Creating a Framework for Systematic Benchmarking of High Performance Computing Systems

Michael M. Pozulp, July 2013

Introduction

The Applications Performance & Productivity (APP) Group of the High-End Computing Capability (HECC) Project is tasked with ensuring maximal performance of the supercomputing systems operated at NASA Ames. As part of this task, one focus is to benchmark current and future systems so as to guide users to the most suitable resources for their applications. The goal of this project is to construct a semi-automated test framework for benchmarking.

Benchmarking Workflow

The benchmarking workflow has three essential steps: parameter selection, code execution, and result interpretation (Figure 1). Code compilation must occur before code execution, but may come before *or* after parameter selection, depending on the whether the altered parameters are compile-time or run-time parameters, and also whether the desired executable has already been created in a previous compilation. Further complications include: intractable combinations of parameters and system configurations, issues in organization and persistence of benchmarking results, and problems of determining benchmark performance or computational success.

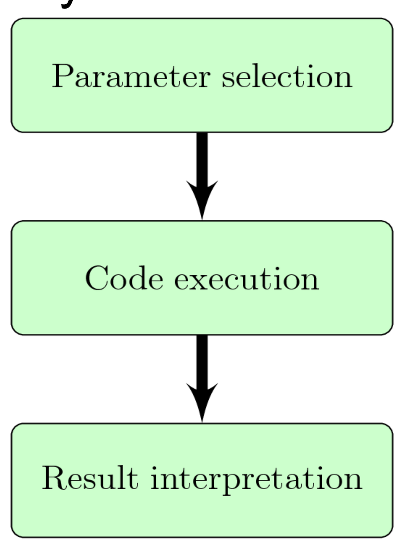

Figure 1. Benchmarking workflow.

Provisioning a Database

The proposed test framework requires a persistent, flexible, scalable, and reliable data storage mechanism (Figure 2) with an interface for efficient insertion and extraction of data records.

