) or re-activation of individual components. While spin-ups have been analyzed for years as part of the disk drive wear out process, the partial shut down of disk drives is more recent (introduced solely for the purpose of power savings) and its impact on the disk wear out is not as well quantified and documented. It is known that a disk drive can survive above 40,000 spin-ups [9]. It is also expected that a disk drive can tolerate more partial spin-ups than full spin-ups.

III. Algorithmic Framework

Here we develop an algorithmic framework that determines the schedule of the periods when a disk drive is placed in power saving modes, such that predefined targets for system quality metrics are met.

The set of input parameters is

- *performance target* D which is guaranteed by the framework. This is a metric of quality that indicates that response time of IOs can degrade by at most $D\%$ in presence of power savings. This parameter changes dynamically over time of the day, different workloads and/or varying system requirements.

- *penalty* P associated with the time it takes to re-activate a disk drive in power saving mode. This penalty depends on the level of power saving, as seen in Table I. This parameter is fixed and does not change.

- *number of re-activations* X that do not affect disk's lifetime. This parameter is fixed and does not change.

In addition to the input parameters above, our framework uses a set of metrics and data structures that are monitored in the system. These are:

- *continuous data histogram (CDH)* of idle times observed in the system. The CDH is a list of tuples (at most a few thousands of them). Each tuple contains a range of idle interval lengths and their corresponding empirical cumulative probability.

- *average response time* RT of IOs unaffected by the power saving modes.

Updating these metrics and data structures over-time ensures that our framework is adaptive and reflects changes in the workload.

A. Schedule of Power Saving Modes

In our framework, the power saving modes are assumed to be *low priority work* at the disk waiting to be served. The amount of such work is assumed to be "infinite", which means that the disk needs to serve as much as possible of this work. With these assumptions, the problem of scheduling power saving modes in disk drives, is formalized as a problem of scheduling two workloads with different priorities. The power saving modes, i.e., the low priority workload, should be scheduled during the idle intervals of the user workload, i.e., the high priority work.

Because there is a penalty to pre-empt the low priority work, i.e., the time to reactivate the disk in a power saving mode, the low priority work cannot be scheduled any time the high priority work is not present without, potentially, significant impact on performance of the high priority work. As a result, in our framework, the schedule of power saving modes in disk drives is proactive, which means that the schedule is determined such that the performance target is not violated. The scheduling output of our framework is described by two parameters I and T, where

- I represents the amount of time the system remains idle before a power saving mode starts.

- T represents the maximum amount of time the disk remains in a power saving mode (i.e., if an IO arrives before T elapses, the power saving mode is interrupted).

The adaptive nature of our framework is reflected in the scheduling pair (I, T). Its values change as the workload and/or input parameters change in the system.

B. Modeling Delays Caused by Power Saving Modes

To meet the performance target we develop the algorithm that estimates correctly the *delay* that power saving modes cause to the IOs when scheduled based on an (I, T) pair. Here, we give a high level overview of the algorithm that calculates the delay propagation due to power saving. For further details we refer the interested reader to [10].

Let us assume that W is the average additional waiting that IOs experience due to power saving. Delay W can be at most P, since P is the time it takes to re-activate the disk. W is estimated as

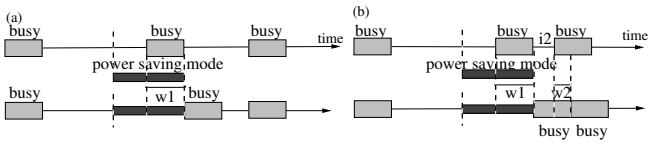

Figure 1. (a) No delay propagation. (b) Delay propagates.

$$W = \sum_{w=1}^{P} w \cdot Prob(w), \tag{1}$$

where w represents a possible delay and $Prob(w)$ its respective probability.

In high-end systems, the average idle time, as shown in Table II, can be orders of magnitude below the penalty P, but can also be highly variable. As a result, if an idle interval is used for power savings and a new IO (i.e., new user busy period) arrives while the disk is still in power saving mode, then the delay may affect (i.e., *propagate*) multiple user busy periods not just the first one immediately following a power saving mode. Figure 1 depicts the delay propagation effect.

To estimate $Prob(w)$ of a delay w occurring in the system, we need to identify correctly the events that may happen during disk reactivation that result in a delay w for a number of IOs. The sum of the probabilities of all these events give $Prob(w)$ for a $1 \leq w \leq P$. During this process the delay propagates as follows:

- *First delay*: User IOs arrive during a power saving mode or during the time the disk is being reactivated. They find an empty IO queue yet the disk drive is not ready for service. As a result, this *first* user busy period consisting of these requests is delayed by some amount of time w_1 ms (where $1 \leq w_1 \leq P$).

