Center for Exascale Radiation Transport

Project Overview and Status
Jim E. Morel, PI

ASC PI Meeting, Monterey, CA, February 11, 2015
Organization

- CERT Overview
- Project Components and Integration
- Status of Experiments, Simulations, and UQ
- Transport Algorithms
- Computer Science Algorithms
- Lessons learned
CERT Overview

• Overarching application is radiative transfer (RT) in the high-energy-density physics (HEDP) regime
• Overarching goal is increased predictive capability for RT
• We use neutron experiments as surrogates for thermal radiation experiments
  o RT and neutron-transport numerics are similar, so methods developed for one largely apply to the other
  o RT HEDP experiments require multiphysics modeling with many sources of error
  o Inability to tie errors to specific sources has plagued multiphysics projects in the past
  o Boltzmann equation is essentially exact for neutrons, which enables us to use hierarchical UQ to infer numerical errors in our simulations
  o Graphite impurity model is the only calibrated model across all levels of our hierarchy
CERT Overview

• We begin with a single block of graphite, a Am-Be source of neutrons, and a single detector.
• As we move up the hierarchy, the geometry becomes increasingly complicated with channels, barriers, and slits within the graphite; the source changes to a pulsed neutron generator, and multiple detectors are used.
Project Components and Integration

- Improved Predictive Capability for RT
- UQ
- Subgrid Models
- Numerics
- Transport Algorithms
- Comp Sci Algorithms
- PDT
- TAXI
- STAPL
- Experiments
- Simulations
- Verification appears in all components
Status of Experiments

• Neutron generator ordered in November 2014.
  - 1 μs pulses, \(10^6\) DT neutrons per pulse, up to 2000 Hz
  - Being built to order - delivery expected May 2015

• “First Year Measurement” is to calibrate “Impurity Model 1” (IM1):
  - Main effect of impurities is absorption of low-energy neutrons
  - IM1: models impurities in each block as an “equivalent” boron concentration

• Initial IM1 experimental design complete. Experiments underway.
  - Begin with random selection of several graphite blocks
  - Four measurements per block (left, center, right, 90 degrees rotated)
  - ~One hour of counting per measurement (<1% statistics)
  - Estimate completion in 2-4 weeks

• Preliminary experiments:
  - Negligible background even when nearby reactor is operating
  - Experimental setup eliminates room-scattered neutrons
  - Calculated sensitivity to impurity concentration is sufficient for accurate calibration
  - Experiments (boric acid in water) underway to experimentally quantify sensitivity
IM1 Experiment
Status of Experiments

• Neutron generator ordered in November 2014.
  o 1 μs pulses, $10^6$ DT neutrons per pulse, up to 2000 Hz
  o Being built to order - delivery expected May 2015

• “First Year Measurement” is to calibrate “Impurity Model 1” (IM1):
  o Main effect of impurities is absorption of low-energy neutrons
  o IM1: models impurities in each block as an “equivalent” boron concentration

• Initial IM1 experimental design complete. Experiments underway.
  o Begin with random selection of several graphite blocks
  o Four measurements per block (left, center, right, 90 degrees rotated)
  o ~One hour of counting per measurement (<1% statistics)
  o Estimate completion in 2-4 weeks

• Preliminary experiments:
  o Negligible background even when nearby reactor is operating
  o Experimental setup eliminates room-scattered neutrons
  o Calculated sensitivity to impurity concentration is sufficient for accurate calibration
  o Experiments (boric acid in water) underway to experimentally quantify sensitivity
Status of Simulations

- Many simulations have been completed for IM1 (MCNP and PDT)
- Many were in error for various reasons. This is being remedied (see lessons learned).
- Once simulation parameters are determined and verified, the IM1 run-set should take $< 5 \times 10^4$ core-hours (e.g., 50 hours on 1000 cores) on a TLCC machine (e.g., cab at LLNL)
Status of UQ

• UQ software is in place for analyzing IM1 data.
• Software has been exercised with simulation data, but not yet with experimental data.
• It will take less than a week to perform the calibration calculations and interpret the results.
• Another 1-2 weeks may be required to finalize the IM1 model.
• We expect to have the IM1 model calibrated before our review in April.
Transport Algorithms

• Scalable parallel transport requires:
  o Parallel algorithms that scale well for a single iteration
  o Iterations counts that do not increase with resolution
  
  Sweep-based methods satisfy these requirements

• Components of an optimal sweep algorithm:
  1. Optimal scheduling algorithm that executes sweeps in the fewest possible stages
     • Provably optimal algorithm introduced at M&C 2013 (M. P. Adams et al.)
     • PDT results demonstrate minimum stage count
  2. Performance model that estimates solve time as a function of problem attributes, machine parameters, partitioning, and aggregation
  3. Algorithm to select partitioning and aggregation parameters that minimize solve time

• We have implemented an optimal sweep algorithm in PDT
  o We have shown that PDT can scale well to $O(10^6)$ cores
PDT/STAPL Scaling

PDT Parallel Efficiency vs. Core Count, Weak Scaling, IBM BG/Q

Parallel Efficiency vs. Core Count

- Performance model
- Model w/extra overhead
- PDT results

Cores: 8, 32, 128, 512, 2,048, 8,192, 32,768, 131,072, 524,288
Transport Algorithms: Recent Progress

