Lazy, Minimal, Eventually Consistent IO with Stitch

Jay Lofstead *Sandia National Labs* John Mitchell *Sandia National Labs*

***Abstract*—Existing IO libraries typically write and maintain the entire simulation domain for application IO. In some applications this is both unnecessary and wasteful. For example, in additive manufacturing (AM) [1] and welding simulations [2], only a small part of the simulation domain is affected at each time step. To address this problem, we have developed a new approach to IO called Stitch – saving both time and space. *Stitch* is an efficient IO API and storage format that enables out-of-core computations and merges outputs written over time to construct the full simulation domain. In essence, *stitch* builds the simulation domain analogously to the way an additive manufactured (AM) part is built.**

I. Introduction

The *spparks* [3] kinetic Monte Carlo simulation has successfully demonstrated welding and AM. In these applications, a laser heat source melts material very locally leaving most of the part and simulation domain unchanged.

While welding and AM was our first motivating examples, finite element codes in particular can have similar characteristics for some problems. For example, simulating the pressure wave in a rod that is struck at one end need only consider the wavefront region rather than the entire rod. This generalized example prompts us to look further at this approach to address a wider variety of simulation problems.

II. Design

Our initial insight was twofold. First, the requirement to know the entire simulation domain is unnecessarily restrictive. Yes, it can offer optimizations in data layout in a custom file format, but we were willing to ignore such constraints. Second, a lesson from *ADIOS* is that for reading performance, reassembling a coherent block from pieces can be much more efficient than forming a single large block at write time and slicing at read time.

A. Simulation Domain Ignorance

Stitch eliminates managing the global size entirely and instead focuses on each IO data block spatial extent and relies on the user to understand the context for the extent with respect to the overall domain.

B. Data Reassembly

Stitch relies on data reassembly to return a requested data region at a given time. However, using *Stitch* it is possible that no data exists in some part of the requested region and/or that data from older time steps are the newest data available. *Stitch* initializes non-existent values on portions of the requested block that may not exist and aggregates slices from older time steps to form the requested region; for preexisting extent, *Stitch* returns state for the most recent time(s) less than or equal to requested time.

III. Implementation

As with any IO library, the API and file format are of particular importance and interest.

A. API

For ease of use, testing, and to encourage adoption by scientists and engineers, *Stitch* offers a fully native Python module. In *Stitch* demonstrations, the *Stitch* library is linked with *Spparks*, a C++ code, which uses the C API to dump *Stitch* files that are subsequently post processed for analysis and image creation using the python interface.

B. File Format

To avoid creation of yet another file format, and to take advantage of data selection and sorting features, SQLite is used. This offers numerous advantages beyond simply offering a file format.

First, tools like Python have native SQLite interfaces so they can directly inspect the Stitch file format for debugging or direct data access (not recommended, but possible). Second, SQLite is a resilient file format offering recovery in the case of a crash without data loss. Third, SQL is a convenient way to select data in a region, order it appropriately, and create a work queue for processing and selecting data into the return buffer.

IV. Initial Results

Detailed results are too much to fit in this space. In essence, *Stitch* nominally enables the same wall clock time with a small fraction of compute resources and storage requirements.

A full conference paper focused on Stitch as an IO library is in preparation for a Q1 2019 submission. A subsequent journal paper focused on the materials science impacts will be prepared shortly after that.

Acknowledgements
Sandia National Laboratories is a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International, Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525. We are grateful to the following Sandia National Laboratory programs for funding this work: 1) ASC Physics and Engineering Models (P&EM) Advanced Manufacturing, Materials Performance and Solid Mechanics, 2) Integrated Modeling and Applications, and 3) Advanced Certification Program. This work was also supported in part under the U.S. Department of Energy National Nuclear Security Agency ATDM project funding and the U.S. Department of Energy Office of Science, under the SSIO grant series, SIRIUS project, program manager Lucy Nowell.

References

[1] T. M. Rodgers, J. D. Madison, and V. Tikare, "Simulation of metal additive manufacturing microstructures using kinetic Monte Carlo," *Computational Materials Science*, vol. 135, pp. 78–89, 2017.

[2] T. M. Rodgers, J. A. Mitchell, and V. Tikare, "A Monte Carlo model for 3D grain evolution during welding," *Modelling and Simulation in Materials Science and Engineering*, vol. 25, no. 6, p. 064006, 2017.

[3] S. Plimpton, A. Thompson, and A. Slepoy, "SPPARKS," http://spparks.sandia.gov/index.html, 2009.