19th Annual IEEE International Symposium on Modelling, Analysis, and Simulation of Computer and Telecommunication Systems

Toward Automating Work Consolidation with Performance Guarantees in Storage Clusters

Feng Yan, Xenia Mountrouidou, Alma Riska, Evgenia Smirni
College of William and Mary
Williamsburg, VA 23187, USA
Email: fyan,xmount,riska,esmirni@cs.wm.edu

***Abstract*—With most of today's systems being highly distributed, from data centers to cloud and storage clusters, there is a prevalent need for robust methodologies for work consolidation to improve load balancing but also to optimize non-traditional performance measures. Such alternative measures may include power savings, e.g., it may be desirable to shut down a lowly utilized node by moving some or all of its work to another node. In this paper, we present a methodology for distributed work consolidation that keeps track of the workload in the various nodes of the cluster and makes intelligent decisions on how much work to move from a sender node to a receiver node in order to minimally "affect" the performance of the receiver node or alternatively limit any performance degradation due to consolidation in a controlled way. The proposed methodology is based on continuously monitoring the workload on sender and receiver nodes, collecting lightweight statistics in the form of histograms of coarse granularity, and deciding when and how to initiate the work transfer. Extensive experimentation using trace-driven simulation confirms the robustness of the methodology.**

I. Introduction

The volume of digitally stored data has grown rapidly and continues to grow with a tremendous pace with the expectation of reaching a total of 35 zetabytes by 2020 [1]. Data centers host most of these data and efficient operation requires meeting power consumption, performance, data reliability, availability, and integrity targets that may change during the lifetime of the system. As workload demands and system targets vary, there is a need to make data server resources available on demand in a robust and transparent way. For example, if the aim is to consume as little power as possible while maintaining specific service level objectives, when work intensity decreases it is natural to aim at consolidating work into a smaller set of available resources in order to allow the rest to be taken off-line such that power consumption is reduced [2], [3].

Storage is one of the main components of a data center and it consumes about 20 to 30 percent of the total power, making work consolidation in storage systems important and relevant. Since data is not all accessed simultaneously, it is common to have an underutilized or even idle storage system [4], [5], [6], making the storage component a promising one for power savings. To this end, there have been efforts in developing techniques that exploit idleness in a storage system and its devices by taking off-line portions of it without impacting data availability or violating performance targets [7], [8], [9], [10]. Even more, there have been efforts that aim at *increasing* idleness in selective storage devices or systems by redirecting portions of the workload [11], [12] or the entire working set from a set of storage devices to another [13] in order to open more opportunities for power savings.

Consolidation of a storage workload to a limited set of devices is undoubtedly beneficial for power consumption but may come at a dear cost: performance of the storage system may suffer. Striking a balance between consistently meeting service level objectives and power savings is difficult given that future workload demands are seldom known a priori. Judicious selection of *which resources* to consolidate and on *which nodes* to initiate power savings is not an easy task. Using simplistic measures such as average node utilization to guide consolidation can result in lamentable system operation, as we show later in this paper. The question now becomes, where to shift data, how much data to shift, and from which senders to which receivers.

In this paper, we present a solution to the above problem by using a quantitative framework that estimates (i.e., predicts) the performance of consolidated storage workloads in the available nodes in the cluster. The predicted consolidated node performance is paired with projected power savings at each node [7] to enable a cluster-wide identification of the nodes that if put off-line can bring the highest power savings, while performance of the storage devices that serve their redirected workload does not degrade beyond a pre-defined response time target.

By pairing the framework that predicts performance of a storage node serving consolidated workload with the framework that estimates power saving capabilities in a storage node, we enable the storage system to determine whether consolidation can be effective. The framework is based on data monitoring at the device level that is routinely done on storage systems and its estimations are lightweight, i.e., they are based on simple histogram scanning that is used as input to simple equations. A critical component is the use of a look-up table that couples arrival intensities and service demands on each node with average expected service time. This look-up table is continuously changing as workload evolves, by continuously "learning" how performance depends on the changing workload. Finally, the framework can be deployed on every node, making this approach highly distributed, thus highly scalable.

This paper is organized as follows. In Section II we give an overview of the framework that estimates power saving

1526-7539/11 $26.00 © 2011 IEEE
DOI 10.1109/MASCOTS.2011.32

capabilities in a storage device such as a disk drive. In Section III we develop the methodology that learns the dependencies between the arrival, service, and response times in a storage device. Section IV presents an extensive set of trace-driven experiments that demonstrates the robustness of the framework. We conclude and discuss future directions in Section VI.

II. Background

In this section, we give an overview of a methodology that demonstrates how a storage device such as the disk drive can identify its own capabilities for power savings [7] as well as how this methodology is used to define how to increase power saving capabilities via workload shaping [11], i.e., redirecting part of workload (or the entire workload) to other nodes in the cluster.

The above methodologies take into consideration the fact that power saving modes in disk drives impact performance. Performance degrades because if a request comes and finds the disk in a power saving mode, it needs to wait for the disk to come back up, which may take up to 20 seconds. This waiting time is several orders of magnitude higher than the average service time in a disk drive. As a result, for high-end systems with stringent performance targets, if scheduled naively, power saving modes may cause a significant drop in performance.

A. Estimating Power Savings Capabilities in Disk Drives

In [7], an analytic framework is proposed to estimate the capabilities for power savings in a disk drive. The framework uses the histogram of idle times to capture the important stochastic characteristics of idleness in the device. The histogram is used to probabilistically assess the occurrence of long idle intervals that can be used to schedule power saving modes in the drive, i.e., slow down the disk or shut it off completely.

Deciding power saving capabilities based on the histogram of idle times for the current workload makes this framework versatile, because it eliminates the need to monitor and make decisions based on device utilization or request arrivals, whose relations to power savings are more complex to capture. Furthermore, the framework incorporates the relation between workload characteristics (such as sequentiality or randomness) and the impact of the periods of power saving on performance via a single *penalty* parameter. Such abstraction of the workload parameters results in an analytic framework that can estimate the power saving capabilities (in portion of time in power savings) for a range of power saving modes and penalties on system performance.

