Computational Methods for Efficient Nuclear Data Management in Monte Carlo Neutron Transport Simulations

by

Jonathan A. Walsh
B.S., Nuclear Engineering (2012)
Oregon State University

Submitted to the Department of Nuclear Science and Engineering
in partial fulfillment of the requirements for the degree of
Master of Science in Nuclear Science and Engineering
at the
MASSACHUSETTS INSTITUTE OF TECHNOLOGY
September 2014

Author ..... Signature redacted .....
Jonathan A. Walsh
Department of Nuclear Science and Engineering
Signature redacted August 21, 2014

Certified by..... .....
Benoit Forget
Associate Professor of Nuclear Science and Engineering
Signature redacted Thesis Supervisor

Certified by..... .....
Kord S. Smith
KEPCO Professor of the Practice of Nuclear Science and Engineering
Signature redacted Thesis Supervisor

Accepted by ....
Mujid S. Kazimi
TEPCO Professor of Nuclear Engineering
Chair, Department Committee on Graduate Students

Computational Methods for Efficient Nuclear Data Management in Monte Carlo Neutron Transport Simulations

by

Jonathan A. Walsh

Submitted to the Department of Nuclear Science and Engineering
on August 21, 2014, in partial fulfillment of the
requirements for the degree of
Master of Science in Nuclear Science and Engineering

Abstract

This thesis presents the development and analysis of computational methods for efficiently accessing and utilizing nuclear data in Monte Carlo neutron transport code simulations. Using the OpenMC code, profiling studies are conducted in order to determine the types of nuclear data that are used in realistic reactor physics simulations, as well as the frequencies with which those data are accessed. The results of the profiling studies are then used to motivate the conceptualization of a nuclear data server algorithm aimed at reducing on-node memory requirements through the use of dedicated server nodes for the storage of infrequently accessed data. A communication model for this algorithm is derived and used to make performance predictions given data access frequencies and assumed system hardware parameters. Additionally, a new, accelerated approach for rejection sampling the free gas resonance elastic scattering kernel that reduces the frequency of zero-temperature elastic scattering cross section data accesses is derived and implemented. Using this new approach, the runtime overhead incurred by an exact treatment of the free gas resonance elastic scattering kernel is reduced by more than 30% relative to a standard sampling procedure used by Monte Carlo codes. Finally, various optimizations of the commonly-used binary energy grid search algorithm are developed and demonstrated. Investigated techniques include placing kinematic constraints on the range of the searchable energy grid, index lookups on unionized material energy grids, and employing energy grid hash tables. The accelerations presented routinely result in overall code speedup by factors of 1.2-1.3 for simulations of practical systems.

Thesis Supervisor: Benoit Forget
Title: Associate Professor of Nuclear Science and Engineering

Thesis Supervisor: Kord S. Smith
Title: KEPCO Professor of the Practice of Nuclear Science and Engineering

Acknowledgments

This material was supported by a Department of Energy Nuclear Energy University Programs Graduate Fellowship. Partial support was provided 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.

First, I thank my advisors, Ben Forget and Kord Smith. Without an identifiable exception, interactions with Professor Forget have been enjoyable, and, more importantly, productive. I greatly appreciate his allowance and encouragement of unplanned research endeavors which bore fruit and are featured in this thesis. The influence of Professor Smith's broad and deep knowledge of nuclear reactor physics phenomena and computational methods is evident in key portions of this work. Further, his recountals of marathon work sessions are acknowledged for their inspiration of another generation of coffee-swilling, office couch-sleeping nuclear engineers.

I also thank core members of the OpenMC code development team, Paul Romano, Bryan Herman, and Nick Horelik. Their work product and willingness to offer constructive advice, which have helped this graduate student and many others, are much appreciated.

Contents

List of Figures

List of Tables

Chapter 1

Overview

Many challenges are encountered in running the kinds of large-scale, highly detailed simulations that are necessary to design advanced nuclear systems and more accurately predict the performance of those already in existence. One of these challenges is the management of large sets of nuclear data on local memory-limited, massively parallel computing architectures. The aims of the work presented in this thesis are the development and analysis of computational methods for the efficient management and utilization of nuclear data in Monte Carlo neutron transport simulations.

In Section 1.1 of this chapter, an introduction to the role that Monte Carlo methods and nuclear data play - and the relations between them - in Monte Carlo simulations is given. Following that introduction, Section 1.2 outlines some of the challenges that are encountered in performing computationally intensive simulations of practical nuclear systems. The general competition between memory requirements and CPU time is discussed with respect to a typical full core nuclear reactor physics simulation. This chapter concludes with an outline of specific computational challenges that are addressed in this thesis. Each particular challenge is briefly described, as are the methods that are developed in this work for its analysis and mitigation.

In Chapter 2 of this thesis, the types of nuclear data that are needed in full core reactor physics simulations, as well as the frequencies with which these data are needed during a simulation, are investigated. The nuclear data access information that is gathered is then used in the conceptualization of an algorithm for more efficiently

managing - from a memory reduction perspective - large nuclear data sets. A communication model for this algorithm is also developed and used to make performance predictions. Then, Chapter 3 presents the derivation and implementation of a new approach for accelerated rejection sampling of the free gas resonance elastic scattering kernel. Verification and runtime results are also given. In Chapter 4, optimizations for the standard energy grid lookup algorithm used by continuous-energy Monte Carlo codes are presented. The theoretical foundations for the search accelerations are laid out as are pertinent details of their implementations. Overall code speedup results are presented for simulations of practical, reactor physics-relevant systems. This thesis concludes with Chapter 5 and its discussion of areas for future research that are related to the present work.

1.1 Nuclear Data in Monte Carlo Neutron Transport Simulations

Monte Carlo methods applied to neutron transport simulations [52] are widely regarded as the gold standard for accurate prediction of the neutronic behavior of nuclear systems. In the context of particle transport, Monte Carlo methods are employed to integrate over the phase-space of a linearized form of the Boltzmann transport equation by using pseudo-random numbers to stochastically sample probability distributions that describe physical phenomena [34]. The outcomes of this sampling describe the behavior of a single particle realization in the system under consideration. If a large number of independent particles are simulated in this manner, quantities describing the average behavior of a system which contains many particles (e.g. reaction rate density) can be accurately calculated by taking the mean numerical values from the collection of simulated particles.

The elegance of the Monte Carlo method is rooted in its allowance of arbitrarily complex probability distributions to describe the elements of a system's behavior. In the simulation of physical systems, this means that there are few inherent ap-

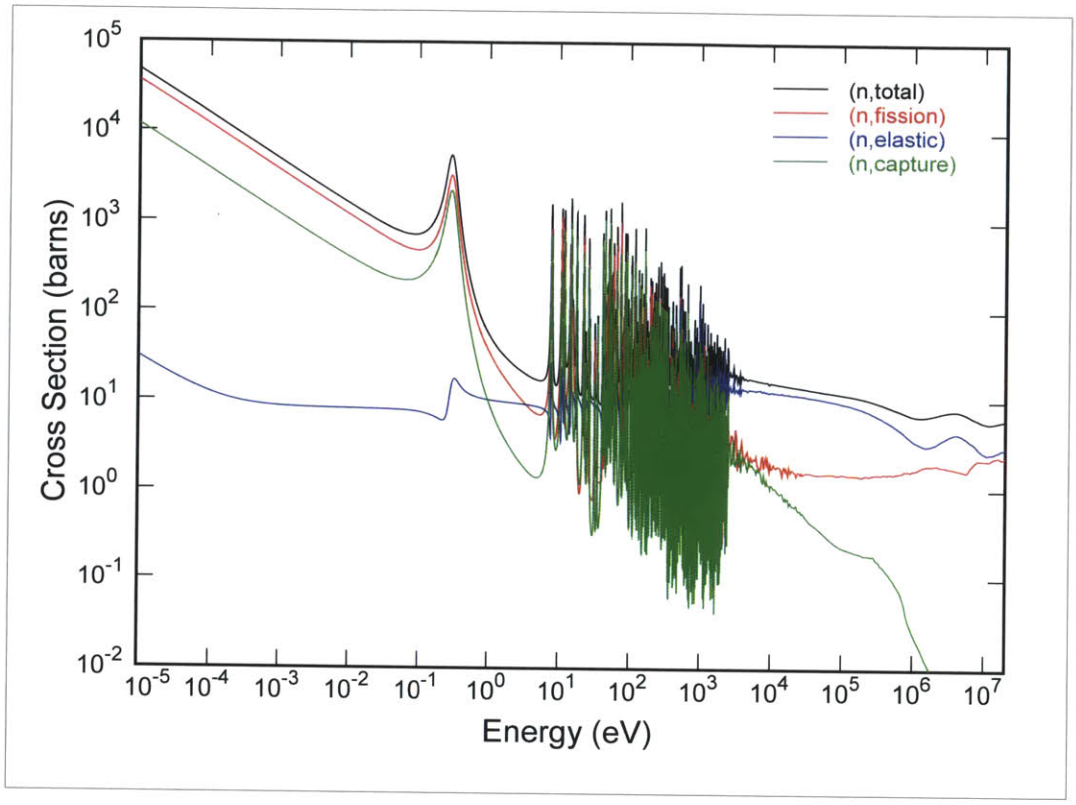

Figure 1-1: Selected ^{239}Pu Reaction Cross Sections [10]

proximations that must be made with regard to physics models or system geometry. The probability distributions that describe the physical phenomena relevant to neutron transport simulations are encoded in nuclear data. These data include neutron reaction cross sections (Figure 1-1) which characterize the probabilities of various neutron-nucleus interactions, secondary angular and energy distributions (Figures 1-2 and 1-3, respectively) which describe the probabilities that a particular reaction will result in secondary particles traveling in a given direction with a given energy, respectively, and many other quantities describing a variety of physical processes.

The nuclear data that are utilized in neutron transport codes go through several processing steps before they are actually used in a simulation. The first of these procedures is nuclear data evaluation. For an individual nuclide, starting with the raw data from theoretical nuclear model calculations and experimental results, an

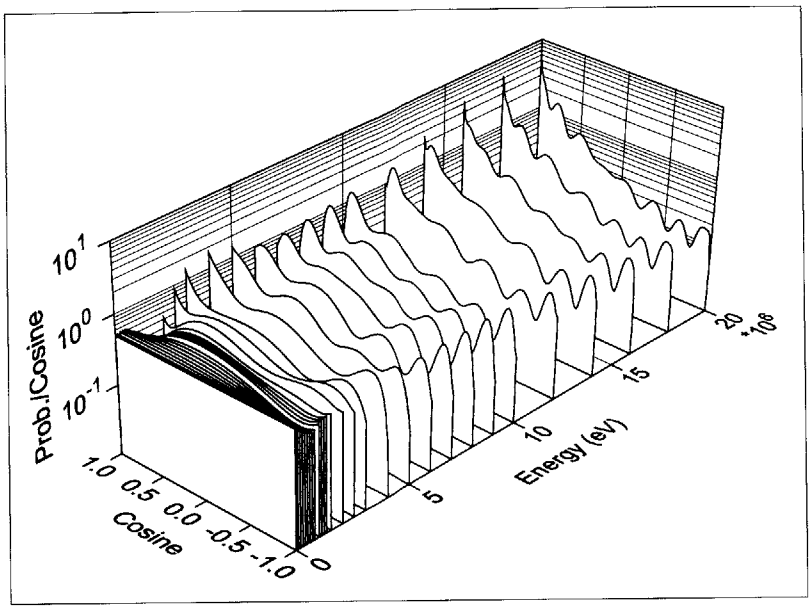

Figure 1-2: ^{239}Pu First Level Inelastic Scattering Secondary Angular Distribution [10]

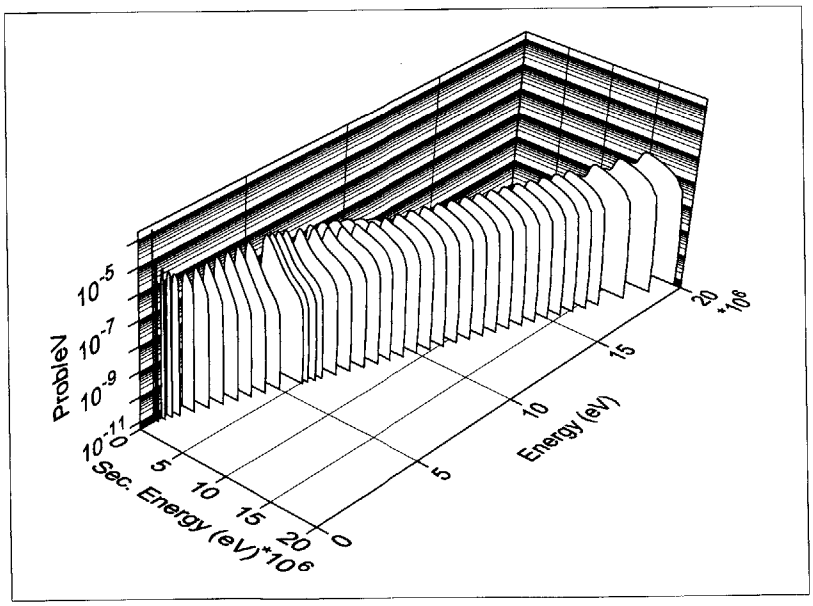

Figure 1-3: ^{239}Pu Continuum Inelastic Scattering Secondary Energy Distribution [10]

Evaluated Nuclear Data Library	Organization
ENDF/B-VII.1 [24]	CSEWG, USA [3]
JEFF-3.2 [8]	NEA[2], International [11]
JENDL-4.0 [61]	JAEA[3], Japan [7]
TENDL-2013 [15]	NRG[4], The Netherlands [12]
CENDL-3.1 [40]	CNDC[5], China [2]
RUSFOND-2010 [14]	IPPE[6], Russia [5]
BROND-2.2 [20]	IPPE, USSR

Table 1.1: Selected Evaluated Nuclear Data File Libraries

evaluated nuclear data file is produced[1]. For a given isotope, several current evaluated nuclear data files may exist as a result of different national and international efforts. These files almost exclusively come in the ENDF-6 format [69], an internationally-utilized standard for evaluated nuclear data files originating from the United States Cross Section Evaluation Working Group (CSEWG) [3], organized out of the US National Nuclear Data Center [9]. CSEWG is also responsible for the production of the US evaluated nuclear data file library. The current version of this library is referred to as ENDF/B-VII.1 [24]. A listing of evaluated nuclear data file libraries, along with the organizations by which they are produced, is provided for reference in Table 1.1.

1.2 Computational Challenges

High-fidelity physical models and arbitrarily complex problem geometries that render problems intractable for deterministic computational methods can be handled relatively easily in Monte Carlo simulations. For this reason, three-dimensional, full core reactor physics simulations, in which highly complex physical processes are occurring throughout a system with highly detailed geometry, provide a seemingly natural area

[1]Collectively, these evaluated nuclear data files are referred to as *evaluated nuclear data libraries.*
[2]Nuclear Energy Agency
[3]Japan Atomic Energy Agency
[4]Nuclear Research and Consultancy Group
[5]China Nuclear Data Center
[6]Institute of Physics and Power Engineering

of application for Monte Carlo methods. However, due to limitations on computational speed and memory, this potential has gone largely unrealized.

Historically, Monte Carlo codes have not been a viable tool for everyday reactor analysis because the types of computers that are required to run detailed, full core Monte Carlo simulations fast enough for practical design purposes simply did not exist. With the advent of massively-parallel architectures, now ubiquitous in high-performance computing, and the push towards exascale supercomputing systems, Monte Carlo methods are increasingly being considered seriously for large reactor physics calculations. However, challenges remain.

1.2.1 A Challenge Problem

The fidelity with which Monte Carlo methods are capable of modeling the physical processes relevant to reactor physics comes at the expense of large data storage requirements. Tally data, which contain the results of a simulation, geometry data, which define the physical dimensions of a system, and nuclear data, which contain necessary input information for a simulation, account for the largest shares of stored data. Memory storage requirements are a limiting factor for realistic reactor physics simulations in which isotopic depletion and temperature feedback effects must be considered. For these calculations, pin-wise reaction rates must be tracked across the core for roughly 400 nuclides in each of approximately 300 axial planes. Each pin may also be divided into upwards of 10 radial rings and 8 azimuthal seqments. The few terabytes of data generated by these tallies far exceeds the available memory per node on virtually all current and proposed supercomputers. The tally server algorithm of Romano, et. al [57] addresses this memory burden by storing tally data on separate server nodes. Large geometry data requirements are also being addressed via domain decomposition methods [45].

This leaves the problem of on-node storage of the hundreds of gigabytes of nuclear data required to even run realistic full core simulations in the first place. For each of the hundreds of nuclides present in a simulation, point-wise continuous-energy neutron reaction cross sections must be stored, along with secondary energy and angular

distribution data. The problem is exacerbated by transient temperatures, temperature gradients, or materials at different temperatures. In these cases, cross section data must be stored on a fine temperature grid of $5 - 10$ K in order to capture the effects of Doppler broadening [70]. It has recently been suggested that temperature dependence of secondary distribution data should also be considered [63].

1.2.2 Memory Requirements and Simulation Runtime

Several methods have been proposed for treating neutron cross section data and managing the associated memory requirements. Storing cross sections for every nuclide - each with an initially unique grid of energy points - on the same unionized energy grid enables faster cross section calculations by requiring only a single binary search on the unionized grid in order to determine the index of the cross section value for every nuclide in the problem. However, use of a unionized energy grid necessitates the storage of many extraneous energy-cross section value pairs for each nuclide. One option for reducing the cross section memory footprint, while still preserving the computational speed advantages of the unionized grid method, involves eliminating the need for storing data for each nuclide on the same unionized energy grid. The Serpent code [49] utilizes a so-called *double indexing* method in which, for each nuclide, a list of pointers of the same length as the unionized energy grid is indexed to the list of a nuclide's data points. This method has been combined with so-called *grid thinning*, in which unnecessary points are removed from the unionized energy grid itself [48]. While these methods have been somewhat successful in reducing the memory burden of cross section data storage, methods which result in larger reductions are needed.

One of the earliest methods aimed at this sort of large memory reduction involves the use of so-called energy *supergroups*. In this method, the cross section data is decomposed into several energy ranges, or supergroups. All of the particles in a given supergroup are simulated until they are at energies below the bottom energy bound of the group. Then, all of the particles in the next lower energy supergroup are simulated, and so on. The benefit of this method is that the data for only one energy supergroup must be stored at a time. This method can be dated as far back as the

late 1950's and the O5R code [30]. In the years since, a few other codes have utilized energy supergroups including the VIM fast critical assembly code[7] [21], the KENO criticality safety code [31], and the MCV module within the RACER reactor analysis code [65]. Recently, there has been renewed investigation of algorithms which divide the required cross section data into energy supergroups [36].

A handful of other methods for reducing the size of cross section data fall under the scope of so-called *on-the-fly* Doppler broadening. The aim of these methods is to store only a subset of cross section data, often only at a single temperature (e.g. 0 K), and, from that, calculate - as opposed to store - the required higher temperature data. A conceptually simple method could rely on using the SIGMA1 method [32] to Doppler broaden 0 K cross section data on-the-fly, rather than using the NJOY nuclear data processing system [50] - which relies on an early variation of the SIGMA1 algorithm - to generate higher temperature data prior to running the simulation. However, the increased computational time required to numerically perform the Doppler broadening convolution integral on-the-fly makes this method impractical. A more promising method for on-the-fly Doppler broadening relies on series expansions to generate cross section data at desired temperatures from the 0 K data [74]. Another method, based on a cross section rejection sampling scheme [72], has been proposed and implemented in Serpent. The most recent method proposed for on-the-fly Doppler broadening [39] relies on the multipole resonance formalism for cross section data [46] which allows for the direct calculation of a Doppler broadened cross section value at a desired temperature. With calculation times that are comparable, or even reduced, relative to linear-linear interpolation of pre-Doppler broadened point-wise cross section data - as is current practice - this method results in greater memory savings than are obtained with the series expansion or rejection sampling methods. One limitation of the on-the-fly Doppler broadening methods is that they are usually only applicable in the resolved resonance energy region.

[7] In VIM, supergroups are referred to as *energy bands*.

1.3 Methods Development and Analysis Objectives

The purpose of the work undertaken in this thesis is the development of methods for alleviating some of the computational problems associated with large Monte Carlo neutron transport simulations. To this end, three specific issues have been identified and addressed. In the following subsections, descriptions of those problems, and the associated methods that are developed and analyzed to address them, are discussed.

1.3.1 Nuclear Data Storage and Communication

In order to guide and assess the development of methods that are aimed at reducing nuclear data memory requirements in reactor physics simulations, it is essential to have a firm understanding of what nuclear data are actually used in those simulations. To this end, work is directed toward quantifying nuclear data access patterns in full core reactor physics simulations. This involves determining the rates at which nuclear data are accessed with attention to such parameters as data type (e.g. point-wise cross sections, probability tables, secondary distributions, etc.), nuclide, energy range (e.g. thermal, resolved resonance, unresolved resonance, fast, etc.), and temperature. To accomplish this, simulations of the three-dimensional, full core BEAVRS model [44] are run with a version of the OpenMC Monte Carlo neutron transport code [58] modified to record nuclear data accesses. Hot zero-power and hot full-power simulations are performed as a means of gathering some information about the dependence of nuclear data access patterns on temperature. It should be noted, though, that these results do not capture the effects of transient temperatures or multiphysics feedback within a simulation. Beginning-of-core and end-of-core simulations are performed to determine the differences in nuclear data accesses resulting from the increased complexity of isotopic compositions encountered in the simulation of a depleted core.

With a collection of results that describe how nuclear data are being accessed in large simulations, it is possible to develop new algorithms for managing that data more efficiently. Another phase of this work is devoted to the conceptual development of a nuclear data server algorithm for managing the storage and flow of nuclear data

within a simulation. Using the data access pattern information obtained from the full core simulations, the frequencies with which certain data are accessed are cross-referenced with the memory requirements of those data. This helps to identify blocks of data which are infrequently accessed but take up large amounts of memory. To save memory on compute nodes, these data can be stored on server nodes and retrieved only when they are needed by the simulation. The nuclear data server node concept could also be extended for use in conjunction with the existing methods for nuclear data memory reduction, previously discussed, to further reduce memory burdens.

With multiple approaches for reducing the memory required to store cross section data, many of which rely on fundamentally unique methods, it is important to have tools for comparing the approaches which quantify potential benefits and drawbacks. This is especially important because, with most of the proposed methods, there is a tradeoff between reducing memory and increasing calculation and communication costs. With this in mind, the final phase of the nuclear data storage and communication work focuses on developing a model for predicting the performance of the proposed nuclear data sever algorithm. This communication model is similar to the one developed by Romano, et. al [57] for predicting the performance of the tally server algorithm in which large sets of tally data are decomposed and stored on server nodes. The model analyzes the memory reduction that is achievable with the method, as well as computational costs associated with the storage, flow, and utilization of data. Taking into account computer architecture-specific parameters allows the model to predict the algorithm's performance on different platforms. The results of the nuclear data profiling studies and the theoretical development of the nuclear data server model are found in Chapter 2.

1.3.2 Resonance Elastic Scattering Models

The accuracy of Monte Carlo neutron transport simulation results relies on the accurate modeling of the individual neutron-nucleus interactions that occur over the course of a simulation. Resonance elastic scattering is one such interaction process that is known to bias the secondary angle-energy distribution of scattered neutrons if

sufficient detail is not preserved in the physical model employed to describe it. Biases in these secondary distributions can manifest themselves as biases in integral results such as reaction rates and reactivity. These biases are of practical consequence as they can result in significant differences in reactor design metrics such as the fuel temperature coefficient.

The model that has most typically been applied by Monte Carlo codes to the treatment of elastic scattering events approximates the scatterer's 0 K elastic scattering cross section as a constant, independent of incident neutron energy. In the vicinity of scattering resonances this approximation is quite inaccurate as cross section values can vary by orders of magnitude in tight energy intervals. The problem caused by this approximation arises from its effect on the distribution of target nuclei velocities that is sampled when calculating the kinematics of a scattering event. At incident neutron energies just below scattering resonances, allowing for the energy dependence of target nuclei cross sections results in relatively more elastic scattering events with higher-velocity target nuclei. This phenomenon results in a higher fraction of scattering events in which the energy of the secondary neutron is greater than that of the incident (i.e. upscatter). Proposed by Rothenstein [59], and first demonstrated by Becker, et. al [18], the *Doppler broadening rejection correction* (DBRC) is currently the model that is most often used by Monte Carlo codes to account for target nuclei elastic scattering cross section energy dependence, if it is accounted for, at all. This method is computationally expensive as it can result in the rejection sampling of a 0 K resonance in the elastic scattering cross section of target nuclei.

In this work, we develop and analyze an accelerated resonance elastic scattering kernel sampling procedure that avoids the costly 0 K elastic scattering resonance rejection sampling. This is accomplished while still accounting for the energy dependence of cross sections, and, thus, preserving the physics of the free gas resonance elastic scattering model as accurately as is possible. More detailed discussions of the resonance elastic scattering process, its effects in practical problems, and the models commonly used to describe it are given in Chapter 3. The new, accelerated resonance elastic scattering kernel sampling technique is also derived and verified. Results illus-

trating the simulation runtime reductions achievable with the method are shown.

1.3.3 Energy Grid Searches

The neutron reaction cross section data that are used by continuous-energy Monte Carlo neutron transport codes most often come in the form of piece-wise linearly interpolable, point-wise energy-cross section pairs. That is, for each nuclide, the cross section values characterizing the probabilities of various interactions are tabulated on a discrete grid of energy values. In the course of a simulation, a code will need to utilize the cross section values corresponding to the current neutron energy in order to compute the probabilities of various events or calculate contributions to the tallies being recorded in the simulation. Because cross section values are tabulated at points on energy grids unique to each nuclide, determining the correct point on a nuclide's energy grid is a prerequisite for determining any desired nuclide cross section value.