This possible first delay w is P for all intervals whose length falls between I and $I + T - P$, for a scheduling pair (I, T), with probability CDH($I + T - P$) - CDH(I). Similarly, for all idle intervals whose length falls between $I + T - w$ and $I + T - w + 1$, the delay caused is w. Therefore, the probabilities of the possible delays $1 \leq w \leq P$ caused to the IOs of the first delayed busy period are

$$Prob_1(w) = \begin{cases} \text{CDH}(I + T - w + 1) - \text{CDH}(I + T - w), \\ \qquad\qquad\qquad\qquad \text{for } 1 \leq w < P \\ \text{CDH}(I + T - P) - \text{CDH}(I), \quad \text{for } w = P, \end{cases} \tag{2}$$

where CDH(.) indicates the cumulative probability value as captured by the CDH of idle times, and w is the possible value of the first delay.

- *Further propagation*: In general, the delay propagates through multiple consecutive user busy periods until all the intermediate idle periods absorb the initial delay w_1. Specifically, the delay propagates for k consecutive user busy periods if $(i_2 + i_3 + ... + i_k) < w_1 < (i_2 + i_3 + ... + i_k + i_{k+1})$.

Note that we discretized the waiting time w in order to calculate the delay propagation. The granularity of w is connected to the granularity of the CDH and affects the accuracy of the framework. The coarser the CDH, the less accurate our solution would be.

C. Meeting Performance Target D

Here, we develop the method to determine the pair (I, T) for scheduling the power saving modes such that performance does not degrade more than D. To control performance degradation, T represents a proactive measure that encompasses the time that the disk stays in a power saving mode. Therefore, the delay P to reactivate the disk is included in T and the relation $T > P$ must always hold.

A pair (I_l, T_j) satisfies performance target D if

$$D \leq \frac{W_{(I_l,T_j)}}{RT_{w/o\ power\ saving}}, \tag{3}$$

where $W_{(I_l,T_j)}$ is derived in Eq. 1. If the pair (I_l, T_j) satisfies performance target D, then we estimate the time in power savings $S_{l,j}$ associated with it. In estimation of $S_{l,j}$ we reflect that for the scheduling pair (I_l, T_j), the effective time in power saving is only $T_j - P$, for all idle intervals longer than $(I_l + T_j - P)$. For all idle intervals with length o between I_l and $I_l + T_j - P$, the time in power saving is $o - I_l$ as captured in the following equation

$$S_{l,j} = \frac{\sum_{o=I_l}^{I_l+T_j-P} p(o) \cdot (o - I_l)}{E[I]} + \frac{\sum_{o=I_l+T_j-P}^{max} p(o) \cdot (T_j - P)}{E[I]}, \tag{4}$$

where $p(o)$ is the probability of the idle interval being of length o, max is the maximum length of the idle intervals in the CDH, and $E[I]$ is the average idle interval length.

We define the scheduling pair (I, T) to be the pair (I_l, T_j) that results in highest time in power saving $S_{l,j}$ after scanning all possible pairs. This scan takes $O(n^2)$ steps where n is the number of values in the CDH. However, if the pair (I_l, T_j) violates performance target D, then the pairs $I_l, T_o > T_j$ are not considered, because between two pairs with the same I and different Ts, the one with the smaller T causes less delay.

D. Meeting Reliability Target X

In addition to the performance goal D, our framework's goal is to meet the reliability target X. The reliability target X represents number of reactivations per time unit (e.g., one day) a disk can have without impacting its lifetime. We convert X into the portion of idle intervals that can be used for power savings during a time unit without violating the reliability target. The conversion is straight forward as the ratio of X to the number of idle intervals per time unit and we denote it by $P(X)$. The total number of idle intervals is readily available from past workload monitoring.

A scheduling pair (I, T) determines that a disk will be put into a power saving mode during an idle interval with probability (1-CDH(I)), i.e., the probability that an idle interval has length greater than I. Because the number of actual power saving modes should be limited by X, a scheduling pair (I, T) sends the disk to power saving mode with probability $P(X)$ / (1-CDH(I)). This means that if disk is idle for I units of time, it will go into power saving mode only with probability $P(X)$ / (1-CDH(I)). Incorporating the reliability target into our framework, requires for the delay W caused by a scheduling pair (I, T) to reflect the fact that the number of power saving modes is bounded by X.

