Parallel Sn Algorithms

Jim Morel
Center for Large-Scale Scientific Simulations
Texas A&M University

40TH ANNIVERSARY LECTURE SERIES

Introduction

- Sn methods for radiation transport play a major role at DOE and NNSA Labs.
- Sn iterative solution techniques are of a wave-front type and require special consideration for parallel implementation.
- Our purpose here is to review basic parallel Sn algorithms and to present some recent advancements.
- Much of the research described here was carried out over the last 6 years by the PSAAP-II Center for Exascale Radiation Transport (CERT), administratively located under the Center for Large-Scale Scientific Simulations at Texas A&M University, in collaboration with NNSA lab partners.
- The work at Texas A&M was led by Marvin Adams with contributions from Lawrence Rauchwerger, Nancy Amato, and many staff, students, and faculty. Our main NNSA laboratory collaborators were LLNL's Teresa Bailey, Adam Kunen, Peter Brown, and Rob Falgout.

Overview

- The Boltzmann transport equation
- Source iteration
- Rectangular-mesh parallelism for sweeps in 1-D, 2-D, and 3-D
- PDT/STAPL Performance
- Unstructured-mesh parallelism
- Special mesh generation
- Mesh cutting
- Load balancing via mesh cutting
- Recent and ongoing research
- An open source Sn code

Sn Multigroup Transport Equations

$$\vec{\Omega}_m \bullet \vec{\nabla} \psi_{m,g} + \sigma_{t,g} \psi_{m,g} = \frac{1}{4\pi} \sigma_{s,g' \to g} \phi_{g'} + \frac{1}{4\pi} q_{g,m}, \; m = 1, M, g = 1, G,$$

$$\phi_g = \int_{4\pi} \psi_g \, d\Omega = \sum_m \psi_{m,g} w_m .$$

- M discrete directions and G energy groups.
- Simplified interaction term –absorption + scattering.
- All coupling between directions and groups occurs on right side of equation.
- Independent advection-removal equation for each direction and group on the left side of equation.

Source Iteration

$$\vec{\Omega}_m \bullet \vec{\nabla} \psi_{m,g}^{\ell} + \sigma_{t,g} \psi_{m,g}^{\ell} = \frac{1}{4\pi} \sigma_{s,g' \to g} \phi_{g'}^{\ell-1} + \frac{1}{4\pi} q_{g,m}, \; m = 1, M, g = 1, G,$$

$$\phi_g^{\ell} = \sum_m \psi_{m,g}^{\ell} w_m.$$

- Discretization for each equation on left yields a block lower-triangular matrix with each block corresponding to unknowns in a spatial cell.
- Each cell coupled only to upwind neighboring cells on rectangular meshes.
- Direct inversion via a back substitution process called a sweep –one sweeps across the mesh in the direction of flow as one solves for the unknowns in each cell.
- Inversion process characterized by a certain number of sequential stages, but there is parallelism within each stage in 2-D and 3-D.
- *Spectral radius independent of mesh size* - approaches 0 as absorption or leakage dominates and 1 as scattering dominates.
- If spectral radius is large, various forms of diffusion preconditioning must be used requiring diffusion solves in addition to transport sweeps.

1-D Sweeps

- Two types of directions: left-to-right and right-to-left.
- Totally sequential.
- Cyclic reduction is possible, but no one does it!