The framework treats power savings as a low priority work that takes place during idle intervals. To this end, it takes as input the histogram of idle times, the penalty during power savings, the response time without savings and an acceptable performance degradation. The output of the framework is a "schedule" that defines when and for how long to put the disk in power savings. The framework calculations are based on scans of the histogram of idle times, which may have several hundred entries only. As a result, scanning is nearly instantaneous making the framework lightweight and compact. An important feature in the framework is that it allows the storage node itself to sort out several power saving options (resulting from combinations of power saving modes and degradation on performance) internally and present to the cluster management module the most effective one. To summarize, the evaluation is lightweight and with minimal overhead which is mostly related to monitoring rather than computation.

B. Workload Shaping for Power Savings

A way to enhance power savings in a single storage device (or a set of devices) is to increase the length of idle intervals. This can be done by identifying portions of the workload that can be redirected somewhere else in the system. In this paper, we refer to this activity as workload shaping [11], [12], [13], [14]. Workload shaping may require to copy some of the data from the storage device that are to be placed in the power saving mode at the *new* destination node [11], [13]. However, the main feature in all these techniques is that the storage devices are consolidated and the ones that remain active take over the load of the storage devices that are to be placed in a low power mode (or even off-line).

These methods define clearly what part of the workload to redirect out of a storage device. [12] proposes to redirect the entire WRITE traffic arriving at the storage device. [13] proposes to redirect the entire active working set. [11] proposes to remove the most frequent busy periods (statistically or according to a probabilistic weighting scheme).

We stress that if the system monitors the idle periods and the busy periods in a system, then it is possible to estimate the power saving capabilities of these workload shaping techniques via the framework in [7]. This requires that the histogram of idle times is updated to reflect the changes in the workload, and this can be done by monitoring the requests within busy periods [11]. For example, for the workload shaping in [12], the histogram of idle times can be updated to reflect the removal of WRITEs. In general, the capabilities of these workload shaping techniques to improve power savings can be quantified [7] and for a given workload, the storage device can determine which workload shaping technique to use and how much load it can off-load to other nodes.

Once the workload to be redirected is removed, the next question is to determine the receiver of this extra load. Consolidation of the workload over a smaller set of storage devices is effective only if the cluster continues to perform without violating performance targets in the system. This paper focuses on this problem and in the following sections we present a method that predicts performance, measured via the response times, in a storage device that serves consolidated work.

III. Performance Degradation Estimation on a Storage Device due to Consolidation

Here, we present a quantitative framework that enables a storage system to intelligently identify which storage devices should be put into power saving modes and which storage devices should serve the consolidated workload such that the performance target is not violated while power savings are the highest possible. In this framework, we assume the performance target is the response time which, on the average, should not be higher than a predefined limit.

Our goal is to construct a lightweight and accurate framework that predicts the response time in a potential *receiver* storage device knowing

- its workload in terms of average arrival rate, average service rate, and average response time;
- the portion of the load in the *sender* device that may potentially go to the *receiver* device and its characteristics such as average arrival rate and average service rate.

We expect average workload statistics such as arrival, service, and response time to be available in the logs that monitor the storage system operation. The workload shaping input is expected to be available from monitoring that is used to facilitate the various workload shaping techniques such as those introduced in Section II.

For the framework to reach a consolidation decision, the following two steps are required. First, estimate the arrival and service rate of the consolidated workload at the receiver storage device. Second, predict the response time of the consolidated workload at the receiver end.

A. Estimating arrival and service rates of the consolidated workloads

In order to be able to predict the arrival and service rates at the consolidated storage node, we assume that it is known the arrival rate λ_s and the service rate μ_s of the portion of the workload from the sender node that will be consolidated at the receiver storage node. It is also assumed that the arrival rate λ_r and the service rate μ_r at the receiver storage node are known. The characteristics of the merged workload, i.e., its arrival rate and its service rate, if a certain pair of nodes is selected for consolidation, is

$$\lambda_c = \lambda_s + \lambda_r \tag{1}$$

and

$$\mu_c = \frac{\lambda_r}{\lambda_s + \lambda_r}\mu_r + \frac{\lambda_s}{\lambda_s + \lambda_r}(\rho \cdot \mu_s) \tag{2}$$

where λ_c and μ_c are the arrival and service rate after consolidation, while ρ is a correction factor that accounts for the change in the service rate from the sender storage node to the receiver storage node. The parameter ρ is meant to capture *only* the effect that differences on disk positioning time has on service time.

Estimating average arrival rate at the receiver storage node, is less complex than estimation of the service rate, given that the load is known on both sender and receiver nodes. The average service rate is more complex, particularly since in storage devices the service process depends highly on workload characteristics (such as randomness and the mix of READs and WRITEs) and the physical characteristics of the device (e.g., rotation speed for a disk drive). The parameter ρ is used to capture the effect of the physical characteristics of the device. As we introduce a new set of data into the active data set on a disk drive, it is possible to break any existing characteristics in the workload such as sequentiality. This can impact the expected service rate. However, if the sender load is expected to be much smaller than the current load at the receiver, then the workload characteristics of the receiver (before consolidation) should continue to dominate the workload. In addition, workload characteristics of the receiver storage node may be preserved if the placement of the new data is done such that the locality of receiver's working set (if exists) is preserved. This is possible especially since in many cases the amount of data to be moved may be small, perhaps only a few GBytes in size. Placement choices and how they can change the disk layout are outside the scope of this paper.

B. Predicting response time of the consolidated workload

In storage devices, it is not straightforward to determine the response time for a workload, given its arrival and service rates, because of all the idiosyncrasies that determine the device service rate, such as workload access patterns and disk characteristics, as well as the complex features of the arrival process, such as variability and burstiness. Our goal is to have an approximate, yet accurate prediction of the response time in the receiver storage device under the consolidated workload with parameters defined in Subsection III-A. For this, we propose to predict the response time, by "learning" the response time patterns over time for the storage nodes in the cluster. The goal is for individual storage devices to monitor and record the observed response times for pairs of arrival and service rates.