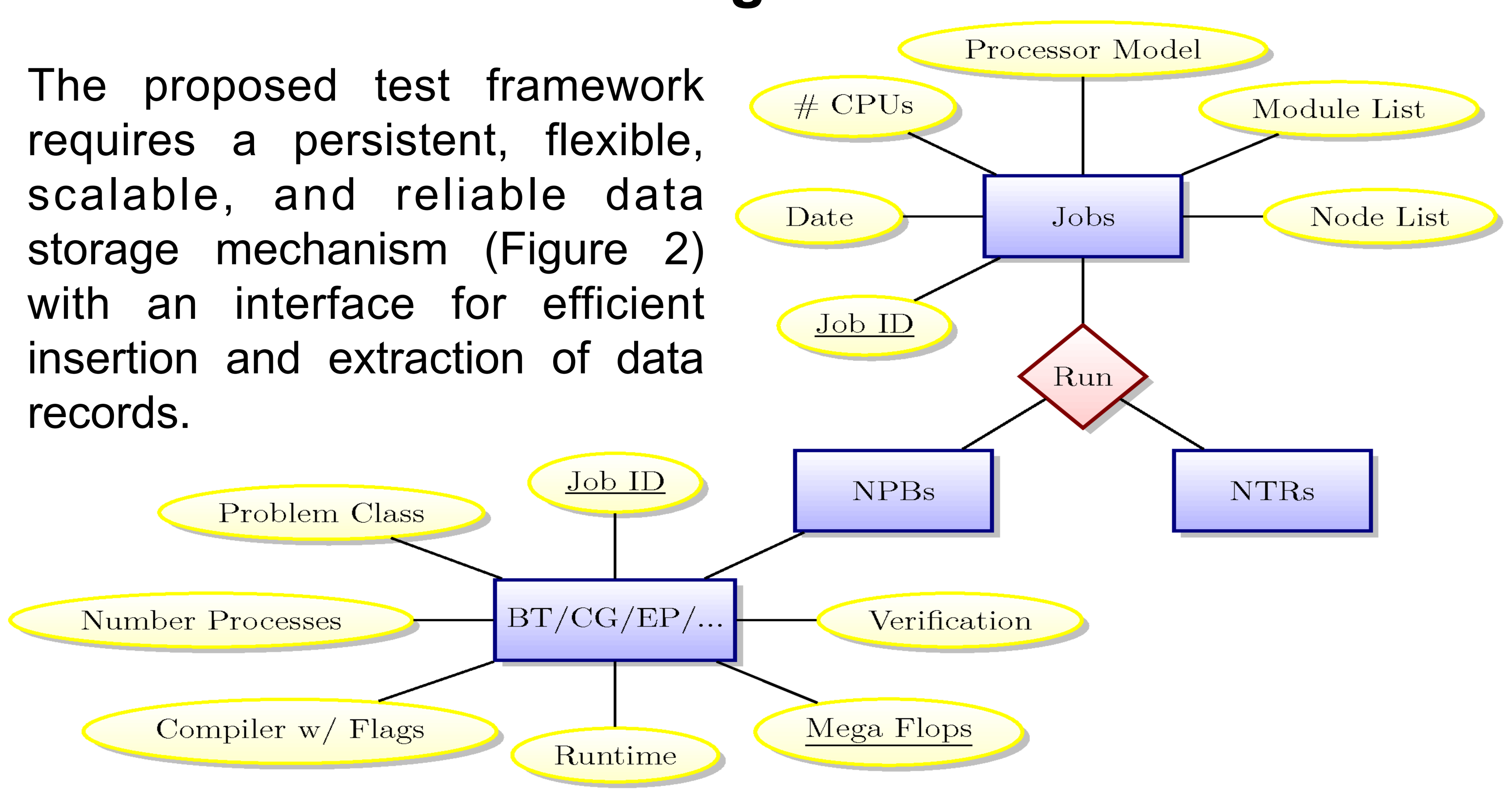

Figure 2. Informal entity-relationship diagram.

In accordance with the relational model, separation of benchmarking parameters from results avoids data duplication and saves on storage.

Implementation Details

A proof-of-concept framework was implemented using Python- and shell-scripting along with a Postgresql database, the scheduling utility cron, and job scheduler PBS Pro. It selects parameters, compiles executables, submits jobs, parses results, inserts/retrieves data, verifies results, and produces graphs of the performance results.

Implementation Results: NAS Parallel Benchmarks (NPBs)

NPB Performance Scaling

An initial proof-of-concept implementation has been utilized to examine parallel performance scaling across the four generations of Intel Xeon processors that compose the Pleiades supercomputer system. From newest to oldest, they are Sandybridge, Westmere, Nehalem, and Harpertown.