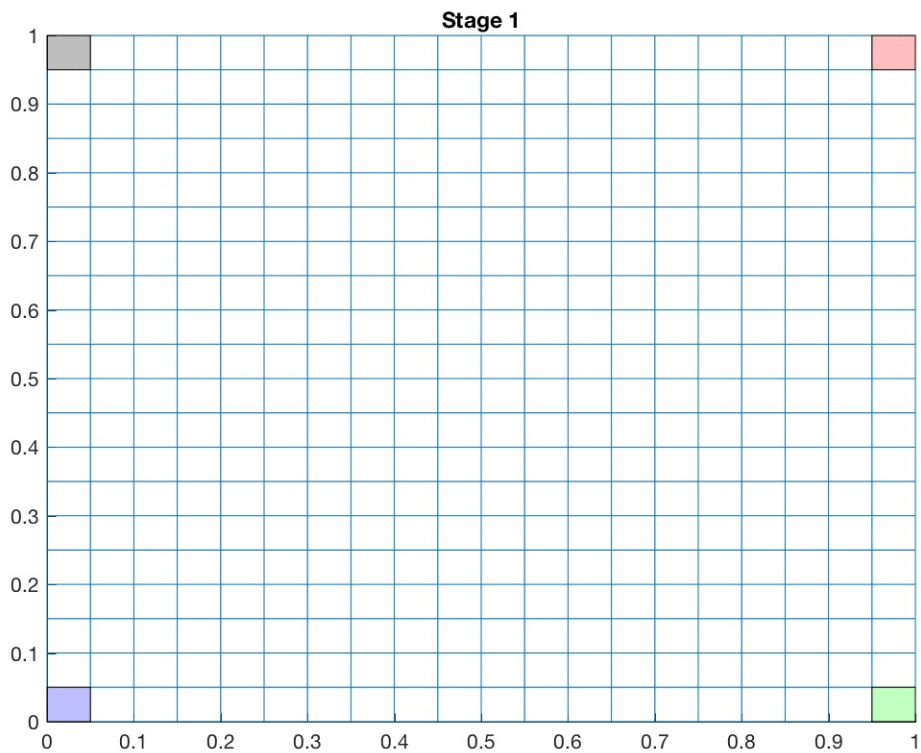
1	2	3	4	5	6	7	8	9	10
10	9	8	7	6	5	4	3	2	1

2-D Rectangular Mesh Sweeps

- Four types of directions (four octants of the unit sphere.)
- Sweeps starts at corners of the mesh.
- Parallelism exists but on 1-D domains (lines of varying size.)

1	2	3	4	5	6	7	8
2	3	4	5	6	7	8	9
3	4	5	6	7	8	9	10
4	5	6	7	8	9	10	12
5	6	7	8	9	10	12	13
6	7	8	9	10	12	13	14
7	8	9	10	12	13	14	16
8	9	10	12	13	14	16	17

2-D Sweep Animation



KBA Partitioning – 2-D Sweeps

- Each square corresponds to a set of spatial cells and their unknowns. A processor could be a core or node or any processing unit.

P1	P2	P3	P4	P5	P6	P7
1	2	3	4	5	6	7
2	3	4	5	6	7	8
3	4	5	6	7	8	9
4	5	6	7	8	9	10
5	6	7	8	9	10	11
6	7	8	9	10	11	12
7	8	9	10	11	12	13