We propose to construct on-line (i.e., as the system operates) *look-up tables* that contain tuples of observed average arrival rate, average service time, and average response time over periods of time. The averaging can be at different granularities from 15 minutes intervals to a few hours. As the system operates, the goal is to update the look-up table at each storage node by adding new tuples that have not been there before, but also by avoiding repetition, such that the size of the table remains relative small. We show in the evaluation section that a look-up table of several hundred entries provides good prediction accuracy and is small enough to facilitate fast searching through it. Constructing the look-up table is part of the "workload learning" process in our framework.

The expectation is that the workload that the receiver storage node sees as a result of consolidation has been, at some point in time, already observed and recorded and can be used in the future for approximate, yet fairly accurate, predictions. As a result, as the estimation of the consolidated arrival rate and

service time is done using Eqs. (1) and (2), the look-up table at the receiver is searched and the closest tuple that matches *both* arrival and service rates is selected as the tuple whose response time value is chosen as the approximate prediction of the performance at the receiver storage node.

An exact match of the estimated arrival and service pair with the tuples available in the look-up table is not expected. However, while constructing the look-up table, the density of the tuples can be controlled. In the look-up tables that we have constructed, we have followed the rules that the differences in the arrival and service rates of the neighboring tuples should be at 10%. While this condition can easily be satisfied for the common cases and low to medium arrival and service rates, the rare events of high arrival rates or very slow service rates may be more difficult to obtain. For such cases, there is a possibility to run off-line benchmarks to populate the look-up tables with rare events. Here, we focus on building the look-up tables on-line. How to populate the tables off-line is not considered in this paper.

The prediction of the response time is only an approximation but as we show in the evaluation section, it serves as an excellent way to quantify performance in a consolidated cluster and select the right pair or pairs of nodes that should consolidate their workload for an overall reduction in the number of active storage devices. We stress again that the average arrival and service rate here is mainly served as the "index" to find the response times measured in the real system environment in the look-up table. The performance effects of workload characteristics such as sequentiality and burstiness are already captured in the look-up tables.

One source of error in our predictions laid out in Sections III-A and III-B, is associated with the fact that we expect the arrival and service rates observed for the past several hours to hold for the next several hours. While this is expected to be the case in clusters where changes in the workload happen gradually, there may be cases when the short past is very different from the close future. The accuracy of the framework presented here would naturally suffer in cases of abrupt and unexpected temporal workload changes. There are methods that can complement our workload prediction framework to account for the abrupt changes. A feedback-loop monitoring could be used to check at small time intervals (every several minutes) if the observed average arrival rate and service rate are close (up to a threshold) to the observations of previous monitoring period. If there is violation, then the estimations should be recalculated around the new observations. The threshold should be large enough to avoid unnecessary oscillations. Another approach is to detect at coarse granularity (i.e., several hours) any obvious regular workload changes, such as the ones that may be associated with daily and weekly business cycles. If such cycles are learned in advance, then they can be predicted and the corresponding actions taken to ensure that the decisions are made on accurate current workload characteristics. There are various aspects to be evaluated and analyzed before such methods can be incorporated into our prediction framework. They are not discussed in this paper and are left for future work. Here we work under the assumption that the short-term past predicts well the short-term future (i.e., the error associated with the differences in the workload between the past and the future is acceptable).

Trace	Util (%)	Mean Arrival Rate	Mean Service Rate	C.V. Arrival Rate	C.V. Service Rate	R/W Ratio
CODE1	5.6	0.0089	0.1596	1.56	0.22	5.48
CODE2	0.7	0.0013	0.1859	1.34	0.06	1.39
FILE1	1.7	0.0033	0.1938	1.07	0.27	8.28
FILE2	0.7	0.0011	0.1596	2.93	0.20	3.63

TABLE I
GENERAL TRACE CHARACTERISTICS.

IV. EXPERIMENTAL EVALUATION

In this section, we evaluate the effectiveness of the consolidation framework having as target to maximize power saving capabilities while controlling performance degradation after the consolidation of resources. We first describe the traces that we use to assess the effectiveness of the consolidation framework. The accuracy of our predictions and estimations are validated by trace-driven simulation. Then, we provide detailed workload characterization that supports the assumptions and decisions used in the development of the estimation and prediction components of the framework. Finally, we use the prediction framework, to decide which storage nodes to consolidate in a cluster. We focus here on a small cluster of four nodes only, to facilitate a clear presentation since the results we present are for each node in the cluster.

A. Traces and Simulation environment

The validation of the proposed consolidation framework is done via trace-driven analysis and simulation. We use a set of enterprise traces measured at the disk level from an application development server ("CODE") and a file server ("FILE") [5]. In each of the measured storage systems, there are tens of storage devices (disks) organized in several RAID groups. The traces do not provide information on the RAID groups. However we have observed that disks of small sets have identical workloads, which allows us to infer with high confidence that those disks belong on the same RAID groups. Although we could have used representative disks of each RAID group, to simplify presentation, we have selected traces that correspond to *only* 4 representative RAID groups, two from each storage subsystem. The total duration of each trace is twelve hours. Each trace record consists of: the arrival time, the departure time, the type of the request (i.e., read or write), the request length in bytes, and the location on the disk. This information allows to calculate exactly a rich set of metrics that we can use in the evaluation process of our framework. In Table I, we show a subset of these metrics of interest.

The data in Table I show that the disks are clearly underutilized, implying that here are opportunities to temporary consolidate data and obtain power savings. Similarly, the low

arrival rates and relatively much higher service rates indicate that temporary consolidation of work in a few disks only may be effective without taking a toll on system performance.

The consolidation framework is based on the assumption that short-term past predicts well the short-term future. In the twelve hour traces, we use the first 6 hours ("short-term" past) to collect statistics with regard to arrival, service, and response time, as well as idleness, and apply the learning on the second 6 hours ("short-term" future) of the trace.

In Figure 1, we plot for each of the four traces, the arrival rate and the service rate for 5 minute intervals as a function of time. In each of the plots we separate with a vertical dashed line the learning period (the first 6 hours) from the testing period (the next 6 hours). The main observation is that both arrival and service rate are fairly stationary, as shown by both the average and the coefficient of variation of the measured metrics.

Figure 1 confirms that the service rate exhibits almost no change throughout the duration of the traces. This observation supports the argument that the workload characteristics for each of the traces remain fairly stable, resulting in an almost deterministic service process.

