Comparison of Four Parallel Algorithms for Domain Decomposed Implicit Monte Carlo

Thomas A Brunner (SNL)

Todd J. Urbatsch and Thomas M. Evans (LANL)

Nicholas A. Gentile (LLNL)

Joint Russian-American Five-Laboratory Conference on Computational Mathematics/Physics

19-23 June 2005

Vienna, Austria

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under contract DE-AC04–94AL85000.

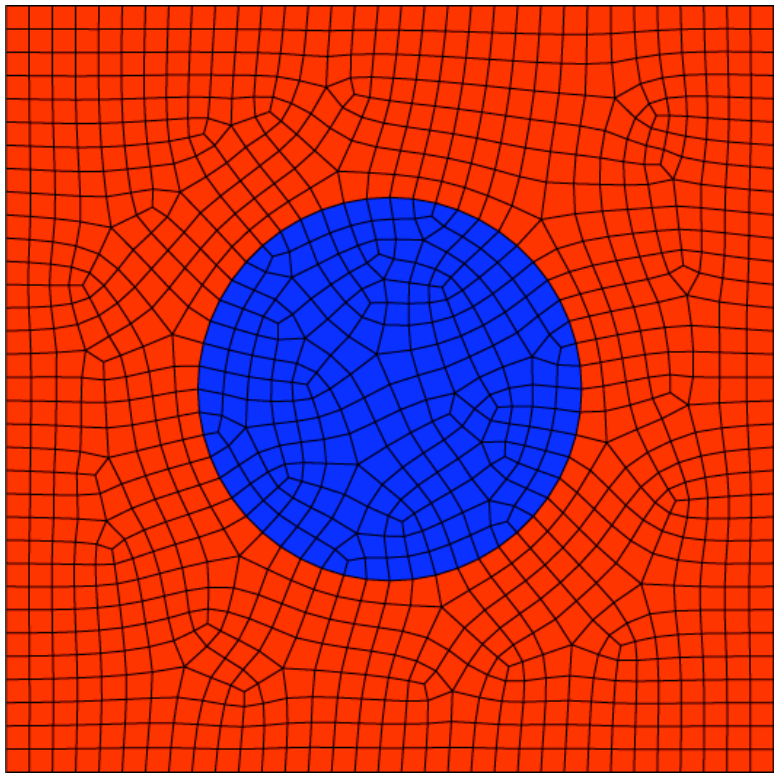
Los Alamos National Laboratory, an affirmative action/equal opportunity employer, is operated by the University of Californian for the United States Department of Energy under contract W-7405-ENG-36.

This work was performed under the auspices of the U.S. Department of Energy by University of California Lawrence Livermore National Laboratory under contract NO. W-7405-ENG-48.

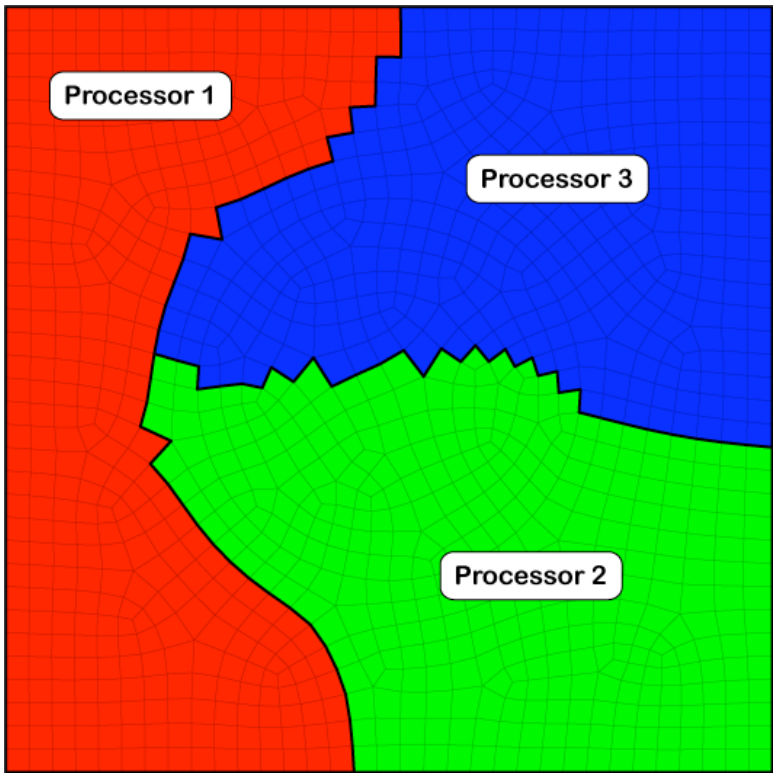
NNSA National Nuclear Security Administration