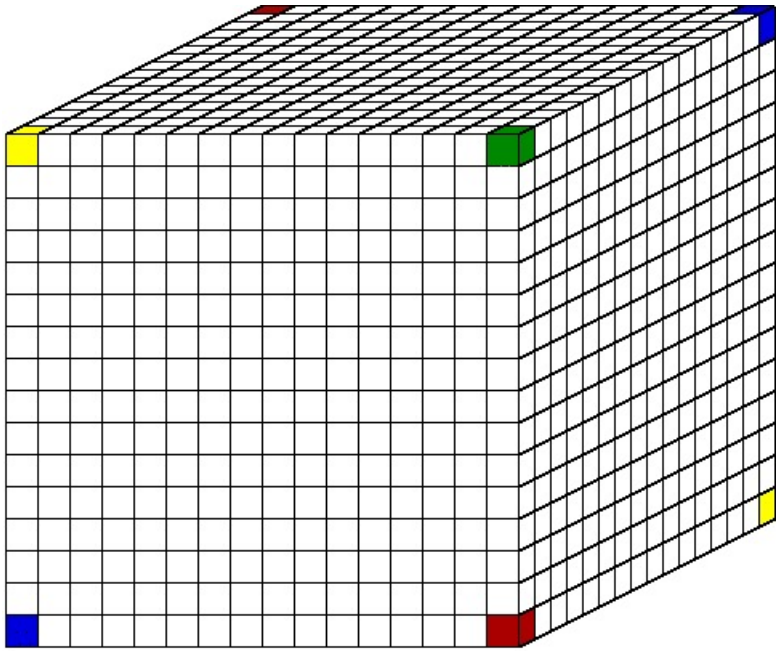
Pipelining

- For a single direction, processors are idle much of the time.
- To reduce the processor idle time, you start another direction in the octant as soon as the previous direction completes the first stage (pipelining.)
- You also start directions in all octants simultaneously, but you must deal with conflicts.
- You can aggregate cells, directions, and energy groups into sets with all directions in a set completed before a communication step.
- Efficiency is a strong function of the aggregation parameters.

3-D Rectangular Mesh Sweeps

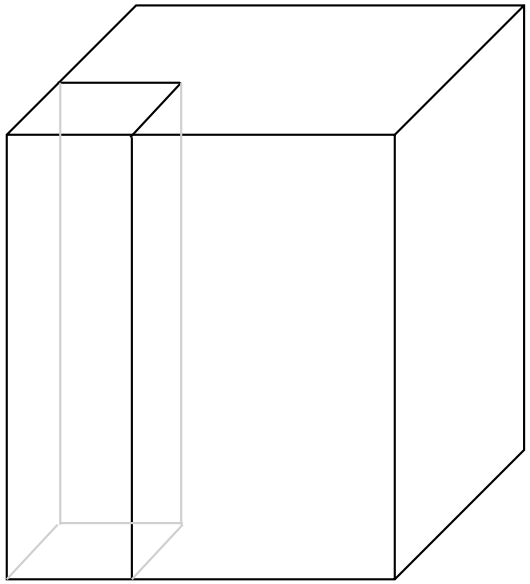
- Eight types of directions corresponding to eight octants of the unit sphere.
- Sweeps start at corners.
- Parallelism exists but on 2-D planes of varying size.
