Title: Status of LANL Efforts to Effectively Use Sequoia

Author(s): Nystrom, William David

Intended for: Report
Web

Issued: 2015-05-14

Disclaimer:

Los Alamos National Laboratory, an affirmative action/equal opportunity employer,is operated by the Los Alamos National Security, LLC for the National NuclearSecurity Administration of the U.S. Department of Energy under contract DE-AC52-06NA25396. By approving this article, the publisher recognizes that the U.S. Government retains nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or to allow others to do so, for U.S. Government purposes. Los Alamos National Laboratory requests that the publisher identify this article as work performed under the auspices of the U.S. Departmentof Energy. Los Alamos National Laboratory strongly supports academic freedom and a researcher's right to publish; as an institution, however, the Laboratory does not endorse the viewpoint of a publication or guarantee its technical correctness.

Status of LANL Efforts to Effectively Use Sequoia

Dave Nystrom

UNCLASSIFIED

Overview

- Work in progress on 3 LANL production applications
- VPIC – a 3D relativistic, electromagnetic Particle-In-Cell code for plasma simulation
- xRage – a 3D AMR mesh, multi-physics hydro code
- Pagosa – a 3D structured mesh, multi-physics hydro code

VPIC Issues

UNCLASSIFIED

VPIC

- VPIC was designed to be efficient on modern architectures and performed well on Roadrunner, achieving sustained 30 percent of theoretical double precision max
- Specially designed to use single precision
- Used Altivec floating point intrinsic functions to achieve good vector performance via a C++ wrapper implementation

VPIC (cont)

- Can use either flat MPI or MPI + Pthreads
- Experimented with both flat MPI using up to 64 ranks per node and MPI + Pthreads using 16 ranks per node and up to 3 threads/core
- MPI + Pthreads with 3 threads/core is about 50 percent faster than flat MPI with 64 ranks/node
- Initial focus on single node performance

VPIC (cont)

- Wrote a C++ wrapper class to use QPX vector intrinsic functions
- Currently results in about a 2x speedup over a portable implementation of wrapper class that does not use intrinsic functions
- More challenging than for Roadrunner because BGQ does not support single precision

VPIC (cont)

- Hopefully, QPX wrapper class still can be improved
- VPIC single node performance is currently about 30 Gflops or about 15 percent of theoretical peak double precision performance

VPIC Status on Sequoia

- Need to integrate and test QPX wrapper class in production version of VPIC
- Need to test VPIC at scale – so far, focus has been on single node performance improvement
- Testing at scale will likely produce its own set of issues like MPI and IO performance
- Suspended work on VPIC for EAP Code 1

xRage Issues

Execution Problem

- Have been working on multiple Sequoia issues for xRage since November, 2014
- Started initially with a problem being able to restart from a dump file on a 20000 PE job
- Initial investigation showed presence of Nans in dump file – used runs on Sequoia and Cielo
- Subsequent investigation using debug executable 1 results in a core dump

Execution Problem (cont)

- Use of Bob Walkup’s allstacks program generates a stack trace
- Running a different problem setup from another user with 32K PEs but with debug executable 1 produces identical core dump and stack trace
- In both cases, core dump occurs before any restart dump files are generated and stack trace points to algorithmic code

Execution Problem (cont)

- Using debug executable 2, an earlier version of the code currently being used in CCC8, results in a core dump with a different stack trace pointing to code used to write HDF files
- Finally, turning off HDF dumps and running again with debug executable 2 results in yet another core dump with a stack trace pointing to code in a physics package

Execution Problem (cont)

- At this point, have 3 core dump scenarios with 2 different versions of code
- Should be able to use bisection to identify commit which introduced earliest core dump
- Hopefully, this will greatly aid the debugging process to fix problem causing earliest core dump
- No attempt yet to debug other 2 core dump problems

Fortran Compiler Issues

- Another significant problem category that xRage has experienced is IBM Fortran compiler issues
- Shortly after beginning work on the execution problem, a segfault of the IBM Fortran compiler was experienced during a build
- Working with the LLNL consultants, this problem was identified and fixed by the IBM compiler team

UNCLASSIFIED

Fortran Compiler Issues (cont)

- The fully tested version of the fixed compiler suite was recently installed as default on the LLNL BGQ machines
- Another problem which has been experienced multiple times by the EAP Team with this code involves an “Internal Compiler Error” when a translation unit contains too many “use module, only xxx” statements

Fortran Compiler Issues (cont)

- The simple fix is to remove the “only” qualifier on the “use module” statement
- But using the “only” qualifier is legal and good programming practice
- A copy of the Export Controlled source code that reproduces this problem has been saved
- The AR Team will attempt to make a reproducer from this source code that can be given to the IBM compiler team

Fortran Compiler Issues (cont)

- Another Fortran compiler issue encountered involved different behavior between the Intel compiler and the IBM compiler regarding “intent (out)” statements
- It appears that an argument labeled as “intent (out)” gets initialized to zero on entry to a subroutine
- If the variable then gets read by mistake problems can occur

Fortran Compiler Issues (cont)

- This problem was encountered when a subroutine argument that was originally a Fortran array was changed to a Fortran type containing a pointer that was being read
- With the Intel compiler, the Fortran type had been properly initialized and the code worked on Cielo and TLCC machines because the value of the type was not reset upon subroutine entry

Fortran Compiler Issues (cont)

