Progress in NUCLEAR SCIENCE and TECHNOLOGY, Vol. 2, pp.716-721 (2011)

ARTICLE

A Monte Carlo Method for Calculation of the Dynamic Behaviour of Nuclear Reactors

Bart L. SJENITZER* and J. Eduard HOOGENBOOM

Delft University of Technology, Delft, The Netherlands

In safety calculations of nuclear reactors, dynamic calculations are of great interest. This type of calculation is mainly done by making an approximation and then using a deterministic code to solve the simplified problem. In this paper a full Monte Carlo method is proposed to perform these calculations on seconds to minutes scale without approximations.

First the sampling of precursor decay is needed. The decay of precursors plays a crucial role in the dynamics of nuclear reactors, but the lifetime of a precursor can go up to 10^2 s. This is much larger than the order of the lifetime of a prompt neutron (10^{-4} s) or the order of a prompt neutron chain (10^{-2} s). Therefore precursors in this simulation are forced to decay according to a modified pdf. This ensures a small variance in the delayed neutron distribution in time.

The simulation scheme has also been adapted for dynamic Monte Carlo. The simulation is now divided in time steps and the ancestor of a particle needs to be tracked throughout the simulation for proper variance estimation. After each time step the system properties can be adjusted.

Finally the dynamic Monte Carlo method is used to calculate the power production in a simple system. The results agree with a point kinetics calculation of the same problem.

KEYWORDS: Monte Carlo, dynamic, time dependent, precursor, neutron chain, prompt neutron, delayed neutron, parallel, safety, simulation

I. Introduction

Monte Carlo simulations are widely used for calculating shielding, eigenvalue and subcritical problems. Although Monte Carlo calculations are costly, they give a result of which the accuracy is exactly known and the calculation costs can be adjusted to meet the required precision.

Nowadays full Monte Carlo codes are only used for the calculation of steady state problems, such as shielding or eigenvalue problems. However, for safety calculations on a nuclear reactor the dynamic behaviour of a reactor is important, for example, the maximum power or temperature reached in an accident scenario. For the calculation of the dynamic behaviour of a nuclear system, deterministic codes exist. These methods need all kinds of approximations in the problem, for example homogenisation, diffusion theory and/or steady state solutions in combination with point kinetics.[1] These approximations introduce uncertainty and the size of this uncertainty is not exactly known, but it is estimated using expensive verification experiments. These experiments are usually still an approximation of the real system, this makes it hard to determine the accuracy of the method. Also hybrid stochastic/deterministic methods are developed and in these methods the shape function is calculated using Monte Carlo, but the time evolution is still approximated using point kinetics.[2–4]

In this paper a full Monte Carlo method is proposed that can be used to calculate the dynamic behaviour of a nuclear system. This system may be sub- or super-critical. The method can simulate delayed neutrons as well as prompt neutrons and can calculate the power production at any time interval. The main focus of this paper is to demonstrate the simulation scheme and to show the feasibility of such a calculation, in contrast to other papers such as Zhitnik et al.,[5] who did not elaborate on the methods used, but mainly focused on the results achieved.

II. Dynamic Monte Carlo Method

1. Precursors

For the implementation of the dynamic Monte Carlo method, there are a few challenges that need to be met. The first challenge is to sample the decay of precursors in such a way that delayed neutrons are being generated constantly. The importance of a constant production of delayed neutrons is shown when comparing the lifetime of a precursor with the lifetime of a prompt neutron chain.

The lifetime of a precursor can vary from an order 10^{-2} s up to 10^2 s. At the end of its lifetime a precursor decays, producing a neutron. This delayed neutron starts a prompt neutron chain, which may produce a new precursor. If a system is exactly critical, on average one precursor per chain will be produced. The prompt neutron chain will die out at some point, but a new chain is started by the delayed neutron that is produced by the new precursor.

To calculate the average lifetime of a prompt neutron chain, first the probability that a neutron produces a new prompt neu-

*Corresponding author, E-mail: B.L.Sjenitzer@TUDelft.nl

tron is needed. This is given by:

$$P_f = k_{eff}(1-\beta) \tag{1}$$

Here k_{eff} is the effective multiplication factor and β is the fraction of delayed neutrons.