Sandia National Laboratories

Particle Simulation On A Decomposed Mesh



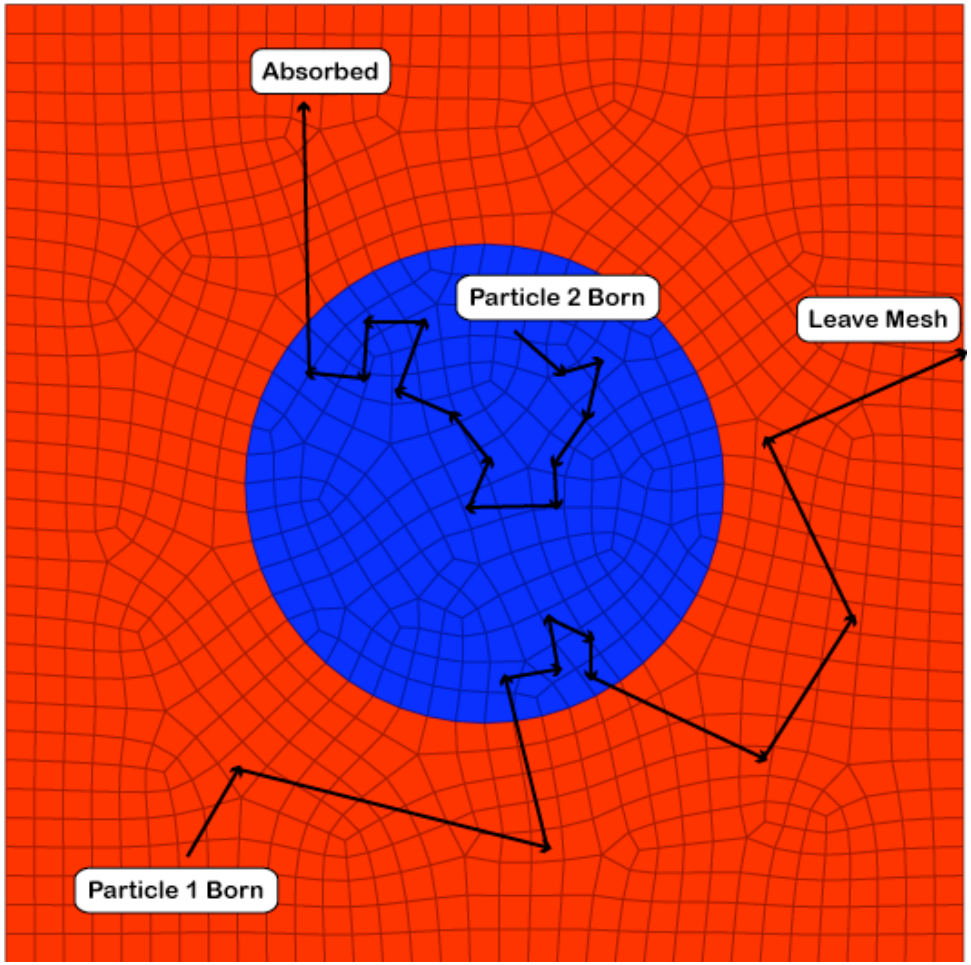
- Typical user mesh
- Cylinder of one material
- Another material outside



