Quantitative Estimation of the Performance Delay with Propagation Effects in Disk Power Savings [*]

Feng Yan[1], *Xenia Mountrouidou*[1], *Alma Riska*[2], *and Evgenia Smirni*[1]

[1]College of William and Mary, Williamsburg, VA, USA, fyan,xmount,esmirni@cs.wm.edu
[2]EMC Corporation, Cambridge, MA, USA, alma.riska@emc.com

Abstract

The biggest power consumer in data centers is the storage system. Coupled with the fact that disk drives are lowly utilized, disks offer great opportunities for power savings, but any power saving action should be transparent to user traffic. Estimating correctly the performance impact of power saving becomes crucial for the effectiveness of power saving. Here, we develop a methodology that quantitatively estimates the performance impact due to power savings. By taking into consideration the propagation delay effects. Experiments driven by production server traces verify the correctness and efficiency of the proposed analytical methodology.

1 Introduction

As one of the main power consumption components in today's data centers, storage systems have emerged as components where power can be saved. In data centers, hard disk drives (HDDs) are still the most widely used storage equipments despite the emergence of new storage technologies such as solid state drives (SSDs) [8, 3]. The extremely low utilization of HDDs indicates that there are plenty of idle periods that can be used for power saving purposes. Spinning down the HDDs during idle periods has been adopted in archival or backup systems [1], and even high-end computing environments [5] for power savings.

User service level agreements require power savings to be transparent to user traffic. To achieve efficient power saving while protecting user performance is not easy. If in a power saving mode, the storage system cannot serve the incoming user traffic immediately, because some time is needed to adjust mechanical components to be able to serve the request that arrived while the system is in power saving mode. Such delay is usually orders of magnitude larger than the user response time and may affect more than a single request, i.e., could propagate to subsequent requests and severely compromise performance. Our target here is to develop a methodology that captures the delay propagation and estimates the performance impact caused by power savings.

In [2] a Markov Model of a cluster of disks is used to predict disk idleness and schedule the spin down of disks for power savings. This model is based on two states, ON and OFF, and a prediction mechanism that relies on a probability matrix. A Dynamic Power Management (DPM) algorithm is introduced in [4] that extends the power savings states from idle and busy (idle and busy here are defined according to whether there are outstanding IO requests in storage system) to multiple power-saving states based on a stochastic optimization. This algorithm has the best power savings, i.e., 25% less, and best performance, i.e., 40% less, compared to other DPM algorithms. Both [5] and [9] use workload shaping techniques for power savings and also use a fixed idle waiting period before entering a power saving mode in order to reduce the performance degradation but often no performance guarantees on user traffic. The framework in [7] schedules power savings with performance guarantees by specifying "when" and "how long" to power down a disk. In this paper, we propose an analytic methodology to enhance the framework in [7] by modeling the propagation delay effects and estimate quantitatively the performance impact of power savings.

Although the main contribution of this work lies in its theoretical aspect, we also conduct trace driven simulations to verify its practical benefit. We drive the evaluation via a set of enterprise disk drive traces and compare the results to other common practice methods adopted in many of today's systems. The excellent agreement between the results from the trace driven simulations and the user predefined targets, corroborates on the robustness of the analytical methodology.

[*] This work is supported by NSF grants CCF-0937925 and CCF-1218758. The authors thank Seagate Technology for providing the enterprise traces used for this work.

Input parameters	
D	*Quality metric - performance:* relative increase in average response time due to power savings.
S	*Quality metric - power savings:* portion of time in power savings. Expressed as percentage.
P	Penalty due to power savings (i.e., time to reactivate a disk from a specific power saving mode).
Monitored metrics	
$p(j)$	Probability of idle interval of length j.
$CDH(j)$	Cumulative probability of an idle interval of length j.
$E[I]$	Average idle interval length.
RT	Average IO request response time.
Intermediate metrics	
W	Average additional waiting time IO requests experience due to the disk being in a power saving mode.
w_i	Additional waiting time affecting IOs in the i^{th} busy period following a power saving mode.
$Prob_i(w)$	Probability of w waiting time for the IOs in the i^{th} busy period following a power saving mode.
i_j	Length of the j^{th} idle interval following a power saving mode.
Output parameters and estimated metrics	
I	Amount of time that should elapse in an idle disk before it is put into a power saving mode.
T	Amount of time that a disk is kept in a power saving mode, unless a new IO arrives.
$D_{(I,T)}$	Achieved average degradation of response time due to power savings.
$S_{(I,T)}$	Achieved time in power savings.

