Monte Carlo simulation: Optimization of Computer memory and time

Sachin Shet[1] and K.V. Subbaiah[2]
[1]Junior Research Fellow, [2]Consultant Professor
Manipal Centre for Natural Sciences
Manipal University
Manipal-576104, Karnataka
Email: iamsachinshet@gmail.com

Ref:Stream/ Basic Science/BS-RF005

Technical Report

Work is done under the project:

DAE-BRNS Project Ref. No. :36(4)/14/40/2015/36077

This work was presented in the seminar Manipal Research Colloquium held at Manipal on 06/04/2017.

Abstract:

Monte Carlo technique is widely employed for solving practical radiation transport problems to arrive at a precise solution in a complex geometrical arrangement of materials. This technique is based on the simulation of a “history of a particle or radiation”, which comprises of tracking all the possible interaction events of particle randomly occurring with matter from its birth (source point for primary particle, collision point of production for secondary particle) till its death (absorption or escape from the system). Although the outcome of each history is prone to statistical uncertainty, the repetition of such a history over a large number would result in the prediction of mean behavior, which is an expected answer for the problem. The simulation of history of a particle calls for the availability of copious quantity of random numbers in the range (0 to 1) and the probability density functions (PDF) or Cumulative distribution functions (CDF), which are deduced from the interaction processes of radiation with matter. The first task can be easily overcome, since there are number of random generators available in the open literature meant for different operating systems satisfying the basic characteristics of random numbers. However, CDFs required are not unique and depend on the type of radiation and the composition of the medium and hence not available in the directly useable form. These have to be generated from the nuclear data table such as ENDF (Evaluated Nuclear Data File), which are available free of cost internationally. The main difficulty with the use of nuclear data table is the number of data points which vary widely from one nuclide to the other. For instance in the case of neutron transport, hydrogen as few thousands data points in the energy grid ranging from 0 to 20 MeV, whereas Uranium as few millions points to cover the same energy range. Therefore storage of data and how it is effectively used in Monte Carlo simulations place a crucial role.

In the present work, the usage of computer memory in handling voluminous unstructured data is optimized by the use of dynamic allocation and de-allocation instead of priori fixed allocation, and it is accomplished with the use of memory address pointers. The sorting and editing (deletion and insertion of members) of huge data table scan be done at a faster speed with the help of doubly linked list, which does not search for elements in sequential order. The details of this work will be presented in sequel.

I. Introduction

The prediction of radiation quantity as it traverses through material media is essential while employing the nuclear radiations for the betterment of human kind. From the radiation quantity, it is possible to deduce many of the parameters of practical interest such energy absorbed, heat generated, damage potential and so on. There are two distinct methods of solution (predicting the quantity of radiation) in vogue namely (i) Deterministic method and (ii) Monte Carlo method. The former method is ideally suited for simple arrangement of materials in geometrical figures. And, the latter method is employed, although it is known to consume a lot of computation time, in situations where the materials are arranged in complex configurations. Many of the systems encountered in day to day applications are composite and complex thereby a lot of interest in use of Monte Carlo method. As a result, rapid development is taking place in inventing new algorithms in Monte Carlo method as well as efficient programming techniques to reduce the computational time. Random numbers and probability tables are the two essential parameters of all Monte Carlo Methods. The basic principle under lying in this technique is the simulation of a "history of a particle or radiation", which comprises of tracking all the physical interaction events occurring randomly with matter from its birth (source point for primary particle, collision point of production for secondary particle) till its death (absorption or escape from the system). Although the outcome of each history is prone to statistical uncertainty, the repetition of such a history over a large number would result in the prediction of mean behavior, which is an expected answer to the problem. The solution accuracy (precision) mainly depends on the sample size. However, it is often experienced for problems of large dimensions having many independent variables, the desired accuracy of the solution cannot be accomplished by just increasing the sample size. To circumvent this difficulty, one has to resort to advanced Monte Carlo methods, which employ variance reduction techniques. Suitability of a particular technique largely depends on the problem and judicious selection of it is an art which demands considerable experience of the user.

In the present work, two basic aspects of Monte Carlo technique pertaining to neutron transport are investigated. They are identification of fast and efficient pseudo random number generators and optimal usage of computer memory in handling voluminous amount of unstructured data. Sorting of both of these issues will make the Monte Carlo method as an attractive alternative for solving radiation transport problems.

II. Monte Carlo Method:

Monte Carlo methods are a class of computational algorithms that rely on repeated random sampling of physical and mathematical systems to compute their results. Major components of Monte Carlo methods are as follows:

- *Random number generator* (RNG)
- *Sampling of Probability distribution functions*
- *Scoring /tally specification*
- *Error estimation*
- *Variance reduction techniques*
- *Parallelization and vectorization*

First two components are discussed in detail under the results and discussion section.

III. Results and Discussions

As mentioned earlier, the investigations are limited to two aspects of Monte Carlo method, in particular reference to solving neutron transport problems with optimization of CPU time and computer random access memory utilization. It is known billions of random numbers may be used to solve any practical problem and the selection of efficient and fast algorithms of RNGs will reduce the CPU time consumed. Towards, this end search has been carried out and several RNG routines are investigated and the analysis presented below.

III.a) Random number generators-CPU time optimization

Random number generators are the back bone of all Monte Carlo techniques. In fact, there are no true random number generators. However, even if there exist any, it is not useful for programming purposes on two counts: (i) these will not be of use for developers for program debugging purposes, and (ii) the end results will be fluctuating which does not throw any light on the correctness of the Monte Carlo program. Therefore, all developers of Monte Carlo program rely on pseudo random number generators in the interval (0 to 1), which are fast and provide reproducible sequence satisfying minimal statistical checks such as uniformity, minimum correlation, long period etc. Hereafter, the word pseudo will be dropped and referred as random numbers.

In the present work 9 random number generating functions available in the published literature, which reportedly generate random numbers uniformly in the interval (0 to 1) and must have long cycle length(Number after which repetition of same sequence starts). All of them need starting integer(s) called seed(s) and the output will be a single or double precision real number in the interval (0 to 1). As we deal with millions of random numbers, the time required to generate this random numbers by the computer play an important role in the Monte Carlo simulation. For the 9 random number generating functions reported to have satisfied minimal statistical tests, we calculate the average and the time required to generate 100 million numbers is presented in the table-1.

Table-1. Random number generating functions and time required for 100 million

Description	Seeds	Cycle Length	Average	Time (sec)
1. r4_random (s1, s2, s3). Linear Congruential 32 bit generator.[1]	**No of Seeds: 3** **1< S1, S2, S3 < 30000**	**6.9×10^{12}**	**0.4998**	**9.6250**
2. r4_uni (s1, s2). Linear Congruential 32 bit generator.[1]	**No of Seeds: 2** **1< S1,S2 <2147483562**	**2.3×10^{18}**	**0.5000**	**7.0156**
3. r8_random (s1, s2, s3). Linear Congruential 64 bit generator.[1]	**No of Seeds: 3** **1 < S1, S2, S3 < 30000**	**6.9×10^{12}**	**0.4998**	**11.3906**
4. r8_uni (s1, s2). Linear Congruential 64 bit generator.[1]	**No of Seeds:2** **1< S1,S2 <2147483562**	**2.3×10^{18}**	**0.5000**	**8.0625**
5. r4_uniform_01 (seed2). Multiplicative pseudo random number generator.[2] [3] [4] [5] 32 bit generator	**No of Seeds: 1** **1< S1 <2147483562**	**NA**	**0.4997**	**5.2655**
6. **r8_uniform_01 (seed3).** Multiplicative pseudo random number generator.[2] [3] [4] [5] 64 bit generator	**No of Seeds: 1** **1< S1 <2147483562**	**NA**	**0.4997**	**8.5156**
7. **r8_uniform_02 (seed1).** Multiplicative pseudo random number generator.[2] [3] [4] [5] 64 bit generator	**No of Seeds:1** **1< S1 <2147483562**	**NA**	**0.4997**	**18.5625**
8. ranC(idumC). Multiplicative pseudo random number generator.[6] It is a 32 bit generator, and it uses intrinsic MIN MAX functions.	**No of Seeds:1** **S1 = Negative integer**	**$>2 \times 10^{18}$**	**0.4999**	**28.4062**
9. ranE(idumE). Multiplicative pseudo random number generator. [6] Uses absolute value function and is a 32 bit generator.	**No of Seeds:1** **S1 = Negative integer**	**$>2 \times 10^{18}$**	**0.5003**	**18.6094**

From the above tables it is seen that the average and time required to generate random numbers varies from one function to another function. Further, it can be observed that the 1st and the 3rd linear congruential generators which uses 3 seeds took more time compared to the 2nd and 4th linear congruential generators which uses only 2 seeds. By using more seeds, steps required to generate a random number will be more hence it take more time to generate a set of random numbers. The 9th Multiplicative pseudo random number generator have large period (>$2x10^{18}$) compared to other function as it is using absolute value with the other normal operations. It can be inferred from the above table, that the 5th takes minimum amount of time for generating random numbers can be used in Monte Carlo Programs.

III.b) CPU time and Unstructured Data storage optimization

Another basic requirement in Monte Carlo simulation is Probability Density functions (PDF) and Cumulative Distribution functions (CDF). For neutron transport, these are deduced based on the physical interactions that take place between neutron and the nuclide. The basic interaction data is presented in the form of tables spanning the neutron energy interval in the range of 0 to 20 MeV. Since the magnitude of interaction probability is not smooth varying function of neutron energy and hence needs to be tabulated at close points of energy. Further, the amount of data tabulated for a nuclide varies from one nuclide to another. For instance, hydrogen has few thousand data points whereas uranium has few millions.

To store and access such unstructured data which varies widely from one nuclide to another consume a large amount of computer memory in conventional programming method. However, the use of "ALLOCATABLE" statement existing in modern FORTRAN programming, allows the user to dynamically allocate the required memory space and thus preventing the reservation a large chunk of unused memory of a computer. Further, the statement of "DEALLOCATE" enables one to reuse the same memory again and again for different purposes.

Another operation which often required is searching for the particular element of data in the huge tabular data. As a customary practice, the search begins with start index till the end of data. This method of searching calls for a lot of CPU time if the data element to be searched lies at the other end of data table. However, if one uses binary search algorithm the number of operations required can be reduced drastically. For instance in a table of natural numbers, to find the interval where "N" lies takes "N" operational steps in sequential method whereas in the binary search method the number is only $\log_2 N$. By adopting binary search method in Monte Carlo simulation, the CPU required can be drastically reduced. Suppose one wants to find a number 999,999,999 in 10^9 natural numbers, in a sequential operation the CPU time required will be 1000 second whereas it will be 32 micro-seconds in a binary search. The same is illustrated in the Fig.1.

To find 999999999 in a sorted list of numbers from 1 to 10^9:

Sequential search: If we try to search by sequential method, it will be taking 999,999,999 steps to reach the target value 999,999,999.

Binary search:

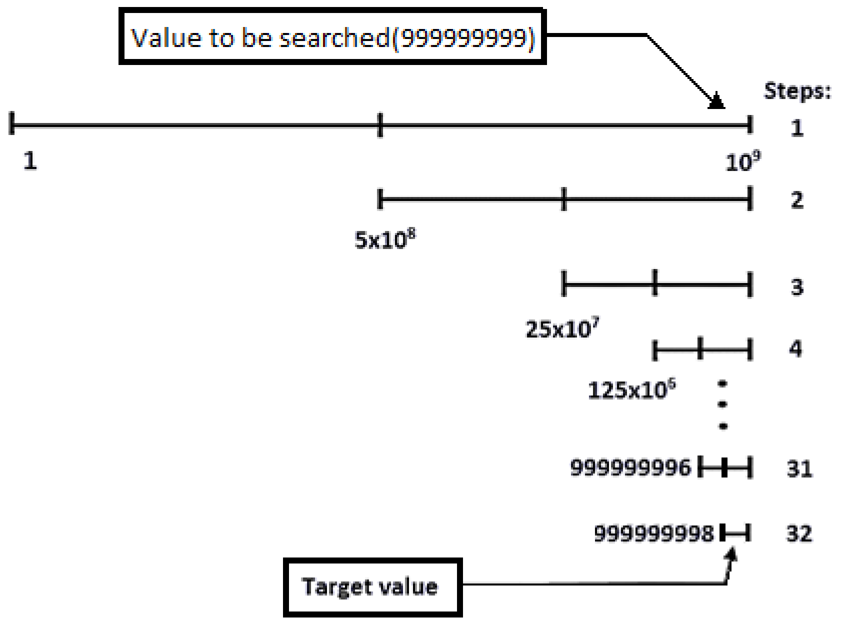

Fig.1. Illustration of "Binary Search" algorithm

IV. Conclusions:

Monte Carlo is power technique to solve radiation transport problems in arbitrary complex geometrical arrangement of materials. This basic principle adopted in this technique is the simulation of a "history of a particle or radiation", which comprises of tracking all the physical events of particle randomly occurring with matter as it traverses through it. Although the outcome of each history is prone to statistical uncertainty, the repetition of such a history over a large number would result in the prediction of mean behavior, which is an expected answer for the problem. However, the main drawback of this method is the memory and time required to solve a given problem prohibiting its usage. The computer memory and the CPU time are the two important parameters to be addressed to make the Monte Carlo technique attractive. The CPU time can be drastically reduced by the proper selection random number generator and based on this work suitable random number generator has been identified. Use of dynamic allocation and de-allocation, it is possible to reduce the memory usage and by putting the dame memory locations for different purposes.   Further, it is demonstrated use of proper search algorithm reduces CPU time drastically.

Acknowledgements:

Authors express sincere thanks to Dr. Kapil Deo Singh and Dr. S. Anand for the training and for the constant encouragement. And I am thankful for the guidance given by other supervisor as well as the panels especially in our project.

References:

1. Brian Wichman, David Hill,
   Algorithm AS 183: An Efficient and
   Portable Pseudo Random Number Generator,
   Applied Statistics, Volume 31

2. Paul Bratley, Bennett Fox, Linus Schrage,
   A Guide to Simulation,
   Second Edition,
   Springer, 1987,
   ISBN: 0387964673,
   LC: QA76.9.C65.B73.

3. Bennett Fox,
   Algorithm 647:
   Implementation and Relative Efficiency of Quasirandom
   Sequence Generators,
   ACM Transactions on Mathematical Software,
   Volume 12, Number 4, December 1986, pages 362-376.

4. Pierre L'Ecuyer,
   Random Number Generation,
   in Handbook of Simulation,
   edited by Jerry Banks,
   Wiley, 1998,
   ISBN: 0471134031,
   LC: T57.62.H37.

5. Peter Lewis, Allen Goodman, James Miller,
   A Pseudo-Random Number Generator for the System/360,
   IBM Systems Journal,
   Volume 8, 1969, pages 136-143.

6. Numerical Recipes in Fortran
   The art of scientific computing
   William H. Press, Saul A. Teukolsky
   William T. Vetterling, Brian P. Flannery

7. Particle-transport simulation with the Monte Carlo method
   Carter, L.L. ; Cashwell, E.D.

8. Monte Carlo N–Particle Transport Code System
   Contributed by: Los Alamos National Laboratory, Los Alamos, New Mexico

9. Monte Carlo methods
   Malvin H. Kalos

The work was carried out under the DAE-BRNS Project,
Sanction No: 36(4)/14/40/2015/36077.