Automating the Run-time Infrastructure for the Performance Evaluation Process

Tara L. McQueen
Delaware State University
Research Alliance in Math and Science
Computer Science and Mathematics Division
Oak Ridge National Laboratory
Mentors: Dr. Stephen L. Scott, Dr. Geoffroy Vallée, Anand Tikotekar

Motivation

- Large effort is required to manually carry out the necessary tasks for the performance evaluation process
- Available tools lack extensive documentation on integrating new benchmark tools
- Effective system management/monitoring is required to ensure that performance evaluations can be carried out
- There is a growing disconnect between performance evaluation software and management/monitoring software
- How can this performance evaluation process be streamlined?

Introduction

- Application performance evaluation
  - Prepare benchmark execution environment
  - Benchmark execution
  - Performance report generation/analysis
- System management/monitoring
  - Allocate resources
  - Node management
  - Job scheduling

System Management - OSCAR

- Open Source Cluster Application Resources (OSCAR)
  - Cluster suite for high performance computing
  - Simplifies install, build, and operation of cluster
  - Cluster management via Graphical User Interface (GUI)
  - OSCAR extended via user contributed packages
- Would be beneficial to have a performance evaluation OSCAR package

Performance Evaluation - Cbench

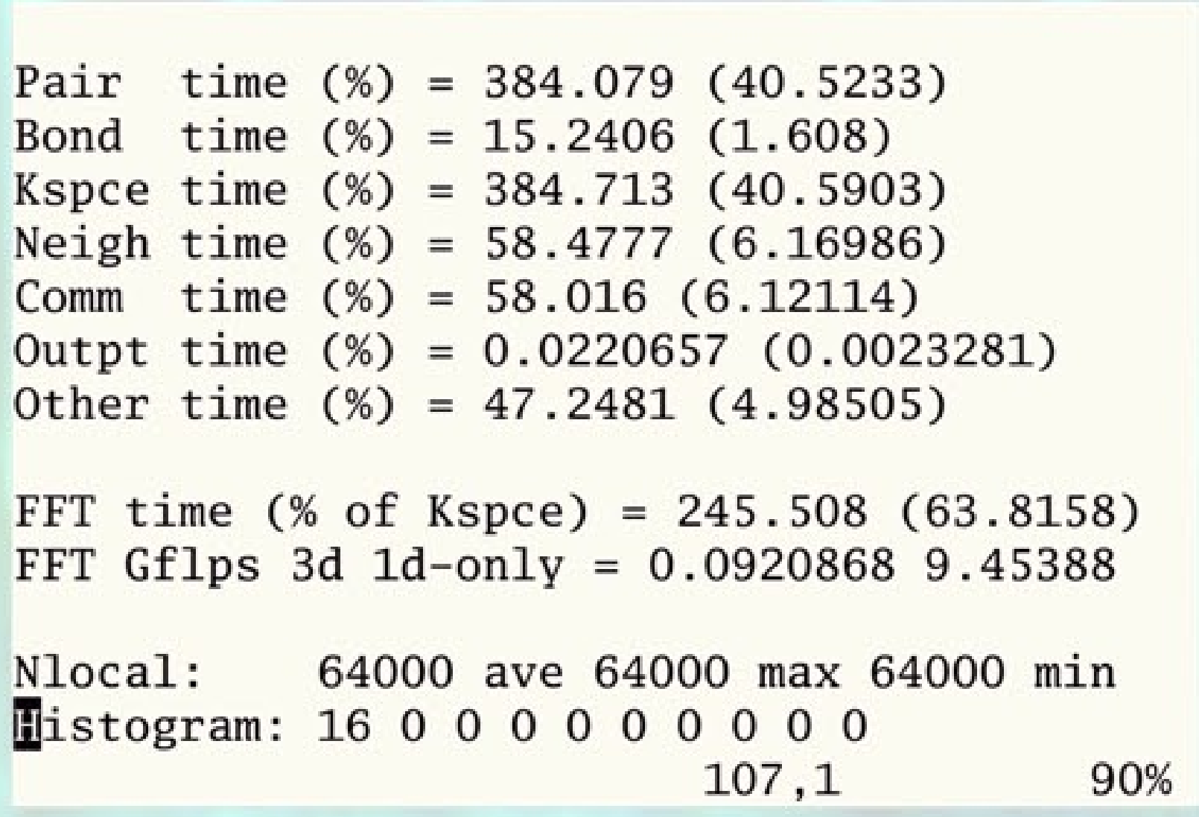
- Extensible cluster benchmarking and testing toolkit
- Collection of tests, benchmarks, applications and utilities
- Currently provided benchmarks/tests
  - HPCC
  - Linpack
  - NPB
- Automates tasks required for performance evaluation
  - Generation/execution of jobs
  - Data parsing
- Provides template scripts for new benchmarks
- Lacks resource allocation, system monitor, and job scheduler
- No extensive documentation for integration of new benchmarks

