Enabling Data Analytics Workflows using Node-Local Storage

Tu Mai Anh Do[1], Ming Jiang[2], Brian Gallagher[2], Albert Chu[2], Cyrus Harrison[2]
Karan Vahi[1], Ewa Deelman[1]
[1]USC Information Sciences Institute, Marina Del Rey, California
[2]Lawrence Livermore National Lab, Livermore, California
{tudo,vahi,deelman}@isi.edu,{jiang4,bgallagher,chu11,cyrush}@llnl.gov

ABSTRACT

The convergence of high-performance computing (HPC) and Big Data is a necessity with the push towards extreme-scale computing. As HPC simulations become more complex, the analytics need to process larger amounts of data, which poses significant challenges for coupling HPC simulations with Big Data analytics. This poster presents a novel node-local approach that uses a workflow management system (WMS) to enable the coupling between the simulations and the analytics in scientific workflows by leveraging node-local non-volatile random-access memory (NVRAM).

KEYWORDS

scientific workflows, Big Data analytics, node-local storage

1 INTRODUCTION

Modern scientific simulation applications are becoming more complex in terms of scale and the amount of data that needs to be analyzed over time. Scientists are now exploring the use of data analytics frameworks, such as Apache Spark, for analyzing data from large-scale simulations. As we approach the era of extreme-scale computing, it has become imperative that we manage the resulting data in an effective fashion. Advances in high-performance computing (HPC) allow simulations to run at ever increasing scale, while the use of Big Data frameworks allow large datasets to be analyzed efficiently. Currently, scientists employ the use of scientific workflows to manage and automate the large-scale simulation and associated analysis of the output datasets.

In trying to combine HPC with Big Data, scientists often run into issues resulting from a mismatch in data representation between traditional HPC and the Big Data ecosystem, such as Apache Spark. This poses challenges in coupling between the two environments, as simulation outputs need to be written out to disk, usually the shared parallel file system, and then ingested by Spark for post-processing. This writing out to disk and then ingesting back increases the time it takes to run the complete end-to-end workflow encompassing both simulation and analysis of the data products.

In this work, we propose a node-local approach that leverages non-volatile random-access memory (NVRAM) to enable data analytics workflows. More specifically, the approach supports the coupling between the HPC simulations and the Big Data analytics, which focuses on two factors: (1) better write performance of generating data to node-local storage compared to global parallel file systems; and (2) the analytic tasks to compute on node-local data stored on the NVRAM to improve the efficiency of local processing data. We evaluate our approach by comparing it to the setup on the Lustre file system in which data is stored and accessed globally.

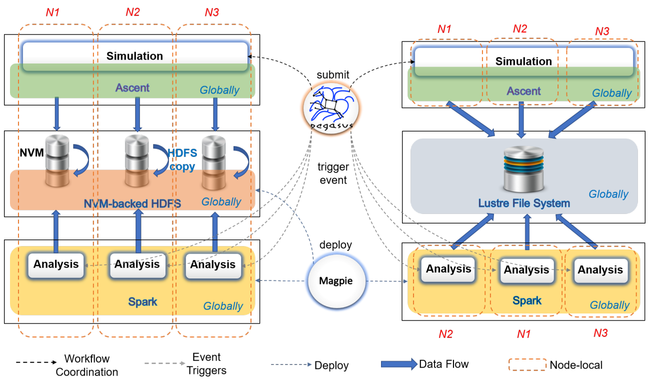

Figure 1: Enabling data analytics workflows on HPC system

In our current approach, the data analysis only starts after the simulation has completed (post-processing). In order to reduce the expensive storage needs for storing large datasets collected over multiple time-steps, the analytics is performed on partial data generated by a few iterations. We plan to explore the node-local technique for analyzing generated simulation data in-situ [2] using event-based triggers [6] for launching workflows as future work.

2 APPROACH

In our proposed node-local approach, instead of storing data to a global parallel distributed file system (Lustre), we decided to use node-local NVRAMs as intermediate storages for partial data generated from the simulation. Instead of writing out to Lustre, we configured the simulation to write out data to the NVRAMs on the compute nodes. We also configured a NVRAM-backed Hadoop Distributed File System (NHDFS) that is built on top of the node-local NVRAMs on the compute nodes. When the data is written locally, we can immediately ingest it into NHDFS. This approach has three advantages:

(1) reduces the latency once data is generated to local NVRAM of the compute node,
(2) analysis tasks can be assigned to compute nodes that possess the data; as a result, the analysis can be performed locally to avoid off-node data transfer, and
(3) NHDFS resolves the data representation mismatch between the HPC simulations and the Big Data analytics.

The overhead of copying data from node-local NVRAMs to NHDFS is discussed in Section 3 and evaluated with the setup on Lustre without HDFS described in Figure 1. The difference between two setups is data placement: node-local NVRAM or Lustre, which is accessible across nodes. In case of Lustre deployment, HDFS setup is not required, as the Spark analysis can ingest data directly from the shared file system. However, there is an overhead of random access over Lustre when the Spark analysis reads in the input data. This node-local approach is designed to be compatible for coupling the simulation with the analytics in both a post-processing and in-situ fashion.