Table 1: Notation used in Section 2. All time units are in ms.

2 Methodology

The framework in [7] schedules the power saving such that extra delays due to power saving is transparent to the user traffic[1]. We summarize the parameters used in the algorithm in Table 1. Transparency to the user is measured by the relative performance degradation D, which is defined as the relative increase in average response time due to power savings.

The basic idea is to limit the time in power saving mode so that the small idle periods are not used for power savings and the system can be proactively ready for serving new user traffic. For this, the framework computes two scheduling parameters: I is the amount of time that should elapse in an idle disk before it is put into a power saving mode and T is the amount of time that a disk is kept in a power saving mode, unless a new IO arrives. The cumulative Distribution Histogram (CDH) of idle times observed in the system is used to compute I and T. Since there may be multiple (I,T) pairs that can offer performance guarantees, we therefore index them as (I_l,T_j). Performance does not degrade more than the target percentage D on the average:

$$D \geq \frac{W_{(I_l,T_j)}}{RT_{w/o\ power\ saving}}, \tag{1}$$

where $RT_{w/o\ power\ saving}$ is the monitored IO average response time without power saving and $W_{(I_l,T_j)}$ is the average IO waiting due to the power saving modes using the scheduling pair (I_l,T_j) .

If (I_l,T_j) satisfies the target D, then the corresponding "time in power savings" $S_{l,j}$ can also be computed. Assume that P is the time necessary to bring a disk drive out of a specific power saving mode. Because T_j includes P, for all idle intervals longer than $(I_l + T_j - P)$, the time in power saving is $(T_j - P)$. For all idle intervals with length o between I_l and $I_l + T_j - P$, the time in power saving is $o - I_l$. Therefore,

$$S_{l,j} = \frac{\sum_{o=I_l}^{I_l+T_j-P} p(o)\cdot(o-I_l)}{E[I]} + \frac{\sum_{o=I_l+T_j-P}^{max} p(o)\cdot(T_j-P)}{E[I]}, \tag{2}$$

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

The framework chooses the scheduling pair (I,T) to be the pair (I_l,T_j) that results in highest time in power saving $S_{l,j}$.

2.1 Delay Estimation Methodology

To make the framework correct and efficient, it is critical to estimate correctly and accurately the *waiting time*

[1] The applications end performance can be impacted by many factors (e.g., CPU, memory, networking, etc.), thus for unbiased analysis, we focus only on the disk performance itself, which is measured by the IO requests average response time.

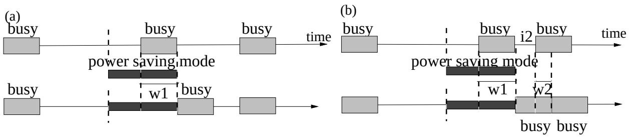

Figure 1: (a) No delay propagation. (b) Delay propagates two busy periods.

(or delay) caused to IOs arriving during or after a power saving mode. Here, we develop a methodology for estimating the delay that can take into consideration of the propagation delay effects.

Recall the W is the average IO waiting due to enabling a power saving mode. Upon an IO arrival, W can be at most P. By denoting a possible delay as w and its respective probability as $Prob(w)$, then

$$W = \sum_{w=1}^{P} w \cdot Prob(w). \tag{3}$$

The power saving mode preemption time P may be longer than the average idle interval and the average busy period. As a result, the delay due to a power saving mode may *propagate* in multiple user busy periods, i.e., it may delay not only the requests in the first busy period immediately following a power saving mode, but also delay the requests in future busy periods. Although all IOs in one busy period get delayed by the same amount, the delay propagates to multiple busy periods and different delays may be caused to IOs in future busy periods because of the activation of a single power saving mode. Figure 1 depicts the delay propagation effect.

