Annals of Nuclear Energy 105 (2017) 161–167

Contents lists available at ScienceDirect

Annals of Nuclear Energy

journal homepage: www.elsevier.com/locate/anucene

Technical note

On the use of delta-tracking and the collision flux estimator in the Serpent 2 Monte Carlo particle transport code

Jaakko Leppänen

VTT Technical Research Centre of Finland, Ltd., Kivimiehentie 3, Espoo FI-02044 VTT, Finland

ARTICLE INFO

Article history:
Received 8 December 2016
Received in revised form 3 March 2017
Accepted 5 March 2017

Keywords:
Serpent
Monte Carlo
Transport simulation
Delta-tracking
Collision flux estimator

ABSTRACT

The Serpent Monte Carlo code was originally developed for the purpose of spatial homogenization and other computational problems encountered in the field of reactor physics. However, during the past few years the implementation of new methodologies has allowed expanding the scope of applications to new fields, including radiation transport and fusion neutronics. These applications pose new challenges for the tracking routines and result estimators, originally developed for a very specific task. The purpose of this paper is to explain how the basic collision estimator based cell flux tally in Serpent 2 is implemented, and how it is applied for calculating integral reaction rates. The methodology and its limitations are demonstrated by an example, in which the tally is applied for calculating collision rates in a problem with very low physical collision density. It is concluded that Serpent has a lot of potential to expand its scope of applications beyond reactor physics, but in order to be applied for such problems it is important that the code users understand the underlying methods and their limitations.

© 2017 Elsevier Ltd. All rights reserved.

1. Introduction

Monte Carlo codes are used in a wide range of radiation and particle transport problems in fields like reactor physics, radiation shielding and fusion research. The calculation is based on the simulated random walk of neutrons, photons or charged particles through the modeled geometry, with the results collected using stochastic estimators. The simplest way to evaluate physical reaction rates is to count the number of times the corresponding interaction was sampled during the transport simulation. In this paper such procedure is denoted as an analog estimator. For most reaction types, however, it is more efficient to evaluate the statistically expected frequency of the event, rather than sampled rate. This can be accomplished using what is denoted here as implicit estimators.

Analog and implicit flux estimators are generally used to evaluate integrals of the form:

$$R = \int_V \int_E \int_{\Omega} \int_t d^3r\, dE\, d\Omega\, dt\, f(\mathbf{r}, \hat{\mathbf{\Omega}}, E, t)\, \psi(\mathbf{r}, \hat{\mathbf{\Omega}}, E, t) \quad (1)$$

where f is an arbitrary response function, ψ is the angular particle flux and the integration is carried over the phase-space variables (position $\mathbf{r}$, energy E and direction of motion $\hat{\mathbf{\Omega}}$) and time t. When the response function is a macroscopic cross section, the integral yields the corresponding reaction rate, or more specifically, the number of reactions over the given integration intervals.

The most commonly used implicit estimators are the track-length flux estimator (TLE) and the collision flux estimator (CFE). The TLE is generally considered superior to the CFE for a very simple reason: track lengths are scored each time the particle crosses over the region of interest, even if no collisions occur inside it. There are at least three typical cases where the difference in efficiency is emphasized:

(i) When the reaction rates are evaluated in a material at very low density
(ii) When the region of interest is small or optically thin
(iii) In time-dependent simulations when reaction rates are divided into very short time bins.

A good example is the calculation of reaction rates induced by high-energy neutrons in the cladding of a fuel pin, which is important, for example, when evaluating the radiation damage to the cladding material. The mean-free-path (MFP) of high energy neutrons is typically measured in tens of centimeters, which means that particles emitted in fission most likely undergo their first collision outside the pin. When a high-energy neutron exits the fuel pin, it always makes a contribution to the track-length estimator. However, the probability that the first collision occurs inside the cladding is relatively low, and once the neutron has collided with

E-mail address: Jaakko.Leppanen@vtt.fi

http://dx.doi.org/10.1016/j.anucene.2017.03.006
0306-4549/© 2017 Elsevier Ltd. All rights reserved.

a light moderator atom, its kinetic energy has most likely dropped below the reaction threshold. Another extreme example is the calculation of flux in void, where the basic CFE always yields a zero result.

Despite its defects, the collision flux estimator is regularly used in Monte Carlo particle transport simulations. It is easy to implement, and in some special cases it may actually result in higher figure-of-merits (FOM) compared to the track length estimator.[1] Probably the most common reason, however, is that the transport routine is based on the Woodcock delta-tracking method (Woodcock et al., 1965). This is also the case for the Serpent code (Leppänen et al., 2015). Delta-tracking is essentially a rejection sampling scheme, which extends the particle paths over one or multiple material boundaries. The tracks are not stopped at the surface crossings, which rules out the use of the TLE for the calculation of reaction rates.