Goals

- Integrate a benchmark tool (Cbench) and system management framework together (OSCAR)
- Add new benchmark into Cbench and documentation

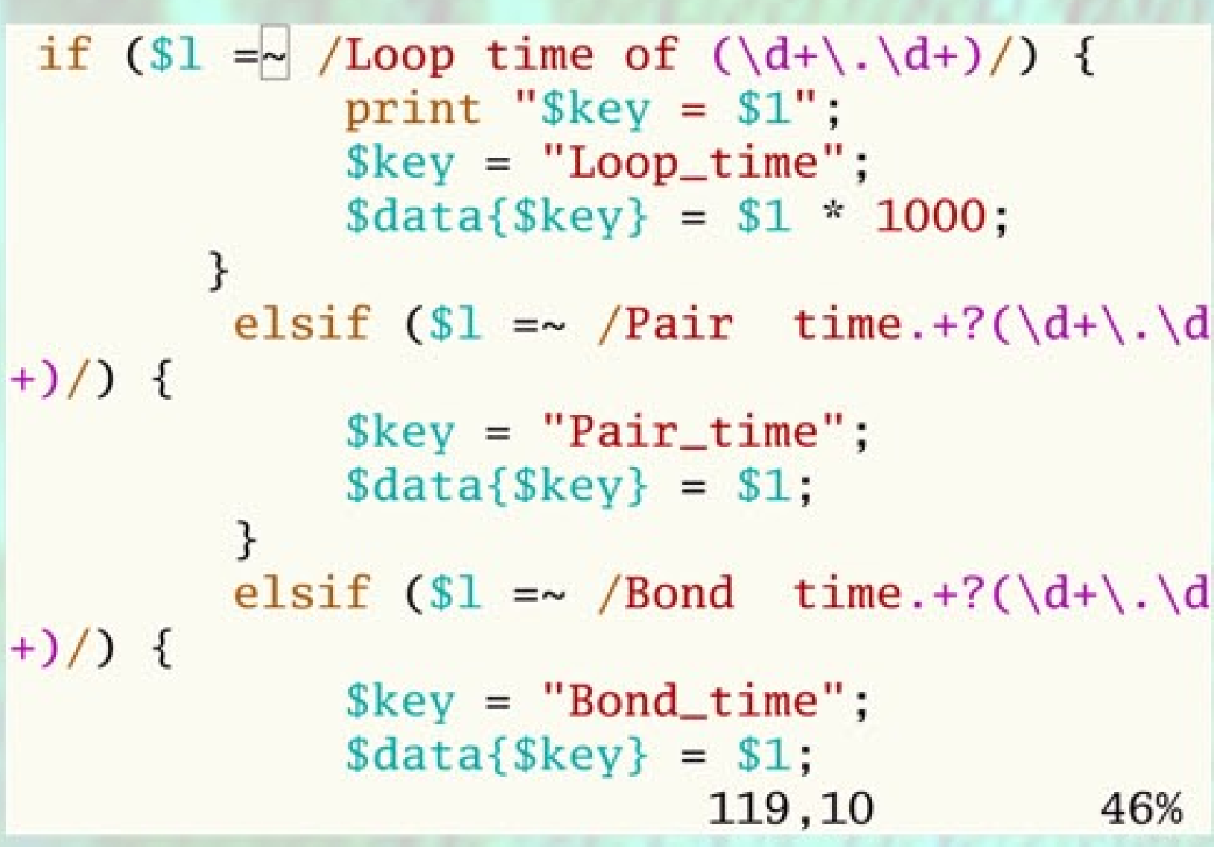
Prototyping

- Selected LAMMPS (Large-scale atomic/molecular massively parallel simulator) for integration into Cbench
- Modified generic template scripts to generate jobs, start jobs and parse data of the LAMMPS benchmark
  - Parser is unique to the specific benchmark
  - Requires additional custom logic
- Created How To guide for instructions of parser development