The determination of the location of the current neutron energy value in a one-dimensional array of tabulated energy values for a given nuclide is typically accomplished with a binary search algorithm. This procedure can prove costly, from a simulation runtime perspective, if the system that is being modeled contains a large number of nuclides and/or nuclides with many points in their respective energy grids. This is often the case in realistic simulations which contain hundreds of nuclides that each exhibit resonance cross section structure that must be accounted for on a fine energy grid.

To lighten the computational burden associated with performing these energy grid lookups, three principle techniques, along with a few minor variations to each, are proposed, implemented, and tested, here, as part of this work[8]. Overall code speedup results are shown for simulations of practical systems in which the optimization schemes are applied. Methods for the optimization of the energy grid search algorithm are presented and analyzed in Chapter 4.

[8]In some cases, where noted, the techniques presented here are extensions of previously proposed energy grid search accelerations.

Chapter 2

Nuclear Data Profiling Studies and Server Model Development

In this chapter we discuss the nuclear data memory requirements and access patterns in Monte Carlo full core reactor physics simulations. Profiling studies are conducted with the OpenMC code to determine the types of nuclear data that are accessed in these simulations, as well as the frequencies of their accesses. The memory requirements associated with these nuclear data are also recorded. The results of these memory and data access profiling studies provide the motivation for the conceptualization of a nuclear data server algorithm in which infrequently accessed data are stored off of compute nodes and retrieved from dedicated server nodes only as necessary, when required by the simulation. A theoretical performance model for the proposed nuclear data server algorithm is derived. With the results of the memory and data access profiling studies as input, the model is used to predict the performance of the nuclear data server algorithm across a range of computing platform hardware parameters.

2.1 Introduction

Monte Carlo neutron transport codes rely on a variety of nuclear data to describe the individual physical processes that occur throughout a simulation. Among these

nuclear data are neutron reaction cross sections, which characterize the probabilities of particular neutron-nucleus interactions, secondary angular distributions which give the probabilities of secondary neutrons emerging from an interaction with specific flight directions, and secondary energy distributions which provide the probabilities of a neutron having a specified energy after an interaction. In the types of Monte Carlo neutron transport simulations where memory considerations become a limiting factor, these three types of nuclear data, along with tally data, typically dominate memory requirements.

In order to develop algorithms that more efficiently manage nuclear data, it is important to understand the specific types of data that are actually needed within a simulation and how often those data are accessed. It is also necessary to understand the memory requirements associated with these different data. The determination and analysis of these nuclear data access patterns and memory requirements comprise the first part of the work presented in this chapter. Then, equipped with quantitative information about the access frequencies and memory requirements for different blocks of nuclear data, we turn to the theoretical development of an algorithm for managing these data in a more computationally efficient manner. A communication model for the proposed algorithm is derived and used to make performance predictions. These predictions take into account the access frequency and memory for a given block of nuclear data, as well as computing platform-specific parameters such as network latency, bandwidth, and neutron simulation rate.

2.2 Profiling Studies

In this section, we discuss the profiling studies that are carried out in order to determine the access frequencies and memory requirements associated with different blocks of nuclear data. The modifications that are made to OpenMC for the purpose of carrying these studies out are outlined in Subsection 2.2.1. In Subsection 2.2.2, nuclear data access patterns in full core reactor simulations are presented and discussed. The memory requirements in these same simulations are also given.

2.2.1 Code Modifications to OpenMC

Nuclear data profiling studies for this work are performed with a modified release 0.5.3 of the OpenMC neutron transport code that is capable of recording nuclear data accesses and memory requirements. Specifically, the tally system within OpenMC is modified to tally the accesses of neutron cross section and energy grid data, as well as secondary angular and energy distribution data whenever they occur within a simulation. Slight code modifications are also made to compute and output the memory requirements for these different types of nuclear data. The data access tallies can be filtered by nuclide, by energy range, and by type of data. Said another way, the modified OpenMC code tallies the accesses of a specific type of data (e.g. elastic scattering cross section), in a defined energy range, for a defined nuclide. The desired data access tallies are specified by the user in the tallies.xml input file. At the end of a simulation, the results of these data access tallies are output as any other tally result would be. The use of this expanded tally system in nuclear data profiling studies is discussed below.

2.2.2 Access Patterns and Memory Requirements

In order to quantify the nuclear data access patterns in large Monte Carlo calculations, simulations of the three-dimensional, full core Benchmark for Evaluation and Validation of Reactor Simulations (BEAVRS) [44] model are performed. The accesses of nuclear data in these simulations are tallied and analyzed. Two different variations of the BEAVRS model are examined. The first variation is simply the beginning-of-core (BOC) hot zero-power (HZP) model that is defined in the BEAVRS specification [1]. This model contains fresh fuel and all core materials are at 600 K. The second model is an end-of-core (EOC) hot full-power (HFP) case. In this model, all fuel is at 900 K and a burnup of 20.0 MWd/kg. The nuclide densities for the fuel materials in the EOC HFP model come from the results of depletion calculations performed with the CASMO-5 lattice physics code [56]. For reference, these nuclide density values are provided in Appendix A. Both simulations are performed with the modified

version of release 0.5.3 of OpenMC compiled with release 4.9.0 of the GNU Fortran (GFortran) compiler [4]. Each simulation consists of 100 inactive neutron cycles, 100 active neutron cycles, and 2.5E6 neutrons per cycle. A fission spectrum energy source, spatially uniform over fissionable material, is used to initiate each run.

For each nuclide that is present in a model, tallies are kept for energy grid accesses, all point-wise reaction cross section data accesses, secondary angular and energy distribution accesses, and, where applicable, unresolved resonance region probability table and $S(\alpha, \beta)$ thermal scattering law table accesses[1]. The memory requirements associated with each of these blocks of data are also recorded. Only a very small subset of the many data access tallies and memory values that result from these simulations are shown in this chapter. This is done for the sake of clarity and to place an emphasis on key results.

For the BOC HZP simulations, the nuclear data access frequencies and memory requirements of three nuclides are selected and shown here. Tables 2.1, 2.2, and 2.3 give the results for ^{1}H, ^{235}U, and ^{106}Cd, respectively. As expected for a nuclide that resides in a material of non-negligible density which occupies a large fraction of the problem geometry (i.e. moderator), the nuclear data of ^{1}H are accessed frequently within a simulation. On average, the simulation of a single neutron will require every type of nuclear data that is present for ^{1}H to be accessed more than 10 times. Somewhat similarly, we see in Table 2.2 that the nuclear data present for ^{235}U are accessed many times within a simulation. However, a notable difference relative to the ^{1}H results is the presence of many more reaction types. We can see that the cross section data for many of the more exotic level inelastic scattering reactions are accessed only a handful of times for every 1E3 neutrons simulated, on average. Secondary angular and energy distributions may only be accessed a few times for every 1E6 neutrons. We can see this effect taken to an extreme with the access frequency results for ^{106}Cd presented in Table 2.3. Nearly all nuclear data are accessed zero - or very few - times over the course of the entire simulation of 250E6 active cycle neutrons.

[1]Data accesses are only tallied in active cycles.

Nuclear Data	Memory [B]	Access Frequency [source neutron^{-1}]
(n,elastic) Angular Dist.	10320	1.09192E+01
$S(\alpha, \beta)$	1560896	4.21083E+01
Energy Grid	4720	3.24538E+01
(n,total) Cross Section	4720	3.24538E+01
(n,absorption) Cross Section	4720	3.24538E+01
(n,elastic) Cross Section	4720	3.24538E+01

Table 2.1: BOC HZP ^{1}H Memory Requirements and Access Frequencies

Nuclear Data	Memory [B]	Access Frequency [source neutron^{-1}]
(n,elastic) Angular Dist.	69752	6.09824E-02
(n,n1) Angular Dist.	3912	2.43160E-05
(n,n2) Angular Dist.	3480	8.95400E-05
(n,n3) Angular Dist.	41216	8.01996E-04
(n,n4) Angular Dist.	2728	1.69368E-04
(n,n5) Angular Dist.	3400	2.04292E-04
(n,n6) Angular Dist.	33048	4.19916E-04
(n,n7) Angular Dist.	2520	2.56088E-04
(n,n8) Angular Dist.	26624	1.03548E-04
(n,n9) Angular Dist.	3272	3.10196E-04
(n,n10) Angular Dist.	20408	4.76720E-05
(n,n11) Angular Dist.	2784	8.27440E-05
(n,n12) Angular Dist.	2448	5.10240E-05
(n,n13) Angular Dist.	2856	6.63840E-05
(n,n14) Angular Dist.	2872	9.84200E-05
(n,n15) Angular Dist.	14784	8.15200E-06
(n,n16) Angular Dist.	2968	7.94720E-05

(n,n17) Angular Dist.	14424	8.18800E-06
(n,n18) Angular Dist.	13152	8.22000E-06
(n,n19) Angular Dist.	13176	6.45600E-06
(n,n20) Angular Dist.	12496	5.32800E-06
(n,n21) Angular Dist.	13608	9.78400E-06
(n,n22) Angular Dist.	14264	4.74800E-06
(n,n23) Angular Dist.	13376	3.50760E-05
(n,n24) Angular Dist.	13296	9.86400E-06
(n,n25) Angular Dist.	12960	4.64800E-06
(n,n26) Angular Dist.	12672	4.55600E-06
(n,n27) Angular Dist.	12288	3.52800E-06
(n,n28) Angular Dist.	11776	4.43600E-06
(n,n29) Angular Dist.	11512	5.69600E-06
(n,n30) Angular Dist.	11032	6.74000E-06
(n,n31) Angular Dist.	8096	4.76000E-07
(n,n32) Angular Dist.	11456	6.27600E-06
(n,n33) Angular Dist.	10920	4.28800E-06
(n,n34) Angular Dist.	10680	2.78800E-06
(n,n35) Angular Dist.	10288	2.22000E-06
(n,n36) Angular Dist.	10048	7.64000E-07
(n,n37) Angular Dist.	9928	8.00000E-07
(n,n38) Angular Dist.	9784	8.00000E-07
(n,n39) Angular Dist.	9784	4.80000E-07
(n,n40) Angular Dist.	9368	4.76000E-07
(n,2n) Energy Dist.	53832	2.52680E-05
(n,3n) Energy Dist.	15624	3.20000E-08
(n,fission) Energy Dist.	309296	9.26668E-01
(n,4n) Energy Dist.	2584	0.00000E+00

(n,n1) Energy Dist.	16	2.43160E-05
(n,n2) Energy Dist.	16	8.95400E-05
(n,n3) Energy Dist.	16	8.01996E-04
(n,n4) Energy Dist.	16	1.69368E-04
(n,n5) Energy Dist.	16	2.04292E-04
(n,n6) Energy Dist.	16	4.19916E-04
(n,n7) Energy Dist.	16	2.56088E-04
(n,n8) Energy Dist.	16	1.03548E-04
(n,n9) Energy Dist.	16	3.10196E-04
(n,n10) Energy Dist.	16	4.76720E-05
(n,n11) Energy Dist.	16	8.27440E-05
(n,n12) Energy Dist.	16	5.10240E-05
(n,n13) Energy Dist.	16	6.63840E-05
(n,n14) Energy Dist.	16	9.84200E-05
(n,n15) Energy Dist.	16	8.15200E-06
(n,n16) Energy Dist.	16	7.94720E-05
(n,n17) Energy Dist.	16	8.18800E-06
(n,n18) Energy Dist.	16	8.22000E-06
(n,n19) Energy Dist.	16	6.45600E-06
(n,n20) Energy Dist.	16	5.32800E-06
(n,n21) Energy Dist.	16	9.78400E-06
(n,n22) Energy Dist.	16	4.74800E-06
(n,n23) Energy Dist.	16	3.50760E-05
(n,n24) Energy Dist.	16	9.86400E-06
(n,n25) Energy Dist.	16	4.64800E-06
(n,n26) Energy Dist.	16	4.55600E-06
(n,n27) Energy Dist.	16	3.52800E-06
(n,n28) Energy Dist.	16	4.43600E-06

(n,n29) Energy Dist.	16	5.69600E-06
(n,n30) Energy Dist.	16	6.74000E-06
(n,n31) Energy Dist.	16	4.76000E-07
(n,n32) Energy Dist.	16	6.27600E-06
(n,n33) Energy Dist.	16	4.28800E-06
(n,n34) Energy Dist.	16	2.78800E-06
(n,n35) Energy Dist.	16	2.22000E-06
(n,n36) Energy Dist.	16	7.64000E-07
(n,n37) Energy Dist.	16	8.00000E-07
(n,n38) Energy Dist.	16	8.00000E-07
(n,n39) Energy Dist.	16	4.80000E-07
(n,n40) Energy Dist.	16	4.76000E-07
(n,nc) Energy Dist.	102776	2.38836E-03
Probability Tables	14744	1.43570E+00
Energy Grid	485088	1.46102E+01
(n,total) Cross Section	485088	1.46102E+01
(n,absorption) Cross Section	485088	1.46102E+01
(n,elastic) Cross Section	485088	1.46102E+01
(n,2n) Cross Section	472	1.82884E-04
(n,3n) Cross Section	216	3.40000E-07
(n,fission) Cross Section	485088	1.46102E+01
(n,4n) Cross Section	48	0.00000E+00
(n,n1) Cross Section	3392	5.33812E-03
(n,n2) Cross Section	2936	5.31381E-03
(n,n3) Cross Section	2480	5.22427E-03
(n,n4) Cross Section	2392	4.42227E-03
(n,n5) Cross Section	2032	4.25290E-03
(n,n6) Cross Section	1816	4.04861E-03

(n,n7) Cross Section	1784	3.62870E-03
(n,n8) Cross Section	1728	3.37261E-03
(n,n9) Cross Section	1704	3.26906E-03
(n,n10) Cross Section	1576	2.95886E-03
(n,n11) Cross Section	1528	2.91119E-03
(n,n12) Cross Section	1480	2.82845E-03
(n,n13) Cross Section	1448	2.77742E-03
(n,n14) Cross Section	1408	2.71104E-03
(n,n15) Cross Section	1360	2.61262E-03
(n,n16) Cross Section	1344	2.60447E-03
(n,n17) Cross Section	1240	2.52466E-03
(n,n18) Cross Section	1192	2.50557E-03
(n,n19) Cross Section	1152	2.45054E-03
(n,n20) Cross Section	1104	2.36760E-03
(n,n21) Cross Section	1056	2.27668E-03
(n,n22) Cross Section	1016	2.18854E-03
(n,n23) Cross Section	984	2.11343E-03
(n,n24) Cross Section	952	1.91758E-03
(n,n25) Cross Section	928	1.77356E-03
(n,n26) Cross Section	888	1.63154E-03
(n,n27) Cross Section	864	1.54262E-03
(n,n28) Cross Section	848	1.46365E-03
(n,n29) Cross Section	824	1.30335E-03
(n,n30) Cross Section	800	1.17742E-03
(n,n31) Cross Section	776	1.09175E-03
(n,n32) Cross Section	752	1.01334E-03
(n,n33) Cross Section	720	7.16592E-04
(n,n34) Cross Section	688	5.87344E-04

(n,n35) Cross Section	664	5.09420E-04
(n,n36) Cross Section	640	4.60040E-04
(n,n37) Cross Section	616	4.30720E-04
(n,n38) Cross Section	600	4.02688E-04
(n,n39) Cross Section	576	3.76408E-04
(n,n40) Cross Section	560	3.57244E-04
(n,nc) Cross Section	1296	2.38836E-03
(n,level) Cross Section	3392	0.00000E+00

Table 2.2: BOC HZP ^{235}U Memory Requirements and Access Frequencies

Nuclear Data	Memory [B]	Access Frequency [source neutron^{-1}]
(n,elastic) Angular Dist.	46264	4.80000E-08
(n,2n) Angular Dist.	160	0.00000E+00
(n,3n) Angular Dist.	160	0.00000E+00
(n,na) Angular Dist.	160	0.00000E+00
(n,np) Angular Dist.	160	0.00000E+00
(n,nd) Angular Dist.	160	0.00000E+00
(n,n1) Angular Dist.	13264	4.00000E-09
(n,n2) Angular Dist.	1048	0.00000E+00
(n,n3) Angular Dist.	1624	0.00000E+00
(n,n4) Angular Dist.	1048	0.00000E+00
(n,n5) Angular Dist.	1048	0.00000E+00
(n,n6) Angular Dist.	976	0.00000E+00
(n,n7) Angular Dist.	1048	0.00000E+00
(n,n8) Angular Dist.	11232	0.00000E+00
(n,n9) Angular Dist.	1048	0.00000E+00
(n,n10) Angular Dist.	880	0.00000E+00

(n,n11) Angular Dist.	880	0.00000E+00
(n,nc) Angular Dist.	2176	0.00000E+00
(n,2n) Energy Dist.	2584	0.00000E+00
(n,3n) Energy Dist.	272	0.00000E+00
(n,na) Energy Dist.	14264	0.00000E+00
(n,np) Energy Dist.	6128	0.00000E+00
(n,nd) Energy Dist.	1944	0.00000E+00
(n,n1) Energy Dist.	16	4.00000E-09
(n,n2) Energy Dist.	16	0.00000E+00
(n,n3) Energy Dist.	16	0.00000E+00
(n,n4) Energy Dist.	16	0.00000E+00
(n,n5) Energy Dist.	16	0.00000E+00
(n,n6) Energy Dist.	16	0.00000E+00
(n,n7) Energy Dist.	16	0.00000E+00
(n,n8) Energy Dist.	16	0.00000E+00
(n,n9) Energy Dist.	16	0.00000E+00
(n,n10) Energy Dist.	16	0.00000E+00
(n,n11) Energy Dist.	16	0.00000E+00
(n,nc) Energy Dist.	6024	0.00000E+00
Energy Grid	87392	8.25072E-04
(n,total) Cross Section	87392	8.25072E-04
(n,absorption) Cross Section	87392	8.25072E-04
(n,elastic) Cross Section	87392	8.25072E-04
(n,2n) Cross Section	4760	0.00000E+00
(n,3n) Cross Section	136	0.00000E+00
(n,na) Cross Section	13640	4.00000E-09
(n,np) Cross Section	8416	0.00000E+00
(n,nd) Cross Section	2256	0.00000E+00

(n,n1) Cross Section	14616	4.00000E-09
(n,n2) Cross Section	13800	0.00000E+00
(n,n3) Cross Section	13560	0.00000E+00
(n,n4) Cross Section	13176	0.00000E+00
(n,n5) Cross Section	12976	0.00000E+00
(n,n6) Cross Section	12952	0.00000E+00
(n,n7) Cross Section	12928	0.00000E+00
(n,n8) Cross Section	12888	0.00000E+00
(n,n9) Cross Section	12768	0.00000E+00
(n,n10) Cross Section	12744	0.00000E+00
(n,n11) Cross Section	12720	0.00000E+00
(n,nc) Cross Section	12696	0.00000E+00

Table 2.3: BOC HZP ^{106}Cd Memory Requirements and Access Frequencies

The general trends observed in the data access frequency results of the BOC HZP case are largely unchanged when the EOC HFP case is considered instead. Many of the same cross section and secondary distribution data for the more exotic threshold reactions, which occupy appreciable space in memory, are accessed quite infrequently in either case. With respect to nuclear data requirements, though, an important difference between the models is that the burned fuel in the EOC HFP case contains dozens of fission products and actinides, each accompanied by cross section and secondary distribution data that must be stored and accessed in the simulation, that are not present in the clean fuel of the BOC HZP case. Tables 2.4 and 2.5 give the data access frequencies and memory requirements for ^{240}U and ^{239}Pu, respectively. We can see that the ^{239}Pu nuclear data - cross sections as well as secondary distributions - are accessed relatively frequently throughout the EOC HFP simulation, as is expected for a common actinide. This is in stark contrast to the data for the much less abundant ^{240}U, the vast majority of which are never accessed

in the simulation. This and other results of these nuclear data profiling studies can be exploited in the development of more efficient data management algorithms.

Nuclear Data	Memory [B]	Access Frequency [source neutron^{-1}]
(n,elastic) Angular Dist.	65576	0.00000E+00
(n,n1) Angular Dist.	43816	0.00000E+00
(n,n2) Angular Dist.	29496	0.00000E+00
(n,n3) Angular Dist.	24224	0.00000E+00
(n,n4) Angular Dist.	26848	0.00000E+00
(n,n5) Angular Dist.	8312	0.00000E+00
(n,n6) Angular Dist.	5088	0.00000E+00
(n,n7) Angular Dist.	5568	0.00000E+00
(n,n8) Angular Dist.	3832	0.00000E+00
(n,n9) Angular Dist.	9216	0.00000E+00
(n,n10) Angular Dist.	7520	0.00000E+00
(n,n11) Angular Dist.	5864	0.00000E+00
(n,n12) Angular Dist.	3968	0.00000E+00
(n,n13) Angular Dist.	5744	0.00000E+00
(n,n14) Angular Dist.	9024	0.00000E+00
(n,n15) Angular Dist.	4440	0.00000E+00
(n,n16) Angular Dist.	5472	0.00000E+00
(n,n17) Angular Dist.	3840	0.00000E+00
(n,n18) Angular Dist.	4344	0.00000E+00
(n,n19) Angular Dist.	5472	0.00000E+00
(n,n20) Angular Dist.	5560	0.00000E+00
(n,n21) Angular Dist.	4344	0.00000E+00
(n,n22) Angular Dist.	7280	0.00000E+00
(n,n23) Angular Dist.	3536	0.00000E+00
(n,n24) Angular Dist.	3432	0.00000E+00
(n,n25) Angular Dist.	3328	0.00000E+00

(n,n26) Angular Dist.	3224	0.00000E+00
(n,n27) Angular Dist.	3224	0.00000E+00
(n,n28) Angular Dist.	3120	0.00000E+00
(n,n29) Angular Dist.	3120	0.00000E+00
(n,n30) Angular Dist.	3120	0.00000E+00
(n,n31) Angular Dist.	3120	0.00000E+00
(n,n32) Angular Dist.	3016	0.00000E+00
(n,n33) Angular Dist.	3016	0.00000E+00
(n,n34) Angular Dist.	2912	0.00000E+00
(n,n35) Angular Dist.	2912	0.00000E+00
(n,n36) Angular Dist.	2808	0.00000E+00
(n,n37) Angular Dist.	2808	0.00000E+00
(n,n38) Angular Dist.	2808	0.00000E+00
(n,n39) Angular Dist.	2808	0.00000E+00
(n,n40) Angular Dist.	2808	0.00000E+00
(n,2n) Energy Dist.	94024	0.00000E+00
(n,3n) Energy Dist.	51528	0.00000E+00
(n,f) Energy Dist.	104	0.00000E+00
(n,nf) Energy Dist.	104	0.00000E+00
(n,2nf) Energy Dist.	104	0.00000E+00
(n,4n) Energy Dist.	20184	0.00000E+00
(n,3nf) Energy Dist.	88	0.00000E+00
(n,n1) Energy Dist.	16	0.00000E+00
(n,n2) Energy Dist.	16	0.00000E+00
(n,n3) Energy Dist.	16	0.00000E+00
(n,n4) Energy Dist.	16	0.00000E+00
(n,n5) Energy Dist.	16	0.00000E+00
(n,n6) Energy Dist.	16	0.00000E+00

(n,n7) Energy Dist.	16	0.00000E+00
(n,n8) Energy Dist.	16	0.00000E+00
(n,n9) Energy Dist.	16	0.00000E+00
(n,n10) Energy Dist.	16	0.00000E+00
(n,n11) Energy Dist.	16	0.00000E+00
(n,n12) Energy Dist.	16	0.00000E+00
(n,n13) Energy Dist.	16	0.00000E+00
(n,n14) Energy Dist.	16	0.00000E+00
(n,n15) Energy Dist.	16	0.00000E+00
(n,n16) Energy Dist.	16	0.00000E+00
(n,n17) Energy Dist.	16	0.00000E+00
(n,n18) Energy Dist.	16	0.00000E+00
(n,n19) Energy Dist.	16	0.00000E+00
(n,n20) Energy Dist.	16	0.00000E+00
(n,n21) Energy Dist.	16	0.00000E+00
(n,n22) Energy Dist.	16	0.00000E+00
(n,n23) Energy Dist.	16	0.00000E+00
(n,n24) Energy Dist.	16	0.00000E+00
(n,n25) Energy Dist.	16	0.00000E+00
(n,n26) Energy Dist.	16	0.00000E+00
(n,n27) Energy Dist.	16	0.00000E+00
(n,n28) Energy Dist.	16	0.00000E+00
(n,n29) Energy Dist.	16	0.00000E+00
(n,n30) Energy Dist.	16	0.00000E+00
(n,n31) Energy Dist.	16	0.00000E+00
(n,n32) Energy Dist.	16	0.00000E+00
(n,n33) Energy Dist.	16	0.00000E+00
(n,n34) Energy Dist.	16	0.00000E+00

