Hybrid Deterministic/Monte Carlo Neutronics using GPU Accelerators

Jeff Willert *, C. T. Kelley*, D. A. Knoll †, Han Dong ‡, Mahesh Ravishankar §,
Paul Sathre ¶, Michael Sullivan ‖, William Taitano **
*Department of Mathematics, North Carolina State University, Raleigh, NC 27695-8205
Email: jawiller@ncsu.edu, tim_kelley@ncsu.edu
† Theoretical Division, MS B216, Los Alamos National Laboratory, Los Alamos, NM 87545
Email: nol@lanl.gov
‡ Computer Science and Electrical Engineering Department
University of Maryland Baltimore County, Baltimore, Maryland 21250 Email: han6@umbc.edu
§ Department of Computer Science and Engineering
Ohio State University, Columbus Ohio 43210 Email: ravishan@cse.ohio-state.edu
¶ Department of Computer Science, Virginia Tech University
Blacksburg, VA 24060, Email: sath6220@cs.vt.edu
‖ School of Electrical and Computer Engineering, University of Texas at Austin, 1 University Station
Mailcode C0803, Austin, TX 78712 Email: mbsullivan@utexas.edu
‖ Chemical and Nuclear Engineering Department, 1 University of New Mexico
Albuquerque, New Mexico 87131 University of New Mexico, Email: taitae25@unm.edu

***Abstract*—In this paper we discuss a GPU implementation of a hybrid deterministic/Monte Carlo method for the solution of the neutron transport equation. The key feature is using GPUs to perform a Monte Carlo transport sweep as part of the evaluation of the nonlinear residual and Jacobian-vector product. We describe the algorithm and present some preliminary numerical results which illustrate the effectiveness of the GPU Monte Carlo sweeps.**

***Keywords*-Neutron Transport, Jacobian-Free Newton-Krylov, GPU, Monte Carlo**

I. Introduction

In a recent paper [1] we considered a Jacobian-Free-Newton-Krylov (JFNK) solver for the Nonliner Diffusion Acceleration (NDA) formulation of the neutron transport equation using a hybrid deterministic/Monte Carlo approach for evaluation of the nonlinear residual. In certain reactor calculations it may be highly preferable to solve the neutron transport equation using Monte Carlo methods [2]. Monte Carlo methods allow for a continuous treatment of space, angle and energy, removing discretization errors. Monte Carlo methods allow for us to simulate the exact physics using known probability distributions and continuous cross-section data. Furthermore, Monte Carlo methods allow us to treat complex geometries exactly. In addition, recent advances in computing allow us to exploit the massively parallel nature of the Monte Carlo simulation. In this paper we show how the Monte Carlo (MC) computations can be efficiently implemented on a GPU.

A. The Neutron Transport Equation

We consider the steady-state, mono-energetic, neutron transport equation in one space dimension with anisotropic scattering in a homogeneous medium [2]–[4]

$$\mu\frac{\partial\psi}{\partial x}(x,\mu) \quad +\Sigma_t\psi(x,\mu) = \frac{\Sigma_s}{2}\int_{-1}^{1}\psi(x,\mu')\,d\mu' + q(x)/2, \tag{1}$$

for $0 < x < \tau$ and $\mu \in [-1,0)\cup(0,1]$. We impose boundary conditions

$$\psi(0,\mu) = \psi_l(\mu), \mu > 0; \psi(\tau,\mu) = \psi_r(\mu), \mu < 0. \tag{2}$$

In (1)

- ψ is intensity of radiation or angular flux at point x at angle $\cos^{-1}(\mu)$
- $\tau < \infty$,
- $\Sigma_s \in C([0,\tau])$ is the scattering cross section at x,
- $\Sigma_t \in C([0,\tau])$ is the total cross section at x,
- ψ_l and ψ_r are incoming fluxes at the boundaries, and
- $q \in C([0,\tau])$ is the fixed source

