



U.S. DEPARTMENT OF
ENERGY



Sandia National Laboratories

"Milestone #4875: Evaluate Application Performance on Advanced Architectures",
Richard Barrett (SNL), David Daniel (LANL), Todd Gamblin (LLNL), Mike Glass
(SNL), Rob Hoekstra (SNL), Louis Howell (LLNL), Christoph Junghans (LANL) Al
McPherson (LANL), and Rob Neely (LLNL)

This milestone was the 2nd in a series of Tri-Lab Co-Design L2 milestones supporting 'Co-Design' efforts in the ASC program. It is a crucial step towards evaluating the effectiveness of proxy applications in exploring code performance on next generation architectures. All three labs evaluated the performance of 2 proxy applications on modern architectures and/or testbeds for pre-production hardware. The results are captured in this document as well as annotated presentations from all 3 laboratories.

Some general outcomes and observations

- Effectively organizing code is the necessary first step in achieving effective use of any computing capability.
- Good scaling of MPI+OpenMP requires limiting the number of OpenMP threads for each MPI rank. However, this induces an increasing memory cost, both from MPI and (potentially) from the application.
- Some applications could be limited by integer instruction rates rather than floating point.
- Abstractions can aid in performance portability.
- Feedback loop to compiler team critical for performance.
- This work has demonstrated the power of collaboration across labs, vendors, universities, and others.
- Proxy applications are valuable for these activities but we must never mistake them for the real thing!

LLNL work focused on three proxy apps: UMT2013, MCB, and AMG2013. UMT is an implementation of deterministic multigroup radiation transport, and can be taken as a representative of the wider class of codes that do their work while stepping through zones of an unstructured mesh. MCB is a Monte Carlo radiation transport

code. MCB loops over particles that move independently but interact with material quantities stored on a mesh. Both of these codes are computation-intensive but they use machine resources in significantly different ways. Both are derived from algorithms used in important production codes. The results for UMT and MCB are the primary material we submit for formal completion of the milestone. The third proxy app, AMG, solves an algebraic multigrid test problem using the *hypre* linear solver library, which is heavily used in applications at LLNL and elsewhere. AMG is more communication-intensive than UMT and MCB, and it also provides a useful confirmation that certain machine characteristics observed with UMT and MCB are actually of broader interest and are not unique to those particular algorithms.

All three of these proxy apps are implemented using MPI+OpenMP. We tested them primarily on Vulcan, a BG/Q machine, part of the Sequoia procurement, and on Cab, a Linux cluster based on Intel Sandy Bridge processors and part of the TLCC2 procurement. All three apps scale very well to large numbers of MPI tasks, though scaling was better on BG/Q than on the TLCC2 machine for reasons we discuss. All three perform well with small numbers of OpenMP threads but less well as the number of threads becomes large. We examine the tradeoffs between MPI and OpenMP performance in some detail.

Large amounts of detailed performance data can be obtained using the various hardware counters on BG/Q. Using this counter data we found, for example, that all three proxy apps use integer instructions more heavily than floating point instructions. For UMT the int-to-float ratio varied between 1 and 3 depending on specific parameters of the test problem, while MCB and AMG used over 90% integer instructions. The integer instruction issue rate is one of the most important constraints on performance for all three proxy apps. Other counter information showed that UMT and one phase of AMG were drawing over half the available memory bandwidth on each BG/Q node. So these apps are not bandwidth limited on BG/Q but could easily become bandwidth limited on a machine with slightly different performance characteristics.

Finally, we show exploratory results from two other sets of tests. We show performance of UMT using the DI-MMAP package to access NVRAM on an Ivy Bridge cluster, and we show tests of all three proxy apps using the MSR library to run on a Sandy Bridge cluster at reduced power.

The context for LANL work is the challenge we face moving our workloads on to future architectures, and specifically for us the imminence of the Trinity ATS system – particularly Trinity phase 2 with its self-hosted Intel Knights Landing (KNL) processors.

SNAP models a discrete ordinates particle transport code, and is designed as a proxy for the LANL production code PARTISN. PENNANT is proxy application for the explicit general unstructured staggered-grid hydrodynamics code FLAG, but focuses

on the basic hydrodynamic operations and data structures without associated complications from multi-material treatments, strength, damage, etc.

For both proxies we made single-node strong-scaling studies on a variety of platforms using a TLCC2 Sandy Bridge node as a baseline. For both SNAP and PENNANT our primary interest has been Intel Knights Corner (KNC) performance, but for PENNANT we also collected data on BlueGene/Q nodes and two generations of NVIDIA GPU's.

Sandia work focused on miniFE and miniAero, and was supplemented by a few others. MiniFE, an implicit finite element solver, has 20 implementations involving 10 programming mechanisms, including Kokkos. Experiences have been incorporated into the Charon electronic simulation code and Trilinos packages. Memory bandwidth bound, different core counts did not meaningfully result in different performance, though memory usage was significantly altered. Architecture-specific data organization, such as an ELL-pack sparse matrix storage format, can improve performance.

MiniAero, a new miniapp designed for exploring aerodynamics, was designed and developed to use Kokkos. An atomics-based implementation was configured, in comparison with a "gather-sum" strategy. The latter performed best, but at a cost of a memory increase. Performance results are inline with our understanding of the different architectures. This work is informing the development of an exascale-capable aerodynamics application.

MiniGhost was configured to demonstrate the value of an integrated MPI + X tasking model. Over-decomposing the domain exposed more parallelism, enabling communication overlap with little intrusion into existing coding styles. MiniMD work resulted in tech transfer to the LAMMPS molecular dynamics package. MiniAMR was developed to illustrate the complexities and impact of different programming strategies for mesh refinement. The developing MiniContact, a solid mechanics miniapp based on Kokkos, is our most ambitious miniapp in terms of the complexity of the computation. An alternative to the traditional global search algorithm is showing promise as a viable alternative, especially on new architectures.

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.