- Typical decomposition
- Load balances zones
- Irregular boundaries

Sandia National Laboratories

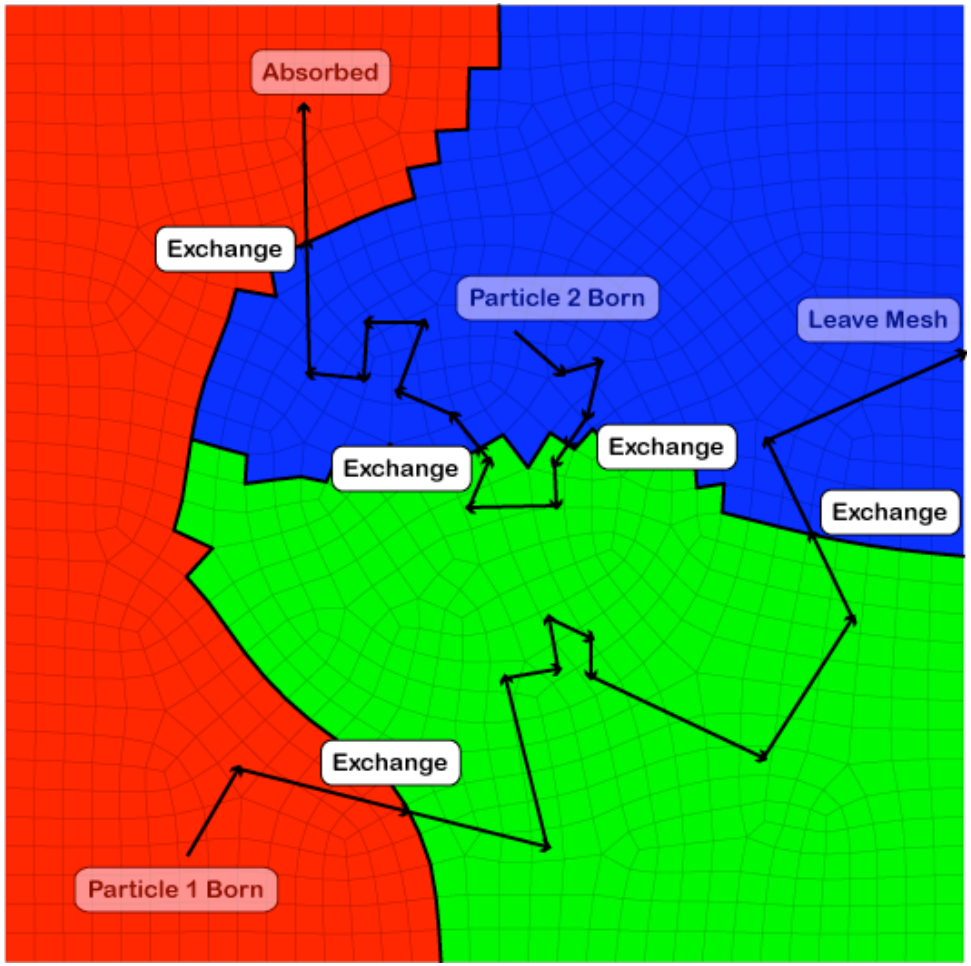
Physical Simulation



- Particles born in mesh
- Scatter in materials
- Deposit energy in zones
- Photons are terminated
  - by leaving mesh
  - being absorbed
  - or reaching cenus (the end of a time step).