The quantity of interest is the scalar flux

$$\phi(x) = \int_{-1}^{1}\psi(x,\mu')\,d\mu'. \tag{3}$$

One way to solve for ϕ is a linear fixed point approach called source iteration. Here one updates an approximation ϕ_c to ϕ by solving the one parameter family of ordinary differential equations

$$\mu\frac{\partial\psi_c}{\partial x}(x,\mu) + \Sigma_t\psi_c(x,\mu) = S(x) \equiv \frac{\Sigma_s}{2}\phi_c(x) + q(x)/2 \tag{4}$$

by integrating forwards with initial data ψ_l for $\mu > 0$ and integrating backwards with final data ψ_r for $\mu < 0$. One

978-0-7695-4818-0/12 $26.00 © 2012 IEEE
DOI 10.1109/DCABES.2012.37

obtains a solution ψ_c and new approximation to the flux is

$$\phi_+(x) = \int_{-1}^{1} \psi_c(x,\mu')\, d\mu'.$$

The fixed point formulation is $\phi_c = \phi_+$. One can express this as a compact fixed point problem

$$\phi - K\phi = g$$

and solve it with, say, a Krylov iterative method [5] rather than simple successive substitution. The Krylov approach performs much better, as do multilevel methods [6], [7].

Another approach is to reformulate the equation as a nonlinear problem for the flux [8]–[13]. We will describe the method in terms of *nonlinear diffusion acceleration* (NDA) [8], [9]. In this method one converts the fixed point problem for ϕ into a "low-order" nonlinear diffusion equation. The low-order equation is coupled to the "high-order" transport equation to make enforce consistency.

Evaluation of the nonlinear residual, given a low-order flux ϕ^{LO}, begins with solving (4) with $\phi = \phi^{LO}$ subject to the original boundary conditions (2). From ψ compute the high-order flux ϕ^{HO} and current J^{HO} from

$$\phi^{HO}(x) = \int_{-1}^{1} \psi(x,\mu')\, d\mu'.$$

and

$$J^{HO}(x) = \int_{-1}^{1} \mu'\psi(x,\mu')\, d\mu'.$$

Define

$$\hat{D} = \frac{J^{HO} + \frac{1}{3\Sigma_t}\frac{d\phi^{HO}}{dx}}{\phi^{HO}}.$$

We have solved the problem if ϕ^{LO} is the solution of the diffusion equation

$$\begin{aligned} F(\phi) &= \tfrac{d}{dx}\left[\tfrac{-1}{3\Sigma_t}\tfrac{d\phi}{dx}\right] + (1-c)\,\phi \\ &\quad + \tfrac{d}{dx}\left[\hat{D}(\phi^{HO}, J^{HO})\phi\right] \\ &= 0, \end{aligned} \tag{5}$$

with appropriate boundary conditions.

The numerical results in [1] and in § III of this paper use the standard second-order finite difference discretization of (5). The fully deterministic approach from [1], [8] used a diamond-difference S^N [2] discretization for (4). We will not go into details of the discetizations in this paper.

We will briefly discuss solvers here and refer the reader to [5], [14], [15] for the details on Newton-Krylov nonlinear solvers. Recall that Newton's method for a nonlinear equation $F(\phi) = 0$ updates a current approximate solution ϕ_c to a new one ϕ_+ by adding the Newton step

$$\phi_+ = \phi_c + s$$

where s is the solution of

$$F'(\phi)s = -F(\phi) \tag{6}$$

and F' is the Jacobian (finite dimensions) or the Fréchet derivative (infinite dimension). In a Jacobian-Free Newton-Krylov solver, one solves (6), the linear equation for the Newton step, with a preconditioned Krylov method.

In [1] we describe a preconditioned JFNK approach for solving (5). In that paper, solving (4) was the dominant cost of the computation. The Jacobian-vector product one needs in the GMRES iteration for the Newton step may be done with a forward difference

$$F'(\phi)v \approx \frac{F(\phi + hv) - F(\phi)}{h}$$

with an appropriate choice of difference increment h [5] or, as we did in [1] an analytic Jacobian-vector product, assuming that product can be evaluated in an efficient way. For the algorithm considered here, evaluation of an analytic Jacobian-vector product $F'(\phi)v$ is a simple application of the chain rule and the solution of (4) with $\Sigma_s v/2$ as the right side. Our approach solves (4) and recovers fluxes and currents with a MC computation. In this way both the nonlinear residual and Jacobian-vector product can be done with MC.