- The concepts of pipelining and aggregation apply.

3-D Sweep Animation



3-D KBA Partitioning

- One domain in a 3x3 decomposition (9 domains total).



PDT/STAPL

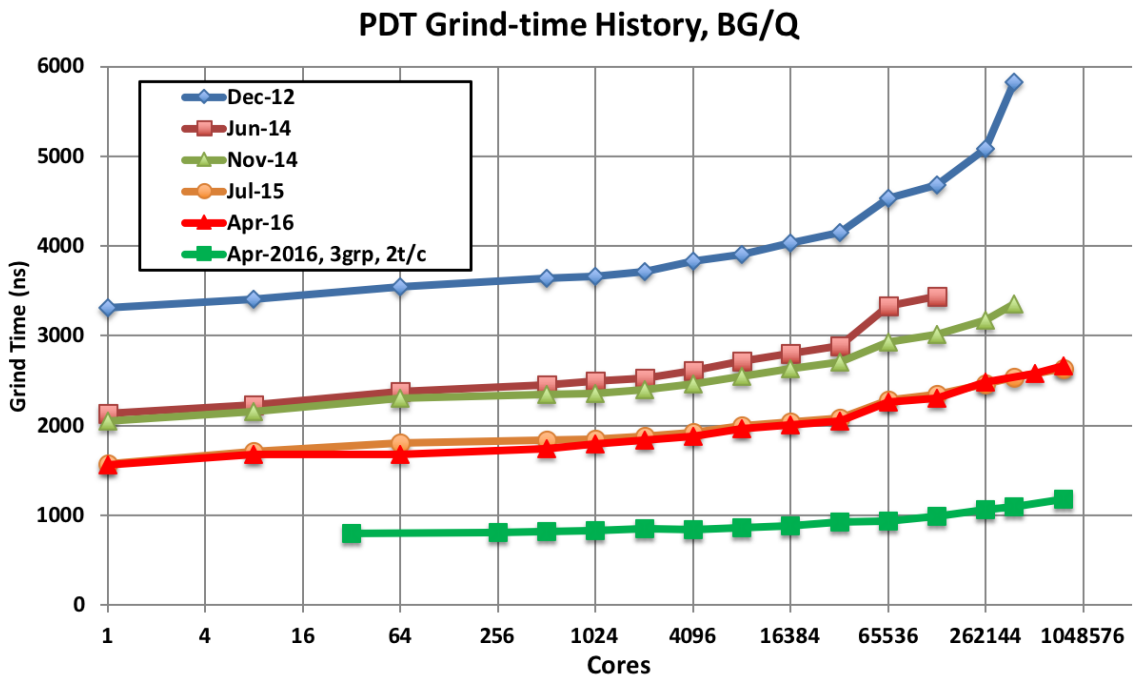
- Our research code was called PDT (parallel deterministic transport).
- All parallel services for PDT were obtained from STAPL (Standard Template Adaptive Parallel Library), which was developed at the Parasol Lab at TAMU led by Lawrence Rauchwerger and Nancy Amato.
- PDT employs a general polyhedral spatial mesh with a piecewise-linear DG finite-element approximation.
- 2-D - Piecewise-linear on interior triangles with discontinuity across cell faces.
- 3-D – Piecewise linear on interior tetrahedra with discontinuity across cell faces.
- Unknowns at nodes of the polyhedra.