Sandia National Laboratories

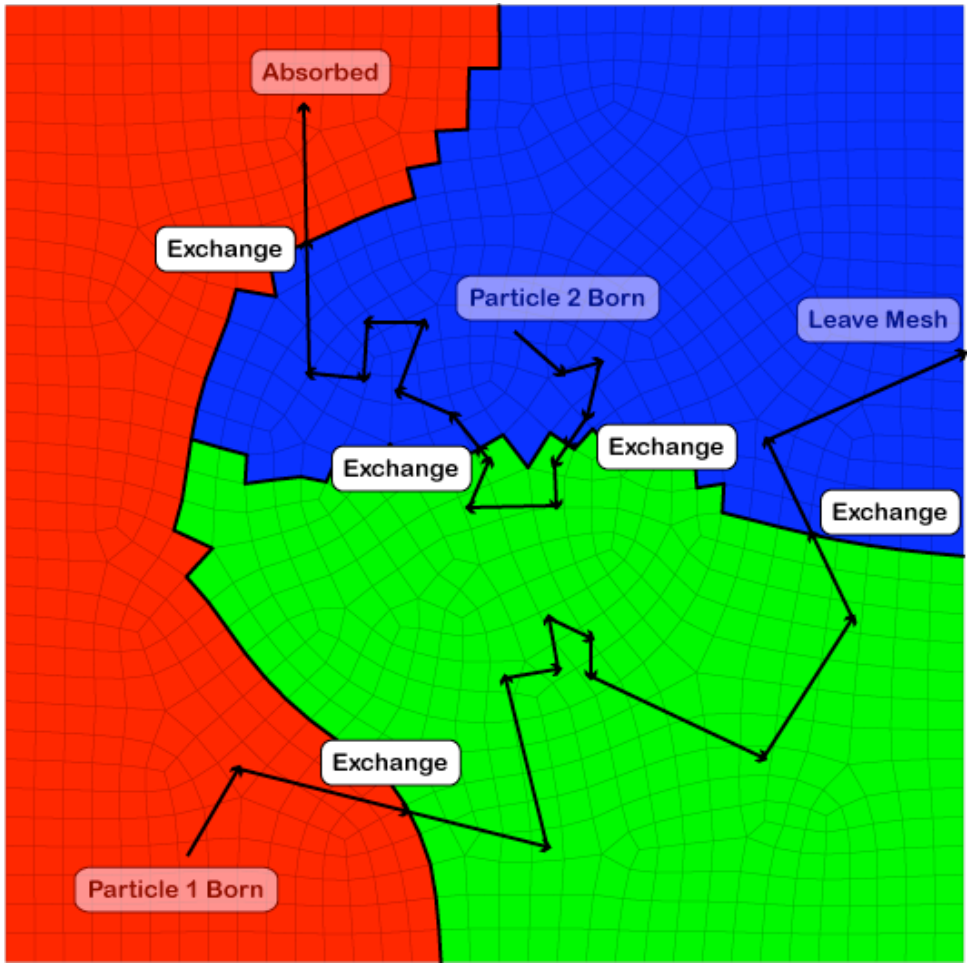
Decomposed Simulation



- New events for decomposed mesh:
  - Particles must be exchanged between processors.
  - Need to determine globally when all particles are finished.
- These four algorithms do these two steps in different ways.

Sandia National Laboratories

Algorithm 1: KULL

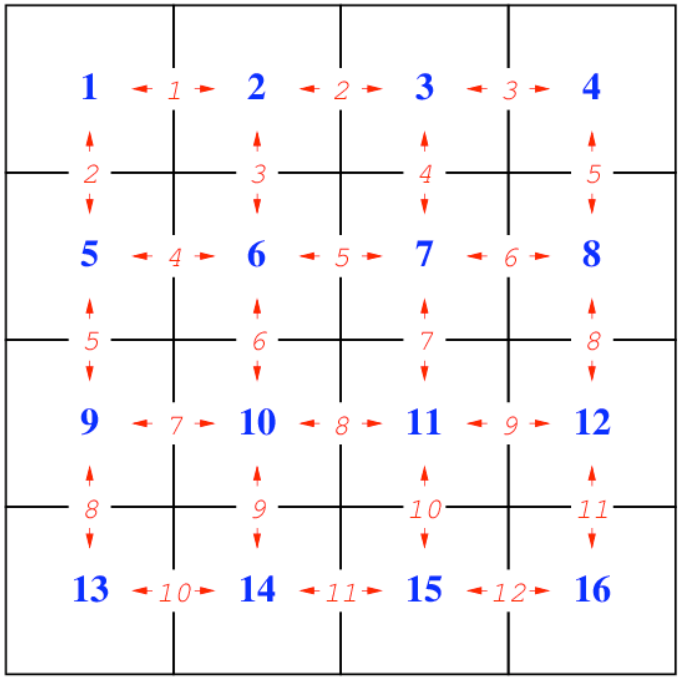


1. Each processor transports local particles until all have:
   - hit processor boundaries,
   - been absorbed,
   - leaked off mesh, or
   - reached census.
2. Blocking exchange particles buffered on processor boundaries with neighbors.
3. Global sum tallying number of remaining particles to simulate on all processors.
4. If particle sum is nonzero, go to step 1.

Sandia National Laboratories

Problems With Algorithm 1

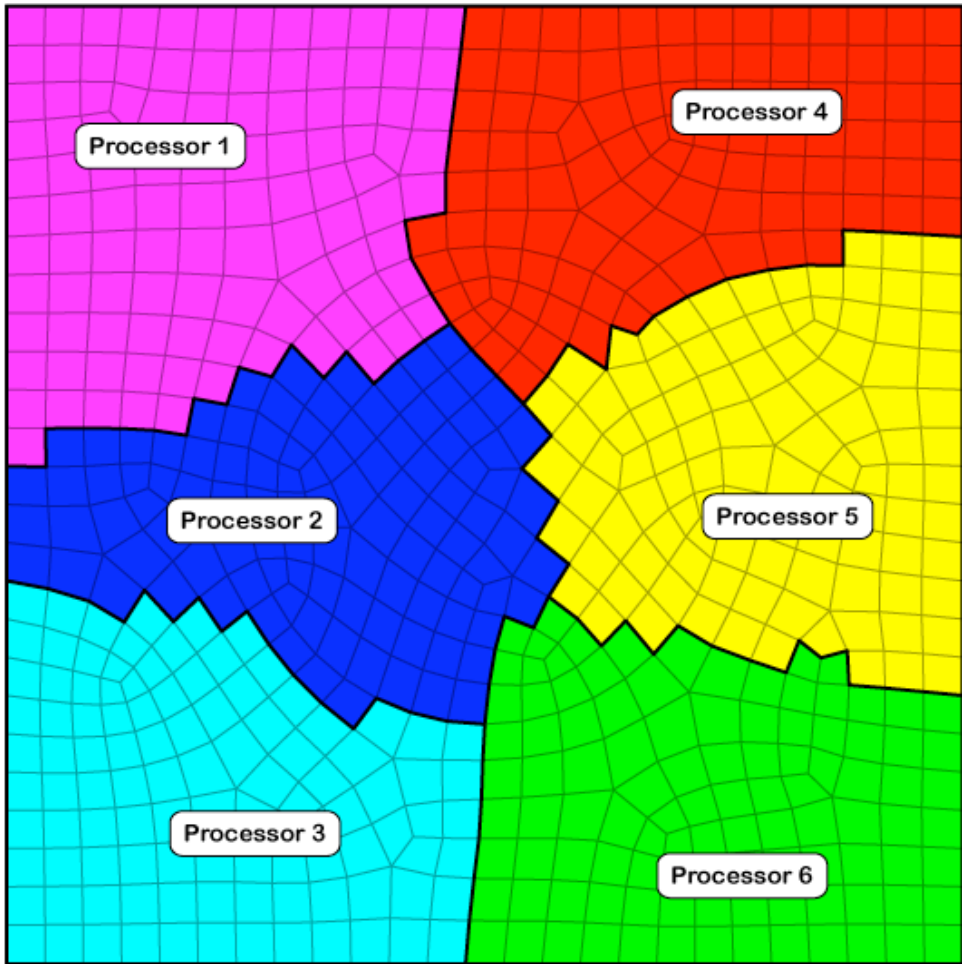
- Blocking sends for particle exchanges cause serialization of communication.
- Multiple synchronization points during a time step for global sum that is used to determine if all particles have finished.
- Algorithm 2 modifies Algorithm 1 to use asynchronous particle exchanges, fixing the first problem.
- Algorithm 3 eliminates the synchronization points, but it still does not scale well.

