An Adaptive Checkpoint Model For Large-Scale HPC Systems

Subhendu Behera
ssbehera@ncsu.edu
North Carolina State University
Raleigh, North Carolina

Lipeng Wan
wanl@ornl.gov
Oak Ridge National Laboratory
Oak Ridge, Tennessee

Frank Mueller
fmuelle@ncsu.edu
North Carolina State University
Raleigh, North Carolina

Matthew Wolf
wolfmd@ornl.gov
Oak Ridge National Laboratory
Oak Ridge, Tennessee

Scott Klasky
klasky@ornl.gov
Oak Ridge National Laboratory
Oak Ridge, Tennessee

ABSTRACT

Checkpoint/Restart is a widely used Fault Tolerance technique for application resilience. However, failures and the overhead of saving application state for future recovery upon failure reduces the application efficiency significantly. This work contributes a failure analysis and prediction model making decisions for checkpoint data placement, recovery, and techniques for reducing checkpoint frequency. We also demonstrate a reduction in application overhead by taking proactive measures guided by failure prediction.

KEYWORDS

High Performance Computing, Failure Prediction, I/O subsystem, Checkpoint/Restart, Burst Buffers

ACM Reference Format:
Subhendu Behera, Lipeng Wan, Frank Mueller, Matthew Wolf, and Scott Klasky. 2019. An Adaptive Checkpoint Model For Large-Scale HPC Systems. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Given the scale of modern-day High-Performance Computing systems, failures are imminent and frequent. Recent investigations [2, 3] develop failure prediction models to improve High-Performance Computing(HPC) resiliency. We aim to utilize the DESH failure prediction model [2] to predict failures in advance with a predicted lead time. With sufficient lead time, a proactive measure can be taken to avoid failure and computation waste. The idea is to raise a failure alarm whenever a recognized pattern of logs is found on a node. With sufficient lead time, appropriate action can be taken to migrate the application from a node under risk of failure to a new and healthy node.

Apart from proactive live migration, log-based failure analysis can also help in making decisions about checkpoint data placement. As already known, an application's efficiency on the HPC system is severely impacted by I/O performance which often presents a bottleneck. To alleviate this problem, Burst Buffers (BBs, fast NVMe storage devices) are now used in HPC I/O subsystems. BBs are used as an intermediate fast storage medium for checkpoint data for faster Checkpoint/Restart. BBs also differ in architecture. For example, Summit's compute nodes have local NVMe storage of 1.6TB attached to them, whereas Cori's compute nodes share BBs that are clustered on special nodes. In the case of clustered BB nodes, the checkpoint data can be retrieved even if a compute node fails. However, the same is not true for local BBs as they become inaccessible upon node failure. Hence the stored data on local BBs is asynchronously transferred to the Parallel File System (PFS). So classifying failures as node failures and soft failures can help to make an appropriate decision regarding the placement of checkpoint data and its recovery. If a failure is a catastrophic node failure then the checkpoint data should be written to PFS. Otherwise, it can be written to local BBs.

We propose a checkpoint model that takes advantage of modern BBs-based I/O subsystems and a failure prediction/analysis model. We also use the adaptive checkpoint model derived from [4] to determine an optimal checkpoint interval that considers proactive fault mitigation rates and makes efficient use of both BBs and the PFS.

2 DESIGN

Our model is derived from the checkpoint model [4] which decides the optimal checkpoint interval while considering the daily write limit of BBs for optimal use of both BBs and the PFS. We use this model as our default model when there is no failure prediction. Our new checkpoint model takes decisions based on the following scenarios.

- `Is a failure Predicted ?` During the periods when a failure is not predicted we use the adaptive checkpoint model [4]. This ensures that the application state is saved efficiently with the use of BBs and the PFS. Checkpoint data is saved on to the BBs or the PFS based on the limit of BB writes (see [4]). If BBs are local, then checkpoints bleed off to the PFS asynchronously and slowly while computation continues.
- `Does the predicted failure have enough lead time ?` In case a failure is predicted, we need to have enough lead time to perform proactive live migration or safeguard checkpoint.

Unpublished working draft. Not for distribution.

Conference'17, July 2017, Washington, DC, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn