POLARIS AND TRACE/PARCS CODE DEVELOPMENT FOR CANDU ANALYSIS

DEVELOPMENT AND EVALUATION OF POLARIS CANDU GEOMETRY MODELLING AND OF TRACE_MAC/PARCS_MAC COUPLING WITH RRS FOR CANDU ANALYSIS

By SIMON YOUNAN, M.A.Sc., B.Eng.

A Thesis Submitted to the School of Graduate Studies in Partial Fulfilment of the Requirements for the Degree Doctor of Philosophy

Descriptive Note

McMaster University DOCTOR OF PHILOSOPHY (2022) Hamilton, Ontario (Engineering Physics)

TITLE: Development and Evaluation of Polaris CANDU Geometry Modelling and of TRACE_Mac/PARCS_Mac Coupling with RRS for CANDU Analysis

AUTHOR: Simon Younan, M.A.Sc. (McMaster University), B.Eng. (McMaster University)

SUPERVISOR: Dr. David Novog

NUMBER OF PAGES: xiv, 163

Lay Abstract

Modern nuclear safety analysis tools offer more accurate predictions for the safety and operation of nuclear reactors, including CANDU reactors. These codes take advantage of modern computer hardware, and also a shift in philosophy from conservative analysis to best estimate plus uncertainty analysis. The goal of this thesis was to adapt a number of modern tools to support CANDU analysis and uncertainty propagation, with a particular emphasis on coupling of multiple interacting models. These tools were then demonstrated, and results analyzed.

The simulations performed in this work were successful in producing results comparable to prior studies along with experimental and operational data. This included the simulation of four weeks of reactor operation including "shim mode" operation. Sensitivity and uncertainty analyses were performed over the course of the work to quantify the precision and significance of the results as well as to identify areas of interest for future research.

Abstract

In the field of nuclear safety analysis, as computers have become more powerful, there has been a trend away from low-fidelity models using conservative assumptions, to high-fidelity best-estimate models combined with uncertainty analysis. A number of these tools have been developed in the United States, due to the popularity of light water reactors. These include the SCALE analysis suite developed by ORNL, as well as the PARCS and TRACE tools backed by the USNRC. This work explores adapting the capabilities of these tools to the analysis of CANDU reactors.

The Polaris sequence, introduced in SCALE 6.2, was extended in this work to support CANDU geometries and compared to existing SCALE sequences such as TRITON. Emphasis was placed on the Embedded Self-Shielding Method (ESSM), introduced with Polaris. Both Polaris and ESSM were evaluated and found to perform adequately for CANDU geometries. The accuracy of ESSM was found to improve when the precomputed self-shielding factors were updated using a CANDU representation.

The PARCS diffusion code and the TRACE system thermalhydraulics code were coupled, using the built-in coupling capability between the two codes. In addition, the Exterior Communications Interface (ECI), used for coupling with TRACE, was utilized. A Python interface to the ECI library was developed in this work and used to couple an RRS model written in Python to the coupled PARCS/TRACE model. A number of code modifications were made to accommodate the required coupling and correct code deficiencies, with the modified versions named PARCS_Mac and TRACE_Mac. The coupled codes were able to simulate multiple transients based on prior studies as well as operational events. The code updates performed in this work may be used for many future studies, particularly for uncertainty propagation through a full set of calculations, from the lattice model to a full coupled system model.

Acknowledgements

I would like to thank my supervisor, Dr. David Novog, for supporting me throughout the research process. Thanks to his guidance, I was able to learn a great deal across multiple areas of nuclear safety analysis and work with the tools used in my research. This thesis would not have been possible with Dr. Novog's connections to the nuclear industry. I would also like to thank the rest of my supervisory committee, Dr. Adriaan Buijs and Dr. Ben Rouben, for being a part of my academic journey and providing further guidance throughout the process.

I would like to thank the SCALE development team at ORNL, particularly Matthew Jessee and William Wieselquist, for the development of SCALE and Polaris, as well as for their support and guidance in working with the SCALE code and Polaris in particular. While the development of the Polaris CANDU extension was performed independently, it would not have been possible without their collaboration in suggesting the approach to take with their knowledge of the code.

I would also like to thank Purdue University and the USNRC for the development of the PARCS and TRACE codes for which much of this thesis relied on.

I would like to thank Kai Groves and Michael Tucker for their work using PARCS and TRACE, for which the work in this thesis was built upon. Thanks to their work, and through providing their CANDU models as a starting point for this work, much duplication of effort was avoided, with this work able to focus on the coupling of PARCS and TRACE models and development of a coupled RRS model. The studies using the original externally coupled models were also used as a reference to compare against the results of the natively coupled models. I would additionally like to thank Michael Tucker for his continued work in developing and executing the SCALE/TRITON and Serpent models to compute the macroscopic cross-sections used in the PARCS models, as well as for his execution of simulated core follow models which were used as a starting point for a number of the shim transient sensitivity cases.

I would also like to thank Anatol Mysen and David Hummel for the development of the RD-14M model used as a starting point in this work.

Table of Contents

List of Figures

List of Tables

Notation and Abbreviations

ADF	Assembly Discontinuity Factor
ANM	Analytical Nodal Method
AOO	Anticipated Operational Occurrence
BEPU	Best Estimate Plus Uncertainty
BWR	Boiling Water Reactor
CANDU	Canada Deuterium Uranium
CBC	RRS adjuster rod control module
CBL	RRS liquid zone control module
CBS	RRS control absorber control module
CEP	Power Error Calculation module
CMFD	Coarse Mesh Finite Difference
ECI	Exterior Communications Interface
ESSM	Embedded Self-Shielding Method
FDM	Finite Difference Method
FINCH	Fully Instrumented Channel
FLU	Flux mapping module
FP	Full Power
HEM	Homogeneous Equilibrium Model
HTS	Heat Transport System
ICFD	In-Core Flux Detector
IR	Intermediate Resonance
JSON	JavaScript Object Notation
LOCA	Loss Of Coolant Accident
LWR	Light Water Reactor
MCA	Mechanical Control Absorber
MCP	Power Measurement and Calibration module
MoC	Method of Characteristics
NEM	Nodal Expansion Method
NR	Narrow Resonance
OOP	Object-Oriented Programming
ORNL	Oak Ridge National Laboratory
PARCS	Purdue Advanced Reactor Core Simulator
PMAXS	Purdue Macroscopic Cross-Section data file format
PHWR	Pressurized Heavy Water Reactor
PP	Present Power
PWR	Pressurized Water Reactor
PyECI	Python ECI package
RBA	Shutoff rod extraction module
RIF	Resonance Interference Factor

RRS	Reactor Regulation System
SDS	Shutdown System
SG	Steam Generator
SGGP	Scale General Geometry Package
SNAP	Symbolic Nuclear Analysis Package
SOR	Shutoff Rod
TRACE	TRAC-RELAP Advanced Computational Engine
USNRC	United States Nuclear Regulatory Commission
WR	Wide Resonance

A	Bell factor
c	Dancoff factor
c_p	Specific heat capacity at constant pressure
D	Neutron diffusion coefficient
$DLIFT$	Differential lift (for RRS liquid zone control)
DPZ	Zone power error
E_P	Power error
$ENIV$	Liquid zone level error from average level
h	Heat transfer coefficient
$\boldsymbol{J}$	Neutron current
k	Thermal conductivity
k	Neutron multiplication constant
k_{eff}	Effective neutron multiplication constant
k_∞	Infinite medium neutron multiplication constant
K_T	Liquid zone spatial control gain coefficient
K_H	Liquid zone level control gain coefficient, high
K_L	Liquid zone level control gain coefficient, low
L	Characteristic length
$\bar{\ell}$	Mean chord length
N_i	Number density for nuclide i
Nu	Nusselt number
P_{DLOG}	Demanded logarithmic power
P_{LIN}	Linear power
P_{LOG}	Logarithmic power
q	Neutron source term
R_{DLOG}	Demanded logarithmic power rate
R_{LOG}	Logarithmic power rate
Re	Reynolds number
v	Velocity
α_T	Liquid zone spatial control phase-in coefficient
α_{TL}	Liquid zone spatial control level phasing factor

α_{TP}	Liquid zone spatial control power phasing factor
Δu_g	Lethargy width of energy group g
$\lambda_{g,i}$	Intermediate resonance parameter for energy group g for nuclide i
μ	Dynamic viscosity
ν	Fission neutron multiplicity
ρ	Density
Σ	Macroscopic cross-section
Σ_a	Macroscopic absorption cross-section
Σ_e	Macroscopic escape cross-section
Σ_{esc}	Same as Σ_e
Σ_f	Macroscopic fission cross-section
Σ_{nr}	Macroscopic total narrow resonance cross-section
Σ_s	Macroscopic scattering cross-section
$\Sigma_{s,g'\to g}$	Macroscopic scattering cross-section from energy group g' to energy group g
Σ_t	Macroscopic total cross-section
Σ_{tr}	Macroscopic transport cross-section
Σ_{wr}	Macroscopic total wide resonance cross-section
$\Sigma_{x,g}$	Macroscopic cross-section for reaction x, for energy group g
σ	Microscopic cross-section
σ_0	Microscopic background cross-section
σ_e	Microscopic escape cross-section
σ_p	Microscopic potential scattering cross-section
σ_x	Microscopic cross-section for reaction x
ϕ	Neutron flux
χ	Fission neutron energy distribution
ψ	Directional flux

Declaration of Academic Achievement

The basis of this research is a collaboration between myself and my supervisor, Dr. Novog. His role was to conceptualize the research topics and guide the path of the research throughout the process. This included the decision to adapt both Polaris as well as TRACE and PARCS for CANDU modelling applications, and to analyze shim operation. Dr. Novog was also responsible for obtaining the TRACE and PARCS codes through CAMP.

All code modifications as well as newly developed scripts were my own work. This included modifications to Polaris to implement CANDU geometry along with other minor modifications to SCALE, as well as the modifications to TRACE and PARCS to produce the TRACE_Mac and PARCS_Mac versions, respectively, along with the modifications and additions to the ECI library to add support for Python coupling, as well as the various Python scripts used both to implement coupled models as well as to drive the simulations of long transients and to process the results.

The Polaris model was implemented by me, based on known CANDU lattice data. Execution of test cases and gathering of results was performed by me, with some guidance from Dr. Novog to determine what would be useful to include and how much would be sufficient for a journal paper.

The TRACE and PARCS CANDU-6 models were originally developed by Kai Groves for his thesis in running a benchmark problem. In his work, the models were coupled externally, using a Python script for data transfer. My work involved implementing the native TRACE/PARCS coupling capability into the model, through creating a MAPTAB file. I then re-ran these simulations and compared them to the original work.

The TRACE RD-14M model was originally developed by Anatol Mysen and modified by David Hummel; my contribution was to modify the model to simulate flow oscillation transients, including adding the header interconnects from the experiments to the model.

The TRACE and PARCS 900 MW CANDU models were originally developed by Michael Tucker for his own work, based on conversion of existing RELAP5 models for TRACE. As with the CANDU-6 models, these had been coupled through external scripts. I converted these models to implement the native TRACE/PARCS coupling capability, as well as to couple the RRS script to the model with the ECI. The cases selected to run were based on discussion with Dr. Novog and were based on selecting a set of cases that would evaluate the coupling between the models. I ran these cases and made updates to the model to solve inconsistencies against the prior studies and data used in the comparisons.

The project to model shim operation was conceptualized by Dr. Novog, and station data from a 900 MW class CANDU reactor was obtained. Dr. Novog was responsible for most of the communication with the utility to obtain the necessary data for adapting the model, for setting up initial conditions, and for comparing the results. Multiple group meetings were also held, and it was up to me to determine what data was needed to carry out the work in order to request. The PMAXS macroscopic cross-section files used in this work were generated by Michael Tucker, as were a number of random core snapshots to compare against the core snapshot provided by the station utility. I performed the development of the driver scripts to perform the shim transient, as well as the execution of the shim simulations and collection of results.

This dissertation, including all three journal papers, were drafted by me, with my supervisor reviewing the papers and providing feedback and editing, which I used to revise the papers. The first two journal papers also went through the peer review process, which led to further editing and revision.

The work in Chapter 4 was conducted through 2018-2019. The work in Chapter 5 was conducted from 2020 through mid-2021. The work in Chapter 6 was conducted from mid-2021 through 2022.

1 Introduction

1.1 Background

The CANDU reactor is a Canadian-designed Pressurized Heavy Water Reactor (PHWR). There are several significant differences from light water reactors, including the use of natural uranium fuel and heavy water moderator, the separation of coolant and moderator using pressure tubes, the horizontal orientation of fuel channels, and the capability of on-line refuelling.

As of 2022, the CANDU design makes up the entirety of Canada's power reactor fleet, with 19 reactors in operation or under refurbishment, all but one in Ontario. While six units at Pickering are nearing end-of-life, six units at Bruce and four units at Darlington are undergoing a refurbishment project which will provide at least 30 additional years of life. In addition, several CANDU units (mainly CANDU-6 type reactors) have been exported to other countries, including South Korea, China, India, Argentina, Romania, and Pakistan. Therefore, the CANDU reactor will continue to serve as an important part of the worldwide nuclear fleet for decades to come.

The topic of safety analysis involves identifying the potential hazards with the technology along with evaluating the effectiveness of the design in minimizing these hazards. Deterministic safety analysis evaluates whether the design is effective at providing safety against design basis events, while probabilistic safety analysis evaluates the risk of consequences from severe accidents such as core damage or unacceptable radiation releases.

The Fukushima Daiichi accident in March 2011 had a significant effect on safety analysis. There was an increased emphasis placed on beyond design basis accidents and severe accidents along with the mitigation of radiation releases from such events. The station blackout event, which is the event that occurred at Fukushima Daiichi, is of particular interest. Many safety analyses look at determining the time available for mitigation strategies to be applied before serious consequences occur, and what changes can be applied to the design to increase this time or reduce the consequences.

Over the years, the use of computer modelling for safety analysis has increased. With the increase in available computer power, the fidelity of these models has also increased. These models perform analysis in several different fields of physics. Reactor physics focuses on the behaviour of the neutron chain reaction as well as the characteristics of the fuel with respect to the chain reaction. System thermalhydraulics focuses on the behaviour of the heat transport and steam systems and the transfer of heat between them and the fuel. Other models can be used for the reactor regulating and safety

systems, the mechanical and thermal performance of the fuel, the mechanical performance of other components such as pressure tubes, or the performance of other systems such as the containment building. There are a multitude of feedbacks present between these systems and thus the models are often coupled together. Finally, as the fidelity of analysis methods has improved, there has been a shift from traditional conservative/bounding licensing analysis to methods which attempt to provide a best estimate of the plant response and with quantified uncertainties.

1.2 Motivation

Many advancements have been made in the modelling and simulation of nuclear power stations since the CANDU plants were first designed and built, both in the available computational power as well as in the numerical methods themselves. Modern programs can take advantage of the gigabytes of memory available on even a laptop computer, along with processors which can perform many billions of calculations per second. This allows model developers to select more accurate methods with fewer approximations and constraints. Some general examples include:

- Reactor physics codes using finer discretization of space, angle, and/or neutron energy, along with more accurate scattering equations.
- Core physics codes modelling full nodal spatial kinetics instead of point kinetics or modal analysis.
- Thermalhydraulic models with more accurate two-phase fluid models and support for larger model sizes (in terms of node counts, number of fuel channel simulated, fidelity of bundle heat transfer meshes, etc.).
- Modelling a larger domain within an individual simulation (for example, modelling an entire fuel assembly instead of approximating it using individual pin models).

One area that has evolved over the years is parallel computing. As CPU speeds increased over the years, they eventually hit a ceiling, resulting in the need for a transition from serial computing, where calculations are performed sequentially, to parallel computing, where many calculations are performed simultaneously. In addition, supercomputers take advantage of parallel computing by connecting many individual machines into a "cluster" of networked computers. Therefore, computer algorithms must be adapted in order to take advantage of parallel computing, by either identifying small calculations which can be performed independently and thus in parallel, or by subdividing a large calculation into many smaller calculations. Some models which are well-suited for parallelization include:

- Monte Carlo reactor physics, which performs neutron physics calculations stochastically by calculating the trajectories of many individual random neutrons.

- Stochastic uncertainty analysis, which statistically generates many individual perturbed models which can be computed independently.
- Self-coupling models, which permits a large model to be subdivided into many small models which communicate through boundary conditions representing the physical connections between the models.

In addition to the advantages in the models themselves, modern codes have usability advantages both from a user perspective and from a developer perspective. Legacy codes are typically written in programming languages such as Fortran 77 and have fixed-form input formats. Modern codes are designed around more modern programming languages such as Fortran 90, C, and C++, and around more modern free-format user input paradigms, often with support from GUI-based applications to streamline the process of input deck preparation, such as the ORNL's Fulcrum editor for SCALE, or the SNAP model editor for NRC programs including PARCS and TRACE.

The motivation for this thesis is to contribute towards the advancement of nuclear safety analysis from legacy codes, often containing conservative modelling assumptions, towards modern best-estimate codes and codes supporting uncertainty analysis. For this thesis, there is a particular emphasis on applying these modern codes to the CANDU design, though the methodologies used can be applied to other reactor types as well, including advanced reactors. Many of these tools are developed in the United States and are thus adapted to Light Water Reactors, but much work has been done to broaden the scope of these tools, and this work both contributes to the development of these tools as well as evaluating their suitability for CANDU modelling. Such codes with broad applications benefit from having a larger user base and thus a greater level of support compared to CANDU-specific tools. There is also an emphasis on code coupling, where multiple codes representing different interacting physical models communicate with one another, allowing for feedback effects between the different models to be simulated. Finally, these is an emphasis on uncertainty propagation, where uncertainties in the inputs, such as nuclear data, are applied through a sequence of tools. This can be used, for example, to quantify uncertainties in anticipated operational occurrences (AOOs) due to the uncertainty of nuclear data or other lattice-level properties.

1.2.1 Objectives

The main objective of this work is to investigate the use of and adapt computer codes for broader reactor applications beyond their existing scope, in particular the use and adoption of tools developed in the USA and which have a large user and experience base. This work focuses on the modification and testing of codes for application to the CANDU design specifically, though similar methodologies could be applied to extend the scope to

other types of reactors. The results of the work are validated against existing analyses, experiments, and data.

The novel contributions of this thesis are:

1. The development of software capability for modelling CANDU geometries in the Polaris module of the SCALE code system. The Polaris code is a state-of-the-art deterministic lattice physics code which enables the rapid calculation of lattice properties and their uncertainties (through linkages with the Sampler module). These modifications extend the capability of Polaris so that it can be applied for CANDU reactor analysis and has been submitted to ORNL for eventual inclusion into a future release of SCALE.
2. Modification of the TRACE-PARCS code package, developed by the USNRC, for analysis of important CANDU accident transients and inclusion of a Reactor Regulating System (RRS) model by modification of the ECI routines in TRACE and development of new emulator Python codes. Such capability did not exist within the academic community and greatly extends the potential applications of TRACE-PARCS for CANDU analysis.
3. Evaluation of the suitability of novel codes designed for LWR analysis to be adapted to CANDU analysis. Several applications are also demonstrated by this work, and it opens up the capability to investigate additional research applications. Of note in the simulation of a CANDU event and comparison with station measured parameters. This event simulation involves the complex interactions between reactor physics and thermalhydraulics with the coupled RRS model described above and takes place over the course of several operating days. The simulation of this complex coupling and RRS response over such a long period of operation represents one of the most significant modelling challenges possible. An understanding of the benefits and limitations of the codes investigated in this work can be used to make decisions on their use in future work.
4. The application of these codes to demonstrate uncertainty propagation, where the effect of nuclear data uncertainty on the outcome of an AOO is determine. Such uncertainty propagation capabilities are vital for best-estimate-plus-uncertainty (BEPU) analysis.

1.3 Outline

This work is presented as a sandwich thesis, consisting of seven major chapters. Chapter 2 presents a literature review in several parts. The literature review includes a theoretical overview of the types of analyses performed in this work, how these analyses are implemented in the codes used in the work, as well as on how these apply to the CANDU reactor design. Chapter 3 presents the methodology of the new work performed

in this thesis, including code modifications, analyses, and validation. These go through additional detail not presented in the later chapters making up the three journal papers of the sandwich thesis.

Chapter 4 presents the first journal paper "Extension and Preliminary Validation of the Polaris Lattice Physics Code for CANDU Analysis", published in *Nuclear Engineering & Design* in 2020. This work focuses on Polaris, a novel Method of Characteristics solver for LWR lattice physics introduced to the SCALE package. This code was of interest due to its faster performance and streamlined input format compared to NEWT, an older lattice physics solver in the SCALE package. The code was extended to allow the modelling of CANDU lattice geometries, followed by performing code-to-code comparisons to evaluate the suitability the code to analyze these geometries.

Chapter 5 presents the second journal paper "Development and Testing of TRACE/PARCS ECI Capability for Modelling CANDU Reactors with Reactor Regulating System Response", published in *Science and Technology of Nuclear Installations* in 2022. This work looks into the coupling of the core physics code PARCS and the system thermalhydraulics code TRACE for a CANDU application. While these two codes have a built-in coupling capability, prior CANDU studies with these codes generally performed external code coupling. In this work, the built-in coupling capability is used. In addition, the ECI library provided with TRACE is used to couple external scripts to the TRACE simulation, including a model representing the CANDU RRS. Several minor changes were made to the codes in order to support proper coupling of the models (with the modified versions being named TRACE_Mac1.0 and PARCS_Mac1.0 to differentiate them from their unmodified counterparts), and simulations were performed to compare both qualitatively and quantitatively against prior simulations, experimental results, and station data.

Chapter 6 presents the third and final journal paper "Modelling and Validation of CANDU Shim Operation Using Coupled TRACE/PARCS with Regulating System Response", submitted to *Science and Technology of Nuclear Installations* in 2022. This work expands upon the work from the second journal paper, by updating the codes used in the prior work as well as creating a driver script to simulate long periods of reactor operation. The driver script executes a sequence of steady-state, depletion, and transient simulations, and transient depletion capability was also added to PARCS. The RRS model was updated to simulate the use of fully instrumented channels (FINCHes) for power calibration. The driver script was used to execute a simulation of shim operation based on a dataset from shim operation at a CANDU 900 reactor. A number of sensitivity cases were executed to evaluate the effect of adjuster rod depletion over the lifetime of the core, adjuster and liquid zone reactivity worth, nuclear data uncertainty, and variation of core snapshots.

Chapter 7 presents the major conclusions of this thesis along with potential development and analyses which could be performed in the future.

2 Literature Review

2.1 Reactor Physics

Reactor physics, or neutronics, is the field of nuclear physics that studies the behaviour of the neutron chain reaction. More specifically, it involves solving the neutron transport equation, either time-dependent or time-independent. The time-independent form is given in Equation (2.1), and is summarized by Equation (2.2).

$$\begin{aligned}&\hat{\boldsymbol{\Omega}}\cdot\boldsymbol{\nabla}\psi\left(\boldsymbol{r},E,\hat{\boldsymbol{\Omega}}\right)+\Sigma_t(\boldsymbol{r},E)\psi\left(\boldsymbol{r},E,\hat{\boldsymbol{\Omega}}\right)\\&=\int_{4\pi}d\hat{\boldsymbol{\Omega}}'\int_0^\infty dE'\Sigma_s\left(\boldsymbol{r},E'\rightarrow E,\hat{\boldsymbol{\Omega}}'\rightarrow\hat{\boldsymbol{\Omega}}\right)\psi\left(\boldsymbol{r},E',\hat{\boldsymbol{\Omega}}'\right)\\&+\frac{1}{k}\frac{\chi(\boldsymbol{r},E)}{4\pi}\int_0^\infty dE'\nu(\boldsymbol{r},E')\Sigma_f(\boldsymbol{r},E)\int_{4\pi}d\hat{\boldsymbol{\Omega}}'\psi\left(\boldsymbol{r},E',\hat{\boldsymbol{\Omega}}'\right)\end{aligned}\tag{2.1}$$

$$Leakage+Absorption+Outscatter=Inscatter+Fission\ Source\tag{2.2}$$

While the physics is well-understood, there are several limitations that apply to reactor physics modelling:

1. The nuclear data, which describes how neutrons and other particles interact with matter, is based upon a combination of theory and experimental data; as the experimental data has uncertainties, so does the evaluated nuclear data.
2. Core geometry is complex, with over 100 thousand fuel pins in a typical core, hundreds of pressure tubes, along with reactivity devices and structural materials, with temperature and density variations throughout the core and within each fuel pin, and variations in depletion between fuel pins and even within fuel pins.
3. The neutron flux is a function of not only position, but also direction and energy. Neutron cross-sections are a function of position and energy, with fine structure with respect to energy due to resonance captures.

The first limitation means that even an exact model will not match reality. However, the uncertainty in the nuclear data is usually quantified, and thus statistical methods can be applied to propagate this uncertainty to the results of the model. These methods can be either deterministic or stochastic. Deterministic methods evaluate the sensitivity of the results to the nuclear data to propagate the uncertainty. Stochastic methods sample the nuclear data and simulate the model multiple times to estimate the uncertainty.

The other two limitations make an exact model computationally intractable. Therefore, reactor physics modelling is broken up into multiple steps:

1. The microscopic and macroscopic nuclear cross-sections are discretized into a many-group structure, usually consisting of between several dozen to several

hundred energy groups. Due to the resonance structure of the nuclear cross-sections, a resonance self-shielding effect exists, therefore, the discretized cross-sections depend on not only the material, but also the problem-specific geometry.
2. An infinitely repeating lattice is modelled using a discretized form of the transport equation (2.1). The flux distribution is solved for this lattice for different conditions, including burnup, fuel temperature, coolant temperature and density, moderator temperature and density, moderator poison concentration, and the presence or absence of reactivity devices.
3. Using a flux-weighted averaging scheme, the macroscopic cross-sections of an individual cell are collapsed into a few-group structure (usually two energy groups) and homogenized into a single material which is equivalent to the infinite lattice.
4. The homogenized lattice properties are used to construct a full-core model for which the flux is solved using the diffusion equation.

For the first step, the nuclear data typically requires tens or hundreds of thousands of points in the energy domain to adequately model the fine structure, including the resolved resonances. However, a lattice physics calculation can practically only proceed with hundreds of energy groups at most. Therefore, the fine structure must be collapsed. To preserve reaction rates, a flux-weighted collapse is performed. However, this requires the fine structure of the flux to be known. The flux in this case depends on the resonance self-shielding effect. Within a fuel lump, the flux for energies within a resonance is depressed compared to the flux for energies between resonances, as the resonance-energy neutrons react with the resonance. This effect is dependent on the problem-specific geometry and materials.

There are several methods which can be used for resonance self-shielding [1]:

1. Calculate the flux directly using an ultrafine or continuous-energy calculation, then collapse the cross-sections into the multi-group structure. This method is accurate, but slow and practical only on simple geometries.
2. Equivalence theory methods, where the self-shielded cross-sections are treated as a function of the temperature as well as a “background cross-section” based on the dilution of the resonant nuclide. These methods capture the spatial self-shielding through an “escape cross-section” which is added to the homogeneous background cross-section. These methods are fast but make many simplifications and cannot account for effects such as spatial dependence or resonance interference.
3. Subgroup methods, which are based on the equivalence theory, but also subdivide each energy group not by energy, but by resonance cross-section, taking advantage of the fact that the flux tends to be inversely proportional to the total

cross-section. This improves the accuracy over the standard equivalence theory, though still cannot account for certain phenomena such as resonance interference.

In the SCALE code package, the two main self-shielding codes used throughout the package are BONAMI (a standard equivalence theory method) and CENTRM (an ultrafine method) [2]. The Polaris lattice physics code introduces a new third method to SCALE, the Embedded Self-Shielding Method (ESSM), which is a modern equivalence theory method which calculates escape cross-sections through solving a set of one-group fixed-source transport equations, which improves the handling of complex geometry over the standard equivalence theory [3]. Meanwhile, the subgroup method is included in codes such as WIMS, HELIOS [1], and DRAGON [4].

Once the multi-group cross-sections are calculated, a lattice physics transport calculation can be performed over an entire assembly or fuel channel. Equation (2.1) is discretized in energy to yield Equation (2.3). Further discretization of this equation must then be performed in space and angle.

$$\begin{aligned}&\widehat{\boldsymbol{\Omega}}\cdot\boldsymbol{\nabla}\psi_g\left(\boldsymbol{r},\widehat{\boldsymbol{\Omega}}\right)+\Sigma_{t,g}(\boldsymbol{r})\psi_g\left(\boldsymbol{r},\widehat{\boldsymbol{\Omega}}\right)\\&=\int_{4\pi}d\widehat{\boldsymbol{\Omega}}'\sum_{\substack{g'=1\\g'\neq g}}^{G}\Sigma_{s,g'\to g}\left(\boldsymbol{r},\widehat{\boldsymbol{\Omega}}'\to\widehat{\boldsymbol{\Omega}}\right)\psi_g\left(\boldsymbol{r},\widehat{\boldsymbol{\Omega}}'\right)\\&+\frac{1}{k}\frac{\chi_g(\boldsymbol{r})}{4\pi}\sum_{\substack{g'=1\\g'\neq g}}^{G}\nu_g(\boldsymbol{r})\Sigma_{f,g'}(\boldsymbol{r})\int_{4\pi}d\widehat{\boldsymbol{\Omega}}'\psi_{g'}\left(\boldsymbol{r},\widehat{\boldsymbol{\Omega}}'\right)\end{aligned}\tag{2.3}$$

For deterministic solvers, this is done using the Method of Characteristics (MoC). This involves converting the transport term $\widehat{\boldsymbol{\Omega}}\cdot\boldsymbol{\nabla}\psi_g\left(\boldsymbol{r},\widehat{\boldsymbol{\Omega}}\right)$ into a derivative along the neutron's path of motion, $\frac{d\psi_g\left(r,\widehat{\boldsymbol{\Omega}}\right)}{ds}$ [1]. The angular term $\widehat{\boldsymbol{\Omega}}$ is then discretized into a quadrature set, while the source terms, on the right hand-side of the equation, along with the cross-sections, are spatially discretized into a mesh. This results in Equation (2.4), which is solved over a set of characteristic rays to get Equation (2.5) [1].

$$\frac{d\phi}{ds}+\Sigma_{tr}\phi=\frac{q}{4\pi}\tag{2.4}$$

$$\phi(s)=\phi(0)e^{-\Sigma_{tr}s}+\frac{q}{4\pi\Sigma_{tr}}(1-e^{-\Sigma_{tr}s})\tag{2.5}$$

How these characteristic rays are constructed varies between codes. In addition to assuming a flat source within a mesh cell, scattering and source terms are often approximated as isotropic (using transport corrected scattering) [1]. The transport code then solves for the flux along each characteristic at the mesh boundary points, along with the source terms in each mesh cell. This is a scheme which iterates until convergence. MoC on its own tends to converge very slowly; some acceleration schemes include coarse mesh rebalance and coarse mesh finite difference (CMFD) [1].

As the typical goal of these lattice physics calculations is to compute homogenized cross-sections for diffusion codes, the transport equation must be solved over the entire domain of possible lattice conditions. Starting with fresh fuel under representative conditions, the computed flux is used to simulation depletion of the fuel for a certain time step or burnup step, with the flux being recomputed at each burnup step. In addition, at each burnup point, the transport equation is solved for many different "branch points" which span variations in many properties, including fuel temperature, coolant/moderator temperature, coolant/moderator density, moderator poison concentration, and insertion or removal of reactivity devices. A more detailed calculation may also perform the depletion under different conditions to treat "history variables", in other words, how the lattice properties vary based on the history of the lattice.

At each branch point, the flux from the transport solution is used to collapse the macroscopic cross-sections down to a few-group structure (typically two groups, one fast and one thermal) and homogenize the lattice cell into a single material representative of the entire lattice cell. Depending on the codes being used, the branch cross-sections may be saved as partial derivatives relative to the base cross-sections from the representative state. For example, this is done by GenPMAXS when converting macroscopic cross-sections into the PARCS PMAXS format [5].

With these cross-sections, the full core can be modelled using a nodal diffusion code. The angular dependence is first eliminated by integrating over all angles. However, this leaves a neutron current term in the expression for the leakage, which is the first term in Equation (2.6) [1].

$$\boldsymbol{\nabla} \cdot \boldsymbol{J_g}(\boldsymbol{r}) + \Sigma_{t,g}(\boldsymbol{r})\phi_g(\boldsymbol{r}) = \sum_{\substack{g'=1 \\ g' \neq g}}^{G} \Sigma_{s,g' \to g}(\boldsymbol{r})\phi_{g'}(\boldsymbol{r}) + \frac{\chi_{\mathrm{g}}(\boldsymbol{r})}{k} \sum_{g'=1}^{G} \nu\Sigma_{f,g'}(\boldsymbol{r})\phi_{g'}(\boldsymbol{r}) \quad (2.6)$$

This term is eliminated by applying Fick's Law (Equation (2.7)), which is reasonable on the scale of the full core when using homogenized lattice cells. This results in the neutron diffusion equation (2.8) [1].

$$\boldsymbol{J_g}(\boldsymbol{r}) = -\mathrm{D}_g(\boldsymbol{r})\nabla\phi_g(\boldsymbol{r}) \tag{2.7}$$

$$-\boldsymbol{\nabla}\cdot \mathrm{D}_g(\boldsymbol{r})\nabla\phi_g(\boldsymbol{r}) + \Sigma_{t,g}(\boldsymbol{r})\phi_g(\boldsymbol{r}) = \sum_{\substack{g'=1 \\ g'\neq g}}^{G} \Sigma_{s,g'\to g}(\boldsymbol{r})\phi_{g'}(\boldsymbol{r}) + \frac{\chi_{\mathrm{g}}(\boldsymbol{r})}{k}\sum_{g'=1}^{G} \nu\Sigma_{f,g'}(\boldsymbol{r})\phi_{g'}(\boldsymbol{r}) \tag{2.8}$$

This can be solved by several numerical techniques, including simple numerical schemes such as the finite difference method (FDM) [1]. The flux in the entire core can then be used to calculate other properties, such as the power in each node. The eigenvalue problem (2.8) can only be solved to a constant factor. In other words, only the spatial and energy distribution of the flux, not its magnitude, can be solved. The magnitude of the flux is then solved by providing an additional constraint, usually the total reactor power.

An alternative methodology for performing reactor physics calculations is to perform a Monte Carlo calculation. Other calculation methods are deterministic, as they numerically solve an approximation of the neutron transport equation. The Monte Carlo method is a stochastic method which solves the neutron transport equation by simulating the underlying phenomenon, simulating the paths of many individual neutrons. As particle interactions are a random process, each simulated neutron path is also random. Statistically, this process solves the transport equation, with the uncertainty depending on the number of simulated neutron paths as well as the quantities which are being solved.

The primary advantage of the Monte Carlo method is that the approximations normally made in deterministic calculations to achieve acceptable performance are no longer required. Discretization in angle can be eliminated. More importantly, discretization in energy can be eliminated, allowing the calculation to be performed in continuous-energy and eliminating the need for a resonance self-shielding calculation. In theory, spatial discretization can be eliminated or partially eliminated as well. In addition, a Monte Carlo calculation can be scaled up to the full 3D core model, solving the transport equation over the full core without a lattice transport step.

The primary disadvantage of the Monte Carlo method is that, while more accurate, it is less precise. As the uncertainty is statistical, the standard deviation of any result is inversely proportional to the square root of the number of neutron histories. The number of histories required for acceptable convergence varies greatly by the results being computed. Typically, more histories are required if local quantities are required, such as

pin powers, or if branch calculations are being performed and thus the difference between two similar lattices is being computed.

2.2 Self-Shielding Calculations

As described previously, it is necessary to perform a resonance self-shielding calculation before performing a multi-group transport calculation. To discretize the continuous-energy cross-sections to the multi-group structure, a flux-weighted averaging is performed. When the flux and cross-sections are slowly varying with respect to energy, this can be approximated well by using a typical flux distribution for the reactor type. For thermal reactors, the flux spectrum can be approximated as Maxwellian at thermal energies, a fission energy spectrum at fission energies, and a 1/E relation at intermediate energies [6].

However, the flux and cross-sections are not slowly varying in the vicinity of a resonance. Neutrons with energies matching a resonance are far more likely to react, corresponding to a large cross-section. As neutrons stream into a fuel lump, neutrons in the resonance will be absorbed more strongly than neutrons outside the resonance, leading to a flux depression in the fuel. The flux-weighted average cross-section correspondingly decreases, as shown in Figure 2.1. The strength of this effect depends on the composition of the fuel and the configuration of the fuel and moderator. Therefore, the self-shielded cross-sections must be computed for each lattice configuration being analyzed.

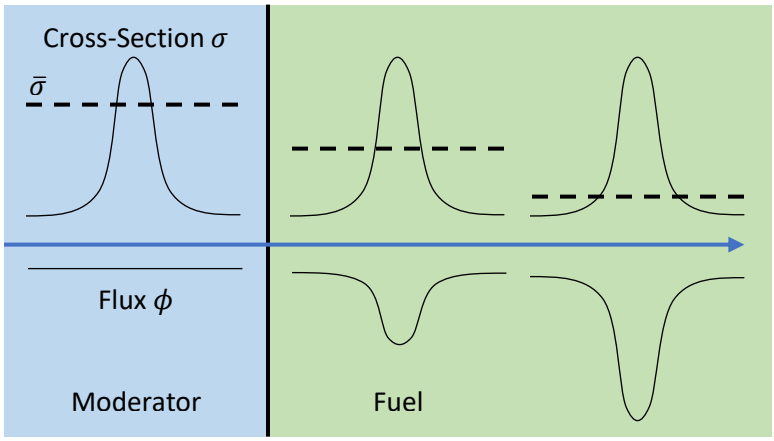

Figure 2.1: Resonance self-shielding effect showing fuel cross-section.

The chosen methods for self-shielding calculations can have a significant effect on the resulting transport solution. If the self-shielding calculation is performed inaccurately, then the amount of resonance capture in the fuel can be overestimated or underestimated, resulting in inaccurate results for k_{eff}, homogenized cross-sections, reactivity coefficients, and depletion.

While ultrafine-group methods provide the most accurate solution to the self-shielding problem, they are too computationally expensive to be practical when many computations are required, such as for branch calculations and especially when performing stochastic uncertainty analysis.

The general principle behind faster-performing self-shielding calculation methods is to, instead of performing a slow computation for each specific problem, perform the slow computations *a priori* and store the results in terms of a correlation that can be used by a faster computation. The two parameters which correlate strongly with the self-shielding effect are the nuclide's Doppler temperature and the "background cross-section".

The effect of temperature is to widen the resonances due to the Doppler broadening effect. The higher the temperature, the faster the fuel atoms move and vibrate. This broadens the distribution of relative velocities and thus neutron kinetic energies from the nuclei's frames of reference. Therefore, in the lab frame of reference, the range of neutron kinetic energies which can interact with a resonance is also broadened. However, this also reduces the peak cross-section of the resonance, reducing the amount of flux depression and therefore weakening the resonance self-shielding effect. Thus, resonance interactions tend to increase with temperature.

The background cross-section is a property of the material composition. The background cross-section is mathematically defined by taking the non-resonant nuclides and lumping them into the resonant nuclide. The contribution to the total microscopic cross-section of the non-resonant nuclides is the background cross-section, as described by Equation (2.9).

$$\sigma_{0,i} = \frac{1}{N_i} \sum_{j \neq i} N_j \sigma_j \tag{2.9}$$

The background cross-section essentially describes how "dilute" the resonance is. The more dilute the resonant nuclide, the smaller the value of N_i, and therefore the larger the value of $\sigma_{0,i}$, going to infinity in the limit of N_i going to zero, i.e., "infinite dilution". The more dilute the resonance, the weaker the flux depression effect, and thus the weaker the resonance self-shielding effect.

More specifically, one can assume that the source term for neutrons around a given resonance comes from scattering from higher energies well outside the resonance, as the resonances are narrow relative to the energy lost in elastic scattering. This is the narrow resonance (NR) approximation [6]. As a result, the source term does not carry any structure from the resonance and can be approximated using a smooth function $C(E)$, usually 1/E at intermediate energies. For a homogeneous material, the resulting flux

profile is described by Equation (2.10). Usually, the background cross-section is only weakly dependent on energy and is approximated as constant around the resonance.

$$\phi(E) \propto \frac{C(E)}{\sigma_0 + \sigma_t(E)} \tag{2.10}$$

The two extreme cases are as follows. For infinite dilution, $\sigma_0 \to \infty$, whereas for no dilution, $\sigma_0 \to 0$. The result is the following flux profiles described by Equations (2.11).

$$\lim_{\sigma_0 \to \infty} \phi(E) \propto C(E), \qquad \lim_{\sigma_0 \to 0} \phi(E) \propto \frac{C(E)}{\sigma_t(E)} \tag{2.11}$$

In the infinitely dilute case, the flux used for weighting the self-shielded cross-sections is independent of the resonance cross-section, while in the zero-dilution case, the flux used for weighting the self-shielded cross-sections is inversely proportional to the resonance cross-section, so the peak of the resonance is weighted less. For intermediate levels of dilution, the weight of the resonance peak takes on an intermediate value. This is the Bondarenko method for resonance self-shielding [6].

For a heterogeneous structure, additional extensions are required. A portion of the neutrons interacting with the fuel will have come from outside the fuel, and, conversely, a portion of the neutrons in the fuel at the resonance energy in the fuel will escape and therefore not interact with the resonance. There is an equivalence theory which can be applied where the effect of the moderator can be described by an "escape cross-section" σ_e equivalent to the background cross-section for homogeneous materials. The two cross-sections can be added to get an effective background cross-section [6].

At lower energies, the narrow-resonance approximation is inaccurate. At low energies, the energy lost from fuel scattering can be assumed to be much less than the resonance width – the wide-resonance (WR) approximation [6]. Equation (2.10) is thus replaced by Equation (2.12) (with absorption cross-section rather than total cross-section).

$$\phi(E) \propto \frac{C(E)}{\sigma_0 + \sigma_a(E)} \tag{2.12}$$

In between these two approximations, it is possible to interpolate, therefore deriving the intermediate resonance (IR) approximation, Equation (2.13) [6].

$$\phi(E) \propto \frac{C(E)}{\sigma_{Pg,i} + \sigma_{ag,i}} = \frac{C(E)}{\frac{1}{N_i}\left(\Sigma_e + \sum_j N_j \lambda_{g,j} \sigma_{pg,j}\right) + \sigma_{ag,i}} \tag{2.13}$$

Σ_e is the escape cross-section while $\sum_j N_j \lambda_{g,j} \sigma_{pg,j}$ is the potential scattering contribution of all nuclides, including the resonant nuclide. $\lambda_{g,j}$ is the IR parameter, equal to 0 for the WR approximation and 1 for the NR approximation. This is the approach used by WIMS [6]. Self-shielded cross-sections or resonance integrals can then then be tabulated in terms of temperature and $\sigma_{Pg,i}$, and any problem can be self-shielded by simply computing $\sigma_{Pg,i}$ and interpolating the self-shielding tables. If these are tabulated using very accurate models of a specific reactor class (for example, a light water reactor enriched UO_2 pin surrounded by water), quick and accurate self-shielding can be performed on any similar model.

The SCALE code package, since version 6.2 [2], has a similar capability using the BONAMI module. Previous versions of SCALE only supported the NR approximation, while SCALE 6.2 added a full IR treatment to achieve acceptable accuracy over the full range of neutron energies. In SCALE, the self-shielding factors are tabulated with respect to temperature and background cross-section. BONAMI uses the formula for the escape cross-section given in Equation (2.14), where $\bar{\ell}$ is the mean chord length, A is the Bell factor, and c is the Dancoff factor [2].

$$\Sigma_e = \frac{(1-c)A}{\bar{\ell}(1+(A-1)c)} \tag{2.14}$$

The mean chord length accounts for the geometry of the fuel lump itself, while the Dancoff factor accounts for the geometry of the lattice. The Dancoff factor is simply the probability that an escaping neutron will stream into another fuel lump without interacting with the moderator. An increase in pin separation or in moderator density will decrease the Dancoff factor. The Bell factor is a correction term used to improve the accuracy of the calculation, and is a function of the geometry, the Dancoff factor, and the product $\Sigma_t \bar{\ell}$.

The Dancoff factor can be computed automatically for regular geometries such as infinite pin lattices. For irregular geometries, the Dancoff factor is typically supplied by the user, and can be computed stochastically using a Monte Carlo code such as MCDancoff [2].

The Embedded Self-Shielding Method (ESSM) is a relatively new method which is based on the IR extension of the Bondarenko method. This starts with the transport equation (2.3) and replaces the source term (right-hand-side) with the source term for neutron slowing down using the IR approximation to get Equation (2.15) [3].

$$\hat{\boldsymbol{\Omega}} \cdot \boldsymbol{\nabla} \psi_g(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}) + \Sigma_{t,g}(\boldsymbol{r}) \psi_g(\boldsymbol{r}, \hat{\boldsymbol{\Omega}}) = \frac{1}{4\pi} \Sigma_{wr,g}(\boldsymbol{r}) \phi_g(\boldsymbol{r}) + \frac{1}{4\pi} \Sigma_{nr,g}(\boldsymbol{r}) \Delta u_g \tag{2.15}$$

The term Δu_g is the lethargy width of the energy group, equal to the integral of $\frac{1}{E}dE$. The $\Sigma_{wr,g}(\boldsymbol{r})$ and $\Sigma_{nr,g}(\boldsymbol{r})$ terms are the portions of the potential scattering source from wide-resonance and narrow-resonance scattering, respectively, defined as follows:

$$\Sigma_{wr,g}(\boldsymbol{r}) = \sum_i N_i\left(1-\lambda_{g,i}\right)\sigma_{pg,i}\,, \qquad \Sigma_{nr,g}(\boldsymbol{r}) = \sum_i N_i\lambda_{g,i}\sigma_{pg,i} \tag{2.16}$$

For a homogeneous mixture, the $\widehat{\boldsymbol{\Omega}}\cdot\boldsymbol{\nabla}\psi_g\left(\boldsymbol{r},\widehat{\boldsymbol{\Omega}}\right)$ becomes zero. Note that in Equation (2.13), the formulation of the escape cross-section treats it as an additional scattering source which obeys the NR approximation. This can also be applied to Equation (2.15), replacing $\widehat{\boldsymbol{\Omega}}\cdot\boldsymbol{\nabla}\psi_g\left(\boldsymbol{r},\widehat{\boldsymbol{\Omega}}\right)$ with a term containing the escape cross-section and adding the escape cross-section to the NR source term:

$$\left(\Sigma_{t,g}(\boldsymbol{r})+\Sigma_{esc,g}(\boldsymbol{r})\right)\phi_g(\boldsymbol{r}) = \Sigma_{wr,g}(\boldsymbol{r})\phi_g(\boldsymbol{r},E) + \left(\Sigma_{nr,g}(\boldsymbol{r})+\Sigma_{esc,g}(\boldsymbol{r})\right)\Delta u_g \tag{2.17}$$

Equation (2.17) is the ESSM formulation of the equivalence theory. Equation (2.15) is a single-group fixed-source transport equation which can be solved for $\phi_g(\boldsymbol{r})$, while Equation (2.17) is an algebraic equation which can be rearranged to get Equation (2.18) to solve for $\Sigma_{esc,g}(\boldsymbol{r})$ [3].

$$\Sigma_{esc,g}(\boldsymbol{r}) = \frac{\Sigma_{a,g}(\boldsymbol{r})\phi_g(\boldsymbol{r})}{\Delta u_g - \phi_g(\boldsymbol{r})} - \Sigma_{nr,g}(\boldsymbol{r}) \tag{2.18}$$

This provides a method to solve for $\Sigma_{esc,g}(\boldsymbol{r})$ without the need of geometry-specific correlations such as Equation (2.14). The slowing-down equation (2.15) is quick to solve using whichever transport solver is available – Polaris uses the same MoC solver it uses for the eigenvalue transport solver. Since there is only one energy group and the source is fixed, the solution is extremely quick. The flux is then plugged into Equation (2.18) to solve for the escape cross-section for each mesh cell. This is done for each energy group, with the potential to parallelize the calculations as each energy group is independent.

There is one caveat, which is that $\Sigma_{a,g}(\boldsymbol{r})$ is dependent on $\Sigma_{esc,g}(\boldsymbol{r})$ as $\Sigma_{a,g}(\boldsymbol{r})$ should be self-shielded. Therefore, the method is applied iteratively. Background cross-sections for each nuclide in each mesh cell are computed from the nuclide-independent $\Sigma_{esc,g}(\boldsymbol{r})$, and the self-shielding tables are interpolated to obtain self-shielded $\sigma_{a,g}(\boldsymbol{r})$ and therefore $\Sigma_{a,g}(\boldsymbol{r})$. The computation of $\phi_g(\boldsymbol{r})$ and $\Sigma_{esc,g}(\boldsymbol{r})$ is then repeated. This is iterated until $\Sigma_{esc,g}(\boldsymbol{r})$ converges.

There are several limitations in this methodology. While fast, it only approximates the self-shielding effect rather than computing an exact flux for self-shielding. There is also

no way to directly treat resonance interference. Resonance interference occurs when there is an overlap between resonances of different nuclides, which self-shield one another. The equivalence theory methods do not model resonance interactions. The resonance structure of a given nuclide is encoded in its self-shielding tables, but the background cross-section calculation completely ignores the absorption from other nuclides. In the ESSM, there is some dependency as $\phi_g(\boldsymbol{r})$ depends on $\Sigma_{a,g}(\boldsymbol{r})$, while the escape cross-section also directly depends on $\Sigma_{a,g}(\boldsymbol{r})$. Treating the additional absorbers increases $\Sigma_{a,g}(\boldsymbol{r})$ but decreases $\phi_g(\boldsymbol{r})$. However, this cannot differentiate between resonances which overlap and resonances which merely share the same energy group. These inaccuracies can be mitigated but not completely eliminated by constructing self-shielding tables from configurations similar to the problems that will be self-shielded.

The third type of method, the subgroup method, lies in between the equivalence theory methods and the ultrafine group methods. In a subgroup method, each energy group to be used in the lattice calculation is subdivided into several subgroups. However, these subgroups are not a function of the neutron energy, but instead of the resonance cross-section. This selection is performed as the neutron flux is primarily related to the resonance cross-section. Therefore, by dividing the energy group based on cross-section, the flux and cross-section variations within each subgroup are reduced, and the relationship between self-shielded cross-sections and background cross-sections is therefore weaker than in the standard equivalence theory method, lessening the impact of approximations used to calculate the background cross-sections [1].

The subgroup method calculates the spatial flux distribution using a transport calculation, similarly to the ESSM. The relative flux between the subgroups can be calculated using the NR approximation, producing an effective self-shielded cross-section for the entire group. The self-shielding tables can then be used to determine the corresponding background cross-section, which is used to update the subgroup cross-sections and iterate the transport calculation until convergence [1].

However, there is another way to apply the subgroup method, which is using probability tables. Instead of treating the subgroups explicitly, the cross-section within each group is treated as a probability distribution. The neutron spectrum can then be integrated as a function of cross-section rather than neutron energy. This approach eliminates the dependence of the subgroup cross-sections on the background cross-section, but also eliminates any physical information on the energy structure of the subgroups [1]. The probability distribution is then discretized into a probability table. The probability tables, however, can be optimized to best agree with ultrafine group methods while using subgroup cross-sections which are independent of the background cross-section.

While the subgroup methods are an improvement in accuracy over the equivalence theory approaches, they still do not account for nonuniform temperature distributions or resonance interference effects.

Properly capturing the resonance interference effect requires additional treatment. This can involve a hybrid calculation (e.g., ultrafine pin cell calculation plus Dancoff correction) or a resonance interference factor (RIF) table, which tabulates ratios of self-shielded cross sections with versus without resonance interference, as a function of the number density of the interfering nuclide. Typically, these tables would be computed for interference with ^{238}U, as this nuclide dominates the resonance interference effect [1]. Another approach is to calculate resonance interference factors on the fly for the actual composition, using either ultrafine calculations or the NR or IR approximations. This, however, can add significant computational requirements [1].

2.3 SCALE/Polaris Code

Polaris is a relatively recent development by Oak Ridge National Laboratories (ORNL). It is provided as an alternative to their existing codes for lattice physics modelling. The greatest distinguishing feature of Polaris is that it does not utilize the geometry and materials input, or the cross-section processing used by most of their other codes.

Previous SCALE-based codes utilized the Scale General Geometry Package (SGGP), which is based on inputting individual surfaces, such as planes, spheres, cylinders, or generalized quadratic surfaces, with the ability to specify units, arrays, and holes to aid in specification of repeating geometry. For example, each unique fuel pin type may be a unit, which can then be placed in a regular lattice using an array or placed in an irregular lattice (such as a CANDU bundle) using holes. However, this can lead to somewhat lengthy specifications, and certain features such as radial and axial meshing for the NEWT transport solver require specification of the individual surfaces. Pins of different compositions must also be specified separately, including pins which initially start with the same composition but are to be depleted separately.

Comparatively, Polaris's geometry input is much shorter, as it only requires the parameters needed to describe a lattice, such as pin geometries, pin pitch, the pin arrangement in the array, along with meshing parameters. Material subdivision for depletion can also be automated. It is therefore far easier for the analyst to specify the geometry. However, this comes at the cost of generalization. The current version of Polaris can only model PWR and BWR geometries. Arbitrary geometries are not supported, nor are CANDU bundle features. However, this limitation is only due to the input specification as the transport solution methodology has no limitation in this regard, and other geometries shall be added to the input specification in the future.

The cross-section processing for most SCALE codes besides Polaris uses XSProc, which integrates the various material and cross-section processing routines used in previous versions of SCALE, including both the BONAMI and CENTRM self-shielding methods. BONAMI is an equivalence theory code which was extended to support the IR approximation, while CENTRM is an ultrafine group method. The common feature between them is a "cell data" section, where the analyst must model each equivalent lattice cell for self-shielding, either specifying equivalent pitch directly or specifying the Dancoff factor. This adds another potential area for input errors. For example, the analyst must remember to update the Dancoff factors when there is a change in the coolant or moderator density.

Polaris implements the Embedded Self-Shielding Method (ESSM) instead. While this is an equivalence theory method, the use of a transport calculation to determine escape cross-sections allows for the "cell data" section to be eliminated. No equivalent cells or Dancoff factors are required because the self-shielding calculation is performed directly on the full lattice.

Polaris also uses a new 2D solver based on the Method of Characteristics (MoC). This solver is typical and makes the usual assumptions for MoC solvers, such as a flat source term within each mesh cell. However, it also supports higher-order scattering, used by default instead of transport-corrected isotropic scattering. The eigenvalue solver applies CMFD for assembly calculations to accelerate convergence [2]. This transport solver is used both for the self-shielding calculation and the eigenvalue calculation, though the self-shielding calculation uses different default parameters as described later in this thesis.

Polaris is also compatible with the Sampler program. Sampler is a super-sequence for stochastic uncertainty analysis, analyzing the uncertainty in a model's results that arise from nuclear data uncertainties or uncertainties in the modelling parameters. Randomized nuclear data is taken from a set of pre-computed random perturbations to the nuclear data. Since computational requirements are increased by a factor equal to the number of samples, the performance of the Polaris ESSM and MoC solver are advantageous for such analyses and obtain orders of magnitude improvement as compared to traditional NEWT simulations for the same cases.

The version of Polaris in the SCALE 6.2 release has been tested and validated for PWR and BWR geometry. Polaris shows good agreement with the continuous-energy Monte Carlo code KENO for PWR analysis using the VERA benchmark, for k_{eff} and pin powers [7]. However, one limitation of the MoC method is that a very small ray spacing is required to accurately model very thin absorber regions, such as burnable poison coatings. This

results in a substantial increase in computation time. One potential alternate methodology would be to homogenize such regions into thicker regions. Lattice calculations for BWR geometries also agree well with KENO [8]. Independent studies have also found good agreement between Polaris and SCALE's previous validated lattice physics code TRITON [9].

2.4 Safety Analysis Using Coupled Neutronics and Thermalhydraulics Models

Safety analysis is a broad field involving the modelling of nuclear reactor systems to determine the outcome of accidents and other abnormal events, particularly with respect to the amount and distribution of radioactive material released in an accident [10]. The types of physics and systems that may be evaluated include neutronics of the reactor core, thermalhydraulics of the heat transport system, mechanical and thermal performance of the fuel, instrumentation and control systems, radiation transport and shielding, and containment analysis.

There are two common approaches to safety analysis that are typically used together: deterministic safety analysis and probabilistic safety analysis. Deterministic safety analysis focuses on the analysis of worst-case credible scenarios, or "design-basis accidents" [10], and demonstrating that the systems in place to mitigate these accidents perform as designed to keep consequences within certain limits. Probabilistic safety analysis determines the likelihood and frequency of events that could lead to an accident and evaluates risk based on how often given consequences are expected to occur.

The design basis accident are those accidents against which the design is to defend against the potential consequences of the accident. A list of such accidents can be generated in one of two ways, either a top-down approach or a bottom-up approach [10]. The top-down approach involves starting with the negative outcome and determining potential events that could result in that outcome, and so on, until a list of initiating events for the design basis is produced. The bottom-up approach, conversely, starts with a potential initiating event and then analyzes the consequences of that event. A related approach to these is the probabilistic approach, which determines risk based on the frequency and likelihood of various systems failing during normal operation or during the course of an event. Overall, a combination of analysis and past experience can be used to establish a design basis for the facility.

Once potential accident sequences are identified, many different types of models can be used to perform safety analysis. These include reactor physics models, which were covered in the previous subsections and are briefly reviewed in this subsection, as well as thermalhydraulics and heat transfer, which is covered in this subsection. Simulation of reactor control is also considered in this thesis.

As the available physical models and correlations have improved and as computers have become more powerful and able to simulate higher-fidelity models, there has been an evolution in safety analysis methodology away from conservative modelling assumptions and towards best-estimate-plus-uncertainty (BEPU) modelling. With traditional conservative models, parameters and correlations would be set for a worst case "bounding" scenario, such that if the system as modelled meets the design safety criteria, the real system is guaranteed to meet or exceed these criteria. For example, a large break LOCA analysis would assume an instantaneous break with no warning of a size that leads to the largest possible energy deposition in the fuel.

A best-estimate model, on the other hand, attempts to get as close to the ground truth as possible. This involves increasing the fidelity of models to incorporate the real-world physics of the system as much as possible and using improved and more specific empirical correlations wherever they are still needed. This also involves increasing the level of detail of a model, such as through modelling a larger number of equivalent fuel channels (or even individual channels) or using nodal methods in a core physics code rather than modal methods or point kinetics. The codes used in this work, including the SCALE package as well as TRACE and PARCS, are considered to be best-estimate codes. Best-estimate-plus-uncertainty aims to also quantify the uncertainties present in a model, including uncertainties both from the input data as well as from modelling assumptions and approximations. SCALE includes the TSUNAMI and Sampler sequences for uncertainty analysis. By quantifying the uncertainty, the confidence level in meeting or exceeding the design criteria may be determined.

The role of the reactor physics models is generally to determine whether control of the chain reaction is maintained in the conditions of the accident, or can be involved in analyzing the evolution of an accident where control of the chain reaction is lost. The models used are typically full-core models with reactivity feedbacks. Lattice physics calculations are used to calculate the effect of different feedbacks on the nuclear properties of the lattice, based on the thermalhydraulic conditions in the core, depletion of the fuel, and positions of reactivity devices. The perturbations to the lattice properties are then applied to the full-core model. The full-core model can be used either to calculate a steady-state critical configuration or to calculate reactor kinetics in a transient simulation.

Software codes that have been developed or adapted and used for CANDU reactor analysis include WIMS and DRAGON at the lattice level along with RFSP at the core level. These codes have been used to assess fuel bundle modifications [11] and have been benchmarked against reactor measurements from the Wolsong plants [12]. DRAGON in particular is useful for its 3D "super-cell" lattice calculations, used to calculate the effect

of the reactivity devices which are perpendicular to the fuel channels. RFSP, in particular, has been developed for CANDU modelling, as it can handle the CANDU-specific systems such as on-line refuelling along with the various flux measurements used for regulating the reactor [13].

Other codes which can be used for reactor physics analysis include TRITON, from the SCALE package, for lattice physics [14], and PARCS for full-core modelling [15]. In addition, MCNP and Serpent are Monte Carlo codes used for reactor physics. MCNP's Monte Carlo method can be considered as a "gold standard", and it has been used in benchmark calculations [16] and as a benchmark against which other codes and methods are compared [17].

Thermalhydraulics is another key field used in safety analysis. This field involves the modelling of fluid and energy flows through the reactor and heat transport system, and for safety analysis they can determine whether cooling of the fuel is maintained during a transient. Thermalhydraulic analysis models solve the momentum, mass, and energy equations for fluid flow. These are conservation equations and are typically solved in an integral form. A general form for these equations is as follows [10]:

$$\frac{D}{Dt}\iiint_V \psi \, dV = \iiint_V \Gamma \, dV + \oiint_S \boldsymbol{S} \cdot \boldsymbol{n} \, ds \tag{2.19}$$

Equation (2.19) states that the substantial derivative of a field variable ψ (either mass, momentum, or energy) is equal to the net volume source/sink term Γ in the volume plus the net surface source/sink term $\boldsymbol{S}$. The substantial derivative is the sum of the partial derivative with respect to time and the net flow of the field variable out of the volume [10]:

$$\iiint_V \frac{\partial \psi}{\partial t} dV + \oiint_S \psi \boldsymbol{v} \cdot \boldsymbol{n} \, ds = \iiint_V \Gamma \, dV + \oiint_S \boldsymbol{S} \cdot \boldsymbol{n} \, ds \tag{2.20}$$

If Gauss's law is applied, then the surface integrals are converted into volume integrals, and the volume integration can be dropped to achieve a derivative form [10]:

$$\frac{\partial \psi}{\partial t} + \nabla \cdot \boldsymbol{\psi v} = \Gamma + \nabla \cdot \boldsymbol{S} \tag{2.21}$$

Either form can be used depending on the analysis being performed and the mathematical approach taken (e.g., finite volume vs. finite difference vs. finite element).

The source terms depend on the field variable. For mass conservation, the only source terms are phase change terms. For the momentum equations, source terms include

gravity as well as stress terms such as pressure and viscosity. For the energy equations, source terms include the work from the momentum sources as well as heat sources/sinks.

With two-phase flow, the system can be modelled with a level of detail ranging from a simple homogeneous equilibrium model (HEM) with three equations (one per field variable) [10] up to detailed models where each phase is separately modelled. As an example of the latter, RELAP-7 uses seven equations, with one equation for each phase for each field variable, plus a seventh equation for the volume fraction of each phase [18]. Codes such as TRACE [19] also include additional equations for modelling non-condensable gas and dissolved solute transport.

However, the conservation equations are insufficient to model fluid flow. There are three equations and four unknowns. The first three unknowns are mass (or density), momentum (or velocity, or mass flow), and energy (or enthalpy, or temperature), with the fourth unknown being pressure. Thus, an equation of state for the fluid being modelled is also needed, which provides the relationship between density, enthalpy/temperature, and pressure [10].

Supporting relationships are also needed for the conservation equations, such as for modelling the interfacial interactions, for both fluid-fluid and fluid-wall, for both momentum and heat transfer, along with thermodynamic properties, flow regimes, and component data (to model components such as pumps and valves) [10]. These are typically empirical correlations, derived through experiments rather than from first principles. One important example is the frictional losses due to fluid flow, which are a function the flow velocity, pipe hydraulic diameter, fluid viscosity, and pipe wall roughness, as well as of the pipe geometry. Another important example is the heat transfer coefficient, which is a function of multiple fluid properties as well as flow velocity. Typically, many of these correlations are written in terms of dimensionless constants for which correlate well with experimental data, including [10]:

- Reynolds Number, $Re = \frac{\rho v L}{\mu}$, which measures the ratio of forces due to momentum and due to viscosity. A low Reynolds number indicates that viscosity dominates, leading to laminar flow, while a high Reynolds number indicates that momentum dominates, leading to turbulent flow. A transition between laminar and turbulent flow typically occurs at values of around 2000-3000 [10]. The L term is a characteristic length, which is the hydraulic diameter for pipe flow.
- Prandtl number, $Pr = \frac{c_p \mu}{k}$, which measures the relative diffusion rates of momentum and heat.

- Nusselt number, $Nu = \frac{hL}{k_f}$, which measures the ratio between convective and conductive heat transfer for a fluid.

A thermalhydraulic analysis code will typically discretize the system using a node-link model. In such a model, components such as pipes and headers are modelled as one or more nodes representing finite volumes within the component, along with links between the volumes within a component as well as between components. Generally, mass and energy information are stored in the nodes, while momentum information can be stored in the nodes, the links, or both. For example, the TRACE code calculates fluid densities and specific enthalpies for nodes, and flow velocities for links [19]. Codes construct a system of equations to solve the conservation equations, the equation of state, and the supporting correlations, for every node and link in the system. These equations are then solved numerically. A time-stepping algorithm is typically employed, for which the state of the model with respect to time is discretized and solved.

In addition to 1D fluid flows representing pipe flow, some codes may also model 2D and 3D flows. For example, TRACE has a vessel model for PWR flow modelling [19], while codes designed for CANDU simulation may include subchannel modelling, where the fuel channel is modelled as a set of coupled subchannels rather than as a single 1D pipe. In addition to modelling the fluid flows, thermalhydraulic codes may also include models of heat conduction in solids, radiative heat transfer, control systems (either generic or design-specific), or even reactor point kinetics. For example, the TRACE code includes such models [19].

Some of the codes developed for thermalhydraulics with CANDU reactors specifically in mind and used for analysis of CANDU reactors include SOPHT [20], TUF [21], and CATHENA [22]. Other thermalhydraulics codes, developed for light water reactor applications, include RELAP-5 [23] and TRACE [19]. These codes have been previously adapted and used for CANDU analysis, including the simulation of station blackout accidents [24] as well as for benchmarking of coupled simulations of transients [25].

There are important feedbacks between reactor physics and thermalhydraulics. The reactor neutronics is affected by the temperature of the fuel, coolant, and moderator, as well as the density of coolant and moderator. The reactor neutronics, in turn, affects the heat generated in the fuel and thus influences the fuel temperature, which propagates in the thermalhydraulics simulation. Therefore, it is common for reactor physics and thermalhydraulics to be simulated together.

Feedback in the core physics code is achieved through the use of branches in the lattice physics properties (i.e., homogenized cross-sections). Each branch describes a

perturbation of the lattice properties (e.g., fuel temperature, coolant density, presence of a reactivity device) and the resulting perturbation on the lattice physics properties compared to the reference lattice. The core physics code then calculates a weighted sum of branches for each node depending on the values of the properties, which are calculated by the coupled codes, such as the thermalhydraulics code. The change in lattice physics properties will result in a change in reactivity and flux distribution.

One example of code coupling for safety analysis is the use of RFSP, TUF, and RRS_em for coupled neutronics, thermalhydraulics, and reactor regulation, to simulate the response of the Bruce A generating station to a loss-of-flow event initiated due to the trip of one or more heat transport circulating pumps [26]. During such an event, the loss of flow in the heat transport system leads to an increase in pressure and temperature along with increased fuel channel voiding. The fuel channel voiding leads to an increase in reactivity and thus an increase in reactor power. The RRS emulator responds to both the neutronic and thermalhydraulic conditions which leads to the insertion of control absorbers to reduce the reactivity and reactor power.

It is also possible to couple the PARCS and TRACE codes. The two codes were designed with a built-in coupling methodology where both codes are contained within the TRACE executable and TRACE will run PARCS if specified by the input model file [27]. PARCS also includes a specification for a mapping input file that couples the neutronic nodes to thermalhydraulic components and heat structures [28]. Therefore, a coupled model requires a TRACE input deck, a PARCS input deck, and a mapping input. The mapping can be either volume-based or table-based. The volume-based mapping is designed for LWR analysis and automatically couples the PARCS core to a TRACE volume and calculates the weights for mapping. The table-based mapping is designed to be more flexible and specifies the mapping weights manually.

TRACE and PARCS are loosely coupled, as they share data each time step while maintaining independence between their solution methods. Data is transferred and PARCS is executed internally at the end of each time step. The coupled solution method is equivalent to a Euler method, as the coupled terms are assumed constant over each time step.

For coupled neutronics and thermalhydraulics, while reactivity feedback is instantaneous to changes in thermalhydraulic conditions, and the prompt response of the reactor power can be rapid. However, the fuel temperature takes time to respond to a change in power, as heat is the time integral of power.

As an example for illustrative purposes, if the specific heat capacity of UO_2 is approximated as 328 J/kg.K and the average density is 10.6 g/cm^3 [10], a CANDU fuel

pellet has a diameter of 12 mm and the linear element rating is 57 kW/m [10], then the rate of temperature increase is:

$$\frac{dT}{dt} = \frac{57000\,\frac{\mathrm{W}}{\mathrm{m}}}{\left(328\,\frac{\mathrm{J}}{\mathrm{kg}*\mathrm{K}} * 10600\,\frac{\mathrm{kg}}{\mathrm{m}^3} * \pi(0.006\ \mathrm{m})^2\right)} = 145\,\frac{\mathrm{K}}{\mathrm{s}} \tag{2.22}$$

Thus, if the power was instantaneously increased from zero to the maximum linear element rating, and the fuel pellets were insulated, it would take approximately 4.9 seconds for the temperature to increase from 290°C (approximate coolant temperature under operating conditions) to 1000°C. Thus, the time constant for fuel temperature response to reactor power in on the order of seconds. The response of the coolant, based on the rate of heat transfer from the fuel to the coolant, should follow a similar response. As the response is slow compared to typical time step sizes, the integrating effect of the temperature response is advantageous. If the time step size is 0.01 seconds, then an instantaneous 100%FP increase will only have a fuel temperature effect of approximately 1 K per time step. For a CANDU reactor, even for a rapidly evolving accident such as a large break LOCA, such power changes occur over a period on the order of one second [25], thus the feedback effects can be considered to be quite slow when compared to reasonable time step sizes, justifying the use of a loose, non-iterative coupling method.

As the necessary interval for coupling updates is potentially much greater than the thermalhydraulic time step size, especially for slowly evolving transients and steady state convergence, PARCS is capable of time step skipping for performance. For steady-state simulations, PARCS can recalculate the steady state every N time steps, for N up to 20. This can also be performed for transient simulations, with the caveat that the PARCS model will run N times slower than the TRACE model, thus is only suitable when attempting to calculate a steady state using the transient model, which may be desired if the steady state reactor power is unknown, as the PARCS eigenvalue solver assumes a known constant reactor power. For transient simulations, PARCS will also dynamically skip TRACE time steps based on the time step size specified in the PARCS input. PARCS will then take a time step only when the sum of the accumulated TRACE time steps is greater than or equal to (within a tolerance) the specified PARCS time step size. This allows for a simulation where the PARCS time step size is larger than the TRACE time step size, as the reactor kinetics can often be modelled with a longer time step compared to the thermalhydraulics.

2.5 TRACE Coupling Using the Exterior Communications Interface (ECI)

TRACE is the flagship thermal-hydraulic code developed by the United States Nuclear Regulatory Commission (USNRC), combining the functionality of RELAP5, TRAC-P, and

TRAC-B, to model system transients including large and small break loss of coolant accidents (LOCAs), for both PWRs and BWRs [27]. Functionality for modelling CANDU systems is also present. Coupling capabilities include a built-in coupling with the PARCS neutronic code as well as the Exterior Communications Interface (ECI) for the coupling of TRACE models or other non-TRACE codes to TRACE [29].

The ECI enables TRACE to be coupled with other codes without requiring any source code modifications to TRACE itself. The ECI provides the interface with which other codes can access a TRACE model's components and exchange data. Two levels of coupling are possible. The first, simpler level is an explicit coupling scheme, where the coupled physical models are solved individually, and they exchange data at predetermined points in the calculation. The second level is a tightly coupled mode where the solution schemes of the coupled codes are integrated together. The choice of coupling level is dependent on the requirements of the numerical methods being coupled with TRACE.

It should be noted at this point that the coupling between PARCS and TRACE is not through the ECI, and thus that PARCS cannot be directly accessed via ECI, only through TRACE. PARCS and TRACE are directly coupled within the executable with their own interface separate from ECI. TRACE-PARCS uses an explicit coupling scheme, where PARCS is executed at the end of each TRACE time step, to converge the flux in an eigenvalue problem or advance the flux in a transient problem. PARCS includes some time-step control in order to allow it to skip TRACE time steps during steady-state or transient simulations to reduce execution times, as generally the neutronic model can accept a larger time step than the thermal-hydraulic model.

With ECI, the coupling is always set up upon initialization, as TRACE defines all variables which may be accessed as well as where in the solution scheme that data may be transferred [29]. Coupled processes make requests for which variables they need and when they need them, and TRACE receives these requests and locates where these variables reside. The data transfers are then scheduled for the requested points in the simulation.

TRACE includes 18 synchronization points at different stages of the simulation, as shown in Figure 2.2. A TRACE simulation involves problem initialization, time step preparation, control system evaluation, fluid solution, heat conduction, and time step finalization. The solution scheme consists of multiple nested loops which allow for the fluid solution to be iterated or backed up under certain conditions. The semi-implicit solution can be iterated until convergence, while the solution can be backed up upon a water packing event or, more generally, if there is a failure or lack of convergence in the

solution, including repeated water packing events. In such cases, TRACE is able to revert to the beginning of the time step and try again with a smaller time step size.

The selection of synchronization points for data transfer depends on the tightness of coupling and on the coupled system being modelled. A loosely coupled scheme will typically transfer data at the beginning and/or end of the time step, or at the beginning and/or end of a specific part of the program flow, such as the control system evaluation. Conversely, a tightly coupled scheme will transfer data within the semi-implicit solver loop as well as at other points as needed.

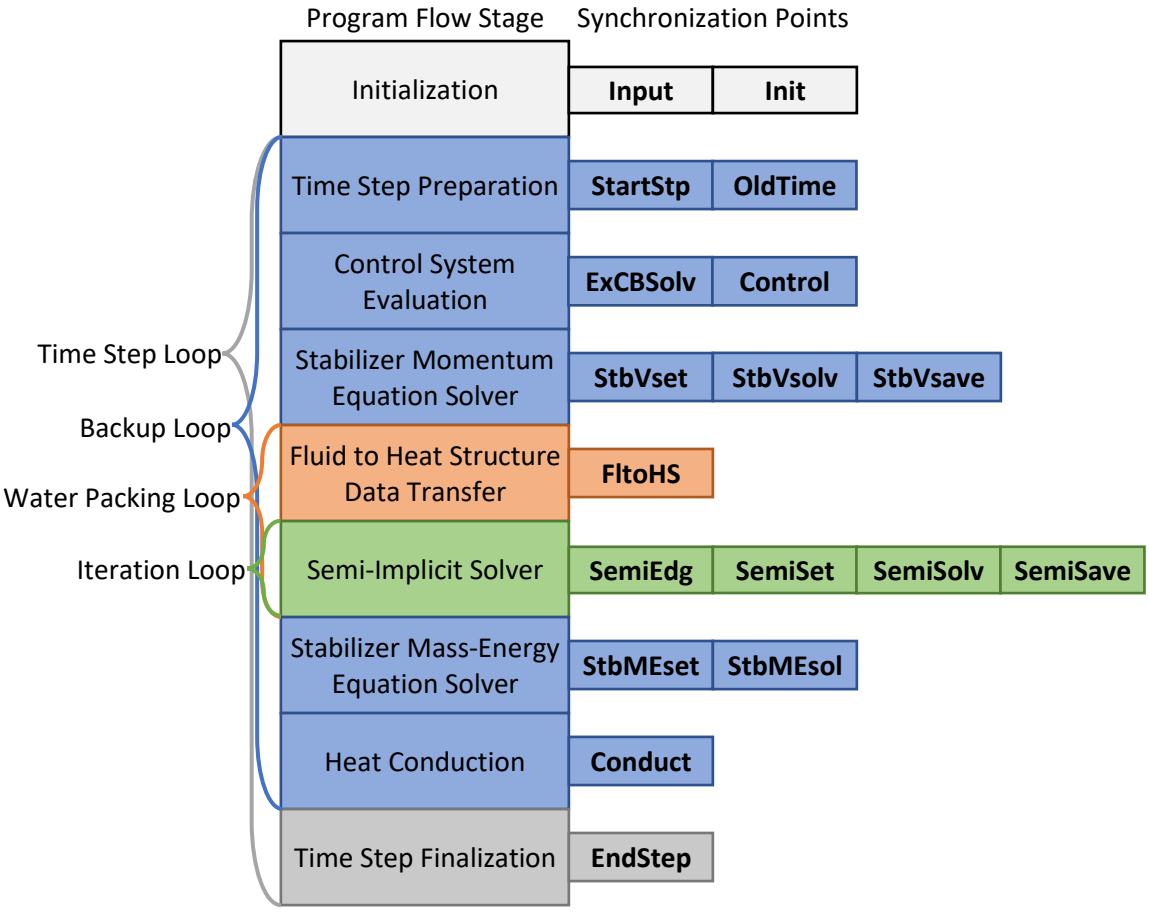

Figure 2.2: TRACE program flow structure with synchronization points [29]

Provided with TRACE is an ECI library, which is a set of code modules (primarily written in Fortran 90) used to provide ECI coupling capabilities to a programmer's code. Most of the modules should be used as-is, unless the programmer is making modifications to the ECI itself. However, the library does include one module, **SpecExTransM.f90**, which includes all of the ECI's program-specific subroutines, i.e., the subroutines that must be written by the programmer to work with their own program. The amount of work

required by the programmer for this module depends on the coupling requirements. Only minimal work is required if the programs are to be explicitly coupled and data requests are to be made only by the coupled program to TRACE.

The two most important subroutines for the programmer to consider for linking models together are **IsMissing** and **GetPointer**. **IsMissing** is a library subroutine used to request a component-variable. A task will call **IsMissing** for any data that it needs from another task in the same job. **GetPointer** is a library subroutine used to locate requested component-variables. It is part of the **SpecExTransM.f90** module and thus has a program-specific implementation. The role of **GetPointer** is to locate a requested component-variable in the task's own data structure. In summary, each component-variable request made by **IsMissing** in one task corresponds to locating that component-variable in a different task using **GetPointer**. As a result, all ECI programs are required to use the TRACE component concept for any data that may be requested by another ECI program.

With this in mind, the scheduling of data transfers takes place in three steps [29]:

1. Each task constructs two lists of component-variable requests, one per transfer direction. Each call to **IsMissing** corresponds to adding a new entry to one of these lists.
2. From these lists, each task constructs a list of missing components, based on component number, and sends it to each other task. Each task receiving a list of missing components will call the **SpecExTransM.f90** subroutine **LookForComp** to determine whether it possesses the missing component. This is used to determine where each component in the coupled simulation resides, the verify that each component exists in exactly one task.
3. Each task sends its component-variable requests to the other tasks corresponding to the locations of the respective components. For each request received, a task calls **GetPointer** to locate the component-variable in its own data structure. From this step, the transfer tables can be constructed.

In addition to standard coupling of individual component-variables or arrays of component-variables, TRACE supports tight coupling for fluid junctions. TRACE itself uses the concept of an "exterior" component, which corresponds to a component that exists in a different task. For an implicit and tightly coupled solution, each process must extend its systems of equations to account for the additional unknowns introduced by the exterior junctions, then a single process collects the partial solutions and solves for the exterior junction unknowns. A detailed description of this tight coupling is beyond the scope of this work.

For the actual simulation, the library includes the template for a subroutine called **TimeEvolve**, which replicates the program flow structure described in Figure 2.2. At each synchronization point, the program makes a call to the **ExtTransfer** subroutine, which sends and receives data for the corresponding synchronization point. All ECI-enabled programs must follow this program flow structure and can perform their own computations as needed between synchronization points.

In this work, it was decided to develop a Python interface to the ECI library. Python has the advantage of supporting object-oriented programming (OOP) as well as extensive module and library support, including the NumPy package for numerical and scientific computing [30]. It is also a more popular programming language and thus a Python interface makes ECI more accessible. One disadvantage compared to Fortran is slower performance, thus making Fortran preferable for performance-critical computations. However, various applications, including control system models, do not require heavy computations, and thus are a good candidate for being written in Python.

Python is capable of being interfaced with other programming languages, including C/C++ and Fortran, allowing for the advantages of both to be combined. Applications include writing performance-critical sections of a Python program in a lower-level language such as C, as well as interfacing with existing code libraries written in other programming languages. For example, NumPy's matrix operations are written in C. The Fortran interface is provided by the F2PY tool included with NumPy [31].

2.6 Reactor Regulation for CANDU Reactors

One of the key design aspects of the CANDU design is its use of digital computers for control and regulation, including the Reactor Regulating System (RRS). CANDU reactor control is broken up into many different modules, each performing a particular task. The RRS itself consists of several modules that, together, keep the reactor operating at criticality, at a steady power, and with an even flux distribution [32]. Figure 2.3 summarizes the different components of the reactor regulating system along with process variables which are used as inputs. There are some differences in the RRS between different CANDU designs but they all follow a similar structure.

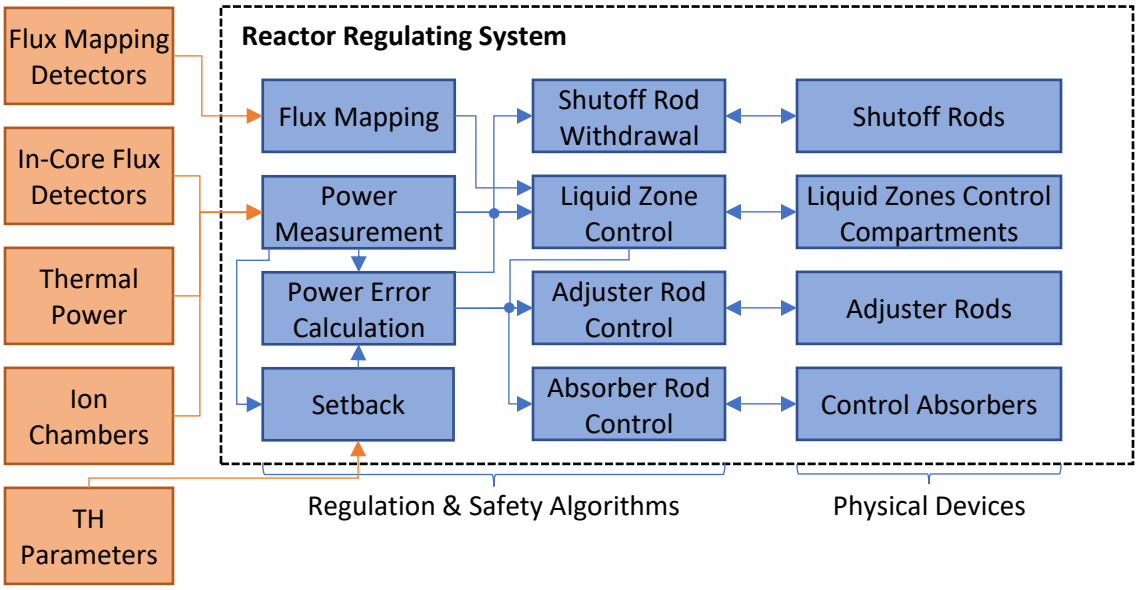

Figure 2.3: Reactor Regulating System Diagram (CANDU-6 Type)

Due to the large size of the CANDU core, the flux in distant parts of the core is only weakly coupled. Large imbalances in flux distribution can arise from relatively small perturbations, including those from reactivity device movements, refuelling, thermal-hydraulic transients, and saturating fission product evolution. In particular, the evolution of xenon-135 in the core leads to positive feedback over a period of hours, not only in bulk reactivity but also in differential reactivity between opposite sides of the core. Therefore, the RRS is required to not only maintain the overall reactor power, but also to cancel out any flux "tilts" and maintain a symmetrical power distribution. The CANDU core is divided into 14 control "zones", each with its own liquid zone control compartment and in-core flux detectors.

Power measurement and calibration (MCP) [32] converts the raw measurements taken from various sensors and flux detectors into useful measurements of reactor power. This includes not only neutron detectors but also thermal power measurements. CANDU reactors include several types of detectors for flux measurements which are used for different purposes:

1. Ion Chambers, typically 3 located just outside of the core.
    - Purpose: Measure absolute power at low powers, and measure rate of power change.
    - Advantages: Nearly instantaneous response to power changes.
    - Disadvantages: Cannot be placed in core; location makes them unsuitable for absolute measurements at high powers or for spatial control.

2. In-core flux detectors (ICFDs, platinum or Inconel), 28 detectors grouped into 14 assemblies.
   - Purpose: Measure flux in the core for bulk and spatial power control at high power.
   - Advantages: Quick response to flux variations.
   - Disadvantages: Residual delayed response components make them unsuitable for low power measurements.
3. Flux mapping detectors (vanadium), 102 in a CANDU-6.
   - Purpose: Produce a detailed map of the flux distribution in the core; used to calibrate in-core flux detectors for spatial control.
   - Advantages: Sensitive to neutrons only.
   - Disadvantages: Delayed response (half-life of 225 seconds).
4. Thermal power measurements, on the primary side and/or secondary side based on power level.
   - Purpose: Measure the absolute thermal power for calibration of neutron power measurements.
   - Advantages: Measurements correspond directly to reactor power (unlike flux readings).
   - Disadvantages: Thermal response is delayed from neutron response; measures total power only.
5. Fully Instrumented Channels (FINCHes), distributed throughout the core.
   - Purpose: Measure the absolute thermal power for bulk and spatial calibration of neutron power measurements.
   - Advantages: Measurements correspond directly to channel powers (unlike flux readings).
   - Disadvantages: Thermal response is delayed from neutron response; FINCH nominals[1] must be calibrated to accurately represent bulk and zone power.

The measurements from these devices serve as input to the MCP module. The presence and number of each type of device is dependent on the specific CANDU design. In general, there is always a source of core and zone flux measurements, a source of core flux rate of change measurements, a source of thermal power measurements against

[1] FINCH nominals are calibrated to represent the channel powers, as calculated by a core physics model, for a target flux shape (100%FP, no tilt). Absolute FINCH powers from thermalhydraulic measurements are converted into relative values for the RRS, where a value of 1 represents a channel power equal to the nominal power. Multiple FINCH readings are averaged for each axial zone pair. In this thesis, channel powers from an RFSP time-average model were used for the nominals.

which to calibrate the flux measurements, and a source of more fine-grained channel power measurements against which to calibrate the zone power readings.

The outputs of the MCP module include the following:

- Reactor thermal power, in full power (FP) units.
- Reactor linear neutron power, in full power (FP) units.
- Reactor logarithmic neutron power, in decades relative to full power.
- Reactor logarithmic rate, in percent present power per second (%PP/s).
- Zone neutron powers, in full power (FP) units relative to the nominal zone powers.

Thermal power calculations are made by measuring heat transport system (HTS) or steam generator (SG) process parameters, or from FINCH measurements. These parameters include flow rates along with inlet and outlet enthalpies, which are in turn derived from temperature, pressure, and density/quality measurements. The physical units (kW or MW) are converted to FP units. For FINCH readings, this is done by dividing the reading by their corresponding nominal power corresponding to a nominal full-power flux distribution, which may be derived using computer models.

The thermal power measurements are used to calibrate the linear and logarithmic neutron power measurements. The general process in calibrating neutron power readings is given by Figure 2.4. Uncalibrated neutron power readings are compared against thermal readings after being filtered to account for the thermal propagation delay, which is a physical property of the system. This difference is then filtered with a 180 second time constant in order to minimize the effects of noise and short-term transients. This filtered difference is the calibration factor, which is added to the uncalibrated power reading to obtain the calibrated neutron power. This is performed separately for the ion chamber and ICFD readings.

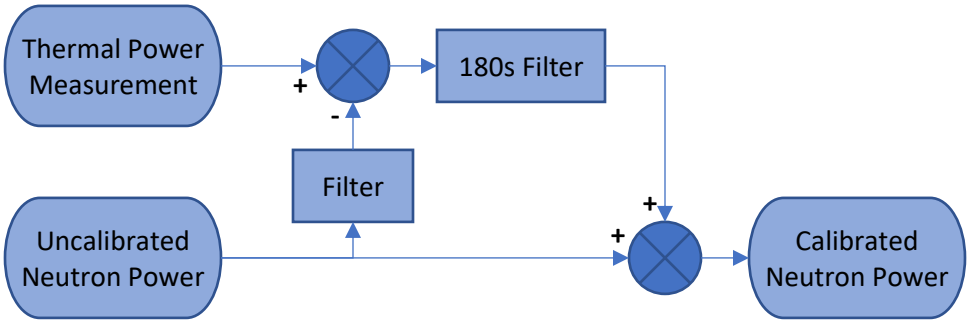

Figure 2.4: General process flow for neutron power calibration.

For zonal neutron powers, the calibration is dependent upon whether the station uses flux mapping (via vanadium detectors) or FINCH readings. Calibration via FINCH readings

uses the same process as in Figure 2.4. Calibration via vanadium detectors uses reconstructed zonal powers, as fractions of the respective nominal zone powers, in place of thermal power measurements. In such a case, the uncalibrated ICFD readings are filtered using a 325 second time constant, corresponding to the 225 second half-life divided by ln(2), in order to match the response of the vanadium detectors.

The separation of linear and logarithmic power is of note as it is more than simply a mathematical transformation between the two. While linear power (P_{LIN}) is always sourced from the ICFD readings, logarithmic power (P_{LOG}) can be sourced from either the ICFD readings or the ion chambers, depending on the operating point of the reactor. This is due to the ICFD readings being unreliable at low power, while ion chamber readings are unsuitable for high power. Above 15%FP thermal, P_{LOG} is based on the ICFD readings and is equal to the base 10 logarithm of P_{LIN}. Below 5%FP thermal, P_{LOG} is equal to the calibrated ion chamber power. Between these two values, P_{LOG} is linearly interpolated between the two readings for a smooth transition.

Power error calculation (CEP) [32] is responsible for calculating the power error between the measured power (from MCP) and the demanded power (generally based on operator input). The resulting power error becomes an input to the various RRS modules responsible for controlling the reactivity devices.

Demanded power is dependent upon whether the station is being operated in turbine-leading or reactor-leading mode. In turbine-leading mode, generator power is specified by the operator, with the reactor power setpoint and demanded power being controlled to maintain the steam generator pressure. In reactor-leading mode, the reactor power setpoint and manoeuvring rate are directly specified by the operator.

CEP operates on logarithmic powers. In order to smoothly control the reactor power, the demanded power is ramped to the setpoint at the specified rate, with a linearization factor applied so that the power is manoeuvred linearly at high power and logarithmically at low power. In addition, a deviation limiter is applied to prevent the demanded power from exceeding the measured power by more than a specified margin. Both the demanded power and its rate of change are inputs to the power error calculation.

The power error calculation contains both a power term and a rate term, as shown in Equation (2.23). The power term takes the difference between the actual and demanded reactor power, in order to maintain the reactor power at the demanded power. The rate term takes the difference between the actual and demanded log rate, in order to provide derivative control of the reactor power as well as to maintain the desired manoeuvring rate during power changes, helping to prevent oscillation of the reactor power around the demanded power. The weight of the power term is controlled by the K_B coefficient,

which is equal to 1.0 above 25%FP, 0.2 below 5%FP, and interpolated for intermediate values, as tightly controlling the absolute power is less important at low powers, and reducing K_B improves system stability. The coefficient for the rate term is fixed at 0.5, as controlling the rate of power change is equally important at low and high powers.

$$E_P = K_B(P_{LOG} - P_{DLOG}) + 0.5(R_{LOG} - R_{DLOG}) \tag{2.23}$$

The liquid zone control module (CBL) [32] monitors the power error, liquid zone compartment levels, and zone powers, and manipulates the valve lift for the 14 liquid zone compartment fill valves. This is the primary method for reactivity control during normal operation, as it allows for fine-grained reactivity control for both the overall reactor and regionally.

CBL calculates both a bulk lift and a differential lift, which are added to a bias which balances out the constant outflow of light water. The bulk lift is directly proportional to the power error – a positive power error will cause all liquid zones to fill while a negative power error will cause all liquid zones to drain. The differential lift is calculated for each zone and consists of both a power term and a level term, as described by Equation (2.24):

$$DLIFT_i = \alpha_{Ti} K_T DPZ_i + (1 - \alpha_{Ti}) K_H ENIV_i + K_L ENIV_i \tag{2.24}$$

In Equation (2.24), DPZ_i is the difference between the measured linear zone power in zone i and the average linear zone power, while $ENIV_i$ is the difference between the average liquid zone water level and the liquid zone water level in zone i. Typical gains are $K_T = 3$, $K_H = 0.006$, and $K_L = 0.001$, thus the power term dominates over the level term under normal circumstances. The α_{Ti} term is used to phase in differential power control and is the product of a bulk power term α_{TP} and a level term α_{TLi}. α_{TP} is ramped from 0 to 1 between 15%FP and 25%FP, as shown in Figure 2.5 [32], as spatial power control is less reliable and less necessary at low powers. α_{TLi} is equal to 1 when the liquid level in zone i is between 10% and 80%, and is ramped to 0 as the liquid level approaches 0% or 90%, as shown in Figure 2.6 [32]. Overall, this means that differential lift defaults primarily to power control, maintaining zone powers at their nominal relative values to eliminate flux tilts, but at low reactor power or for very high or very low individual zone fills, differential lifts transition to level control, trying to bring the respective zone fills closer to the average fill.

In addition, once the total valve lift is calculated, limitations are applied based on the zone compartment level to prevent excessively high or low fill levels, as these result in operational issues. These limitations are given by Figure 2.7 [32].

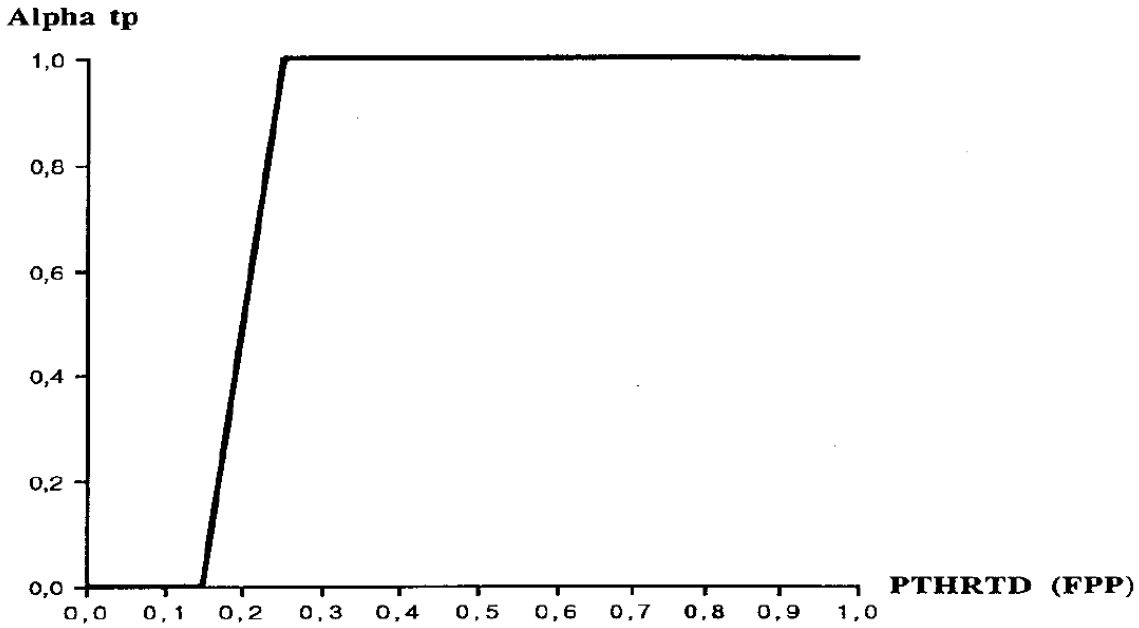

Figure 2.5: Bulk power term α_{TP} for liquid zone control

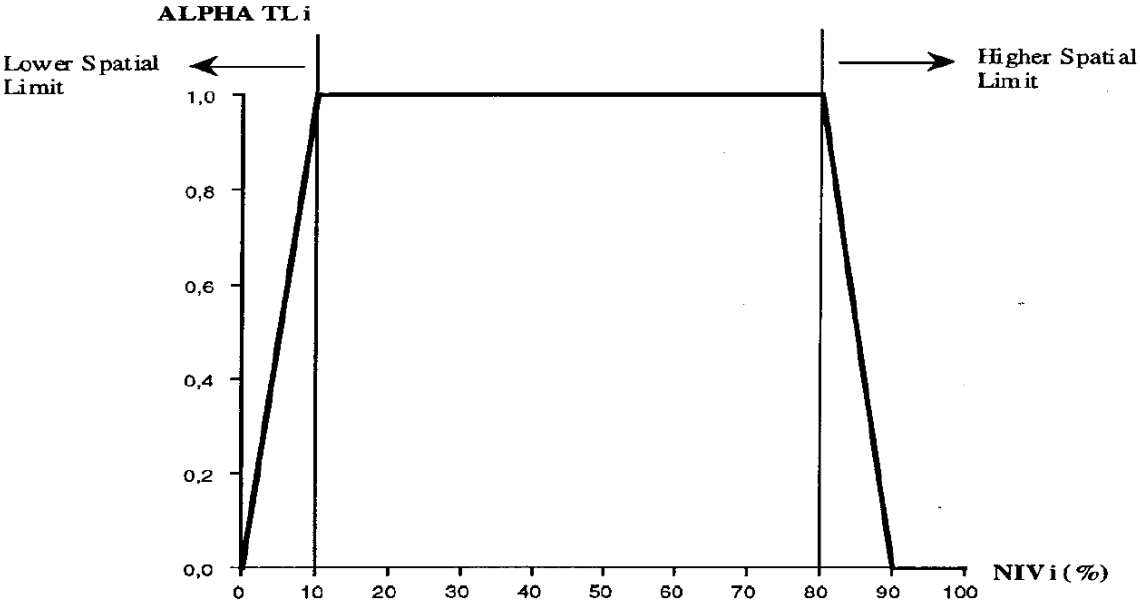

Figure 2.6: Spatial level term α_{TLi} for liquid zone control

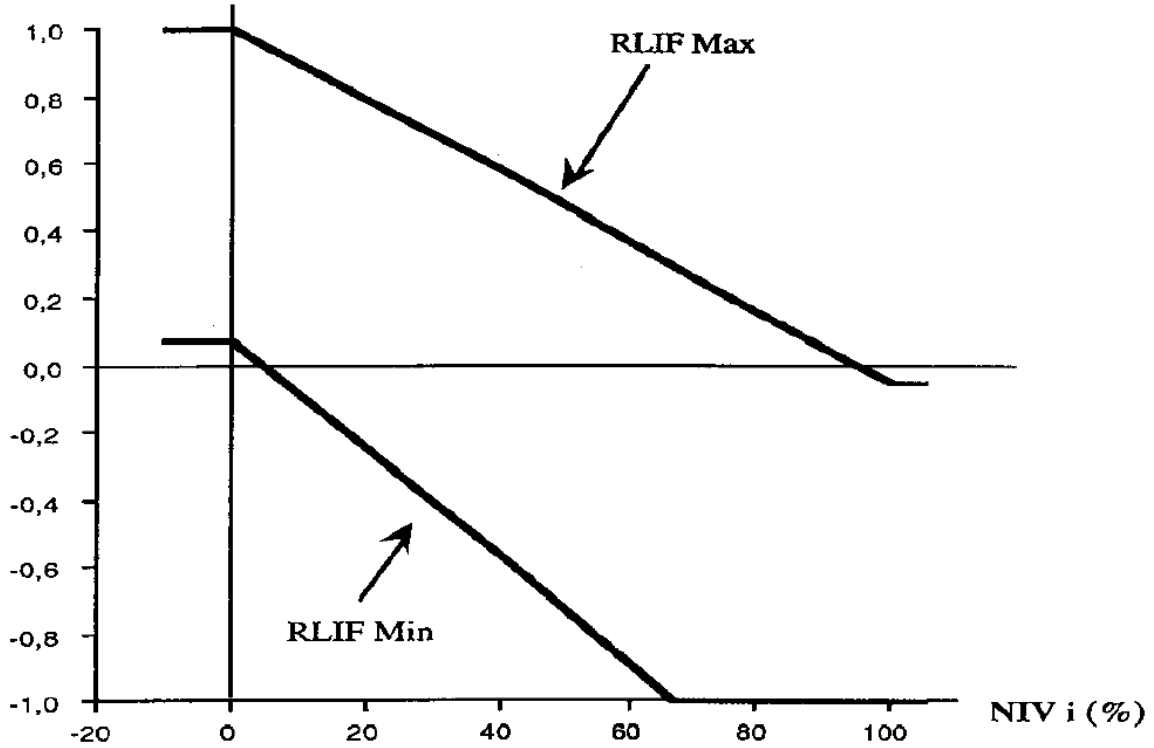

Figure 2.7: Limits for liquid zone valve lift based on liquid zone level

The adjuster rod module (CBC) and control absorber module (CBS) receive the power error and average liquid zone fill level as inputs [32]. The modules control the drive motors of the adjuster rods and control absorbers, respectively. Rods are driven out of the core if the power error is sufficiently negative or the average liquid zone level is too low, while rods are driven into the core if the power error is sufficiently positive or the average liquid zone level is too high. Overall, adjuster rods and control absorbers are driven only when liquid zone control is insufficient for controlling reactivity, either because the reactivity change is too large or too quick for the liquid zone control to compensate.

Adjuster rods are divided into 8 banks, normally in-core. A bank is withdrawn when the power error drops below -3% or the average zone level drops below 20%. A bank is re-inserted when the power error exceeds 3% or the average zone level exceeds 70%. Once a bank starts to be driven, it will be fully driven in or out of the core unless an opposing movement is actioned. Other rules further inhibit adjuster movements, for example, preventing adjuster movements at full power which could cause overpower in certain fuel channels.

Control absorbers are divided into 2 banks, normally out-of-core. A bank is inserted when the power error exceeds 1.5% or the average zone level exceeds 80%. A bank is

withdrawn from the core when the power error drops below -3% or the average zone level drops below 70%. Otherwise, both banks are held in place. This has the overall effect of inserting control absorbers only when negative reactivity is needed and withdrawing them as soon as negative reactivity is no longer needed.

Shutoff rods are divided into 2 banks. While shutdown system 1 (SDS1) is responsible for dropping them in the core, shutoff rod extraction (RBA) [32] is part of the RRS. Once a reactor trip has been cleared, the two banks of shutoff rods can be withdrawn, one bank at a time, as long as the logarithmic power rate is below 7% and the power error from CEP is negative.

In reactors with flux mapping detectors, the flux mapping module (FLU) [32] reconstructs the flux distribution in the core. This is done through modal synthesis, which constructs a flux distribution as a linear combination of 15 orthogonal flux modes which best fits the detector readings. This flux distribution is then used to compute 14 zone powers, which are used to calibrate the zone power measurements from the ICFDs.

The reactor setback module monitors several plant parameters, including local neutron flux, zone powers, along with conditions in the steam generators, deaerator, moderator, and end shields [32]. If any parameter crosses a setpoint, a setback is triggered. The setback works by overriding the demanded power and rate in CEP, so that the reactor power is ramped down gradually through the normal RRS control module response. The setback persists until either the triggering parameter becomes normal again or the reactor power reaches the setback endpoint. The endpoint power and setback rate depend on the parameter triggering the setback.

The reactor stepback module, while not part of the RRS itself, is still part of the digital reactor control. Similarly to the setback, it monitors several plant parameters, including zone powers, reactor logarithmic power change rate, heat transport system pressure, heat transport system circulating pump status, and for turbine trip [32]. If any parameter crosses a setpoint, a stepback is triggered. Unlike a setback, a stepback indicates that a rapid power reduction is required, though not necessarily a complete shutdown. This power reduction is achieved by cutting power to the 4 mechanical control absorber clutches, allowing the rods to drop into the core by gravity. The power to the adjuster rod motors is also removed, preventing their movement. Generally, a stepback continues until either the stepback condition is cleared or the reactor power reaches (or is predicted to reach) a given endpoint power, at which point the clutches are re-energized to arrest the mechanical control absorbers. Thus, the absorbers may be fully or partially dropped in the core.

One particular application of the stepback is upon a turbine trip. With no steam passing through the turbine, steam is bypassed to the condenser. However, the condenser is only able to sustain roughly 60%FP worth of steam flow in this manner. A partial stepback is thus used to reduce reactor power to roughly 60%FP, where the reactor can continue to operate in poison prevent mode [10]. This prevents a "poison-out" condition where build-up of xenon-135 prevents criticality, keeping the reactor available to reconnect to the grid as soon as possible. If a poison-out were to occur, the reactor would be unable to start for roughly two days.

3 Methodology

3.1 Polaris Geometry Additions

Typical lattice physics codes only include primarily low-level geometry automation. In such codes, individual surfaces are specified in the input file, and the volumes divided by these surfaces filled with different materials. These codes can typically automate the placement of regular or irregular repeated geometry, such as the fuel pins in a fuel assembly or bundle. High-level geometry automation can be achieved through writing scripts which translate the high-level description into a lower-level input description.

Polaris, by contrast, performs the high-level automation itself. The input file specifies the basic parameters which define a PWR or BWR assembly, and the code itself is responsible for the low-level geometry construction. This shifts the responsibility of accurately specifying the low-level geometry from the analyst to the software developer. However, this also restricts the set of problems that Polaris can model when compared to other codes with more generalized geometry specifications.

To add CANDU modelling support to Polaris, it was necessary to add the capability to the source code, specifically to the module responsible for parsing the input file, along with the module responsible for converting the high-level geometry specification into a low-level geometry specification. First, a “cluster” geometry was added to the code. In contract to the rectangular lattice used to specify LWR geometry, the cluster geometry is a circular lattice geometry. A cluster geometry specification includes the number of pins in each ring, the radial pitch of each ring relative to the centre of the cluster, the azimuthal pitch of each ring in degrees, as well as the type of pin to insert in each lattice position. Both the CANDU fuel bundle and the RBMK fuel assembly can be described by a cluster geometry.

Second, a “shell” card was added to allow specifying the circular shell that is to surround the bundle. This is in contract to the BWR channel box which is rectangular in shape with rounded corners. The circular shell is used to specify the pressure tube, gas annulus, and calandria tube of the CANDU fuel channel.

Additionally, several smaller additions were performed, such as adding predefined material specifications for certain CANDU-specific materials as well as adding a “CANDU” option to the “system” card to provide a different set of default properties from the “PWR” and “BWR” options.

Many of the existing Polaris cards used for LWR specification are also used for CANDU specification, including the “pinmap” card for specifying the pin type at each lattice

position, as well as the "pin" cards that specify the materials and geometry of the individual fuel pins.

3.2 Verification and Validation of Polaris Self-Shielding

As there was little prior work on evaluating the ESSM for CANDU applications, it was necessary to evaluate the results of ESSM, both the self-shielded cross-sections and their effect on lattice physics predictions.

The self-shielding methods available in the SCALE package are the ESSM, CENTRM, and BONAMI. Both CENTRM and BONAMI are available in the TRITON sequence, while ESSM is only available for Polaris. In addition, it is also possible to evaluate self-shielded cross-sections by performing a continuous-energy Monte Carlo calculation; this was done using KENO-VI. All codes use ENDF/B-VII.1 nuclear data processed through the AMPX package.

To evaluate the significance of biases in individual self-shielded cross-sections, it is necessary to compute the sensitivity of integral quantities to the individual self-shielded cross-sections by nuclide and energy group. These quantities can include k_{eff}, reaction rates, and homogenized few-group cross-sections. To compute the sensitivities, the TSUNAMI code was used. TSUNAMI-3D was used to compute the sensitivity to k_{eff} using KENO in continuous energy, while TSUNAMI-2D was also used to compute the sensitivity to k_{eff} and reaction rates using the NEWT transport code. The TSUNAMI-3D results are more accurate but less precise. Sensitivities to scattering cross-sections converge very slowly, along with the sensitivities of reaction rates, therefore the TSUNAMI-2D results were used to supplement the TSUNAMI-3D results where it was impractical to achieve adequate convergence.

Continuous energy KENO was also used to tally reference multigroup cross-sections against which the self-shielded cross-sections were compared. By tallying flux and reaction rates in each fuel pin, the continuous-energy cross-sections can be collapsed into 252-group cross-sections that accuracy represent the self-shielded cross-sections of the pins, and with a precision determined by the number of particle histories.

In addition to evaluating the multigroup self-shielding tables distributed with SCALE, it is also possible to generate new self-shielding tables using the IRFFACTOR sequence. To use this sequence, a series of pin cells or homogeneous mixtures is specified, covering a range of background cross-sections from infinite dilution to no dilution. For each cell, the background cross-sections for the IR approximation are determined, and the corresponding self-shielded cross-sections are calculated using CENTRM. These results are used to create new self-shielding tables, that when used on any of the representative cells, should give the same result whether BONAMI or CENTRM is used. A cell with similar

properties to the representative cells should give similar results between BONAMI (or ESSM) and CENTRM. Therefore, the generation of new self-shielding tables representative of the problem being analyzed should result in more accurate self-shielding when using BONAMI or ESSM than using the distributed libraries based on LWR pin cells.

3.3 Development of Python Interface for Exterior Communications Interface (ECI)

The Exterior Communications Interface (ECI) is a coupling interface provided with the TRACE code that allows the development of coupled codes without the need to modify the TRACE source code [29]. The two major components of ECI are the ECI driver and the ECI library.

The ECI driver is a Java daemon which is responsible for setting up the inter-process communication using either sockets or a shared memory map. When an ECI central process is started, it will communicate its task list to the ECI driver. The driver is then responsible for starting the satellite processes and establishing the communications between them. The driver does not take part in the solution process itself.

The ECI library is set of modules, primarily written in Fortran 90, used to add ECI coupling capability to existing codes or to develop new ECI-enabled codes. This can be done directly for Fortran 90 codes, or an interface can be developed for codes in other languages. Several Fortran 90 examples are also included with ECI, including examples of TRACE to TRACE, TRACE to non-TRACE, and non-TRACE to non-TRACE coupling. TRACE to TRACE coupling can be used to break up a large model into smaller pieces which can be solved in parallel, while TRACE to non-TRACE coupling can be used to couple different physical or control models to a TRACE model.

The Python ECI package was developed in this work to provide another option for developing ECI-enabled programs. This package consists of a modified version of the ECI library, an additional Fortran 90 interface which is exposed to Python, and a Python module which provides an object-oriented method of creating and accessing ECI-linked variables. The Fortran-Python interface is created using F2PY [31], which allows Fortran arrays to be treated like NumPy arrays and for Fortran functions and subroutines to be called from Python.

The Python ECI package (PyECI) consists of the following components:

1. A modified version of the ECI Fortran library distributed with TRACE.
2. An interface module, written in Fortran, whose primary purpose is to handle the data allocation for ECI variables requested by a Python program.

3. A set of Python modules which provide an object-oriented interface for ECI-enabled Python programs.

An ECI-enabled Python program must additionally use a Python version of the TimeEvolve template, that is equivalent to the Fortran TimeEvolve template included with the ECI library and was developed for this thesis.

A detailed technical description of PyECI can be found in the Appendix, Section 9.1. This description provides more detail on each component in PyECI and in particular the implementation of the Python modules, as well as the steps involved in implementing an ECI-enabled Python program.

Figure *3.1* shows how data moves between the different modules for a program using the ECI Python package. Data retrieved by the ECI library is stored in a `values` array, which is accessed by **Variable** objects. The Python program then accesses these **Variable** objects. In the other direction, the Python program stores data in the `values` array through the **Variable** objects, where the ECI library can retrieve the data and send it to other ECI processes. Global variables, such as delt, can be accessed directly by the program or through the ECI package.

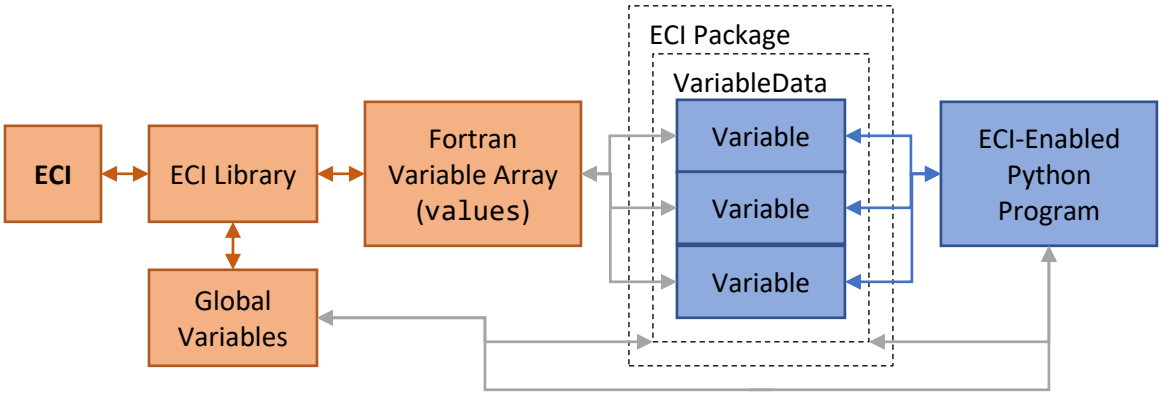

Figure 3.1*: General ECI program data linkage with Python interface.*

3.4 Development of ECI-Coupled Reactor Regulation System Model

TRACE includes a general-purpose control system model which can be used for modelling various reactor control systems. The TRACE control system model consists of three types of components: signal variables, control blocks, and trips. Signal variables read physical system properties and other simulation parameters and make them available to the control system model. Control blocks perform mathematical or logical operations on one or more input variables, which can include signal variables or other control blocks. Trips monitor a signal variable or control block and can be set or reset

based on the input variable crossing a setpoint. The behaviour of fluid and power components can be linked to the control system, for example, to control the opening of a valve, the speed of a pump, fluid boundary conditions provided by a FILL or BREAK component, or the power generated by a heater. When coupled with PARCS, the positions of reactivity devices can also be linked to the control system.

However, it is also possible to couple more complex behaviour into the control system model using the ECI. Coupled codes can access control blocks to read their values or replace the present values with their own. In this manner, complex behaviours can be implemented by writing code in a more familiar programming language. Due to the complexity of the CANDU RRS, it was decided to develop a model of the CANDU RRS in Python and couple it to the TRACE-PARCS model using the ECI.

Figure 3.2 describes the different components of the RRS model and how they are connected in terms of information flow. Note that this figure is similar to Figure 2.3, except that the model also includes the stepback routine as well as SDS1, which are not part of the RRS itself. All components within the dotted line are part of the RRS model itself, while components outside the dotted line are parameters from PARCS and TRACE that act as inputs to the RRS model.

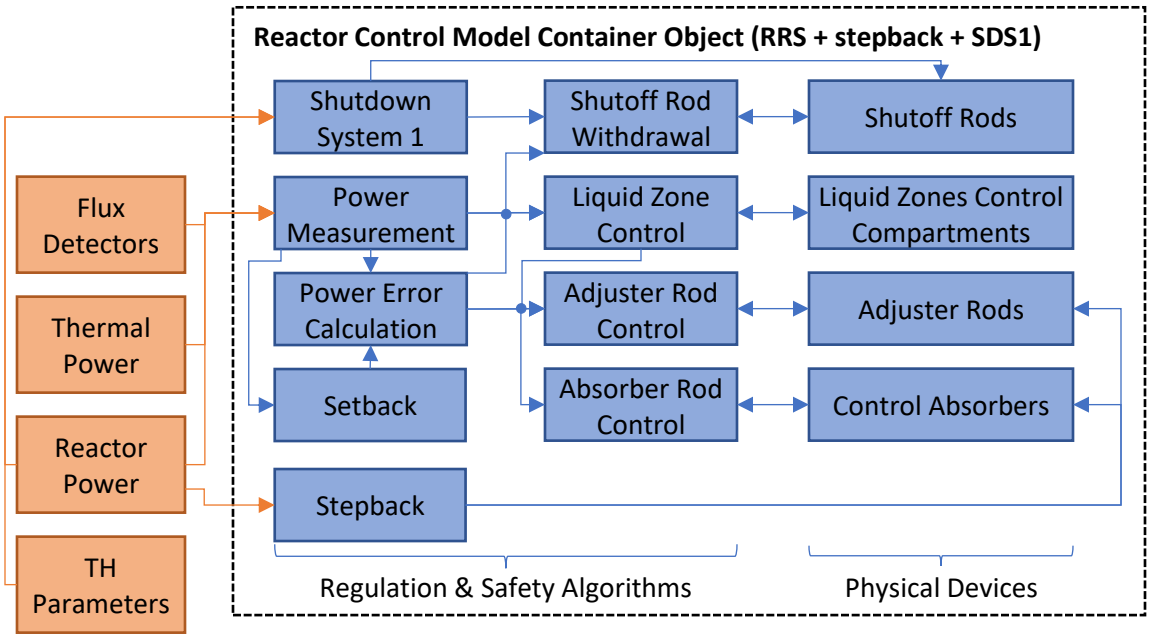

Figure 3.2: Reactor control model as implemented in this work

The basic building block in the RRS model is the StateContainer class, which is described in the Appendix, Section 9.1.4.8. The RRS model is constructed in a modular, objected oriented manner, with each block in Figure 3.2 representing an individual Python

class derived from the StateContainer class. The StateContainer itself, as the name suggests, contains an instance of another class named State (see Appendix, Section 9.1.4.7), which contains all of the data related to the time-dependent state of a system. The decision to create a StateContainer class was done to more clearly separate the time-dependent state from other independent properties of the derived classes.

This implementation was chosen to support TRACE's time-step backup and checkpoint restart capabilities. Each module in the RRS retains both its old-time state and new-time state. When a new time step begins, the new-time state overwrites the old-time state. When a time-step backup occurs, the new-time state is reset to the old-time state. To perform a restart dump, the state is serialized into a JSON-formatted text file.

In the RRS model, the RRS object acts as the top-level object in the hierarchy, containing all of the individual StateContainer-derived modules, as well as being a StateContainer-derived object itself. When advanced in time, the RRS object automatically advances all sub-modules in time. The sub-modules can be grouped into two categories – those which represent digital components of the reactor control, and those which represent the physical devices themselves. As the digital logic of the RRS operates in discrete intervals, the RRS object also has a role of synchronizing the time advancement of its sub-modules.

Figure 3.3 summarizes the process taken when RRS performs a time step. If the end time falls beyond the time at which the next RRS loop would occur, either the "fast" loop every 0.5 seconds or the "slow" loop every 2 seconds (which may also include a "calibration" loop, every 16 seconds), the simulation advances to the point of the next RRS loop instead of to the end of the time step. All physical devices are updated to simulate advancing time. An iteration of the corresponding RRS loop is performed, then the entire process is repeated until the end of the time step. There are two exceptions to this process:

1. If a stepback is active, the loop which monitors the stepback occurs every 0.25 seconds, and the process is adjusted to accommodate this extra stopping point.
2. The shutdown system is updated every TRACE time step, independent from the lockstep of the rest of the modules.

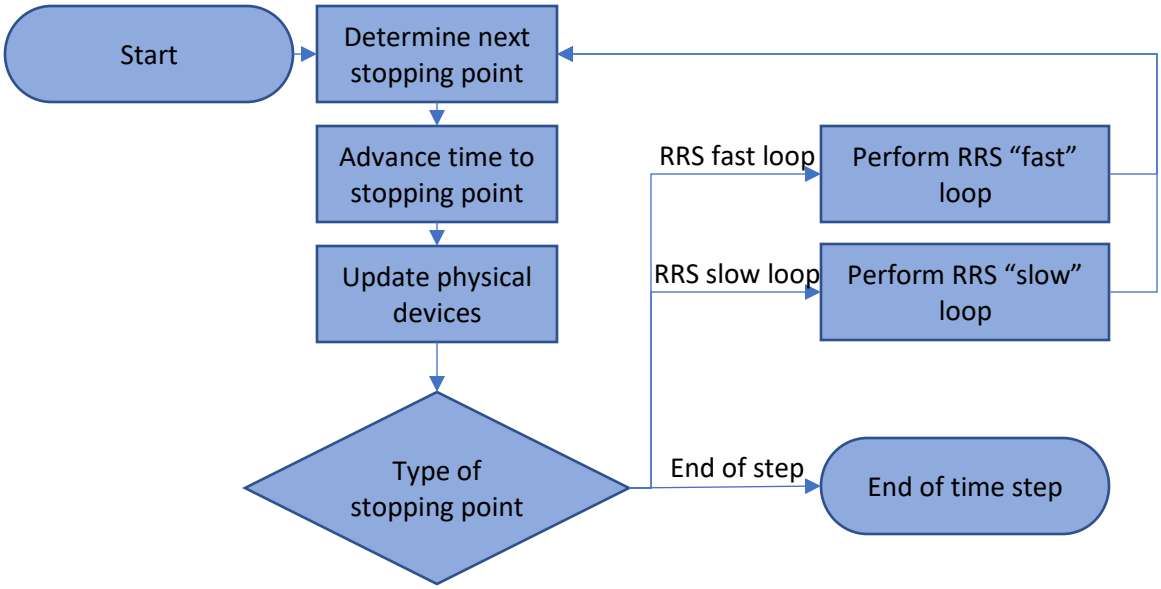

Figure 3.3: Reactor control time step program flow

Several additional features are present. Each RRS module is also able to report whether it has converged onto a steady state, and the RRS simulator reports convergence if all such modules report convergence. As well, there is functionality to record variables from the RRS state to TRACE control blocks, which are automatically updated on every time step.

The power error calculation module also has additional functionality for steady-state calculations. In the steady-state eigenvalue calculation mode of PARCS, reactor power is not allowed to vary with time, and is instead an independent variable. Therefore, the power error feedback of the RRS is unsuitable for steady-state calculations. Instead, the RRS model should try to maintain an eigenvalue of unity, corresponding to a critical reactor. To accomplish this, the power error is overridden to be directly proportional to the difference between k_{eff} and a k_{eff} setpoint, the latter defaulting to unity. The gain for this feedback can be modified as needed.

In addition, the RRS simulator includes the ability to decouple the RRS time from the TRACE time as a steady-state acceleration scheme. As the flux and reactivity feedbacks are instantaneous in the steady-state calculation, and the goal is to converge on a steady state rather than accurately model the time evolution, the RRS simulator can be configured to take a fixed step size on every TRACE time step, irrespective of the actual size of the thermal-hydraulic time step. This effectively "fast-forwards" the reactivity devices, accelerating their convergence.

Separate from the RRS model is a simplified flux detector model, also based on the StateContainer class. Each simulated flux detector averages the neutron flux in one or more PARCS nodes, and can model a prompt or delayed response, consisting of up to one prompt component along with any number of delayed components. This response can then be input into the RRS power measurement model.

Directly modelling the fully instrumented channel (FINCH) measurements through thermal-hydraulic measurements is not possible due to the channel lumping scheme used in the thermal-hydraulic model. Therefore, FINCH measurements are also implemented as a StateContainer-derived class which reads PARCS powers, applying a filter to simulate the delayed thermal response.

3.5 Methodology for Long Transients

While TRACE can handle relatively large time step sizes for slowly evolving systems, very long simulations still require significant computation time, and for coupled simulations, PARCS and the reactor control system set a maximum limit on the practical time step size. In addition, for very long transients, it is important to consider the evolution of fuel burnup and its reactivity feedback, which is not supported by the distributed version of PARCS.

For the case where the reactor is operating in a quasi-steady state when not considering burnup, the transient can be modelled as a series of steady state runs with burnup calculations performed after each steady state convergence. While depletion calculations are not supported by PARCS when coupled with TRACE, the depletion calculation can be performed as a standalone calculation using the converged reactivity feedbacks from the coupled calculation.

For the case where the reactor physics evolves rapidly or where the evolution of the thermalhydraulics is required, a full transient calculation using TRACE is required. To accomplish depletion in this scenario, there are several alternatives. The depletion can be performed on a steady state representative of average conditions during the transient or can be performed by integrating the nodal powers over the transient and running an external depletion calculation. However, with source code access, it was possible to add a transient depletion capability, which updates the depletion and cross-sections on each time step.

Even with transient depletion capability, it is still not optimal to perform slow, long-duration transients as a single transient simulation, as the time evolution in the transient mode is slow. Figure 3.4 presents a hybrid methodology for this type of simulation, using a shim transient as an example. During such a transient, the fuelling machine is unavailable, so there is a gradual decrease in core reactivity due to depletion,

compensated for by a gradual draining of the liquid zones by RRS. When the average liquid zone level is sufficiently low, additional positive reactivity is added by withdrawing one bank of adjuster rods. For such a transient, the gradual draining of the liquid zones happens over a period of days, while the adjuster rod withdrawal transient happens over a period of minutes. Thus, the slow portion of the transient can be modelled as a series of quasi-steady models, with a depletion step interval measured in hours. When the zone level reaches the threshold to initiate the adjuster withdrawal procedure, a transient model is used to simulate the adjuster withdrawal procedure.

In both cases, saturating fission products such as xenon-135 are modelled transiently. Their concentrations are held constant during the quasi-steady-state calculation, with the time-dependent solutions applied to update their concentrations both for each depletion step as well as for each time step of the coupled transient.

This procedure can be utilized for any transient where the total duration of the transient is very long, but the duration of the "active" portions of the transient are comparatively short, as long as the evolution of the transient is governed by depletion, and the parameter which governs the timing of events can be determined and measured through the steady-state runs.

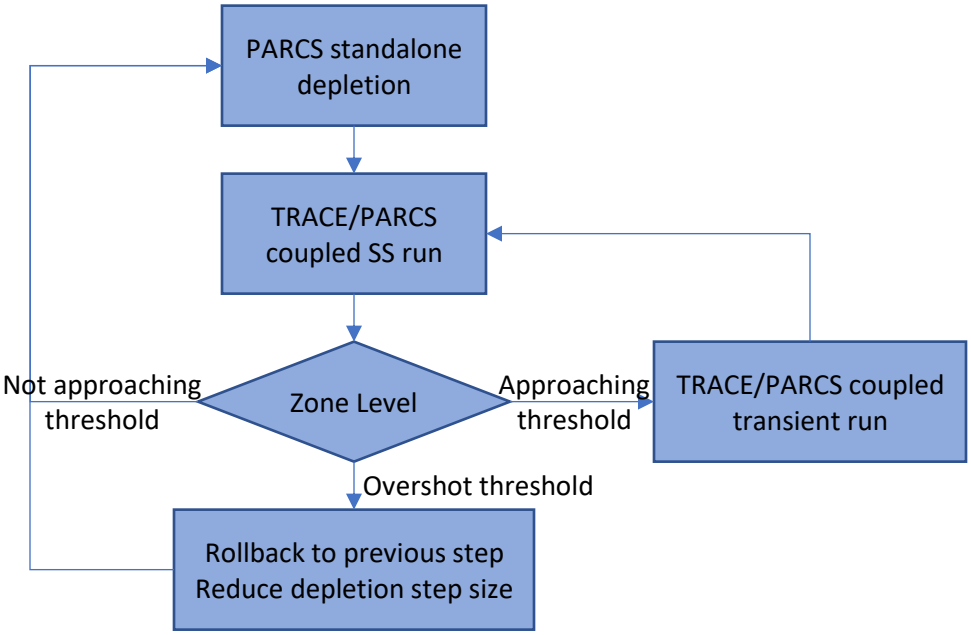

Figure 3.4: Hybrid methodology for performing a shim transient

The withdrawal of adjuster rods increases the reactor's channel and bundle power peaking factors. Therefore, to remain within channel and bundle power limits, the overall

reactor power must be derated prior to withdrawing adjuster rods. This has the side effect of triggering a xenon transient, which temporarily introduces additional negative reactivity to the core. As well, the CANDU design has a small power coefficient of reactivity [33], in contrast to the large negative power coefficient found in LWRs. Therefore, the dominant reactivity effect from the power manoeuvre is due to the xenon transient. If this reactivity change exceeds the worth of the withdrawn adjuster rods, the average liquid zone level will decrease below the threshold at which the adjuster rod bank was withdrawn. This can lead to very low average liquid zone levels, compromising spatial control or even necessitating the premature withdrawal of additional adjuster rod banks.

One strategy to counteract this effect is to perform the power manoeuvre early, at a higher average zone level, anticipating that the additional negative reactivity from the xenon transient will trigger the adjuster rod bank withdrawal by RRS within minutes, as the average zone level decreases to the adjuster withdrawal threshold. Typically, the adjuster rod banks are locked in place by the operator, with a bank of adjuster rods unlocked and placed under RRS control only when the manoeuvre is to be performed. The xenon peak is reached within several hours of the manoeuvre, after which the transient introduces positive reactivity to counteract the effects of depletion, leading to a gradual increase in the average zone level over the course of roughly one day or less, before depletion once again becomes the dominant reactivity effect. The exact timings of each event depend on the size of the power manoeuvre, which determines the magnitude and duration of the xenon transient.

The redistribution of power also influences the spatial control component of the liquid zone control system. As liquid zone control attempts to minimize zone flux tilts, more light water will be added in the vicinity of the withdrawn adjuster rods, which can lead to a significant imbalance in liquid zone compartment levels, potentially compromising spatial control.

4 Paper 1

4.1 Publication Details

S. Younan, D. Novog, "Extension and Preliminary Validation of the Polaris Lattice Physics Code for CANDU Analysis," *Nuclear Engineering and Design*, vol. 361, 2020. doi: 10.1016/j.nucengdes.2020.110540

The code extensions and modifications were designed and implemented by the first author (Simon Younan). The validation exercises, carried out via code-to-code comparison, as well as comparison of the reference code to experimental data, were also carried out by the first author. Support was provided by the research supervisor (Dr. David Novog) in the form of research conceptualization and project administration. Both authors worked together to conceptualize the validation tests. The SCALE code developers, particularly Matthew Jessee and William Wieselquist, were consulted over the course of the development. The paper was written by the first author, with Dr. David Novog providing support in revising and editing the paper.

Special thanks must be given to the SCALE development team, particularly Matthew Jessee and William Wieselquist for their development of SCALE codes including Polaris, along with Kang Seog Kim and Doro Wiarda for their support on using AMPX and IRFFACTOR for generation of self-shielding factors.

4.2 Preface

The field of safety analysis uses reactor physics calculations to predict the behaviour of a reactor core under varying postulated conditions. This is frequently done in two steps, consisting of a high-fidelity lattice physics calculation followed by a lower-fidelity full-core calculation. The lattice physics calculation is used to calculate flux-weighted volume-averaged nuclear properties of a fuel assembly or channel under a wide range of conditions. The lattice nodes can then be homogenized for the core physics calculation.

One part of reactor physics safety analysis is sensitivity and uncertainty analysis. Uncertainties are present not only in the properties of the lattice model, but in the nuclear data that determines neutron reaction rates with different materials. A sensitivity analysis determines how much a given output parameter (such as reactivity) changes for a given change in one or more input parameters (such as neutron cross-sections), while an

uncertainty analysis determines the uncertainty of a given output parameter given the uncertainties of relevant input parameters. Uncertainty analysis involves the propagation of uncertainties throughout a calculation.

There are two approaches to uncertainty analysis. One is a deterministic approach, which propagates uncertainties mathematically. While this gives precise results, it requires codes to be designed or adapted to perform the additional calculations required for the uncertainty propagation. As well, the deterministic approach generally involves linear approximations, thus reducing accuracy, particularly when larger uncertainties and greater non-linear effects are involved. The other approach is a stochastic uncertainty analysis, where the uncertain input parameters are randomly sampled, and the calculation repeated many times. The stochastic approach can be readily applied to most calculations, provides accurate results, and uncertainties can straightforwardly be propagated through a sequence of different calculations (such as lattice physics followed by core physics), but requires many samples for statistical convergence, multiplying the required calculational effort.

The SCALE code package is a set of reactor physics codes distributed by ORNL, designed primarily for criticality safety, radiation shielding, lattice physics, and uncertainty analysis. The most recent release as of this writing is version 6.2, released in 2016. This release introduced a number of modernizations, including intermediate resonance self-shielding models, a stochastic uncertainty analysis code (Sampler), and the new Polaris module for lattice physics to provide an alternative to the existing TRITON module. The Polaris module is the focus of this work.

Polaris, when compared to TRITON, provides improved performance along with a greatly simplified input structure. This is primarily achieved through the embedded self-shielding method, introduced for both its speed as well as its ability to operate on the same lattice geometry as the transport solver, avoiding the need to specify equivalent pin cells or Dancoff factors to perform the self-shielding calculation. The caveat is that the geometry specification is limited to pin cells and LWR lattices. This work developed a geometry specification for CANDU channels.

While the transport solver is similar to existing solvers, the embedded self-shielding method relies on self-shielding factors included in the multi-group nuclear data libraries, with the distributed libraries based upon self-shielding calculations on LWR pin cells. As CANDU reactors are moderated by heavy water, additional sets of self-shielding factors based on heavy water pin cells, as well as homogeneous mixtures of both light and heavy water, were evaluated alongside the distributed self-shielding factors. The results of the Polaris code using different sets of self-shielding factors were compared to the results of

TRITON-based computations, and the code-to-code biases were compared to the stochastically calculated nuclear data uncertainties from Sampler. In this work, Sampler was executed on a set of 60 pre-generated samples of cross-sections, using the Polaris model.

In addition, the sensitivities to cross-sections by energy group for k_{eff} and key reaction rates were computed and used to identify and isolate the biases resulting from the self-shielding methods. The sensitivities were calculated using the TSUNAMI sensitivity and uncertainty analysis code, on both a 3D KENO continuous energy Monte Carlo model and a 2D TRITON model using the NEWT transport solver, with the 3D calculation only being used for the sensitivity of k_{eff}, as the sensitivities for reaction rates do not converge adequately within a reasonable number of Monte Carlo particle histories. The continuous energy model was also used to tally cross-sections in the 252-group library structure; these cross-sections are taken as the reference "ground truth" cross-sections for the CANDU model. The self-shielded cross-sections from the other codes are then extracted, and the cross-section biases are multiplied by their respective sensitivities in order to obtain the resulting biases of k_{eff} and reaction rates.

4.3 Summary of Contributions

The SCALE 6.2 code package, including source code, was provided by ORNL through the RSICC with an individual license agreement. I communicated with ORNL to discuss the goals of the project and to acquire knowledge on how the code was structured and where in the code I would need to implement my changes to add CANDU geometry support to the Polaris code. I performed the code changes independently and presented my progress to ORNL through the SCALE user group meetings, and shared my code changes with ORNL upon completion, for integration into a future SCALE release.

For the verification and validation work, the scope of the work and the simulation cases to run were discussed with the research supervisor. I had prepared the input files for the simulations and executed the simulations, along with developing scripts to automate a portion of this work. I collected the results of the simulations and compiled them into a presentable format.

The writing of the journal paper, including the presentation of the results, was my own work, with editorial support from both Dr. Novog and from the peer review process. The peer review process also led to some modifications and additions to the methodology and corresponding adjustment of results.

Nuclear Engineering and Design 361 (2020) 110540

Contents lists available at ScienceDirect

Nuclear Engineering and Design

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

Extension and preliminary validation of the Polaris lattice physics code for CANDU analysis

Simon Younan*, David Novog

McMaster University, 1280 Main Street West, Hamilton, Ontario L8S 4L8, Canada

ARTICLE INFO

Keywords:
Lattice physics
CANDU reactors
Method of characteristics
Embedded self-shielding method

ABSTRACT

Polaris is a 2D lattice physics code available for the study of light water reactors (LWRs). It includes a method of characteristics (MoC) transport solver and the embedded self-shielding method (ESSM). Its advantages include a simplified user input structure, quick performance, and integration into the SCALE package, utilizing its nuclear data libraries, along with other codes such as the ORIGEN depletion code and the Sampler stochastic uncertainty analysis sequence. This work extends the Polaris code to support CANDU lattice transport calculations and performs some preliminary verification and validation, comparing the code to NEWT and KENO, along with to benchmark experimental data. Heavy-water specific self-shielding factors were also compared to those distributed with SCALE. In general, some differences are observed, but these are generally comparable to other code-to-code differences and smaller than nuclear data uncertainties. The generation of self-shielding factors has a noticeable effect, particularly on the benchmark coolant void reactivity calculation. Heavy-water specific self-shielding factors, while not required, can potentially improve the accuracy of ESSM-based calculations, but the choice of parameters can significantly affect the results. Overall, the Polaris transport and self-shielding calculations should be suitable for CANDU analysis.

1. Introduction and literature review

1.1. Overview of Polaris

Polaris is a lattice physics sequence in SCALE, introduced in version 6.2 (Oak Ridge National Laboratory, 2017), provided as an alternative to, but not a replacement of, the older TRITON sequence. Its defining features include a streamlined input model and an implementation of the embedded self-shielding method (ESSM).

Unlike TRITON, Polaris accepts geometry input through specification of typical LWR pin and lattice properties, rather than the unit and surface definitions used in TRITON. The Moonraker module is then responsible for building the lattice out of units and surfaces to the user's specification. This greatly reduces the amount of repetition in the user input, along with automating tasks such as pin and material subdivisions. The cost is a lack of flexibility, being unable to model not only exotic lattices, but even unsupported yet typical lattices such as CANDU or VVER lattices.

The Embedded Self-Shielding Method (ESSM) is a modified Bondarenko method which uses a set of independent one-group transport solutions over the problem geometry to calculate the escape cross-section, eliminating the requirement of calculating Dancoff factors with only a minimal performance penalty over the standard Bondarenko method (Jessee et al., 2014). Unlike TRITON, Polaris does not require a CELLDATA section, eliminating a potential source of model inconsistency that can arise from user error. This method is further detailed in the following section and in the referenced literature.

Several studies have performed verification and validation for Polaris, for the currently-supported PWR and BWR applications. For BWR models, k_∞ is found to be within a ± 2 mk bias target as compared to a reference continuous energy calculation for most models, and generally showing a small positive bias of about $+1$ mk on average for unrodded configurations, when compared to continuous energy KENO (Jessee et al., 2017, p. 1305). Actinide and fission product concentrations for depleted BWR fuel, compared to assay measurements, show similar trends as TRITON on PWR fuel (Gauld and Mertyurek, 2019). For PWR analysis, a preliminary version of Polaris calculated reasonably accurate quantities, with the exception of certain quantities such as macroscopic fission cross-sections for a rodded assembly, while being an order of magnitude faster than TRITON (Labarile et al., 2015, p. 422.5).

The main goal of this project is to demonstrate that the Polaris code can be extended to support the CANDU lattice geometry, and compare results to existing SCALE modules for CANDU calculations, as well as

* Corresponding author.
E-mail addresses: younans@mcmaster.ca (S. Younan), novog@mcmaster.ca (D. Novog).

https://doi.org/10.1016/j.nucengdes.2020.110540
Received 26 September 2019; Received in revised form 21 January 2020; Accepted 27 January 2020
Available online 14 February 2020

for benchmark against well-established criticality experiments. The work focuses on the applicability of the ESSM self-shielding factors as provided within SCALE (based on LWR spectra) to CANDU conditions.

1.2. Overview of self-shielding methods

The three self-shielding methods available in the SCALE code system are CENTRM, BONAMI, and ESSM. CENTRM is a continuous-energy transport solver which solves for the pointwise flux spectrum for resonance energies, for simple geometries such as pin cells. BONAMI is a Bondarenko method code, upgraded to use the Intermediate Resonance (IR) approximation to permit full energy range Bondarenko calculations. ESSM is the embedded self-shielding method used by Polaris. These methods are described in detail in the SCALE manual (Oak Ridge National Laboratory, 2017); a summary of this literature follows.

Traditionally, SCALE multi-group sequences first apply BONAMI to self-shield the AMPX library over all energies, then apply CENTRM to refine the solution for the resolved resonances (Williams, 2011, p. 160). Originally, BONAMI only supported the narrow resonance (NR) approximation, which was suitable for higher energies above 20 keV. This is still the default sequence for TRITON, but the addition of the IR approximation allows for the option of using BONAMI on its own, when speed is more important than accuracy.

1.2.1. CENTRM

CENTRM is the most rigorous and mathematically accurate self-shielding method. The method solves directly for the problem-specific continuous-energy flux in order to collapse into the desired self-shielded multi-group structure. All resonance interactions, and therefore resonance interference effects, are accurately accounted for. This comes at two costs. The first cost is in terms of geometrical accuracy. The continuous-energy solver can only operate on pin cells and other equally simple structures. Irregular pin structures are accounted for through the Dancoff factor, which describes the probability of a neutron streaming from the fuel pin to another fuel pin without scattering, and may be calculated for each pin in an irregular structure by deterministic or Monte Carlo methods. CENTRM can then calculate a Dancoff-equivalent lattice pitch for a regular lattice, and run the solver. The second cost is speed. Even with the simple geometry, self-shielding of a single pin takes several minutes. This can become time-consuming for problems with many unique pins, or with many branch points. This can be simplified by TRITON by assigning multiple depletion materials to one self-shielding material as an approximation (Oak Ridge National Laboratory, 2017, p. 3–29); the compositions are averaged for the self-shielding calculation.

1.2.2. BONAMI

The traditional method of resonance self-shielding, also known as the Bondarenko method, is to generate a nuclear library with a self-shielding factor table. The flux behaviour depends on the background cross-section, or the strength of the scattering of all nuclides in the system besides the resonant nuclide (Knott and Yamamoto, 2010, p. 924). Flux-weighted average cross-sections are computed for different background cross-sections, which are then tabulated in the nuclear library.

The background cross-section consists of a “volume component” for other nuclides in the fuel, and a “surface component” for the moderator (2010, p. 925). The latter, also known as an “escape cross-section” depends on the pellet geometry as well as a Dancoff factor (2010, p. 926).

BONAMI is a straightforward implementation of this method. In SCALE 6.2, it is extended to apply an intermediate resonance (IR) approximation, allowing for the use of full-range Bondarenko factors (Oak Ridge National Laboratory, 2017). For a regular lattice, the Dancoff factor can be calculated automatically; the analyst inputs the Dancoff factor in other cases.

In SCALE, the Bondarenko factors are tabulated in the multi-group library, and interpolation is performed to obtain problem-specific self-shielded cross-sections. The calculation of background cross-sections along with table interpolation is much faster than the continuous-energy solver of CENTRM. For the NR approximation, there is an analytical solution. This is used for all nuclei where $Z < 40$. For the more general IR approximation, there is no analytical solution, so, to generate the tables, CENTRM solves the transport equation for representative models. For ^{235}U, ^{238}U, ^{239}Pu, ^{240}Pu, ^{241}Pu, ^{91}Zr, and ^{96}Zr, the model is a LWR pin cell. For other nuclides where $Z > 39$, the model is a homogeneous mixture of the resonant nuclide with hydrogen.

1.2.3. ESSM

The Embedded Self-Shielding Method (Williams and Kim, 2012) is similar to the Bondarenko method, relying on the same approximations and most of the same calculation steps. The primary difference between BONAMI and ESSM is that the latter has a different method of calculating the escape cross-section. When the IR approximation is applied to the slowing-down equation, Eq. (1.1) is obtained (Jessee et al., 2014, p. 3). Eq. (1.1) is a one-group fixed-source transport equation – Polaris uses its own transport solver to calculate the flux for the entire geometry.

$$\hat{\Omega}\cdot\nabla\psi_g(\bar{r},\hat{\Omega}) + \Sigma_{t,g}(\bar{r})\psi_g(\bar{r},\hat{\Omega}) = \frac{1}{4\pi}\Sigma_{wr,g}(\bar{r})\phi_g(\bar{r}) + \frac{1}{4\pi}\Sigma_{nr,g}(\bar{r})\Delta u_g \tag{1.1}$$

$$\Sigma_{wr,g}(\bar{r}) \equiv \sum_i N_i(\bar{r})(1-\lambda_{g,i})\sigma_{p,i},\ \Sigma_{nr,g}(\bar{r}) \equiv \sum_i N_i(\bar{r})\lambda_{g,i}\sigma_{p,i}$$

Under equivalence theory, the effect of the non-fuel materials is replaced by an equivalent NR scatterer. In Eq. (1.1), the transport term $\hat{\Omega}\cdot\nabla\psi_g(\bar{r},\hat{\Omega})$ expresses the effect of the non-fuel materials, and applying equivalence theory yields Eq. (1.2):

$$(\Sigma_{t,g}(\bar{r}) + \Sigma_{esc,g}(\bar{r}))\phi_g(\bar{r}) = \Sigma_{wr,g}(\bar{r})\phi_g(\bar{r}) + (\Sigma_{nr,g}(\bar{r}) + \Sigma_{esc,g}(\bar{r}))\Delta u_g \tag{1.2}$$

By plugging in the flux from Eq. (1.1), Eq. (1.2) can be solved for the escape cross-section, which is then used to calculate background cross-sections and self-shielding factors for the entire geometry, without requiring a Dancoff factor or equivalent regular lattice. As the above equations depend on self-shielded cross-sections (e.g. $\Sigma_{t,g}(\bar{r})$), the calculation is iterated until the cross-sections converge. This is done independently for each energy group, as shown in Fig. 1.1. Each iteration can take a fraction of a second, with the entire calculation being much quicker than CENTRM. As well, while more detailed geometries take longer to solve than simple geometries, ESSM solves the entire geometry at once, rather than one pin at a time.

1.2.4. Summary of self-shielding methods

Overall, CENTRM makes the fewest assumptions on the physics of the resonance self-shielding effect. By comparison, BONAMI and ESSM make several assumptions:

1. The scattering source can be described by the IR approximation.
2. Resonance interference effects are unimportant.
3. The self-shielding factor tables are representative for the system under analysis.

Assumption 2 may be considered true when only the overall assembly properties are desired, such as k_∞ and macroscopic cross-sections, but is untrue when considering nuclide densities and reaction rates for certain nuclides, such as ^{151}Sm (Knott and Yamamoto, 2010, p. 1061). However, resonance interference effects with ^{238}U can be significant (2010, p. 1062). In SCALE, Bondarenko iterations crudely

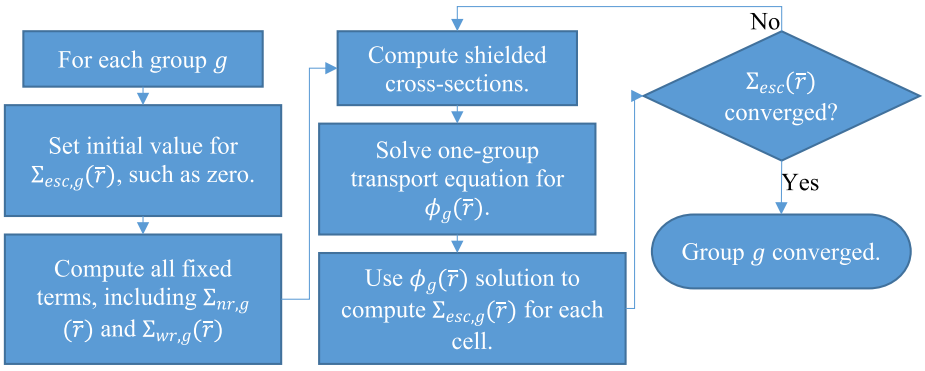

Fig. 1.1. Flowchart for ESSM escape cross-section calculation (Jessee et al., 2014, p. 4).

account for resonance interference (Oak Ridge National Laboratory, 2017, p. 7–198). The effect of resonance interference is also reduced by resolving the low-lying ^{238}U resonances in the multi-group structure.

The current version of Polaris and its implementation of ESSM have only been tested on light water reactor designs, as those are the only designs currently supported. ESSM in general has been implemented and tested with other codes and models, with one study using it for double-heterogeneous fuel, though new self-shielding factors were required (Li et al., 2017, p. 2). Therefore, the extension of Polaris to other reactor types should include a validation of the self-shielding results to show whether the same self-shielding factors can be used, or whether new self-shielding factors are required for each reactor class.

Assumption 3 shall be tested in this study. While the 252-group SCALE library should be suited for most classes of thermal reactors, the self-shielding factors were generated using LWR models. Therefore, the verification and validation should determine whether these LWR-based self-shielding factors and assumptions are appropriate for HWRs.

1.3. Previous literature on CANDU lattice physics

Typical reactor physics analysis for CANDU reactors is similar to that for light water reactors, in that a lattice physics code calculates the few-group homogenized cross-sections, while a diffusion code uses these cross-sections to solve a full-core model. A typical lattice cell code used for CANDU reactors is WIMS-AECL. However, one challenge with modelling the CANDU reactor is that control devices are perpendicular to the fuel channels, so must be modelled in 3D, by codes such as MULTICELL or DRAGON (Yoo et al., 2015, p. 2). Alternatively, a Monte Carlo code may be used to generate the group constants if it supports such capability, including the capability of performing a B_1 spectrum correction (2015, p. 2). One study showed that the two methods produced group constants generally within 2% of each other, along with good agreement in nodal calculations for k_{eff} and power distribution (2015, p. 6).

Light water reactor analysis typically uses assembly discontinuity factors (ADFs) which are ratios of the heterogeneous and homogeneous fluxes at each surface. These allow for flux discontinuities in the nodal solution – where the heterogeneous flux is continuous at an interface, if the ADFs are unequal, the homogeneous flux becomes non-continuous. However, CANDU reactor analysis using a finite difference method without discontinuity factors typically gives adequate results, so discontinuity factors can often be omitted (Kim et al., 2012, p. 1138). However, significant differences appear for control device insertion and for reflector cells, and a study shows that the use of ADFs can potentially improve the results of nodal calculations (2012, p. 1140). To account for the behaviour near the reflector, common practice in CANDU analyses is to use a multi-cell treatment wherein a 3 × 3 or larger lattice is solved using a deterministic code and suitable few-group cross sections for the lattice cells near the reflector are extracted.

2. Methodology

2.1. Polaris geometry and code changes

The current version of Polaris only has geometry support for LWRs. While it can model various features present in PWRs and BWRs, it cannot model more general configurations, such as the circular cluster lattice present for the CANDU bundle, or the circular pressure tube and calandria tube. This is not an intrinsic limitation of the ATLAS geometry package, but a limitation of the structure of the Polaris input (Oak Ridge National Laboratory, 2017).

Other SCALE codes use the SCALE Generalized Geometry Package, which is extremely flexible. Units can be modelled using quadratic surfaces, and embedded within one another using arrays or holes. This allows for almost any desired geometry to be modelled, which can be particularly important for shielding calculations. However, the analyst must model individual surfaces, or simple shapes such as cylinders, cuboids, spheres, and prisms, and construct the potentially complex model geometry as a combination of these simple surfaces. Subdivision of pin materials and geometry into rings must be done manually, though TRITON includes an aliasing feature to avoid having to repeat material definitions (Oak Ridge National Laboratory, 2017, p. 3–40).

Polaris takes a more constrained approach. Rather than making no assumptions in what the analyst might want to model, Polaris assumes that the analyst is modelling standard light water assembly designs. The input format is structured as such, specifying the assembly size and pitch, along with pin dimensions and pin mapping. Common features such as a channel box, water rods, or a cross (all commonly found in BWR assembly designs) may be modelled as well. Control assemblies (both the PWR rodlet-type design and the BWR blade design) may be modelled (Oak Ridge National Laboratory, 2017). Their insertion may be toggled when specifying branches for lattice physics. The meshing of pins into rings and sectors is streamlined, with both geometric and material subdivisions being automated.

Adding support for a new type of geometry, such as the CANDU lattice, required code upgrades in two areas. First, a new “cluster” geometry builder was added, which can take the input parameters describing a CANDU lattice (such as lattice pitch, pin arrangement, and tube radii), and construct an ATLAS geometry model. Several functions are shared with or similar to those from the existing lattice builder, and adapted to work with the new cluster builder. Geometry cells are tagged appropriately for Polaris functionality to work correctly, including automated material subdivision and pin power edits. Second, the input format was extended to allow the analyst to specify that they wish to model a CANDU lattice, and supply the input parameters. These parameters become inputs for the cluster geometry builder. Some goals for these changes included:

- Making the input format flexible without adding unnecessary complexity.

- Reusing and extending existing input cards where it makes sense, without affecting LWR input files.
- Supporting existing features if it is reasonable to do so.
- A reasonably efficient builder (comparable to the existing LWR builder).
- Supporting symmetry boundary conditions to speed computational time.

These changes were successfully implemented in the Polaris code, using SCALE version 6.2.3 as a starting point. The "cluster" geometry builder as described above was added alongside the existing "lattice builder", specializing in CANDU-type circular cluster lattices rather than square-pitched lattices. Input cards were added to specify the properties of this new geometry, including a "geom CLUSTER" card for the fuel bundle, and a "shell" card for the concentric circular surfaces making up the pressure tube and calandria tube. The system card received a "CANDU" option that specifies CANDU default properties for the different materials. Several new predefined materials were added, including heavy water for the coolant and moderator, Zr-2.5%Nb alloy for the pressure tube, and CO_2 for the gas gap between the pressure tube and the calandria tube. Existing cards, including the "pinmap" card, were extended to be compatible with the cluster geometry.

Several other changes or additions were performed to facilitate code testing or to evaluate their effects on the calculation. Self-shielded cross-section edits were added for data analysis. These edits may be implemented into the code in a more permanent manner in the future.

Polaris supports power iteration for k_∞ convergence, with optional CMFD acceleration (Jessee et al., 2014, p. 5). In Polaris, for LWR geometries, the CMFD mesh, which must be rectangular, is constructed automatically, using the assembly pitch along with the dimensions of structural materials surrounding the assembly, such as the channel box, narrow gap, and wide gap. Mesh cells need not be the same size. For CANDU geometries, there is no natural rectangular grid that fits the circular geometry, so the ability to specify the grid dimensions was included in the input. The grid is then overlaid on the geometry, with any mesh cells overlapping the gridlines being divided.

When CMFD acceleration was tested on the CANDU geometry, it was found to not provide a significant performance benefit, while reducing the robustness of the calculation, in that the code was more likely to fail prior to completing the calculation. This is in contrast to NEWT, where CMFD acceleration provides at least an order of magnitude acceleration in convergence, albeit with an even greater decrease in robustness. Therefore, CMFD acceleration was disabled in Polaris for the analysis, and only standard power iteration was used.

During the study, it was found that the cross-section for within-group scatter could exceed total elastic scatter for certain energy groups during Sampler runs, which is physically impossible. Both TRITON and Polaris simply discard the within-group scatter data and use values from the unshielded scattering probability matrix, but an inconsistency in the implementation caused Polaris to discard the data for all subsequent source groups as well, even for those with valid data. This portion of the subroutine in Polaris was corrected to be consistent with its implementation in TRITON. However, a fix for the underlying issue, stemming from the algorithm which perturbs and interpolates self-shielding factors in the ClarolPlus component of Sampler, is beyond the scope of this study, but has been reported to the developers.

The implementation of certain features not necessary for the verification and validation efforts of this study was deferred. The addition of reflector cell support, with a vacuum boundary condition, is not included in this paper, but will be included when the upgrades are merged into the official SCALE release.

There are still some limitations to CANDU analysis using Polaris, even with these extensions. Most notably, 3D geometry cannot be modelled, so control devices (which are perpendicular to the fuel channels in CANDU) cannot be included in the branch calculations. NEWT suffers the same limitation.

The model additions are not strictly limited to CANDU or similar heavy water reactor designs. Any circular lattice can be modeled using the new cluster geometry and shell card. This includes the Canadian Supercritical Water-Cooled Reactor (Yetisir et al., 2016), which uses light water coolant and heavy water moderator. The central flow tube in this design would be modelled as a large pin, conceptually similar to modelling a BWR water rod. The cluster geometry can also support single fuel channel modelling for an RBMK, which also uses circular assemblies, except with graphite instead of heavy water as the moderator (Barkauskas et al., 2017, p. 44). Due to the significant differences in fuel and moderator materials from a CANDU reactor, however, the physics of these reactors greatly differ from those of the CANDU, and the validation of Polaris to accurately model these reactor designs would require a separate analysis.

2.2. Verification and validation efforts

2.2.1. Lattice physics calculations

In order to validate the results calculated by Polaris, both intermediate calculations and final results were compared to existing codes. Intermediate calculation comparisons include qualitative examination of flux spectra, reaction rates and resonance impacts. Final results of a calculation include k_∞, lattice physics parameters (such as few-group cross-sections), coolant void reactivity (CVR), fuel temperature coefficient (FTC), and the results of the depletion calculation.

Several other codes were compared with Polaris in this study. The primary reference solution was generated using the Monte Carlo code KENO-VI in continuous energy mode. The continuous energy calculation does not require a separate self-shielding calculation, and it is possible to precisely calculate self-shielded cross-sections by tallying the flux and reaction rates in the multi-group structure. Also considered as a reference is the transport code NEWT with the CENTRM self-shielding methodology.

Various preliminary calculations were performed to investigate the capabilities and limitations of each code, in order to settle on a final set of codes with which to perform the comparative analysis. The set of codes and self-shielding methods selected are:

- Polaris, self-shielded using ESSM.
- KENO-VI, continuous-energy.
- KENO-VI, multi-group, self-shielded using CENTRM.
- NEWT, self-shielded using CENTRM.
- NEWT, self-shielded using BONAMI.

All cases, except for those utilizing Polaris, utilize the TRITON sequence.

Effects on the transport results arise both from direct differences in the transport calculation, and from differences in material composition from the depletion calculation. For some of the calculations, these effects were separated by running transport calculations on the compositions from the KENO-VI continuous energy transport/depletion calculation in order to separate the effects of depletion from the transport solution.

2.2.2. Sensitivity analysis of self-shielded cross-sections

It is also possible to look at the effect of the self-shielding calculation on its own, either by comparing two models with the same transport code but different self-shielding codes, or by directly looking at the difference in the self-shielded cross-sections. For the latter, the reference values were tallied using KENO-VI in continuous-energy mode. These were then compared to self-shielded cross-sections from CENTRM, BONAMI, and ESSM.

It is important to know which differences have the most significant effect on the transport solution – presenting the error in individual self-shielded cross-sections on their own is meaningless. While it is possible to consider the "raw" self-shielded cross-section biases for each nuclide

Table 2.1
Source of Self-Shielding Factors for Libraries Being Tested.

Library	$Z < 40$	$Z \geq 40$	Major Actinides
Default 252 g	NR Analytical	^{1}H Homogeneous	Pin Cell in H_2O
D_2O	NR Analytical	^{2}H Homogeneous	Pin Cell in D_2O
H_2O Homogeneous	NR Analytical	^{1}H Homogeneous	^{1}H Homogeneous
D_2O Homogeneous	NR Analytical	^{2}H Homogeneous	^{2}H Homogeneous

and energy group, these values are not particularly useful for the analysis. It is much more useful to consider the differences in self-shielded cross-sections as a perturbation, and calculate the effect of this perturbation on integral quantities such as k_∞ and reaction rates.

Since the reaction rate is the product of the cross-section and the flux, energy groups where both of these are large will be more important to self-shield accurately than other groups, and perturbations in these cross-sections will have a larger effect on the integral quantities. To determine which groups are most important, a sensitivity analysis is carried out using TSUNAMI to calculate the sensitivity of the integral quantities to the individual cross-sections, by nuclide and energy group. The quantities investigated were k_∞ along with various reaction rates for ^{235}U, ^{238}U, and ^{239}Pu. The bias for these quantities due to the self-shielding calculation can be estimated by multiplying these sensitivities by the relative error of each cross-section.

$$\frac{\Delta R_g}{R} = \frac{\sigma_{X,g} - \sigma_{X,g}^{(ref)}}{\sigma_{X,g}^{(ref)}} * S_{R,\sigma_{X,g}} \tag{2.1}$$

Eq. (2.1) is then summed over any desired set of nuclides, reactions, material regions, and energy groups to get the total bias for integral response R due to all cross-sections of that set.

TSUNAMI-3D was used to calculate the sensitivity of k_∞ using KENO-CE. Unlike the calculations described in Section 2.2.1, DBRC is enabled for the TSUNAMI-3D calculation, along with the KENO-CE calculation used to tally reference self-shielded cross-sections.

TSUNAMI-2D was used to calculate the sensitivities of both k_∞ and reaction rates using NEWT. The advantage of using TSUNAMI-2D is that the TSUNAMI-3D calculation is very slow (especially for reaction rates) and some sensitivities, such as to scattering cross-sections, do not have adequate statistical convergence without using an impractical number of histories.

All calculations were performed using one set of mid-burnup fuel compositions. The results presented in this paper, in Section 3.6, compare the self-shielded cross-sections from CENTRM and ESSM, $\sigma_{X,g}$, with the tallied cross-sections from KENO-CE, $\sigma_{X,g}^{(ref)}$. The relevant sensitivities $S_{R,\sigma_{X,g}}$ are obtained from the TSUNAMI-3D calculation when the integral response R is k_∞ and the reaction X is an absorption reaction (capture, fission, etc.). Otherwise, the TSUNAMI-2D calculation is used for the sensitivity. The effect on R is calculated using Eq. (2.1), and summed over all fuel materials, nuclides, and reactions. The results can then be plotted per unit lethargy for all energy groups, or summed over energy groups to give the total effect on R.

This method only captures the effect of self-shielding by the difference in the 1D cross-sections, and ignores the effect of differences in the 2D scattering matrices between the different self-shielding methods. TSUNAMI is limited to calculating sensitivities due to cross-section perturbations, and does not calculate sensitivities due to scattering matrix perturbations. Therefore, the effect of scattering matrix self-shielding is not calculated in this study.

2.3. Generation of self-shielding factors

The self-shielding factors that are distributed with SCALE are based on light water reactor characteristics (i.e. compositions, geometry, and flux spectrum). However, SCALE includes the IRFFACTOR code (Kim et al., 2019) that can be used to recalculate these self-shielding factors with a different model. The IRFFACTOR routines can be run using with a homogeneous or heterogeneous assumption. In homogeneous mode, the resonant nuclide is mixed with a background nuclide in different ratios, with a trace amount of a fissile nuclide added if necessary. The flux spectrum is then calculated using CENTRM, and the self-shielded cross-sections are calculated using PMC. In heterogeneous mode, the user constructs a set of CENTRM models spanning the desired range of background cross-sections. The background cross-sections are calculated using ESSM. The results are interpolated to overwrite the existing self-shielding factors in the library (Wiarda et al., 2016, p. 57).

Therefore, four libraries were used for the analysis – the distributed library and three modified libraries, using IRFFACTOR to replace the self-shielding factors in the distributed library. Table 2.1 details the difference between how the self-shielding factors are generated in each library. The "Major Actinides" category includes not just ^{235}U, ^{238}U, ^{239}Pu, ^{240}Pu, and ^{241}Pu, but also ^{91}Zr and ^{96}Zr. All libraries are based on ENDF/B-VII.1.

The mass lumping parameter "alump" was set to 0 for the D_2O pin cell, as the default causes significant issues when solving the pointwise spectrum for a D_2O pin cell.

2.4. Analysis case list and parameters

Table 2.2 summarizes the cases that were tested and compared for the analysis.

To make the modelling between the codes as consistent as possible, the following parameters were selected:

- No critical spectrum correction for NEWT or Polaris (as it is not supported in KENO).
- No Doppler Broadening Rejection Correction (DBRC) in KENO, as there is no equivalent in the deterministic codes.
- TRITON's default nuclide tracking set (addnux = 2) for depletion (Oak Ridge National Laboratory, 2017), to ease the computational burden for KENO calculations.
- IR approximation with no Bondarenko iterations in BONAMI, when performing full-range calculations without CENTRM. This is the default setting in ESSM.

As for the library generation, the CENTRM mass lumping parameter "alump" is set to zero.

In TRITON, the 37-element CANDU bundle is modelled using four depletion materials, based on the pitch of the centre of the pin from the centre of the bundle. One self-shielding calculation is performed for each depletion material. The automated material subdivision was disabled in Polaris to be consistent with this modelling. Geometric subdivision for the flux solution is still performed in both NEWT and Polaris.

The depletion comparison focused primarily on the major actinides ^{235}U and ^{239}Pu, tracking the depletion of ^{235}U and the production of ^{239}Pu in the fuel. All of the SCALE codes use ORIGEN for the depletion

Table 2.2
Case list for analysis.

Case	Transport Code	Self-Shielding Method	Nuclear Data Library
KENO-CE	KENO-VI		ENDF/B-VII.1 CE
KENO CENTRM	KENO-VI	CENTRM	Default 252 g
NEWT CENTRM	NEWT	CENTRM	Default 252 g
NEWT BONAMI	NEWT	BONAMI	Default 252 g
Polaris	Polaris MoC	ESSM	Default 252 g
Polaris D_2O	Polaris MoC	ESSM	D_2O
Polaris H_2O Homogeneous	Polaris MoC	ESSM	H_2O Homogeneous
Polaris D_2O Homogeneous	Polaris MoC	ESSM	D_2O Homogeneous

calculation. However, TRITON and Polaris have slightly different predictor-corrector schemes (Oak Ridge National Laboratory, 2017). After an initial bootstrap transport solution, TRITON solves the transport equation at the middle of each burnup step, using it to deplete the fuel for that burnup step (corrector) and extrapolating to the middle of the next step (predictor). Polaris solves the transport equation at the beginning of each step, depleting to the middle of the step (predictor), then solving again to deplete the entire step (corrector). With the default MATREX solver, this difference causes inconsistent solutions, especially for saturating fission products. Therefore, the CRAM solver was utilized instead, due to its improved consistency. For TRITON, this change had to be forced in the code itself, as it cannot be specified in the input file.

For deterministic codes, meshing and quadrature were selected so that further refinement would affect the results being studied by fractions of a percent or less. The default angular quadrature was used for both NEWT ($S_N = 6$, 6 directions per octant) and Polaris (20 azimuthal, 3 polar per octant).

For KENO, the number of histories varied based upon the analysis step. For depletion, fewer histories are required compared to LWR analysis as there are only four depletion materials. 10 million histories were selected to be used per depletion step, as this converges the uncertainty for saturating fission products to a small fraction of a percent, and the uncertainty for other nuclides to be negligible. This study considers this a reasonable compromise between computation time and precision. For reactivity calculations, 100 million histories were selected due to the tight convergence required on k_∞. 100 million histories were also selected for tallying reference collapsed cross-sections to compare with self-shielded cross-sections from the multi-group calculations. For the TSUNAMI-3D sensitivity calculations, 2.5 million histories were selected, as the calculation is computationally burdensome, and the sensitivities do not need to be tightly converged in order to identify which cross-sections are most significant. In addition, TSUNAMI-2D results were used where TSUNAMI-3D convergence was inadequate.

Typical parameters for Monte Carlo calculations include:

- 500 generations of 100,000 neutrons for the KENO-CE reference depletion calculation.
- 500 generations of 5000 neutrons for other depletion calculations.
- 1000 generations of 100,000 neutrons for reactivity calculations, to minimize the statistical uncertainty in order to get sufficiently converged results for reactivity coefficients.
- 1000 generations of 100,000 neutrons for the reference KENO-CE calculation to tally effective cross-sections for the 252 group structure.
- 500 generations of 5000 neutrons per generation for the TSUNAMI-3D sensitivity calculation.

Polaris supports automatic subdivision of materials by pin and ring, for both self-shielding and depletion. NEWT and KENO only allow for manual subdivision. To simplify the comparison, the automatic subdivision was disabled, and a subdivision scheme consistent with that used for NEWT and KENO was used. This results in the 37-element bundle being subdivided into four materials.

2.5. CANDU lattice model

Material and geometry parameters used in the lattice model are given in Tables 2.3 and 2.4, respectively.

As CANDU fuel pins have a collapsible cladding, there is no fuel-clad gap. The gas gap material refers to the space between the pressure tube and the calandria tube. The specification of the gas gap material is not important, as it has a negligible effect on the calculation. A material is only specified due to "void" material not being supported by the homogenization routine or by Polaris models. The gas gap material is not modelled for the TSUNAMI calculations and the cross-section calculations described in Section 2.2.2. The full geometry of the fuel channel is depicted in Fig. 2.1.

2.6. ZED-2 benchmark

In addition to investigating a CANDU lattice, a comparison using the ZED2-HWR-EXP-001 benchmark (Atfield, 2011) was performed. This benchmark consists of 28-element assemblies containing natural UO_2. The assemblies are inserted into fuel channels with similar geometry to CANDU, except vertically oriented. The level of D_2O moderator in the calandria is adjusted until criticality is achieved. Eight different configurations were measured, varying the configuration of voided and cooled channels.

By definition, k_{eff} for the real experiment is exactly 1 for all cases. However, the benchmark k_{eff} for the model is not exactly 1, due to uncertainties in the modelling and solution within the codes. These include geometric simplifications, model truncation, removal of less significant impurities, numerical uncertainties, and small differences in temperature. The biases due to each of these simplifications and uncertainties were estimated in the evaluation (Atfield, 2011, p. 75), to determine the benchmark k_{eff}, which is the value of k_{eff} that should be calculated for the model if the calculation and its underlying nuclear data are perfectly accurate. Nuclear data uncertainties and transport approximations will then introduce an additional bias between the calculated k_{eff} and the benchmark k_{eff}.

For this comparison, the geometry as described in the "simplified model" (Atfield, 2011, p. 48) was modelled in KENO, with material specifications copied from the MCNP sample input file. KENO was run for each configuration, in continuous-energy mode. By comparing the calculated k_{eff} to the benchmark, the suitability of KENO as a reference can be evaluated.

One difference between this analysis and the CANDU lattice model analysis, is that, for this benchmark analysis, continuous energy KENO is run with Doppler-broadening rejection correction (DBRC) enabled. At low temperature, the effect of DBRC should be small.

As the lattice physics codes (NEWT and Polaris) cannot directly model the full ZED-2 core, they model a single channel lattice instead. This lattice is also modelled in KENO. The model for this lattice is based on the WIMS-AECL lattice model included in the benchmark (Atfield, 2011, p. 92). By comparing the lattice code results to the KENO results, the lattice code results can be indirectly compared to the benchmark results.

Another limitation is that the current progress on the Polaris CANDU geometry model only allows for a square-pitched lattice, while the benchmark experiment consists of a hexagonal-pitched lattice. Therefore, in Polaris, the lattice was modelled as a volume-conserving square-pitched lattice. Both lattices were modelled in NEWT to determine the effect of changing the lattice geometry.

The full core KENO models were executed with 200 inactive generations, 800 active generations, and 60,000 neutrons per generation (48 million active histories). The channel models used the same number of histories as the full core models.

3. Results and discussion

3.1. Transport solution results for fresh fuel

Table 3.1 summarizes the lattice physics results for fresh fuel for each of the cases, including standard deviations for Monte Carlo

Table 2.3
Material Definitions for CANDU Model.

Material	Description	Density (g/cm^3)	Temperature (K)
Fuel	Natural UO_2 – SCALE default composition	10.3732	973.15
Clad	Zircaloy-4 – SCALE default composition	6.56	563.15
Coolant	D_2O – 99.75 wt% purity	0.804	563.15
Pressure Tube	Zirconium – 2.5 wt% niobium	6.57	563.15
Gas Gap	Carbon dioxide	0.0001786	341.15
Calandria Tube	Zircaloy-2 – SCALE default composition	6.56	341.15
Moderator	D_2O – 99.75 wt% purity	1.0858	341.15

Table 2.4
Geometry Definitions for CANDU Model.

Measurement	Element	Length (cm)
Fuel Pin Radii	Pellet O.R.	0.6122
	Cladding I.R.	0.6122
	Cladding O.R.	0.654
Fuel Ring Radii (Central Pin Centreline to Pin Centreline)	Ring 1	1.50
	Ring 2	2.90
	Ring 3	4.35
Pressure Tube Radius	Inner	5.1689
	Outer	5.6032
Calandria Tube Radius	Inner	6.4478
	Outer	6.5875
Channel Pitch		28.575

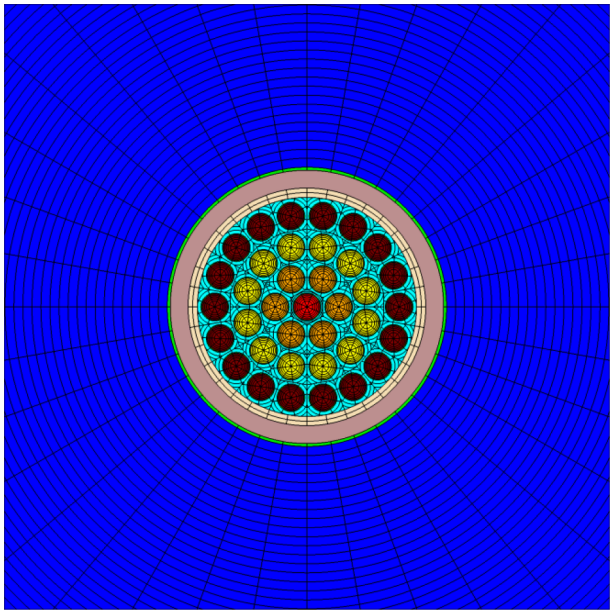

Fig. 2.1. Geometry of a CANDU channel lattice cell.

calculations. The code-to-code differences in initial k_∞ are well within one standard deviation of the nuclear data uncertainty (8.28 mk). The three cases with recalculated self-shielding factors predict k_∞ and fuel temperature coefficient (FTC) values closer to the KENO-CE reference. The CENTRM cases produce results comparable to the KENO-CE reference, but the BONAMI case significantly underpredicts the coolant void reactivity (CVR).

3.2. Depletion for a CANDU lattice

Fig. 3.1 shows the bias in the ^{235}U depletion trend. Both the CENTRM self-shielding method and the NEWT transport solver add a small positive bias over the continuous-energy KENO reference. The BONAMI self-shielding method adds a larger negative bias, but still only half of a standard deviation of the nuclear data uncertainty. All of the Polaris cases come closest to the continuous energy KENO case.

For ^{239}Pu, Fig. 3.2 shows that all of the cases come very close to the KENO-CE case, except for the BONAMI case, and, to a lesser extent, the default library Polaris case. All biases are less than one standard deviation of the nuclear data uncertainty.

For ^{135}Xe, a small bias in equilibrium concentration is observed. This bias is much smaller than the nuclear data uncertainty, as shown in Fig. 3.3. The reactivity effect of such a bias is on the order of 10 pcm, which is much smaller than fluctuations caused by refuelling, and much smaller than the liquid zone controller worth (UNENE, 2019, chap. 5).

For ^{149}Sm, the bias between KENO-CE and deterministic codes is more noticeable, as shown in Fig. 3.4. Polaris shows a larger bias than the other codes, but in all cases the bias is less than the nuclear data uncertainty. The fluctuations between depletion steps arise due to statistical uncertainties from the Monte Carlo methodology. Increasing the number of particle histories would reduce these fluctuations, however, the selected number of particle histories is sufficient to show the trends and compare the different codes.

In general, all three modified libraries result in smaller biases compared to the base Polaris case with the distributed library, and comparable biases to one another.

Table 3.1
Summary of Fresh Fuel Transport Solution Results.

Case	k_∞	ρ vs. KENO-CE (mk)	CVR (mk)	FTC (pcm/K)
KENO-CE	1.11718 ± 0.00006		16.08 ± 0.07	−1.09 ± 0.03
KENO CENTRM	1.11558 ± 0.00004	−1.29 ± 0.08	16.00 ± 0.05	−1.06 ± 0.02
NEWT CENTRM	1.11604	−0.91 ± 0.06	16.15	−1.07
NEWT BONAMI	1.12117	3.18 ± 0.06	15.11	−1.02
Polaris	1.11916	1.58 ± 0.06	16.05	−1.01
Polaris D_2O	1.11752	0.27 ± 0.06	15.86	−1.07
Polaris H_2O Homogeneous	1.11799	0.65 ± 0.06	16.03	−1.07
Polaris D_2O Homogeneous	1.11788	0.56 ± 0.06	16.04	−1.07

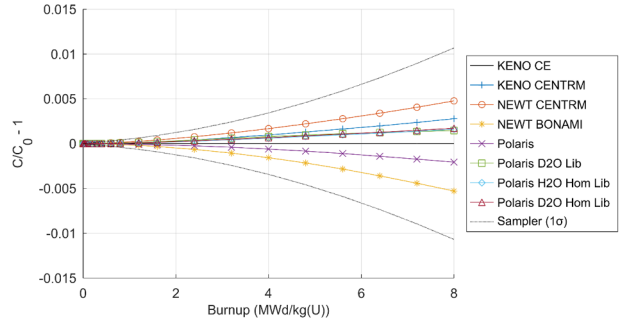

Fig. 3.1. Code Bias for Different Transport, Self-Shielding Calculations; Nuclear Data (^{235}U).

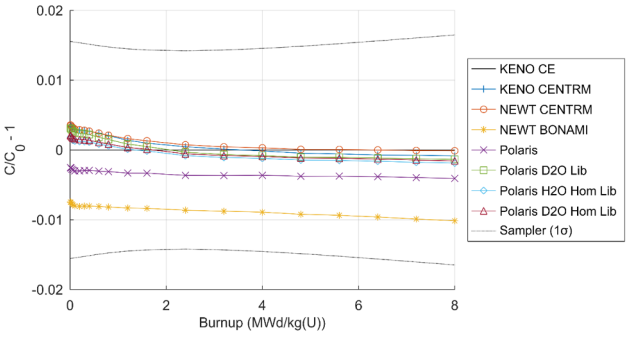

Fig. 3.2. Code Bias for Different Transport, Self-Shielding Calculations; Nuclear Data (^{239}Pu).

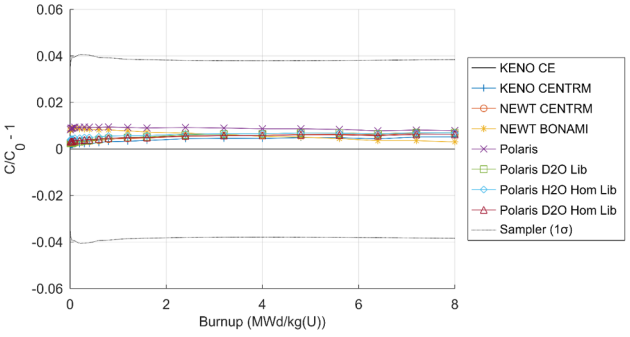

Fig. 3.3. Code Bias for Different Transport, Self-Shielding Calculations; Nuclear Data (^{135}Xe).

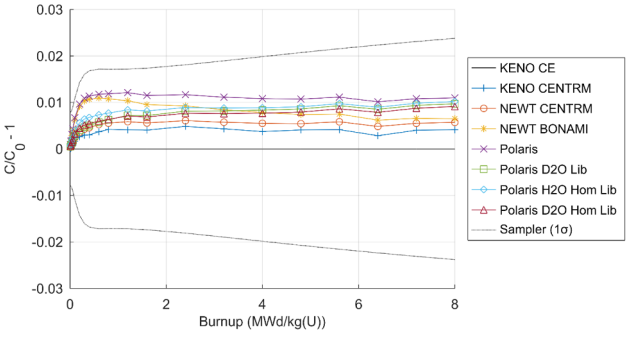

Fig. 3.4. Code Bias for Different Transport, Self-Shielding Calculations; Nuclear Data (^{149}Sm).

3.3. Reactivity of a CANDU lattice

Fig. 3.5 shows the effect of the transport and depletion calculation on k_∞. The CENTRM-based calculations give a k_∞ very close to the reference KENO-CE calculation. The BONAMI calculation starts with a

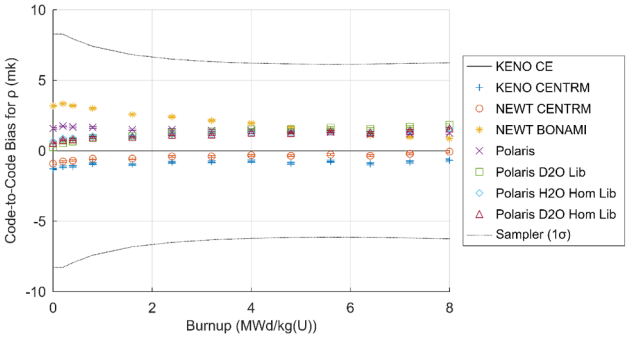

Fig. 3.5. Excess reactivity comparison (nominal conditions).

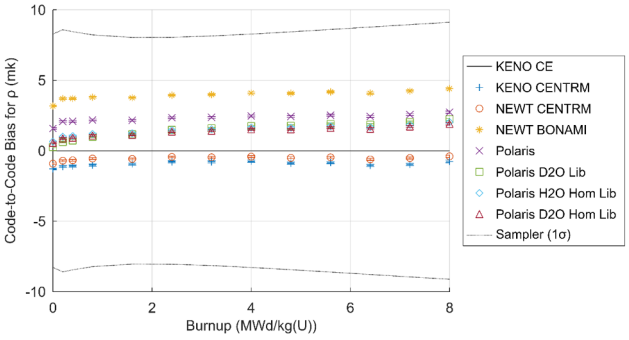

Fig. 3.6. Excess reactivity comparison – reference depletion calculation (KENO-CE).

+3 mk reactivity bias, which decreases as the fuel is depleted. The Polaris calculations start with a small reactivity bias, with the modified library cases coming very close to the KENO-CE reference. However, the Polaris k_∞ results converge together for depleted fuel.

Fig. 3.6 isolates the effect of the transport solution by using the reference depleted compositions from KENO-CE for all calculations. There is little difference for the CENTRM results, while the BONAMI and Polaris results have a slightly higher bias for depleted fuel compared to fresh fuel. The modified library cases no longer fully converge onto the distributed library case, showing that much of this convergence was due to differences in the depleted compositions. It should be noted that the trend in k_∞ is much more important for lattice physics, as well as for uncertainty propagation, than the absolute value of k_∞. The practical effect of a small k_∞ bias is a small change in average core burnup, and there is little impact on safety analysis. The codes are observed to follow very similar trends.

3.4. Reactivity coefficients for a CANDU lattice

Figs. 3.7 and 3.8 show the trend of lattice coolant void reactivity (CVR) and fuel temperature coefficient (FTC) with depletion, including

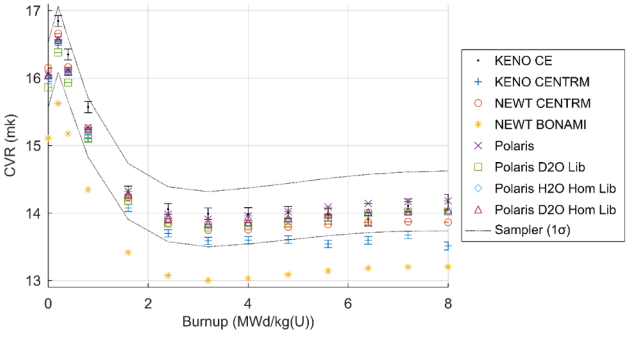

Fig. 3.7. Trend of Coolant Void Reactivity for a CANDU Lattice.

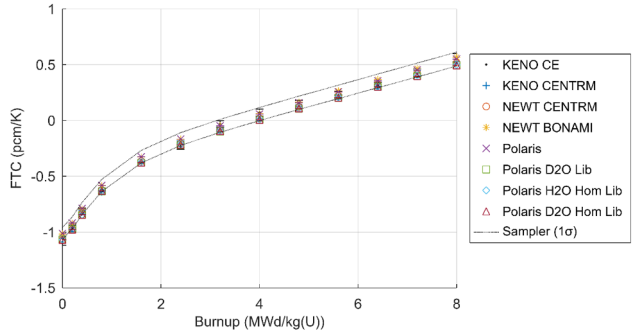

Fig. 3.8. Trend of Fuel Temperature Coefficient for a CANDU Lattice.

Table 3.2
Difference in Group Constants between NEWT and Polaris (time-average).

Quantity	NEWT	Polaris	Rel. Error (%)	Sampler σ (%)
$\nu\Sigma_{f2}(\times 10^{-3}\ \mathrm{cm}^{-1})$	4.500	4.525	0.55	0.78
$\nu\Sigma_{f1}(\times 10^{-3}\ \mathrm{cm}^{-1})$	0.8548	0.8555	0.09	2.78
$\Sigma_{a2}(\times 10^{-3}\ \mathrm{cm}^{-1})$	3.893	3.909	0.40	0.77
$\Sigma_{a1}(\times 10^{-3}\ \mathrm{cm}^{-1})$	1.783	1.776	−0.40	1.35
$\Sigma_{tr2}(\times 10^{-3}\ \mathrm{cm}^{-1})$	387.0	386.9	−0.03	1.46
$\Sigma_{tr1}(\times 10^{-3}\ \mathrm{cm}^{-1})$	243.7	242.9	−0.31	1.14
$\Sigma_{1\to2}(\times 10^{-3}\ \mathrm{cm}^{-1})$	8.669	8.677	0.09	1.67
$\Sigma_{2\to1}(\times 10^{-3}\ \mathrm{cm}^{-1})$	0.06750	0.06782	0.47	2.00
ϕ_2/ϕ_1	2.192	2.195	−0.35	2.07

one-sigma error bounds for the nuclear data uncertainty on the Polaris case. Most of the cases make similar predictions for the CVR, though multigroup KENO predicts noticeably lower values compared to continuous energy KENO, while NEWT with BONAMI significantly underpredicts CVR compared to the reference calculation. All cases predict the same trend for FTC, with little variation.

3.5. Group constants

Table 3.2 shows the difference between results calculated with Polaris (using ESSM) versus NEWT (using CENTRM), as KENO does not calculate group constants. The biases between the codes are rather small. Additionally, the trends in group constants with depletion are observed to be similar between the different codes. The relative error tends to be nearly constant with respect to burnup.

Table 3.3 compares Polaris to NEWT for the different sets of self-shielding factors used for Polaris calculations. All three modified libraries give comparable results, being closer to NEWT for thermal cross-sections, but further from NEWT for fast group absorption.

When looking at the results with respect to burnup, $\nu\Sigma_{f2}$ shows a significant positive bias for the base Polaris case for low burnups, and a smaller positive bias for the modified libraries. This is partly due to the critical spectrum correction being disabled – the same comparison performed with the critical spectrum correction enabled shows reduced biases for $\nu\Sigma_{f2}$.

Table 3.4
k_∞ Reaction Bias due to Self-Shielded Cross-Sections.

Library/Method	Bias Thermal (pcm)	Bias Inter. (pcm)	Bias Fast (pcm)	Bias Total (pcm)
CENTRM	+13.73	−28.58	(+23.02)	−14.85
Default 252 g	+202.72	+3.02	+0.47	+206.21
D_2O	+49.38	−9.01	+0.57	+40.95
H_2O Homogeneous	+34.86	+3.06	+1.49	+39.40
D_2O Homogeneous	+24.52	+1.11	+0.57	+26.21

3.6. Effect of self-shielding on k_∞ and reaction rate biases

The differences between KENO-CE, CENTRM, and ESSM are most noticeable for k_∞ (Table 3.4) and for the ^{238}U(n,γ) reaction rate (Table 3.5). The reaction rate bias is calculated for thermal (below 0.625 eV), intermediate, and fast (above 20 keV) energies, as well as for the total. The fast-range bias is ignored in the total for CENTRM due to the default being to use BONAMI for fast energies, which shows a negligible bias for fast energies. It should be noted that the bias is calculated for each of the 252 energy groups, so the tabulated results are a summation over multiple energy groups, all nuclides in the fuel, and all reactions.

Fig. 3.9 shows the per-group effect of the self-shielded cross-sections on k_∞, while Fig. 3.10 shows the same for the ^{238}U radiative capture reaction (relative to total fission). The values in Tables 3.4 and 3.5 are obtained by integrating these plots.

For intermediate energies, the differences in self-shielded cross-sections appear as small fluctuations with many energy groups having close to zero bias, but with some groups, with small widths, contributing a large bias per unit lethargy, either positive or negative. For ^{238}U in particular, CENTRM performs better, as the magnitude of these biases is smaller. When summed over the entire energy range, though, ESSM appears to have a smaller bias as the contributions offset and cancel out more completely.

For thermal energies, the biases are more significant, appear in "wider" groups, and tend to span several energy groups. These are much more significant for ESSM than they are for CENTRM. With ESSM, the biases can be reduced with the D_2O F-Factors library or with the homogeneous self-shielding factors.

3.7. ZED-2 benchmark

Table 3.6 shows that KENO gives results comparable to the benchmark results, within a few tenths of a percent. The bias shows an additional −1 mk trend from the voided case to the cooled case, which suggests a calculation-to-experiment bias of approximately +1 mk for the coolant void reactivity. Most of the previous calculations in the benchmark paper show a similar trend (Atfield, 2011). The actual coolant void reactivity bias may vary as the two configurations also differ in leakage (due to changes in the critical moderator height).

Table 3.7 shows the coolant void reactivity for a single channel of the model in different codes. The KENO-CE and Polaris results are clustered around 15 mk, while the CENTRM-based results are clustered around 14.6 mk. One notable exception is that the Polaris calculation

Table 3.3
Relative errors from self-shielding factors versus NEWT (time-average).

Library	Original		D_2O Heterogeneous		H_2O Homogeneous		D_2O Homogeneous	
Cross-Section	Value	Error (%)	Value	Error (%)	Value	Error (%)	Value	Error (%)
$\nu\Sigma_{f2}$	4.525	0.55	4.518	0.40	4.517	0.38	4.515	0.35
Σ_{a2}	3.909	0.40	3.906	0.32	3.906	0.35	3.905	0.31
Σ_{a1}	1.776	−0.40	1.770	−0.75	1.768	−0.83	1.768	−0.81

All values are in units of $\times 10^{-3}\ \mathrm{cm}^{-1}$.

Table 3.5
$^{238}U(n,\gamma)$ Reaction Bias due to Self-Shielded Cross-Sections.

Library/Method	Bias Thermal (%)	Bias Inter. (%)	Bias Fast (%)	Bias Total (%)
CENTRM	−0.0180%	+0.1155%	(+0.0923%)	+0.0975%
Default 252 g	−1.0073%	−0.0200%	−0.0015%	−1.0287%
D_2O	−0.3167%	+0.0311%	−0.0018%	−0.2874%
H_2O Homogeneous	−0.3180%	+0.0009%	−0.0045%	−0.3215%
D_2O Homogeneous	−0.2001%	+0.0058%	−0.0018%	−0.1961%

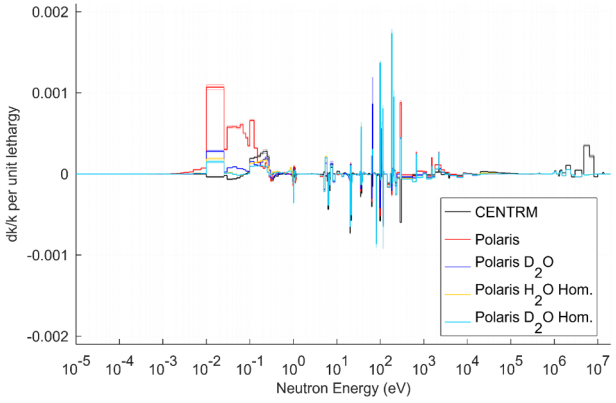

Fig. 3.9. k_∞ Bias Per Unit Lethargy from Self-Shielded Cross-Sections.

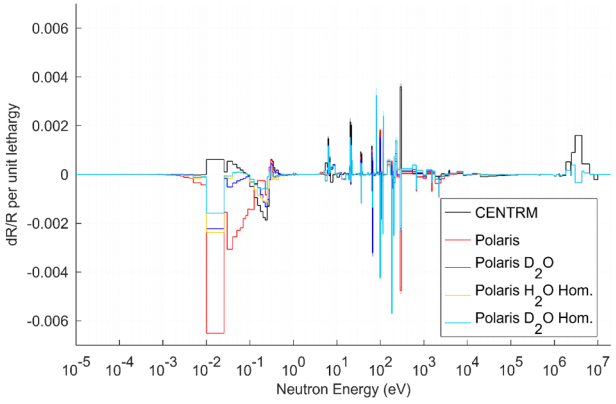

Fig. 3.10. $^{238}U(n,\gamma)$ Reaction Rate Bias Per Unit Lethargy from Self-Shielded Cross-Sections.

using the D_2O heterogeneous self-shielding library gives a result about 1 mk lower than the other Polaris results. One possible reason for part of this difference is that the D_2O library is based on self-shielding of heavy water pin cells, which is the same as for the CENTRM cases. The rest of the bias is likely to come from the choice of pin cell models for generating self-shielding factors for different background cross-sections. While the bias for this case matches up with the observed calculation to experiment bias in KENO, this should only be taken as a coincidence. The effect of the channel boundary condition (square or hexagonal) has a negligible effect when the volume is conserved.

Table 3.7
Code-to-Code Comparison of Benchmark Lattice Cell.

Code	Model (Boundary)	Library	CVR (mk)
KENO	Hexagon	ENDF/B-VII.1 CE	15.08 ± 0.11
KENO	Square	ENDF/B-VII.1 CE	15.07 ± 0.10
KENO	Hexagon	ENDF/B-VII.1 252 g	14.50 ± 0.07
NEWT	Hexagon	ENDF/B-VII.1 252 g	14.61
NEWT	Square	ENDF/B-VII.1 252 g	14.64
Polaris	Square	ENDF/B-VII.1 252 g	15.14
Polaris	Square	D_2O Heterogeneous	14.08
Polaris	Square	H_2O Homogeneous	14.93
Polaris	Square	D_2O Homogeneous	14.94

Additionally, the group constants were compared for the NEWT case and the Polaris case, with similar results as for the CANDU lattice, where the differences between the codes were smaller than the uncertainties present due to nuclear data.

3.8. Preliminary assessment of performance

A formal assessment of the performance of Polaris, compared to TRITON/NEWT, was not carried out. NEWT, with CMFD acceleration, was observed to converge several times more quickly than Polaris, while ESSM is several times quicker than CENTRM. However, the performance is highly sensitive to the meshing of the geometry as well as the angular quadrature. The default quadrature was used in both codes; Polaris solves 10 times as many angles as NEWT. Using the default Polaris quadrature in NEWT results in a calculation more than an order of magnitude slower, such that Polaris becomes the better-performing code. In addition, NEWT CMFD acceleration is unreliable for CANDU models, especially for depletion calculations, and disabling CMFD acceleration results in very slow convergence.

Conversely, the Polaris meshing, ray spacing, or quadrature can be optimized to minimize the calculation time for a given accuracy requirement. Since the ESSM self-shielding calculation is fast, significant

Table 3.6
Comparison of KENO-CE Core Calculation to Benchmark Experiment.

Configuration	Voided Channels	H_c (cm)	Benchmark k_{eff}	Calculation k_{eff}	(C − E)/E
1	55	191.002	1.0028 ± 0.0028	0.99887 ± 0.00011	−0.39% ± 0.28%
2	54	191.619	1.0028 ± 0.0028	0.99905 ± 0.00010	−0.37% ± 0.28%
3	52	192.724	1.0028 ± 0.0029	0.99902 ± 0.00010	−0.38% ± 0.29%
4	50	193.829	1.0029 ± 0.0029	0.99910 ± 0.00011	−0.38% ± 0.29%
5	48	195.002	1.0028 ± 0.0029	0.99903 ± 0.00011	−0.38% ± 0.29%
6	36	200.450	1.0031 ± 0.0030	0.99877 ± 0.00010	−0.43% ± 0.30%
7	18	205.929	1.0031 ± 0.0030	0.99861 ± 0.00011	−0.45% ± 0.30%
8	0	209.088	1.0033 ± 0.0031	0.99849 ± 0.00011	−0.48% ± 0.31%

speed improvements are possible, whereas CENTRM calculations are often the bottleneck when using NEWT.

4. Conclusions

Overall, this research shows that Polaris can be extended to model the CANDU lattice, and produce reasonable calculations which are comparable to those from other SCALE sequences such as TRITON. Most of the code-to-code biases evaluated in this study are much smaller than the uncertainties from the nuclear data, suggesting that any additional uncertainty due to the choice of code should be small. While the methods used by Polaris for resonance self-shielding are less rigorous than those used by CENTRM, they are adequate for the purpose, and the performance advantage justifies considering its use. However, further verification and validation needs to be carried out before the code can be considered ready for CANDU safety analysis.

The results of the study show that, even with the distributed SCALE library with the original light water self-shielding factors, the results from ESSM are reasonable with the biases on results such as k_∞ and depletion trajectories being insignificant relative to the estimated uncertainties stemming from the nuclear data. However, the CANDU-specific self-shielding factors, as well as homogeneous-based self-shielding factors, can provide marginal improvement. The homogeneous-based self-shielding factors, for both light and heavy water, despite being less equivalent to the actual CANDU model than pin cells, perform better overall compared to the originally distributed self-shielding factors. The D_2O pin cell f-factors perform adequately when appropriate parameters are set, particularly setting alump = 0 when using D_2O as a moderator.

Overall, the code-to-code differences in k_∞ arise from a combination of the differences in the transport calculation itself and the differences in the calculated depleted compositions. When the composition difference is removed, the code-to-code biases in k_∞ are nearly constant with respect to burnup. The burnup-dependence for homogenized constants, when looking at the relative error between NEWT and Polaris, is also nearly constant in most cases, even with the effect of differing depleted compositions, with $\nu\Sigma_{f2}$ being an exception.

The choice of reference pin cell models for different background cross-sections can have a significant effect on the results. A noticeable bias in CVR from the KENO model was observed for the ZED-2 benchmark, when using the D_2O pin cell f-factor library. Future evaluations should consider the parameters for the IRFFACTOR calculation, along with considering the background cross-sections for CANDU fuel nuclides under different conditions when developing pin cell cases. In contrast, the homogeneous libraries have only a few parameters, and very little that could affect the result of the calculation of self-shielding factors.

CRediT authorship contribution statement

Simon Younan: Methodology, Software, Validation, Formal analysis, Investigation, Data curation, Writing - original draft, Writing - review & editing, Visualization, Project administration. **David Novog:** Conceptualization, Writing - review & editing, Supervision, Project administration, Funding acquisition.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The authors would like to acknowledge the SCALE development team, and in particular Matthew Jessee and William Wieselquist, for their development of the SCALE package, including Polaris, and their support of my research. The authors would also like to acknowledge Kang Seog Kim and Doro Wiarda for support and guidance on AMPX and IRFFACTOR.

Funding: This work was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) and the University Network of Excellence in Nuclear Engineering (UNENE).

References

Atfield, J.E., 2011. 28-Element Natural UO2 Fuel Assemblies in ZED-2.

Barkauskas, V., Plukienė, R., Plukis, A., Remeikis, V., 2017. Generation of RBMK-1500 spent nuclear fuel one-group cross-section libraries and their evaluation against experimental data. Lith. J. Phys. 57, 42–53. https://doi.org/10.3952/physics.v57i1.3455.

Gauld, I.C., Mertyurek, U., 2019. Validation of BWR spent nuclear fuel isotopic predictions with applications to burnup credit. Nucl. Eng. Des. 345, 110–124. https://doi.org/10.1016/j.nucengdes.2019.01.026.

Jessee, M.A., Wieselquist, W.A., Evans, T.M., Hamilton, S.P., Jarrell, J.J., Kim, K.S., Lefebvre, J.P., Lefebvre, R.A., Mertyurek, U., Thompson, A.B., Williams, M.L., 2014. Polaris : a new two-dimensional lattice physics analysis capability for the scale code system. Physor 14.

Jessee, M.A., Wieselquist, W.A., Gentry, C.A., Mertyurek, U., 2017. BWR geometry enhancements for the Polaris lattice physics code. Trans. Am. Nucl. Soc. 1301–1305.

Kim, K.S., Williams, M.L., Wiarda, D., Clarno, K.T., 2019. Development of the multigroup cross section library for the CASL neutronics simulator MPACT: method and procedure. Ann. Nucl. Energy 133, 46–58. https://doi.org/10.1016/j.anucene.2019.05.010.

Kim, Y., Kim, W.S., Hartanto, D., Cho, B.H., 2012. Effectiveness of the nodal equivalence theory in CANDU reactor. In: Transactions of the American Nuclear Society. San Diego, pp. 1138–1140.

Knott, Dave, Yamamoto, Akio, 2010. In: Handbook of Nuclear Engineering. Springer US, Boston, MA, pp. 913–1239. https://doi.org/10.1007/978-0-387-98149-9_9.

Labarile, A., Miró, R., Barrachina, T., Verdú, G., 2015. TRITON vs POLARIS. Comparison Between Two Modules for LWRs Modelling in SCALE 6.2. In: 24th International Conference Nuclear Energy for New Europe, Portorož, pp. 1–10.

Li, J., Cao, L., Zu, T., Wu, H., He, Q., 2017. Resonance self-shielding treatment for fully ceramic micro-encapsulated fuels with the embedded self-shielding method. In: Proceedings of the 2017 25th International Conference on Nuclear Engineering, Shanghai.

Oak Ridge National Laboratory, 2017. SCALE Code System, Oak Ridge.

UNENE, 2019. The Essential CANDU - A Textbook on the CANDU Nuclear Power Plant Technology. University Network of Excellence in Nuclear Engineering (UNENE), Hamilton.

Wiarda, D., Dunn, M.E., Greene, N.M., Williams, M.L., Celik, C., Petrie, L.M., 2016. AMPX-6: A Modular Code System for Processing ENDF/B Evaluations, Oak Ridge.

Williams, M.L., 2011. Resonance self-shielding methodologies in SCALE 6. Nucl. Technol. 174, 149–168. https://doi.org/10.13182/NT09-104.

Williams, M.L., Kim, K., 2012. The embedded self-shielding method. In: 2012 Advances in Reactor Physics – Linking Research, Industry, and Education (PHYSOR 2012), Knoxville.

Yetisir, M., Gaudet, M., Pencer, J., McDonald, M., Rhodes, D., Hamilton, H., Leung, L., 2016. Canadian supercritical water-cooled reactor core concept and safety features. CNL Nucl. Rev. 5, 189–202. https://doi.org/10.12943/CNR.2016.00042.

Yoo, S.Y., Shim, H.J., Kim, C.H., 2015. Monte Carlo few-group constant generation for CANDU 6 core analysis. Sci. Technol. Nucl. Install. 2015. https://doi.org/10.1155/2015/284642.

5 Paper 2

5.1 Publication Details

S. Younan, D. Novog, "Development and Testing of TRACE/PARCS ECI Capability for Modelling CANDU Reactors with Reactor Regulating System Response," *Science and Technology of Nuclear Installations*, vol. 2022. doi: 10.1155/2022/7500629

The code modifications to TRACE and PARCS, along with the development of a Python interface to the ECI library, were performed by the first author (Simon Younan). The development of an RRS model in Python, coupled to TRACE and PARCS using the ECI, was also performed by the first author. The TRACE and PARCS CANDU models used in this work were models developed by Kai Groves (CANDU-6 model) and Michael Tucker (900 MW CANDU model) for prior studies. The RD-14M TRACE model was initially developed by Anatol Mysen and modified by David Hummel for prior studies. All of these models were further modified by the first author to be adapted to the built-in coupling capability and to the transients being analyzed in this work. The research supervisor (Dr. David Novog) was responsible for research conceptualization, project administration, and obtaining code licenses. Both authors worked together to select test cases to analyze. The paper was written by the primary author, while Dr. David Novog provided support in reviewing and editing the paper.

5.2 Preface

A nuclear power station consists of many different interacting systems and physics. This includes the neutronics of the reactor itself, along with the system thermalhydraulics and heat transport, mechanical and thermal performance of the fuel, containment modelling, and various control systems for regulating the physical systems. Generally, different codes are developed for each type of physics, but due to the interactions, the codes must be able to share data on the simulation to each other. In other words, the codes must be coupled together.

Models may either be loosely or tightly coupled together. With loose coupling, the state of each model is shared with the other models to act as model inputs, but the solution methods themselves remain independent. Data is exchanged at set intervals and the models can either run sequentially or concurrently. This type of coupling can either be done externally with a script controlling the individual codes that simulate the models

or can be done internally by the codes themselves if designed or modified to do so. With tight coupling, the solution methods themselves are also coupled, usually allowing for better convergence, but codes must be designed to have their solution methods integrated.

The USNRC system thermalhydraulics code TRACE and the core neutronics code PARCS are designed to be coupled together, with both codes being distributed as a single executable. Together, they can be used for the analysis of a LWR or CANDU reactor. The two codes are loosely coupled; PARCS data transfer and execution occur at the end of each TRACE time step. Additionally, TRACE includes the Exterior Communications Interface, and the ECI library is distributed with the code. The ECI provides an interface for other codes to be loosely or tightly coupled into TRACE.

Prior work in Dr. Novog's research group using the PARCS and TRACE codes involved coupling the two codes externally using scripts. This was done as there were a number of limitations with the built-in coupling, including being unable to dynamically manipulate reactivity devices. These works include the Master's thesis of Kai Groves, as well as some core follow work performed by Michael Tucker. While these works could be performed using external coupling, there was interest in taking advantage of the built-in coupling capabilities of the two codes.

There are several motivations in using a built-in coupling capability rather than using external scripts. The first is that a built-in coupling capability should receive official support and validation from the code maintainers, thus only the minor modifications and additions to this capability need to be verified and validated. It also avoids the need to develop a coupling procedure in the first place. Secondly, it avoids the need for file-based I/O which creates a potential performance bottleneck. Finally, it allows for potential performance improvements as the simulation runs in a single execution, avoiding the need for repeated code and model re-initialization.

For this work, source code access was obtained and the built-in coupling capabilities were investigated. It was found that dynamic reactivity device coupling was implemented but that minor code changes were required to make it functional. The models from the prior works of the research group were modified to function using the built-in coupling. In addition, the actual control of the reactivity devices still required separate code from TRACE and PARCS, and the ECI was used to couple the reactivity device control to TRACE. It was decided to implement an interface between the ECI library and Python, to permit development of Python code that can be coupled to TRACE using the ECI, as such an interface has applications beyond the scope of this work.

5.3 Summary of Contributions

The source code for TRACE version 5.1262 and PARCS version 3.31 was provided through a group license to the research group. This work is part of a larger project to incorporate the TRACE and PARCS codes into CANDU safety analysis, using the ECI in TRACE to couple additional codes and models. Both TRACE and the ECI were developed by the USNRC, while PARCS was developed by a research group at Purdue University and distributed as both a standalone version and a version integrated with TRACE. I performed minor source code changes to PARCS and TRACE for the purpose of facilitating code coupling as needed for the work.

Also provided was the ECI library for developing new ECI-enabled code modules. This library is developed primary in FORTRAN 90 and is intended to interface with FORTRAN 90 programs. I developed modifications to this library and added an interface that uses the F2PY tool (part of NumPy) to compile into a Python-accessible package, and I developed additional object-oriented Python modules to access this interface, creating a package (referred to as PyECI in this thesis) that allows for Python programs to interface with TRACE through the ECI. This package allows for wider access to using the ECI due to the ease of developing Python programs and scripts, and the scope for PyECI stretches far beyond its specific uses in this thesis. The main limitation of PyECI is the relatively poor performance of Python when compared to FORTRAN 90, making it unsuitable for computationally intensive tasks unless these tasks can be offloaded to libraries suitable for high-performance work (e.g., NumPy). Thus, PyECI is best suited for small scripts or models that need to be coupled with the TRACE solution.

I also developed a model of the key components of the CANDU RRS in Python, using PyECI to couple the model to TRACE and PARCS. This model is based primarily on publicly available knowledge. The test cases for both the TRACE-PARCS coupling and ECI coupling were discussed with the research supervisor. Two sets of TRACE-PARCS CANDU models were acquired, one being the basis of Kai Groves's M.A.Sc. thesis [25], and the other being the basis of Michael Tucker's ongoing thesis work. Using the built-in TRACE-PARCS coupling methodology, I adapted Groves's work and reproduced a portion of his results. In addition, I adapted Tucker's model to use the built-in TRACE-PARCS coupling and RRS Python script to simulate the transients investigated by this paper. This paper also adapted an existing RD-14M model, where I modified the model to perform analysis of flow oscillation transients.

The writing of the journal paper, including the presentation of the results, was my own work, with editorial support from both Dr. Novog and from the peer review process. The peer review process also led to some modifications and additions to the methodology and corresponding adjustment of results.

Hindawi
Science and Technology of Nuclear Installations
Volume 2022, Article ID 7500629, 31 pages
https://doi.org/10.1155/2022/7500629

Research Article

Development and Testing of TRACE/PARCS ECI Capability for Modelling CANDU Reactors with Reactor Regulating System Response

Simon Younan and David R. Novog

Department of Engineering Physics, McMaster University, Hamilton L8S 4L8, Canada

Correspondence should be addressed to Simon Younan; younans@mcmaster.ca

Received 28 June 2021; Revised 14 December 2021; Accepted 29 January 2022; Published 27 March 2022

Academic Editor: Guglielmo Lomonaco

The use of the USNRC codes TRACE and PARCS has been considered for the coupled safety analysis of CANDU reactors. A key element of CANDU simulations is the interactions between thermal-hydraulic and physic phenomena with the CANDU reactor regulating system (RRS). To date, no or limited development has taken place in TRACE-PARCS in this area. In this work, the system thermal-hydraulic code TRACE_Mac1.0 is natively coupled with the core physic code PARCS_Mac1.0, and RRS control is implemented via the exterior communications interface (ECI) in TRACE. ECI is used for coupling the external codes to TRACE, including additional physical models and control system models. In this work, a Python interface to the TRACE ECI library is developed, along with an RRS model written in Python. This coupling was tested using a CANDU-6 IAEA code coupling benchmark and a 900 MW CANDU model for various transients. For the CANDU-6 benchmark, the transients did not include RRS response, however, the TRACE_Mac1.0/PARCS_Mac1.0 coupling and ECI script functionality was compared to the previous benchmark simulations, which utilized external coupling. For the 900 MW CANDU simulations, all aspects of the ECI module and RRS were included. The results from the CANDU-6 benchmark when using the built-in coupling are comparable to those previously achieved using external coupling between the two codes with coupled simulations taking 2*x* to 3*x* less execution time. The 900 MW CANDU simulations successfully demonstrate the RRS functionality for the loss of flow events, and the coupled solutions demonstrate adequate performance for figure-of-eight flow instability modeling.

1. Introduction

In nuclear power plants, including CANDU stations, many phenomena arise because of coupled interactions between the reactor physic (nuclear chain reaction) and system thermal-hydraulic phenomena. These range from simple reactivity feedbacks to 3D spatial power changes to coupled flow instabilities. For this reason, many core physic and system thermal-hydraulic codes include coupling capabilities. This study focuses on the USNRC-developed codes, PARCS (for core physics), and TRACE (for system thermal-hydraulics) [1, 2]. While these codes are designed for the safety analysis of light water reactors, they have capabilities for the analysis of other reactor types, including CANDU reactors. For example, TRACE and PARCS were used to model the IAEA ICSP benchmark problem, including an uncertainty analysis [3].

In addition to coupled phenomena, it is also important to consider the effect of the reactor's control systems on transient reactor response. In TRACE, these are typically modeled using signal variables and control blocks. Signal variables can read process parameters, while control blocks perform analog and logical calculations. Both can be used to actuate devices, such as valves and pumps. The coupled TRACE-PARCS code then allows the TRACE control the blocks to alter reactivity device configurations in PARCS and mimic the response of a reactor control system. This reactivity device coupling was introduced in PARCS 2.7 [4] and used in a previous study for pressurized water reactor (PWR) analysis [5,6] but is not officially documented in the code's user manuals.

Currently, most work in CANDU safety analysis is performed using codes developed specifically for CANDU

reactors. While these codes operate under the same principles as more general-purpose safety analysis codes, they include models and correlations that are better suited for CANDU reactors specifically. The examples of codes developed and used for CANDU design and analysis include RFSP for core physics and reactor regulation [7], along with CATHENA and TUF for system thermal-hydraulics [8,9]. The Canadian Nuclear Safety Commission (CNSC) has been evaluating the use of various independent codes for CANDU safety analysis, particularly TRACE, motivating various studies on its applicability [10], including this study.

Most previous studies involving the implementation of CANDU-specific control systems embed the reactor regulating system (RRS) capabilities into the source code. One study of a CANDU loss of flow transient using RELAP5 added station-specific controller routines to the source code [11]. Alternatively, RRS emulators can be developed within an outside environment and included as part of external coupling methodologies. It was done in a prior study using TRACE and PARCS to simulate multiple CANDU transients [3]. In such an arrangement, an external script is developed, which executes each coupled code independently over short periods of time and exchanges information at the end of each time step. During this information exchange, the response of the reactor regulating system can be determined and used to alter the associated input files to include control device changes prior to initiating the next time step. While the former methodology allows for fast execution times and avoids the I/O bottlenecks in the second method, it is inflexible compared to an external RRS emulator. Ideally, a solution involving the close coupling of the thermal-hydraulic and physic phenomena with an external RRS emulator would be the ideal solution. It would provide the flexibility of RRS development within its own shell while still maintaining the computational advantages. Using TRACE-PARCS with the exterior communications interface (ECI) module provides one avenue to achieve these goals.

TRACE includes a general coupling capability with the exterior communications interface, which allows TRACE to be coupled with any code that uses the ECI library [12]. It can be used for additional physical models (e.g., fuel performance models, subchannel models, CFD models) and for detailed control systems that would be difficult to implement using only TRACE control blocks. This study focuses on the implementation of RRS response via the ECI interface.

This study develops a model for much of the CANDU RRS and tests the coupling framework and RRS model on two existing coupled PARCS-TRACE models (one CANDU-6 and one 900 MW class CANDU). This work converts the models to utilize the built-in PARCS-TRACE coupling and integrate the other systems using ECI capability with the goal of streamlining the execution of the models while still obtaining accurate results.

While one of the goals of ECI is to avoid the source code changes to TRACE, some changes were nevertheless required. This work utilizes the modified versions of TRACE V5.1262 and PARCS V3.31, hereafter referred to as TRACE_Mac1.0 and PARCS_Mac1.0, respectively, wherever the results and discussion are specific to the modified codes. The changes are relatively minor, primarily to facilitate code coupling for this work's applications, and they do not add or modify any physical models.

2. Materials and Methods

2.1. PARCS-TRACE Coupling Background. The PARCS core physic code is included within the distribution of TRACE. The two codes are compiled into a single executable.

The PARCS and TRACE data structures are not linked directly but are linked by an internal "general interface." PARCS and TRACE access shared data transfer tables rather than directly accessing each other's data structures. Upon initialization, PARCS sets up the data transfer tables based on the data in the MAPTAB file. The MAPTAB file is a user-created file that specifies how the nodes of a PARCS model are to be coupled with TRACE components [13].

TRACE and PARCS are coupled explicitly as the two solvers run independently. At the end of each TRACE time step, thermal-hydraulic data is passed to PARCS, and a step is performed in PARCS—converting the flux in steady state mode or advancing time in transient mode. The updated core power data is passed back to TRACE for the next time step. The accuracy of this coupling is equivalent to externally controlled coupling for a given data transfer frequency. This accuracy has been evaluated in a previous study using the external coupling methodology [3]. However, there is a performance advantage in performing the data transfers in-memory, keeping both codes initialized in memory, and running in lockstep, making it practical to perform data transfer on every TRACE time step. In addition, PARCS has the capability to skip TRACE time steps based on user input as oftentimes it is unnecessary to update the neutronic model on every thermal-hydraulic time step.

To model CANDU reactivity devices, this work utilized an undocumented ability to couple PARCS control rod banks to TRACE signal variables. Rather than physical control rods in a light water reactor (LWR), the models in this work mimic the liquid zone devices and adjusters of a CANDU [14]. It is done using the **%CRSIG** card [4] in the MAPTAB file, followed by an arbitrary number of signal variables to rod bank pairs. On each time step, the values of signal variables are mapped to the position values of the corresponding rod banks. The PARCS upper and lower limit values are not utilized. Thus, the appropriate conversion must be incorporated on the TRACE/ECI side if rod banks have different amounts of travel, which is the case for CANDU reactivity devices.

Therefore, there are three ways to manipulate or control banks during a PARCS transient, which are as follows:

(1) Specified bank movement table that updates bank positions as a function of time. This function runswithin PARCS without interaction from TRACE (other than TRACE determining the time step size).

(2) Specified SCRAM bank movement table that drops in all banks or a subset of banks. The SCRAM may be triggered within PARCS by relative power and/or be

triggered by TRACE when the PARCS input specifies a TRACE trip signal.

(3) Fully coupled to TRACE signal variables using the CRSIG card.

This research focuses on method 3, which is the only method that can dynamically react to the rest of the system in a generalized fashion.

2.2. Exterior Communication Interface (ECI) Overview. The exterior communications interface is a library of Fortran and C subroutines that permits the coupling of other codes to TRACE without modifying the TRACE source code or executable [12]. It is also capable of coupling multiple TRACE submodels together to parallelize the simulation of a larger model. Using ECI, through TRACE, allows one to directly access and control reactivity device configurations within PARCS.

The two main components of the ECI are as follows [12]:

(1) The ECI library, which is embedded within TRACE. A separate copy of the library is included so that programmers may embed it in their own programs.

(2) The ECI driver, which is a standalone Java program that must run in the background, is responsible for starting child "satellite" processes and setting up the actual interprocess communication through sockets or shared memory.

The ECI coupling model is "request-driven" [12]. ECI defines 18 synchronization points at which the parallel processes exchange data, corresponding to different points in the TRACE program flow, as shown in Figure 1. Each program specifies its requests during the initialization of the simulation. Each request identifies the variable to be coupled, along with a synchronization point and the direction of data transfer. Multiple requests may exist for the same variable. ECI then locates all of the requested variables and constructs the transfer tables. During execution, the central process (usually TRACE) is responsible for time step control and status monitoring, while satellite processes can request a smaller time-step size or report on their convergence status.

PARCS itself does not have ECI support. Thus, ECI programs can only interact with PARCS indirectly through TRACE. It is done by transferring data from signal variables, fluid components, or heat structures in TRACE. It limits the manipulation of PARCS to control banks by the manipulation of TRACE control blocks and signal variables. In Figure 1, PARCS initialization and PARCS execution occur after **Input** and before **EndStep**, respectively.

The ECI library, distributed with TRACE, is the set of codes required to create an ECI-compatible program. It is written primarily in Fortran 90 and designed to work with Fortran 90 programs by including the library in the program's source code (rather than as a dynamic or static library). While most of the modules can be used as-is, two modules are templates that must be completed by the programmer, SpecExTrans and TimeEvolve [12]. The former allows the program to locate variables requested by other processes, while the latter contains the required program flow and synchronization points.

An ECI satellite program will typically retrieve initial conditions at **Init** and old-time values at **OldTime**, as shown in Figure 1. Values can be sent and new-time values retrieved wherever appropriate.

A complete ECI program, therefore, requires the following:

(1) The ECI library distributed with TRACE

(2) A completed version of the SpecExTrans module if its variables are to be visible to other processes

(3) The TimeEvolve subroutine that includes the synchronization points and program flow from Figure 1 and all of the program's own computations

(4) A subroutine that sets up all of the data requests that the program requires

(5) A "main" function or a subroutine that performs all of the necessary ECI subroutines, including reading command-line arguments, preparing the data transfers, and calling TimeEvolve

Specific implementation details are omitted for brevity. They may be found in the ECI manual [12]. The program structure is summarized in Figure 2.

2.3. The Python ECI Package and RRS Module. This work modified and adapted the Fortran ECI library to produce a Python interface, compiling the modified ECI library using the F2PY program [15], creating a compiled Fortran library that can be imported by a Python program or module. It allows for the creation of ECI-compatible Python programs. This package contains items 1 and 2 as outlined above and allows for the development of items 3–5 as a Python program.

The actual package consists of the following components:

(1) The distributed ECI library with modifications to improve compatibility with Python, such as improved handling of command line arguments being passed through Python.

(2) A Fortran module named PyInterface, which includes the subroutines to be exposed to Python using F2PY. These subroutines call the ECI library's internal subroutines. This interface also includes an allocatable array, which is used to facilitate data transfers.

(3) An object-oriented Python module is named __init__.py. It introduces an object named VariableData, which contains and manages ECI-linked variables. This object manages the Fortran allocatable array along with the data transfer requests.

A Python program using this package will follow the same program flow as an ECI-enabled Fortran program described in the previous section and in Figure 2. In summary, such a program will set up all of its data requests using a VariableData object. Then, it executes a Python

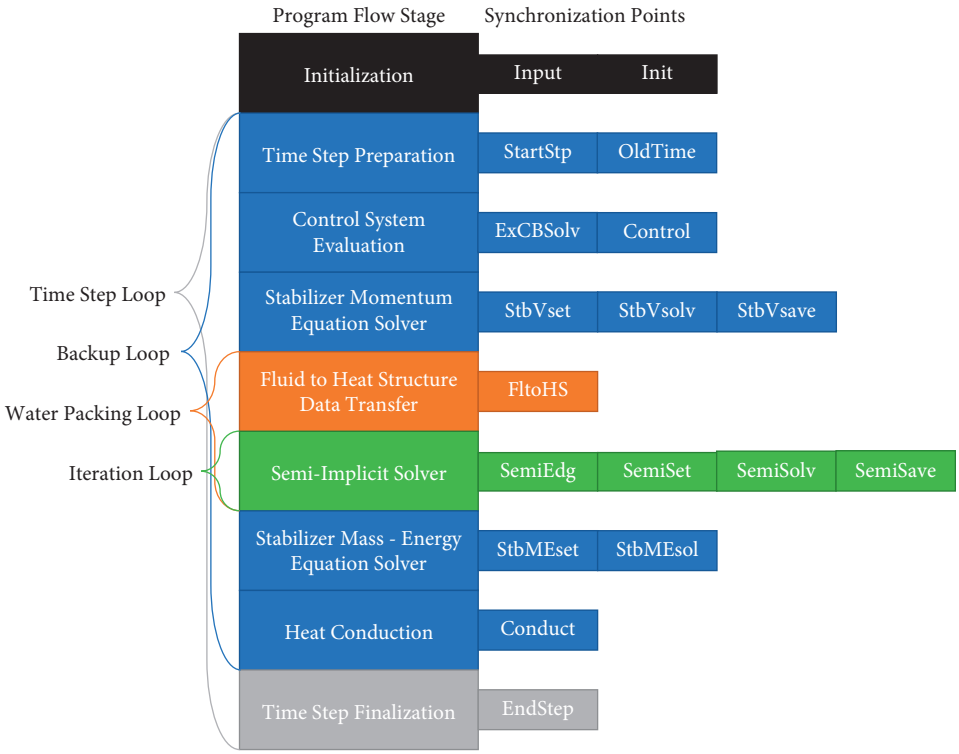

Figure 1: TRACE program flow structure with synchronization points [12].

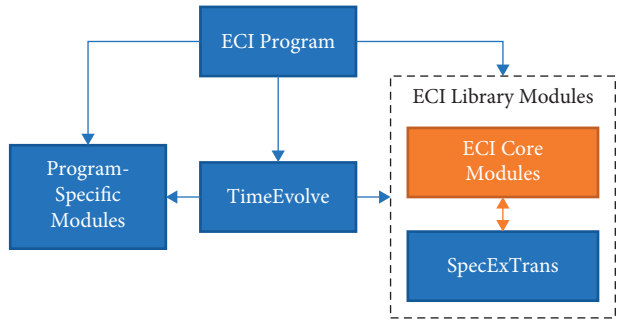

Figure 2: High-level structure of ECI program. From the entry point, the program must call ECI library subroutines to initialize ECI coupling, including scheduling data transfers. The program must then call the TimeEvolve subroutine, which is responsible for synchronization and all program-specific computations.

version of TimeEvolve, following the same time step structure shown in Figure 1. Each requested component variable, corresponding to a TRACE component, is accessed using a Variable object. This object-oriented approach adds a layer of abstraction between the program and the underlying Fortran array as the program can simply call "get" and "set" methods to retrieve or change the data. The full implementation is summarized in Figure 3, showing how each component of the program is linked and how data is passed through ECI and through the Python ECI package.

The ECI library also includes a number of global variables used for time step control, program flow control, and error reporting. Unlike component variables, these global variables are directly exposed by the interface created by F2PY and can be accessed directly by the program or through the ECI package.

The ECI package is used to couple a reactor regulating system (RRS) model to the reactor model, along with a model for shutdown system 1 (SDS1) trips. The program is written in a modular fashion so that each RRS module (e.g., the liquid zone control) and each set of physical devices (e.g., the actual liquid zone compartments) are represented by individual Python objects. These Python objects are linked to ECI variables along with each other. Figure 4 outlines the RRS module, showing the implemented components in blue and the ECI-linked input variables in orange.

During each time step, each control system function is modeled. Physical devices advance their state on each time step, while control modules update on a fixed interval (0.5 seconds for most RRS modules, and 2.0 seconds for liquid zone spatial control) to match the discrete time step nature of the digital RRS update frequency in CANDU reactors.

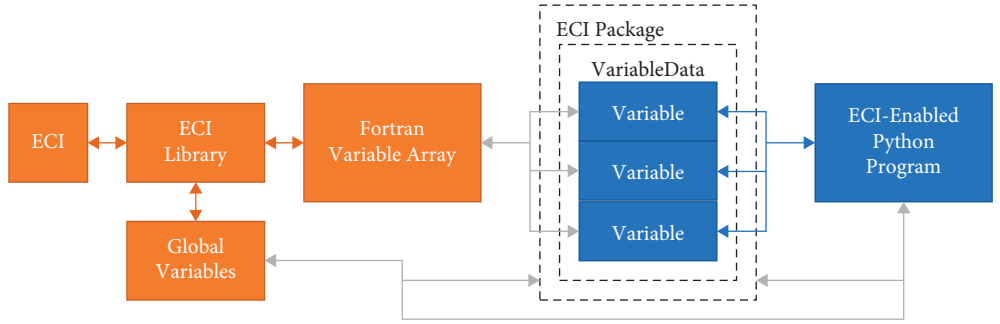

Figure 3: General ECI program hierarchy with Python interface. The ECI package and Python program are written in Python, while the ECI library and Fortran variable array are written in Fortran 90. Component variables are linked through a variable object. Global variables used for time step and program flow control are exposed directly by the interface.

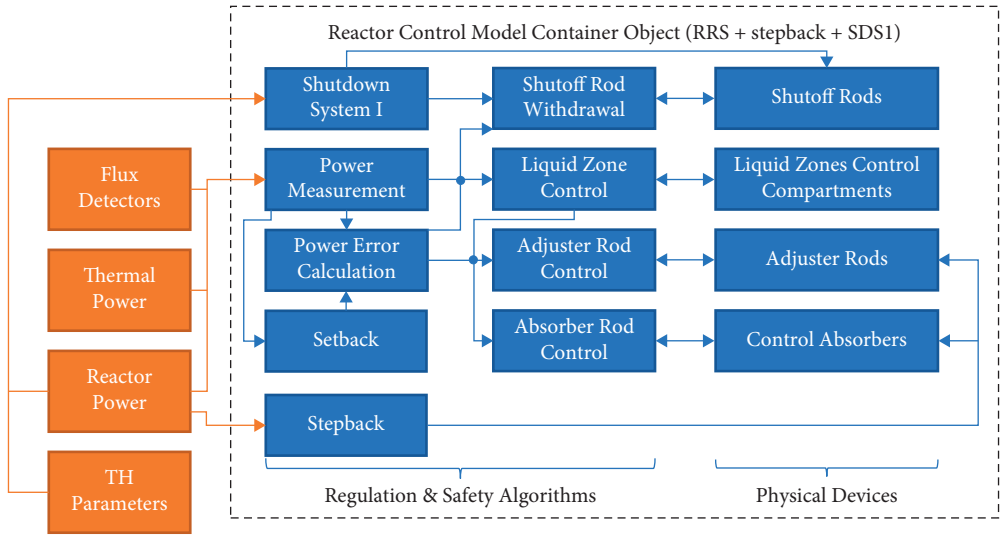

Figure 4: Reactor control model as implemented in this work. An overview of how different components are connected is shown along with the TRACE and PARCS model parameters that provide input to the control model.

SDS1 trips are also checked on every time step (because the SDS systems can actuate at any instant and are not subjected to the 0.5 s discrete RRS response limits). Much of the implementation is based on the description in [16].

Both backup and restart capabilities are implemented. If TRACE requests a time step back-up, the states of the RRS modules are reverted to their old-time values. Restart capability is implemented by writing the RRS state to a JSON-formatted file at the end of the simulation and reading the state at the start of a restarted run.

The following simplifications were made in the current version of the model:

(i) No flux mapping routine (FLU) or fully instrumented channel (FINCH) mapping is included for spatial flux reconstruction. Instead, zonal powers are determined directly from the relevant PARCS nodes, equivalent to every channel being a FINCH with no measurement lag time.

(ii) CANDU Setback is an RRS function that gradually reduces reactor powers and can be triggered from multiple signals, including from neutronic or process parameters. In this work, only neutronic setbacks are considered. Furthermore, setback on high local flux is not implemented as its true implementation relies on FLU [16]. Therefore, setback is only implemented on high zonal flux and high flux tilt.

(iii) Stepback in a CANDU is an RRS response to a large detected perturbation and causes a large reduction in rector power over a short interval of time. Stepback triggers were implemented for neutronics parameters only, along with a manual stepback to 60% (to simulate a turbine trip Stepback).

(iv) SDS1 trip triggers were implemented only for high neutron power, high log rate, low inlet feeder flow, and high heat transport system (HTS) pressure. The low flow trip uses the total flow rather than individual channel flows since the TRACE model aggregates the channels in groups of 60.

(v) The platinum detectors are treated as perfect detectors that only respond to the thermal neutron flux with no delayed components.

(vi) The ion chambers are not modeled, and the reactor power and log-neutron-rate signals from PARCS are used in its place. The PARCS reactor power is also used for the SDS1 high power trip.

Several additional Python programs are coupled with ECI, which are as follows:

(1) A power calculator that reads the nodal powers and calculates channel and zone powers before passing them back to TRACE. It makes these powers visible to ECI, SNAP, and AptPlot. ECI is utilized here as the summation blocks built into TRACE currently use an inefficient implementation for data retrieval.

(2) A data collection process that reads specified ECI variables every time step and outputs them to a tab-delimited data file for analysis.

(3) A profiler that serves no purpose, except to measure the time spent waiting at each synchronization point. This data can be used to optimize the model's runtime by revealing which steps take up the most CPU time.

2.4. Coupling of PARCS and ECI to TRACE. For modeling a CANDU reactor, the PARCS and TRACE models were coupled using a MAPTAB file created to map the PARCS neutronic nodes with the TRACE fluid and heat structure cells. Each fuel channel is mapped to the corresponding representative thermal-hydraulic channel. Channel powers are summed to get the total average channel power for each group of 60 channels, while fuel and coolant properties are shared among all channels for a given representative channel. Both models use the same axial division (one bundle length per node/cell) so the axial mapping is 1 : 1.

Coupling can be specified using either of the two sets of cards:

(1) VOLRMAP cards: used for the automatic mapping of the PARCS mesh to the TRACE VESSEL or CHAN component. It is applicable to LWR analysis.

(2) TABLE cards: used for the manual mapping of PARCS mesh cells to TRACE fluid and HTSTR components. For CANDU analysis, it is necessary to use the TABLE cards.

As the TABLE cards must specify the mapping for each individual PARCS mesh cell, a script was written to automate the writing of these cards based on the mapping in the specification, accounting for the flow direction of each flow pass.

ECI programs are coupled by adding a "task list" file. This file contains a list of all the processes to run along with command-line arguments. When TRACE is run, it will read the task list and start all of the necessary processes.

While the use of ECI coupling reduces the need to modify the original source codes, certain changes were required in this work. As mentioned in the introduction, the codes with these modifications are referred to as TRACE_Mac1.0 and PARCS_Mac1.0, respectively.

(i) One signal variable parameter had two conflicting data validation checks when used with the %CRSIG card. As the parameter was otherwise unused for this type of signal variable, the %CRSIG-specific validation check was removed.

(ii) Signal variable type 119 (PARCS cell power) was found to not function. Additional code was implemented to make the signal variable functional.

(iii) A new signal variable type was added to retrieve the PARCS thermal flux.

(iv) The POWER component was modified so that, in a coupled PARCS_Mac1.0-TRACE_Mac1.0 model, noncoupled POWER components (i.e., those not mapped to PARCS) will function as they would in a standalone TRACE model.

(v) The SpecExTrans ECI module in TRACE was modified to permit access to signal variable data rather than requiring indirect access through control blocks to reduce data propagation delays in the coupling scheme.

(vi) The size of the ECI transfer buffer was increased to accommodate the transfers of larger arrays, such as individual bundle powers.

(vii) An error that could periodically occur when calculating D_2O properties was modified to force a time step backup instead of terminating the simulation.

2.5. The CANDU-6 IAEA Code Coupling Benchmark. The IAEA "Numerical Benchmarks for Multiphysics Simulation of Pressurized Heavy Water Reactor Transients" [17] is a set of standardized tests for the coupled simulation of postulated pressurized heavy water reactor (PHWR) transients. The purpose of the benchmarks is to provide test problems that may be implemented in different physic and thermal-hydraulic codes. The following events are simulated on a stylized CANDU-6 model as part of the benchmark:

(1) Steady state: to establish initial conditions prior to a transient, the coupled model is run until convergence. The converged model can then be used as a starting point for performing transient analysis. A null transient is used to test if the initial conditions are sufficiently converged. The fully coupled TRACE-PARCS model is compared with other codes using internal or external coupling to determine the agreement in the converged steady-state solution.

(2) Adjuster absorber rod withdrawal: in this loss of regulation (LOR) transient, adjuster rods 7 and 14 are withdrawn at a rate of 10 cm/s. The simulation is run for 25 seconds. There is no credit for SDS. The key result is the set of channels that exceeds its respective critical channel power [17].

(3) Coolant pump rundown: in this loss of flow (LOF) transient, heat transport pump #2 begins a rundown

with a specified pump speed profile. The simulation is run for 25 seconds. There is no credit for SDS. The key result is the set of channels that exceeds its respective critical channel power [17].

(4) Inlet header break: in this loss of coolant accident (LOCA), there is a break in the inlet header 2 of 0.0645 m^2 over the first 0.1 seconds of the transient, with an external pressure of 1 atm. SDS-1 is triggered at 120% of nominal core power, and the rod drop follows a specified profile. The simulation is run for 5 seconds. The key result is the maximum bundle enthalpy [17].

The reactor regulating system (RRS) is not credited for any of these transients, and thus, it does not need to be simulated in this model. The adjuster rod withdrawal can be carried out using the MOVE_BANK card in PARCS.

The benchmark is not described in full detail here, however, a summary of key features follows. The CANDU-6 design has 380 fuel channels and a two-loop heat transport system, with each loop serving half the core (left and right halves). Each loop has two flow passes in opposite directions. Adjacent channels have opposite flow directions, forming a checkerboard pattern as shown in Figure 5. Each flow pass has its own inlet and outlet headers. In the thermal-hydraulic model, only the main circuit, pressurizer, and some loop and pass interconnects are modelled. Each flow pass is represented by seven thermal-hydraulic channels, for a total of 28 thermal-hydraulic channels for the core, as shown in Figure 6. The boilers have representative flow resistance on the primary side, however, the secondary side is modeled as a boundary condition with a constant heat transfer coefficient and temperature. The pressurizer is also simplified, being represented by a large pipe. A boundary condition of saturated liquid at 10 MPa is connected to the pressurizer to set initial conditions in the steady state model.

In both neutronic and thermal-hydraulic models, each fuel channel is axially divided into 12 nodes of one bundle length each. A radial reflector is present around the core with a width of slightly more than 2 channel pitches. There is no axial reflection.

For the core neutronic model, the nominal thermal power is 2000 MW. The model is highly simplified, and most structural materials are not modeled. The burnup distribution provided in the benchmark includes a significant flux tilt. The control devices modeled include 21 adjuster rods, 28 shutoff rods, and 14 liquid zone controllers. Adjuster rods start inserted at their nominal position, liquid zone controllers start 50% filled, and shutoff rods start outside the core. The adjuster rods and liquid zone controller positions are not changed during steady state or transient analysis, except if specified by the specific transient. The benchmark specifications included a full set of branch structures and a reduced set, with most participants electing to use the reduced set. For this work, the set of cross-sections generated with SCALE, available in Ref. [3], was primarily used. The transients were also modeled with the benchmark cross-sections that utilize a reduced branch structure for comparison purposes.

The initial model for this work was developed in Ref. [3] based on the specification for the code coupling benchmark. In Ref. [3], external coupling via scripts was used for TRACE and PARCS coupling. For the transients in Ref. [3], each code is run for 0.1 seconds of simulation time per step, until the end of the transient simulation is reached. It is a leap-frogging calculation with data exchange occurring at a coarse time step (0.1 seconds), while each individual code can divide the coarse time step into multiple fine time steps as needed.

In this work, the model was modified to fully utilize TRACE-PARCS coupling and ECI coupling, and while RRS was not utilized, the ECI scripts were still tested for functionality. This coupled mode permits a tighter degree of coupling, with data exchange occurring at every TRACE time step. Since data exchange occurs entirely in memory, file management is greatly simplified, as files do not have to be generated for every step of data exchange.

2.6. The 900 MW CANDU Model and Simulations. The 900 MW CANDU model is based on work performed in Ref. [18] related to station blackout transients and was converted from RELAP5 to TRACE. In addition to the 480 fuel channels, different reactor powers, and loop flows, several other notable changes were made compared to the CANDU 6 model discussed above. Compared to the CANDU-6 benchmark, this model includes pressurizer level control using feed and bleed, pressure control using pressurizer heaters, and heat transfer by the pressure tube and calandria tube, and it models the secondary side of the steam generators, including pressure and level control. However, only eight representative fuel channels are modeled, two per core pass. Unlike the CANDU-6 model, the 900 MW CANDU model is not based on a specific benchmark. Thus, the comparison of the model results to prior work is more qualitative in nature.

Figure 7 shows the nodalization of the neutronic calculation and its mapping to the thermal-hydraulic model in Figure 8, showing how the eight representative fuel channels map to the 480 actual fuel channels. Axially, both models have fuel channels that are divided into 12 nodes of one bundle length each, with no axial reflector regions in the neutronic model. On the primary side, shown in Figure 8, the fluid boundary conditions are the feed and bleed flows along with the pressurizer steam bleed. The steam generator models are shown in Figure 9. The primary components of each steam generator, as shown in Figure 10, are the upper and lower boiler regions, the preheater, the downcomer, and the steam drum. The details of steam separation are not implemented in this model. Instead, the steam drum is modeled as a single large node, modeling the steam generator liquid inventory. Only dry steam is permitted to flow into the steam dome node, while only liquid is permitted to flow into the downcomer.

The reactor physic model includes liquid zone controllers, adjuster rods, mechanical control absorbers, and shutoff rods. All devices are in their normal positions during a steady state calculation, except for the liquid zone

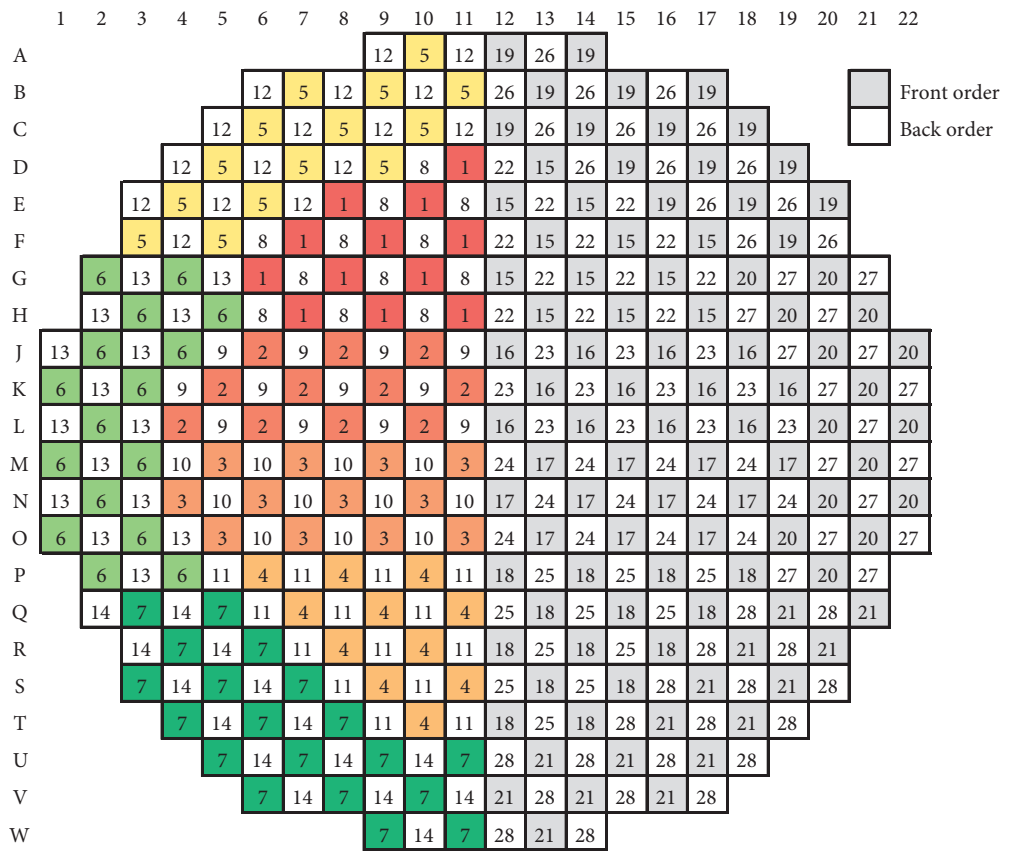

Figure 5: Neutronic radial channel nodalization and mapping to thermal-hydraulic channels (reflector not shown) [3].

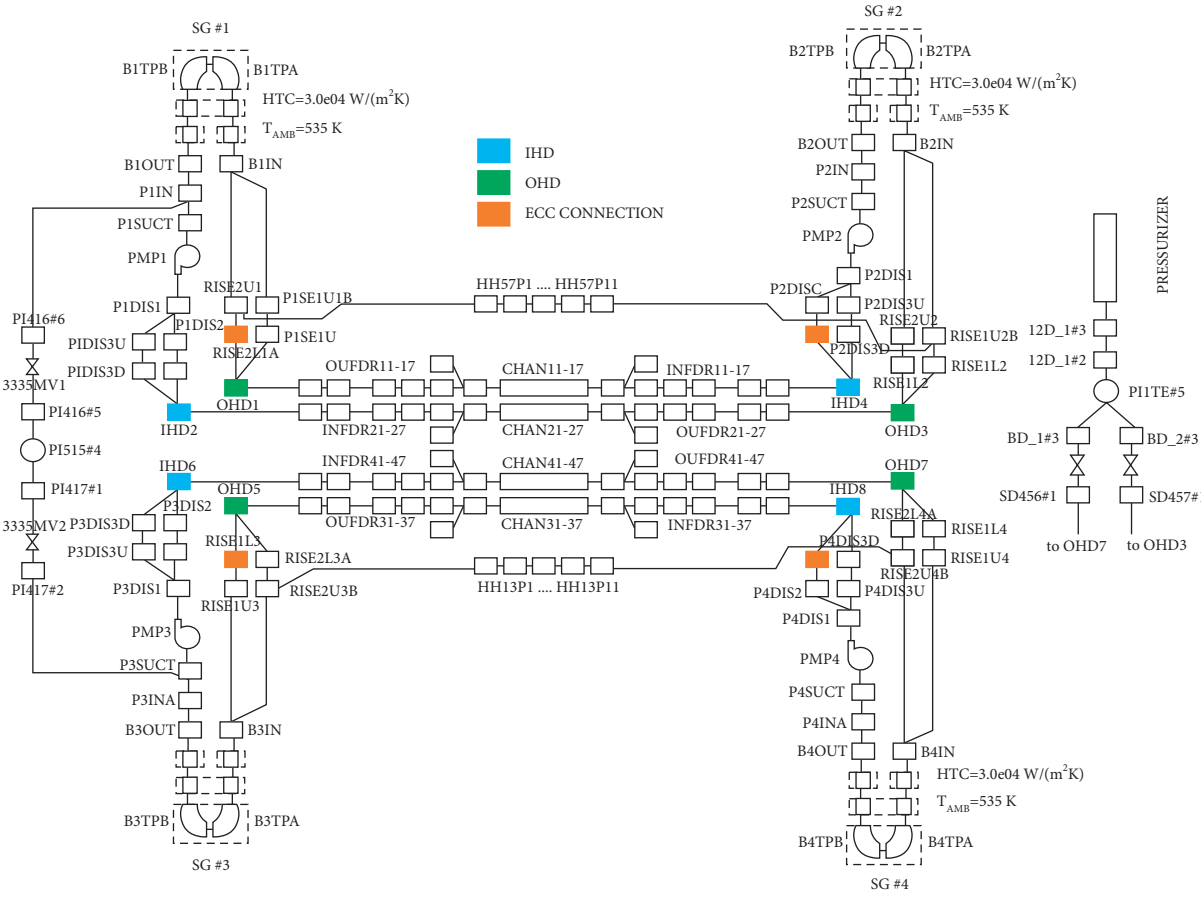

Figure 6: Thermal-hydraulic nodalization as specified by the benchmark [17].

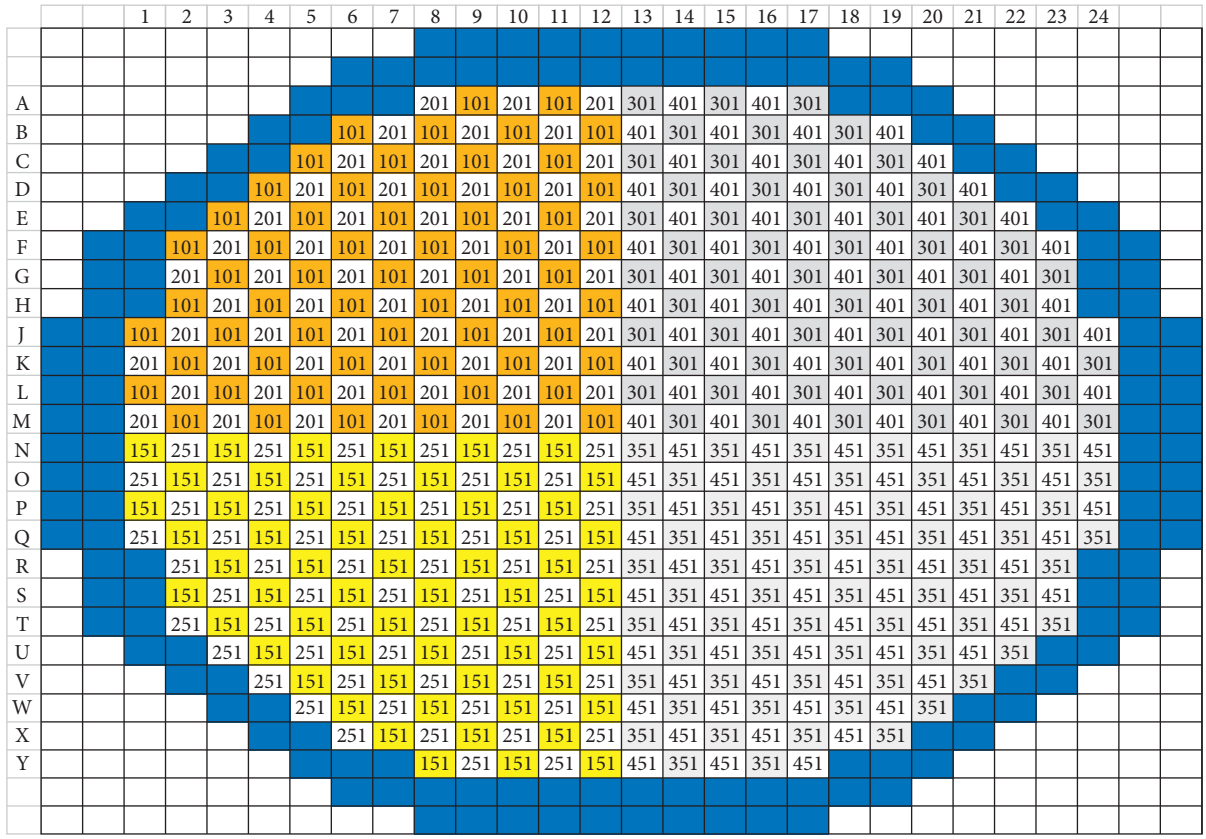

Figure 7: Fuel channel radial nodalization and mapping—reflector cells shown in blue.

controllers, which are adjusted to control reactivity and power distribution according to the liquid zone control algorithm, which is part of the RRS model. During transients, the reactivity devices are controlled by the RRS and respond based on the RRS program rules.

For steady-state calculations, where PARCS is running an eigenvalue calculation, the goal is to converge the zone levels to their equilibrium positions as quickly as possible, not to simulate the actual dynamics of the zone control system. Since the power level is fixed, k_{eff} replaces reactor power as the variable for bulk control to converge. To achieve the goals for steady-state analysis, there are several differences in the operation of the RRS model in a steady-state analysis.

(1) The time step size for RRS may be decoupled from the time step size of TRACE. As PARCS converges the flux each time it runs the eigenvalue calculation, the flux responds instantaneously to control device movements, regardless of the time step size taken by TRACE. Therefore, the RRS model steps forward by a fixed value of 0.5 seconds for each time step taken by TRACE-PARCS.

(2) A modified version of the power error calculation (CEP) module is used, which gives a power error proportional to the reactivity. Therefore, reactivity devices will respond to converge the core reactivity to zero. Under transients, the RRS control goes back to ensure convergence on power.

(3) The setback, stepback, and reactor trip modules are disabled to avoid spurious triggers while the steady state is being converged.

The tests that were performed with this model are summarized as follows:

(1) Two functionality tests designed to ensure that the coupled model with RRS behaves as expected

(2) Two-phase flow instability tests that induce figure-of-eight flow oscillation in an off-normal state

(3) A loss of flow station transient, where the initiating event is a loss of Class IV power

Prior to testing the coupled system against data, some simple tests were performed on the functionality of the coupled system. These include a zone control failure (fill valve fails closed for one zone) and an adjuster rod pull (rods 1 and 9). There are no reference results for these integrated tests. Hence, they are evaluated based on whether the observed behavior is consistent with the expected control system's behavior and CANDU phenomena.

The first integral tests examine the figure-of-eight flow oscillation phenomenon, where data is available from a scaled CANDU integral test facility, described in the next section, along with the theory and simulation data on a CANDU-6 facility [19]. The figure-of-eight flow oscillation is an instability characterized by low-frequency density waves ($T = 14$ s) propagating through the system [19]. Under

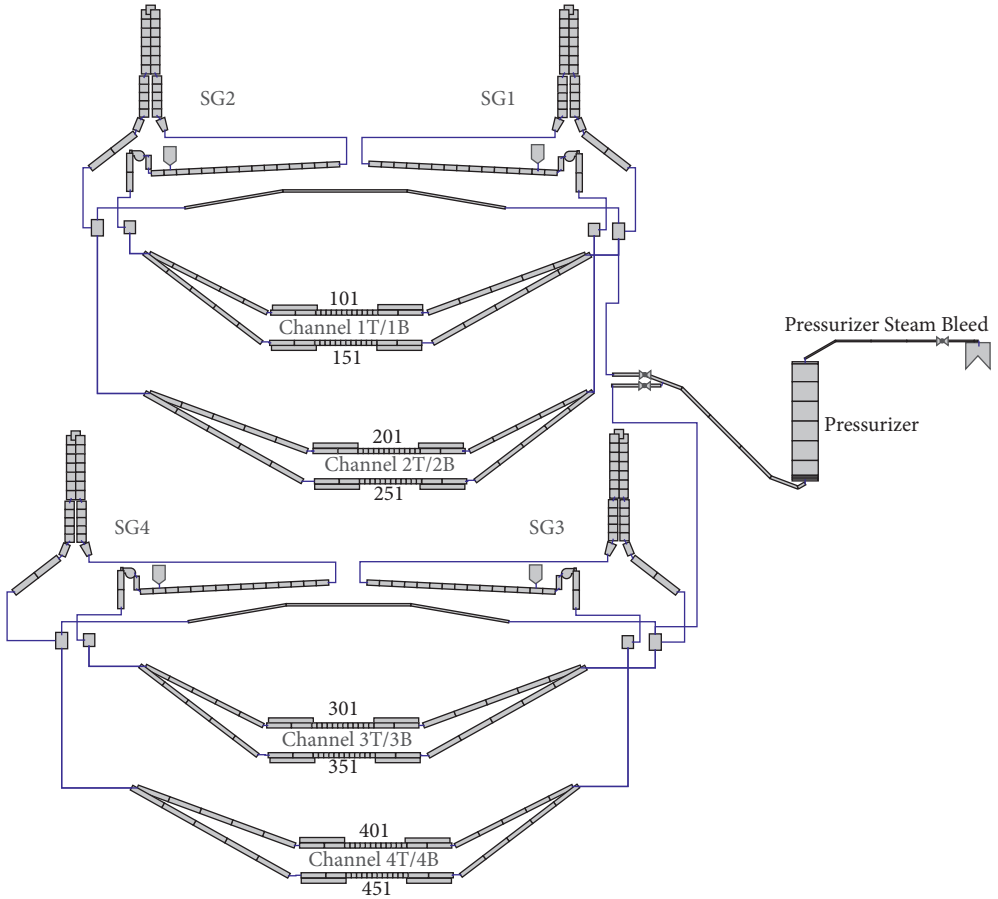

Figure 8: Thermal-hydraulic nodalization of 900 MW CANDU primary side.

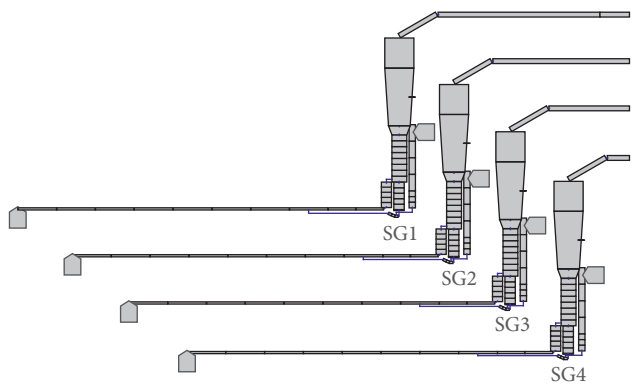

Figure 9: Thermal-hydraulic nodalization of 900 MW CANDU steam generators secondary side.

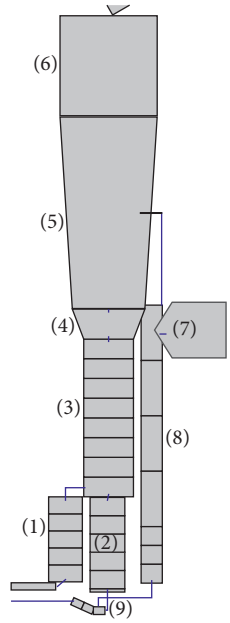

Figure 10: Thermal-hydraulic nodalization of 900 MW CANDU steam generators close-up, showing (1) preheater, (2) lower boiler, (3) upper boiler, (4) steam riser, (5) steam drum with steam separators/dryers, (6) steam dome, (7) moisture separator drains, (8) downcomer, and (9) leakage plate.

certain conditions, normal perturbations can result in diverging oscillation, eventually leading to reactor setback, stepback, or trip. The instability is very sensitive to reactor configuration, with the most recent CANDU plants designed to preclude this instability using a loop interconnect. To generate the instability, the 900 MW CANDU model was modified as follows:

(i) Removal of the interconnects within each loop. The loop interconnect is a balance line that promotes stability in the two-loops and was adopted in later designs like CANDU-900. The pressurizer connection between the two loops is maintained for all cases.

(ii) Setback, stepback, and reactor trip are disabled.

(iii) The increasing of the outlet header quality above the nominal levels to roughly 3–4% of the flow quality. It can be done by increasing reactor power (to 108% full power in this study) or reducing system pressure. Both cases are demonstrated.

In the TRACE model, the interconnects are replaced with valve components. These are open during the initial steady state convergence but closed at the start of the transient. The interconnects remain closed, except for the subcases, where the interconnects are reopened to test the effect of the interconnects at dampening flow oscillations.

The first case begins with a steady-state calculation at 108% full power (FP) and then disconnects the interconnect line. Once large oscillations develop, two subcases are tested to see if the oscillations can be stopped: unblocking the interconnects and lowering the reactor power.

The second case begins with nominal conditions (at 100%FP) and decreases the system pressure setpoint to 9300 kPa along with blocking the interconnects.

The results of these simulations are compared with prior theory and simulation [19]. It is not a benchmark comparison. Hence, an exact match is not expected, however, it is expected that the phenomenon is reproduced under the conditions where it is expected and has the expected properties.

The final transient that was simulated was a coupled loss of flow (LOF) event resulting from a loss of class IV power event that occurred at an operating CANDU station. To mimic the loss of power during the event, HTS circulation pumps, feed pumps, and steam generator feedwater pumps lose power at zero time, coincident with a turbine trip. The auxiliary feedwater pump is started two seconds later based on available information from the station. The turbine trip triggers a stepback to 60%FP [18].

It was necessary to add the condenser steam discharge valves (CSDVs), atmospheric steam discharge valves (ASDVs), and main steam safety valves (MSSVs) to the LOF model. The CSDVs are available for the first 13.5 seconds of the transient until the condenser vacuum is lost, while the ASDVs and MSSVs are available for the entire transient. The pressure thresholds for these valves are listed in Table 1. SDS1 is available so that the reactor can trip on high neutron power, high neutron log rate, low inlet feeder flow, or high HTS pressure.

The following model simplifications and assumptions are present in this model:

(i) The feedwater pumps are not explicitly modeled. Hence, the feedwater flow rate is modeled as a boundary condition instead. The trip of the main

Table 1: Pressure setpoints for steam discharge valves.

Valves	Threshold pressure (kPa)	100% open pressure (kPa)
CSDVs	5050	5320
ASDVs (before 13.5 s)	5320	5350
ASDVs (after 13.5 s)	5085	5115
MSSV bank 1	5652	5653
MSSV bank 2	5824	5825
MSSV bank 3	5996	5997
MSSV bank 4	6169	6170

feedwater pump was simulated separately, and the results were applied to the feedwater flow rate in the main model. The auxiliary feedwater pump is modeled as a flow rate of 10 kg/s per steam generator, starting from 12 seconds after they are started up (i.e., 14 seconds after the start of the transient).

(ii) Only the main circuit of the HTS is modeled, with the feed and bleed modeled as flow rate boundary conditions. Feed flow is disabled at the start of the transient. Bleed flow uses values taken from station data.

(iii) The stepback forecasts the reactor power by 0.25 seconds (equal to the interval of the stepback algorithm during a stepback), and the deceleration of the absorbers at the end of the stepback is treated as being instantaneous.

(iv) Control absorbers and shutdown rods are modeled as being dropped by gravity with damping, with the damping set to match the insertion rate in [18].

(v) ASDV and CSDV flow areas were set to achieve the nominal flow rates in [18]. MSSVs are set to the flow area provided in [18].

(vi) The pressurizer level setpoint is fixed at 6.5 m to match the station data and reference simulations.

2.7. The RD-14M Model. The RD-14M facility is an experimental facility designed to be a full-length but scaled model of the CANDU primary heat transport system, including 10 fuel channel simulators, which are scaled channels with seven fuel elements each (compared to 37 fuel elements for a typical CANDU bundle). The facility is used to model various phenomena related to the CANDU heat transport system and provides experimental data on these phenomena. The ability of computer codes to model these phenomena can be evaluated by modeling the RD-14M facility and comparing the simulation's results to the experimental data [20].

In this work, previously presented in Ref. [20], an RD-14M TRACE model was adapted to perform flow instability tests, simulating the same phenomenon as for the 900 MW CANDU model. The primary and secondary sides of the model are shown in Figures 11 and 12, respectively. The pressure boundary condition on Header 8, along with the emergency coolant injection system, both shown in Figure 11, were used in a previous study, simulating a header break experiment [10], however, they are isolated from the rest of the system in this work. The surge tank functions as a pressure boundary condition using the TRACE pressurizer component. This component automatically adds or removes energy at a specified rate to maneuver its pressure to the setpoint. On the secondary side, the feedwater flow rate and temperature along with the steam line outlet pressure are specified as boundary conditions, as shown in Figure 12. These remain fixed throughout the simulations performed in this work.