Optimal Sweeps in PDT/STAPL

- We have formulated a performance model for PDT using rectangular partitions, and implemented an algorithm that selects partitioning and aggregation parameters at runtime to minimize time to solution.
- We have published a paper on optimal sweep algorithms for rectangular decompositions with results from PDT:
  - Michael P. Adams, Marvin L. Adams, W. Daryl Hawkins, Timmie Smith, Lawrence Rauchwerger, Nancy M. Amato, Teresa S. Bailey, Robert D. Falgout, Adam Kunen, Peter Brown, Provably optimal parallel transport sweeps on semi-structured grids, *Journal of Computational Physics,* **407** (2020) 109234.

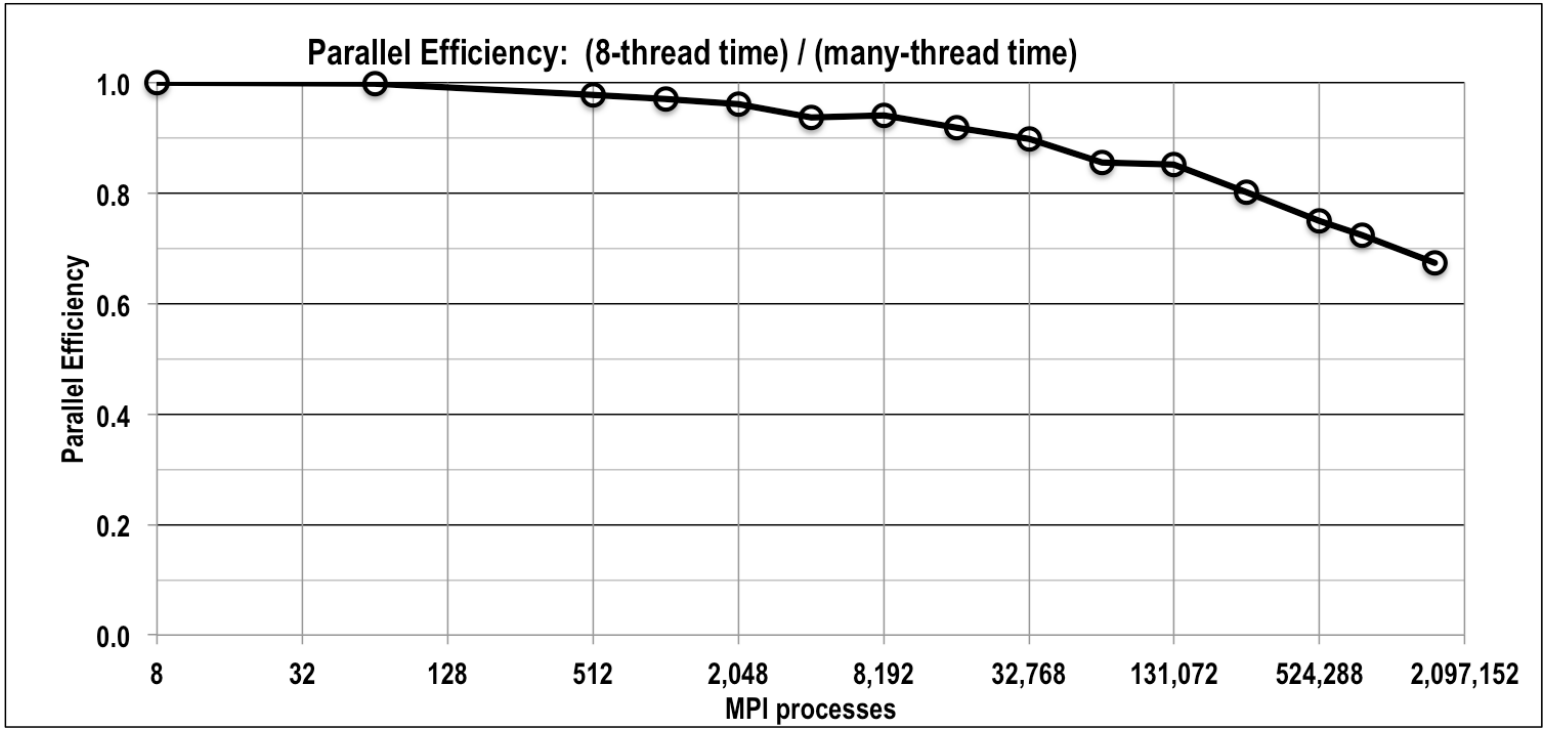
PDT/STAPL Performance

- PDT/STAPL performance for rectangular meshes has been steadily improving.



PDT/STAPL Performance

- Weak scaling (unknowns per thread constant) on Mira BG/Q out to 768K cores with two threads per core – almost 70% efficiency on 1.6 million parallel threads.



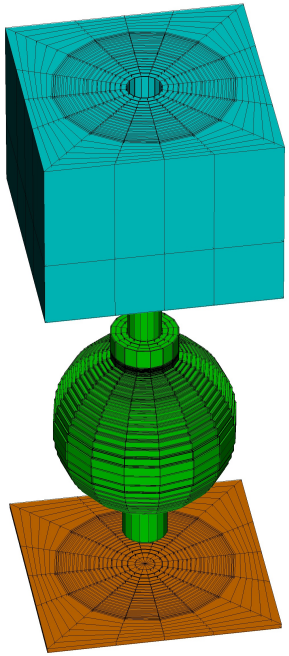
Unstructured Meshes

- Over the last several years we have turned our attention to unstructured meshes.
- On unstructured meshes, some cells can become interdependent causing the block lower-triangular structure to be lost. These occurrences are called cycles.
- This means you must lag some angular fluxes during the sweep, effectively iterating on part of the advection operator in addition to the scattering operator.
- This also means more iterations overall, possible degradation of preconditioners, and significant complications if you wrap the iterations in a Krylov method.
- Worse yet, you cannot use rectangular decompositions with arbitrary unstructured meshes. If you use standard ParMETIS type decompositions, you will have re-entrant domains, which introduce effective cycles and far more of them then you get from the mesh itself. We call these decomposition cycles.

