Smart compression in multi-tiered storage platform

Brief Description: Applications today are highly data intensive. This creates a great stress on existing storage systems. To reduce this stress, data compression techniques were introduced to reduce the data movement footprint in large clusters. This data compression is performed at a cost of compute resources. Hence, researchers have explored the ideal balance between cost of compression and the benefit of compression for various application workloads. Another recent trend is to employ multiple layers of faster storage close to compute cluster to reduce the I/O bottleneck in current large clusters. This new advancement in storage could possible help data compression techniques become more efficient by masking I/O operations for compression, reducing memory footprint, and enabling in-situ data compression opportunities. In this project, students are required integrate our intelligent compression library into a multi-tiered storage platform. This project provides opportunity for students to learn and gain hand-on experience on various compression techniques and storage devices.
Prerequisite: The project will require students to be proficient in C++, OOP concepts, and Linux operating system.
Recommended team-size: 1-2 students.

Automated profiling in multi-tiered storage platform

Brief Description: Modern supercomputers employ multiple fast storage close to compute cluster to reduce the effect of I/O bottleneck. To better understand and quantify supercomputers performance, system architects employ several profilers. Moreover, applications today have highly diverse storage requirements based on which the applications are scheduled on these large supercomputers. The storage requirements are met with allocations in the supercomputer. This creates a dynamic environment for the application to run each time it given an allocation in the system. Hence, running the profilers manually is not a desirable approach. In this project, students should explore and automate several system-level and application-level profilers. This will enable fast, efficient, and accurate run-time optimization for these complex dynamic systems. This project provides opportunity for students gain hand-on experience on system-level tools and have a deeper understanding on the kind of complex applications used in supercomputers.
Prerequisite: The project will require students to be proficient in c++, scripting languages (e.g., bash, python, etc), and compiling Linux softwares.
Recommended team-size: 2-3 students.

Buffer Schema generator for multi-tiered storage platform

Brief Description: Modern supercomputers employ multiple fast storage close to compute cluster to reduce the effect of I/O bottleneck. There have been several multi-tiered storage systems which help applications to transparently and efficiently utilize these devices. One such system is Hermes. It enables multi-tiered buffering for increasing performance of the application. Moreover, applications today have highly diverse buffering requirements. Hence, there is a need for building an efficient buffer schema generator which will enable users to provide their buffering schema. In this project, students need to explore various buffering requirements applications might have, the best way to represent this requirement (which is both minimalistic and descriptive), and finally, build a parser which can understand this schema and efficiently load it into the system. This project provides opportunity for students to learn and gain hand-on experience on high-performance file-formats and great experience in a software design process.
Prerequisite: The project will require students to be proficient in C++, OOP concepts, Linux operating system, and scripting languages (e.g., bash, python, etc).
Recommended team-size: 1-2 students

Interference-aware application scheduler for supercomputers

Brief Description: Modern supercomputers run hundreds of thousands of applications on thousands of nodes. They require complex resource management software to manage the mapping of application on these nodes. Slurm is the most popular resource management software/scheduler. Moreover, modern supercomputers employ several layers of storage hierarchy to reduce the problem of I/O bottleneck. Hence, supercomputers have several shared and node-local storage resource. The shared storage resources, which are used by many applications, suffer from I/O interference. Hence, there is a need for schedulers such as Slurm to become interference-aware. This will maximize the resource utilization and performance of the system. In this project, students will explore existing interference-aware scheduling algorithms and integrate them with a real-scheduler such as Slurm. This projects provides students an opportunity to build an interference-aware module for the Slurm scheduler and an opportunity to work with real supercomputers to validate the module.
Prerequisite: The project will require students to be proficient in C++, OOP concepts, Linux operating system, and scripting languages (e.g., bash, python, etc).
Recommended team-size: 2-3 students

Data streaming in multi-tiered environments

Brief Description: Data streaming is a programming paradigm which supports reactive real-time computation on irregular, potentially infinite, data flow. Many popular platform provide data-streaming framework to easily consume data-streams. Some example of them are Apache Spark, Flink, and Apache Kafka. Apache Kafka is a disk-based stream processing framework. The disk-based approach enables Kafka to allow stream-replay and have a fault-tolerant solution. Even though data-streaming is extremely matured in cloud environment, same cannot be said for HPC environment. Another recent trend, in HPC, is to employ multiple layers of faster storage close to compute cluster to reduce the I/O bottleneck in current large clusters. This new advancement in storage could possible help disk-based data streaming platforms to become more efficient by masking I/O operations, reducing memory footprint, and enabling in-situ data processing opportunities. In this project, students will be developing a multi-tiered module for disk-based stream processing frameworks such as Kafka, Pulsor, etc. This projects provides students an opportunity to learn popular data-streaming platforms and explore the impact of various storage devices on these platforms.

Prerequisite: The project will require students to be proficient in C++, OOP concepts, and Linux operating system.

Recommended team-size: 2-3 students