For that, we correct Eq. (2) to reflect that the *first* delay caused to the IOs by a power saving mode is not with probability CDH(I) but $P(X)$ / (1-CDH(I)). Note that if $P(X) > 1 - \text{CDH}(I)$, i.e., there are less than X idle intervals with length greater than I, then all power saving modes determined by the scheduling pair (I, T), since they don't violate X. Therefore, Eq. (2) becomes:

$$Prob_{1,X}(w) = C \cdot Prob_1(w) \tag{5}$$

where C is defined as

$$C = \begin{cases} \frac{P(X)}{\text{1-CDH(I)}}, & \text{for } P(X) < 1 - \text{CDH}(I) \\ 1, & \text{otherwise.} \end{cases} \tag{6}$$

The reliability target is reflected similarly in the estimation of power savings, thus Eq. (4) becomes

$$S_{l,j,X} = C \cdot S_{l,j} \tag{7}$$

where C is defined in Eq. (6).

IV. Experimental Evaluation

In this section we present detailed experiments which show that our framework estimates scheduling parameters that closely approximate the optimal ones for power savings.

We use a set of enterprise traces at the disk level from an application development server ("Code") and a file server ("File") [11]. These traces are characterized by very low utilization yet their idleness is highly fragmented. Table II shows that the traces are quite diverse, thus constitute an excellent set to evaluate the framework's ability to estimate the best (I, T) parameters for each trace. The columns labeled "Time in Power Savings" include the percentage of time relative to the duration of the entire trace that is used for power savings if *all* idle intervals that can be used for Level 3 or Level 4 savings are indeed used, and if *perfect* knowledge of future workload is available. This value represents an absolute bound on power savings.

For each trace, we first estimate the (I, T) parameters using the analytic methodology presented in Section III. Then we run a trace-driven simulation to validate the accuracy of scheduling decision (I, T), where the power saving modes are activated only after I time units elapse. The disk remains in a power saving mode for at most T time units. A new IO arrival *always* preempts a power savings mode and reactivates the disk drive. Recall that because of the reliability constraint, we randomly select only X out of all idle intervals eligible for savings, i.e., longer than I.

Table III gives an overview of the effectiveness of our framework. All columns labeled "Estim." are estimated by the framework and the ones labeled "Actual" are given by trace driven simulation. The "Target D" column is the user provided input acceptable degradation in performance. The columns labeled "Performance Degradation" should be less or equal to "Target D" if there are no performance violations. The next two columns labeled "Time in Power Savings" give the total effective time in power savings, i.e., the duration of powering down the disk during idle interval omitting the penalty P over a period of time which in our case is the twelve hour trace period. Finally, S_{max} corresponds to the optimal value found by exhaustive search of all possible (I, T) pairs to identify the one that offers best savings with performance degradation equal or under the target D.

The penalty to reactivate the drive is set to $P = 500$ ms (Level 3). The number of allowed power saving periods per a 12-hour period in all experiments is set to $X = 15$, which represents a conservative setting given that a disk drive may tolerate as much as 40,000 spin ups [9] during its lifetime (assumed to be 4 years).

The main observations from this table are:

- The target performance degradation is never violated by the estimated or actual degradation. In addition, the estimated analytic value is *always* higher than the actual simulation value, because the analytic framework offers a conservative approximate solution depending on the granularity of CDH bins.
- Our framework always estimates excellent scheduling parameters for maximum power saving while limiting the number of spin downs per day. This observation is valid across all experiments.
- The time in power savings estimated analytically by the framework is accurate, see its proximity to the actual values given by simulation. High accuracy here is critical because we can quickly decide whether it is worth to engage in power savings or not.
- For $D = 1\%$, it becomes difficult for the framework to capture the very small variations in performance. Recall that the accuracy of the framework depends on the CDH bin granularity.

Overall, the table shows that our framework is robust across all workloads and D values, with excellent accuracy for both power and average delay estimation, without compromising on the reliability constraint X.

V. Conclusions

We have presented a compact analytic model that given performance and reliability targets, it provides answers on "when" and for "how long" idle periods in disk drives can be utilized to put the system in a specific power saving mode such that the targets are met. We demonstrate the robustness of the framework using a set of traces from enterprise storage systems and exhibit its the remarkable accuracy to correctly predict and schedule power savings close to maximum.

Trace	Util	Idle Length		Time in Power Savings (%)		Trace	Util	Idle Length		Time in Power Savings (%)	
	(%)	Mean (in *ms.*)	CV	Lev. 3	Lev. 4		(%)	Mean (in *ms.*)	CV	Lev. 3	Lev. 4
Code 2	0.5	1681.6	2.3	92	87	Code 4	0.1	8293.67	7.8	97	94
File 1	1.7	767.5	2.3	70	53	File 3	0.1	2046.51	9.1	87	79

Table II
GENERAL TRACE CHARACTERISTICS.