The arrival rate in each of the traces is not as deterministic as the service rate (i.e., its coefficient of variation is between 1 and 3) but the average, that is used in our predictive framework, changes only slightly. The highest change we notice is for the FILE2 trace, for which, as we show later, also the prediction errors are higher.

Another interesting observation in Figure 1 is that often higher arrival rates (more work) correspond to higher service rate (i.e., faster service). This is attributed to the specifics and optimizations of disk scheduling that always aims at minimizing seeks in drives. For our specific target of consolidated workloads, only slightly increasing the load at some receiver storage node may actually result in lower service times because of optimization of the disk service process. Consequently, response times are expected to suffer minimally from the additional load.

In a cluster, there is a high chance that the disks or storage devices in general can be heterogeneous. This is the reason why in Eq. (2), we introduce the correcting factor ρ, that on the average captures the differences in physical capabilities (such as rotation speed) between different disks. While, there may be ways to determine ρ off-line, here we estimate ρ by analyzing the service rates for requests of the same or similar sizes. We group requests based on their sizes, in a effort to separate the random portion of the workload (short requests) from the sequential portion of the workload (long requests), because the differences in service rates can mostly be observed in the random rather than sequential portion of the workload.

In our four nodes, the type of disk and rotation speed are not known. In addition, we do not know the sequentiality/randomness of the workloads. We do know that both CODE traces are from the same storage system as are both FILE traces. The first inclination is to set $\rho = 1$ for the pair of CODE traces and the pair of FILE traces. Figure 2 left shows for the FILE1 and FILE2 traces the service rates (measured by MBytes/ms) as a function of the request size for each IO request. In order to eliminate the effect of seek optimization for queued requests on the service times, the plots show service rates only for the requests within a busy period in our traces.

Clearly, because of sequentiality, service rates increase as request sizes increase. The service rates of short IOs (left part of the plot) are almost indistinguishable between the two disks (points overlap with each other in the figure). Even the rates of large IOs are also similar. This suggests that both short and long IOs behave similarly on both FILE1 and FILE2, suggesting the same "random" (and "sequential") behavior in the two traces. This justifies our choice of $\rho = 1$. The right graph of Figure 2 plots the same metrics but now for CODE1 and FILE1. The behavior captured by the right graph is very similar as the behavior of the left graph in Figure 2, suggesting the same "random" behavior across FILE and CODE traces too,[1] The plots in Figure 2 justify setting $\rho = 1$ in Eq. (2) even when evaluating pairings of FILE and CODE traces. This process for estimating ρ can be automated by allowing each storage node to maintain together with other metrics a small histogram of requests sizes and the service rates observed for each of them.

B. Response Time Prediction

We predict performance (measured via average response time) on the device that serves the consolidated workload using a look-up table, which records the observed response time for a pair of observed arrival and service rate. The prediction of the arrival and service rate at the receiver node is done using Eqs. (1) and (2). For each node, we store the average arrival rate (observed in the short-past which will serve as the prediction of its near-future arrival rate). In the experiments presented here "short-past" and "near-future" are intervals of 6 hours each, that correspond to the first and the second part of the trace in Figure 1.

An additional information needed is the amount of work to be shifted to the receiver server. We use the workload shaping techniques outlined in [11] that also calculate which requests are to be moved provided that the intermediate buffer (i.e., the total data to be transferred) is equal to 1 GB, 5 GB, or 10 GB. Note that the size of these buffer sizes is relatively small. Moving just a small amount of data rather than the entire working set (e.g., as proposed in [13]) can be very beneficial for consolidation purposes. The data that we move is based on workload characterization of the most frequently accessed blocks or groups of blocks [11]. As a result this intelligent data copy may relieve the disk from the most highly accessed blocks and increase the idleness used for power savings considerably. The small amount of data movement can also relieve the concern of power consumption during copying and make the data placement problem easier. In addition, placing only a small amount on data on the

[1]We have plotted the service rates as a function of request size for all combinations of FILE and CODE traces. The behavior is very similar to that reported in Figure 2 and is not shown here in the interest of space.

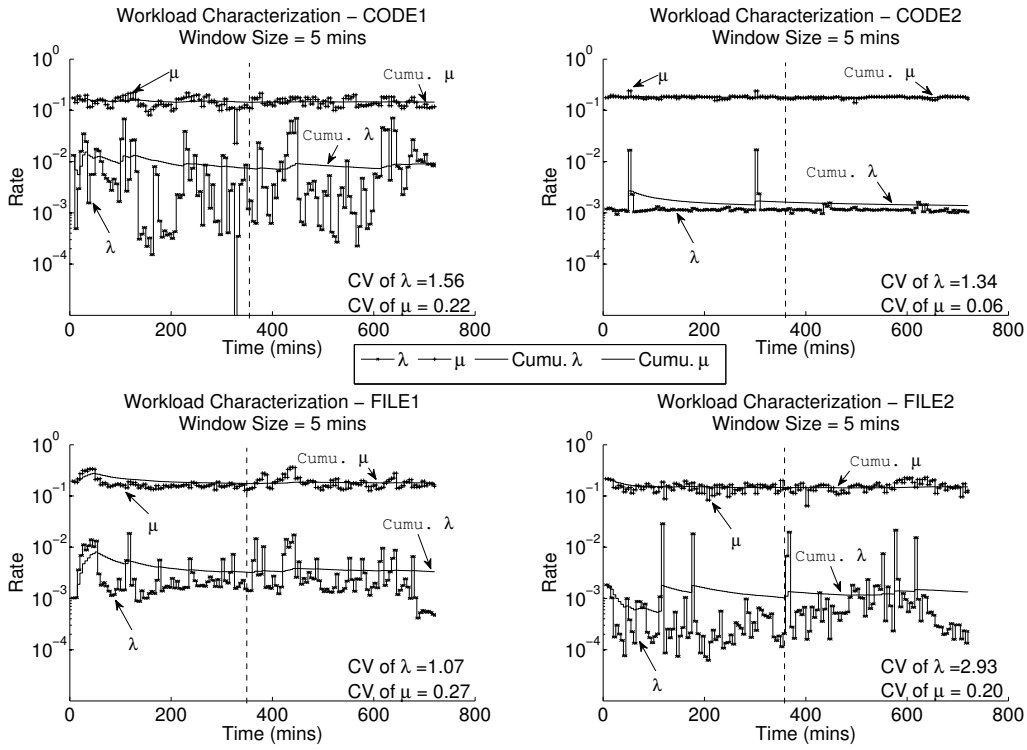

Fig. 1. The workload characterization of the four traces. λ is the Mean Arrival Rate of each window (5 mins), μ is the Mean Service Rate of each window, Cumu. λ is the Cumulative Mean Arrival Rate across the time, Cumu. μ is the Cumulative Mean Service Rate across the time. The rate is measured in msec and plotted in log scale. The vertical dash line in Time = 360 mins separate the first part (left side, which we use as learning period) and second part (right side, which we use as testing period) of the entire trace.