3 EVALUATION

In this section, we evaluate the node-local approach with a complete post-processing pipeline. The pipeline comprises of two jobs in which an HPC simulation job is followed by a representative Spark analysis job. We use Ascent [4], a framework that provides an API to interfere the simulation data at a certain frequency of cycles. We chose Cloverleaf3D [5], a 3D Lagrangian-Eulerian hydrodynamics benchmark, which is a proxy application integrated into Ascent for our experiments. Apache Spark is a well-known framework for Big Data. We opted to use the Random Forest implementation in Spark MLlib as a representative analysis on Big Data to couple with the Cloverleaf3D simulation on HPC platforms. Output generated from the simulation is used to train the model in Random Forests. Spark is used to distribute data to workers for concurrent analysis.

Cloverleaf3D deploys the simulation across multiple nodes via the Message Passing Interface (MPI) and enables OpenMP for multi-threading inside nodes. The Ascent custom Python interface allows each compute node to run a Python script at a certain frequency to write out data to local NVRAM and then copy it to NHDFS. In this experiment, Cloverleaf3D generates data every 20 cycles.

For Lustre setup, Cloverleaf3D outputs data to Lustre as a shared file system instead. Spark then assigns tasks to workers, which are distributed over nodes in Spark clusters. For the NHDFS setup, we configured Cloverleaf3D to output directly to NHDFS.

We modeled the pipeline as a workflow in Pegasus [3], which is a popular workflow management system (WMS) for executing scientific pipelines. Additionally, we use Magpie to setup a multi-node cluster with NHDFS and Spark for our experiments. Magpie [1] comprises of batch scripts developed at LLNL to run Big Data frameworks on HPC platforms.

We deployed and tested our approach on the Catalyst system at LLNL. Catalyst is a 150 teraFLOP/s system with 324 nodes, each with 128 gigabytes of dynamic random access memory, and 800 gigabytes of NVRAM per compute node. It is designed specifically for experimentation with HPC and Big Data analytics applications.

Cloverleaf is configured to run on 8 nodes of Catalyst with OpenMP for multi-threading of 24 threads per node to make use of physical cores available on the node. Similarly, the Spark cluster contains 8 executors and the executor on every single node is configured with 24 cores. We ran the experiment with two setups: the node-local approach on NVRAM and the traditional approach that uses Lustre for data sharing. Data are generated from the simulation and then ingested by the analytics at the size of 6GB, 60GB, and 600GB. For each run, we measure the runtime of the simulation, the analytic and the entire pipeline.

3.1 Experimental Results

Our experimental results indicate that there are improvement gains especially for the Spark analysis with our proposed NVRAM-based node-local approach. Figure 2, shows the improvement of approximately 28% achieved by executing the Spark analysis on NVRAM over Lustre for 600GB input data generated from Cloverleaf3D. The total time of entire pipelines on NVRAM, therefore, improves 19% compared to Lustre.

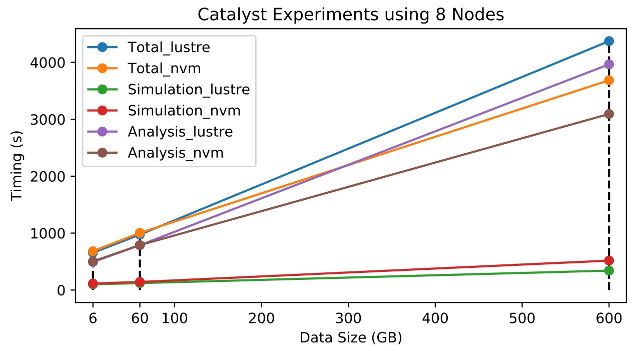

Figure 2: Data placement impact on pipeline performance

We can observe the overhead of NHDFS copying from the timing of the simulation. The simulation suffers 34% slowdown for copying data from local NVRAMs to NHDFS in node-local approach whereas Spark can access data from Lustre without the help of HDFS in Lustre setup. For the total time, this overhead is expected to be hidden by the greater random-access performance of NVRAM.

Our experiments confirm that it is worthwhile to incur an extra cost of copying data twice (first to local NVRAM and then ingesting it into NHDFS) in order to achieve better performance both for Spark analysis and overall pipeline, especially at large data sizes.

4 CONCLUSIONS

In this poster, we described a novel node-local approach that uses NVRAM to achieve performance gains for a pipeline consisting of an HPC simulation job followed by a Big Data analytics job. We tested the approach on an HPC system (Catalyst) with the simulation proxy application Cloverleaf3D in Ascent and the representative Random Forest machine learning algorithm in Spark MLlib. The benefits of the approach are confirmed with the improvement of total time approximately 19% over Lustre.

ACKNOWLEDGMENTS

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344 (LLNL-ABS-755833), with funding provided by LDRD 16-ERD-036. This work was also partially funded by the National Science Foundation under grant #1741040.

REFERENCES

[1] LLNL-CODE-644248. Magpie. http://github.com/LLNL/magpie
[2] J. Bennett et al. 2012. Combining In-situ and In-transit Processing to Enable Extreme-scale Scientific Analysis. In *SC '12*. 49:1–49:9.
[3] E. Deelman et al. 2015. Pegasus, a Workflow Management System for Science Automation. *Future Gener. Comput. Syst.* 46, C (2015), 17–35.
[4] M. Larsen et al. 2017. The ALPINE In Situ Infrastructure: Ascending from the Ashes of Strawman. In *ISAV '17*. 42–46.
[5] A. Mallinson et al. 2014. Experiences at Scale with PGAS Versions of a Hydrodynamics Application. In *PGAS '14*. 9:1–9:11.
[6] S. Pandey et al. 2018. Event-Based Triggering and Management of Scientific Workflow Ensembles. HPC Asia 2018.