(n,n35) Energy Dist.	16	0.00000E+00
(n,n36) Energy Dist.	16	0.00000E+00
(n,n37) Energy Dist.	16	0.00000E+00
(n,n38) Energy Dist.	16	0.00000E+00
(n,n39) Energy Dist.	16	0.00000E+00
(n,n40) Energy Dist.	16	0.00000E+00
(n,nc) Energy Dist.	178960	0.00000E+00
Probability Tables	12416	1.36171E+00
Energy Grid	121376	1.44365E+01
(n,total) Cross Section	121376	1.44365E+01
(n,absorption) Cross Section	121376	1.44365E+01
(n,elastic) Cross Section	121376	1.44365E+01
(n,2n) Cross Section	936	0.00000E+00
(n,3n) Cross Section	568	0.00000E+00
(n,f) Cross Section	121376	0.00000E+00
(n,nf) Cross Section	1808	0.00000E+00
(n,2nf) Cross Section	608	0.00000E+00
(n,4n) Cross Section	312	0.00000E+00
(n,3nf) Cross Section	424	0.00000E+00
(n,n1) Cross Section	2776	0.00000E+00
(n,n2) Cross Section	2600	0.00000E+00
(n,n3) Cross Section	2504	0.00000E+00
(n,n4) Cross Section	2416	0.00000E+00
(n,n5) Cross Section	2344	0.00000E+00
(n,n6) Cross Section	2312	0.00000E+00
(n,n7) Cross Section	2280	0.00000E+00
(n,n8) Cross Section	2248	0.00000E+00
(n,n9) Cross Section	2200	0.00000E+00

(n,n10) Cross Section	2184	0.00000E+00
(n,n11) Cross Section	2160	0.00000E+00
(n,n12) Cross Section	2144	0.00000E+00
(n,n13) Cross Section	2128	0.00000E+00
(n,n14) Cross Section	2104	0.00000E+00
(n,n15) Cross Section	2080	0.00000E+00
(n,n16) Cross Section	2056	0.00000E+00
(n,n17) Cross Section	2032	0.00000E+00
(n,n18) Cross Section	2016	0.00000E+00
(n,n19) Cross Section	2000	0.00000E+00
(n,n20) Cross Section	1976	0.00000E+00
(n,n21) Cross Section	1952	0.00000E+00
(n,n22) Cross Section	1904	0.00000E+00
(n,n23) Cross Section	1856	0.00000E+00
(n,n24) Cross Section	1784	0.00000E+00
(n,n25) Cross Section	1720	0.00000E+00
(n,n26) Cross Section	1648	0.00000E+00
(n,n27) Cross Section	1600	0.00000E+00
(n,n28) Cross Section	1552	0.00000E+00
(n,n29) Cross Section	1496	0.00000E+00
(n,n30) Cross Section	1456	0.00000E+00
(n,n31) Cross Section	1424	0.00000E+00
(n,n32) Cross Section	1392	0.00000E+00
(n,n33) Cross Section	1304	0.00000E+00
(n,n34) Cross Section	1264	0.00000E+00
(n,n35) Cross Section	1232	0.00000E+00
(n,n36) Cross Section	1208	0.00000E+00
(n,n37) Cross Section	1184	0.00000E+00

(n,n38) Cross Section	1160	0.00000E+00
(n,n39) Cross Section	1136	0.00000E+00
(n,n40) Cross Section	1112	0.00000E+00
(n,nc) Cross Section	1936	0.00000E+00

Table 2.4: EOC HFP ^{240}U Memory Requirements and Access Frequencies

Nuclear Data	Memory [B]	Access Frequency [source neutron^{-1}]
(n,elastic) Angular Dist.	42864	9.98317E-03
(n,n1) Angular Dist.	21592	2.42152E-04
(n,n2) Angular Dist.	21888	1.83016E-04
(n,n3) Angular Dist.	16240	3.61480E-05
(n,n4) Angular Dist.	16040	1.84320E-05
(n,n5) Angular Dist.	13760	6.34800E-06
(n,n6) Angular Dist.	3584	2.12800E-05
(n,n7) Angular Dist.	13696	4.36400E-06
(n,n8) Angular Dist.	2904	8.93600E-06
(n,n9) Angular Dist.	2496	1.00360E-05
(n,n10) Angular Dist.	2368	5.49600E-06
(n,n11) Angular Dist.	2392	3.54800E-06
(n,n12) Angular Dist.	4936	1.15320E-05
(n,n13) Angular Dist.	2656	1.22600E-05
(n,n14) Angular Dist.	18168	9.18000E-06
(n,n15) Angular Dist.	2576	6.84400E-06
(n,n16) Angular Dist.	17952	4.87200E-06
(n,n17) Angular Dist.	2216	3.73600E-06
(n,n18) Angular Dist.	2880	7.60000E-07
(n,n19) Angular Dist.	16232	1.11600E-06

(n,n20) Angular Dist.	17640	7.08000E-07
(n,n21) Angular Dist.	15424	1.55200E-06
(n,n22) Angular Dist.	14264	9.16000E-07
(n,n23) Angular Dist.	13376	6.33200E-06
(n,n24) Angular Dist.	13296	1.88000E-06
(n,n25) Angular Dist.	12960	8.04000E-07
(n,n26) Angular Dist.	12672	8.16000E-07
(n,n27) Angular Dist.	12288	7.04000E-07
(n,n28) Angular Dist.	11776	8.28000E-07
(n,n29) Angular Dist.	11512	1.18400E-06
(n,n30) Angular Dist.	11032	1.31200E-06
(n,n31) Angular Dist.	8096	6.80000E-08
(n,n32) Angular Dist.	11456	1.11200E-06
(n,n33) Angular Dist.	10920	7.40000E-07
(n,n34) Angular Dist.	10680	5.24000E-07
(n,n35) Angular Dist.	10288	4.20000E-07
(n,n36) Angular Dist.	10048	1.08000E-07
(n,n37) Angular Dist.	9928	1.48000E-07
(n,n38) Angular Dist.	9784	1.84000E-07
(n,n39) Angular Dist.	9784	9.60000E-08
(n,n40) Angular Dist.	9368	7.20000E-08
(n,2n) Energy Dist.	50352	1.48400E-06
(n,3n) Energy Dist.	14432	4.00000E-09
(n,fission) Energy Dist.	324760	4.91352E-01
(n,4n) Energy Dist.	1792	0.00000E+00
(n,n1) Energy Dist.	16	2.42152E-04
(n,n2) Energy Dist.	16	1.83016E-04
(n,n3) Energy Dist.	16	3.61480E-05

(n,n4) Energy Dist.	16	1.84320E-05
(n,n5) Energy Dist.	16	6.34800E-06
(n,n6) Energy Dist.	16	2.12800E-05
(n,n7) Energy Dist.	16	4.36400E-06
(n,n8) Energy Dist.	16	8.93600E-06
(n,n9) Energy Dist.	16	1.00360E-05
(n,n10) Energy Dist.	16	5.49600E-06
(n,n11) Energy Dist.	16	3.54800E-06
(n,n12) Energy Dist.	16	1.15320E-05
(n,n13) Energy Dist.	16	1.22600E-05
(n,n14) Energy Dist.	16	9.18000E-06
(n,n15) Energy Dist.	16	6.84400E-06
(n,n16) Energy Dist.	16	4.87200E-06
(n,n17) Energy Dist.	16	3.73600E-06
(n,n18) Energy Dist.	16	7.60000E-07
(n,n19) Energy Dist.	16	1.11600E-06
(n,n20) Energy Dist.	16	7.08000E-07
(n,n21) Energy Dist.	16	1.55200E-06
(n,n22) Energy Dist.	16	9.16000E-07
(n,n23) Energy Dist.	16	6.33200E-06
(n,n24) Energy Dist.	16	1.88000E-06
(n,n25) Energy Dist.	16	8.04000E-07
(n,n26) Energy Dist.	16	8.16000E-07
(n,n27) Energy Dist.	16	7.04000E-07
(n,n28) Energy Dist.	16	8.28000E-07
(n,n29) Energy Dist.	16	1.18400E-06
(n,n30) Energy Dist.	16	1.31200E-06
(n,n31) Energy Dist.	16	6.80000E-08

(n,n32) Energy Dist.	16	1.11200E-06
(n,n33) Energy Dist.	16	7.40000E-07
(n,n34) Energy Dist.	16	5.24000E-07
(n,n35) Energy Dist.	16	4.20000E-07
(n,n36) Energy Dist.	16	1.08000E-07
(n,n37) Energy Dist.	16	1.48000E-07
(n,n38) Energy Dist.	16	1.84000E-07
(n,n39) Energy Dist.	16	9.60000E-08
(n,n40) Energy Dist.	16	7.20000E-08
(n,nc) Energy Dist.	104416	2.56696E-04
Probability Tables	54320	1.51588E+00
Energy Grid	426272	1.44365E+01
(n,total) Cross Section	426272	1.44365E+01
(n,absorption) Cross Section	426272	1.44365E+01
(n,elastic) Cross Section	426272	1.44365E+01
(n,2n) Cross Section	992	1.88600E-05
(n,3n) Cross Section	472	6.80000E-08
(n,fission) Cross Section	426272	1.44365E+01
(n,4n) Cross Section	80	0.00000E+00
(n,n1) Cross Section	3368	8.67260E-04
(n,n2) Cross Section	2936	6.25108E-04
(n,n3) Cross Section	2888	4.42092E-04
(n,n4) Cross Section	2768	4.05944E-04
(n,n5) Cross Section	2728	3.87512E-04
(n,n6) Cross Section	2624	3.81164E-04
(n,n7) Cross Section	2576	3.59884E-04
(n,n8) Cross Section	2528	3.55520E-04
(n,n9) Cross Section	2464	3.46584E-04

(n,n10) Cross Section	2416	3.36548E-04
(n,n11) Cross Section	2384	3.31052E-04
(n,n12) Cross Section	2352	3.27504E-04
(n,n13) Cross Section	2328	3.15972E-04
(n,n14) Cross Section	2304	3.03712E-04
(n,n15) Cross Section	2280	2.94532E-04
(n,n16) Cross Section	2256	2.87688E-04
(n,n17) Cross Section	2216	2.82816E-04
(n,n18) Cross Section	2176	2.79080E-04
(n,n19) Cross Section	1984	2.71148E-04
(n,n20) Cross Section	1944	2.67456E-04
(n,n21) Cross Section	1888	2.59228E-04
(n,n22) Cross Section	1864	2.55172E-04
(n,n23) Cross Section	1824	2.48080E-04
(n,n24) Cross Section	1776	2.27428E-04
(n,n25) Cross Section	1736	2.11796E-04
(n,n26) Cross Section	1688	1.96144E-04
(n,n27) Cross Section	1656	1.86000E-04
(n,n28) Cross Section	1624	1.76900E-04
(n,n29) Cross Section	1576	1.59200E-04
(n,n30) Cross Section	1536	1.44044E-04
(n,n31) Cross Section	1504	1.33644E-04
(n,n32) Cross Section	1472	1.24328E-04
(n,n33) Cross Section	1384	8.86000E-05
(n,n34) Cross Section	1344	7.27040E-05
(n,n35) Cross Section	1304	6.32560E-05
(n,n36) Cross Section	1280	5.72400E-05
(n,n37) Cross Section	1256	5.36080E-05

(n,n38) Cross Section	1224	5.03320E-05
(n,n39) Cross Section	1200	4.70440E-05
(n,n40) Cross Section	1176	4.46600E-05
(n,nc) Cross Section	2168	2.56696E-04
(n,level) Cross Section	3368	0.00000E+00

Table 2.5: EOC HFP ^{239}Pu Memory Requirements and Access Frequencies

2.3 Nuclear Data Server Model

The observation that, in large reactor physics simulations, much of the nuclear data that are loaded at initialization are accessed infrequently, or not at all, motivates the development of a nuclear data server algorithm in which unneeded or rarely needed nuclear data are stored on dedicated server nodes, rather than the compute nodes. Data servers have previously been applied to Monte Carlo neutron transport simulations in the context of off-node tally data storage [57].

The off-node storage of nuclear data provides a reduction in on-node memory requirements. However, there are computational costs associated with the data server concept. Namely, when the off-node data are finally needed within a simulation, they must be communicated to a compute node[2]. In order to better understand these costs, we derive a communication model for the proposed algorithm.

2.3.1 Derivation

In this subsection we present the derivation of a communication model for the nuclear data server algorithm concept. Taking into account the size in memory of individual

[2]Instead of storing nuclear data in the primary memory of server nodes and communicating it to compute nodes, as needed, a variant of the proposed algorithm might call for the access of data directly from secondary memory. The computational costs of this alternative (i.e. latency and data transfer times associated with reading from a hard disk drive) can be analyzed in a manner analogous to that of the proposed procedure.

blocks of nuclear data, the frequencies with which those data are accessed, and system parameters for the computing platform on which the simulation is being carried out, the model can be used to make performance predictions for the algorithm. The derivation of this model closely parallels the derivation of a similar model for the tally server algorithm [57].

Symbol	Meaning
c	compute processes
p	total processes
s	data server processes
D_i	i^{th} block of nuclear data memory [B]
$\bar{A}_i$	mean accesses of i^{th} nuclear data block per particle
t_0	batch simulation time w/ p processes [s]
t	batch simulation time w/ c processes [s]
N	particles per batch
t_c	compute time per batch [s]
t_s	server communication time per batch [s]
L	network latency [s]
B	network bandwidth [B/s]
μ	mean neutron simulation time [s]

Table 2.6: Nuclear Data Server Communication Model Variables

We start by defining relevant variables that will be used in the derivation in Table 2.6. Assuming ideal parallel scaling, we have

$$t_c = t_0 = \frac{N\mu}{p} \tag{2.1}$$

in the absence of nuclear data servers and

$$t_c = \frac{N\mu}{c} \tag{2.2}$$

with the use of servers. The total batch simulation time, with data servers employed, is then given by

$$t = t_c + t_s = \frac{N\mu}{c} + \frac{N}{c}\sum_i \bar{A}_i \left[L + \frac{D_i}{B}\right]. \tag{2.3}$$

Because a compute node cannot proceed with its operations before the data required for those operations is received from a server node, this expression assumes a blocking-type communication model in which the compute nodes receive the required data in discrete events. At any given time, a compute process is either receiving data from a server node, performing computations, or sitting idle. Dividing Eq. 2.3 by Eq. 2.1 gives us a ratio of the time spent simulating a batch with data servers to the time spent simulating a batch without data servers,

$$\frac{t}{t_0} = \frac{t_c + t_s}{\frac{N\mu}{p}} = \frac{p}{c}\left(1 + \frac{1}{\mu}\sum_i \bar{A}_i \left[L + \frac{D_i}{B}\right]\right). \tag{2.4}$$

With a fixed number of total processes, it is desirable to use as few server nodes as possible so that more processes can be devoted to performing computations. It is also desirable to use a number of server nodes sufficient to ensure that no compute process must wait for a server to complete a communication with some other compute node before it is able receive the data it needs. Realization of this ideal scenario requires that the total time spent communicating nuclear data exactly equals the total time spent in the simulation. That is,

$$\frac{ct_s}{s} = \frac{ct_s}{p - c} = t_c + t_s. \tag{2.5}$$

After some algebraic manipulation, we can obtain an expression for p/c which may then be substituted into Eq. 2.4. The result of this substitution can be manipulated and simplified to yield an expression for the relative simulation runtime overhead that results from employing servers for storing nuclear data off-node,

$$\frac{t - t_0}{t_0} = \frac{2}{\mu}\sum_i \bar{A}_i \left[L + \frac{D_i}{B}\right]. \tag{2.6}$$

Parameter	Value
network latency, L [s]	$2.0\text{E}-6$
network bandwidth, B [B/s]	$4.0\text{E}+9$
mean neutron simulation time, μ [s]	$1/140$

Table 2.7: Titan Supercomputer-Specific Performance Parameters

2.3.2 Performance Predictions

Using the communication model for the nuclear data server algorithm concept, we can make predictions of the algorithm's performance. These predictions are in the context of the same BOC HZP and EOC HFP full core reactor models analyzed previously. The performance of the algorithm will depend on the memory requirements and access frequencies of the nuclear data in a simulation, as well as the computing platform on which the simulation is performed.

To make the predictions we seek, numerical values are needed for each of the variables that appear in Eq. 2.6. The values for the nuclear data memory requirements and access frequencies - D_i and $\bar{A}_i$, respectively - are taken from the BEAVRS model profiling studies already described. As for the computing platform-specific parameters, we will take the Titan Cray XK7 supercomputer [16] at Oak Ridge National Laboratory as an initial reference system. Best-estimate values for each of the required parameters are given in Table 2.7. We use the same values that are used in the analysis of the tally server algorithm [57].

So, from the profiling studies, we have memory requirements and access frequencies associated with different blocks of nuclear data. Using these values and the system performance parameters in Table 2.7, we can use Eq. 2.6 to determine the runtime overhead penalty that is incurred by storing different blocks of nuclear data off-node. We can also vary the numerical values for the system parameters to examine the sensitivity of runtime overhead to these variables.

There is also the issue of deciding which blocks of nuclear data should be stored off-node. Because we want to maximize the ratio of incremental on-node memory reduction to incremental runtime overhead, we sort the blocks of data in order of

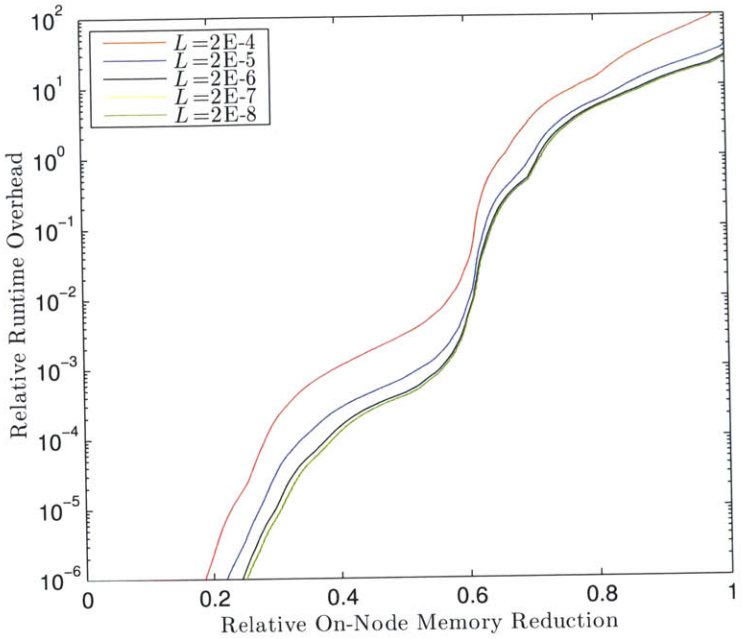

Figure 2-1: BOC HZP Overhead Predictions for Varying Network Latency

decreasing values for the expression

$$\frac{D_i / \sum_i D_i}{\frac{2\bar{A}_i}{\mu}\left[L + \frac{D_i}{B}\right]}. \tag{2.7}$$

The resulting list of nuclear data blocks gives the order in which the blocks should be chosen for off-node storage to get the most memory reduction bang for the communication overhead buck.

Using this optimal off-node storage scheme, we now look at the runtime overhead associated with storing different fractions of the entire nuclear data set off-node. Using the BOC HZP access frequencies and memory requirements, Figures 2-1, 2-2, and 2-3 show the predicted runtime overhead for varying network latency, network bandwidth, and mean neutron simulation time values, respectively. The results shown in each plot come from varying only one parameter at a time; when one parameter is varied, the others are held constant at the best-estimate values given in Table 2.7.

For the BOC HZP case, with the best-estimate system parameter values for the Titan supercomputer, the communication model predicts that more than 60% of the

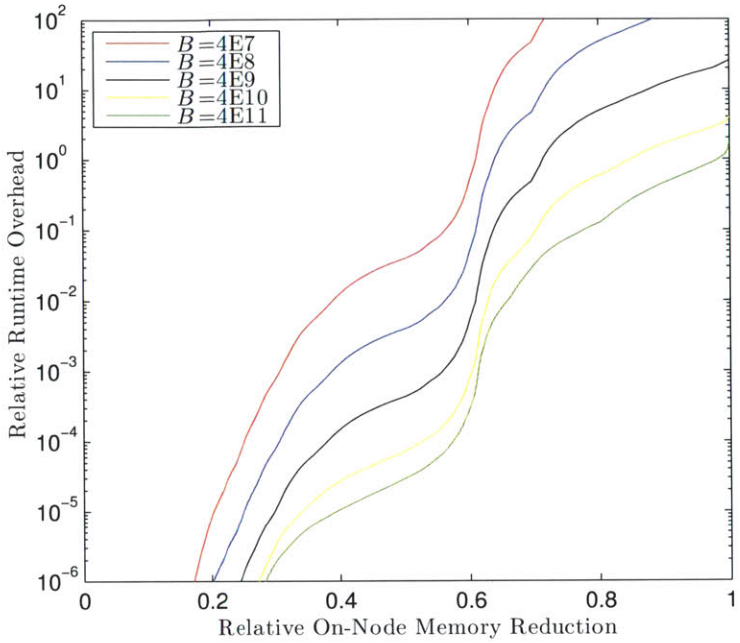

Figure 2-2: BOC HZP Overhead Predictions for Varying Network Bandwidth

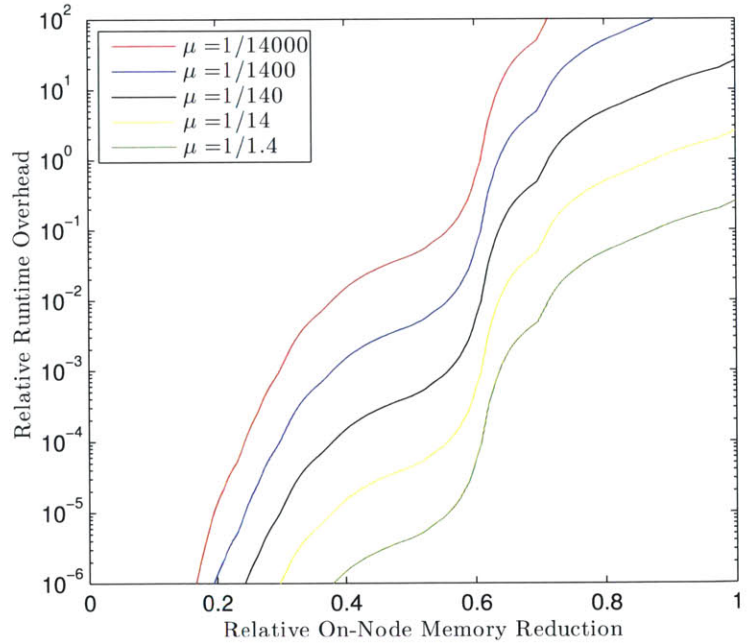

Figure 2-3: BOC HZP Overhead Predictions for Varying Simulation Rate

combined cross section and secondary distribution data can be stored, and accessed from, off-node without incurring a runtime overhead penalty in excess of 1%. However, the predicted overhead is very sensitive to the fraction of data that is stored off-node. If we store data accounting for 70% of the total memory on servers, instead of 60%, the runtime overhead jumps to nearly 80%. Favorable results are observed over a range of system parameter values. With any one parameter increased or decreased - whichever is *disadvantageous* with respect to runtime overhead - by two orders of magnitude, on-node memory requirements can be reduced by more than half with a runtime penalty of less than 10%. It is also interesting to look at the limiting case of 100% off-node data storage. It is predicted that this can be achieved at the expense of a 25-fold increase in runtime.

As is done for the BOC HZP case, overhead predictions are obtained by using the profiling results from the EOC HFP BEAVRS simulation. Figures 2-4, 2-5, and 2-6 show these predictions for varying network latency, network bandwidth, and mean neutron simulation time values, respectively. Again, using best-estimate system performance parameter values, the communication model predicts that the nuclear data accounting for more than 60% of the total cross section and secondary distribution memory requirements can be placed on server nodes without incurring an overhead penalty of more than 1%. Also, in the EOC HFP case, with any one parameter varied by two orders of magnitude in the direction that will result in a greater overhead, it is predicted that 50% of the total cross section and secondary distribution data can be stored off-node at an overhead penalty of less than 1%.[3]

2.4 Conclusions

The development of algorithms for efficiently managing nuclear data in large Monte Carlo neutron transport simulations should be informed by knowledge of the types of data that are needed in the simulations as well as how often they are needed. The first

[3]Caution should be exercised when interpreting the memory reduction and runtime overhead predictions. The trends based on computing platform parameters and increased off-node storage are more reliable than any specific numerical value.