Special Mesh Generation

- Rather than initially develop algorithms to handle cycles, we instead developed mesh generation techniques to avoid them.
- In particular, we developed the ability to generate 3-D polyhedral prismatic meshes obtained by extrusion of 2-D triangular meshes. These meshes did not admit mesh cycles and also enabled us to aggregate certain sets of directions, which was essential for adequate efficiency.

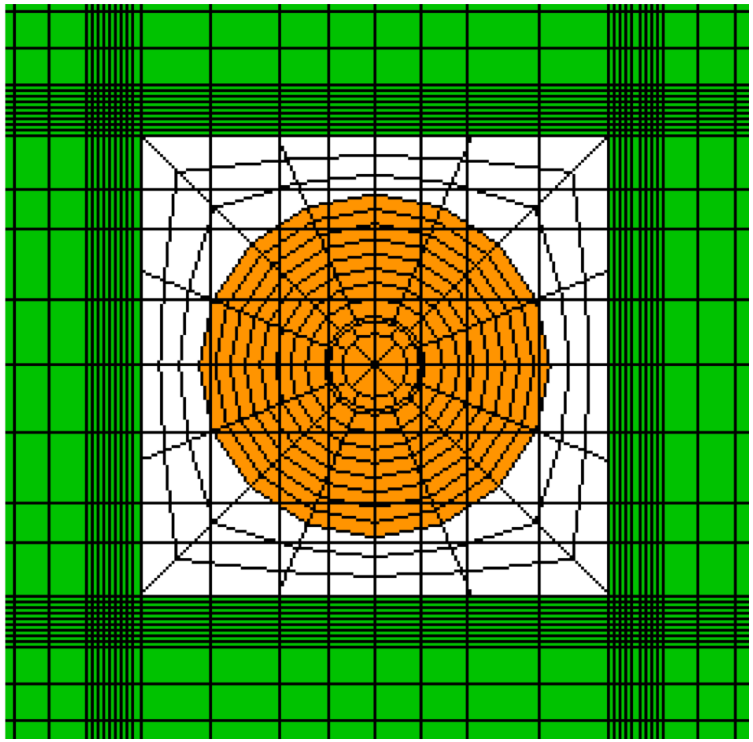
Cut-Away of Extruded Prism Mesh



Mesh Cutting

- In addition, we developed “mesh cutting” procedures that enabled us to convert unstructured prismatic meshes to semi-structured prismatic meshes without loss of resolution.
- The basic idea is to take an unstructured mesh, lay a rectangular macro-mesh over it that necessarily conforms only to the outer boundary, then incorporate the macro-mesh into the unstructured mesh.
- This yielded non-reentrant rectangular domains with each domain corresponding to a macro-mesh cell.
- Even though the mesh within a macro-cell is unstructured, we are able to use our rectangular-mesh technology for sweeping on these meshes.
- In an attempt to obtain better load balancing, we initially used non-uniform rectangular macro-meshes, getting to within a maximum load imbalance of factor or 2 or so for typical CERT calculations.