- On Sequoia, the code segfaulted because the pointer had been set to zero
- Would be nice if there was an “flint” tool that could detect these types of errors – or if the IBM compiler could detect the error at compile time
- There are several thousand uses of “intent (out)” in xRage – are there more instances of this type of error lurking?
- Need to fix compiler problems – will be using for Sierra as well

Regression Test Issues

- Extensive work to refactor and reorganize xRage is being done in open
- Source code repository lives in open and gets pushed to secure to run on ASC machines like Cielo and Sequoia
- Code changes get tested in open via nightly regression tests on machines supporting Intel compilers like Cielito and TLCC machines

Regression Test Issues (cont)

- But code changes do not get tested in open on rzuseq because till recently connectivity issues made it difficult to easily ssh from LANL to rzuseq and then checkout repository on rzuseq from a LANL machine
- Hopefully this will change in future but there is much work to do to achieve parity with other machines supporting Intel compilers

Regression Test Issues (cont)

- As a result, currently nightly regression tests almost never run on Sequoia because of build failures
- Builds fail because of problems in the IBM Fortran compiler and differences between the Fortran compilers for Intel and IBM in terms of Fortran syntax accepted
- Failure to regularly run nightly regression tests allows execution bugs to creep into code that often wind up being debugged at scale

Regression Test Issues (cont)

- Near term goal is to have nightly regression tests running regularly on rzuseq and have developers fix BGQ problems in open rather than in secure on Sequoia
- Finally, when the nightly regression tests do run on Sequoia, they are mostly successful. Fixing the compiler issues and eliminating the other barriers to these tests running regularly could be a big help in the EAP Team and their users being able to successfully use Sequoia.

Performance Issues

- Have recently been evaluating xRage performance on 4K nodes using different values for ppn i.e. 8, 16, 32, 64
- A major problem with code is very significant fixed memory footprint per rank for replicated data such as EOS tables
- This problem can be mitigated on some of our LANL machines by using shared memory techniques like KSM and SYSV Shared Memory POSIX support to have only one copy of tables on a node

Performance Issues (cont)

- Developer of this capability has indicated that Sequoia does not have a working SYSV shared memory – but perhaps could use POSIX mmap
- We need to research this topic for Sequoia and learn how to develop this capability – because for now, real runs are limited to 8 ppn
- But with special adjustments to the problem input files, it has been possible perform test runs on 4K nodes with ppn values of 8, 16, 32

Performance Issues (cont)

- Initial results have shown some subroutines which do not scale for the case of increasing the PEs on a fixed number of Sequoia nodes
- Need to investigate these subroutines which do not scale, understand why and fix

xRage Summary

- Need to fix currently know bugs, 3, and demonstrate some problems that run correctly on Sequoia as compared to other LANL machines
- Need to continue pushing for better IBM Fortran compiler with known issues fixed
- Need to make effective use of rzuseq for nightly regression tests

xRage Summary (cont)

- Need to reduce the fixed memory footprint required per MPI rank
- Need to fix scalability issues for subroutines which do not scale as PE count increases for a fixed number of nodes
- Need to be able to efficiently run on Sequoia with ppn of 32 and 64 – currently stuck at 8

xRage Summary (cont)

- Finally, there is a LANL user, Brian Haines, who is successfully running problems at scale on Sequoia as a part of CCC8 and finds Sequoia a very important resource for his work.

Pagosa Issues

Pagosa Summary

- Not a code that I have been very involved with yet
- This code has had some success running smaller jobs on Sequoia
- The latest upgrades to IBM MPI libraries seem to have helped on a very large job, but still in progress
- Need to see successful job completion including post-processing and analysis – also more jobs
- This code currently uses 16 ppn – need to try runs with ppn values of 32 and 64 – maybe 2x speedup

Overall Summary

- xRage and Pagosa are getting useful work done on Sequoia – but it is challenging
- Making steady progress on fixing issues with 3 major LANL applications codes to help them run more effectively on Sequoia
- Several issues remain to be resolved before production work can proceed smoothly – but that is the goal

Overall Summary (cont)

- Work we are doing to support running our applications on Sequoia seems very relevant and useful as preparation for the second phase of Trinity with KNL

Thanks

- Thanks to LLNL consulting team, especially John Gyllenhaal for work on resolving compiler issues
- Thanks to IBM compiler team for rapid response on fixing an identified problem with compiler
- Thanks to Bob Walkup of IBM for help using QPX hardware intrinsics
- Thanks for additional EAP project disk space

Thanks (cont)

- Thanks for help from Brian Albright and Ben Bergen on learning how to build and run VPIC
- Thanks to various members of the EAP Team and their users for help in building and running their codes, providing input setups, resolving build problems, providing insight into various aspects of their codes, especially Mike McKay, Marcus Daniels, John Grove, Lee Ankeny, Brian Haines, Chuck Wingate

Areas for Help

- We could use help developing a path forward to using shared memory on a node to reduce the fixed memory footprint per MPI rank for replicated data for xRage on Sequoia
- Is KSM the best approach? Will that work with BGQ? Is mmap the best approach for BGQ? Other options?
- Need continued help improving IBM Fortran compiler

Areas for Help (cont)

- Any help identifying and evaluating static analysis tools for Fortran could be very useful
- Could such a tool identify improper usage of “intent(out)” statements?
- We may need help getting EAP infrastructure in place on rzuseq

Questions?