B. Monte-Carlo

In recent years it has become increasingly popular to approximate the solution the neutron transport equation using stochastic methods. In order to make these stochastic calculations more efficient, we would like to create hybrid algorithms that utilize accelerators such as NDA [1], [8] or Quasi-Diffusion [8], [13], which were originally developed for deterministic methods. We will use MC simulation to approximate the scalar flux, current and Eddington tensor.

MC simulations use random number generation to sample probability distributions that describe likelihoods for physical events. By simply considering the physical process we're trying to model we can write a MC algorithm that will approximate these desired physical quantities.

For every neutron in the system which we are modeling, we must track its location and direction. If we isolate a single neutron, we can describe its "particle history" in very simple terms:

1) The particle is "born" at some location in the medium. The probability of being born in any given location is dependent on the fixed source, the scattering source, and, in general, the fission source. We'll only concern ourselves with the fixed and scattering sources in this application.
2) The particle travels in some straight-line direction away from its starting location until it undergoes an interaction ("collision") within the medium or it leaves the medium. It is important to note that neutrons interact only with the medium, not other neutrons. This makes each particle history independent of one another.

3) At the point of a collision, one of two things may occur. The neutron may be absorbed by the medium and this concludes the particle history. Also, the neutron may scatter off the material at the location of interaction. In this case, the particle will assume a new direction and continue along its particle history. If the particle scatters, we return to Step (2) and continue.

Once all of the particle histories have been observed, we can tally the physical quantities in which we are interested. We consider both *face tallies* and *track-length tallies*. With *face tallies*, a particle contributes to the overall tally each time it cross the face of a cell. In this case, when a particle crosses a face, we add a small contribution based on its relative weight factor to the tally. In a particle does not cross any cell faces, the particle does not contribute to the tally statistics.

With *track-length tallies*, every particle will offer some contribution to the over tallies. In this case, a particle gives a contribution to the overall tally in every cell which it travels. In this case, the contribution to a given cell is based on the distance the particle traveled within that cell times the particle's relative weight factor. Track-length tallies are generally preferable because they give a more "continuous" tally and each particle gives us some information that helps shape the overall physical quantities.

Within every algorithm for solving the neutron transport equation, we must compute one or more transport sweeps, that is solve (4) for ψ_c and then compute fluxes and currents.

This same idea can be used to build a MC transport sweep (i.e. compute the action of a transport sweep using a MC simulation). Just as we do for the deterministic case, we build a fixed source term for (4) and ask the MC simulation to solve a scattering-free problem. This amounts to simulating particle flights in which all particles are absorbed at the point of their first interaction (collision within the medium).

In this case, our MC algorithm simplifies dramatically. We have removed the need to loop through the simulation process within each particle history and all logic has been removed from the particle simulation process. The new, simplified algorithm for simulating a single particle history results:

1) Determine a starting location for the particle and a starting weight based on the source term $S(x)$.
2) Use random number generation to compute a direction for the particle to travel.
3) Use random number generation to compute a distance for the particle to travel.

After the particle travels the pre-determined distance, it undergoes a collision within the medium and is absorbed. This concludes the particle history, and there is no need to test whether or not the particle has been absorbed or whether or not the particle has left the medium (however, this will need to be handled during the tallying phase). All logic has been eliminated from the simulation. It is important to note that tallying has been greatly simplified as well. Instead of being forced to tally fluxes, currents and Eddington tensors between every interaction, we must only tallying along one flight per particle.