To estimate $Prob(w)$ of a delay w, we identify the events that happen during disk reactivation that result in a delay w and estimate their corresponding probabilities. These events are the basis for the estimation of the average waiting W due to power savings. Without loss of generality, we assume that a disk reactivation affects at most K consecutive user busy periods. The larger the K, the more accurate our framework is. In general, the larger the P, the larger the K to improve estimation accuracy. In our estimations, K is set to be equal to P, which represents the largest practical value that K could take. During disk reactivation, the delay propagates as follows:

- *First delay*: IOs arrive during a power saving mode or disk reactivation and find an empty queue and a disk that is not ready for service. These IOs would have made up the *first* user busy period if the disk would have been ready. Their waiting due to power saving is w_1 ms (where the index $i = 1$ indicates the first busy period and $1 \leq w_1 \leq P$).

- *Second delay*: IOs in the “would-be” second busy period in the absence of the power saving mode, could also be delayed if the above wait w_1 is longer than the idle interval i_2 that would have followed the above first busy period. The waiting time experienced by the IOs of the second busy period following a power saving mode is $w_2 = (w_1 - i_2)$.

- *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 $(i_2 + i_3 + ... + i_K) < w_1 < (i_2 + i_3 + ... + i_K + i_{K+1})$. The IO waiting times due to this power saving mode are w_j for $1 \leq j \leq K$.

Denoting with $Prob_k(w)$ the probability that wait w occurs to the IOs of the k^{th} delayed busy period, we estimate the probability of delay w as $Prob(w) = \sum_{k=1}^{K} Prob_k(w)$. Without loss of generality, we measure the idle interval length as well as the wait within a granularity of 1 ms. The granularity depends on the monitoring method, usually, the coarser granularity, the less accuracy, but less monitoring overhead. As a result, the delay P may occur only to IOs of the first delayed busy period, because for the IOs of the second (or higher) delayed busy period the intermediate idle interval would absorb some of the delay and would therefore reduce it. The same argument can be used to claim that the delay of $P-1$ can occur to only IOs of the first and second delayed busy periods. In general, it is true that the delay $w = P - k$ may occur only to the IOs of the first $k+1$ delayed busy periods ($0 \leq k \leq K$).

The fact above is used as the base for our recursion that computes $Prob(w)$ for $1 \leq w \leq P$. The base is $w = P$ and $Prob(w = P) = Prob_1(P)$ because the delay P is caused *only* to the IOs of the first delayed busy period. For a scheduling pair (I, T), the delay to the first busy period following a power saving mode is P for all idle intervals whose length falls between I and $I + T - P$. The probability of this event is given as CDH($I + T - P$) - CDH(I), where CDH$(.)$ indicates the cumulative probability value of an idle interval in the monitored histogram.

The delay w_1 caused to the IOs of the first busy period following a power saving mode may be any value between 1 and P. Using the CDH of idle times, the probability of any delay w_1 caused to the IOs of the first busy period are given by the equation below

$$Prob_1(w) = \begin{cases} \text{CDH}(I + T - w + 1) - \text{CDH}(I + T - w), \\ \qquad\qquad\qquad\qquad \text{for } 1 \leq w < P \\ \text{CDH}(I + T - P) - \text{CDH}(I), \text{ for } w = P, \end{cases} \tag{4}$$

If the length i_2 of the idle interval following the first delayed busy period is less than w_1, then the IOs of the second busy period may be delayed too by, $w_2 = (w_1 - i_2)$. The IOs of the second busy period are delayed by $w_2 = w$

if (1) the idle interval following the first delayed busy period is i_2, which happens with probability $p(i_2)$, and (2) the first delay was $w_1 = w + i_2$, which happens with probability $Prob_1(w + i_2)$. Assuming independence, the probability $Prob_2(w)$ is given by the equation

$$Prob_2(w) = \sum_{j=1}^{P-w} Prob_1(w+j) \cdot p(j), \qquad (5)$$

where $Prob_1(w + j)$ for $1 \leq j \leq P - w - 1$ is defined in Eq. (4) and $p(j)$ is the probability of an idle interval of length j.

The delay $P - 1$ can occur only to the IOs of the first busy period with probability $Prob_1(P - 1)$ and to the second busy period with probability $Prob_2(P - 1)$. Using Eqs. (4) and (5), we get

$$Prob(P-1) = Prob_1(P-1) + Prob(P) \cdot p(1). \qquad (6)$$

This implies that $Prob(P - 1)$ depends only of $Prob_1(.)$ and $Prob(P)$ which are both defined in Eq. (4) and represents how the base $Prob(P)$ of our recursion is used to compute the next probability $Prob(P - 1)$.

