Title: An Overview of the Implicit Monte Carlo Algorithm

Author(s): Long, Alex Roberts

Intended for: Share with external LDRD collaborator

Issued: 2021-04-15

Disclaimer:
Los Alamos National Laboratory, an affirmative action/equal opportunity employer, is operated by Triad National Security, LLC for the National Nuclear Security Administration of U.S. Department of Energy under contract 89233218CNA000001. By approving this article, the publisher recognizes that the U.S. Government retains nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or to allow others to do so, for U.S. Government purposes. Los Alamos National Laboratory requests that the publisher identify this article as work performed under the auspices of the U.S. Department of Energy. Los Alamos National Laboratory strongly supports academic freedom and a researcher's right to publish; as an institution, however, the Laboratory does not endorse the viewpoint of a publication or guarantee its technical correctness.

An Overview of the Implicit Monte Carlo Algorithm

Alex R. Long
Computational Physics and Methods (CCS-2)
Los Alamos National Laboratory

Seedling LDRD 2021

Operated by Triad National Security, LLC, for the U.S. Department of Energy's NNSA

Implicit Monte Carlo models the exchange of energy between radiation and matter

- Implicit Monte Carlo simulates x-rays moving and exchanging energy with matter
- The radiation field is represented with particles that have discrete energy weights separate from their frequency
- One Monte Carlo particle represents many physical particles at a given frequency
- Together all particles represent the total radiation energy in the system
- Simulating all particle histories solves the Boltzmann transport equation and gives us the amount of energy absorbed into the material, which is used by hydrodynamics codes (thus *rad-hydro* simulations)

Monte Carlo transport can be thought of as a particle moving in straight lines until it interacts

- The initial particle angle is isotropic (uniformly distributed over unit sphere)
- For IMC, all particles travel in straight lines at the speed of light
- Particles can undergo three possible events:
  - Scatter
  - Cross a spatial cell boundary
  - Reach the end of the time step (known as *census*)
- The distance is calculated for each event, the event with the minimum distance is the event that actually occurs:

$$d_{event} = \min(d_{scatter}, d_{boundary}, d_{census}).$$

After event an event is chosen, the particle state and material state are updated

- The particle deposits energy into the material as it moves to the chosen event. This is known as *continuous absorption*, which reduces solution variance
- **Scatter** After a scattering event occurs, a new angle is sampled for the particle
- **Boundary** A particle moves out of a spatial cell, generally moving to a different spatial cell with different physical data
- **Census** Particle is done transporting (end of a `while` loop)

A `while` loop of event processing makes up the core of the history-based IMC algorithm

```
Result: Complete all IMC particle histories for a single timestep
for particle : all_particles do
    while particle.active() do
        d_scatter = get_distance_to_scatter();
        d_boundary = get_distance_to_scatter();
        d_census = particle.get_distance_remaining();
        d_event = min(d_scatter, d_boundary, d_census);
        exchange_energy_with_material(d_event, particle);
        if d_event == d_census then
            particle.set_inactive();
        else
            // particle scatters or enters next cell
            process_event(d_event);
        end
    end
end
```

What math functions and data are used in selecting an event?

- **Scatter** To determine the distance to a requires the natural log of a random number and the probability of a scatter per unit distance (the *scattering opacity*, σ_s)

$$d_{scatter} = \frac{\log \xi}{\sigma_s}$$

- **Boundary** This is a ray trace operation—find the nearest plane that intersects a ray. This can be simple or complicated depending on the mesh type. Our work will focus on block AMR, which has a very simple representation.
- **Census** This event just requires reading a field carried in the particle: the time until the end of timestep

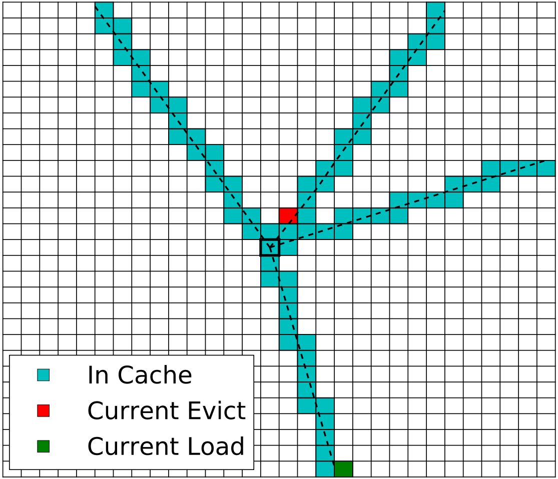
What math functions and data are used in processing an event?

- **Scatter** Scattering generally has two components:
  - Sampling a new frequency group: this requires sampling from a cumulative distribution function to determine a new frequency for the particle, which means sampling a random number and reading up to N_g floats.
  - Sampling a new angle: this requirs sampling two random numbers and $\sin$ and $\cos$ trigonometric functions
- **Boundary** The particle's cell index is updated, no math or data is required
- **Census** The particle's state is set to `census`, no math or data is required

IMC in the streaming limit has many data loads, few operations and little data reuse

- High energy density physics simulations are moving towards high fidelity (many frequency groups), 3D runs
- Implicit Monte Carlo (moving x-ray emission energy around) can take up to 80% of runtime in multiphysics simulations
- Memory footprint per mesh cell is increasing, cache size per core is decreasing
- “Everything the light touches” needs to be loaded into memory—we know from VTune analysis IMC is spending lots of time serving last level cache misses on KNL architectures

IMC in the streaming limit has many data loads, few operations and little data reuse



- Four particles sourced in a single cell require data from 84 cells
- The fourth particle history begins to evict the last used cell data from a hypothetical cache
- This issue is exacerbated with more particle histories per cell

We know exactly how much data is required to process a particle history in each cell

- All of the physical data in IMC is temperature dependent and is different for each cell
- To process a particle in multigroup within a cell, we need the following data:

Name	Number	Total Size (bytes)
Absorption opacity	n_{groups}	$8n_{groups}$
Scattering opacity	n_{groups}	$8n_{groups}$
Fleck factor	1	8
Cell vertices	2^{dim}	$8(2^{dim})$
Neighbor indices	$2dim$	$8(2^{dim})$

- For a 3D problem with 50 groups: $Cell_{size} = 920\mathrm{bytes/cell}$

Multiple particles are processed in batches in the event-based IMC algorithm

- To make IMC more SIMD friendly, the traditional `while` loop is inverted and all particles are processed together

```
Result: Complete all IMC particle histories for a single timestep
while not_done do
    for particle : all_particles do
        d_scatter = get_distance_to_scatter();
        d_boundary = get_distance_to_scatter();
        d_census = particle.get_distance_remaining();
        d_event = min(d_scatter, d_boundary, d_census);
        put_particle_in_event_queue(particle);
    end
    // all particles in a given queue will do the same thing (SIMD)
    for event : event_queue do
        not_done = proccess_event(event);
    end
end
```