CS554 Project Ideas

MATRIX:DJLSys - Exploring Resource Allocation Techniques for Distributed Job Launch under High System Utilization

Overview

Launching large number (millions and beyond) of jobs in an efficient way is critical to the performance of resource management systems deployed on large-scale distributed systems. However, today's state-of-the-art resource managers (e.g. SLURM) have centralized job launching paradigm, where a single server/controller is in charge of the whole process of launching jobs to all the compute nodes, such as resource allocation, job creation, job launching, and returning collection. This centralized architecture has inherent scalability issues. We propose using a fully-connected distributed server architecture which is comprised of multiple controllers with each one managing a partition of compute nodes. Controllers are designed to be able to steal resources from each other. A distributed key-value store (KVS) (e.g. ZHT) is used to keep all the resource and job information for all the controllers, which therefore don't need to communicate with each other for resource allocation. One of the significant problems of distributed job launch is how to allocate resources for jobs under high system utilization where there isn't much idle resource efficiently. This project aims to explore various resource allocation techniques (e.g. totally random selection, asynchronously history-based caching, or global and lazily partial knowledge) for efficient distributed job launching under high system utilization.

Relevant Systems and Reading Material

SLURM resource manager: http://www.schedmd.com/slurmdocs/slurm.html
ZHT paper and source code: http://datasys.cs.iit.edu/projects/ZHT/index.html
Distributed job launch report and source code: https://github.com/kwangiit/dist_job_launch (for the source code, please use the "distjoblaunch_2" source file folder)

Methodology

Working directly with our real implementation of distributed job launch source code

Preferred/Required Skills

Required: Linux, C/C++, scripting language, sockets and multi-threading
Preferred: Git version control and source code management system

Parameters

Different resource allocation algorithms, poll interval and failure times to de-allocate resources

Metrics

Throughput (jobs/sec), efficiency, latency, etc

Project Mentor

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