- Tested on a cluster with 16 nodes
- Required effort is nontrivial



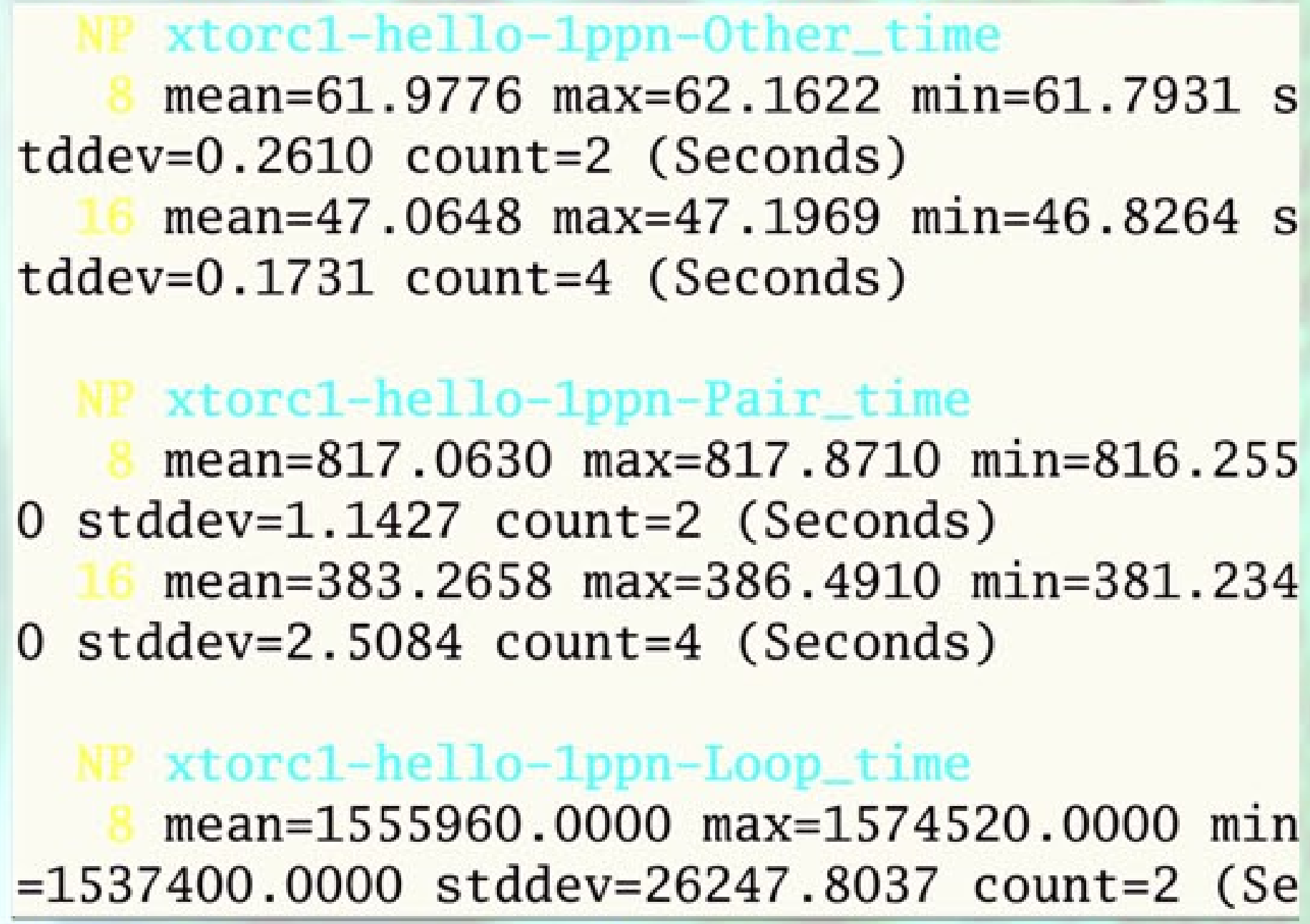
```
Pair  time (%) = 384.079 (40.5233)
Bond  time (%) = 15.2406 (1.608)
Kspce time (%) = 384.713 (40.5903)
Neigh time (%) = 58.4777 (6.16986)
Comm  time (%) = 58.016 (6.12114)
Outpt time (%) = 0.0220657 (0.0023281)
Other time (%) = 47.2481 (4.98505)

FFT time (% of Kspce) = 245.508 (63.8158)
FFT Gflps 3d 1d-only = 0.0920868 9.45388

Nlocal:    64000 ave 64000 max 64000 min
Histogram: 16 0 0 0 0 0 0 0 0 0
                              107,1          90%
```