If the system is critical ($k_{eff} = 1$) the probability that a chain of prompt neutrons has a length of n neutrons is:

$$P(n) = P_f^{(n-1)}(1-P_f) \tag{2}$$

If the system is not prompt super critical, the average chain length can be calculated using:

$$\overline{n} = \sum_{n=1}^{\infty} nP(n) = \frac{1}{1-P_f} \tag{3}$$

For a critical system with $\beta = 0.007$, the average chain length becomes 143 prompt neutrons. Now assume that the lifetime of a neutron is on average 10^{-4} s and therefore the lifetime of a neutron chain is order 10^{-2} s. This is significantly shorter than the lifetime of a precursor.

The power production only takes place during the lifetime of the prompt neutron chain and after a prompt neutron chain there can be a long time with no power production or any other tally, until a delayed neutron starts a new chain. This creates much variance as illustrated in **Fig. 1**. Therefore it is better to have more often a delayed neutron.

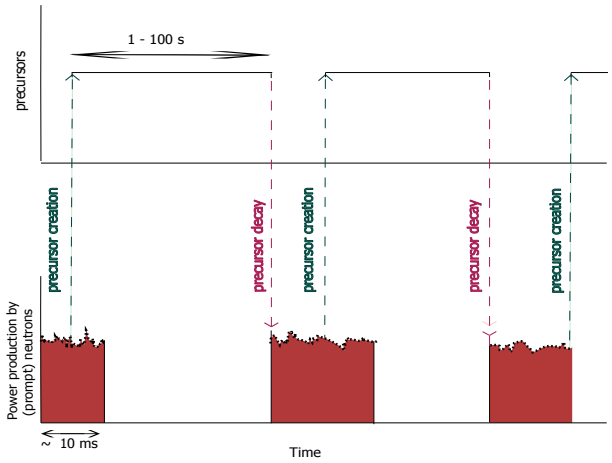

Fig. 1 The difference in lifetime between precursors and prompt neutron chains creates variance in the power production.

A way to solve this problem is to force precursors to decay at regular intervals. To do this the neutron produced by the precursor that is forced to decay must be weighted to ensure a fair game. If the simulation time is divided in intervals and a precursor is forced to have a decay in every time interval there is a new prompt neutron chain in every time interval. This is described by Legrady and Hoogenboom.[6)]

The probability that the precursor has a forced decay at time t inside a time interval between t_1 and $t_1 + \Delta t$ is chosen to be uniformly distributed:

$$\bar{p}(t) = \frac{1}{\Delta t} \tag{4}$$

The probability of a precursor of delayed group i having natural decay is

$$p_i(t) = \lambda_i e^{-\lambda_i(t-t_0)} \tag{5}$$

When using multiple precursor groups these groups can be combined and the probability then becomes:

$$p(t) = \sum \frac{\beta_i}{\beta} \lambda_i e^{-\lambda_i(t-t_0)} \tag{6}$$

This implies that all precursor groups are combined in a single precursor particle and this particle incorporates the decay probability from all groups. However, it should be noticed that, because this precursor particle combines the different decay rates, it no longer has a pure exponential decay.

To keep an unbiased game the weight of the resulting decay must be reduced, so the probability of the forced decay times the weight equals the probability of a natural decay at that time. The weight of the delayed neutron becomes:

$$w_n(t) = \frac{p(t)}{\bar{p}(t)} = \Delta t \sum \frac{\beta_i}{\beta} \lambda_i e^{-\lambda_i(t-t_0)} \tag{7}$$

Now every precursor will produce a new prompt neutron chain in every time interval.

Precursor Spatial Distribution

When initializing a system, first a criticality calculation is done until the source has converged. This is the steady state neutron distribution and from this distribution the precursor and prompt neutron distribution can be calculated. The number of precursors can be calculated using

$$\frac{\partial C_i}{\partial t} = \beta_i \nu \Sigma_f \phi(\mathbf{r},t) - \lambda_i C_i(\mathbf{r},t) \tag{8}$$

The number of precursors at a location $\mathbf{r}$ in steady state then becomes

$$C_{i0}(\mathbf{r}) = \frac{\beta_i}{\lambda_i} \nu \Sigma_f \phi(\mathbf{r}) \tag{9}$$

and for all precursor groups together this becomes

$$C_0(\mathbf{r}) = \frac{\beta}{\lambda^b} \nu \Sigma_f \phi(\mathbf{r}) \tag{10}$$

where λ^b is the inverse averaged λ:

$$\lambda^b = \frac{\beta}{\sum \frac{\beta_i}{\lambda_i}} \tag{11}$$

The fraction of prompt neutrons at location $\mathbf{r}$ now becomes

$$\frac{n_0(\mathbf{r})}{n_0(\mathbf{r}) + C_0(\mathbf{r})} = \frac{\frac{1}{\mathsf{v}}\phi(\mathbf{r})}{\frac{1}{\mathsf{v}}\phi(\mathbf{r}) + \frac{\beta}{\lambda^b}\nu\Sigma_f\phi(\mathbf{r})} = \frac{1}{1 + \frac{\beta}{\lambda^b}\mathsf{v}\nu\Sigma_f} \tag{12}$$

Here v is the neutron speed.

Precursor Time Distribution

With exponential decay a particle has no age. There is always the same probability that a particle has its next decay, no matter what time it has lived before. In this case however as stated before a combined precursor particle does not have pure exponential decay.

The precursors from different groups decay with their own decay constant. Therefore the importance of a precursor group in the combined particle varies over time:

$$\frac{C_i(t)}{C(t)} = \frac{\frac{\beta_i}{\beta} e^{-\lambda_i t}}{\sum_j \frac{\beta_j}{\beta} e^{-\lambda_j t}} \tag{13}$$

When the particle is created, $t = 0$, the fraction per precursor group is simply $\frac{\beta_i}{\beta}$. The time evolution of the importance per groups is shown in **Fig. 2**.

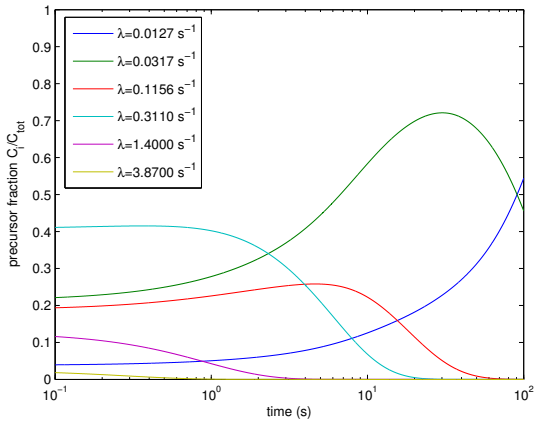

Fig. 2 The ratio between the precursors groups in a combined precursor particle. All groups together add up to 1, but over time different precursor groups are most important.

In a stationary case, the fraction of precursors per group remains the same, this number can be calculated combining Eqs. (9) and (10):

$$\frac{C_{i0}}{C_0} = \frac{\beta_i}{\beta} \frac{\lambda^b}{\lambda_i} \tag{14}$$

To achieve this in a Monte Carlo simulation the combined precursor particles should be started with an age between $-\infty$ and 0. This way the ratios between the different precursor groups are correct.

Now the effective weight of the precursor is altered since it has already had a decay probability before $t = 0$. The effective weight can be calculated by subtracting the decay probability between the birth of the particle at t_0 and the starting time $t = 0$:

$$w_{\text{prec},0} = 1 - \int_{t_0}^{0} \sum \frac{\beta_i}{\beta} \lambda_i e^{-\lambda_i (t - t_0)} \mathrm{d}t = \sum \frac{\beta_i}{\beta} e^{\lambda_i t_0} \tag{15}$$

Here $t_0 < 0$ and

$$\sum_{\text{all precursors}} w_{\text{prec},0} = C_0 \tag{16}$$

2. Calculation Scheme

Monte Carlo codes that are now used, are made for calculating steady-state problems. In a dynamic problem, however, the system can change in time. Not only the system properties change, such as temperature and materials, also the neutron flux will vary. For a Monte Carlo code to cope with the new system, the calculation scheme needs to be adapted.

To ensure that the Monte Carlo method can adapt to its changing environment, the simulation is split in time intervals. The size of these time steps can be chosen freely, but all precursors are forced to have a decay in every time step. If a particle crosses the time boundary of a time step, it is stored for the next time interval.

After each time interval the system properties can be adjusted, for example via coupling with a thermohydraulics code. In this case the dynamic Monte Carlo code outputs a power profile to the thermohydraulics code and the thermohydraulics code then calculates a new temperature profile. Then the Monte Carlo code continues with the next time interval, taking into account the cross sections at the new temperatures.

Parallel Calculations

To use this calculation scheme also in parallel, the results should be independent from the number of processors used. To do this, the random numbers used need to be the same with any number of processors. This is done by assigning every particle with a number and generate a seed depending on this number. In the dynamic Monte Carlo scheme a large number of new particles can be produced, but it is unknown what the number of new particles is. A processor does not know what the number of newly produced particles is on another processor, so it is impossible to assign a particle number to these particles. Therefore a newly produced particle needs to continue with the random number sequence of the parent particle to ensure that the same random numbers are used independent of the number of processors.

At the beginning of a time interval all particles get a new particle number from which a random seed is produced. Then the particles are distributed over the available processors together with the particle numbers. Next, all processors will simulate particles and all their progeny and if a particle reaches the end of the time interval it will be stored. When all particles are finished the stored particles are returned to the master process, which will combine the stored particles in the right order and assign the particles a new seed and redistribute them over the processors. This is shown in **Fig. 3**.

Tallies

Tallying in a dynamic calculation is also different from tallying in a steady-state calculation; in a steady-state calculation there is no explicit time-factor. To calculate a tally with a time factor, such as flux or power production a trick is needed. A source neutron can be considered as a neutron per second. This introduces a time dimension into the equation.

For a dynamic calculation this is not needed. There is an explicit time factor and the tallying becomes more straight forward. The total amount of neutrons or energy deposited per time interval can be calculated.

Another difference is the normalisation of the results. In a time-independent calculation the results are normalised to one source neutron per second. In a criticality calculation this is a normalisation to the number of neutrons in a cycle. For a dynamic calculation, however, the results need to be normalised to the total neutron weight at the beginning of the first time interval.

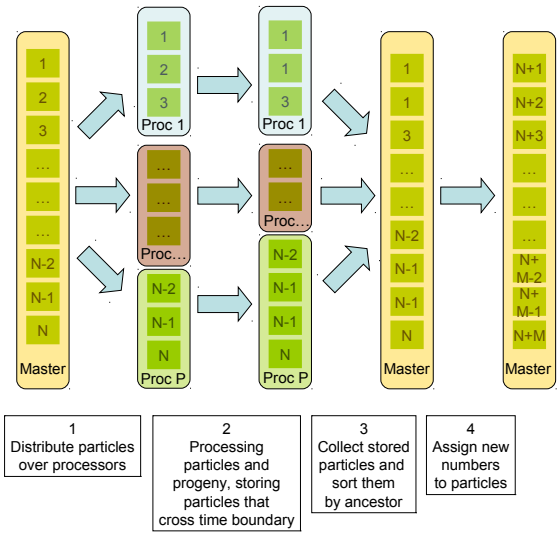

Fig. 3 In a parallel calculation, the master processor distributes the N particles over all P processors. These processors simulate the particles, generating M new particles for the next time step. These particles are sent back to the master processor and the master sorts them by ancestry. Then the particles are assigned a new number for the next time step.

Also to calculate the variance in the tally, the original neutron needs to be known. The time intervals are not statistically independent, but there is a covariance between the time intervals. Therefore calculating the variance per particle in a time interval would underestimate the actual variance. Instead the contribution per original neutron needs to be tallied including all its progeny and the variance per original neutron must be calculated.

3. Variance Reduction

This new calculating structure requires new variance reduction techniques. The number of neutrons can increase or decrease rapidly in a dynamic calculation and prompt neutron chains play an important role in this type of calculation.

One of the new variance reduction techniques is the sampling of the precursor decay as described in Sect. 1 The precursors are forced to produce a delayed neutron in every time interval ensuring low variance in the tallies per time interval. The average weight of the resulting neutrons is best to be kept around the average weight of the neutrons present. This can be achieved by increasing the weight of the precursors using Russian roulette.

Adjusted Weight Windows

In the dynamic Monte Carlo scheme it is also possible to adjust the weight windows after each time interval. This way the simulation can adjust to varying circumstances. For example, if the neutron flux increases the weight windows can be set at higher thresholds making the calculation cost similar for all time intervals.

Also when a particle is stored at a time boundary, Russian roulette is applied which gives all surviving particles the same weight. This includes Russian roulette on precursors, so the expected weight of the delayed neutrons is also the same. Now the starting weight of all particles in a time interval is similar, reducing the variance.

Prompt Neutron Chain

When there is variance in the prompt neutron chain length this will also increase the variance in the tallies, since the prompt neutron chains will, in most cases, stay in one time interval. Therefore variance in the chain length, creates variance in the number of neutrons and thus in the scores.

To calculate the variance in chain length the average squared chain length is needed:

$$\overline{n^2} = \sum_{n=1}^{\infty} n^2 P(n) = \frac{P_f + 1}{(Pf - 1)^2} \tag{17}$$

Combining this with Eq.(3) the variance can be calculated.

$$\sigma^2 = \overline{n^2} - \overline{n}^2 = \frac{P_f}{(P_f - 1)^2} \tag{18}$$

Now if $P_f \rightarrow 1$ the variance becomes infinite and it is even worse when considering the fact that a neutron can also produce more than one new neutron.

Implicit Fission

A method to reduce the variance in the prompt neutron chains is to use implicit fission. This technique is similar to implicit capture, but instead of always having a next scattering event, a particle always produces a new fission neutron. The weight of this neutron is equal to the probability of producing a new fission neutron. Together with implicit capture this implies that there are two particles to continue after each collision. The big increase in particle numbers is compensated by using Russian roulette on particles that have too small importance.

III. Numerical Example

This is very nice in theory, but does it work in practice? To demonstrate the dynamic Monte Carlo scheme a simple sample problem has been set up. In this problem there is a small rectangular box of fissile material. The box is 10 by 12 by 24 centimetre and is placed in vacuum. The system properties are given in **Table 1**. The system has one energy group and the neutrons have a speed of 2.2×10^4 cm/s. There are six precursor groups; the properties of the precursors are given in **Table 2**.

Now the system is started and after ten seconds there is a reactivity insertion: Σ_a of the system is decreased from 0.5882 cm^{-1} to 0.5870 cm^{-1}. After a prompt jump the neutron population will start to grow exponentially. Then at $t = 40$ s the system is returned to the critical state and after a negative prompt jump the system will return to a new stable state.

The simulation is done with 1×10^6 particles and the time interval is 100ms. The Russian roulette threshold is set at 0.25 times the average neutron weight and the survival weight is set at 2 times the Russian roulette threshold.

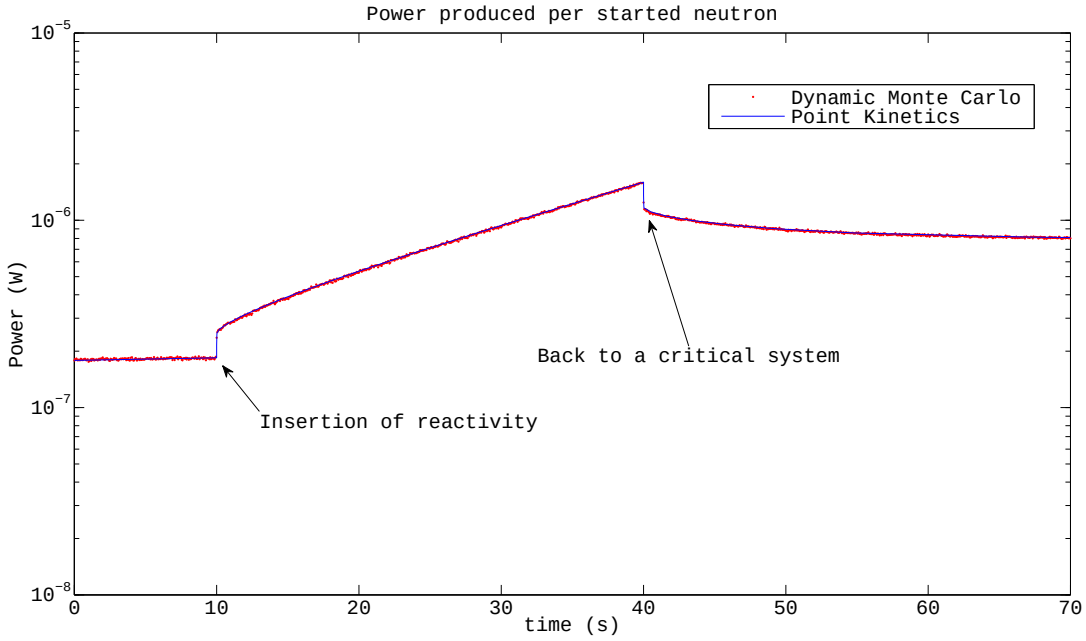

Fig. 4 A critical system has a reactivity insertion at t=10 s. After a prompt jump the power starts to increase further, until the reactivity is set back to 0 at t=40 s. Now the system returns to a new stable state. The dynamic Monte Carlo simulation agrees with the point-kinetics analysis of the system.

Table 1 Material properties that have been used in the test problem. The box is made out of a homogeneous material

Material properties
Σ_t = 1 cm^{-1}
Σ_f = 0.25 cm^{-1}
Σ_s = 0.4118 cm^{-1}
ν = 2.5
β = 0.00685

Table 2 The precursors are divided in six families, here the fractions and decay constants per precursor family i are given. Also the total delayed fraction and average decay constant are shown.

Group	λ (s^{-1})	β
1	0.0127	0.00026
2	0.0317	0.001459
3	0.1156	0.001288
4	0.311	0.002788
5	1.4	0.000877
6	3.87	0.000178
av/tot	0.0784	0.00685

To verify the results produced by the Dynamic Monte Carlo code, the same problem has been calculated using a point kinetics code. The results are plotted in **Fig. 4** and the results agree perfectly with each other.

IV. Summary and Further work

In this paper a method has been proposed to do a full Monte Carlo simulation on a dynamic system. To incorporate the effect of delayed neutrons in the model, without creating large variance, precursors are forced to produce a delayed neutron at regular intervals.

Also the simulation scheme has been adapted for a system that can change in time. This scheme can be seen as a hybrid between a steady-state shielding calculation and a criticality calculation. Instead of cycles, this simulation is divided in time intervals. After each time interval the system properties and the variance reduction can be adjusted.

In a dynamic Monte Carlo simulation there is also a large variance in the length of the prompt neutron chains. To reduce this variance implicit fission has been applied. Implicit fission works similar to implicit capture and creates a new fission neutron at every collision. This technique reduces the variance, but the calculation time increases.

The scheme has been demonstrated in a test case and the Monte Carlo method produces the same results as a point kinetics calculation. There is still some variance visible in the dynamic Monte Carlo solution, this is mainly due to the variance in the prompt-neutron chain length. Although a few techniques have been used to reduce this variance, more research is needed on this topic.

Acknowledgement

This work is partially funded by the Integrated Project NURISP (contract n^o 232124) in the 7th Euratom Framework Programme of the European Union.

References

1) S. Goluoglu, H. L. Dodds, "A Time-Dependent, Three-Dimensional Neutron Transport Methodology," *Nucl. Sci. Eng.*, **139**, 248-261 (2001).
2) C. Bentley, R. DeMeglio, M. Dunn *et al.*, "Development of a Hybrid Stochastic/Deterministic Method for Transient, Three-Dimensional Neutron Transport," *Transactions ANS*, **76**, 221-223 (1997).
3) M. Shayesteh, M. Shahriari, "Calculation of time-dependent neutronic parameters using Monte Carlo method,"*Ann. Nucl. Energy*, **36**, 901-909 (2009).
4) N. Z. Cho, S. Yun, "Space-Time Kinetics via Monte Carlo Method," *Proc. PHYSOR-2008*, Sep. 14-19, 2008, Interlaken, Switzerland (2008), [CD-ROM].
5) A. K. Zhitnik, N. V. Ivanov, V. E. Marshalkin *et al.*, "The TDMCC Monte Carlo Capability for Spatial Kinetics Calculations of Reactor Cores," *ANS winter meeting 2004*, (2004)
6) D. Legrady, J. E. Hoogenboom, "Scouting the feasibility of Monte Carlo reactor dynamics simulations," *Proc. PHYSOR-2008*, Sep. 14-19, 2008, Interlaken, Switzerland (2008), [CD-ROM].