II. Hybrid NDA with GPUs

The final part of the algorithm is the use of GPUs to perform the Monte-Carlo transport sweep with NDA, *i. e.* accumulate fluxes and currents from (4). If we consider the MC transport sweep, we realize that there is a significant potential for parallelism. Since each particle history is independent within the MC transport sweep routine, we can simulate each of these particles in parallel. It is important to remember that particles only interact with the medium, not each other. GPUs allow us to take advantage of massive thread-level parallelism in accelerating this process.

Furthermore, if we consider the mathematics required to simulate a particle history, the operations required are simple. We generate three to four random numbers and perform additions, multiplications and some division. These simple operations are well-suited for a GPU implementation of the algorithm. GPUs excel when they're asked to perform a large volume of simple, parallel computations in which memory storage is low. In addition, by removing all (or most) of the logical checks within the algorithm, we've made the routine more GPU-friendly.

Given the choice of N particle histories per function evaluation and N_x spatial cells for binning the scalar flux, current and Eddington tensor, our storage cost is $O(4N + 3N_x)$. It is important to recall that $N \gg N_x$. This can be broken down into storage for the simulation phase and the tallying phase. During the simulation phase, we must store a particle weight, starting location, ending location and direction for each particle (four vectors of length N). During the tallying phase, we reduce the data from the simulation into a scalar flux, current and Eddington tensor (three vectors of length N_x or $N_x + 1$). Since N can be quite large, we may simulate only a subset (or batch) of particles before performing a partial tally. If we choose to use b batches, our storage cost may be reduced to $O(\frac{4N}{b} + 3N_x)$.

We have also investigated using the GPUs for particle simulation only, and letting the CPU execute the tallying phase. In this case, storage becomes less of a concern. Here, as particle histories are generated, the data is periodically sent from the GPU to the CPU for tallying. Using this method, we incur a communication expense, but it allows us to better use our resources if CPU(s) are available. One further option is to perform a partial tally on the GPU and send these partial tallies to the CPU where they are collected and a final tally is computed. This option has the advantage that the storage requirements are still low for the GPU, yet communication costs have been reduced. In this case, we

are only required to send three vectors of length N_x from the GPU to the CPU.

A. Algorithmic Description

The formal description of the algorithm is below. We express the use of MC in both the evaluation of the nonlinear residual F and the Jacobian-vector product J by making the number of particles N_{MC} and explicit argument. Hence $F(\phi, N_{MC})$ will denote the approximation of $F(\phi)$ with N_{MC} particles and $J(\phi, d, N_{MC})$ will be the Jacobian vector product $F'(\phi)d$ using MC with N_{MC} particles.

Newton-GMRES-MC

Evaluate $R_{MC} = F(\phi, N_{MC})$; $\tau \leftarrow \tau_r \|R_{MC}\| + \tau_a$.
while $\|R_{MC}\| > \tau$ **do**
  Use GMRES with a limit of I_{max} iterations to find d such that $\|J(\phi, d, N_{MC}) + R_{MC}\| \leq \eta \|R_{MC}\|$
  if the GMRES iteration fails **then**
    $N_{MC} \leftarrow 100 * N_{MC}$
    Evaluate $R_{MC} = F(\phi, N_{MC})$
  else
    $\lambda = 1$
    Evaluate $R_{Trial} = F(\phi + \lambda d, N_{MC})$
    while $\|R_{Trial}\| > (1 - \alpha\lambda)\|R_{MC}\|$ and $\lambda \geq \lambda_{min}$ **do**
      $\lambda \leftarrow \lambda/2$
      Evaluate $R_{Trial} = F(\phi + \lambda d, N_{MC})$
    end while
    if $\lambda \geq \lambda_{min}$ **then**
      $\phi \leftarrow \phi + \lambda d$
      $R_{MC} = R_{Trial}$
    else
      $N_{MC} \leftarrow 100 * N_{MC}$
      Evaluate $R_{MC} = F(\phi, N_{MC})$
    end if
  end if
end while

III. Computational Results

In this section we report timings for a transport sweep. The hardware configuration was an Nvidia Tesla c2075 GPU and an Intel(R) Core(TM) i5-2400 CPU with 8GB RAM.

