CS554 Project Ideas

Slurm:Data - Exploring Data-Aware HPC Scheduling with Slurm

Overview

HPC applications are becoming data-intensive that consume and produce large volumes of data with complex data dependencies. Examples of these applications are VISTA (Astronomy), LIGO (Astrophysics), BLAST (Bioinformatics), and ALTAS (High Energy Physics). Current resource managers are ignorant of data locations (local disk vs remote parallel file systems e.g. GPFS, PVFS, Lustre) and the cost to move it. Accessing and retrieving large volumes of data remotely is currently just a remarkable side effect on scheduling computations.

Data-aware scheduling is the solution for this exascale challenge. Data movement will require forethought; moving large amounts of data without intelligent data-aware scheduling will contribute to long idle times on systems as jobs wait for data to become available. Cost models need to be developed for the various storage systems with unique attributes related to capacity and latency. Hierarchical storage systems are more common and more complex, this can be seen with the advent of Burst Buffers and non-volatile memory on the compute nodes. The data-aware management of these resources has to be addressed in the resource manager.

This work is to explore the job scheduling techniques for data-intensive HPC applications with the aids of burst buffer and key-value stores through both real system and simulations. The real system could be implemented as a combination of hardware (e.g. RAM, NVRAM, SSDs) and software (e.g. Slurm++). The Slurm++ resource manager would consider the burst buffer as a secondary category of resource and implement the logic of periodically check-pointing to enable all the compute nodes pushing/pulling job data to/from the burst buffer, which then dumps the data to the parallel file system asynchronously for persistence. We will utilize distributed key-value stores (e.g. ZHT) to manage the job, compute nodes, and burst buffer metadata.

The candidate simulators are the CODES simulation framework from ANL/RPI, and other simulators that we have been developing such as the key-value store simulator, SimMatrix, and SimSLURM++. The questions to answer are whether the current resource manager are scalable towards exascale without the proposed scheduling techniques, and whether the proposed techniques can perform better for high-failure computing systems and data-intensive HPC applications. From the simulations we would gain insights to drive the development of the proposed techniques in Slurm++ and to deploy the resource manager on a real machine.

Relevant Systems and Reading Material

Burst Buffer: http://www.mcs.anl.gov/papers/P2070-0312.pdf
ZHT:
- Paper: assigned in the course webpage
- Source code: https://github.com/mierl/ZHT

SLURM resource manager: http://www.schedmd.com/slurmdocs/slurm.html
Slurm++ resource manager:
- Paper: assigned in the course webpage
- Source code:
  - Slurm++: https://github.com/kwangiit/SLURMPP_V2
  - SimSlurm++: https://github.com/kwangiit/SimSlurmpp

CODES simulation framework: http://www.mcs.anl.gov/research/projects/codes/
Key-Value Store simulation:
- Paper: http://datasys.cs.iit.edu/publications/2013_SC13-KVS.pdf
- Codebase: https://github.com/kwangiit/KVSSim

Methodology

1. Exploring the design choices through the SimSlurm++ and/or CODES simulators.
2. Implementing the data-aware scheduling techniques in the Slurm++ system

Preferred/Required Skills

Required: Linux, C/C++, Java, scripting language, sockets and multi-threading

Metrics

Throughput (jobs/sec), bandwidth (MB/sec), efficiency, latency; for real implementation, the Amazon EC2 testbed should be used with up to 128 VM instances; for simulations, either SimMatrix, SimSLURM++, or CODES/ROSS should be used.

Project Mentor

Ke Wang, http://datasys.cs.iit.edu/~kewang/