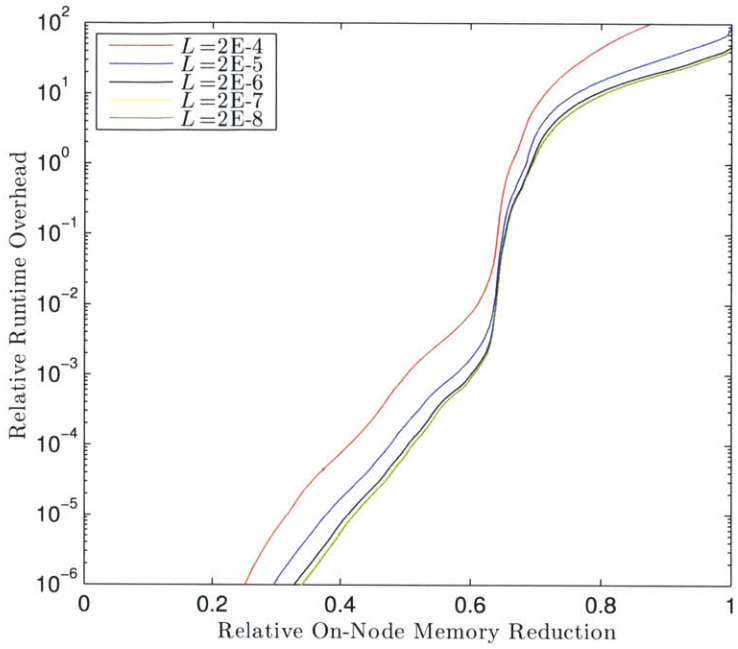

Figure 2-4: EOC HFP Overhead Predictions for Varying Network Latency

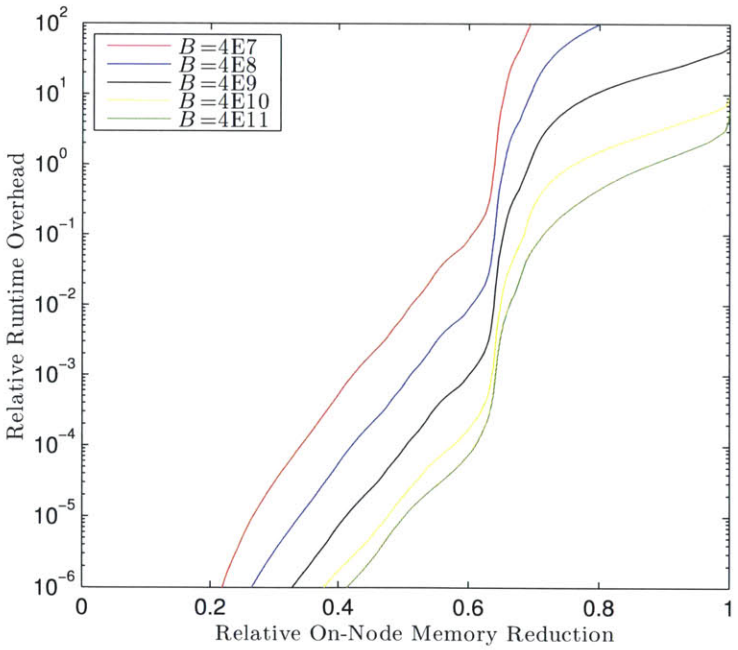

Figure 2-5: EOC HFP Overhead Predictions for Varying Network Bandwidth

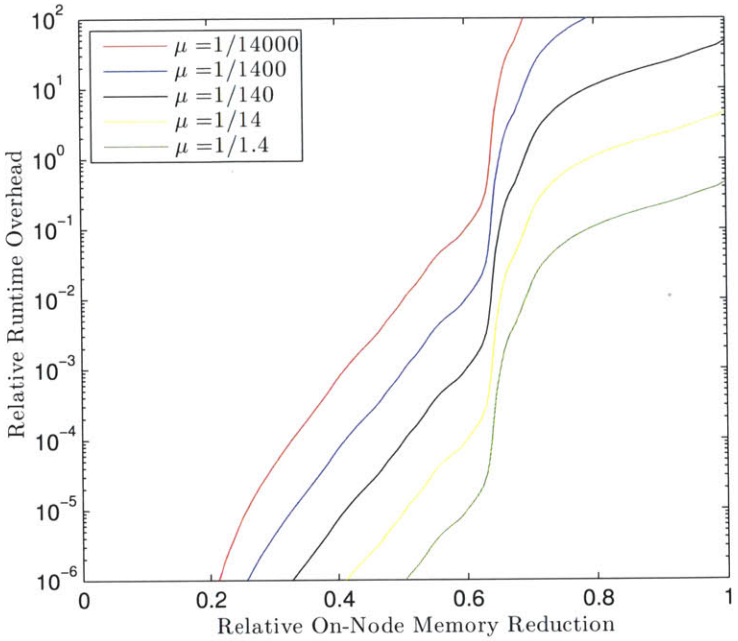

Figure 2-6: EOC HFP Overhead Predictions for Varying Simulation Rate

segment of work presented in this chapter deals with profiling studies that are carried out in order to ascertain this data access information. Armed with these results, we have proposed a nuclear data server algorithm concept in which infrequently accessed nuclear data are stored on dedicated server nodes, off of the compute nodes, in order to reduce on-node nuclear data memory requirements. A communication model for the proposed algorithm is then derived. Taking the results of the profiling studies and system performance parameters for the Titan supercomputer as input, the server communication model is used to make algorithmic performance predictions. These predictions indicate that the nuclear data server algorithm may allow appreciable on-node memory reductions, with only modest runtime overhead, in practical full core reactor physics simulations.

Chapter 3

Accelerated Resonance Elastic Scattering Kernel Sampling

In this chapter[1] we present the derivation and investigation of a new Doppler broadening rejection sampling approach for the exact treatment of free gas resonance elastic scattering in Monte Carlo neutron transport codes. Implemented in OpenMC, this method correctly accounts for the energy dependence of cross sections when treating the thermal motion of target nuclei in elastic scattering events. The method is verified against both stochastic and deterministic reference results in the literature for ^{238}U resonance scattering. Upscatter percentages and mean scattered energies calculated with the method are shown to agree well with the reference scattering kernel results. Additionally, pin cell and full core k_{eff} results calculated with this implementation of the exact resonance scattering kernel are shown to be in close agreement with those in the literature. The attractiveness of the method stems from its improvement upon a computationally expensive rejection sampling procedure employed by an earlier stochastic resonance scattering treatment. With no loss in accuracy, the accelerated resonance elastic scattering kernel sampling algorithm is shown to reduce overall runtime by 3-5% relative to the Doppler broadening rejection correction method for both pin cell and full core thermal reactor benchmark problems. This translates to a 30-40% reduction of the runtime overhead.

[1]The work presented in this chapter is published as Reference [73].

3.1 Introduction

At sufficiently high incident neutron energies, elastic scattering can be accurately modeled with zero-velocity target nuclei. In the epithermal energy range, however, the thermal motion of target nuclei can have a significant effect on differential scattering kernels. These differences can, in turn, significantly affect macroscopic values such as the effective multiplication factor of a system. Therefore, it is important to have an accurate model for the kinematics of epithermal elastic scattering. Such a model requires that target velocities be drawn from the exact bivariate distribution in both speed and direction-of-flight [55].

Typically, stochastic treatments of epithermal elastic scattering make the assumption that the distribution of target speeds takes on that of an isotropic Maxwell-Boltzmann ideal gas. Historically, the procedure for sampling the target velocity distribution has also relied on the simplifying assumption that, in any given elastic scattering event, the cross section of the target nuclide is effectively constant over the narrow range of probable relative neutron speeds [23]. This assumption leads to inadequate results for elastic scattering from heavy resonant nuclides which can exhibit rapid cross section variations over small energy intervals (i.e. resonances). Alternate stochastic treatments of resonance elastic scattering have been shown to correctly reproduce the exact theoretical scattering kernels. However, these methods give rise to appreciable decreases in computational efficiency.

It is the aim of this work to develop a resonance elastic scattering treatment that correctly reproduces exact free gas scattering kernels and that also reduces the undesirable sampling inefficiencies of previously proposed methods. The accelerated resonance elastic scattering kernel sampling method derived here is implemented in the OpenMC Monte Carlo neutron transport code [58]. The method is then verified through comparisons with reference literature results [64][71][75][42]. Various results computed for neutrons with incident energies near ^{238}U resonance energies - where resonance scattering effects are most pronounced - are presented. The results of thermal reactor pin cell and full core eigenvalue calculations are also analyzed. Sample

target velocity rejection rates and runtimes are examined in order to assess the relative computational efficiency of the accelerated sampling scheme proposed in this work.

3.2 Elastic Scattering Models

Correct treatment of the elastic scattering process is vital to the accuracy of reactor physics simulations. Multiple procedures, with varying degrees of physical fidelity, have been developed and implemented in Monte Carlo codes.

3.2.1 Asymptotic Model

The simplest model for the target velocity in an elastic scattering event is to assume that the target is at rest. Utilizing this zero-velocity target assumption is equivalent to modeling elastic scattering with the well-known asymptotic kernel for isotropic scattering in the center-of-mass system [35],

$$P(E \rightarrow E') = \begin{cases} \frac{1}{E(1-\alpha)} & \alpha E \leq E' \leq E \\ 0 & \text{otherwise} \end{cases}, \tag{3.1}$$

with E being the incident neutron energy, E' being the scattered neutron energy, and $\alpha \equiv (A-1)^2/(A+1)^2$, where A is the ratio of target mass to neutron mass. While this model is reasonable for scattering events with high incident neutron energies, it does not accurately capture the effects of target motion on thermal scattering kernels. Also, it has been shown that the asymptotic model, which does not permit upscattering into resonances, can dramatically misrepresent epithermal scattering kernels for resonant target nuclei [55]. The misrepresentation manifests itself as a reduction in resonance absorption. This reduction has been shown to introduce biases of a couple of hundred pcm in k_{eff} results for LWR configurations. Even greater errors are observed in simulations of high-temperature reactor systems [47].

3.2.2 Constant Cross Section Model

In order to eliminate the asymptotic model's assumption of a target at rest and take into account the thermal motion of target nuclei when considering the kinematics of elastic scattering, the velocity of the target must be determined. The ideal gas model is widely used for the treatment of epithermal neutron scattering in Monte Carlo codes [23]. In this model the motion of target nuclei is assumed to be isotropic, with speeds characterized by the classical Maxwell-Boltzmann distribution [19]. This distribution for target speed v_t in an ideal gas at temperature T is given by

$$\begin{aligned} M(T, v_t) &= \frac{4}{\sqrt{\pi}} \beta^3 v_t^2 e^{-\beta^2 v_t^2}; \\ \beta &\equiv \sqrt{\frac{A m_n}{2kT}}. \end{aligned} \tag{3.2}$$

Here, k_{B} is the Boltzmann constant and m_n is the mass of a neutron.

The convolution of the Maxwell-Boltzmann distribution with the product of the relative speed between the neutron and target, v_{rel}, and the 0 K elastic scattering cross section yields an expression for the effective, reaction rate-preserving, Doppler broadened scattering cross section [32],

$$\sigma_s(T, v_n) = \frac{1}{2v_n} \int \int v_{\mathrm{rel}} \sigma_s(0, v_{\mathrm{rel}}) M(T, v_t) dv_t d\mu, \tag{3.3}$$

in which v_n is the neutron speed. The relative speed is given by

$$v_{\mathrm{rel}} = |\vec{v_n} - \vec{v_t}| = \sqrt{v_n^2 + v_t^2 - 2\mu v_n v_t}, \tag{3.4}$$

with μ being the cosine of the angle between the initial neutron and target direction vectors.

The Doppler broadened cross section can be recast as a joint probability density function (PDF),

$$P(v_t, \mu | v_n) = \frac{v_{\mathrm{rel}} \sigma_s(0, v_{\mathrm{rel}}) M(T, v_t)}{2 v_n \sigma_s(T, v_n)}, \tag{3.5}$$

for the correlated μ and v_t variables. The correlation of μ and v_t means that Eq.

(3.5) cannot be sampled directly by sampling the PDF's of μ and v_t independently [59].

After assuming that the 0 K scattering cross section of the target varies negligibly over the range of practically attainable v_{rel} values, the integral over μ in Eq. (3.3) can be evaluated analytically [23], enabling use of the sampling procedure outlined below. With the assumption of a constant cross section, the PDF is described by

$$P_{\text{CXS}}(v_t, \mu | v_n) \propto v_{\text{rel}} M(T, v_t). \tag{3.6}$$

The constant cross section approximation (CXS) is central to the target velocity sampling algorithm detailed by Gelbard [41]. This algorithm, in slightly varying forms, has long been the standard method for treating epithermal elastic scattering in Monte Carlo codes such as MCNP [68], MC21 [66], and OpenMC [58]. The approximation has been justified with the reasoning that the scattering cross sections of light nuclei are typically slowly varying in energy, and that heavy nuclei, whose scattering cross sections can vary sharply in energy, contribute so little to neutron moderation through elastic scattering that the effects of the approximation are negligible [68].

The sampling of Eq. (3.6) can be simplified with the inclusion of canceling $v_n + v_t$ terms which allow the distribution to be rewritten as

$$\begin{aligned} P_{\text{CXS}}(v_t, \mu | v_n) = C_{\text{CXS}} \frac{v_{\text{rel}}}{v_n + v_t} [v_n v_t^2 e^{-\beta^2 v_t^2} + v_t^3 e^{-\beta^2 v_t^2}]; \\ C_{\text{CXS}} = \frac{2\beta^3}{v_n \sqrt{\pi}}. \end{aligned} \tag{3.7}$$

Having no dependence on target velocity, C_{CXS} is simply a normalization constant. Then, μ can be sampled uniformly and v_t can be obtained by sampling the distribution given by the bracketed terms in Eq. (3.7). The sampled target velocity specified by μ and v_t is then accepted with a probability equal to the ratio

$$R_{\text{CXS}} = \frac{v_{\text{rel}}}{v_n + v_t}. \tag{3.8}$$

3.2.3 Energy Dependent Cross Section Model

The CXS implementation of the ideal gas model addresses the inadequacy of the asymptotic model insofar as it assigns, through the procedure outlined in the previous section, non-zero velocities to target nuclei. However, in the target velocity sampling procedure, the energy dependence of cross sections is neglected. It was shown analytically by Ouisloumen and Sanchez [55] that, in the epithermal region, the strong energy dependence of resonant nuclei scattering cross sections can result in scattering kernels that are highly distorted from those given by the asymptotic model. The CXS ideal gas model cannot accurately calculate epithermal resonance scattering kernels because it neglects the scattering cross section energy dependence that is largely responsible for the distortion of resonant nuclei scattering kernels from an asymptotic shape. There exist multiple methods for correctly incorporating the effects of energy dependent scattering cross sections in epithermal scattering treatments.

Scattering Law Tables

Typically, $S(\alpha, \beta)$ scattering law tables are used to specify secondary energy and angular distributions for neutron scattering in the thermal region, where chemical binding effects may be significant. The capability to generate tables for nuclei with energy dependent cross sections was introduced into the NJOY Nuclear Data Processing System [50] by Rothenstein [60]. The use of $S(\alpha, \beta)$ tables in modeling scattering from resonant nuclei, demonstrated by Dagan [33], forgoes the sampling of a target velocity and, in doing so, avoids the problems encountered with the CXS model. However, the scattering law method requires the generation of $S(\alpha, \beta)$ tables for each nuclide with cross section energy dependence that must be considered. In general, tables must be generated on unique, fine energy grids in order to capture the resonance cross section structure of individual nuclides [17]. This results in undesirable increases in nuclear data memory requirements.

Doppler Broadening Rejection Correction

An alternate, more general stochastic method for the exact treatment of resonance scattering was suggested by Rothenstein [59]. The Doppler broadening rejection correction (DBRC) corrects for the CXS approximation with a modification of the PDF from Eq. (3.7). The energy dependent cross section term is reintroduced and canceling $\sigma_{s,\max}^{0\text{ K}}$ terms are added so that

$$P(v_t, \mu | v_n) = C_{\text{DBRC}} \frac{\sigma_s(0, v_{\text{rel}})}{\sigma_{s,\max}^{0\text{ K}}} \frac{v_{\text{rel}}}{v_n + v_t} \times (v_n v_t^2 e^{-\beta^2 v_t^2} + v_t^3 e^{-\beta^2 v_t^2}); \qquad C_{\text{DBRC}} = \frac{2\beta^3 \sigma_{s,\max}^{0\text{ K}}}{v_n \sqrt{\pi} \sigma_s(T, v_n)}. \tag{3.9}$$

The $\sigma_{s,\max}^{0\text{ K}}$ terms represent the maximum 0 K scattering cross section on the interval of practically attainable v_{rel} values. This interval is determined by v_n and the maximum value of v_t that has a non-negligible probability of being sampled from the Maxwell-Boltzmann distribution. Typically, this interval is chosen to be $[v_n - 4/\beta, v_n + 4/\beta]$ in agreement with the bounds of integration used to Doppler broaden cross sections in the SIGMA1 algorithm [32].

The addition of the $\sigma_{s,\max}^{0\text{ K}}$ terms takes into account the effects of an energy dependent cross section through an additional rejection criterion. Samples of μ and v_t accepted through the standard CXS algorithm are subjected to this additional criterion and accepted with a probability

$$P_{\text{DBRC}} = \frac{\sigma_s(0, v_{\text{rel}})}{\sigma_{s,\max}^{0\text{ K}}}. \tag{3.10}$$

The DBRC was first demonstrated by Becker, et al. [18] and has since been implemented and investigated in several Monte Carlo codes including MCNP [68], TRIPOLI [67], MC21 [66], and, in the course of this work, OpenMC [58].

Though the DBRC has been shown to correctly reproduce resonance elastic scattering kernels, there are computational costs associated with the additional rejection criterion. For incident neutron energies in close proximity to resonances, the rejection

sampling of a 0 K cross section leads to an inordinate number of discarded samples. Applied to reactor physics simulations, the DBRC has been observed to incur computational performance penalties of 10-15% [64][71].

Weight Correction Method

The weight correction method (WCM) is another stochastic procedure for exactly treating free gas resonance scattering [47][53]. In the WCM algorithm, μ and v_t values are independently sampled, just as in the CXS algorithm. However, in order to account for the sampled target velocity coming from the constant cross section PDF in Eq. (3.7), rather than the exact, energy dependent PDF in Eq. (3.5), the WCM applies a correction factor to the scattered neutron weight, w, such that the updated weight becomes

$$\begin{aligned} w_{\text{new}} &= w\frac{P(v_t,\mu|v_n)}{P_{\text{CXS}}(v_t,\mu|v_n)} = \frac{\sigma_s(0,v_{\text{rel}})}{\sigma_s(T,v_n)}F(\beta v_t); \\ F(\beta v_t) &= \frac{(\beta^2 v_t^2+\frac{1}{2})\,\text{erf}\,(\beta v_t)+\sqrt{\frac{1}{\pi}}\beta v_t \exp\left(-\beta^2 v_t^2\right)}{\beta^2 v_t^2}. \end{aligned} \tag{3.11}$$

In practice, $F(\beta v_t)$ is approximated as unity in order to avoid expensive function evaluations. The error introduced by this approximation is negligible [53].

The speed of a calculation performed with the WCM is virtually unchanged from that of a calculation performed with the CXS method because all of the same target velocities are accepted, with the only difference between the methods being the additional correction factor. The downside of the method is that the particle weights may fluctuate dramatically, leading to increased variance in the scattered energy-angle distributions. Consequently, an increase in the variance of any tallies dependent on these distributions (e.g. k_{eff}) will be observed. The WCM has been shown to be significantly less computationally efficient than the DBRC method in figure of merit studies conducted by Trumbull and Fieno [71] and is, therefore, not investigated further in this work.

3.3 Accelerated Resonance Elastic Scattering Kernel Sampling

The motivation for the proposed method is improved computational efficiency of resonance elastic scattering kernel sampling through avoidance of the inefficient rejection sampling of 0 K scattering cross sections near resonance energies. Rather than sample a target velocity relatively efficiently and then rejection sample a 0 K cross section, as is done in the DBRC algorithm, the accelerated resonance elastic scattering (ARES) kernel sampling method calls for directly sampling the 0 K cross section data distribution and then rejection sampling a different function.

3.3.1 ARES Kernel Sampling Algorithm

The method is based on the principle that any PDF of the form

$$P(\vec{x}) = Cf(\vec{x})g(\vec{x}) \tag{3.12}$$

with a normalization constant, C, and a bounded $g(\vec{x})$ can be sampled by first drawing a value from $f(\vec{x})$, and then rejection sampling $g(\vec{x})$, and vice versa [23]. In the former case, the sample value, $\vec{x}_s$, drawn from $f(\vec{x})$, is accepted with a probability equal to the ratio

$$R_{\text{acc}} = \frac{g(\vec{x}_s)}{\max(g(\vec{x}))}. \tag{3.13}$$

Eq. (3.12) can be cast in terms of the target velocity sampling problem as

$$P(v_t, \mu|v_n) = C_{\text{ARES}} f(v_t, \mu|v_n) g(v_t, \mu|v_n) \tag{3.14}$$

where

$$f(v_t, \mu|v_n) = v_{\text{rel}} \sigma_s(0, v_{\text{rel}}) U_{\text{speed}}(v_t) \tag{3.15}$$

is the function to be sampled first,

$$g(v_t, \mu|v_n) = M(T, v_t)U_{\text{angle}}(\mu) \tag{3.16}$$

is the function on which rejection sampling is applied,

$$C_{\text{ARES}} = \frac{1}{v_n \sigma_s(T, v_n)U_{\text{speed}}(v_t)} \tag{3.17}$$

is a normalization constant,

$$U_{\text{angle}}(\mu) = \begin{cases} 1/2 & -1 \leq \mu \leq 1 \\ 0 & \text{otherwise} \end{cases} \tag{3.18}$$

is the uniform (i.e. isotropic) distribution of physically meaningful μ values[2], and

$$U_{\text{speed}}(v_t) = \begin{cases} 1/v_{t,\max} & 0 \leq v_t < v_{t,\max} \\ 0 & \text{otherwise} \end{cases} \tag{3.19}$$

is a uniform distribution of target speeds over the range of values that have a non-negligible probability of occurring. The value of $v_{t,\max}$ is the maximum target speed that does not have a negligible probability of being sampled from the Maxwell-Boltzmann distribution. As in SIGMA1 and the DBRC method, $v_{t,\max}$ is taken to be $4/\beta$, which corresponds to a maximum target energy of $16k_{\text{B}}T$.

The ARES method proceeds by sampling a target velocity from $f(v_t, \mu|v_n)$. This is accomplished in two steps. First, a v_{rel} value is sampled directly from the distribution given by the $v_{\text{rel}}\sigma_s(0, v_{\text{rel}})$ term in Eq. (3.15) on the interval $[v_n - 4/\beta, v_n + 4/\beta]$. Using the tabulated 0 K cross section data in ACE format, which is linearly interpolable in energy, the direct sampling of this distribution is fairly straightforward. Because $U_{\text{speed}}(v_t)$ is a constant, it has no bearing on the sampling of a v_{rel} value. Second,

[2]This same distribution appears implicitly in the CXS and DBRC algorithms where μ is sampled isotropically. Here, we include it explicitly to highlight the rejection of unphysical μ values that is required in the ARES procedure.

$U_{\text{speed}}(v_t)$ is uniformly sampled on the interval $[0, v_{t,\max}]$ to obtain a target speed. With v_{rel} and v_t now known, μ can be directly calculated by rearranging Eq. (3.4) so that

$$\mu = \frac{v_n^2 + v_t^2 - v_{\text{rel}}^2}{2 v_n v_t}. \tag{3.20}$$

With values of μ and v_t fixed, a sample target velocity has now been completely specified.

Next, we must perform rejection sampling on $g(v_t, \mu | v_n)$. This amounts to accepting the sampled target velocity with a probability

$$P_{\text{ARES}} = \frac{M(T, v_t)}{\max(M(T, v_t))} \frac{U_{\text{angle}}(\mu)}{\max(U_{\text{angle}}(\mu))}. \tag{3.21}$$

The rejection sampling is performed in two steps. First, we check for satisfaction of the ratio

$$\xi_1 \leq \frac{U_{\text{angle}}(\mu)}{\max(U_{\text{angle}}(\mu))}, \tag{3.22}$$

with ξ_1 being a random number drawn uniformly from the unit interval. Because $U_{\text{angle}}(\mu)$ is constant on the physically meaningful range of values, $[-1, 1]$, and zero elsewhere, we may simply accept all values of μ that are in the physical range and reject all those that are not. Second, using the sampled v_t, we check that

$$\xi_2 \leq \frac{M(T, v_t)}{\max(M(T, v_t))} = \frac{M(T, v_t)}{M(T, 1/\beta)} \tag{3.23}$$

is satisfied, with $1/\beta$ being the most probable target speed, where the Maxwell-Boltzmann distribution takes on its maximum value. If the inequalities in Eqs. (3.22) and (3.23) both hold, the sampled target velocity, specified by the values of μ and v_t, is accepted. If either of the inequalities does not hold, the algorithm starts over with sampling a new v_{rel}, and proceeds until Eqs. (3.22) and (3.23) are simultaneously satisfied. At this point, a target velocity has been accepted and the two-body kinematic equations may be solved, just as in the CXS and DBRC algorithms.

3.3.2 Additional Considerations

There are many possible variations to the presented algorithm that may marginally affect computational efficiency and accuracy. To begin with, the ARES method need not be applied over the entire range in which the treatment of resonance scattering is desired. Instead, a combination of resonance scattering methods could be used, with different methods being applied to different segments of the energy range. In particular, intuition might lead one to suspect that the ARES method should really only be applied in the vicinity of resonance energies, where the performance of DBRC is worst, and where ARES avoids the costly 0 K cross section rejection sampling. In between resonances, it might be advantageous to switch back to the DBRC method because the cross section variation is much less severe, leading to improved rejection sampling efficiencies. In contrast, for ARES, a Maxwell-Boltzmann distribution is rejection sampled with a uniform bound whether the incident neutron energy is near a resonance or not. Though the cost of this sampling procedure is relatively small, it is still non-negligible and may not be justified away from resonances.

With regard to the sampling of a v_{rel} value from the distribution given in Eq. (3.15), it should be noted that the direct sampling is not, strictly speaking, exact. This is owing to the nonlinearity of the $v_{\text{rel}}\sigma_s(0, v_{\text{rel}})$ term in the energy variable. The 0 K cross section data is, by itself, linearly interpolable in energy. However, after multiplication by speed, which is proportional to the square root of energy, exactly accurate linear interpolation is no longer possible. Still, because of the small separation in energy between consecutive data points in regions of appreciable cross section energy dependence, the distribution can, with a negligible effect on results, be treated as piecewise-linear. This allows for the building of a cumulative distribution function (CDF) through simple trapezoidal integration of the data points, as well as linear interpolation between CDF points when performing the direct sampling. The integration can be performed at the initialization of a simulation - in which case the CDF points are stored on the same energy grid as the 0 K scattering cross section - or, in the interest of memory reduction, as necessary throughout a simulation and only

at energy points corresponding to the applicable domain of v_{rel}. If, for some nuclide, the approximation of the distribution as piecewise-linear is found to be inaccurate, a minor correction is required. In this case, the v_{rel} term can simply be removed from the distribution that is to be directly sampled and incorporated into the second distribution, on which rejection sampling is performed[3]. So, instead of distributions given by Eq. (3.15) and Eq. (3.16), we have

$$f(v_t, \mu|v_n) = \sigma_s(0, v_{\text{rel}}) U_{\text{speed}}(v_t) \tag{3.24}$$

and

$$g(v_t, \mu|v_n) = v_{\text{rel}} M(T, v_t) U_{\text{angle}}(\mu). \tag{3.25}$$

This leads to an additional, relatively efficient rejection criterion, given by

$$\xi_3 \leq \frac{v_{\text{rel}}}{\max(v_{\text{rel}})} = \frac{v_{\text{rel}}}{v_n + v_{t,\max}}, \tag{3.26}$$

that would have to be satisfied in order for the target velocity to be accepted. If it is ever necessary, this rejection ratio can be tested first, before the calculation of $M(T, v_t)$ in Eq. (3.23), because it relies only on the sampled v_{rel} and not any additional computations.

