Improving HPC Development Through Empirically Guided Analysis

Motivation

High performance computing systems are being applied to an increasingly wide variety of domains, including nuclear physics, crash simulation, fluid dynamics, climate modeling and bioinformatics. They are also becoming radically less expensive: the recently announced $1.5 million dollar BlueGene/L supercomputer will execute 24 teraflops per second. In comparison, the former fastest supercomputer, Japan's Earth Simulator, executes 36 teraflops per second but costs over $350 million dollars.

Unfortunately, these numbers do not tell the whole story. The DARPA High Productivity Computing Systems Program [1] and the Workshop on the Roadmap for the Revitalization of High-End Computing [2] note that dramatic increases in low-level HPC benchmarks of processor speed, memory access, and dollars/flop do not necessarily translate into increased development productivity. In other words, while the hardware is clearly getting faster and cheaper, the developer effort required to exploit these advances is becoming prohibitive. Software engineering, not hardware engineering, is becoming the limiting factor in the advance of HPC.

To address these issues, we have pursued collaborative research with SUN Microsystems and MHPCC with two broad goals: (1) The study of HPC development from a software engineering perspective, with the goal of identifying key HPC productivity bottlenecks and subsequent development of new tools or techniques to overcome them; and (2) The development of more comprehensive measures of HPC productivity for use in evaluating new and existing HPC architectures, which take into account not only the low-level hardware components, but the higher-level development costs associated with producing usable HPC applications using the architecture. Our work together so far has included the development of a framework for HPC productivity evaluation that uses "Purpose-Based Benchmarks" [3], and the organization of two workshops on HPC and Software Engineering [4].

Past Accomplishments

With the funding allocated to us during the academic year 2004-2005, we were pleased to accomplish all goals set forth in our application. Specifically, we achieved the following milestones through our research activities:

1. ***Implementation of the Truss Optimizer Benchmark on an MHPCC platform -*** For this research milestone, we developed a complete solution to the Truss Benchmark on the Squall system at MHPCC, making use of the IBM xlf compiler and MPI libraries provided in Squall's environment. The code was written in C++ and is portable between IBM and GNU compilers and UNIX environments. As anticipated, this benchmark provides an advantage over traditional ones, such as LINPACK, as it measures development activities, such as time required to implement the software, in addition to measuring runtime performance.

2. ***Acquisition of Truss Optimizer Benchmark measurements on Squall –*** In parallel with implementation, we were able to collect and analyze data for the Truss Benchmark project. Specifically, we were able gain traditional HPC measurements, such as speedup, and software engineering measurements, such as developer effort, code size and code complexity. We have also generated an informational website [6] documenting and interpreting our results, as well as providing open source code and data.

3. ***Extension of the Hackystat automated software measurement framework to support HPC measurement –*** For this milestone, we implemented a set of tools to collect measurement data particularly important to HPC development. One of the tools was a sensor to automatically collect information about code size and is able to distinguish between sequential and parallel code, such that their sizes could be measured independently. Another tool we created captured and sent C++ unit test data from its invocation to the Hackystat system. All of these tools require minimal developer interaction to install and configure and then collect data automatically with no direct developer intervention.

In addition to meeting these milestones, another rewarding aspect of our work was the acceptance of a paper [7] into the 2005 High-Performance Computer Architecture Conference in San Francisco and the presentation of this paper at the workshop for Productivity and Performance in High-End Computing held in conjunction with the conference. As a result of our participation in the conference, we were able to network with HPC professionals, promote the work performed and facilities offered at MHPCC to attendees and generate interest in our research.

Improving HPC Development Through Empirically Guided Analysis

Proposed Research

Now that we have direct and tangible results from developing the Truss Benchmark in an HPC environment on Squall and have created tools to collect and analyze HPC related data from our experience, we can apply and expand our tools and knowledge to projects at MHPCC and the broader HPC community.

The Information and Computer Sciences department has recently hired Dr. Henri Cassanova to teach software development and conduct research in the field of HPC. This provides a unique opportunity for collaboration between the University of Hawaii and MHPCC, such that students can utilize and promote MHPCC resources for their coursework and assignments and we can conduct empirical studies from a subject population local to the university. This opportunity enables us to apply our tools to other HPC projects, solicit feedback from a significant pool of HPC developers and create new tools or refine older ones to fully capture the HPC development process within the MHPCC environment.

In pursuit of these goals during the 2005-2006 academic year, the proposed funding will be used to support the following research activities.

1. **Implement a framework for evaluating HPC development approaches for MHPCC** – In order to understand HPC productivity, a comparison of the development approaches must be conducted. We propose to extend the Hackystat system to be able to evaluate multiple HPC packages, such as MPI, OpenMP, Co-Array Fortran and UPC. Specifically, we expect to be able to measure cost, in terms of development time, size, in terms of source lines of code and code expansion factor, as a ratio between size of a parallel solution and size of a serial solution.

2. **Enhance HPC tool support for the Hackystat system**– While implementing the Optimal Truss problem, we identified additional metrics and data that are important to HPC development. For example, when executing parallel code, there are many instances in which the code will fail on one or more processors, but not all. We propose to add a new tool to Hackystat to capture runtime data, especially failures, from applications such as mpirun and poe, and automatically classify the failures to assist developers in understanding them. These tools will run in the MHPCC environment.

3. **Categorical identification and classification of HPC bottlenecks** – There are many potential productivity bottlenecks in HPC, many of which constitute unnecessary repetitive actions. For example, frequent editing of IP addresses in machine files typically indicates that a developer is experiencing runtime problems with one or more processors. In our research, we propose to extend Hackystat to capture, classify and quantify bottlenecks, such as machine configuration in addition to many others.

4. **Collaboration between UH and MHPCC to conduct empirical studies of HPC development** – We are excited to collaborate with MHPCC and Dr. Cassanova to conduct empirical studies with students using HPC resources on Squall. Such experimentation is expected to yield empirical data for the HPC community as well as increased exposure to resources available at MHPCC.

At the conclusion of this year of research and development, we expect that the Hackystat/MHPCC technology infrastructure will be mature enough for general deployment and use, making MHPCC a leading center for empirical study of the software engineering of high performance computing applications.

References

[1] DARPA High Productivity Computing Systems Program, http://www.highproductivity.org/
[2] The Roadmap for the Revitalization of High-End Computing, http://www.cra.org/Activities/workshops/nitrd/
[3] S. Faulk, J. Gustafson, P. Johnson, A. Porter, W. Tichy, and L. Votta, *Measuring HPCS Productivity*, International Journal of High Performance Computing and Applications, vol. 18, no. 4, Winter 2004.
[4] International Workshop on Software Engineering for High Performance Computing System Applications, http://csdl.ics.hawaii.edu/se-hpcs/, 2004, 2005.
[5] Hackystat, A Framework for Automated Software Engineering Measurement, http://hackydev.ics.hawaii.edu
[6] Optimal Truss Problem, Specifications, Implementation and Insights, http://csdl.ics.hawaii.edu/Research/Truss
[7] P. Johnson, M. Paulding, *Understanding HPC Development through Automated Process and Product Measurement with Hackystat*, High Performance Computer Architecture Conference, February 2005.