Similarly, we determine the probabilities of delays propagated to the IOs of the busy periods following the power saving mode and establish recursion for all $1 \leq w \leq P$. For clarity, we show how we develop the next recursive step and then generalize. Specifically, the delay w_3 is caused to the IOs of the third delayed busy period and w_3 takes values from 1 to at most $P - 2$ (recall that the granularity of the idle interval length is 1 ms).

$$Prob_3(w) = \sum_{j=1}^{P-w} Prob_1(w+j) \sum_{j_2=1}^{j-1} Prob_2(j-j_2) \cdot p(j_2). \qquad (7)$$

The delay of $P - 2$ does not propagate beyond the third delayed busy period and its probability is given as the sum of probabilities of its occurrence to IOs of the first delayed busy period, $Prob_1(P - 2)$, second delayed busy period, $Prob_2(P - 2)$, and third delayed busy period, $Prob_3(P - 2)$. Using Eqs. (4), (5), and (7) we obtain

$$\begin{aligned} Prob(P-2) &= Prob_1(P-2) + \qquad (8) \\ &\quad Prob_1(P-1) \cdot p(1) + Prob_1(P) \cdot p(2) + \\ &\quad Prob_1(P) \cdot p(1) \cdot p(1) \end{aligned}$$

Substituting $Prob_1(P - 1) + Prob(P) \cdot p(1)$ with $Prob(P - 1)$ from Eq. (6) we get

$$\begin{aligned} Prob(P-2) &= Prob_1(P-2) + \qquad (9) \\ &\quad Prob(P-1) \cdot p(1) + Prob(P) \cdot p(2). \end{aligned}$$

In general, for the k^{th} delayed busy period, delay w occurs with probability $Prob_k(w)$ given by the equation

$$\begin{aligned} Prob_k(w) &= \sum_{j=1}^{P-w} Prob_1(w+j) \cdot \qquad (10) \\ &\quad \sum_{o_2=1}^{j-1} Prob_2(j-o_2) \cdot \\ &\quad \sum_{o_3=1}^{o_2-1} Prob_3(o_2-o_3) \cdot \ldots \cdot \\ &\quad \sum_{o_{k-1}=1}^{o_{k-2}-1} Prob_{k-1}(o_{k-2}-o_{k-1}) \cdot p(o_{k-1}). \end{aligned}$$

Recursion in Eq. (10) is generalized using probabilities defined in Eq. (11) as follows

$$Prob(w) = Prob_1(w) + \sum_{j=w+1}^{P} Prob(j) \cdot p(j-w). \qquad (11)$$

To estimate the average delay W, first all $Prob_1(w)$ for $1 \leq w \leq P$ can be estimated using Eq. (4). Then starting from $w = P$, all probabilities $Prob(w)$ for $1 \leq w \leq P$ are computed using the recursion in Eq. (11). Note that the granularity of the CDH bins determines the granularity of the recursion step. In the above presentation, we assume, without loss of generality, that each bin is 1 ms.

3 Experimental Evaluation

In this section, we evaluate the delay propagation methodology using a set of disk-level enterprise traces collected at mid-size enterprise storage systems hosting dedicatedly server applications such as a development server ("Code") and a file server ("File") [6]. Each trace corresponds to a single drive and is collected at the disk level, therefore each request is a single task arriving at the disk. The traces record the arrival and departure time of each disk-level request allowing for exact calculation of the histogram of idle times.

Table 2 gives an overview of these traces. The traces are characterized by very low utilization, but highly fragmented idleness. Notice the differences in the mean idle intervals and their coefficients of variation (C.V.s), which suggests there are many small idle periods and the delay propagation effects may be severe.

We use our delay estimation methodology to estimate the delay due to power savings and computes the scheduling pair (I, T) that can satisfy user predefined targets. Specifically in the trace-driven simulation, the power saving modes are activated only after I idle 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, which takes P units of time. In our experiment,

Trace	Util	Idle Length (ms)	
	(%)	Mean	CV
Code 1	5.6	192.6	8.4
Code 2	0.5	1681.6	2.3
File 1	1.7	767.5	2.3
File 2	0.7	2000.2	3.8

Table 2: General Trace characteristics. All traces have a duration of 12 hours.

we assume that the power saving is at Level 3 [7], i.e., the drive heads are unloaded from the drive platters but without slowing the platter's rotation. The drive consumes 15-20% less power than the "active" idle mode, where the disk can serve the new user requests immediately. The penalty to reload the disk heads is about half a second, so the P is set to 500 *ms*.