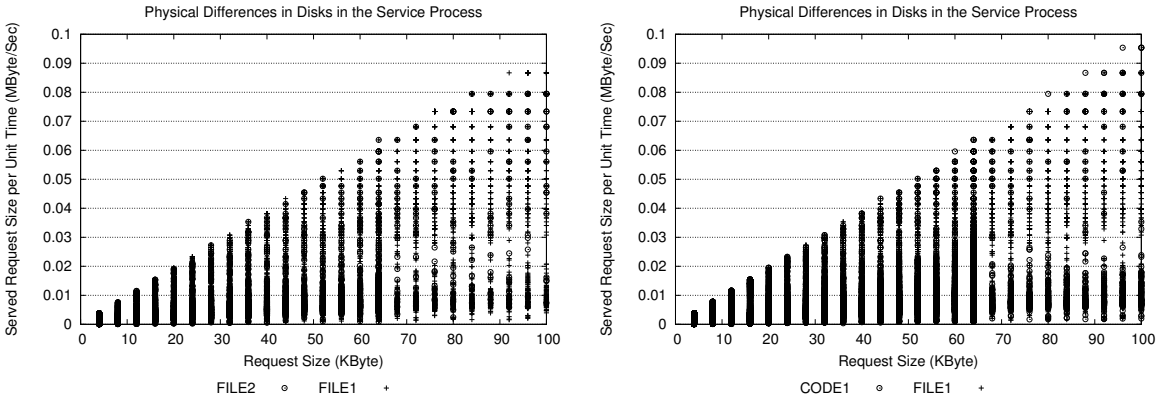

Fig. 2. Identifying effects of physical differences in disks in the service process.

receiver end minimizes the odds of dramatically increasing the service process due to the additional work. Note here the redirected workload would have to be served someplace anyway, so by making another disk serve it does not increase the overall power consumption. Figure 3 illustrates the IO load to be redirected from the sender node to the receiver node for two workload shaping techniques: Busy Period Offloading (BP-Offload.) and Probalistic Offloading (Prob.-Offload.) [11]. The BP-Offload. offloads the most frequently accessed busy periods, i.e., groups of blocks between two idle intervals, until the predefined buffer is filled. The Prob.-Offload. removes a number of busy periods based on the correlation of the length of idle intervals succeeding the busy period (e.g., a long idle interval following another long one), and aims at concatenating

long idle intervals. The figure clearly illustrates that irrespective of the sender or the workload shaping technique used, the redirected workload is only a fraction of the overall workload. Therefore, the overall load at the receiver may only increase a little. Consequently the expectation is that the performance at the consolidated nodes will degrade only slightly, if it degrades at all.

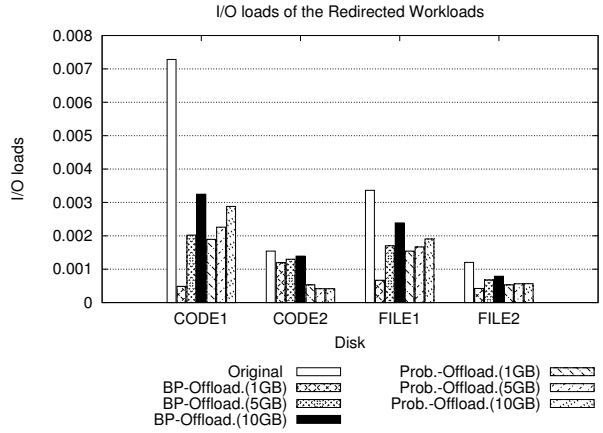

Fig. 3. IO load to be redirected from the node that would be set off-line to the node that would remain active.

An integral part of the workload prediction is to approximate the response time at the receiver storage node after work is consolidated. We achieve this by building look-up tables that hold observations of different pairs of arrival and service rates and the corresponding response times, see Section III. The look-up tables are constructed from the traces of Table I. Since the length of our traces is relatively limited, then we used varied observation lengths from 15 minutes to one hour that result in look-up tables of 512 entries each. Here, because the physical capabilities of all storage devices are very similar (see Figure 2), we merge the tables constructed by all traces into a single table.

Selecting the best matching tuple in the look-up table is an integral part of our prediction. Naturally, all possible arrival/service exact values are not going to be found, so we approximate by exploring values that are within 10% of the anticipated arrival and service rates. While we expect the tables to be dense enough to allow for 10% match, in case no matching pair is found, we rescan the look-up table with 5% higher difference in matching. Among the set of pairs that meets these criteria, we select the one that would best continue to maintain the "trend" of the observed response time. For example, if the current response time in both storage nodes under consideration are higher than the one predicted from the look-up table, then that prediction is not possible. This is supported by our assumptions that the service rates will change only based on the differentiator ρ and consequently the response time should be at the minimum not better than both of them. If there are still multiple tuples that have matched our criteria, then we go for the one that minimizes the sum of the differences between the observed rates and the anticipated ones.

The response time estimations for all cases under consideration are given in Figures 4 and 5. Note that except for the bar that is labeled "Original" for each receiver (and that corresponds to the response time without any consolidated workload), the graph also shows the framework's prediction (labeled "Estimation") and the actual response time after consolidation (labeled "Actual"). Recall that all framework estimations are done using the first half of the trace for all nodes. The simulation validates the accuracy of the estimations on the second half of the traces. The consolidated workloads in the second half of the trace maintain the same service process as measured in the second half of the trace, since the assumption is that the small areas that will hold the replicated data can be placed such that the service process does not degrade. In most of the cases the framework overestimates response time, so the chances for wrong suggestions are small. Overall, the framework is successful in identifying pairs of sender-receiver nodes given certain power and performance targets.