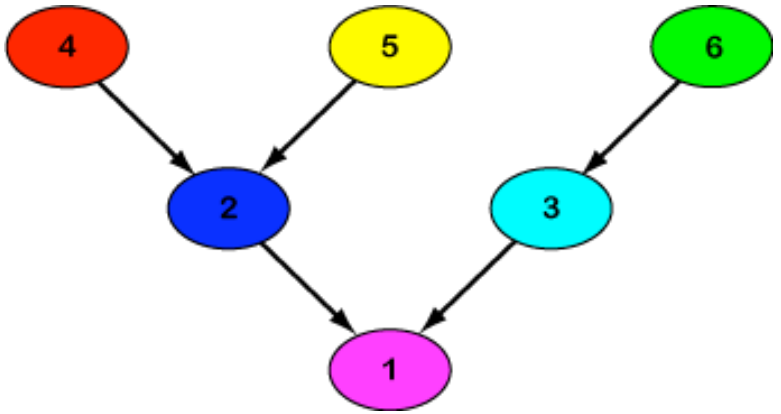


Another decomposition to illustrate the serialization of Algorithm 1.

Sandia National Laboratories

Algorithm 4: Improved Milagro





- More frequent asynchronous particle exchanges of smaller, fixed-size buffers.
- Master (Processor 1) keeps track of total number of particles completed
  - Collected asynchronously through a binary tree.