In implementing the ARES method in OpenMC, one minor, yet noteworthy, departure is made from the algorithm, as presented. It can be helpful to think of the algorithm as consisting of two sequential parts - direct sampling of Eq. (3.15), followed by rejection sampling of Eq. (3.16). But, in practice, slight efficiency gains may be realized by reordering certain operations. Namely, the uniform sampling of a v_t value can be performed first without affecting the rest of the algorithm. This allows for checking the rejection criterion given by Eq. (3.23) as the very next step. An early check of this criterion, which must be satisfied in any case, enables the bypassing of all other computations in the event that the sampled v_t is not accepted.

[3]Alternatively, the energy grid can be made finer until any error incurred as a result of the approximation of piece-wise linearity is reduced to an acceptable level.

3.4 Verification, Results, and Analysis

The presented results serve two purposes. The first is to verify that the ARES method does, in fact, correctly reproduce exact free gas scattering kernels, as the DBRC method has been shown to do [64][75][71]. The second is to demonstrate that using the ARES method in reactor physics simulations results in improved computational performance relative to the DBRC method. In a tangential effort, the newly implemented OpenMC DBRC capability is also verified so that it can be used with confidence in additional verification studies of ARES for which no reference literature results exist.

As part of the verification of the proposed method, we compare upscatter percentages and mean scattered energies calculated with ARES to DBRC and deterministic reference results from the literature. Asymptotic, CXS, DBRC, and ARES scattering kernels are also computed and compared. In order to assess the computational performance of the new method, we calculate sample target velocity rejection rates at various energies with both the DBRC and ARES methods. Additionally, thermal reactor pin cell and full core benchmark simulations are performed with both methods to quantify efficiency gains that are realized by using ARES instead of DBRC in calculations of practical interest. The benchmark calculations also serve to further verify ARES against the DBRC method.

All calculations in this work are performed with ENDF/B-VII.0 nuclear data [25]. In eigenvalue calculations, the specified target velocity sampling method is applied only to ^{238}U over the range $5.0 - 210$ eV. The CXS model is applied below this range and the asymptotic model is applied above. The lower energy bound is chosen so that the range includes the lowest-lying ^{238}U s-wave resonance at 6.67 eV. Below the selected bound, the error in the scattering kernel that results from invoking the CXS approximation is negligible. The upper bound of the range is chosen to agree with previous benchmark simulations performed with the DBRC method [64]. All other nuclides are treated with the CXS model up to a threshold energy of $400k_{\mathrm{B}}T$, at which point the asymptotic model is applied.

	Reference [42]	MCNP	MC21	OpenMC		
Energy [eV]	Deterministic	DBRC	DBRC	DBRC	ARES	CXS
6.52	84.45	83.64 (0.19)	84.31 (0.04)	84.29 (0.04)	84.35 (0.04)	30.33 (0.05)
7.20	28.20	28.03 (0.05)	28.13 (0.04)	28.17 (0.05)	28.36 (0.05)	29.37 (0.05)
36.25	55.41	53.69 (0.01)	55.34 (0.05)	55.26 (0.05)	55.82 (0.05)	12.72 (0.03)
37.20	7.27	7.72 (0.02)	7.18 (0.03)	7.19 (0.03)	7.21 (0.03)	12.47 (0.03)

Table 3.1: Code-to-Code Comparison of Computed Upscatter Percentages (1σ)

3.4.1 Upscatter Percentages

The first verification of ARES is a comparison of ^{238}U upscatter percentages computed at 1000 K for incident neutron energies just above and just below the 6.67 eV and 36.67 eV resonances. The percentages computed with ARES are compared with the deterministic results of Ghrayeb, et al. [42], which, it should be noted, are obtained under the assumption of isotropic center-of-mass scattering, in Table 3.1. DBRC results computed with MCNP [64], MC21 [71], and OpenMC are also presented, along with CXS results from OpenMC. The upscatter percentages computed with ARES in OpenMC agree very well with both the deterministic and DBRC reference results, especially when compared to the values obtained with the CXS method.

For further verification, upscatter percentages are computed with ARES at different temperatures for incident neutron energies just above and just below the first three ^{238}U s-wave resonances. In Table 3.2, these values are compared to deterministic literature results and DBRC results generated with OpenMC. Again, very good agreement is observed between the deterministic, DBRC, and ARES results.

Though not all of the ARES upscatter percentages lie within two standard deviations of the reference results, examination of the tabulated values reveals that the various DBRC results often lie several standard deviations away from the deterministic values, and each other. The differences between the results obtained from the deterministic, DBRC, and ARES scattering kernel implementations are negligible compared to the systematic errors observed in the results computed with the CXS algorithm. In light of the relatively minor and unbiased differences between the deterministic, DBRC, and ARES results, we proceed to the next step in the verification

		Reference [42]	OpenMC	
Energy [eV]	Temperature [K]	Deterministic	DBRC	ARES
6.52	300	62.17	61.26 (0.05)	61.59 (0.05)
	600	82.84	82.82 (0.04)	82.88 (0.04)
	1000	84.45	84.29 (0.04)	84.35 (0.04)
7.20	300	16.58	16.23 (0.04)	16.63 (0.04)
	600	23.59	23.47 (0.04)	23.83 (0.04)
	1000	28.20	28.17 (0.05)	28.36 (0.05)
20.20	300	5.57	5.50 (0.02)	5.60 (0.02)
	600	15.36	15.48 (0.04)	15.67 (0.04)
	1000	30.41	30.65 (0.05)	31.15 (0.05)
21.50	300	6.61	6.44 (0.02)	6.59 (0.02)
	600	11.46	11.42 (0.03)	11.63 (0.03)
	1000	15.51	15.39 (0.04)	15.68 (0.04)
36.25	300	7.18	6.87 (0.03)	7.03 (0.03)
	600	30.54	30.66 (0.05)	31.20 (0.05)
	1000	55.41	55.26 (0.05)	55.82 (0.05)
37.20	300	3.62	3.51 (0.02)	3.61 (0.02)
	600	6.12	6.04 (0.02)	6.17 (0.02)
	1000	7.27	7.19 (0.03)	7.21 (0.03)

Table 3.2: Upscatter Percentages (1σ) Computed with DBRC and ARES

of ARES.

3.4.2 Mean Scattered Energies

As in the analysis of upscatter percentages, mean scattered energies are computed with ARES at different temperatures for incident neutron energies just above and just below the first three ^{238}U s-wave resonances. Again, the computed values are compared to deterministic literature results and DBRC results generated with OpenMC. In Table 3.3, very good agreement is seen between scattered energies calculated with ARES and scattered energies calculated either deterministically or with the DBRC method.

3.4.3 Scattering Kernels

As a third step in the verification of ARES, scattering kernels at two energies near the 36.67 eV ^{238}U resonance are computed and plotted, along with the kernels computed

		Reference [42]	OpenMC	
Energy [eV]	Temperature [K]	Deterministic	DBRC	ARES
6.52	300	6.53	6.53 (0.0001)	6.53 (0.0001)
	600	6.59	6.59 (0.0001)	6.59 (0.0001)
	1000	6.61	6.61 (0.0001)	6.61 (0.0001)
7.20	300	7.14	7.14 (0.0001)	7.14 (0.0001)
	600	7.14	7.14 (0.0001)	7.14 (0.0001)
	1000	7.14	7.14 (0.0001)	7.14 (0.0001)
20.20	300	19.98	19.98 (0.0002)	19.98 (0.0002)
	600	19.99	19.99 (0.0002)	19.99 (0.0002)
	1000	20.04	20.04 (0.0003)	20.04 (0.0003)
21.50	300	21.31	21.31 (0.0001)	21.31 (0.0001)
	600	21.31	21.30 (0.0002)	21.30 (0.0002)
	1000	21.29	21.29 (0.0002)	21.29 (0.0002)
36.25	300	36.01	36.01 (0.0002)	36.01 (0.0002)
	600	36.13	36.13 (0.0002)	36.13 (0.0002)
	1000	36.26	36.26 (0.0002)	36.26 (0.0002)
37.20	300	36.87	36.86 (0.0002)	36.86 (0.0002)
	600	36.83	36.82 (0.0003)	36.82 (0.0003)
	1000	36.73	36.72 (0.0004)	36.72 (0.0004)

Table 3.3: Mean Scattered Energies (1σ) [eV] Computed with DBRC and ARES

with the asymptotic model, and both the CXS and DBRC methods. Figures 3-1 and 3-2 illustrate scattering kernels computed for an incident neutron energy of 36.25 eV at 300 K and 1000 K, respectively. The kernels depicted in Figures 3-3 and 3-4 are computed for an incident energy of 37.2 eV at 300 K and 1000 K, respectively.

The expected deviations of the CXS, DBRC, and ARES kernels from the asymptotic kernel are clear in each plot. The discrepancy between the CXS kernel and the exact DBRC and ARES kernels is also readily apparent. There is very good visual agreement between the DBRC and ARES kernels in each of the four cases presented. As anticipated, the differences between the exact scattering kernels and the kernels predicted by the asymptotic model are amplified by increases in temperature. Finally, higher probabilities of upscatter are observed for incident energies just below the resonance than are observed for incident energies just above the resonance. All of these general trends are to be expected and have been demonstrated previously [55].

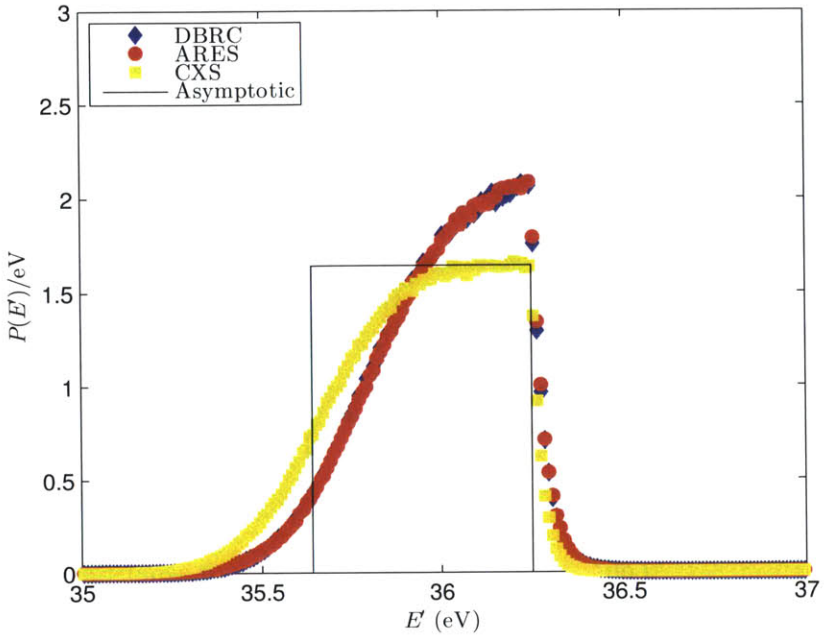

Figure 3-1: 36.25 eV Scattering Kernel at 300 K

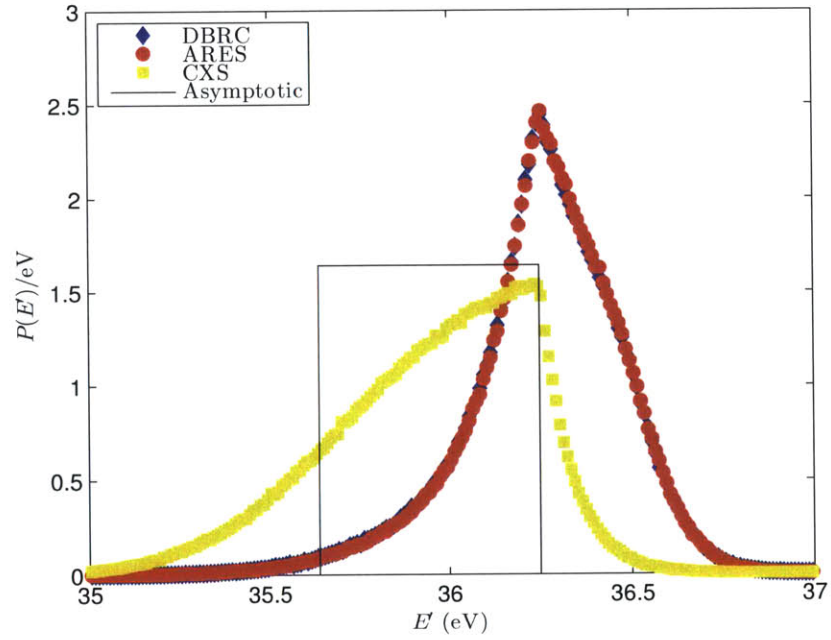

Figure 3-2: 36.25 eV Scattering Kernel at 1000 K

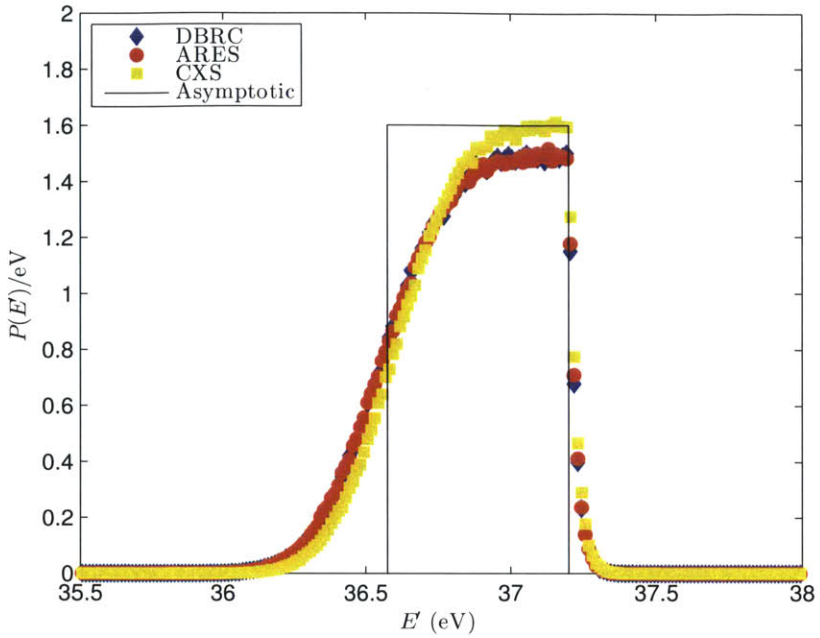

Figure 3-3: 37.2 eV Scattering Kernel at 300 K

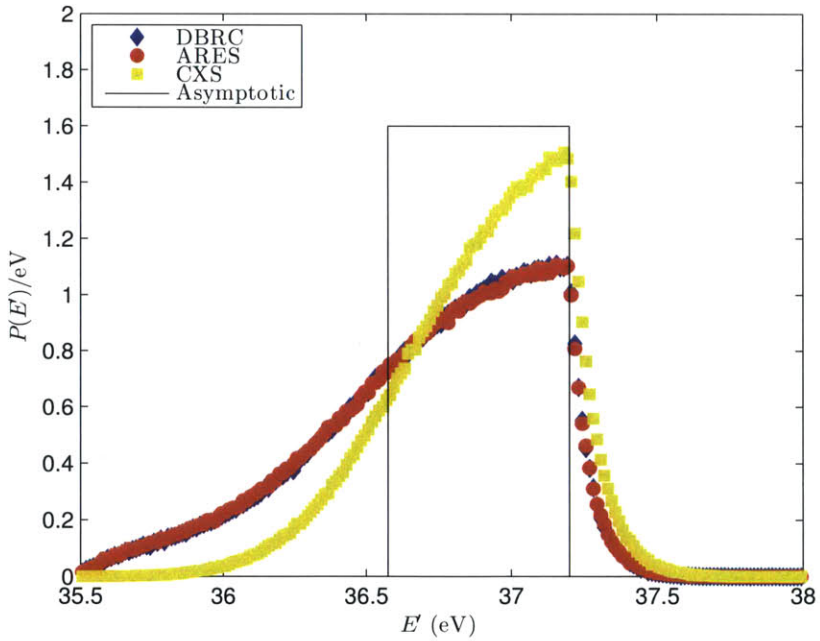

Figure 3-4: 37.2 eV Scattering Kernel at 1000 K

3.4.4 Rejection Rates

After a successful verification of ARES against both the deterministic and DBRC reference results, demonstrated in the preceding comparisons of upscatter percentages, mean scattered energies, and scattering kernel distributions, we transition to an investigation of the computational efficiency of the method. With the principle aim of ARES being to improve upon the rejection sampling scheme employed by the DBRC method, the results presented in this section relate only to the ARES and DBRC methods.

Because the efficiency of a resonance scattering method depends directly on the number of rejected target velocity samples, the number of rejections per sampled target velocity acceptance is a quantity of particular interest. Moreover, both integral and differential rejection rates can be considered, with the former giving a picture of the overall efficiency of a method and the latter giving information about sampling efficiencies at specific energies. A comparison between the methods' differential rejection rates is especially enlightening in that it provides some insight as to whether one method performs better in a certain energy range relative to the other method. Analysis of the methods' relative efficiencies in different energy intervals is necessary if a successful hybrid scheme, which relies on different methods for treating different energies, is to be constructed.

Examination of differential rejection rates also reveals the incident energies that result in the greatest share of total rejections. In any effort to improve overall computational efficiency, attention must be paid primarily to these energies. With this in mind, we look to Figure 3-5, which shows the distribution over energy of the number of sample target velocity rejections per incident neutron for a thermal pin cell benchmark problem that is described in the next section. Resonance scattering is treated in the $5.0 - 210$ eV energy range. For ease of interpretation, Figure 3-5 displays only the portion of this range that spans the 20.87 eV resonance, as well as energies a few eV above and below.

The dependence on energy of the number of observed rejections shown in Figure

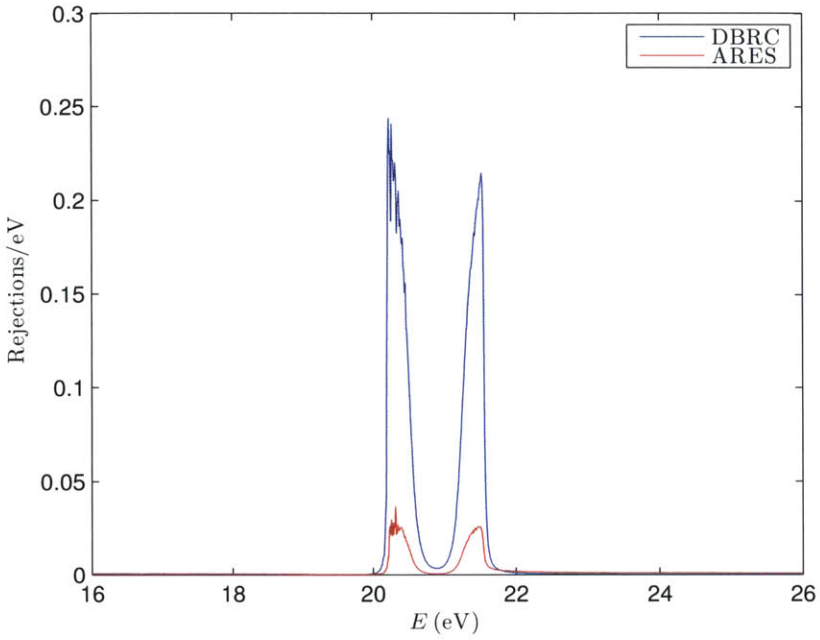

Figure 3-5: Distribution of Rejections in Energy (5.0 wt.% Enr. HFP Pin Cell)

3-5 is representative of behavior seen over the entire energy range in which resonance scattering methods are applied. That is, the majority of rejections occur at energies just above and just below resonances. Away from resonance energies, the number of rejections is relatively minimal. It can also be seen that, in the vicinity of resonances, where most of the rejections occur, the ARES method requires significantly fewer rejections than does the DBRC method. This is demonstrated in Table 3.4, which shows the average number of rejections per acceptance at energies near resonances for both the DBRC and ARES methods. These particular energies are selected to be near where the peak rejection rates occur, just above and below the first three ^{238}U s-wave resonances. In contrast, farther away from resonance energies, the DBRC method results in fewer rejections relative to the ARES method. This is highlighted by the values in Table 3.5, which shows the average number of rejections per acceptance at energies far from resonances. Taken together, these observations about the relative efficiencies of the two methods lead to the concept of a hybrid method. Such a method can make use of the ARES method near resonances and the DBRC method away from resonances. However, because such a vast number of rejections occur near

Energy [eV]	DBRC	ARES
6.35	589.6 (1.9)	384.5 (1.2)
7.00	71.1 (0.2)	53.3 (0.2)
20.25	24190.2 (80.6)	11053.1 (37.0)
21.55	360.8 (1.1)	160.1 (0.5)
35.75	5583.6 (18.6)	1440.0 (4.6)
37.55	332.4 (1.1)	195.1 (0.6)

Table 3.4: Rejections per Accepted Target Velocity (1σ) Near Resonances (1000 K)

Energy [eV]	DBRC	ARES
5.00	1.05 (0.0002)	13.78 (0.0132)
14.00	1.03 (0.0002)	13.82 (0.0133)
29.00	1.06 (0.0003)	13.81 (0.0133)
50.00	1.03 (0.0002)	13.82 (0.0133)

Table 3.5: Rejections per Accepted Target Velocity (1σ) Far from Resonances (1000 K)

resonances, where ARES is much more efficient than DBRC, it is likely that this type of hybrid method will lead to no appreciable efficiency gains compared to an ARES-only approach.

3.4.5 Pin Cell Benchmark

A set of thermal pin cell benchmark problems was proposed by Mosteller [54] to assess Doppler reactivity defect calculations. The benchmark specifications describe infinite pin cell lattices with low-enriched uranium (LEU) fuel as well as reactor-recycle and weapons-grade mixed-oxide (MOX) fuels. For each type of fuel, multiple compositions are given. Because the benchmark was developed to assess the Doppler reactivity defect, pin cell specifications are given at two different fuel temperatures. At the hot zero-power (HZP) condition, all materials in the problem are at 600 K. At the hot full-power condition (HFP), all materials are again at 600 K, except for the fuel, which is at 900 K. Sunny, et al. [64] used the LEU benchmark problems to investigate the effects of the correct treatment of free gas resonance scattering - via the DBRC method - on Doppler reactivity defect calculations with MCNP. Zoia, et al. [75] and

		MCNP	OpenMC	
	Enrichment (wt.%)	DBRC	DBRC	ARES
HZP $T_{\text{fuel}} = 600$ K	0.711	-0.00006 (0.00025)	-0.00044 (0.00006)	-0.00031 (0.00007)
	1.6	-0.00071 (0.00035)	-0.00056 (0.00010)	-0.00077 (0.00010)
	2.4	-0.00038 (0.00038)	-0.00068 (0.00012)	-0.00062 (0.00012)
	3.1	-0.00073 (0.00040)	-0.00073 (0.00013)	-0.00083 (0.00013)
	3.9	-0.00087 (0.00040)	-0.00094 (0.00014)	-0.00079 (0.00014)
	4.5	-0.00048 (0.00041)	-0.00078 (0.00015)	-0.00086 (0.00015)
	5.0	-0.00115 (0.00042)	-0.00072 (0.00015)	-0.00092 (0.00015)
HFP $T_{\text{fuel}} = 900$ K	0.711	-0.00057 (0.00027)	-0.00111 (0.00008)	-0.00109 (0.00008)
	1.6	-0.00182 (0.00035)	-0.00151 (0.00009)	-0.00147 (0.00009)
	2.4	-0.00164 (0.00037)	-0.00186 (0.00012)	-0.00166 (0.00011)
	3.1	-0.00155 (0.00038)	-0.00161 (0.00012)	-0.00171 (0.00012)
	3.9	-0.00140 (0.00041)	-0.00184 (0.00013)	-0.00187 (0.00014)
	4.5	-0.00194 (0.00040)	-0.00183 (0.00015)	-0.00177 (0.00014)
	5.0	-0.00154 (0.00040)	-0.00185 (0.00015)	-0.00156 (0.00015)

Table 3.6: LEU Pin Cell k_{eff} Differences (1σ) Relative to CXS Applied Below $400k_{\text{B}}T$

Trumbull and Fieno [71] extended these studies to the MOX fuel types with TRIPOLI and MC21, respectively. Here, the effects of different resonance scattering treatments on the eigenvalues and simulation runtimes of the LEU benchmark problems are investigated. Differences in k_{eff} that result from using the OpenMC DBRC epithermal scattering model relative to CXS and the OpenMC ARES epithermal scattering model relative to CXS, all for ^{238}U, are shown in Table 3.6. The differences between the results produced by the MCNP DBRC and CXS implementations are also given. In the reference case, relative to which the displayed differences are computed, the CXS model is applied up to an energy of $400k_{\text{B}}T$. In addition, Table 3.7 compares the differences in k_{eff} calculated with DBRC and ARES in OpenMC with those calculated by the TRIPOLI implementation of the DBRC method [75]. In the reference case used for the computation of these differences, the CXS model is applied up to an energy of 210 eV in both codes. At both HZP and HFP conditions, agreement between the results produced with the MCNP and OpenMC implementations of DBRC and the results produced with ARES is very good, as evidenced by nearly all calculated differences lying within one or two standard deviations of each other. Similar agreement is observed for the alternate reference case in which TRIPOLI and OpenMC DRBC results and ARES results are compared. As anticipated, increased temperature exacerbates the effects of incorrectly modeling epithermal resonance scattering

		TRIPOLI	OpenMC	
	Enrichment (wt.%)	DBRC	DBRC	ARES
HZP $T_{\text{fuel}} = 600$ K	0.711	-0.00066 (0.00013)	-0.00070 (0.00006)	-0.00057 (0.00007)
	1.6	-0.00113 (0.00014)	-0.00081 (0.00011)	-0.00102 (0.00011)
	2.4	-0.00110 (0.00014)	-0.00115 (0.00012)	-0.00109 (0.00012)
	3.1	-0.00115 (0.00016)	-0.00103 (0.00014)	-0.00113 (0.00014)
	3.9	-0.00102 (0.00016)	-0.00140 (0.00013)	-0.00125 (0.00013)
	4.5	-0.00099 (0.00017)	-0.00122 (0.00014)	-0.00130 (0.00013)
	5.0	-0.00115 (0.00017)	-0.00111 (0.00014)	-0.00131 (0.00014)
HFP $T_{\text{fuel}} = 900$ K	0.711	-0.00123 (0.00013)	-0.00140 (0.00007)	-0.00138 (0.00007)
	1.6	-0.00163 (0.00014)	-0.00189 (0.00009)	-0.00185 (0.00010)
	2.4	-0.00225 (0.00014)	-0.00207 (0.00012)	-0.00187 (0.00012)
	3.1	-0.00203 (0.00016)	-0.00213 (0.00013)	-0.00223 (0.00013)
	3.9	-0.00219 (0.00016)	-0.00231 (0.00013)	-0.00234 (0.00014)
	4.5	-0.00222 (0.00016)	-0.00230 (0.00014)	-0.00224 (0.00013)
	5.0	-0.00224 (0.00017)	-0.00266 (0.00014)	-0.00237 (0.00014)

Table 3.7: LEU Pin Cell k_{eff} Differences (1σ) Relative to CXS Applied Below 210 eV

with the CXS model leading to greater differences between CXS and either DBRC or ARES eigenvalues.