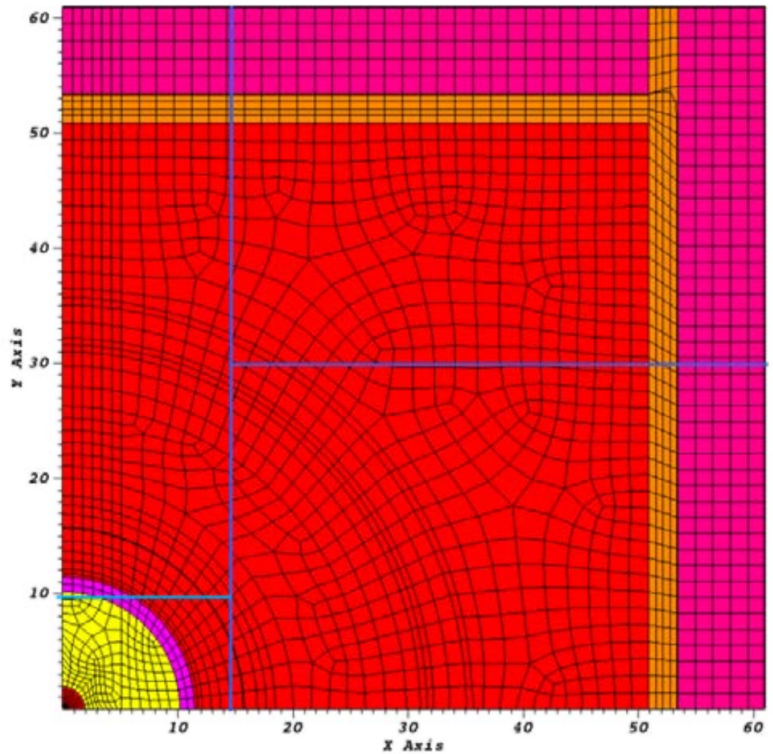
Non-uniform Rectangular Macro-Mesh



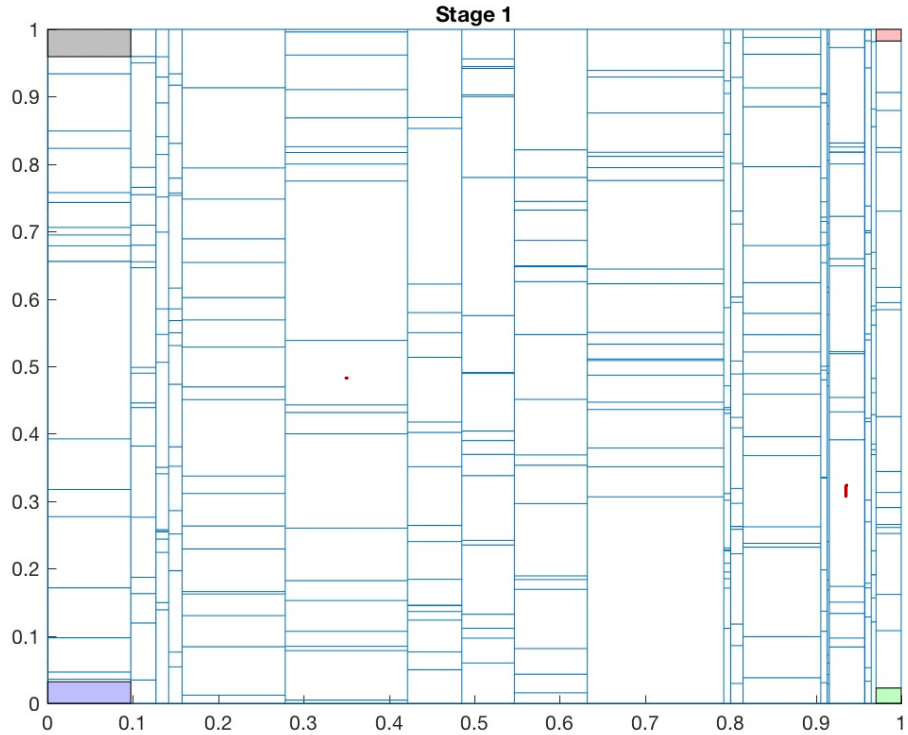
Load Balancing

- We then used hanging-node rectangular macro-meshes to get essentially perfect load balancing, but this had its drawbacks.
- Achieving load balancing in this way could dramatically increase the stage count for a given problem.
- The bottom line was that optimal efficiency did not correspond to perfect load balancing, but rather to an imbalanced load.
- We have developed an algorithm for approximately optimizing efficiency with this type of load balancing, but it remains a research topic.