In Table I we tabulate the number of particles, the average time for a transport sweep, and the ratios of the timings from one suite of particles to the next. The table shows that the timings scale very well with the number of particles.

Table I
Transport Sweep Timings

Particles (Millions)	Time (ms)	Ratio
0.03	0.7	
0.10	1.3	2.0
1.00	8.8	6.6
10.00	85.8	9.8
20.00	167.4	2.0

IV. Conclusion

In this paper we describe a Jacobian-free Newton-Krylov solver for the NDA formulation of the neutron transport equation. The novel feature in this paper is the use of a GPU to solve the low-order problem. We discuss the algorithm, the GPU implementation of the MC transport sweep, and present computational results.

Acknowledgment

This work was been partially supported by the Consortium for Advanced Simulation of Light Water Reactors (www.casl.gov), an Energy Innovation Hub (http://www.energy.gov/hubs) for Modeling and Simulation of Nuclear Reactors under U.S. Department of Energy Contract No. DE-AC05-00OR22725 and Los Alamos National Laboratory contract No. 172092-1.

References

[1] J. Willert, C. T. Kelley, D. A. Knoll, and H. K. Park, "Hybrid deterministic/monte carlo neutronics," 2012, submitted for Publication.

[2] E. E. Lewis and W. F. Miller, *Computational Methods of Neutron Transport*. Grange Park, IL: Americal Nuclear Society, 1993.

[3] S. Chandrasekhar, *Radiative Transfer*. New York: Dover, 1960.

[4] I. W. Busbridge, *The Mathematics of Radiative Transfer*, ser. Cambridge Tracts. Cambridge: Cambridge Univ. Press, 1960, no. 50.

[5] C. T. Kelley, *Iterative Methods for Linear and Nonlinear Equations*, ser. Frontiers in Applied Mathematics. Philadelphia: SIAM, 1995, no. 16.

[6] ——, "Multilevel source iteration accelerators for the linear transport equation in slab geometry," *Trans. Th. Stat. Phys.*, vol. 24, pp. 679–708, 1995.

[7] ——, "A fast multilevel algorithm for integral equations," *SIAM J. Numer. Anal.*, vol. 32, pp. 501–513, 1995.

[8] D. A. Knoll, H. Park, and K. Smith, "Application of the Jacobian-free Newton-Krylov method to nonlinear acceleration of transport source iteration in slab geometry," *Nuclear Science and Engineering*, vol. 167, pp. 122–132, 2011.

[9] ——, "A new look at nonlinear acceleration," *Nuclear Science and Engineering*, vol. 99, pp. 332–334, 2008.

[10] D. Y. Anistratov, "Nonlinear quasidiffusion acceleration methods with independent discretization," *Nuclear Science and Engineering*, vol. 95, pp. 553–555, 2006.

[11] W. A. Wieseiquist and D. Y. Anistratov, "The quasidiffusion method for transport problems in 2d cartesian geometry on grids composed of arbitrary quadrilaterals," *Nuclear Science and Engineering*, vol. 97, pp. 475–478, 2007.

[12] M. M. Miften and E. W. Larsen, "The quasi-diffusion method for solving transport problems in planar and spherical geometries," *Transport Theory Statist. Phys.*, vol. 22, pp. 165–186, 1993.

[13] V. Y. Gol'din, "A quasi-diffusion method for solving the kinetic equation," *USSR Comp. Math. and Math. Phys*, vol. 4, pp. 136–149, 1967, original published in Russian in Zh. Vych. Mat. I Mat. Fiz. 4,1078(1964).

[14] C. T. Kelley, *Solving Nonlinear Equations with Newton's Method*, ser. Fundamentals of Algorithms. Philadelphia: SIAM, 2003, no. 1.

[15] D. A. Knoll and D. E. Keyes, "Jacobian-free newton krylov methods: A survey of approaches and appliciations," *J. Comp. Phys.*, vol. 193, pp. 357–397, 2004.