As for a comparison of the computational efficiencies of the methods, Table 3.8 contains the runtime overhead percentages[4] that are observed when the OpenMC implementations of the CXS, DBRC, and ARES methods are applied over the 5.0 – 210 eV energy range. For each combination of enrichment, temperature, and resonance scattering method, the percentage value is calculated using the median runtime out of eight independent serial simulations. General trends in the results do not change depending on whether the mean or median runtime values are used. With the ARES method, we see runtime overhead of 7-8%. Depending on enrichment and temperature, this amounts to a 30-40% reduction in the runtime overhead compared to the DBRC method. The efficiency gains realized by applying ARES instead of DBRC in the epithermal region are greater at HFP conditions than at HZP conditions. Also, for a given temperature, the efficiency of ARES relative to DBRC improves slightly with increasing enrichment. Only a small increase in runtime - approximately 1% - is observed when switching from the CXS method to ARES.

[4]Percentages are calculated relative to a standard reference case in which the CXS approximation is applied below $400k_{\text{B}}T$. At HZP and HFP conditions, $400k_{\text{B}}T$ corresponds to 20.68 eV and 31.02 eV, respectively. The asymptotic model is applied above the $400k_{\text{B}}T$ cutoff.

	Enrichment (wt.%)	CXS	DBRC	ARES	Difference (DBRC-ARES)
HZP $T_{\text{fuel}} = 600$ K	0.711	6.4	10.6	7.4	3.2
	1.6	6.3	10.7	7.6	3.1
	2.4	6.8	10.9	7.7	3.2
	3.1	6.9	11.2	7.9	3.3
	3.9	6.7	11.5	7.8	3.6
	4.5	7.1	11.6	8.0	3.6
	5.0	6.9	11.6	7.9	3.7
HFP $T_{\text{fuel}} = 900$ K	0.711	6.6	12.3	8.0	4.3
	1.6	6.8	12.6	8.0	4.6
	2.4	6.9	12.9	8.0	4.9
	3.1	7.0	12.8	7.9	4.8
	3.9	7.4	13.8	8.3	5.5
	4.5	7.3	13.1	8.6	4.6
	5.0	7.1	13.6	8.3	5.2

Table 3.8: LEU Pin Cell Runtime Overhead [%]

Extending the energy range over which the CXS method is applied from $400k_{\text{B}}T$ up to 210 eV results in runtime overheads of 6-7%. This, taken along with the observation that the inaccuracies in computed eigenvalues that occur as a result of the application of the CXS approximation actually increase as the energy range is extended, supports a determination that there is no clear, practical advantage to using the CXS method in an attempt to account for resonance scattering effects induced by the thermal motion of target nuclei. Indeed, there are clear, practical disadvantages of increased runtime and greater inaccuracy in calculated eigenvalues. Because the CXS method explicitly neglects the root cause of resonance scattering effects (i.e. the dependence of cross sections on energy), the ineffectiveness of applying the approximation to resonant scatterers over a broader energy range is not unexpected.

3.4.6 Full Core Benchmark

In the interest of assessing the performance of the ARES method in full core thermal reactor simulations, eigenvalue calculations are carried out with the three-dimensional BEAVRS model [44]. Differences in k_{eff} and runtime overhead at both HZP and HFP conditions are displayed in Tables 3.9 and 3.10, respectively. As in the analysis of pin cell runtimes, overhead percentage values are calculated using the median runtime out of eight independent serial simulations. All results are calculated relative to the

	CXS	DBRC	ARES
HZP	0.00025 (0.00005)	-0.00050 (0.00005)	-0.00048 (0.00006)
HFP	0.00029 (0.00005)	-0.00137 (0.00005)	-0.00140 (0.00005)

Table 3.9: BEAVRS k_{eff} Differences (1σ) Relative to CXS Applied Below 400 $k_{\text{B}}T$

same reference case in which the CXS approximation is applied below $400k_{\text{B}}T$ and the asymptotic model is applied above the $400k_{\text{B}}T$ cutoff for all nuclides. The results found in Tables 3.9 and 3.10 are computed with the specified resonance scattering model applied to ^{238}U in the $5.0-210$ eV interval. All other nuclides are treated with the CXS below $400k_{\text{B}}T$ and the asymptotic model above.

The k_{eff} values calculated with DBRC and ARES are in very good agreement. The gains in runtime efficiency realized when using ARES, rather than DBRC, in the full core problem are somewhat lower than in the pin cell calculations presented earlier. This is likely due to the dependence of runtime on the performance of the code as a whole and not simply on the resonance scattering treatment. The full core benchmark requires tracking of particles across many more cells and materials than are present in the pin cell benchmarks. This increases the relative fraction of runtime spent performing operations unrelated to resonance scattering and, in doing so, reduces the runtime overhead attributable to resonance scattering treatments. Even so, at HFP conditions for the three-dimensional, full core BEAVRS model, a 2.7% reduction in absolute runtime is achieved by utilizing the proposed ARES method instead of the DBRC. This is a 31% reduction in runtime overhead. Both the pin cell and full core investigations of the ARES method focus only on resonance scattering in ^{238}U. When several additional resonant nuclides are included in the resonance scattering treatment, as in depletion calculations, the efficiency gains of ARES may be somewhat amplified.

	CXS	DBRC	ARES	Difference (DBRC-ARES)
HZP	4.0	4.6	4.0	0.6
HFP	4.9	8.7	6.0	2.7

Table 3.10: BEAVRS Runtime Overhead [%]

3.5 Conclusions

A new method for the exact treatment of epithermal free gas resonance scattering in Monte Carlo neutron transport codes is presented. The proposed ARES method is verified against reference upscatter probability and mean scattered energy results in the literature. Differences in eigenvalues that result from applying the exact DBRC and ARES resonance scattering methods, instead of the CXS approximation, are shown to be in excellent agreement with reference literature results for the Mosteller LEU pin cell benchmark problems. Differences in the eigenvalues for the three-dimensional, full core BEAVRS model computed with the OpenMC implementations of DBRC and ARES are also in excellent agreement with each other. Comparisons between DBRC and ARES rejection rates show that ARES requires many fewer rejections near resonances than does DBRC. Further, the large reduction in the number of rejections near resonances that is observed with the ARES method is shown to reduce overall runtimes by 3-5% relative to the DBRC method for problems of practical interest. In both pin cell and full core thermal reactor benchmark simulations, this corresponds to a 30-40% reduction in runtime overhead.

Chapter 4

Optimizations of the Energy Grid Search Algorithm

In this work we propose, implement, and test various extensions of optimizations of the typical energy grid-cross section pair lookup algorithm in continuous-energy Monte Carlo neutron transport codes. The key feature common to all of the optimizations is a reduction in the length of the vector of energies that must be searched over when locating the index of a particle's current energy. Ceteris paribus, a reduction in energy vector length yields a reduction in CPU time. The methods we present here are physics-informed. That is, they are designed to utilize the embedded physical information in a simulation in order to reduce the length of the vector to be searched. More specifically, the optimizations take advantage of information about scattering kinematics, neutron cross section structure and data representation, and also the expected characteristics of a system's spatial flux distribution and energy spectrum. The techniques that we present are implemented in the OpenMC Monte Carlo neutron transport code as part of this work. The gains in computational efficiency - as measured by overall code speedup - associated with each of the optimizations are demonstrated in OpenMC simulations of realistic systems. Depending on the system, simulation parameters, and optimization method employed, overall code speedup factors of $1.2 - 1.6$ are routinely observed.

4.1 Introduction

In the theories of neutron scattering and neutron reactions, the probability of a neutron interacting with a nuclide in a particular way is characterized by a cross section [37]. The values of these cross sections, which depend strongly on the energy of the incident neutron, are different for every nuclide and for every type of interaction. Monte Carlo codes utilize cross sections when simulating the behavior of neutrons in a system. The neutron cross section data utilized in modern Monte Carlo codes often come in ACE (A Compact ENDF[1]) Format data libraries that are generated with the NJOY nuclear data processing code system [50]. These pre-processed libraries contain data in so-called point-wise, continuous-energy form which consists of energy-cross section pairs that are piece-wise linearly interpolable in both energy and cross section. So, for each nuclide in a simulation, there is a one-dimensional grid of energies, and for each type of interaction associated with that nuclide, there is a corresponding one-dimensional grid of cross section values.

In a Monte Carlo neutron transport simulation, due to the nuclide dependence of cross section values, every time that a neutron enters a material with a composition of nuclides different from the composition of the previous material, a new set of cross section values is needed to describe interaction probabilities in the new material region. Similarly, due to the neutron energy dependence of cross section values, each time that the laboratory system energy of a neutron changes - as in an interaction with a nuclide - a new set of cross section values is needed to describe interaction probabilities for the new neutron energy. As a result, every time that a neutron moves into a new material or changes energy, total interaction cross sections must be computed for the material at the current neutron energy so that the tracklength between the neutron's current location and next collision site can be sampled. Additionally, various cross section values are needed at collision sites and cell boundaries for the sampling of reaction types and the accumulation of tallies [23].

[1]Evaluated Nuclear Data File (ENDF) may refer to a file format, or an actual file of evaluated nuclear data. ENDF-6 [69] is the internationally adopted format for the storage of the *raw* evaluated nuclear data. ENDF may also refer to the libraries of evaluated nuclear data files distributed by the USA (e.g. ENDF/B-VII.1 [24]).

There are different methods by which Monte Carlo codes handle neutron energy grid and cross section data. The methods have evolved to balance the competition between memory and speed that is frequently encountered when dealing with the management of data in Monte Carlo simulations. While a detailed accounting of all the cross section treatment methods currently in use is beyond the aim of this introduction, considering a few common procedures will be useful.

First, there is the option of separately storing each individual nuclide's energy grid and cross sections. When it comes time in a simulation that a cross section value for some nuclide is needed, that value is computed by linear-linear interpolation between the cross section values corresponding to the energies on the energy grid which bound the neutron's current energy so that the microscopic cross section σ at neutron energy E for some interaction x is given by

$$\sigma_x(E) = \sigma_x(E_i) + \frac{E - E_i}{E_{i+1} - E_i}(\sigma_x(E_{i+1}) - \sigma_x(E_i)) \tag{4.1}$$

where E_i and E_{i+1} are the nearest lower and upper bounding energy grid points, respectively, for the exact neutron energy. In order to carry out this interpolation, a search of the nuclide's energy grid must be performed so that the index of one of the bounding energy points can be determined. A binary search of the sorted energy grid is employed for the task. This algorithm achieves $O(\log(N_{\text{grid}}))$ scaling whereas a linear search displays scaling of $O(N_{\text{grid}})$ where N_{grid} is the length of the grid to be searched [29]. Historically, this treatment of energy grid-cross section data has been widely used. It is the default procedure in the MCNP6 Monte Carlo particle transport code [43]. Its main advantage, which still resonates for obvious reasons, is that it preserves, with complete accuracy, all of the physical information contained in an ACE cross section file with the minimum allowable memory requirement. It's chief shortcoming is that energy grid lookups and cross section computations must be performed on-the-fly, whenever they are needed, for each nuclide required by the simulation.

The unionized energy grid treatment [48] is one method which has been shown

to significantly cut down simulation runtime relative to the individual nuclide energy grid method just described. This algorithm, originally implemented in the Serpent code [49], calls for the construction of a unionized grid containing all of the energy points from the individual grid of every nuclide in the problem. Then, the individual reaction cross sections for each nuclide are stored on this same grid. A single energy grid for the entire problem can reduce the time spent in performing binary searches to locate the appropriate energy grid index. In fact, a single binary search determines the energy and cross section indices for each nuclide in the problem because they all utilize the same grid structure. This means that a single interpolation factor (the energy term in Eq. 4.1) can be used for all nuclides, further speeding any cross section computations. In addition, the unionized grid provides a natural method for storing material cross sections that are frequently accessed within a simulation such as those for the total, elastic scattering, absorption, and fission interactions.

While this method has been shown to yield impressive speedups over treatments with unique energy grids for each nuclide [48], the drawback of the unionized grid is that it can dramatically increase the memory burden of cross section data storage. Because each nuclide's cross section grids contain points corresponding to the energy grid points from every other nuclide in the problem, there is storage of many unneeded data points. This is an important consideration, especially with respect to current and proposed high-performance computing (HPC) architectures. The on-node memory of these systems can become limiting for large, practical Monte Carlo simulations [51]. For example, we can consider a full core depletion problem from the reactor physics field. In a simulation of this type, it may be required that the inventories of hundreds of nuclides be tracked. Each of these nuclides may have up to hundreds of thousands of energy and cross section grid points; the unionized energy grid will have millions. For each of these grid points, the cross section values for multiple reactions must be stored. If we allow for the existence of the same nuclide at multiple temperatures, as is encountered in physical reality, we may need to store cross section data on a fine temperature grid. Intervals of $5-10$ K have been suggested as a possible requirement [70]. The cross section data burden associated with the problem described can exceed

100 GB of memory[2] - a value that exceeds the on-node memory capacity of most available HPC systems.

As a means of reducing the added memory burden introduced by the unionized grid method, a companion double indexing scheme was developed [48]. With this scheme, rather than storing each nuclide's cross section data on the dense unionized grid for the entire problem, a code must only store one dense grid - that of the unionized energies. Then, for each nuclide, a mapping is constructed from the indices of the problem's unionized energy grid to the indices of that nuclide's individual energy grid and cross sections. In this way, a single binary search on the unionized grid is still sufficient to determine the indices on individual nuclide grids, and without the added memory requirements of storing each nuclide's cross section data on the dense grid. Depending on the problem, this double indexing of grid values can reduce the memory burden of the repeated unionized grid method by roughly one half [48].

Recently, energy grid search algorithms making use of hashing functions have received renewed attention [22]. In this type of algorithm, as applied to the problem at hand, a function will take the neutron energy and map it directly to bounding indices on an energy hash grid. These hash grid indices are then mapped to indices on individual nuclide energy grids. By quickly mapping a neutron energy to a narrowed range of energies from a full grid, and then performing a binary search in that subset of energies[3], the time spent looking up an energy grid index can be reduced.

The search of an energy grid to determine the index corresponding to an energy grid point adjacent to the current neutron energy, and - after the interpolation step is complete - the cross section values at that energy, is computationally expensive with respect to speed. This is true whether a unionized or individual nuclide energy grid scheme is employed, though, as discussed, unionized schemes are typically faster. Depending on the problem, approximately 30% of the total runtime of a simulation performed with MCNP6 is spent looking up energy grid indices [22]. It has been suggested that this value can reach 85% [62] in full core reactor simulations performed

[2]This value may approach 1 TB if a unionized grid is employed.

[3]If the narrowed energy range has few enough points, a linear search may be faster.

with the OpenMC Monte Carlo neutron transport code[4] [58]. Development and demonstration of techniques for reducing this expense are the aims of this work.

In Section 4.2 we detail the theory behind the proposed energy grid search optimizations and briefly sketch out their implementation in OpenMC. Section 4.3 gives an overview of the test cases that we use to measure the effectiveness of the new optimization schemes as well as the numerical results of those tests. A concluding discussion is offered in Section 4.4.

4.2 Energy Grid Search Optimizations

In this section we propose optimizations to the energy grid search algorithm in continuous-energy Monte Carlo neutron transport codes. The theory behind each of the optimizations is presented along with brief descriptions of how they are implemented in the OpenMC code. Other considerations are addressed where appropriate. Also, it is important to note that none of the presented optimizations alter the physics of a simulation in any way. The optimizations do not change the energy grid indices that are identified by the energy grid lookup algorithm. All that is changed is the procedure by which those indices are determined.

4.2.1 Scattering Kinematics-Constrained Searches

The models used to treat neutron-nucleus scattering kinematics in Monte Carlo codes make reasonable assumptions about collision physics as well as the behavior of the modeled system, in general, in order to simplify calculations without appreciably impacting the accuracy of the results. Often, these assumptions place restrictions on the outcome of a particular interaction. We can take advantage of these constrained outcomes in designing methods for accelerating energy grid lookups. Two such methods are presented below.

[4]It should be noted that this increased value does not necessarily indicate an inherently slower energy grid search algorithm. Since we are quoting percentages of overall simulation runtimes, the differences in values between the two codes are almost certainly due to differences in the times spent performing other operations.

Stationary Target Nuclei

When the speed of an incident neutron is sufficiently high (i.e. much greater than the speeds associated with the thermal motion of target nuclei), the kinematics of a neutron-nucleus interaction can safely be treated as if the target nuclide is at rest in the laboratory frame of reference with a velocity of zero [19]. When the target-at-rest approximation is invoked, by conservation of energy and linear momentum, all elastic scattering events result in a reduction of the neutron's kinetic energy in the laboratory coordinate system. That is, upscattering is not possible.

Also, cases of threshold inelastic scattering,[5] which occur when the combined kinetic energy of the neutron-nucleus system in the center-of-mass reference frame is sufficient to reach the first excited level of the target nuclide and are characterized by a final constellation with total kinetic energy less than that of the initial constellation, should be considered. With a resting target, threshold inelastic scattering occurs at incident neutron energies of a few MeV in light nuclides and at tens of keV in heavier nuclides [37]. These energies are much greater than the energies of thermally agitated nuclei that are encountered in practical systems which are typically a fraction of an eV. Therefore, the kinematics of threshold inelastic scattering can be treated with the target-at-rest approximation, and, by conservation of energy, upscattering cannot occur.

So, we see that any scattering event in which the kinematics are treated with the target-at-rest approximation will result in a loss of neutron kinetic energy in the laboratory system. This fact can be exploited to construct an energy grid search optimization. In Monte Carlo codes, when handling scattering kinematics, it is a standard procedure to neglect the thermal motion of target nuclei when the incident neutron energy is above some threshold. This threshold is often taken to be $400k_BT$ for all nuclides except ^{1}H - to which no threshold is applied [68] - where k_B is the Boltzmann constant and T is the temperature of the material being traversed by the

[5]This type of inelastic scattering event is to be distinguished from the low energy inelastic scattering that arises as a result of the effects of chemical binding of target nuclei in molecules.

neutron[6]. When a neutron's incident energy is greater than this threshold, we know that, after a scattering event, it's energy will be less than it was before the event. This means that we do not have to search an entire nuclide or unionized energy grid to determine a post-scatter energy grid index. We only have to search the section of the energy grid corresponding to energies less than the pre-scatter neutron energy.

This is accomplished by first storing the index of the energy grid point - on a unionized or individual nuclide grid - just above the current neutron energy, E_{MAX}. The current energy is the maximum energy that the neutron can possibly have after its next scatter from a stationary nucleus and E_{MAX} is its nearest upper bound on the energy grid in question. Then, if the next event is a scatter, we simply have to search a subset of the energy grid, $\{E_1, E_2, ..., E_{\mathrm{MAX}}\}$, to find the post-collision neutron energy grid index.

Thermal Motion of Target Nuclei

At incident neutron energies below the the $400k_{\mathrm{B}}T$ target-at-rest cutoff, the thermal motion of target nuclei can no longer be ignored in the handling of scattering kinematics with negligible consequences and the upscattering of neutrons to higher-than-incident energies is now allowed. However, it is still possible to constrain the portion of a nuclide or unionized energy grid that must be searched to find the index of a post-scatter energy.

In generating effective, temperature dependent (i.e. Doppler broadened) cross section data for use in transport simulations, nuclear data processing codes typically employ some variation of the SIGMA1 algorithm for numerical Doppler broadening [32]. In the Doppler broadening algorithm, the target nuclei are assumed to behave as a free gas with isotropic velocities described by the Maxwell-Boltzmann distribution. It is further assumed that that the maximum energy of a target nuclide with which a neutron can interact is $16k_{\mathrm{B}}T$. This is because the probability of a particle existing in a Maxwell-Boltzmann free gas at an energy greater than $16k_{\mathrm{B}}T$ is negligible. Consistency requires that the same $16k_{\mathrm{B}}T$ maximum target energy is utilized by Monte

[6]As a point of reference, $400k_{\mathrm{B}}T$ for a material at 900 K translates to an energy of 31.0 eV.

Carlo codes when treating scattering kinematics.

With the maximum target nuclide energies, target nuclide masses, and range of incident neutron energies ($0-400k_BT$) all known, the kinematic equations can be used to determine the maximum energy to which a neutron can be upscattered in a given problem. This value, E_{upscat}, and the corresponding upper bound energy grid index, can be determined for a single nuclide, single material, or entire problem at simulation initialization. The energy grid search optimization based on this knowledge simply restricts the post-scatter searchable energy grid to $\{E_1, E_2, ..., E_{\text{upscat}}\}$ whenever the incident neutron energy is less than $400k_BT$, or whatever thermal motion cutoff energy is used in a particular simulation.

4.2.2 Material Energy Grid Unionization

The optimization presented in this section follows closely the unionized energy grid method [48] used in Serpent with the distinguishing feature being that a unionized energy grid and cross section values are constructed for each material, individually, rather than globally for the entire problem.

At any moment in a Monte Carlo transport simulation, the most cross section information that is needed is the total or reaction cross section for a single material, summed over the nuclides in that material. Therefore, at the time the required cross section is computed, the cross section data for all other materials are irrelevant. This implies that we can, at least[7], neglect the nuclides which are not in the current material when computing a cross section. The individual nuclide energy grid method outlined earlier exploits this naturally but the global unionized grid scheme does not.

Constructing unionized energy-cross section grids on a material-by-material basis preserves the key advantage of the global unionization - that is, a single binary search will identify the desired energy grid indices of multiple nuclides - while reducing the size of the energy grid that must be searched. In addition to the potential speedup associated with the search of a grid with a reduced length, this scheme can also

[7]If the cross sections for only a single nuclide are needed, as in sampling the type of reaction, all other nuclides in the problem are extraneous.

reduce memory because, for each nuclide, we must only store values on the unionized material grid instead of a global grid containing all energy points for all nuclides in the problem. A unionized material grid may also be used in conjunction with an appropriately-modified double indexing methodology which maps indices on the unionized material grid to indices on the individual grids of the nuclides in the material to further lessen memory demands.

There are a few caveats related to the use of unionized material grids worth noting here. With a global unionized grid, when a neutron streams into a new material, the grid indices for the nuclides in that material at the current energy are immediately known. They were determined in the last binary search that occurred, regardless of the material, because the global grid contains the energy points for all nuclides in the problem. That is, the computational expense for this immediate knowledge of indices is paid earlier in the simulation. With a unionized material grid, a binary search must occur whenever the neutron enters a new material, because, in general, knowledge of the previous material's energy indices is of no value in the current material.

Also, the benefits of a unionized material grid may be somewhat reduced in simulations where the same nuclide is contained in materials with different unionized grids. This is because a nuclide's energy index in one material's unionized grid will generally not be the same as its index in another material's unionized grid if the two grids are not identical. However, this is not as much of a handicap as it may, at first, appear to be. In the case of a depletion calculation, where there will be many different fuel material compositions (in terms of nuclide densities), the unionized material grid for those materials can actually be the same. When dealing in microscopic cross sections, it does not matter how much of a nuclide is present, just whether or not it is present, when deciding if storage of its cross sections is required. Additionally, in the case of non-isothermal fuel conditions, with the same nuclide existing at different temperatures, we are no worse off by using the unionized material grid than by using the global grid. The nuclides' different energy grids for the different temperatures must be stored in either case.

4.2.3 Energy Hash Table Searches

The optimizations presented in this section rely on hashing functions that take the neutron energy at which a cross section must be calculated and return the indices, into either an individual nuclide or globally unionized energy grid, which correspond to energies that bound the current energy. The following subsections discuss two simple hashing schemes, as well as the problem of determining an optimal hashing treatment.