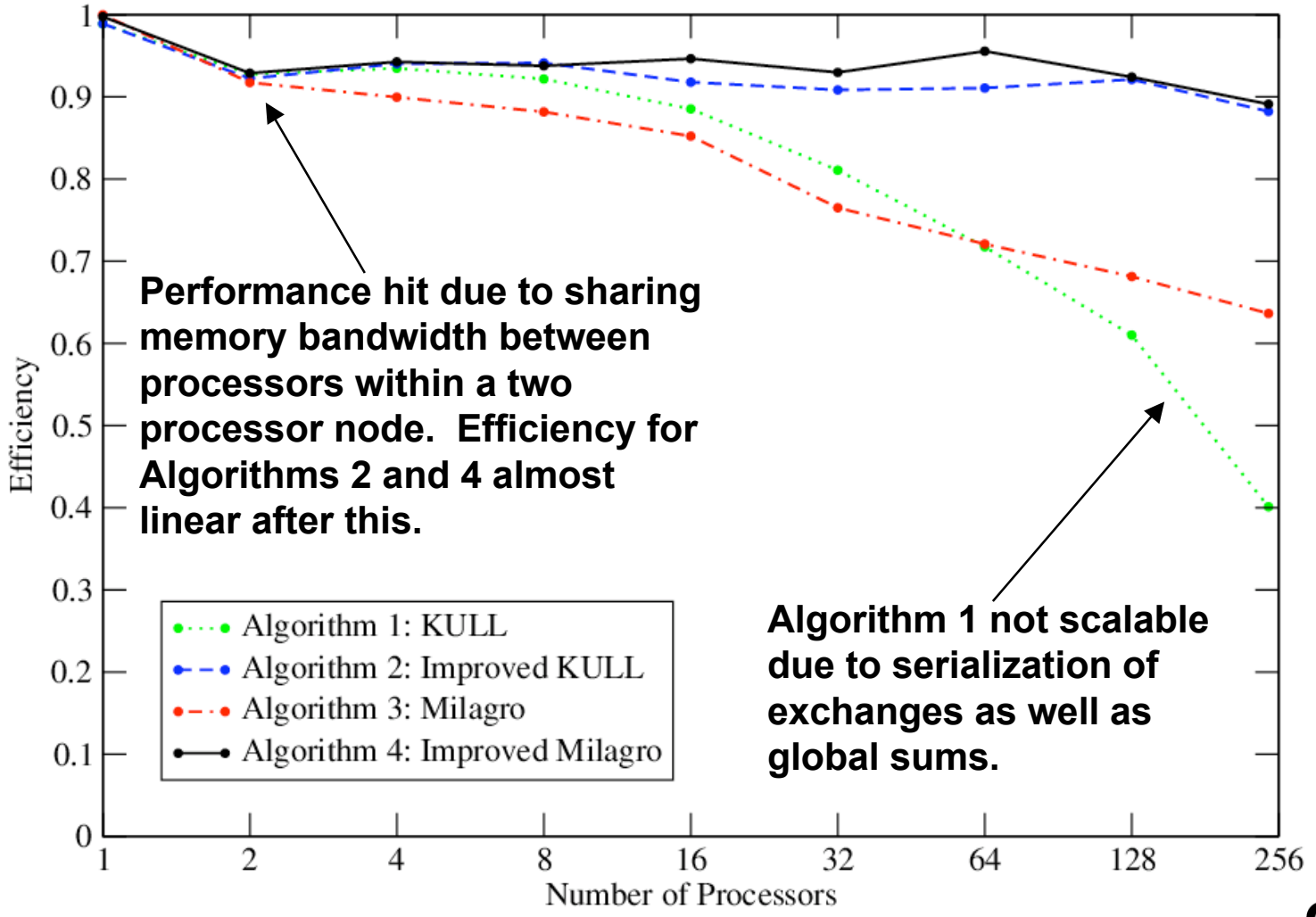
Sandia National Laboratories

Test Problem 1: A Hot Box

- Uniform temperature box
  - 60x60x60 zones
  - Each zone about one mean free path
- Reflecting boundary conditions
- Perfectly load balanced
- Constant work (strong) scaling study
  - Subdivided same exact problem on multiple decompositions
  - Same exact result from all four algorithms