Hanging Node Macro-Mesh



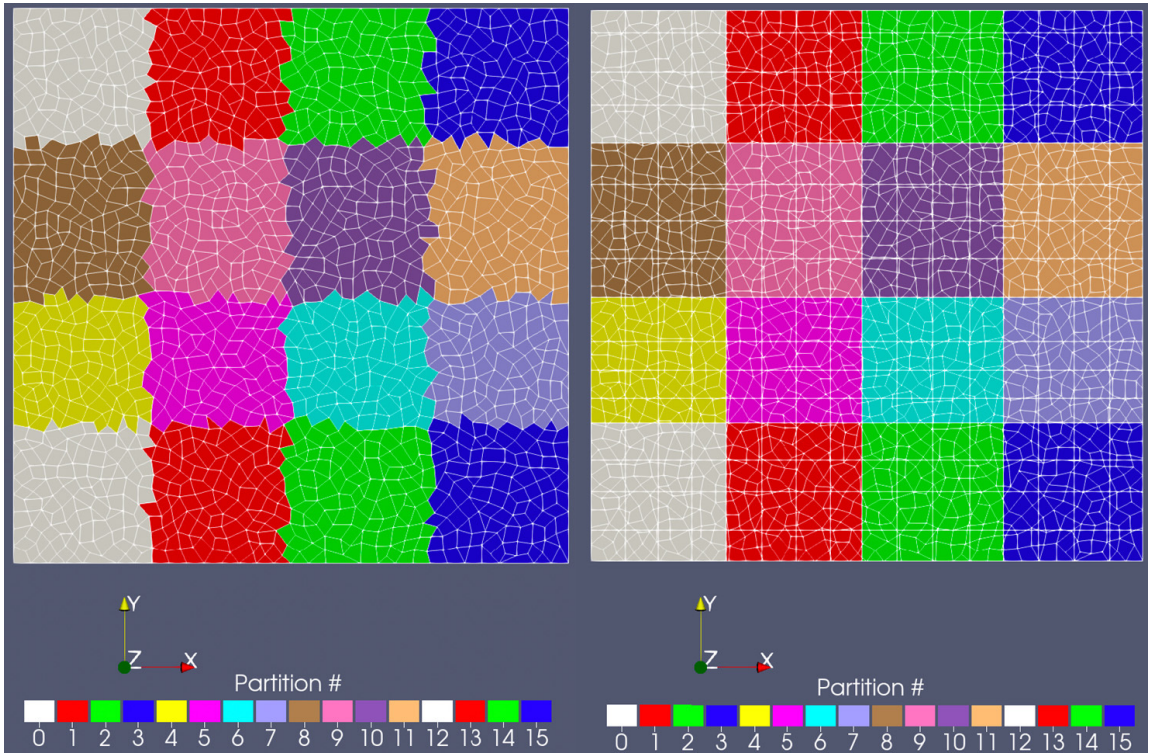
Increased Cycles 20x20 Macro Mesh



Recent and Ongoing Research

- We have developed a new Sn code called Chi-Tech that has largely replaced PDT.
- This code has the technology handle both mesh and decomposition cycles.
- We recently compared mesh cutting versus no cutting for extruded meshes with KBA-style decompositions with strong scaling from 32 to 2048 processes.
  - Jan I. C. Vermaak, Jean C. Ragusa, Marvin L. Adams , Jim E. Morel, "Massively parallel transport sweeps on meshes with cyclic dependencies," *Journal of Computational Physics*, **425** (2021) 109892.
- With no cutting, the number of source iterations increased by a factor of 2.25 (32 processes) to a factor of 3.08 (2048 processes), but the run time only increased from a factor of about 1.24 to a factor of about 1.6 (cutting significantly increases the number of cells.)
- We are now investigating cutting versus no-cutting with standard parMETIS partitioning on fully unstructured meshes (LANL-funded).
- We are also investigating a GPU implementation by which the scattering sources are represented on a polyhedral mesh while the sweeping is done on the underlying tetrahedral mesh (LLNL-funded).

No-Cut Versus Cut Partitions



Chi-Tech Sn Code

- We have received permission from NNSA to make Chi-Tech an open source code.
- If you are interested, you can access the repository at https://github.com/chi-tech .
- We welcome the opportunity to collaborate on parallel Sn algorithms.