Logarithmic Energy Spacing

This energy grid lookup optimization closely follows a recently reintroduced method [22]. In it, the energy range for which cross section data exist is divided logarithmically. This can be done on either a nuclide-by-nuclide or global grid basis. That is, in the individual nuclide case, the j^{th} nuclide's energy range is divided into N^j_{hash} equal lethargy bins of lethargy width

$$\Delta u^j_{\text{hash}} = \frac{\log\left(\frac{E^j_{N_{\text{grid}}}}{E^j_1}\right)}{N^j_{\text{hash}}} \tag{4.2}$$

where E^j_1 and $E^j_{N_{\text{grid}}}$ are the minimum and maximum values on the nuclide's energy grid, respectively. Then, as a pre-processing step, before the simulation begins, the indices on the nuclide's original energy grid that correspond to a set of coarse hash table energies $\{E^j_{H,1}, E^j_{H,2}, ..., E^j_{H,N^j_{\text{hash}}+1}\}$, where

$$E^j_{H,N^j_{\text{hash}}-i+1} = \frac{E^j_{N_{\text{grid}}}}{\exp\left(i\Delta u^j_{\text{hash}}\right)}, \text{ for } i = N^j_{\text{hash}}, N^j_{\text{hash}} - 1, ..., 1, 0, \tag{4.3}$$

are stored. Now, in a simulation, whenever a cross section must be computed, the indices in the coarse hash table that bound the current neutron energy can be determined directly from a rearranged form of Eq. 4.3. These hash indices are then mapped to the corresponding indices on the original nuclide energy grid. Once the subset of the original nuclide grid determined by these indices is isolated, a binary

search is performed within the subset to find the exact index for the current energy. This scheme increases the speed of energy grid lookups by reducing the length of the searchable grid to the length of the grid subset bounded by the hash table energies. As implemented in a beta release of MCNP6 [22], this method is applied to individual nuclide energy grid lookups. In this work, we implement the method in OpenMC and extend its use to global unionized grids.

Constant Energy Spacing

As will be discussed shortly, the choice of an effective hash table energy structure is not entirely straightforward. To first order, the number of cross section computations, and, thus, energy grid lookups, required by a simulation is proportional to the flux. In many systems, neutron slowing down results in a flux that approaches a constant value, in lethargy-space, over a wide range of energies. Without any information about the spacing of energy grid points, this suggests that the choice of logarithmically-spaced hash table energies is a good one. However, in order to capture the widely varying cross section structure encountered in different energy regions, energy grid points are generally not evenly spaced. This suggests that hash table structures other than logarithmic may be effective. With this in mind, a simple constant energy spacing hash table is also employed. The j^{th} nuclide's energy range is divided into N^j_{hash} bins spaced equally in energy with energy bin widths of

$$\Delta E^j_{\text{hash}} = \frac{E^j_{N_{\text{grid}}} - E^j_1}{N^j_{\text{hash}}}. \tag{4.4}$$

In a fashion similar to the case of constant lethargy spacing, we proceed by constructing the coarse hash table energy grid $\{E^j_{H,1}, E^j_{H,2}, ..., E^j_{H,N^j_{\text{hash}}+1}\}$, where

$$E^j_{H,i+1} = i\Delta E^j_{\text{hash}}, \text{ for } i = 0, 1, ..., N^j_{\text{hash}}. \tag{4.5}$$

The nuclide or unionized energy grid indices that correspond to the hash table energies are stored. Just as before, for a given neutron energy, the bounding hash table energies

can be directly calculated and mapped to the stored nuclide or unionized grid indices. These indices are then used as the bounds for a binary search on a subset of the original energy grid to arrive at the desired energy grid index and corresponding cross sections.

Optimal Energy Spacing

In the general case, selection of an optimal energy hash table structure - that is, the selection of an ensemble of hash table energy boundaries - is non-trivial. The optimized structure of table boundaries will be a complicated function of the energy spectrum of the neutron flux of the system as well as the density - in energy space - of the individual or unionized energy grids on which cross sections are represented for nuclides present in the system. To illustrate this point, we briefly outline the problem that must be solved in order to determine this optimal spacing of hash energies.

We start with the goal of minimizing the time spent in energy grid searches. That is, we wish to identify the sets of energy hash table boundaries in a problem with J total nuclides $\{E^j_{1/2}, E^j_{3/2}, ..., E^j_{N^j_{\text{hash}-1/2}}, E^j_{N^j_{\text{hash}+1/2}}\}$, where $E^j_{i-1/2}$ and $E^j_{i+1/2}$ are the energies that bound the i^{th} hash table bin of the j^{th} nuclide, that minimize the value of

$$T(\{\Delta E^j_i\}) = \sum_{j=1}^{J} \sum_{i=1}^{N^j_{\text{hash}}} L^j_i(\{\Delta E^j_i\}) \times \bar{t}^j_i(\{\Delta E^j_i\}), \tag{4.6}$$

the total time spent in a simulation performing energy grid lookups. Here, L^j_i and $\bar{t}^j_i$ stand, respectively, for the number of energy grid lookups that must be performed in energy hash bin i of nuclide j and the mean time spent in each of these lookups. Each of these quantities is dependent on the set of energy hash grid spacings $\{\Delta E^j_i\}$. With the $O(\log(N_{\text{grid}}))$ scaling of the binary search algorithm and an expression for the number of required lookups given by

$$L^j_i(\{\Delta E^j_i\}) \propto \int_{\Delta E^j_i} dE' \int_0^\infty dE \phi(E) f(E \rightarrow E'); \quad E' \in [E^j_{i-1/2}, E^j_{i+1/2}], \tag{4.7}$$

we are left wanting to minimize

$$T(\{\Delta E_i^j\}) \propto \sum_{j=1}^{J} \sum_{i=1}^{N_{\text{hash}}^j} \int_{\Delta E_i^j} dE' \int_0^\infty dE \phi(E) f(E \to E') \times \log\left(N_{\text{grid},i}^j(\{\Delta E_i^j\})\right);$$
$$E' \in [E_{i-1/2}^j, E_{i+1/2}^j]. \tag{4.8}$$

Solving this minimization problem is an exercise in non-linear optimization. In practice, it is not a problem we wish to solve as its solution depends in a complicated way on the physics of the system at hand. In particular, the transfer function $f(E \to E')$ varies with nuclide dependent, energy dependent cross section values and models of reaction physics. The number of energy grid points in a hash bin $N_{\text{grid},i}^j$ will also depend in a complicated fashion on energy boundaries as the energy spacing between energy grid-cross section pairs can fluctuate dramatically due to the need to capture resonance structure of cross sections with point-wise data. Finally, the energy dependent neutron flux $\phi(E)$ is likely the most problematic term in Eq. 4.8. It also is a function with strong variations in energy. To further complicate the matter, we do not have a solution for the flux, a priori, as we do, in theory, for the transfer function and number of energy grid points in a given interval. This means that, to solve the generalized energy hash spacing optimization problem so that we may speed up our simulation, we must first run the simulation with an un-optimized spacing so that the flux is determined[8]. This is not practical or desirable and will not be given further consideration here.

4.3 Results and Analysis

In this section we present the descriptions and results of simulations that are carried out in order to determine the effectiveness of the proposed optimizations in achieving their aim - overall code speedup. Speedup is defined here as the ratio of runtime for

[8]There are, of course, simplifying assumptions that can be made about the energy dependence of cross sections and flux that would facilitate a more tractable problem formulation. This is beyond the scope of the present work.

a simulation with no energy grid search optimizations employed to the runtime of the same simulation in which an optimization scheme is applied. An attempt is made to select test problems that exhibit physical phenomena relevant to practical reactor physics and criticality applications. With this in mind, all of the models for the test problems shown here are derived from the Benchmark for Evaluation and Analysis of Reactor Simulations (BEAVRS) [44].

Continuous-energy point-wise cross section data from the endf71x nuclear data library [27] are utilized in all simulations. Thermal scattering from ^{1}H nuclei bound in light-water molecules is treated with the continuous $S(\alpha, \beta)$ data found in the ENDF71SaB library [28]. These libraries consist of ENDF/B-VII.1 nuclear data [24] processed with the NJOY code system [50].

All simulations are run with 20 batches, 10 of which are inactive. Each batch consists of 1E4 particles. A spatially uniform fission source over fissionable material is used to initiate each run. For the problems considered, these simulation parameters are typically sufficient to converge on an effective multiplication factor eigenvalue, k_{eff}, in the inactive cycles[9]. For the hashing optimization runs, the presented results are obtained with 1E4 bins[10]. The only tallies present in these simulations are those for k_{eff}. Verification that the introduction of each optimization does not alter the physics of a simulation - as it shouldn't[11] - is carried out by simply confirming that, for every batch, the batch eigenvalue, the batch Shannon entropy, the running mean eigenvalue, and the running standard error of the mean eigenvalue are all unchanged from the reference cases.

The reference results are obtained from release 0.6.0 of OpenMC [13]. Each of the optimization methods is implemented in its own branch of that same release. These

[9]This is not a primary concern, even if they are not. Because we are only monitoring code speedup, and the reference and optimization cases are the exact same problems from a physical standpoint, runtime results would still be relevant and of interest. Results from a simulation with an unconverged eigenvalue would indicate the speedup to be expected in the period of the simulation in which the eigenvalue is converging.

[10]Comparable results, and a reduction in the already-modest memory overhead, can be obtained with a reduction to 1E3 bins.

[11]As we have discussed, the energy grid index found with each algorithm does not differ from the reference case - only the method by which that index is identified does.

branches are not modified in any way other than to implement one specific optimization method apiece; no combined optimizations are tested. All code is compiled with release 4.8.2 of the GNU Fortran (GFortran) compiler [4]. Every simulation is performed in serial on an Intel Core i7-3740QM Processor [6].

4.3.1 Fresh Pin Cell

First, we extract a 3.1 wt.% enriched clean fuel pin cell at beginning-of-core, hot zero-power conditions, with fuel and all other materials at 600 K, from the full BEAVRS model. Reflective boundary conditions are applied on each of the six pin cell faces resulting in an infinite two-dimensional array of this pin cell type. Overall code speedup results for each of the optimizations relative to OpenMC 0.6.0 global and nuclide grid reference cases are given in Table 4.1. Except for the material union grid runs, in this and all other test problems, all optimization runs are performed with the same energy grid unionization scheme - global or individual nuclide - as the reference cases to which they are compared. The unionized material grid optimization runs are performed with no other optimizations than the use of a material grid. Noticeable speedup is observed for each of the optimizations. The hashing methods are somewhat less effective in accelerating energy grid searches on the global grid. Relative to the nuclide grid reference case, each optimization speeds up the code, on an overall basis, by at least a factor of 1.1, with the material grid run seeing a speedup of more than 1.3.

Optimization	Reference - Global Grid	Reference - Nuclide Grids
Kinematics	1.08	1.12
Material Grid	1.19	1.31
Lethargy Hash	1.02	1.23
Energy Hash	1.02	1.14

Table 4.1: Overall Code Speedup for Fresh Pin Cell Simulations

4.3.2 Depleted Pin Cell

Next, we examine the speedup from applying each of the optimizations to a simulation of the same two-dimensional, infinite pin cell array as in the previous section, except with 900 K fuel at a burnup of 20.0 MWd/kg.[12] In this case, the hashing optimizations offer speedups of ~1.2, while the material union grid speeds the overall simulation up by a factor greater than 1.6 relative to the nuclide energy grid reference case. All results can be seen in Table 4.2.

Optimization	Reference - Global Grid	Reference - Nuclide Grids
Kinematics	1.06	1.09
Material Grid	1.39	1.63
Lethargy Hash	1.03	1.26
Energy Hash	1.01	1.18

Table 4.2: Overall Code Speedup for Depleted Pin Cell Simulations

4.3.3 Unmoderated Depleted Pin Cell

Here, we take the infinite depleted pin cell lattice from the previous section and remove all of the moderator. Also, all materials are set to 300 K. Again, we see the unionized material grid speed up the simulation more than the other methods. However, all methods result in a speedup by a factor of at least 1.2 relative to the individual nuclide grid reference case. Also, possibly owing to the harder spectrum of this model and a resultant flux that does not exhibit as definite of a $1/E$ form as a more moderated system, the constant energy hashing scheme slightly outperforms that with the constant lethargy spacing.

[12]Nuclide densities for the fuel in this simulation are obtained by running a depletion calculation for the same model with the CASMO-5 lattice physics code [56].

Optimization	Reference - Global Grid	Reference - Nuclide Grids
Kinematics	1.11	1.20
Material Grid	1.23	1.43
Lethargy Hash	1.01	1.25
Energy Hash	1.08	1.28

Table 4.3: Overall Code Speedup for Unmoderated, Depleted Pin Cell Simulations

4.3.4 Fresh Full Core

In order to see the effect of model complexity on the attainable speedup with each of the optimizations, we now look at the three-dimensional, full core BEAVRS model. The speedup factors are, in general, reduced relative to the pin cell case. This is expected as simulations of greater complexity involve particle tracking across many more geometric regions and materials, thus increasing the fraction of time spent by a code performing operations other than energy grid lookups. This reduction in the relative time spent searching on energy grids results in lower speedup factors. Still though, all of the methods exhibit speedup of approximately 1.1.

Optimization	Reference - Global Grid	Reference - Nuclide Grids
Kinematics	1.04	1.06
Material Grid	1.13	1.23
Lethargy Hash	1.02	1.16
Energy Hash	1.01	1.09

Table 4.4: Overall Code Speedup for Fresh Full Core Simulations

4.3.5 Depleted Full Core

Finally, we take a look at the speedup attained with each of the individual optimizations applied to a three-dimensional, full core BEAVRS model that has been depleted

out to 20.0 MWd/kg[13]. For the first time, an overall code slowdown is observed with an optimization scheme applied. However, the observed slowdowns - which only occur with the hashing schemes, and only when they are compared to the global union grid reference case - are relatively minor. The material union grid optimization again performs well, offering speedups of ~1.2 and ~1.4 relative to the global and nuclide grid reference cases, respectively.

Optimization	Reference - Global Grid	Reference - Nuclide Grids
Kinematics	1.00	1.06
Material Grid	1.22	1.43
Lethargy Hash	0.97	1.14
Energy Hash	0.95	1.07

Table 4.5: Overall Code Speedup for Depleted Full Core Simulations

4.4 Conclusions

The presented extensions of energy grid lookup algorithm optimizations are shown to result in non-negligible overall speedup of the OpenMC Monte Carlo neutron transport code in simulations of practical systems. If the additional memory requirement is permissible, applying the unionized energy grid scheme on a material-by-material basis is an attractive alternative to the use of individual nuclide grids. Additionally, the material union grids offer performance that is superior to the global union grid for all of the systems considered here. The kinematic constraint on energy grid lookups offers non-negligible overall code speedup at the expense of only a few kB for even the most demanding depletion calculations. This trivial modification should be part of the default energy grid lookup algorithm in point-wise continuous-energy Monte Carlo codes. The energy hash table methods also provide simple means of speeding up simulations that can be relatively cheap from a memory perspective.

[13]As in the pin cell case, the nuclide densities for this model are obtained from a CASMO-5 depletion calculation.

However, as the speedup with respect to the global unionized grid runs is typically minimal, with slight code slowdowns observed in some cases, application of these methods should likely be restricted to simulations in which individual energy grids for each nuclide are used. With moderate levels of speedup observed with each of the proposed optimizations, individually, investigation of applying these techniques in concert with one another is warranted. Also, efforts to move away from point-wise energy grid data, altogether, as through direct cross section value calculations using the multipole resonance formalism [46], should receive attention.

Chapter 5

Conclusions

In this thesis three computational challenges regarding the efficient management of nuclear data in Monte Carlo neutron transport simulations are explored. At least one method is proposed and analyzed for tackling each of the three challenges. In all cases but that of the nuclear data server algorithm, the proposed methods are also implemented in the OpenMC Monte Carlo code and demonstrated to achieve their goal of improved efficiency with respect to either code speedup or reduced memory requirements. This section provides brief summaries of the computational methods that have been explored in this work. It concludes with a discussion of areas for future computational methods research, focusing on the treatment of nuclear data in Monte Carlo codes.

5.1 Nuclear Data Storage and Communication

The nuclear data memory requirements of large reactor physics simulations - mainly in the form of neutron cross sections and secondary angular and energy distributions - exceed the on-node memory available on most current and proposed high-performance computing architectures. Profiling studies investigating the details of these memory requirements are conducted in this work. Also, OpenMC is modified to record the frequencies with which the nuclear data associated with these memory burdens are accessed. The results of the profiling studies are in the form of a set of tallies that give

the access frequencies of different blocks of nuclear data and the memory requirements for each block.

To reduce on-node memory requirements, a conceptual data decomposition algorithm which would make use of dedicated nuclear data server nodes is proposed and analyzed. The algorithm relies on the infrequent access of many blocks of nuclear data. The blocks that are accessed rarely within a simulation can be stored on server nodes and communicated to compute nodes only as needed.

A communication model for the proposed data server algorithm is also derived. Using the model, performance predictions relating on-node memory reduction to simulation runtime overhead are made. These predictions indicate that, in a variety of practical scenarios, with quite tolerable runtime overhead penalties, on-node memory requirements can be significantly reduced via the application of a nuclear data server algorithm.

5.2 Resonance Elastic Scattering Models

In order to account for their thermal motion when calculating the kinematics of an elastic scattering event, target nuclei are typically modeled as existing in a free gas characterized by isotropic direction-of-flight vectors and speeds drawn from the Maxwell-Boltzmann ideal gas distribution. The bivariate secondary energy-angle distribution for a neutron that undergoes an elastic scattering event depends on the assumptions of the free gas model as well as the energy dependent 0 K elastic scattering cross section of the scatterer.

To simplify the sampling of this distribution, the energy dependence of the scattering cross section is often neglected. It is inappropriate to apply this simplification to heavy resonant nuclides which exhibit fine cross section structure in the energy variable. One of the end results of not accounting for the rapid cross section variations (i.e. resonances) that occur in narrow energy intervals is an underprediction of upscatter rates at incident neutron energies just below resonance values. This effect is exacerbated by increased temperature. Underprediction of upscatter rates just below

resonance energies means underprediction of capture rates in those resonances. The biased capture rates then show up as biases in important system design metrics such as the fuel temperature coefficient.

The Doppler broadening rejection correction (DBRC), a recently adopted method that avoids the approximation of the 0 K elastic scattering cross section as a constant in energy, is computationally costly. It requires the rejection sampling of 0 K resonances. This one inefficient sampling routine is observed to increase overall runtime by ~15% in practical simulations of thermal reactor systems.

A new, accelerated resonance elastic scattering kernel rejection sampling technique is derived, implemented, and verified in this work. The new scheme is shown to reproduce the exact free gas resonance elastic scattering kernels. Additionally, for thermal reactor pin cell and full core systems, it is demonstrated to reduce runtime overhead by more than 30% relative to simulations performed with the DBRC.

5.3 Energy Grid Searches

Throughout the course of a simulation, nuclide cross section values are needed in order to determine the probabilities of various neutron-nucleus interactions and also compute tally results. This requires that energy grid searches be performed so that the index of the energy nearest the current neutron energy is identified. The energy grid lookup procedure is computationally intensive and occupies a significant fraction of overall simulation runtime. To reduce the time spent in performing energy grid searches, three methods have been proposed, implemented, and demonstrated.

First, a kinematic constraint on the searchable energy grid range is suggested to reduce the number of energy points that must be searched. Next, the energy grid unionization concept is extended to materials to reduce the number of extraneous grid points from nuclides that are not present in the neutron's current material. Lastly, hashing methods are investigated as a means for reducing the energy ranges that are needed in a final binary search step. All three of the acceleration techniques are shown to give non-negligible overall code speedup in simulations of practical, reactor

physics-relevant systems.

5.4 Future Research

Many opportunities exist for the research and development of computational methods for the treatment of nuclear data in Monte Carlo neutron transport simulations. Improvements in the computational efficiency of large, high-fidelity simulations as well as the physics models used in those simulations are both desirable. The following subsections describe, at a high level, areas that are in need of attention.

5.4.1 Nuclear Data Decomposition

Continuation of the research into the nuclear data server algorithm presented in this thesis is warranted. With the derived communication model predicting attractive on-node memory requirement reductions with the incurrence of only modest run-time overhead, the task of implementing the proposed algorithm in a Monte Carlo code should be undertaken. The implementation of the algorithm with allow for the verification of performance trends that are only predicted theoretically in this work.

5.4.2 Resonance Elastic Scattering Kernel Sampling

The development of the accelerated resonance elastic scattering kernel sampling scheme, as presented, is complete[1]. However, there are valuable sensitivity studies and further analyses to be carried out. On a nuclide-by-nuclide basis, determination of the energy range over which resonance elastic scattering effects are important will allow for narrowed application of the method, and, due to the resonance scattering treatment methods' computational expense, a corresponding reduction in simulation runtime. These energy ranges will depend on temperature. They will also depend on the spectrum of the system being simulated. Finally, investigation of resonance

[1]Some work remains to make the procedure for directly sampling the 0 K elastic scattering cross section compatible with continuous - rather than point-wise - cross section representations, such as those allowed by the multipole formalism. A simple solution is to accept a slight memory increase and generate only the 0 K cross section values on an energy grid.

elastic scattering, and the methods for modeling it, in *very* hot, fast systems may reveal a dramatic advantage in applying the new sampling routine, rather than the DBRC.

5.4.3 Energy Grid Searches

Further investigation of the energy grid search optimizations described in this thesis may be fruitful. A more explicit accounting of the memory requirements associated with each of the methods should be performed. Scaling studies on shared-memory and distributed-memory computing systems are also desirable. Additionally, the application of fractional cascading algorithms [26] to energy grid lookups in Monte Carlo neutron transport codes is an area worth exploring [38].

5.4.4 Unresolved Resonance Energy Region Nuclear Data Methods

For incident neutron energies that are sufficiently high, but still below the fast energy region, there is resonance structure in reaction cross sections that cannot be resolved with current experimental capabilities. This provides a challenge for the simulation of systems with an appreciable population of neutrons at these intermediate energies. The problem is that, at any given energy in the unresolved region, cross section values are not known within, roughly, an order of magnitude. However, the theoretical statistical distributions describing cross section value probabilities are known.

It is common procedure in Monte Carlo codes to treat cross sections in the unresolved region through the use of the so-called *probability table* method which relies on discrete distributions for the sampling of cross section values. The data required by the method - cross section values and associated probabilities at discrete energies - are pre-generated and loaded by the code at initialization.

An alternate approach is to generate unresolved resonance region cross section values on-the-fly, within the Monte Carlo simulation, directly from the unresolved resonance parameters and statistical distributions. This will reduce memory require-

ments by eliminating the need for probability table storage, and also provide cross section distributions that are continuous in both energy-space and cross section-space, in contrast to the discrete probability tables. This on-the-fly treatment can also be extended to investigate the effects of varying the frequency with which resonance realizations are generated - per simulation, per cycle, per neutron, continuously, etc. Such an investigation may expose important issues regarding unresolved resonance region cross section uncertainties and more fundamental statistical concerns.

5.4.5 Fast Energy Region Nuclear Data Methods

Secondary angular and energy distribution data comprise a significant portion of the nuclear data memory requirements in the types of large reactor physics simulations that are dealt with in this thesis. Many of these data are for threshold reactions which occur in the fast neutron energy range. Because current theoretical nuclear reaction models lack the general predictive capabilities to produce nuclear data that can be used with confidence in real simulations, tabulated data based on semi-empirical nuclear reaction systematics models are commonly used. Work towards improved nuclear reaction models, and computational methods for handling the data produced by those models, gives rise to a wealth of exciting research opportunities.

