

Improving the I/O Performance and Memory Usage of the Xolotl Cluster Dynamics Simulator

Philip C. Roth
Oak Ridge National Laboratory
rothpc@ornl.gov

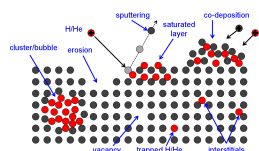
Sophie Blondel
University of Tennessee-Knoxville
sblondel@utk.edu

David E. Bernholdt
Oak Ridge National Laboratory
bernholdtde@ornl.gov

Brian D. Wirth
University of Tennessee-Knoxville and
Oak Ridge National Laboratory
bdwirth@utk.edu

Xolotl [1]

- Pronounced “show-low-tul”
- Cluster dynamics simulator used to predict gas bubble evolution in solids

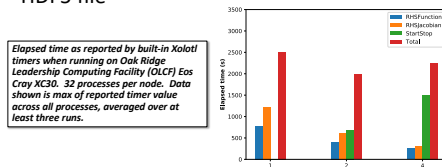


- Solves Advection-Diffusion-Reaction cluster dynamics equations with incident flux

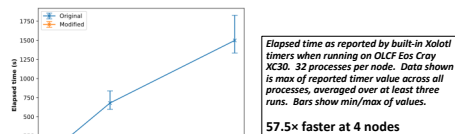
$$\delta_t \bar{C} = \phi \cdot \rho + D \nabla^2 \bar{C} - \nabla \bar{v} C - \bar{Q}(\bar{C})$$
- Used in two U.S. Department of Energy (DOE) Scientific Discovery through Advanced Computing (SciDAC) projects
 - Plasma Surface Interactions (PSI) [2]: simulate He/H bubble formation near surface of diverter in fusion reactor
 - Fission Gas Simulation (FGS) [3]: simulate Xe bubble formation in nuclear fuel of fission reactor
- Used in coupled-code simulations for DOE Office of Fusion Energy Sciences FY2018 Plasma Materials Interactions (PMI) Theory Milestones
 - Many concurrent 1D instances, each with one process per core on 6 nodes
- One focus of collaboration between PSI, FGS projects with SciDAC Institute for Computer Science and Data (RAPIDS) [4]
- C++, MPI, PETSc solver, HDF5, Boost
- Has built-in performance data collection infrastructure (timers, event counters, hardware counters)

I/O

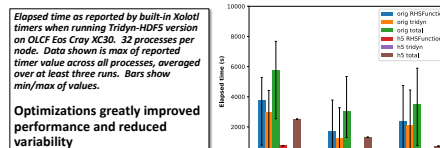
- During PMI Theory Milestone experiments, built-in timers showed troubling trend in startStop1D() PETSc monitor function
 - startStop1D() called every time step
 - Some time steps it writes checkpoint data to an HDF5 file



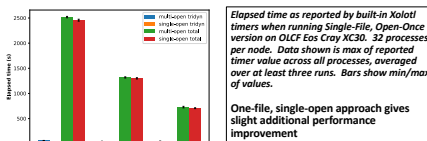
- Profiling with HPCToolkit [5] revealed expensive MPI_Allreduce() in startStop1D
- Code inspection showed misuse of HDF5 API
 - Code transferred all data to all processes
 - All processes wrote entire dataset
 - MPI_Allreduce() on one integer, for each grid point – highly sensitive to communication latency
- Changing to use true parallel write yielded substantial performance and performance variability improvement in startStop1D



- Built-in timers also revealed performance/variability problem with tridyn() PETSc monitor function
- Like startStop1D(), tridyn()
 - Was called every time step
 - Wrote file used later in workflow
 - Used MPI collectives on small data for every grid point
- Converting tridyn() monitor to use true parallel writes via HDF5 API greatly improved performance and performance variability



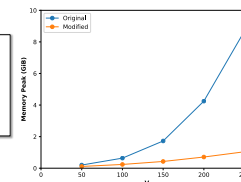
- startStop1D() and tridyn() still opened/created and closed checkpoint files every time they wrote data
- Changing to open files once at program initialization and keep open during run yields minor additional performance improvement



Memory Usage

- Xolotl exhibited out-of-memory errors with smaller-than-expected problem sizes
- Using Valgrind Massif memory profiler [6], determined peak memory usage occurred when describing item coupling during PETSc solver initialization
 - Required constructing two large 2D matrices (>10K items per dimension) of Boolean data
 - Very few non-zeros in Xolotl case
- PETSc converts dense-form matrices to sparse representation
- Proposed alternative PETSc API to take sparse form; accepted to PETSc repository
- Significant reduction in peak memory usage (up to 88% per node)

Xolotl peak memory usage as reported by Massif as reaction network size increases. V is number of vacancies in maximum-sized cluster. Maximum He is V x 4. Before modification, V=200 would not run on OLCF Eos Cray XC30 node. After modification, V>300 is possible.



References

- <https://github.com/ORNLFusion/xolotl>
- <https://collab.cels.anl.gov/display/PSISciDAC2/Plasma+Surface+Interactions+2>
- <https://collab.cels.anl.gov/display/FissionGasSciDAC2>
- <https://rapids.lbl.gov>
- L. Adhianto, S. Banerjee et al., “HPCToolkit: Tools for Performance Analysis of Optimized Parallel Programs,” Concurrency and Computation: Practice and Experience 22(6):685-701, 2010
- N. Nethercote and J. Seward, “Valgrind: A Framework for Heavyweight Dynamic Binary Instrumentation,” Proceedings of the 28th ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI’07), San Diego, California USA, June 2007