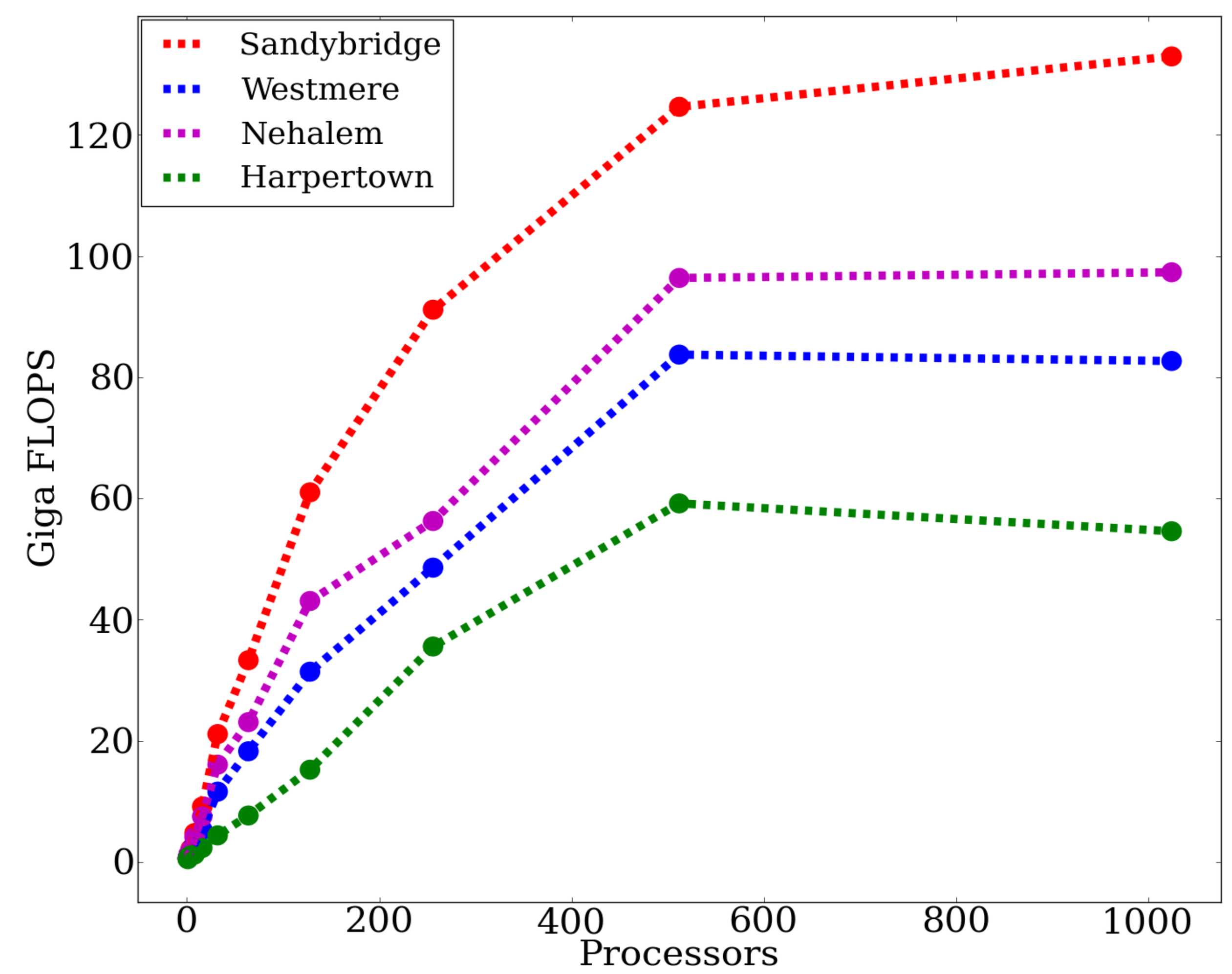

Figure 3. CG benchmark Class C scaling results by processor type.

It was hypothesized that as the number of processors increased toward infinity, communication-intensive applications would hit a network-imposed performance ceiling and nullify any performance gains resultant from hardware improvements in newer generations of Intel Xeon processors. That ceiling was observed simultaneously for all four generations at 512 processes, but the performance did not converge to a common value. Rather, using data collected within the framework, it is apparent that performance remained better for newer generations, and worse for older (Figure 3).

NPB Performance Variation

In addition to performance scaling, the framework has been used to analyze substantial performance variation in the NPB applications run on Pleiades during the NPB performance scaling investigation.