Appendix A

Fuel Nuclide Densities

A.1 End-of-Core Hot Full-Power BEAVRS

A.1.1 1.6 wt% Enriched UO_2 Fuel at 20.0 MWd/kg

Nuclide	Density $[\frac{\text{atoms}}{\text{cm}^3}]$
O-16	4.58964E+22
O-17	1.11800E+20
U-234	2.09669E+18
U-235	9.55151E+19
U-236	4.31429E+19
U-237	1.58907E+17
U-238	2.17719E+22
U-239	1.92551E+16
U-240	6.30245E+12
Np-235	3.22225E+10
Np-236	8.15227E+12
Np-237	3.92605E+18
Np-238	1.77895E+16
Np-239	2.77510E+18

Pu-237	8.54449E+10
Pu-238	1.02917E+18
Pu-239	9.28323E+19
Pu-240	4.20613E+19
Pu-241	2.06315E+19
Pu-242	7.85072E+18
Pu-243	2.63035E+15
Pu-244	2.10043E+14
Am-241	2.92505E+17
Am-242	1.43887E+15
Am-243	9.87148E+17
Am-244	1.36265E+15
Am-242m	4.97654E+15
Cm-242	1.25214E+17
Cm-243	2.23803E+15
Cm-244	2.24374E+17
Cm-245	7.80786E+15
Cm-246	6.08964E+14
Cm-247	4.00312E+12
Cm-248	1.49986E+11
Se-79	1.83406E+17
Kr-83	1.53791E+18
Kr-85	8.95529E+17
Sr-89	1.84778E+18
Sr-90	1.80753E+19
Y-91	2.88033E+18
Zr-93	2.35348E+19
Zr-94	2.49976E+19

Zr-95	4.64665E+18
Zr-96	2.63278E+19
Mo-95	1.83277E+19
Mo-98	2.70451E+19
Mo-99	2.33572E+17
Mo-100	3.02836E+19
Tc-99	2.62706E+19
Ru-101	2.54787E+19
Ru-102	2.66365E+19
Ru-103	3.08954E+18
Ru-105	1.11197E+16
Ru-106	7.90993E+18
Rh-103	1.50707E+19
Rh-105	7.62794E+16
Ag-109	1.56804E+18
Ag-111	2.47894E+16
Ag-110m	1.48487E+16
Sn-126	5.22810E+17
Sb-125	2.52888E+17
Sb-126	7.92986E+14
Te-127m	2.63231E+16
Te-129m	6.06336E+16
I-127	1.13821E+18
I-129	3.73121E+18
I-131	3.74950E+17
I-135	2.47010E+16
Xe-131	1.14898E+19
Xe-133	4.94736E+17

Xe-134	3.64341E+19
Xe-135	4.91410E+15
Xe-136	5.85199E+19
Cs-133	2.81113E+19
Cs-134	2.36267E+18
Cs-135	4.50444E+18
Cs-136	1.71142E+16
Cs-137	2.92161E+19
Ba-138	2.98876E+19
Ba-140	9.94713E+17
La-139	2.78794E+19
La-140	1.34238E+17
Ce-141	2.38567E+18
Ce-142	2.54747E+19
Ce-143	9.21818E+16
Ce-144	1.25505E+19
Pr-143	9.00879E+17
Nd-143	1.69535E+19
Nd-144	1.54557E+19
Nd-145	1.50613E+19
Nd-147	3.16164E+17
Nd-148	8.22980E+18
Pm-147	5.15217E+18
Pm-148	4.42691E+16
Pm-149	5.46007E+16
Pm-148m	3.43115E+16
Sm-147	9.21923E+17
Sm-148	1.89646E+18

Sm-149	4.73395E+16
Sm-150	6.29298E+18
Sm-151	2.36623E+17
Sm-152	2.82301E+18
Eu-153	2.39299E+18
Eu-154	4.07335E+17
Eu-155	1.66421E+17
Eu-156	1.34770E+17
Gd-155	5.26308E+14
Gd-157	1.64545E+15

A.1.2 2.4 wt% Enriched UO_2 Fuel at 20.0 MWd/kg

Nuclide	Density $[\frac{\text{atoms}}{\text{cm}^3}]$
O-16	4.58296E+22
O-17	1.11637E+20
U-234	3.52418E+18
U-235	2.01044E+20
U-236	5.73278E+19
U-237	1.69800E+17
U-238	2.16206E+22
U-239	1.65013E+16
U-240	4.65979E+12
Np-235	2.94504E+10
Np-236	9.03969E+12
Np-237	4.16232E+18
Np-238	1.58087E+16

Np-239	2.37935E+18
Pu-237	8.33658E+10
Pu-238	8.92180E+17
Pu-239	9.82269E+19
Pu-240	3.55892E+19
Pu-241	1.81491E+19
Pu-242	4.92279E+18
Pu-243	1.48568E+15
Pu-244	8.79904E+13
Am-241	2.68947E+17
Am-242	1.07415E+15
Am-243	5.30807E+17
Am-244	6.41887E+14
Am-242m	4.62340E+15
Cm-242	8.68812E+16
Cm-243	1.31288E+15
Cm-244	9.88503E+16
Cm-245	3.18958E+15
Cm-246	1.76770E+14
Cm-247	9.96658E+11
Cm-248	3.01112E+10
Se-79	1.91576E+17
Kr-83	1.76301E+18
Kr-85	9.96533E+17
Sr-89	2.19280E+18
Sr-90	2.03674E+19
Y-91	3.32757E+18
Zr-93	2.52929E+19

Zr-94	2.65605E+19
Zr-95	4.95423E+18
Zr-96	2.74375E+19
Mo-95	1.92436E+19
Mo-98	2.73467E+19
Mo-99	2.36434E+17
Mo-100	3.03639E+19
Tc-99	2.68168E+19
Ru-101	2.53246E+19
Ru-102	2.56213E+19
Ru-103	2.81959E+18
Ru-105	9.45306E+15
Ru-106	6.31152E+18
Rh-103	1.43835E+19
Rh-105	6.68801E+16
Ag-109	1.17823E+18
Ag-111	1.96990E+16
Ag-110m	9.40221E+15
Sn-126	4.62352E+17
Sb-125	2.25927E+17
Sb-126	7.15743E+14
Te-127m	2.82857E+16
Te-129m	5.91542E+16
I-127	1.02926E+18
I-129	3.44974E+18
I-131	3.68814E+17
I-135	2.48102E+16
Xe-131	1.17161E+19

Xe-133	4.99416E+17
Xe-134	3.68360E+19
Xe-135	6.25434E+15
Xe-136	5.67809E+19
Cs-133	2.87277E+19
Cs-134	2.08419E+18
Cs-135	5.78300E+18
Cs-136	1.61285E+16
Cs-137	2.92790E+19
Ba-138	3.05276E+19
Ba-140	1.02708E+18
La-139	2.86084E+19
La-140	1.37902E+17
Ce-141	2.46741E+18
Ce-142	2.62094E+19
Ce-143	9.75925E+16
Ce-144	1.32843E+19
Pr-143	9.55616E+17
Nd-143	1.91234E+19
Nd-144	1.47614E+19
Nd-145	1.59001E+19
Nd-147	3.25277E+17
Nd-148	8.21943E+18
Pm-147	5.66653E+18
Pm-148	4.16685E+16
Pm-149	5.22581E+16
Pm-148m	3.99122E+16
Sm-147	1.01256E+18

Sm-148	1.76822E+18
Sm-149	6.00326E+16
Sm-150	6.15902E+18
Sm-151	2.75155E+17
Sm-152	2.72770E+18
Eu-153	2.16582E+18
Eu-154	3.59051E+17
Eu-155	1.41821E+17
Eu-156	9.73768E+16
Gd-155	5.87823E+14
Gd-157	1.64226E+15

A.1.3 3.1 wt% Enriched UO_2 Fuel at 20.0 MWd/kg

Nuclide	Density $[\frac{\text{atoms}}{\text{cm}^3}]$
O-16	4.58519E+22
O-17	1.11691E+20
U-234	4.87192E+18
U-235	3.16393E+20
U-236	6.68074E+19
U-237	1.71766E+17
U-238	2.15066E+22
U-239	1.46586E+16
U-240	3.68601E+12
Np-235	2.73239E+10
Np-236	9.61000E+12
Np-237	4.24131E+18

Np-238	1.39912E+16
Np-239	2.11437E+18
Pu-237	8.18476E+10
Pu-238	7.88298E+17
Pu-239	1.02317E+20
Pu-240	3.09401E+19
Pu-241	1.61670E+19
Pu-242	3.40500E+18
Pu-243	9.42683E+14
Pu-244	4.54600E+13
Am-241	2.46518E+17
Am-242	8.31673E+14
Am-243	3.29947E+17
Am-244	3.59501E+14
Am-242m	4.25465E+15
Cm-242	6.45228E+16
Cm-243	8.63327E+14
Cm-244	5.37227E+16
Cm-245	1.64070E+15
Cm-246	7.01905E+13
Cm-247	3.58275E+11
Se-79	1.96494E+17
Kr-83	1.90655E+18
Kr-85	1.05639E+18
Sr-89	2.41663E+18
Sr-90	2.17226E+19
Y-91	3.61536E+18
Zr-93	2.63422E+19

Zr-94	2.74943E+19
Zr-95	5.14957E+18
Zr-96	2.81132E+19
Mo-95	1.97899E+19
Mo-98	2.75491E+19
Mo-99	2.38284E+17
Mo-100	3.04447E+19
Tc-99	2.71827E+19
Ru-101	2.52720E+19
Ru-102	2.50630E+19
Ru-103	2.64067E+18
Ru-105	8.32689E+15
Ru-106	5.37361E+18
Rh-103	1.40111E+19
Rh-105	6.01513E+16
Ag-109	9.58190E+17
Ag-111	1.63744E+16
Ag-110m	6.81165E+15
Sn-126	4.26924E+17
Sb-125	2.09970E+17
Sb-126	6.69448E+14
Te-127m	2.98000E+16
Te-129m	5.78318E+16
I-127	9.65942E+17
I-129	3.28472E+18
I-131	3.64274E+17
I-135	2.48843E+16
Xe-131	1.18806E+19

Xe-133	5.02400E+17
Xe-134	3.71181E+19
Xe-135	7.61404E+15
Xe-136	5.53552E+19
Cs-133	2.91450E+19
Cs-134	1.89785E+18
Cs-135	7.01775E+18
Cs-136	1.54866E+16
Cs-137	2.93526E+19
Ba-138	3.09425E+19
Ba-140	1.04902E+18
La-139	2.90760E+19
La-140	1.40376E+17
Ce-141	2.52089E+18
Ce-142	2.66661E+19
Ce-143	1.01249E+17
Ce-144	1.37367E+19
Pr-143	9.92598E+17
Nd-143	2.05618E+19
Nd-144	1.42160E+19
Nd-145	1.64286E+19
Nd-147	3.31560E+17
Nd-148	8.22443E+18
Pm-147	6.01916E+18
Pm-148	3.91804E+16
Pm-149	5.03738E+16
Pm-148m	4.46165E+16
Sm-147	1.07339E+18

Sm-148	1.67353E+18
Sm-149	7.34659E+16
Sm-150	6.05843E+18
Sm-151	3.17155E+17
Sm-152	2.66374E+18
Eu-153	2.01692E+18
Eu-154	3.26138E+17
Eu-155	1.25523E+17
Eu-156	7.57529E+16
Gd-155	6.57225E+14
Gd-157	1.69660E+15

Bibliography

[1] Benchmark for Evaluation and Validation of Reactor Simulations (BEAVRS), August 2014. http://crpg.mit.edu/pub/beavrs.

[2] China Nuclear Data Center, August 2014. http://www.ciae.ac.cn/eng/CIAE/index.htm.

[3] Cross Section Evaluation Working Group (CSEWG), August 2014. http://www.nndc.bnl.gov/csewg/.

[4] GNU Fortran, July 2014. https://gcc.gnu.org/fortran/.

[5] Institute of Physics and Power Engineering, August 2014. http://www.ippe.obninsk.ru/podr/cjd/.

[6] Intel Core i7 processor specifications, July 2014. http://www.intel.com/content/www/us/en/processors/core/core-i7-processor/Corei7Specifications.html.

[7] Japan Atomic Energy Agency, August 2014. http://www.jaea.go.jp/english/index.html.

[8] JEFF-3.2 evaluated data library - Neutron data, August 2014. https://www.oecd-nea.org/dbforms/data/eva/evatapes/jeff_32/.

[9] National Nuclear Data Center, August 2014. http://www.nndc.bnl.gov/.

[10] Nuclear Data Viewer, August 2014. http://t2.lanl.gov/nis/data/ndviewer.html.

[11] Nuclear Energy Agency, August 2014. https://www.oecd-nea.org/.

[12] Nuclear Research and Consultancy Group, August 2014. http://www.nrg.eu/.

[13] OpenMC Monte Carlo particle transport code, July 2014. https://github.com/mit-crpg/openmc.

[14] RUSFOND-2010, August 2014. http://www.ippe.ru/podr/abbn/english/libr/intr-rosfond.php.

[15] TENDL-2013: TALYS-based evaluated nuclear data library, August 2014. http://www.talys.eu/tendl-2013.html.

[16] Titan Cray XK7, August 2014. https://www.olcf.ornl.gov/computing-resources/titan-cray-xk7/.

[17] B. Becker. *On the Influence of the Resonance Scattering Treatment in Monte Carlo Codes on High Temperature Reactor Characteristics.* PhD thesis, University of Stuttgart, Germany, 2010.

[18] B. Becker, R. Dagan, and G. Lohnert. Proof and implementation of the stochastic formula for ideal gas, energy dependent scattering kernel. *Annals of Nuclear Energy*, 36:470–474, 2009.

[19] George I. Bell and Samuel Glasstone. *Nuclear Reactor Theory.* Division of Technical Information, United States Atomic Energy Commission, 1970.

[20] A.I. Blokhin, A.V. Ignatyuk, V.N. Manokhin, M.N. Nikolaev, and V.G. Pronyaev. *BROND-2.2, Russian Evaluated Neutron Reaction Data Library.* IAEA-NDS-90 Rev.8. International Atomic Energy Agency, 1994.

[21] R.N. Blomquist, R.M. Lell, and E.M. Gelbard. *VIM - A Continuous Energy Monte Carlo Code at ANL.* ORNL/RSIC-44. April 21-23, 1980.

[22] Forrest B. Brown. *New Hash-Based Energy Lookup Algorithm for Monte Carlo Codes.* LA-UR-14-24530. Los Alamos National Laboratory, 2014.

[23] L.L. Carter and E.D. Cashwell. *Particle-Transport Simulation with the Monte Carlo Method.* TID-26607, ERDA Critical Review Series. U. S. Energy Research and Development Administration, Technical Information Center, Oak Ridge, TN, 1975.

[24] M.B. Chadwick, M. Herman, P. Obloinsk, M.E. Dunn, Y. Danon, A.C. Kahler, D.L. Smith, B. Pritychenko, G. Arbanas, R. Arcilla, R. Brewer, D.A. Brown, R. Capote, A.D. Carlson, Y.S. Cho, H. Derrien, K. Guber, G.M. Hale, S. Hoblit, S. Holloway, T.D. Johnson, T. Kawano, B.C. Kiedrowski, H. Kim, S. Kunieda, N.M. Larson, L. Leal, J.P. Lestone, R.C. Little, E.A. McCutchan, R.E. MacFarlane, M. MacInnes, C.M. Mattoon, R.D. McKnight, S.F. Mughabghab, G.P.A. Nobre, G. Palmiotti, A. Palumbo, M.T. Pigni, V.G. Pronyaev, R.O. Sayer, A.A. Sonzogni, N.C. Summers, P. Talou, I.J. Thompson, A. Trkov, R.L. Vogt, S.C. van der Marck, A. Wallner, M.C. White, D. Wiarda, and P.G. Young. ENDF/B-VII.1 nuclear data for science and technology: Cross sections, covariances, fission product yields and decay data. *Nuclear Data Sheets*, 112:2887–2996, 2011.

[25] M.B. Chadwick, P. Obloinsk, M. Herman, N.M. Greene, R.D. McKnight, D.L. Smith, P.G. Young, R.E. MacFarlane, G.M. Hale, S.C. Frankle, A.C. Kahler, T. Kawano, R.C. Little, D.G. Madland, P. Moller, R.D. Mosteller, P.R. Page,

P. Talou, H. Trellue, M.C. White, W.B. Wilson, R. Arcilla, C.L. Dunford, S.F. Mughabghab, B. Pritychenko, D. Rochman, A.A. Sonzogni, C.R. Lubitz, T.H. Trumbull, J.P. Weinman, D.A. Brown, D.E. Cullen, D.P. Heinrichs, D.P. McNabb, H. Derrien, M.E. Dunn, N.M. Larson, L.C. Leal, A.D. Carlson, R.C. Block, J.B. Briggs, E.T. Cheng, H.C. Huria, M.L. Zerkle, K.S. Kozier, A. Courcelle, V. Pronyaev, and S.C. van der Marck. ENDF/B-VII.0: next generation evaluated nuclear data library for nuclear science and technology. *Nuclear Data Sheets*, 107:2931–3060, 2006.

[26] Bernard Chazelle and Leonidas J. Guibas. Fractional cascading: I. a data structuring technique. *Algorithmica*, 1:133–162, 1986.

[27] Jeremy Lloyd Conlin, D. Kent Parsons, Steven J. Gardiner, Mark Gray, A.C. Kahler, M. Beth Lee, and Morgan C. White. *Continuous Energy Neutron Cross Section Data Tables Based upon ENDF/B-VII.1.* LA-UR-13-20137. Los Alamos National Laboratory, 2013.

[28] Jeremy Lloyd Conlin and Donald K. Parsons. *Release of Continuous Representation for $S(\alpha, \beta)$ ACE Data.* LA-UR-14-21878. Los Alamos National Laboratory, 2014.

[29] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. *Introduction to Algorithms.* MIT Press, 2009.

[30] R.R. Coveyou, J.G. Sullivan, and H.P. Carter. *The Oak Ridge Random Research Reactor Routine (O5R): A General-Purpose Monte Carlo Code for the IBM-704.* ORNL-2609, TID-4500. Oak Ridge National Laboratory, 1958.

[31] S.N. Cramer. The KENO-V code. In *Monte Carlo Methods in Nuclear Reactor Analysis*, 1984.

[32] D.E. Cullen and C.R. Weisbin. Exact Doppler broadening of tabulated cross sections. *Nuclear Science and Engineering*, 60.3:199–229, 1976.

[33] R. Dagan. On the use of $S(\alpha, \beta)$ tables for nuclides with well pronounced resonances. *Annals of Nuclear Energy*, 32:367–377, 2005.

[34] James J. Duderstadt and William R. Martin. *Transport Theory.* Wiley, 1979.

[35] J.J. Duderstadt and L.J. Hamilton. *Nuclear Reactor Analysis.* Wiley, 1976.

[36] Kyle G. Felker, Andrew R. Siegel, Kord S. Smith, Paul K. Romano, and Benoit Forget. The energy band memory server algorithm for parallel Monte Carlo calculations. In *Joint International Conference on Supercomputing in Nuclear Applications and Monte Carlo*, Paris, France, Oct. 27–31 2013.

[37] Anthony Foderaro. *The Elements of Neutron Interaction Theory.* MIT Press, 1971.

[38] Benoit Forget, Paul K. Romano, and Colin Josey. Personal communication.

[39] Benoit Forget, Sheng Xu, and Kord Smith. Direct Doppler broadening in Monte Carlo simulations using the multipole representation. *Ann. Nucl. Energy*, 64:78–85, 2014.

[40] Z.G. Ge, Y.X. Zhuang, T.J. Liu, J.S. Zhang, H.C. Wu, Z.X. Zhao, and H.H. Xia. The updated version of Chinese evaluated nuclear data library (CENDL-3.1). In *Proc. International Conference on Nuclear Data for Science and Technology*, 2010.

[41] E.M. Gelbard. *Epithermal Scattering in VIM.* FRA-TM-123. Argonne National Laboratory, 1979.

[42] S.Z. Ghrayeb, M. Ouisloumen, A.M. Ougouag, and K.N. Ivanov. Deterministic modeling of higher angular moments of resonant neutron scattering. *Annals of Nuclear Energy*, 38:2291–2297, 2011.

[43] T. Goorley, M. James, T. Booth, F. Brown, J. Bull, L.J. Cox, J. Durkee, J. Elson, M. Fensin, R.A. Forster, J. Hendricks, H.G. Hughes, R. Johns, B. Kiedrowski, R. Martz, S. Mashnik, G. McKinney, D. Pelowitz, R. Prael, J. Sweezy, L. Waters, T. Wilcox, and T. Zukaitis. Initial MCNP6 release overview. *Nuclear Technology*, 180:298–315, 2012.

[44] N. Horelik, B. Herman, B. Forget, and K. Smith. Benchmark for evaluation and validation of reactor simulations (BEAVRS). In *Proc. Int. Conf. on Mathematics and Computational Methods Applied to Nuclear Science and Engineering (M&C 2013)*, 2013.

[45] Nicholas Horelik, Benoit Forget, Andrew Siegel, and Kord Smith. Domain decomposition and terabyte tallies with the OpenMC Monte Carlo neutron transport code. In *PHYSOR 2014 - The Role of Reactor Physics toward a Sustainable Future*, Kyoto, Japan, September 28 - October 3 2014.

[46] R.N. Hwang. A rigorous pole representation of multilevel cross sections and its practical applications. *Nuclear Science and Engineering*, 96:192–209, 1987.

[47] D. Lee, K. Smith, and J. Rhodes. The impact of ^{238}U resonance elastic scattering approximations on thermal reactor Doppler reactivity. *Annals of Nuclear Energy*, 36.3:274–280, 2009.

[48] J. Leppanen. Two practical methods for unionized energy grid construction in continuous-energy Monte Carlo neutron transport calculation. *Annals of Nuclear Energy*, 36.7:878–885, 2009.

[49] J. Leppanen. *Serpent a Continuous-energy Monte Carlo Reactor Physics Burnup Calculation Code.* VTT Technical Research Centre of Finland, 2013.

[50] R.E. Macfarlane and D.W. Muir. *The NJOY Nuclear Data Processing System, Version 91.* LA-12740-M. Los Alamos National Laboratory, 1994.

[51] William R. Martin. Challenges and prospects for whole-core Monte Carlo analysis. *Nuclear Engineering and Technology*, 44:151–160, 2012.

[52] Nicholas Metropolis and S. Ulam. The Monte Carlo method. *Journal of the American Statistical Association*, 44:335–341, 1949.

[53] T. Mori and Y. Nagaya. Comparison of resonance elastic scattering models newly implemented in MVP continuous-energy Monte Carlo code. *Journal of Nuclear Science and Technology*, 46.8:793–798, 2009.

[54] R.D. Mosteller. *Computational Benchmarks for the Doppler Reactivity Defect.* LA-UR-06-2968. Los Alamos National Laboratory, 2006.

[55] M. Ouisloumen and R. Sanchez. A model for neutron scattering off heavy isotopes that accounts for thermal agitation effects. *Nuclear Science and Engineering*, 107.3:189–200, 1991.

[56] Joel Rhodes, Kord Smith, and Deokjung Lee. CASMO-5 development and applications. In *PHYSOR-2006*, 2006.

[57] Paul K. Romano, Andrew R. Siegel, Benoit Forget, and Kord Smith. Data decomposition of Monte Carlo particle transport simulations via tally servers. *J. Comput. Phys.*, 252:20–36, 2013.

[58] P.K. Romano and B. Forget. The OpenMC Monte Carlo particle transport code. *Annals of Nuclear Energy*, 51:274–281, 2013.

[59] W. Rothenstein. Neutron scattering kernels in pronounced resonances for stochastic Doppler effect calculations. *Annals of Nuclear Energy*, 23:441–458, 1996.

[60] W. Rothenstein. Proof of the formula for the ideal gas scattering kernel for nuclides with strongly energy dependent scattering cross sections. *Annals of Nuclear Energy*, 31:9–23, 2004.

[61] K. Shibata, O. Iwamoto, T. Nakagawa, N. Iwamoto, A. Ichihara, S. Kunieda, S. Chiba, K. Furutaka, N. Otuka, T. Ohsawa, T. Murata, H. Matsunobu, A. Zukeran, S. Kamada, and J. Katakura. Jendl-4.0: A new library for nuclear science and engineering. *Journal of Nuclear Science and Technology*, 48:1–30, 2011.

[62] Andrew R. Siegel, Kord Smith, Paul K. Romano, Benoit Forget, and Kyle G. Felker. Multi-core performance studies of a Monte Carlo neutron transport code. *International Journal of High Performance Computing Applications*, 28:87–96, 2014.

[63] V. Sobes. *Coupled Differential and Integral Data Analysis for Improved Uncertainty Quantification of the $^{63,65}Cu$ Cross-Section Evaluations.* PhD thesis, Massachusetts Institute of Technology, 2014.

[64] E.E. Sunny, F.B. Brown, B.C. Kiedrowski, and W.R. Martin. Temperature effects of resonance scattering for epithermal neutrons in MCNP. In *Proc. Int. Conf. on the Physics of Reactors: Advances in Reactor Physics (PHYSOR 2012)*, 2012.

[65] T.M. Sutton, F.B. Brown, F.G. Bischoff, D.B. MacMillan, C.L. Ellis, J.T. Ward, C.T. Ballinger, D.J. Kelly, and L. Schindler. *The Physical Models and Statistical Procedures Used in the RACER Monte Carlo Code.* No. KAPL-4840. Knolls Atomic Power Laboratory (KAPL), Niskayuna, NY, 1999.

[66] T.M. Sutton, T.J. Donovan, T.H. Trumbull, P.S. Dobreff, E. Caro, D.P. Griesheimer, L.J. Tyburski, D.C. Carpenter, and H. Joo. The MC21 Monte Carlo code. In *Proc. Int. Conf. on Mathematics & Computations and Supercomputing in Nuc. Applications*, 2007.

[67] Tripoli-4 Project Team, 2008. *Tripoli-4 User Guide.* CEA-R-6169. French Alternative Energies and Atomic Energy Commission, 2008.

[68] X-5 Monte Carlo Team. *MCNP A General Monte Carlo N-Particle Transport Code, Version 5.* LA-UR-03-1987. Los Alamos National Laboratory, 2008.

[69] A. Trkov, M. Herman, and D.A. Brown. *ENDF-6 Formats Manual.* BNL-90365-2009 Rev.2. National Nuclear Data Center, Brookhaven National Laboratory, 2012.

[70] T. Trumbull. *Treatment of Nuclear Data for Transport Problems Containing Detailed Temperature Distributions.* No. LM-05K180. Knolls Atomic Power Laboratory (KAPL), Niskayuna, NY, 2006.

[71] T.H. Trumbull and T.E. Fieno. Effects of applying the Doppler broadened rejection correction method for LEU and MOX pin cell depletion calculations. *Annals of Nuclear Energy*, 62:184–194, 2013.

[72] T. Viitanen and J. Leppanen. Explicit temperature treatment in Monte Carlo neutron tracking routines - first results. In *Proc. Int. Conf. on the Physics of Reactors: Advances in Reactor Physics (PHYSOR 2012)*, 2012.

[73] Jonathan A. Walsh, Benoit Forget, and Kord S. Smith. Accelerated sampling of the free gas resonance elastic scattering kernel. *Ann. Nucl. Energy*, 69:116–124, 2014.

[74] G. Yesilyurt, W.R. Martin, and F.B. Brown. On-the-fly Doppler broadening for Monte Carlo codes. *Nuclear Science and Engineering*, 171:239–257, 2012.

[75] A. Zoia, E. Brun, C. Jouanne, and F. Malvagi. Doppler broadening of neutron elastic scattering kernel in TRIPOLI-4. *Annals of Nuclear Energy*, 54:218–226, 2013.