Gradient based method to Learn Temporal Energy Efficient Data Storage Array Policies

1st Ramesh Doddaiah
Dell Technologies
Hopkinton, USA
ramesh.doddaiah@dell.com

2nd Aurongzeb Deeder
Dell Technologies
Texas, USA
deeder_aurongzeb@dell.com

***Abstract*—Data storage arrays are made up of hundreds of costly hardware resources such as, CPU's, memory, flash storage, hard disks, compression, de-duplication, and encryption hardware components. Energy consumption of these resources depends on array workloads across different time windows. Forecasting array energy requirements is hard as a result most of these hardware resources are powered on forever leading to high and recurring operating expenditure. In this paper, we propose a Stochastic gradient descent based method to learn temporal energy efficient policies across different time windows. The method learns to proactively power down various storage array resources without affecting the customer quality of service such as low-latency and IOPs (Inputs and outputs per second).**

***Index Terms*—Data storage array, Gradient method, Energy efficient policy generation**

I. Introduction

The energy consumption of a data storage array varies based on internal and external workloads. For example, compressing large block IOs(Input Output) leads to more energy consumption compared to compressing small block IOs. Unnecessary compression, encryption, or decryption of each IOs leads to higher energy consumption. It is quite hard to figure out the compression ratio of incoming IO without compressing the data first. If the incoming IO is encrypted, then it should be decrypted first before trying to compress. Also, if the incoming IO is red-hot, we end up compressing the red- hot data that could be changed soon. This doubles the need for energy to decompress and encrypt the data to the host in subsequent reads. Unnecessary compression, encryption, decryption, and deduplication of red-hot data leads to a higher data footprint consuming more disk space and in turn more energy. Applying compression, encryption, decryption, and deduplication on cold-data leads to a lower data footprint consuming less disk space and in turn less energy. A balance needs to be found between red-hot and cold-data distribution, and to figure out when to apply data reduction techniques to reduce the array energy consumption.

II. Problem Definition

In this paper, we build extent-based statistics for each logical volume. ARIMA (Autoregressive Integrated Moving Average) based time series forecasting is applied to these extents to forecast the red-hot and cold extents for various time windows. We avoid applying data reduction to red-hot data thus saving more energy and we apply data reduction to cold data thus saving more disk space and energy overall. In each time window, our stochastic gradient descent-based algorithm learns to power ON and power OFF each of the data storage array hardware resources. It is hard to manually learn the combinations of required storage array resources for different workloads in each time window. Our algorithm starts by randomly selecting a few hardware resources to power ON from a uniform distribution and gradually learns to power down the hardware resources one by one and checks the MSE (Minimum Squared Error) loss by comparing the new IO latency and IOPs with the expected QoS for the respective storage group devices. It will learn to power OFF more resources until the time MSE loss starts to converges. At that time, it reaches a equilibrium and learns an efficient number of various hardware resources that need to be powered on to maintain the low-latency and high IOPs as per the QoS. We collect such good policies and store them in our policy repositories.

III. Algorithm

1) **Input:** Expected QoS and Current IOPs and Response Time per Storage Group
2) **Output:** Temporal Energy Efficient Policy
3) $\boldsymbol{\theta} \leftarrow random\_uniform(low = 0, high = 1)$
4) Perturb each $\boldsymbol{\theta}$ to Power-ON (1) or Power-OFF (0) respective data storage array hardware resource.
5) Root mean square error between new IOPS and Response Time and expected IOPS and Response Time to compute the LOSS
6) Get the new gradient, clamp between 0 and 1, and update the Temporal Policy to new values
7) Above steps are run until the LOSS converges
8) Final policy explains which hardware resources needs to be powered ON for a given workload in a time window to meet the expected Response Time and IOPS QoS

As shown in figure 1, each array hardware resource has an entry in $\boldsymbol{\theta}$. For different time windows this algorithm generates different energy efficient data storage policies. This solution is applied on mid-range and enterprise storage arrays.

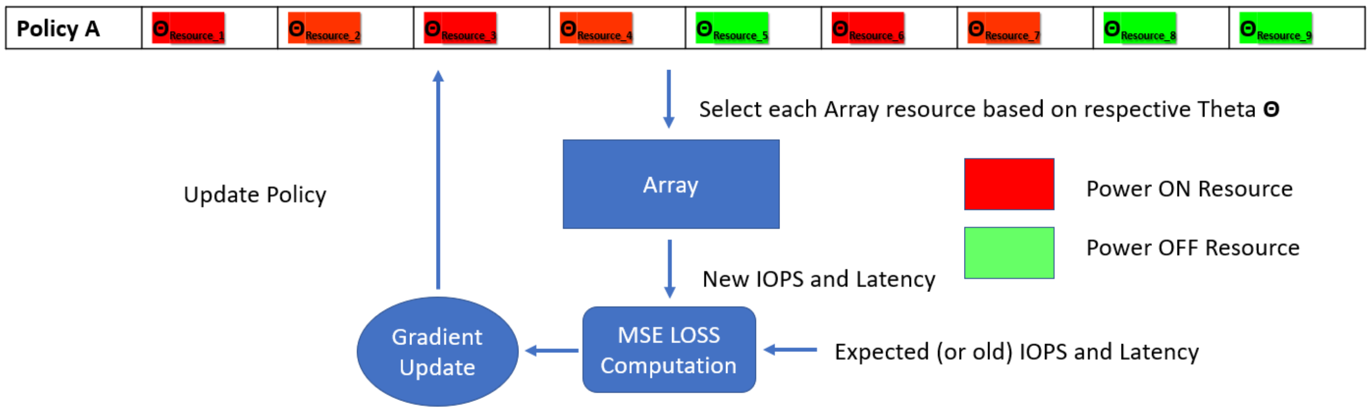

Fig. 1. Temporal energy efficient data storage array policy generation algorithm. Gradient learns to update the Tensor to power ON and OFF array resources to learn various temporal policies.

IV. Conclusion

Searching manually to choose which array resources to power down is cumbersome across different time windows. This gradient-based method learns to power ON and OFF required array resources as needed to reduce power consumption and meet customer service quality.