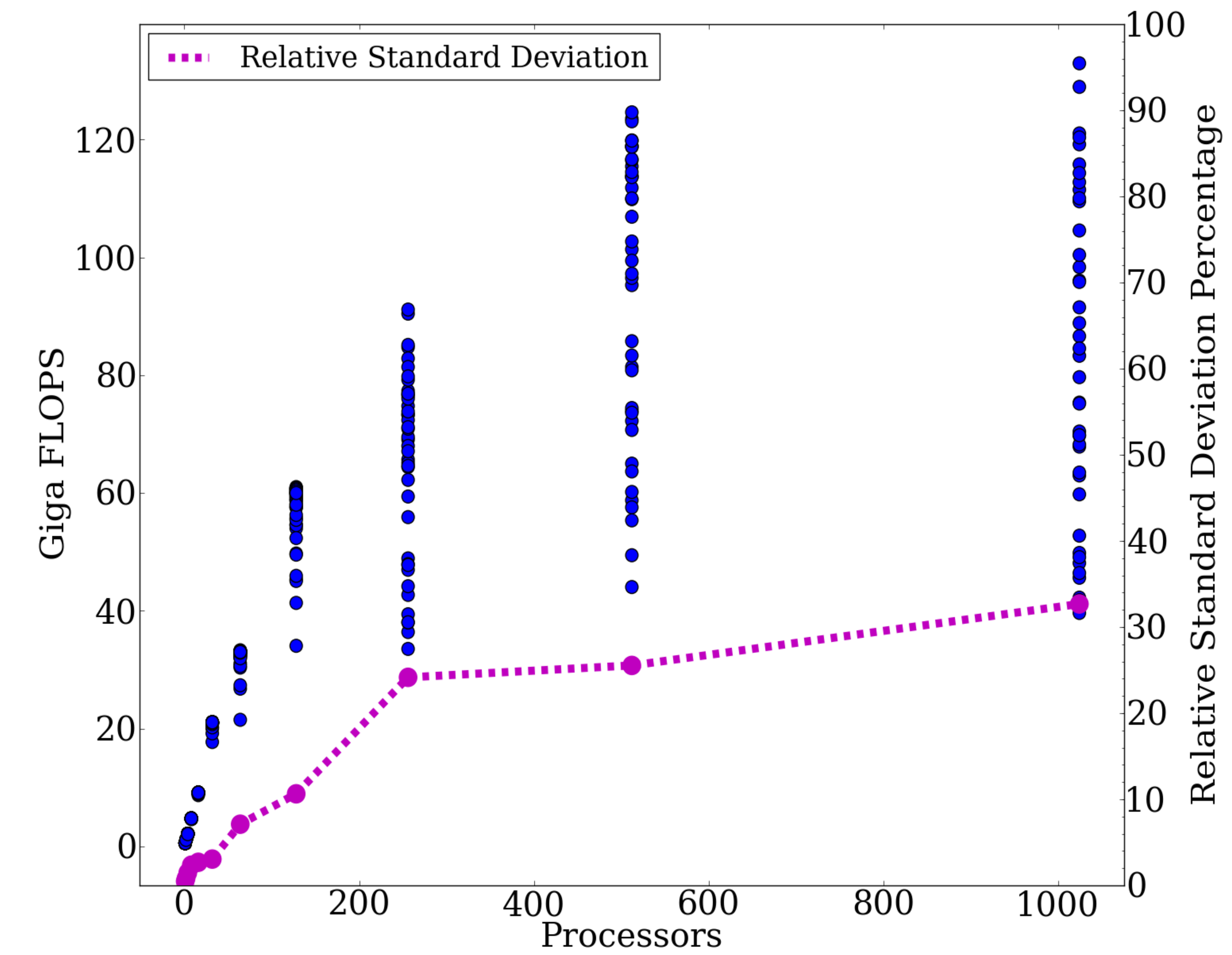

Figure 4. CG benchmark Class C results on Sandybridge processors.

It is hypothesized that (1) for each particular NPB, the extent of the variation corresponds to the communication intensity of the application kernel, and (2) there is a quantifiable correlation between performance degradation and the logical network distance that communicated data must traverse. From data collected within the framework, one may observe substantial relative standard deviation, that is, the ratio of the standard deviation to the average, converted to a percentage, increasing along with the number of processors (Figure 4).

Conclusion

Scientific computing operations running high performance systems must benchmark those systems. The challenge of creating a useful test framework for benchmarking is immense, but so is the benefit. A proof-of-concept implementation has been used to: streamline personal workflow with automation, support team workflow with centralized result sharing, and assist in performance analysis with result visualization capabilities. In running NPBs, it has acquired data necessary to conduct ongoing investigations into NPB performance variation and to examine hypotheses linking communication intensity of application kernels and logical network distances to application performance variation and degradation.

Future Work

Maximal generality of the benchmarking test framework requires sufficient abstraction of its specification in order to capture the common generalities of benchmarking workflows. To address the tremendous variation in specific workflows and benchmarking tasks, future implementations of this framework should provide APIs for user-level customization and scripting. With an API providing the interface, users would be able to mold their workflow to the high-level framework, unifying any dissonant pieces and crafting an automated mechanism for benchmarking. In a sense, a complete system would have the user step away from the details of job scripting and replace it with a new paradigm of framework scripting.

Acknowledgments: Work performed under the expert direction of Mentor and Division Chief Piyush Mehrotra, APP Group members Henry Jin, Bob Hood, Johnny Chang and help from many others in the NASA Advanced Supercomputing (NAS) Division at NASA Ames Research Center.