Towards Efficient Big Data Management with Transient Storage Systems

Advisors:

- Gabriel Antoniu - gabriel.antoniu@inria.fr (main advisor)
- Nathanaël Cheriere - nathanael.cheriere@irisa.fr

Keywords

HPC, Big Data, data management, distributed storage systems, transient storage systems

Subject:

We live in the era of Big Data. Huge amounts of data are collected from diverse sources, such as sensor arrays, social media, or scientific simulations and other experiments. As data must be analyzed to extract meaningful information, new techniques and tools (such as MapReduce [1]) have recently been developed to manage and process increasingly larger amounts of data at an appropriate speed to meet application requirements.

In the High Performance Computing (HPC) domain, huge amounts of data are collected from complex simulations of physical phenomena (such as climate modeling, galaxy evolution, etc.), and data analysis techniques from the field of Big Data are used to extract pertinent informations. Needless to say, **data storage is a cornerstone of Big Data processing**. Whereas traditional HPC systems usually separate computational resources from storage, using parallel file systems, new supercomputers now often have local storage devices (such as SSDs or NVRAM) located on their compute nodes. This local storage allows users to deploy new types of distributed storage systems along with the applications. Such a storage system, deployed only for the duration of an application's execution, is called a **transient storage system**.

This internship focuses on answering the following research question: "*How can we efficiently deploy and terminate a transient storage system?*". Indeed, a transient storage system only exists during the run time of the application that uses it, initially hosting no data. When the application terminates, data would be lost if not backed up. Thus, two steps are needed: loading data from the persistent storage at initialization time, and dumping the data onto the persistent storage before termination. Both operations involve large data transfers that may slow down the initialization and termination of such storage systems. Optimizing these transfers would lead to faster application deployments and termination.

Multiple relevant contributions can be achieved during this internship depending on the affinity of the student.

1. **Theoretical contributions:** establish a lower bound (mathematical model) of the duration of the operations of loading and dumping data in order to identify their

bottlenecks, and have a baseline for the evaluation of implementations of such systems. This contribution would be in the continuity of [2] and [3].
2. **Algorithmic contributions:** optimize the algorithms used to transfer the data between nodes. For instance, one can consider the network topology, or use AI to monitor the application's behavior and load/dump data in the background.
3. **Experimental contributions:** implement the algorithms in Pufferbench (a benchmark designed to evaluate commission and decommission algorithms - https://gitlab.inria.fr/Puffertools/Pufferbench) to experiment with the algorithms previously designed.
4. **Validation contributions:** implement and evaluate a microservice able to efficiently run the studied operations using Mochi (a set of libraries designed to quickly build distributed storage systems - https://www.mcs.anl.gov/research/projects/mochi/).

Transient storage systems often act as caches for distributed applications, similarly to burst buffers [4]. For such a use, a transient storage system should be able to choose which data to load (*resp.* dump), when to load it, and where to place it. A diverse set of strategies can be used: from "loading everything at start-up" to "loading only when the data is accessed", with more complex approaches using access monitoring and machine learning to predict which data will be needed by the application in the near future. Some of these strategies could be studied depending on the progress and affinities of the student.

This subject will be done in close collaboration with Matthieu Dorier from Argonne National Laboratory (ANL, USA). In addition to the possibility to interact with top-level researchers and scientists from ANL, the work can involve the use of large-scale HPC experimental facilities available at Inria and at ANL.

Skills and abilities

- C++
- Algorithmics
- Mathematics
- Affinity with distributed storage systems

Bibliography

1. Dean, Jeffrey, and Sanjay Ghemawat. "MapReduce: simplified data processing on large clusters." *Communications of the ACM*51.1 (2008): 107-113.
2. Cheriere, Nathanaël, and Gabriel Antoniu. "How fast can one scale down a distributed file system?." *Big Data (Big Data), 2017 IEEE International Conference on*. IEEE, 2017. Available at https://hal.archives-ouvertes.fr/hal-01644928/document
3. Cheriere, Nathanaël, Matthieu Dorier, and Gabriel Antoniu. *A Lower Bound for the Commission Times in Replication-Based Distributed Storage Systems*. Inria Rennes-Bretagne Atlantique, 2018. Available at https://hal.archives-ouvertes.fr/hal-01817638/document
4. Liu, Ning, et al. "On the role of burst buffers in leadership-class storage systems." *Mass Storage Systems and Technologies (MSST), 2012 IEEE 28th Symposium on*. IEEE, 2012.