Even though Serpent users are generally well informed of the limitations of delta-tracking and the collision flux estimator, there are certain misconceptions that appear in discussions over and over again. The purpose of this paper is to point out that the efficiency problems related to the CFE are not as straightforward as they may first seem. The basic theory of tracking algorithms and result estimators is presented below, followed by a description of the methodology implemented in Serpent. The performance of the CFE and its dependence on the virtual collision frequency is then demonstrated by an example involving a simple photon skyshine problem. The paper concludes in a discussion of the implications of the presented results.

2. Background

The computational algorithms in Monte Carlo particle transport codes can be roughly divided into physics routines, handling the sampling of interactions and the production of secondary particles, geometry and tracking routines, responsible of transporting the particles through the modeled system, and stochastic estimators, used for collecting various results during the simulation. A brief overview of most common particle tracking routines and result estimators is provided below.

2.1. Surface- and delta-tracking

The Monte Carlo tracking routine handles the simulated random walk of particles from one interaction to the next. It can be shown (Lux and Koblinger, 1991) that the particle path length between collisions in an infinite homogeneous medium follows an exponential distribution, for which the probability density function (PDF) can be written as:

$$p(l) = \Sigma_{\text{tot}} e^{-l\Sigma_{\text{tot}}} \tag{2}$$

where material total cross section Σ_{tot} determines the total interaction probability per traveled path length. This distribution can be sampled using the inversion technique, which results in a simple formula for the next path length:

$$l = -\frac{\log \xi}{\Sigma_{\text{tot}}} \tag{3}$$

where ξ is a uniformly distributed random variable on the unit interval.

The prerequisite of obtaining (3) from (2) is that the interaction probability (i.e. the total cross section Σ_{tot}) is constant over the sampled path. The most common case where this condition does not apply is when the particle is transported through a heterogeneous geometry, and the path crosses the boundary between two materials. Since the path length was sampled using an interaction probability associated with the first material, the sample is not statistically valid beyond the boundary surface. Similar conditions apply to inhomogeneous material regions, in which either the density or the microscopic cross sections depend on position. Such conditions are encountered, for example, in coupled simulations of operating nuclear reactors with steep density gradients in the coolant channel and temperature gradients inside fuel pins.

The PDF of free path length in a heterogeneous geometry becomes a piece-wise continuous function, which cannot be sampled using a simple formula similar to (3). In practice there are two options to account for the heterogeneity:

(i) The particle track is stopped at each material boundary, and a new path length is sampled using the total cross section of the next material[2]
(ii) The discontinuities in the interaction probability are handled using a rejection sampling algorithm

These two options form the basis of the surface- and the delta-tracking method, respectively. Inhomogeneous material regions are most typically handled by dividing the geometry into a number of homogenized sub-zones approximating the spatial distribution.

Surface-tracking is the most commonly-used tracking algorithm in Monte Carlo codes, and it takes advantage of the fact that the remaining particle path length is independent of the distance traveled within the medium. If it is known that the next collision occurs beyond the boundary surface, the point where the track crosses to the other side can be taken as the starting point of a new sample. Surface-tracking requires stopping the track at each material boundary and calculating the distances to the nearest boundary surfaces, which corresponds to determining the points of discontinuity in the piece-wise continuous PDF. This operation may become computationally expensive in complex geometries, especially when the dimensions are small compared to the particle mean-free-path.

Delta-tracking is essentially based on a rejection sampling scheme, in which the PDF of free particle path length is sampled in two parts. The total interaction probability is made uniform over the entire geometry by the introduction of virtual collisions. A virtual collision is an interaction that preserves the energy and direction of motion of the particle, and therefore does not affect the statistics of the random walk in any way. The interaction probability in each material can be adjusted by adding an arbitrary virtual collision probability in the total, which for cross sections can be written as:

$$\Sigma'_{\text{tot},m}(E) = \Sigma_{\text{tot},m}(E) + \Sigma_{0,m}(E). \tag{4}$$

The adjusted total cross sections can then be set equal in all materials:

$$\Sigma'_{\text{tot},1}(E) = \Sigma'_{\text{tot},2}(E) = \Sigma'_{\text{tot},3}(E) = \cdots = \Sigma_{\text{maj}}(E), \tag{5}$$

which means that instead of a piece-wise continuous distribution, the PDF of free path length is written using a single exponential function:

[1] An example of a case where the CFE may perform better than the TLE is a mesh tally with very fine spatial subdivision. Scoring the TLE requires sub-diving the particle tracks into a very large number of segments, which requires additional CPU time.

[2] Some Monte Carlo codes sample the distance to the nearest collision site in units of particle mean-free-path. Instead of re-sampling the path length at each boundary crossing, the sampled value is reduced according to the ratio of the traveled path length to the MFP in the medium. The remaining fraction is used to determine the distance traveled in the next material zone. Adjusting the sampled path lengths avoids re-evaluating the computationally expensive log-function. This method is not implemented in Serpent.

Download English Version:

https://daneshyari.com/en/article/5475113

Download Persian Version:

https://daneshyari.com/article/5475113

Daneshyari.com