Sandia National Laboratories

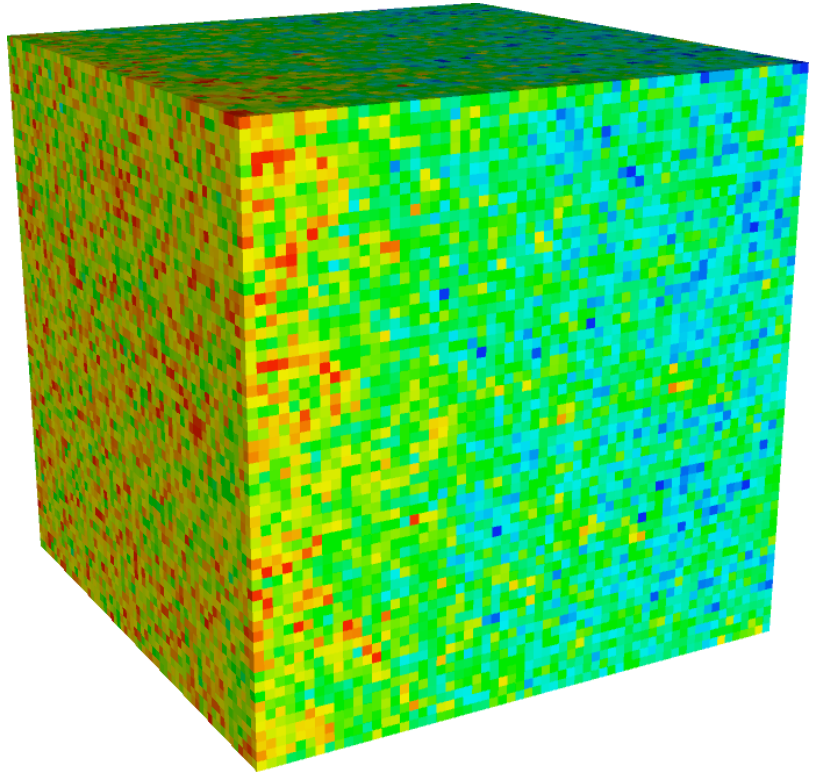
Hot Box Constant Work Study



Sandia National Laboratories

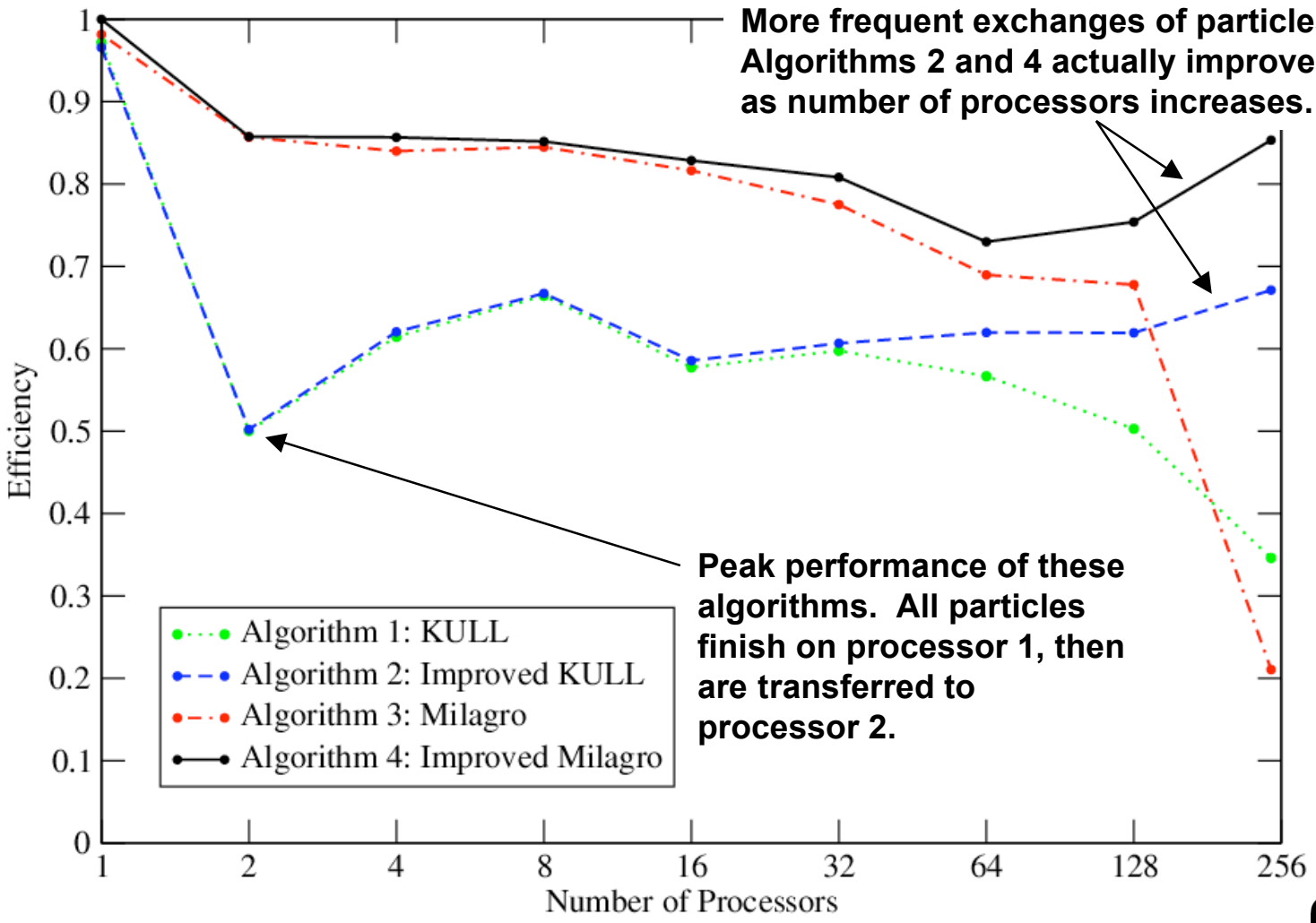
Test Problem 2: Mild Load Imbalance

- Vacuum Box
- Reflecting boundaries
- Initially Cold
- Temperature source on one side
- Initially load imbalanced
- Becomes balanced by end



Sandia National Laboratories

Vacuum Box Constant Work Scaling



More frequent exchanges of particles in Algorithms 2 and 4 actually improve efficiency as number of processors increases.

Peak performance of these algorithms. All particles finish on processor 1, then are transferred to processor 2.

Sandia National Laboratories

Conclusions

- The improved Milagro algorithm, Algorithm 4, scales almost linearly for load balanced problems.
- Critical to avoid all synchronization points in algorithms, either through blocking sends or global sums.
- Each processor must have the same parallel overhead.
- More frequent exchanges of particles can help in load imbalanced problems.

Sandia National Laboratories