We show the correctness and robustness of our delay estimation methodology by comparing the simulation results with the various user predefined performance targets. We also show the efficiency of the framework by comparing its performance to common practices used for power savings in storage systems. The most common approach is to idle wait for a fixed amount of time before putting a disk into a power saving mode. Usually the fixed amount of time is set to be a multiple of the penalty P to bring back the disk into the operational state. Here we show results obtained when the idle wait I is set to $2P$. A second approach is to guide power savings by the current utilization levels in the storage node (i.e., disk drive). Here, we apply the first approach of fixed idle wait only if the utilization in the last 10 min is below a pre-defined threshold (set to the average utilization in the trace).

In Figure 2, we plot the performance degradation and power saving results of the framework and the above two common practice methods. In the interests of space, we only show the results of Code 2. Three performance targets are evaluated: a strict performance target of 10%, a normal target of 50%, and a loose target of 100%. For the two common practice methods, the performance target cannot be set beforehand and slowdown may be unbounded. In practice, in order to limit the performance slowdown, a fixed idle wait and/or a utilization threshold are set such that the system goes into power savings only occasionally.

In Figure 2, the y-axis is in log-scale. Absolute values are shown above each bar. These results come from the trace-driven simulations. By comparing the absolute values above each bar with the performance targets under each bar, the agreement suggests the correctness and effectiveness of our methodology. In addition, the fixed idle wait method for $I = 2P$ results in a slowdown of 5662%, i.e., several orders of magnitude more than the framework for less than 10 times the power savings. The utilization-guided method reduces performance degradation of the fixed idle wait method, but its power savings are 10 times lower than our framework for similar performance slowdowns. These results clearly illustrate the efficiency of the framework in the delay estimation methodology in terms of meeting performance targets while achieving high power savings.

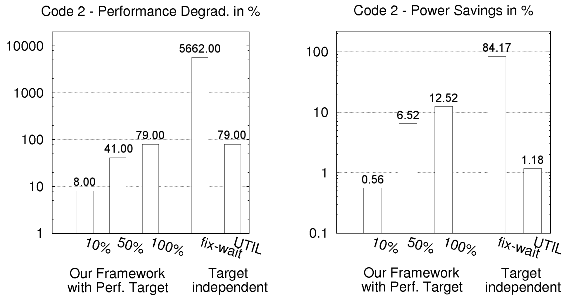

Figure 2: Performance degradation and time in power savings for Code 2 under the framework we use and other common practices, i.e., fixed idle wait and utilization-guided. Because y-axis is in log scale, the y-axis values are shown for each bar.

References

[1] COLARELLI, D., AND GRUNWALD, D. Massive arrays of idle disks for storage archives. In *Proceedings of the ACM/IEEE Conference on Supercomputing* (2002), pp. 1–11.

[2] GARG, R., SON, S. W., KANDEMIR, M. T., RAGHAVAN, P., AND PRABHAKAR, R. Markov model based disk power management for data intensive workloads. In *Cluster Computing and the Grid* (2009), pp. 76–83.

[3] GRUPP, L., DAVIS, J., AND SWANSON, S. The bleak future of nand flash memory. In *Proceedings of the USENIX Conference on File And Storage Technologies* (2012).

[4] IRANI, S., SHUKLA, S., AND GUPTA, R. Online strategies for dynamic power management in systems with multiple power-saving states. *ACM Transactions in Embedded Computing Systems 2* (2003), 325–346.

[5] NARAYANAN, D., DONNELLY, A., AND ROWSTRON, A. I. T. 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.

[6] RISKA, A., AND RIEDEL, E. Disk drive level workload characterization. In *Proceedings of the USENIX Annual Technical Conference* (May 2006), pp. 97–103.

[7] RISKA, A., AND SMIRNI, E. 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.

[8] VASUDEVA, A. Are SSDs ready for enterprise storage systems? White paper at: http://www.snia.org/, 2011.

[9] VERMA, A., KOLLER, R., USECHE, L., AND RANGASWAMI, R. SRCMap: Energy proportional storage using dynamic consolidation. In *Proceedings of 8th USENIX Conference on File and Storage Technologies (FAST'10)* (2010), pp. 154–168.