C. Consolidation Decisions

In order to make decisions on how to consolidate the storage devices in a cluster such that the power savings are as high as possible without violating performance targets, each node should know "how much power it can save" and "how to achieve such power savings". We use the workload shaping proposed in [11] and the framework proposed in [7] to predict power saving benefits for a given workload shaping method as well as the amount of data and the load that needs to be redirected for that purpose. In Figure 6, we show power savings for each of the four storage nodes in our cluster for BP-Offload. and Prob.-Offload. workload shaping schemes. The figure shows the percentage of time that the sender (disk) can be placed in low power mode. As with all estimations in this paper, the workload shaping estimations are done based on monitored metrics during the first half of the traces that are applied (i.e., tested) the second half of the traces.

The "Original" bar corresponds to the time that the system can be in power savings *if* there is no workload shaping and only the observed idleness in the storage node is taken into consideration. The graph also reports savings for the two workload shaping methods and three different sizes of data to be moved (i.e., buffers equal to 1, 5, and 10 GBytes). The content of the buffers (i.e., "what" is going to be replicated in the storage node) depends on workload shaping. For details on the shaping methods and their performance, we refer the reader to [11].

Figure 6 shows that if we assume a cluster of 4 storage nodes serving the workload of CODE1, CODE2, FILE1 and FILE2, then CODE1 and CODE2 have the highest potential for power savings. While for some workloads the buffer size matters, a medium buffer size of 5 GBytes performs well overall.

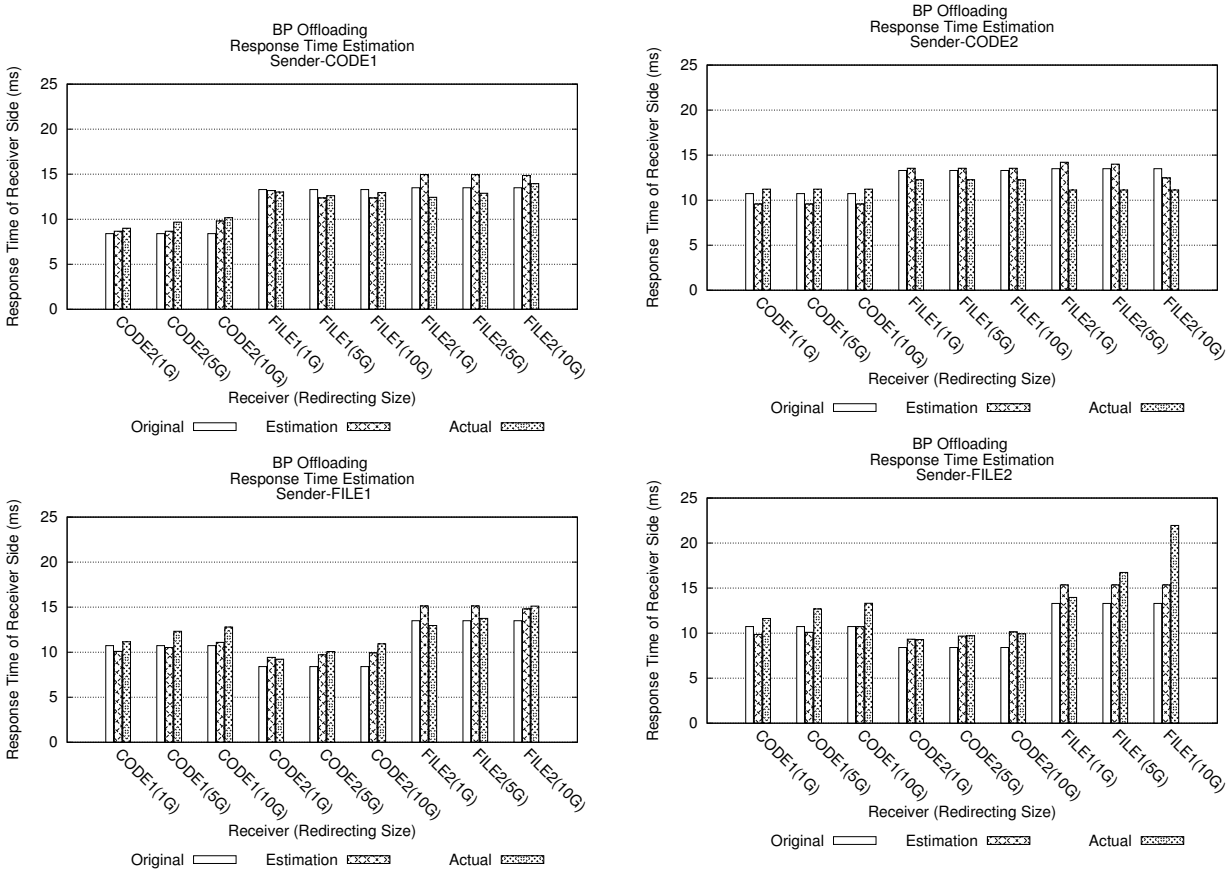

Fig. 4. Performance measured via response time for BP-Offload..

Replicating the data from each storage node as defined in Figure 6 somewhere in the cluster and redirecting the IO load accessing that data to the consolidated node, would result an additional load for each cluster, see Figure 3. The values shown in the plots are estimations of how the accesses on the data replicated to other nodes would look in the near future based on the observations from the first half of each trace.

Figure 6 suggests the best disks to offload work elsewhere but does not tell us "where" to move this work. Figure 4 and Figure 5 give the answers to this question. Here, we present the estimation and validation of performance degradation (in terms of average response time at the receiver end) for each of the possible (sender, receiver) pairs in the storage cluster. Each plot represents the possible pairings of a sender storage node and the potential receivers. The inaccuracies in the prediction of the consolidated response time come from the changes in the workload between the learning period and the testing period, as well as the density in the look-up table and their ability to provide a close enough match to the predicted arrival and service rate pairs that are used to locate in the look-up table the expected response time.