"Code 2"						**"Code 4"**				
	Performance Degradation		Time in Power Saving		Max Time in Power Saving	Performance Degradation		Time in Power Saving		Max Time in Power Saving
Target D	Estim.	Actual	Estim.	Actual	S_{max}	Estim.	Actual	Estim.	Actual	S_{max}
1	1	0	0.09	0.09	0.33	1.00	1.00	8.18	4.99	12.57
5	5	0	0.28	0.32	0.33	4.00	1.00	13.68	8.03	13.07
10	10	2	0.29	0.33	0.33	9.00	3.00	21.47	18.89	18.89
20	20	20	0.31	0.35	0.35	20.00	10.00	35.73	35.35	35.35
100	22	21	0.31	0.35	0.37	31.00	25.00	37.79	37.51	37.57

"File 1"						**"File 3"**				
	Performance Degradation		Time in Power Saving		Max Time in Power Saving	Performance Degradation		Time in Power Saving		Max Time in Power Saving
Target D	Estim.	Actual	Estim.	Actual	S_{max}	Estim.	Actual	Estim.	Actual	S_{max}
1	1	0	0.50	0.39	0.39	1.00	0.00	2.69	1.77	5.76
5	5	3	0.73	0.69	0.70	4.00	2.00	6.32	4.42	5.76
10	7	4	0.75	0.71	0.71	10.00	4.00	8.47	6.98	6.98
20	7	4	0.73	0.71	0.71	20.00	6.00	12.02	10.79	10.80
100	7	4	0.73	0.71	0.71	28.00	21.00	13.45	11.17	11.17

Table III
POWER SAVINGS AND PERFORMANCE DEGRADATION ESTIMATED USING OUR FRAMEWORK AND SIMULATION. VALUES ARE IN (%).

ACKNOWLEDGMENTS

This work is supported by NSF grants CCF-0811417 and CCF-0937925. The authors thank Seagate Technology for providing the enterprise traces used for this work.

REFERENCES

[1] D. Colarelli and D. Grunwald, "Massive arrays of idle disks for storage archives," in *Proceedings of the ACM/IEEE Conference on Supercomputing*, 2002, pp. 1–11.

[2] D. Narayanan, A. Donnelly, and A. I. T. Rowstron, "Write off-loading: Practical power management for enterprise storage," in *Proceedings of the USENIX Conference on File And Storage Technologies (FAST)*, 2008, pp. 253–267.

[3] A. Riska and E. Smirni, "Autonomic exploration of trade-offs between power and performance in disk drives," in *Proceedings of the 7th IEEE/ACM International Conference on Autonomic Computing and Communications (ICAC)*, 2010, pp. 131–140.

[4] Q. Zhu, Z. Chen, L. Tan, Y. Zhou, K. Keeton, and J. Wilkes, "Hibernator: helping disk arrays sleep through the winter," in *Proceedings of the ACM Symposium on Operating Systems Principles (SOSP)*, 2005, pp. 177–190.

[5] A. Verma, R. Koller, L. Useche, and R. Rangaswami, "SRCMap: Energy proportional storage using dynamic consolidation," in *Proceedings of 8th USENIX Conference on File and Storage Technologies (FAST'10)*, 2010, pp. 154–168.

[6] Y. Chen, A. Das, W. Qin, A. Sivasubramaniam, Q. Wang, and N. Gautam, "Managing server energy and operational costs in hosting centers," in *SIGMETRICS*, 2005, pp. 303–314.

[7] Seagate Technology, "Constellation ES: High capacity storage designed for seamless enterprise integration," Product overview at: http://www.seagate.com, 2009.

[8] Hitachi Global Storage Technologies, "Power and acoustics management," White paper at: http://www.hitachigst.com, 2007.

[9] K. Li, R. Kumpf, P. Horton, and T. Anderson, "A quantitative analysis of disk drive power management in portable computers," in *Proceedings of the USENIX Winter 1994 Technical Conference on USENIX Winter 1994 Technical Conference*, 1994, pp. 22–36.

[10] X. Mountrouidou, A. Riska, and E. Smirni, "PREFiguRE: a performance, power, and reliability framework for disk drives," in *IFIP Performance 2011 (under review)*.

[11] A. Riska and E. Riedel, "Disk drive level workload characterization," in *Proceedings of the USENIX Annual Technical Conference*, May 2006, pp. 97–103.