• **Volumetric/Overloaded Partitioning**
  o Used at LLNL in past (studied by Bailey and Falgout - M&C 2009)
  o We developed “overloaded” partitioning for which we created an optimal scheduler
  o In some problems it has higher parallel efficiency than contiguous partitioning

• **Reflecting-boundary algorithm**
  o Exploits partitioning and scheduling features such that P/N processors on 1/Nth domain gives same performance as P processors on full domain (octant symmetry)
  o Implemented in PDT

• **Improved performance modeling**
  o Better represents actual implementation; predicts run time per sweep, not just efficiency
  o Continues to allow inference of where code needs work and performance improvements
  o Most constants obtained from fit to single-core performance data

• **Arbitrary grids**
  o Cell-sets are unions of rectangular/brick regions, each of which can contain arbitrary mesh
  o Can sweep cell-sets globally as on bricks, but locally must respect different sweep orderings for different directions

• **Improved Grind Times**
  o We have reduced grind time by 2X by profiling both PDT and STAPL and eliminating inefficiencies
Computer Science Research

- CERT CS research conducted within STAPL project
  - Parallel programming framework for C++
  - Similar abstractions as C++ standard template library (STL)
- PDT implemented using STAPL components
- Simulation efforts benefit from ongoing CS research efforts
Computer Science Research

- CERT CS research conducted within STAPL project
  - Parallel programming framework for C++
  - Similar abstractions as C++ standard template library (STL)
- PDT implemented using STAPL components
- Simulation efforts benefit from ongoing CS research efforts
Computer Science Research

- CERT CS research conducted within STAPL project
  - Parallel programming framework for C++
  - Similar abstractions as C++ standard template library (STL)
- PDT implemented using STAPL components
- Simulation efforts benefit from ongoing CS research efforts
Computer Science Research

- CERT CS research conducted within STAPL project
  - Parallel programming framework for C++
  - Similar abstractions as C++ standard template library (STL)
- PDT implemented using STAPL components
- Simulation efforts benefit from ongoing CS research efforts
Computer Science Research

- CERT CS research conducted within STAPL project
  - Parallel programming framework for C++
  - Similar abstractions as C++ standard template library (STL)
- PDT implemented using STAPL components
- Simulation efforts benefit from ongoing CS research efforts
Computer Science Algorithms

- Mixed-mode support in STAPL runtime
  - MPI + threads
- Nested Parallelism
  - Proof of concept complete
  - Expression of nested parallel algorithms being implemented
  - Mapping execution of nested parallel algorithms to system being implemented
- Kripke implementation using STAPL
  - Kripke is transport miniapp from LLNL
  - Kripke/STAPL implementation uses mixed-mode and nested parallelism
- Lessons learned from Kripke will feed into PDT
Kripke Overview

• Transport mini-app from LLNL
• Explicit MPI+OpenMP implementation
• Parallelizes Sweep Operation:
  o Nesting Level 1:
    • Parallelizes over spatial domain
    • Spatial domain distributed across the computer
    • Sweeps in all directions are started in parallel
    • Sweep traverses nodes of the computer
  o Nesting Level 2:
    • Parallelism within node
    • Parallelizes over direction or energy domain
    • Kripke explores impact of data striding and loop nesting
      o Different kernels implemented to change nesting of loops over space, energy, and direction

• STAPL implementation of Kripke
  o Enables parallelization over spatial domain within node as well
  o CERT Deep Dive on February 17-18 will demonstrate STAPL component use in Kripke
Lessons Learned

• We must automate:
  o generation and verification of input files, including cross-section libraries
  o processing and verification of output files, including extraction of QOIs
  o archival of runset files

  Scripts for these purposes are being written by a postdoc that started last month

• The time required for students to learn how to code in PDT and STAPL is too long.

  New documentation and training material will be generated by a new postdoc arriving in March
Tuesday February 17, 2015
8:30-9:00  Continental Breakfast
9:00-9:15  Welcome
9:15-9:45  CERT Motivating Problem Overview
9:45-10:45 STAPL Overview
10:45-11:00 Break
11:00-11:30 STAPL pContainers
11:30-12:00 STAPL pViews
12:00-12:30 STAPL pAlgorithms and Skeletons
12:30-1:30 Lunch
1:30-2:00  STAPL Runtime System
2:00-2:15  Kripke Radiation Transport Miniapp Overview
2:15-3:30  Parallel Algorithm Development by way of example using Kripke
3:30-4:00  Break
4:00-5:30  Implementing Parallel Containers by way of example using Kripke
6:00 Dinner

Tuesday Evening

The STAPL implementation of Kripke will be provided to attendees along with accounts on a local Cray system to allow them to explore the implementation on their own time.

Wednesday February 18, 2015
8:30-9:00  Continental Breakfast
9:00-10:00 Feedback on the previous day
10:00-12:00 Lessons from Developing Parallel Transport using STAPL
12:00-1:00 Lunch
1:00-2:00  Wrap-up
2:00     Adjourn
PDT Arbitrary Grids

- Reactor grids

- 2D triangular grids built independently on rectangular regions, then *stitched at region interfaces* (forming some polygons)
Numerics

• Implementation of new grid capabilities in PDT
• Development of new exponentially-convergent IMC algorithm
• HOLO least-squares method
• Good results for the multigrid method for the least-squares Sn equations