The decision on which pairs of storage nodes to choose for possible consolidation is done based on the storage performance target: average response time has to be always below a certain value. For example, if the receiver must have an average response time less than 10 ms, then if either BP-Offload. or Prob.-Offload. are used at the sender, pairing CODE1 as sender with CODE2 as receiver is a good idea, ditto for FILE2 as a sender and CODE2 as receiver. However, once CODE2 receives the load from CODE1 (buffer size of 10 GB), then no other pairings in the cluster would satisfy the performance condition. With such performance targets, power savings can be initiated only 40% of time on a single storage node. Also note that our estimation (and validation) points to a counterintuitive choice of a node to be put off-line: CODE1 is the one with the highest utilization in the cluster of 5.6 (see Table I), i.e., an unlikely choice for any scheme that makes decisions based solely on utilization levels.

If the performance target is response time of 15 ms, then most pairs can be selected except the following two cases: FILE2 as a sender and FILE1 as a receiver when sender using BP-Offload., and FILE1 as a sender and FILE2 as a receiver when the sender uses either of the offloading methods. Then, the best choice is to select two nodes to be turned off and two nodes to serve the consolidated workloads. Since CODE1 and CODE2 provide the highest power savings, then they can be selected as senders and FILE1 and FILE2 as receivers. For a 1 GByte buffer (the smallest buffer size), CODE1 achieves

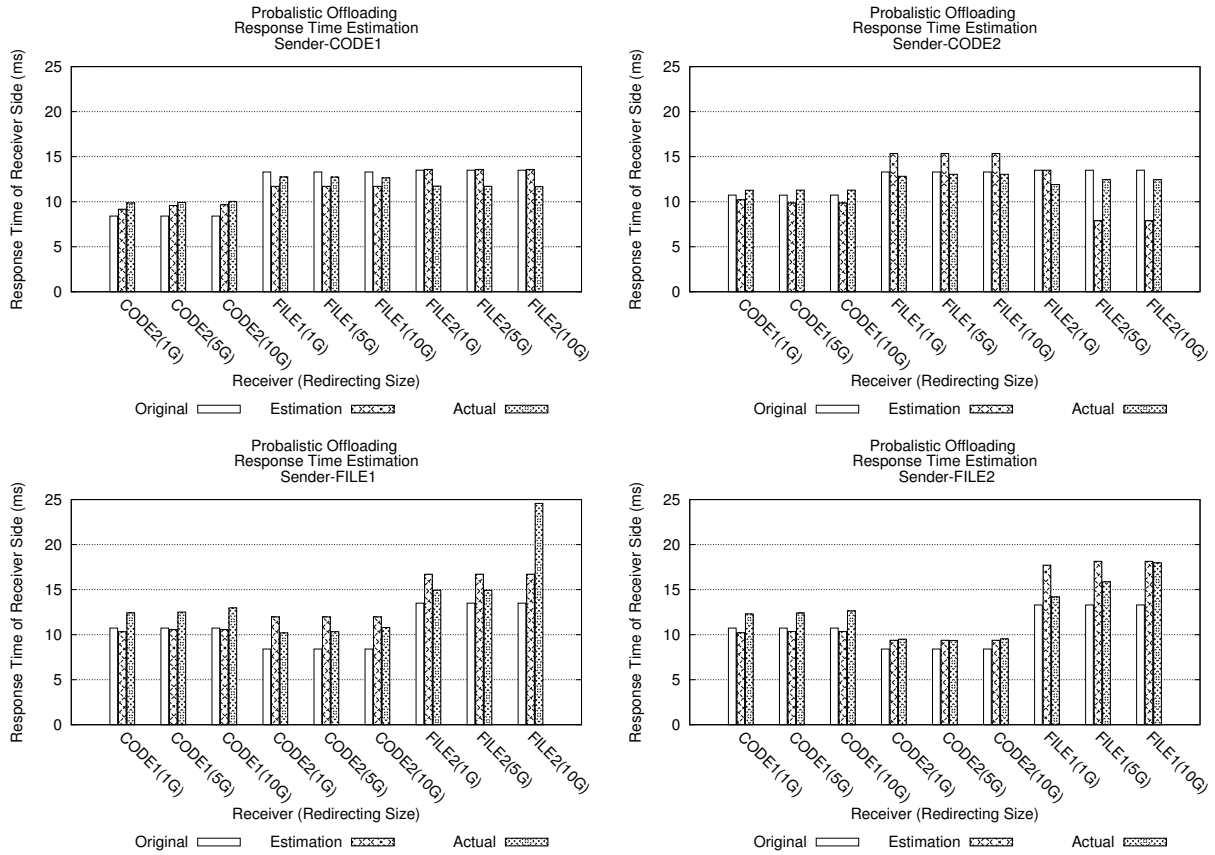

Fig. 5. Performance measured via response time for Prob.-Offload..

40% time in power saving under Prob.-Offload. while CODE2 achieves almost 40% time in power saving under BP-Offload., for a total of two devices providing 40% time in power savings for each.

Such decisions are not obvious, since all traces have low utilizations and any pairing of nodes would represent an opportunity for schemes that make decisions only based on utilization may result in detrimental savings. For example, FILE1 does not have high power savings compared to other traces, and FILE1 would perform poorly if FILE2 offloads its work on it.

V. Related Work

There is a large body of work on storage and resource consolidation to improve efficiency in a data center. pClock [3] is a framework that multiplexes workloads statistically while achieving performance isolation via scheduling. pClock guarantees deadlines for well behaved workloads and latency requirements are met as long as burst sizes and IO rates do not exceeded specified limits. HARMONY [15] is a server and storage virtualization framework based on continuous monitoring and adjustments to different workload, that aims at load balancing to improve performance and reduce resource overloading. In [16] the authors find via experimentation the time it takes to offload work from servers to Virtual Machines (VMs) can be given by measurements from a single VM, the performance degradation due to VM migration is longer than the migration time, and show that parallel migration leads to shorter times.

Efforts have been placed to use learning techniques in predicting workload and performance in storage systems. In [17], the authors use a relative fitness model to predict the performance difference between two storage devices so that the changes of I/O rate can be captured when workload moves from one device to another. In [18], the author develop a Profile Hidden Markov Models based methodology by recognizing primitive operations in a trace, aiming to identify the application I/O access patterns.