Fig. 1. log.lammps: data from LAMMPS benchmark before parsing



```
 if ($1 =~ /Loop time of (\d+\.\d+)/) {
                print "$key = $1";
                $key = "Loop_time";
                $data{$key} = $1 * 1000;
           }
            elsif ($1 =~ /Pair  time.+?(\d+\.\d
+)/) {
                $key = "Pair_time";
                $data{$key} = $1;
            }
            elsif ($1 =~ /Bond  time.+?(\d+\.\d
+)/) {
                $key = "Bond_time";
                $data{$key} = $1;
                              119,10         46%
```

Fig. 2. lammps.pm: parse module code for LAMMPS



```
  NP xtorc1-hello-1ppn-Other_time
   8 mean=61.9776 max=62.1622 min=61.7931 s
tddev=0.2610 count=2 (Seconds)
  16 mean=47.0648 max=47.1969 min=46.8264 s
tddev=0.1731 count=4 (Seconds)

  NP xtorc1-hello-1ppn-Pair_time
   8 mean=817.0630 max=817.8710 min=816.255
0 stddev=1.1427 count=2 (Seconds)
  16 mean=383.2658 max=386.4910 min=381.234
0 stddev=2.5084 count=4 (Seconds)

  NP xtorc1-hello-1ppn-Loop_time
   8 mean=1555960.0000 max=1574520.0000 min
=1537400.0000 stddev=26247.8037 count=2 (Se
```

Fig. 3. lammps.output_parse: parsed LAMMPS data in statsmode

Cbench in OSCAR

- Separation of Cbench into core components and benchmarks
- Modification of Cbench Makefiles
- Binary packages using Redhat Package Management (RPM) created:
  - Cbench's core files
  - HPCC (high-performance computing challenge) benchmark
- Cbench now available via OSCAR

Results

- Integration of LAMMPS into Cbench
- Structured How To guide for new benchmark integration into Cbench
- Benchmarking framework for OSCAR
- Provided OSCAR users with adequate scalable testing and benchmarking resources
- Simplified further benchmark integration into OSCAR by documentation

The Research Alliance in Math and Science (RAMS) Program is sponsored by the Office of Advanced Scientific Computing Research, U.S. Department of Energy. The work was performed at the Oak Ridge National Laboratory, which is managed by UT-Battelle, LLC under Contract No. De-AC05-00OR22725. This work has been authored by a contractor of the U.S. Government, accordingly, the U.S. Government retains a nonexclusive, royalty-free license to publish or reproduce the published form of this contribution , or allow others to do so, for U.S. Government purposes.

Oak Ridge National Laboratory
U.S. Department of Energy