The framework proposed in this paper differs significantly from previous approaches in that it guarantees low performance degradation due to the consolidation and provides a methodology to estimate beforehand the performance of receiver after consolidation using minimal information.

VI. Conclusions and Future Work

We propose a method for consolidating workloads in a storage cluster while meeting performance targets such that the

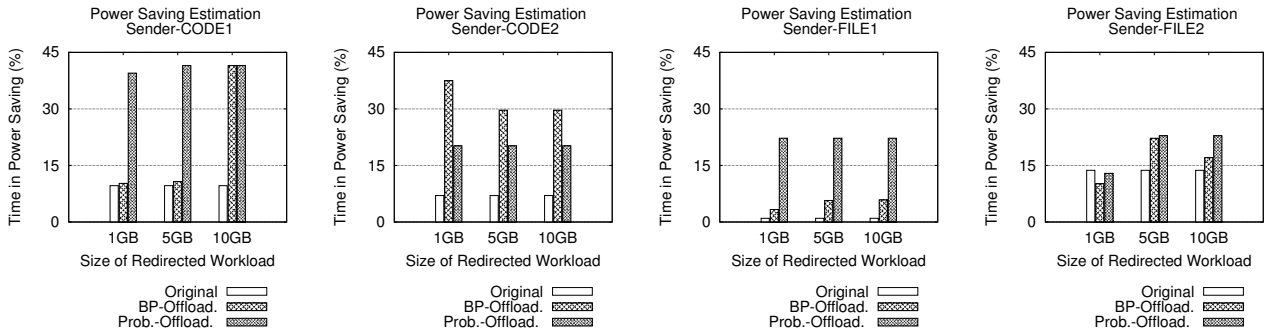

Fig. 6. Power Saving by different offloading methods.

transparency to the end users is preserved. Our estimation provides criteria for the work consolidation between sender and receiver storage nodes in the cluster that aims at maximizing metrics such as power savings. At the center of the framework is a learning method that enables prediction of performance in presence of workload consolidation. The proposed framework offers great flexibility in proposing alternatives that can maintain performance guarantees that remain below pre-advertised values.

In the future, we intend to improve on the accuracy of our method to estimate the receiver response time. We plan to explore different learning methods to detect any regular cycles, such as the daily and weekly business cycles, in order to reduce the impact of abrupt changes in workload characteristics. For further accuracy, we intend to incorporate an off-line component into the process of populating the look-up tables to ensure uniformity over the entire state space, particularly to capture extreme cases that are only rarely encountered in average workloads that are expected to run on the system. We also want to expand the current framework to cater to other storage features (in addition to consolidation) such as data replication for availability, reliability, back up, and virtualization.

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. We also thank the anonymous referees and our shepherd William Knottenbelt for useful suggestions that have improved the presentation of this work.

REFERENCES

[1] International Data Corporation (IDC), "Digital universe report, 2010," http://www.emc.com/digital_universe.

[2] Q. Zhu, J. Zhu, and G. Agrawal, "Power-aware consolidation of scientific workflows in virtualized environments," in *Proceedings of the 2010 ACM/IEEE International Conference for High Performance Computing, Networking, Storage and Analysis*. IEEE Computer Society, 2010, pp. 1–12.

[3] A. Gulati, A. Merchant, and P. Varman, "pclock: an arrival curve based approach for qos guarantees in shared storage systems," *SIGMETRICS Performance Evaluation Review*, vol. 35, no. 1, pp. 13–24, June 2007.

[4] R. Golding, P. Bosch, C. Staelin, T. Sullivan, and J. Wilkes, "Idleness is not sloth," in *In Proceedings of the Usenix Technical Conference UNIX Advanced Computer Systems*, 1995, pp. 201–212.

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

[6] L. Eggert and J. Touch, "Idletime scheduling with preemption intervals," in *In Proceedings of the 20th ACM Symposium on Operating Systems Principles (SOSP)*, 2005, pp. 249–262.

[7] 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.

[8] 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.

[9] C. Weddle, M. Oldham, J. Qian, and A. Wang, "PARAID:a gear-shifting power-aware raid," in *In Proceedings of 5th USENIX Conference on File and Storage Technologies (FAST'07)*, 2007, pp. 245–269.

[10] J. Guerra, W. Belluomini, J. Glider, K. Gupta, and H. Pucha, "Energy proportionality for storage: impact and feasibility," *SIGOPS Operating Systems Review*, vol. 44, no. 1, pp. 35–39, March 2010.

[11] X. Mountrouidou, A. Riska, and E. Smirni, "Adaptive workload shaping for power savings on disk drives," in *Proceeding of the second joint WOSP/SIPEW international conference on Performance engineering*. ACM, 2011, pp. 109–120.

[12] 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.

[13] 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.

[14] Q. Zhu, F. M. David, C. F. Devaraj, Z. Li, and Y. Zhou, "Reducing energy consumption of disk storage using power-aware cache management," in *Proceedings of the International Symposium on High-Performance Computer Architecture (HPCA)*, Febuary 2004, pp. 118–129.

[15] A. Singh, M. Korupolu, and D. Mohapatra, "Server-storage virtualization: integration and load balancing in data centers," in *Proceedings of the 2008 ACM/IEEE conference on Supercomputing*, 2008, pp. 53:1–53:12.

[16] M. Zhao and R. Figueiredo, "Experimental study of virtual machine migration in support of reservation of cluster resources," in *Proceedings of the 2nd international workshop on Virtualization technology in distributed computing*, ser. VTDC '07, 2007, pp. 5:1–5:8.

[17] M. Mesnier, M. Wachs, R. Sambasivan, A. Zheng, and G. Ganger, "Modeling the relative fitness of storage," *ACM SIGMETRICS Performance Evaluation Review*, vol. 35, no. 1, pp. 37–48, 2007.

[18] N. Yadwadkar, C. Bhattacharyya, K. Gopinath, T. Niranjan, and S. Susarla, "Discovery of application workloads from network file traces," in *Proceedings of the 8th USENIX conference on File and storage technologies*. USENIX Association, 2010, pp. 14–14.