These simulations were performed to reproduce similar conditions as those in the experimental tests [21]. In these experiments, pump speed reductions and system pressure reductions were used to induce oscillations. Two different interconnect designs based on different scaling methodologies were tested, along with tests using no header interconnect. One interconnect, labeled "geometric similarity," was scaled based on conserving the momentum equation, while the other, labeled "dynamic similarity," was scaled based on conserving the ratio of the interconnect flow resistance and the heat transport system flow resistance. These labels are shown in Figure 11. It should be noted that at most one of these interconnects is connected at any time.

In these experiments, flow oscillations were produced when no header interconnect was used. The "geometric similarity" interconnect stabilizes the system against a pressure reduction but not a pump speed reduction, while the "dynamic similarity" interconnect stabilizes the system against both reductions. In the experiments where oscillations occur, the oscillation period is roughly 19 seconds [21], which is longer than the 14 seconds expected for a CANDU-6 heat transport loop [19]. As RD-14M is a full-length loop, comparable oscillation periods are expected.

Simulations were performed following the experimental procedures based on the available details [21], and the results were compared with the experimental results. If the results differed significantly, then the sensitivity of the results to the test procedure and test parameters was evaluated. RD-14M TRACE_Mac1.0 simulations do not include PARCS coupling, though the perturbations were performed using an ECI-coupled script.

3. Results and Discussion

3.1. CANDU-6 IAEA Code Coupling Benchmark. The CANDU-6 benchmark model described in the methodology section was simulated using the coupled PARCS_Mac1.0-TRACE_Mac1.0 for the following cases:

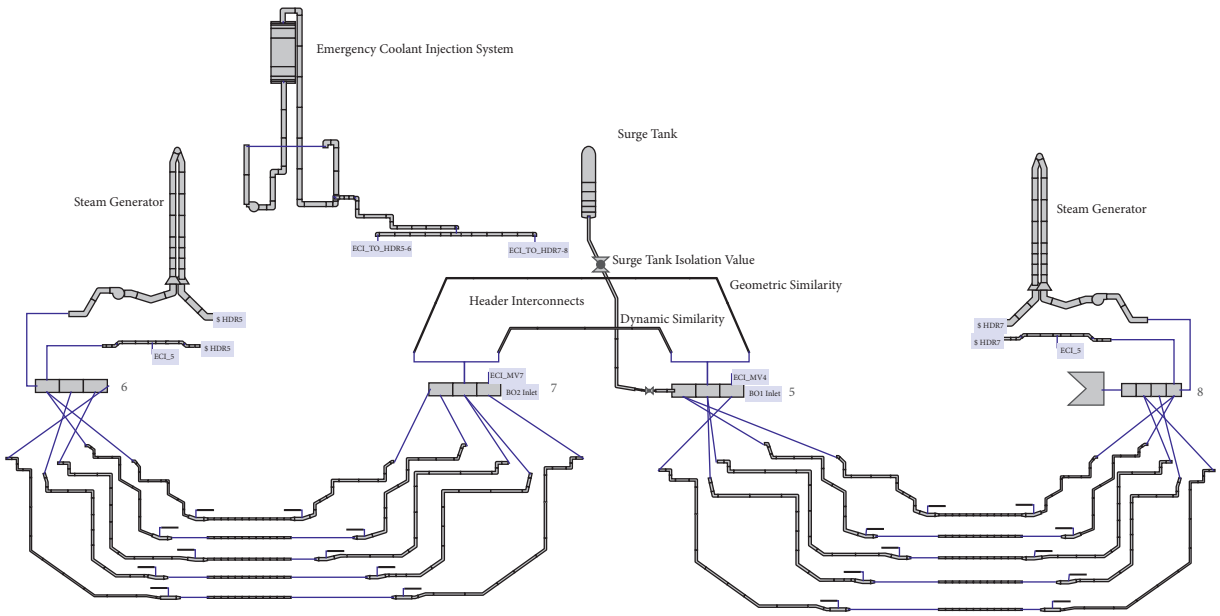

Figure 11: Thermal-hydraulic nodalization of the RD-14M primary side.

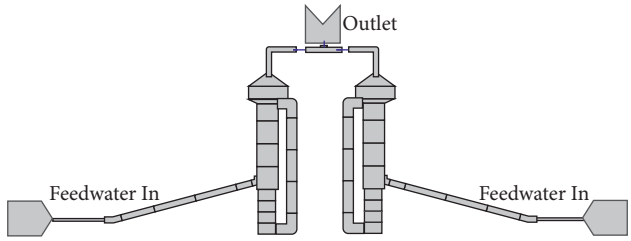

Figure 12: Thermal-hydraulic nodalization of the RD-14M secondary side.

(1) Steady state

(2) Inlet header break—loss of coolant accident (LOCA)

(3) Single coolant pump rundown—loss of flow (LOF)

(4) Adjuster rod withdrawal—loss of regulation (LOR)

Successful functionality of the coupled model has been demonstrated, and the general behavior of the models is consistent with that of the original version of the model that used an external coupling script. The model and cases were run for the full SCALE-generated branch structure and the reduced "benchmark specifications" branch structure from [3].

The quantitative results, shown in Table 2, agree well with the referenced externally coupled results [3], with the differences of at most a few percent. The difference in the results between the two different sets of cross-section data agrees with the prior study, with the reduced branch structure cases reaching higher maximum powers.

Possible sources of differences between the two studies include the increased data transfer frequency from using the built-in coupling (especially for the LOCA scenario), along with a difference between code versions, preventing a direct comparison. The referenced study included a sensitivity study to the information exchange frequency. Increasing this frequency from every 0.1 seconds to every 0.01 seconds for the LOCA simulation increased the power peak by roughly 45 MW, along with making it occur 0.04 seconds earlier [3]. An information exchange frequency of 0.05 seconds results in a very similar peak power every 0.01 seconds [3].

Figure 13 shows the evolution of the core power during LOCA. Voiding in the broken flow loop results in positive reactivity in half of the core, resulting in a power excursion with a power tilt toward the voided half of the core. As the shutdown rods take a couple of seconds to drop into the core, peak channel power occurs in the lower left quadrant of the core, as the power initially decreases at the top of the core before decreasing in the rest of the core.

Figure 14 shows the end state of the broken flow pass for the LOCA transient. The outlet header and fuel channels are almost fully voided. Cladding temperatures are elevated in most of this flow pass, with Figure 15 showing their evolution in time. The clad temperatures in CHAN24 exceed 1000 K.

Figure 16 shows the end state of the adjuster rod withdrawal transient. The reactor power is elevated throughout the entire core but with localized peaking where the adjuster rods have been withdrawn.

The original externally coupled model and the internally coupled model were run to compare the performance of the two models. The greatest performance benefit is found when

Table 2: Quantitative power results for ISCP benchmark model.

Case	Parameter	Full SCALE XS			Benchmark XS		
		Tight coupling	Ref. [3]	Error (%)	Tight coupling	Ref. [3]	Error (%)
Steady State	Max channel power (MW)	6.838 (S9)			6.881 (S9)	6.935 (S9)	
LOCA	Max core power (MW)	3433 (0.86 s)	3460 (0.90 s)	−0.8	4058 (0.91 s)	4000 (0.95 s)	+1.4
LOCA	Max channel power (MW)	14.10 (1.01 s, S8)			19.34 (1.08 s, S8)		
LOCA	Max bundle power (kW)	1779 (1.01 s, S8)			2446 (1.08 s, S8)		
LOCA	Core energy deposition (MJ)	3985			4602		
LOCA	Max channel deposition (MJ)	17.16 (S9)			21.40 (S9)		
LOCA	Max bundle deposition (kJ)	2129 (S10)			2672 (S8)		
LOF	Max core power (MW)	2177 (25.0 s)	2170 (25.0 s)	+0.3	2280 (25.0 s)	2260 (25.0 s)	−0.9
LOF	Max channel power (MW)	7.50 (25.0 s, S9)			7.91 (25.0 s, S9)		
LOF	Max bundle power (kW)	930 (25.0 s, S9)			982 (25.0 s, S9)		
LOR	Max core power (MW)	2621 (25.0 s)	2730 (25.0 s)	−4.0	2903 (25.0 s)	2970 (25.0 s)	−2.3
LOR	Max channel power (MW)	9.71 (25.0 s, L18)			10.90 (25.0 s, L18)		
LOR	Max bundle power (kW)	1314 (25.0 s, L18)			1479 (25.0 s, L18)		

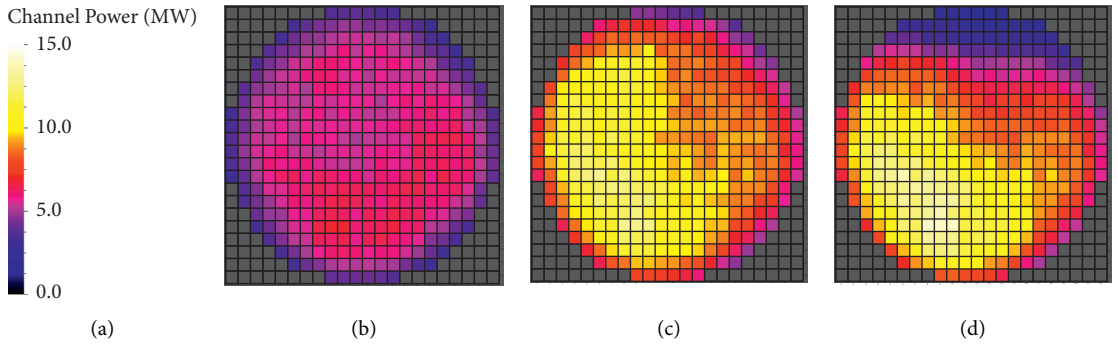

Figure 13: Inlet header breaks channel power profile. Color legend provided by (a). Profiles given are as follows: (b) before the transient, (c) at peak core power, and (d) at peak channel power.

running transient analysis, with a run time of 160 seconds for the internally coupled case, compared to 505 seconds for the externally coupled model—approximately a factor of 3 difference. This benefit arises from avoiding the need for frequent restarts as is needed for the externally coupled case.

There is less of a performance benefit for steady-state analysis. PARCS can be configured to skip TRACE time steps, though there is currently an arbitrary limit of 20 TRACE time steps per PARCS execution. Each time PARCS is executed by TRACE, it runs until convergence, adding significant computation time compared to a standalone TRACE run, as well as when compared to the externally coupled model, where only a small number of restarts and PARCS executions are required.

3.2. 900 MW CANDU Model Tests. As mentioned in the methodology section, three sets of tests were performed, which are as follows:

(1) Two functionality tests to evaluate that the coupled model is working as expected

(2) Flow instability tests that induce flow oscillations in an off-normal state

(3) A loss-of-flow transient initiated by a loss of Class IV power

As mentioned in the methodology section, two functionality tests were performed to ensure robust RRS behavior.

The first test was performed to simulate an abnormal operating occurrence (AOO), where the liquid zone control value fails, resulting in the draining of the liquid zone. The transient begins with the fill rate for Zone 5 being set to zero. It causes the liquid zone to rapidly empty, as shown in Figure 17. The result is increasing the reactor power, shown in Figure 18, which the RRS responds to by increasing the fill rate of the other zones, particularly adjacent zones (Zones 6 and 12), for which the spatial power error is the greatest. When the zone power in Zone 5 exceeds 110% of its nominal value, a setback is triggered to bring the zone power below 105% as expected. The result is an average reactor power of approximately 95%. At this point, the reactor continues to operate in a new steady state, though the average liquid zone controller level gradually decreases because of the build-up of xenon-135. After 50 minutes (not shown), the average zone water level is approximately 20%.

The adjuster driven transient begins with adjuster rods 1 and 9 being withdrawn at the maximum rate possible based on the adjuster drive motor design. The adjuster and absorber rod movements are shown in Figure 19. It causes a small bulk power excursion by roughly 1% along with a flux tilt, as shown in Figure 20. The RRS setback logic is triggered

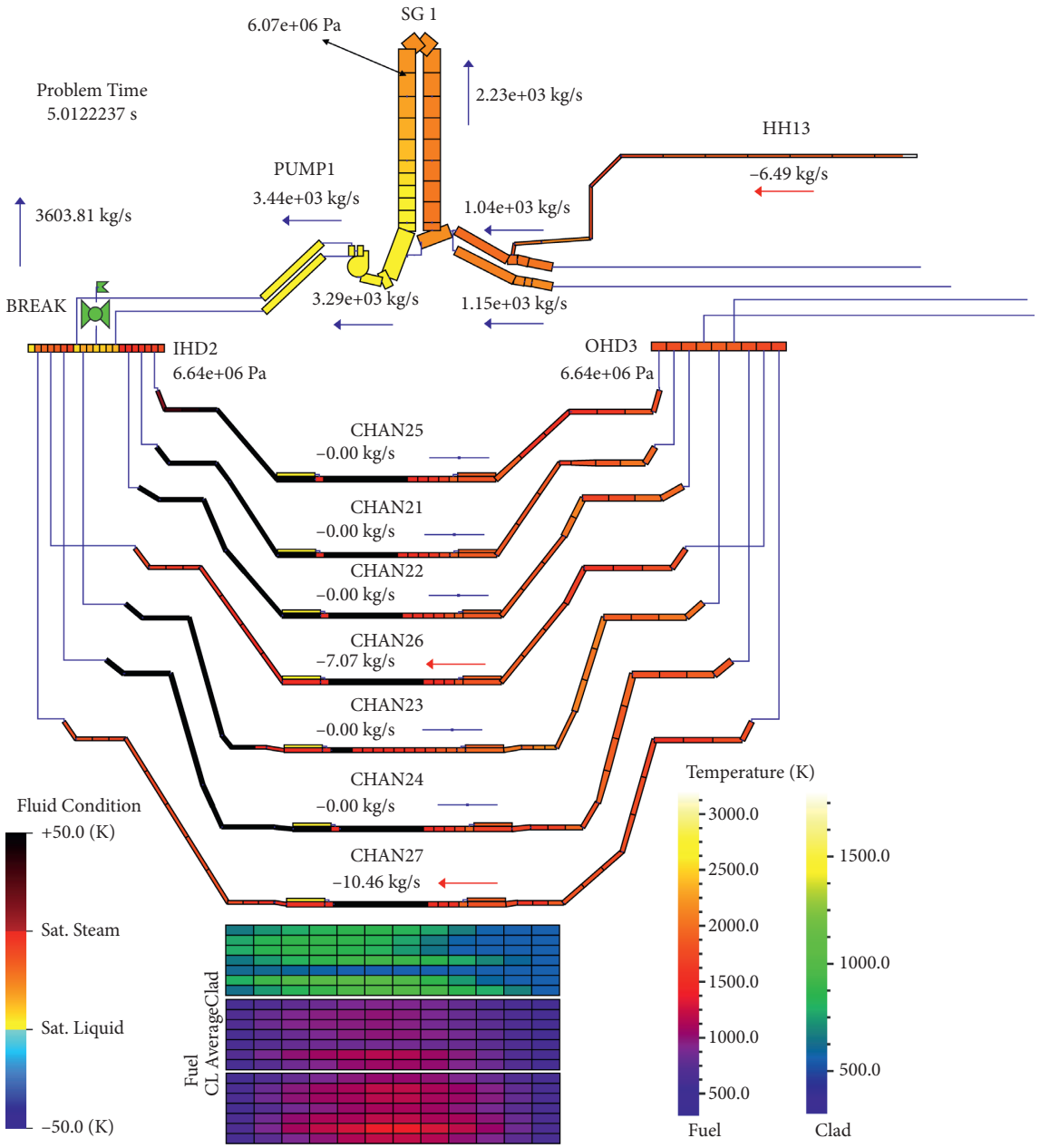

Figure 14: LOCA broken pass thermal-hydraulic conditions after 5.0 seconds.

first, with liquid zone levels increasing rapidly as shown in Figure 21, however, high zone powers then trigger the stepback logic multiple times, resulting in partial absorber rod drops and the reduction of bulk power to 80%. Afterwards, there is further setback to 60% full power because of zone tilt.

In the RRS simulator produced by this work, the RRS attempts to maintain the current reactor power after a stepback. As the reactor power is still decreasing, this results in a brief negative power error that triggers the withdrawal of the first two adjuster banks, as seen in Figure 19. The banks stop at 25% of the maximum withdrawal as the power error briefly exceeds positive 3%, triggering RRS to stop withdrawing adjusters and start reinserting one bank. As bank C includes one of the failed rods that is detected as being withdrawn, it tries to reinsert bank C, however, nothing happens as one rod is stuck out and the others are already fully inserted.

After the initial stage of the transient, RRS responds to the xenon transient by withdrawing the control absorbers (up to a 75% average liquid zone level) and then reducing the average zone level. The liquid zone levels and reactor power remain highly asymmetric because of the asymmetric adjuster withdrawal.

Overall, the functionality tests show behavior consistent with what is expected based on the design of the RRS model in this work.

The flow instability tests using TRACE_Mac1.0 and PARCS_Mac1.0 were carried out by running a steady-state

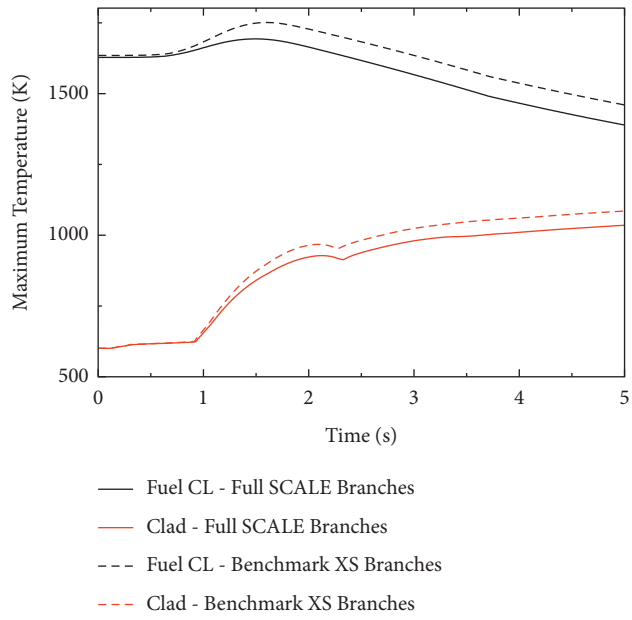

Figure 15: LOCA maximum fuel/clad temperatures.

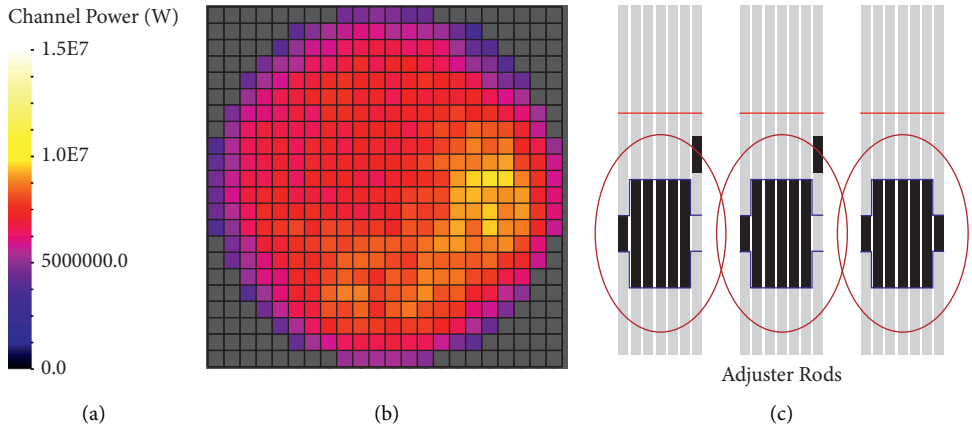

Figure 16: End state for adjuster rod withdrawal. Color legend provided by (a) for the power profile given by (b). Final adjuster rod positions for the transient given by (c), shown in three groups of 7 from front to back, with each group shown from left to right.

calculation with the header interconnects unblocked, and the reactor power set to the initial power for the respective case, either 108%FP or 100%FP. Then, the transient run is performed using the steady state calculation's conditions as the initial conditions for the transient. For the transient runs, the header interconnects are blocked at time $t = 0$. Then, for the system pressure reduction case, the pressure setpoint is set to 9300 kPa at time $t = 0$. In both cases, flow oscillations were produced.

Figure 22 shows the base case for the 108%FP scenario. At this power, an average void fraction of 14% corresponds to an average flow quality of 2%. Flow oscillations grow until 340 seconds to a large amplitude, with these oscillations also appearing in the reactor outlet header (ROH) pressures. When the oscillations grow large enough, the coolant voiding in the core rapidly changes and moves from one end of the core to the other as the voiding at any given time is the greatest toward the outlets of the fuel channels of the lower-pressure flow pass. The result is oscillations in both the end-to-end flux tilt and total reactor power at rates which RRS is incapable of compensating for. At this point, reactor setback, stepback, or trip would be expected, however, for this simulation, these are disabled from acting so that the continuation of the transient can be observed. The average system pressure increases as the coolant swells because of both the loss of regulation and the loss of cooling effectiveness. The RRS response includes the movement of adjuster rods and control absorbers, which adds absorbing material to the top half of the core, resulting in a top/bottom flux tilt. The combination of these effects leads to an overall increase in system damping, suppressing the oscillations. At this point, the outlet header void fractions are higher and the system pressure is lower than during the initial steady state. The average flow quality at this point is 4%. The oscillations begin

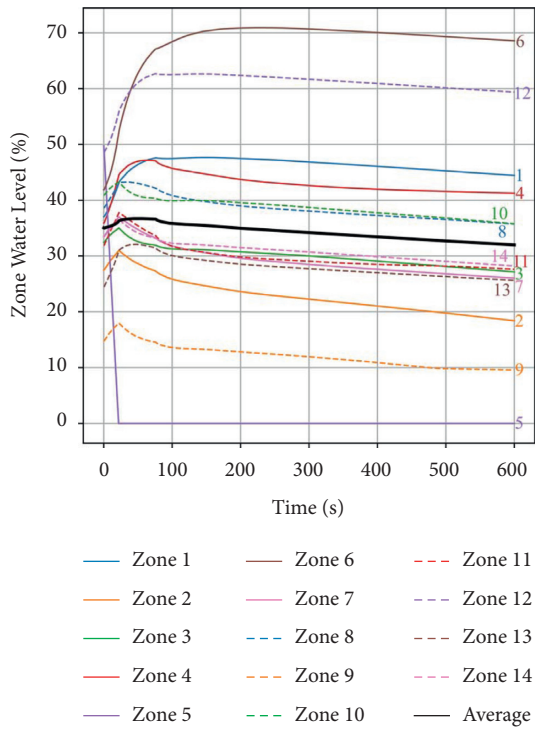

Figure 17: Liquid zone levels for zone control failure transient.

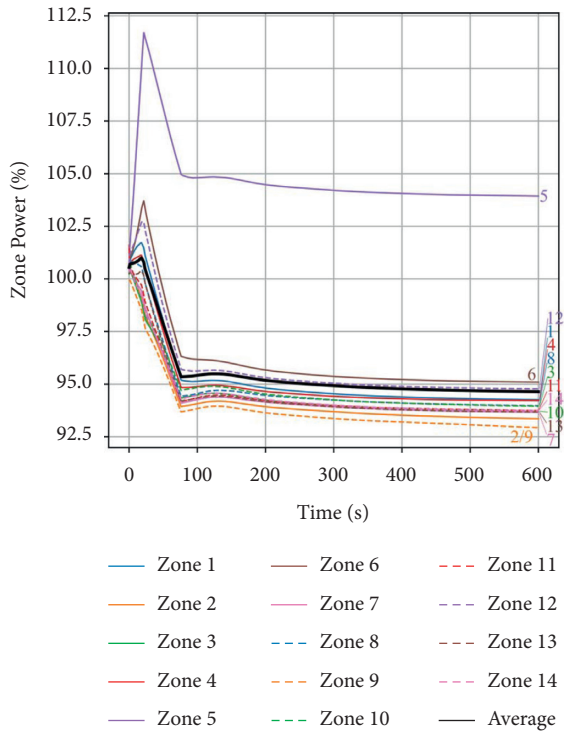

Figure 18: Zone powers for zone control failure transient.

to grow again as the outlet header void fraction decreases toward the initial steady-state value, as shown in Figure 23. The average flow quality is approximately 3% at the time which oscillations begin to grow again. It repeats every several minutes for as long as the simulation continues to run.

The period of oscillations is dependent on the amplitude, ranging from 15 seconds at low amplitude to 12 seconds at high amplitude. These are close to the 14 seconds predicted in literature [19]. The period is reduced at high amplitude as one voided region is fully compressed on each half-cycle,

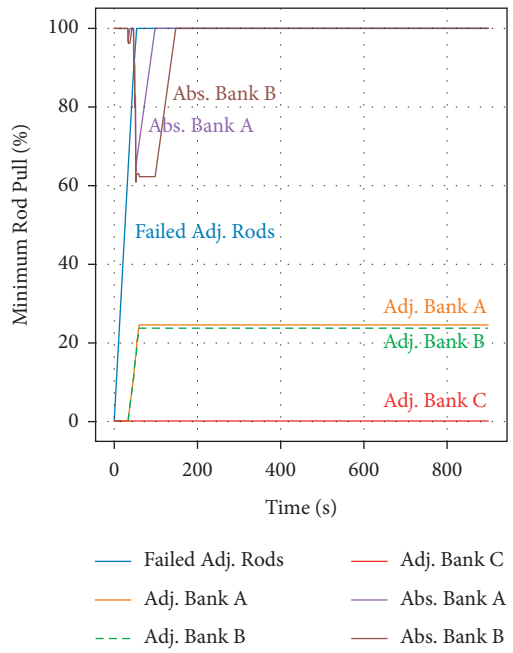

Figure 19: Adjuster and absorber levels for adjuster rod failure transient.

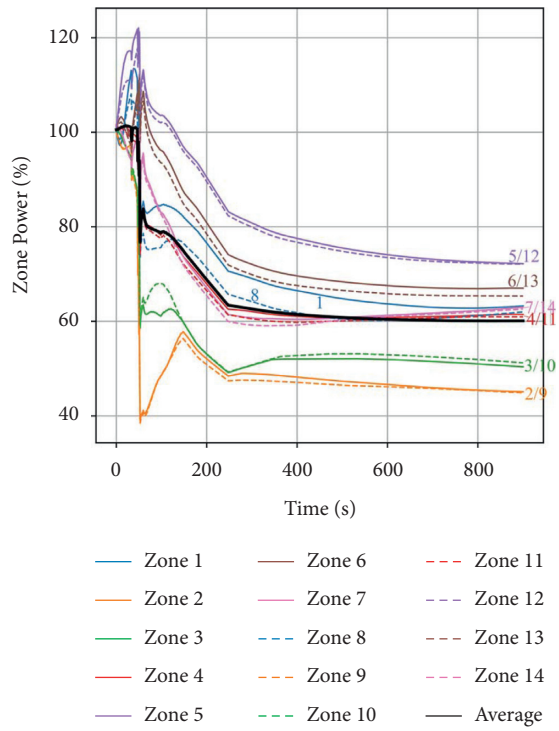

Figure 20: Zone powers for adjuster rod failure transient.

resulting in the oscillations "bouncing" off the incompressible liquid.

Figure 24 shows the same case, except that the reactor power setpoint is reduced to 100%FP at 200 seconds. It reduces the amplitude of the oscillations, and they eventually stop after another 400 seconds. This figure also shows how the total reactor power begins to oscillate when the amplitude of the flow oscillations is large. Figure 25 includes a second power decrease to 90%FP at 300 seconds, which stops the flow oscillations more quickly than maintaining 100%FP.

Figure 26 shows the base case until 200 seconds, at which point the header interconnects are unblocked. The result is that the oscillations are reduced to a low amplitude, as expected.

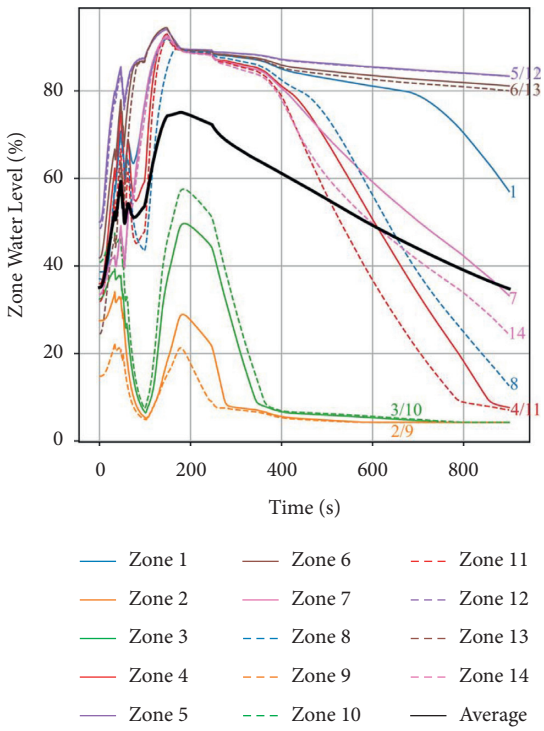

Figure 21: Liquid zone levels for adjuster rod failure transient.

Figure 27 shows the base case for the reduced pressure scenario. The oscillations begin once the system reaches the 9300 kPa setpoint and behave similarly to the increased power case in Figure 22. Once again, the oscillations are eventually dampened by the increasing coolant voiding and the control rod movements. Figure 28 shows that reducing the reactor power to 90%FP stops the oscillations, even at the reduced system pressure. Figure 29 shows that connecting the balance headers also suppresses the oscillations.

The final simulation using this model with TRACE_Mac1.0 and PARCS_Mac1.0 is the loss of flow event initiated by a loss of Class IV power, as detailed in the methodology section. The most significant effects on this are the shutdown of the heat transport circulating pumps, along with a turbine trip. The turbine trip has the dual effect of quickly stopping steam flow out of the steam generators along with triggering a reactor stepback to 60%. The heat transport system feed pumps and the main boiler feed pumps also shut down. Table 3 summarizes the subsequent events that occur as a consequence of the transient. The low flow trip timing is very similar to the referenced study [18]. One significant difference is that the pressure in this study never reaches the threshold to actuate the HTS liquid relief valves.

Figure 30 shows the effect of the transient on the core neutronics along with the circulating flow rate. Initially, the reactivity and fission power begin to increase as the void fraction near the channel outlets increases. However, the stepback quickly adds negative reactivity to being the reactor power down to 60%FP. The behavior of the stepback depends on the time extrapolation of the reactor power and the kinematics of the control absorbers. With the assumptions made in this model, the stepback is triggered briefly for a

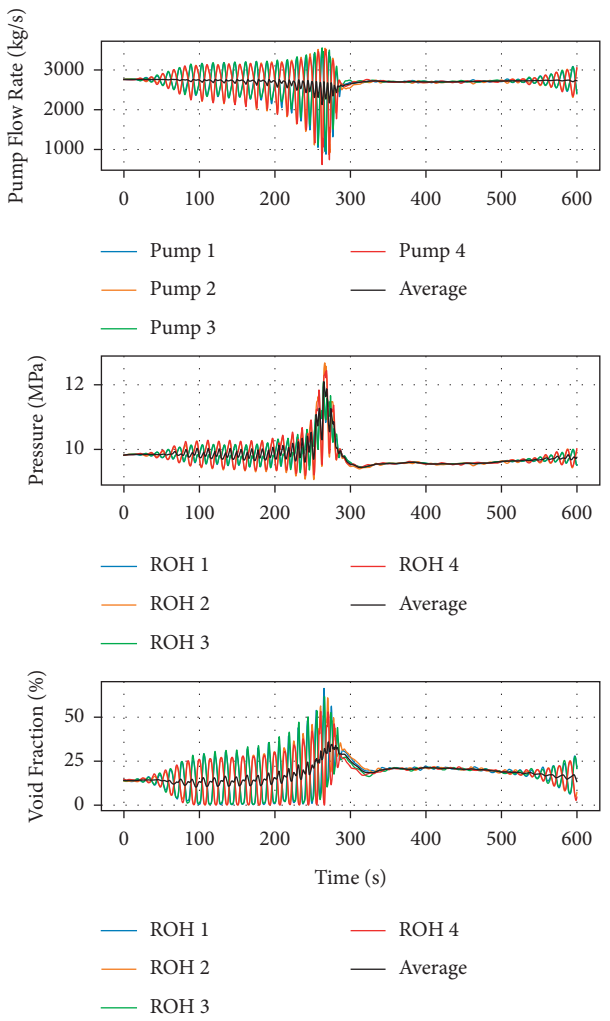

Figure 22: Flow rate oscillations at 108% full power with blocked header interconnects. The even pair of pumps and headers oscillate out-of-phase with the odd pair.

second time shortly after the first stepback ends as the power remains over 60%FP for a brief period. The impact of this secondary RRS action was minimal, however. A low flow trip is triggered at 3.4 seconds into the transient, with the shutdown rods being dropped into the core 0.3 seconds later. It reduces the reactor power to decay heat levels within 2 seconds.

Figure 31 shows the effect of the transient on the heat transport pressure. The flow reduction results in coolant swelling, which rapidly increases the system pressure until the reactor trip reduces the rate of heat generated in the core, at which point the coolant shrinks and the system pressure decreases. The peak pressure in this study is lower than that in the previous studies, however, it is still comparable to the available station data.

Figure 32 shows the effect on the steam generator pressures. The pressure rapidly increases as the turbine stop valves close and prevent the removal of energy from the steam generators. This initial behavior of the steam generator pressure (first 15 seconds) is similar to the results from Ref. [18], though at a somewhat higher pressure. The

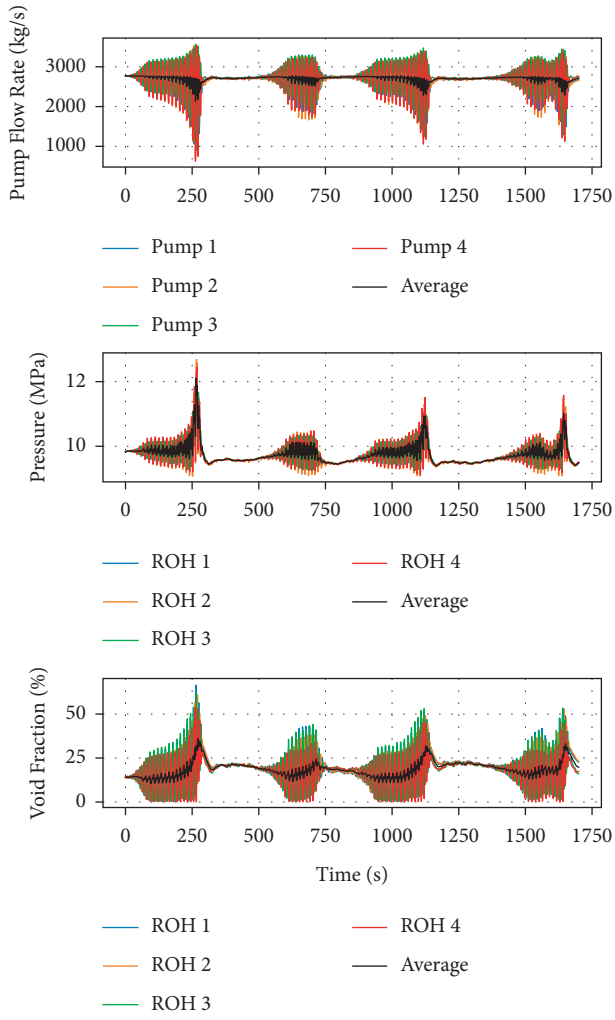

Figure 23: Flow rate oscillations at 108% full power with blocked header interconnects—full simulation. The even pair of pumps and headers oscillate out-of-phase with the odd pair.

pressure peaks at approximately 5300 kPa as the ASDVs and CSDVs open. Then, it further increases when the condenser becomes unavailable, for which the behavior in this study is closer to the station data than to the reference simulations, for both the value and timing of the peak pressure. It occurs at roughly 30 seconds into the transient, at 5440 kPa, at which point the energy input from the HTS matches the energy output through the ASDVs. The pressure then decreases at a similar rate to prior studies. However, while the pressure levels off at 5200 kPa in the station data, it continues to decrease toward the ASDV setpoint of 5085 kPa in this study.

Figure 33 shows the effect of the transient on the reactor inlet header (RIH) temperature. The steady-state temperature is a function of the heat transfer efficiency of the steam generator. Typically, the steam generator model is tuned to achieve the desired RIH temperature, whether matching station data or a design value. In this study, no specific tuning was performed. For the real event, the measured temperature immediately prior to the transient was 263.8°C, which is very close to the temperature in this study and almost 2°C lower than the temperature in the previous studies. The temperature initially increases and peaks 10 seconds into the transient, consistent with the previous studies. This increase is caused by the interruption of feedwater flow, which greatly reduces the heat transfer efficiency in the preheater section of the steam generators.

Figure 34 shows the effect of the transient on the steam generator level. Overall, the trend is dominated by shrinkage in the steam generator as the rate of heat input from the HTS decreases, and the water level follows a similar trend to the station data and RELAP results. The steam generator water level also decreases because of the mass imbalance caused by the loss of the boiler feed pumps. The auxiliary feed pumps activate, however, their flow rate is lower than the rate of steam flow through the ASDVs.

3.3. RD-14M Model Flow Instability Tests. The RD-14M flow transient simulations [20] using TRACE_Mac1.0 were performed to model the figure-of-eight flow oscillations similar to those simulated in the 900 MW CANDU model. In this case, the reference data are the results from the experiments performed at the actual facility [21]. As in the

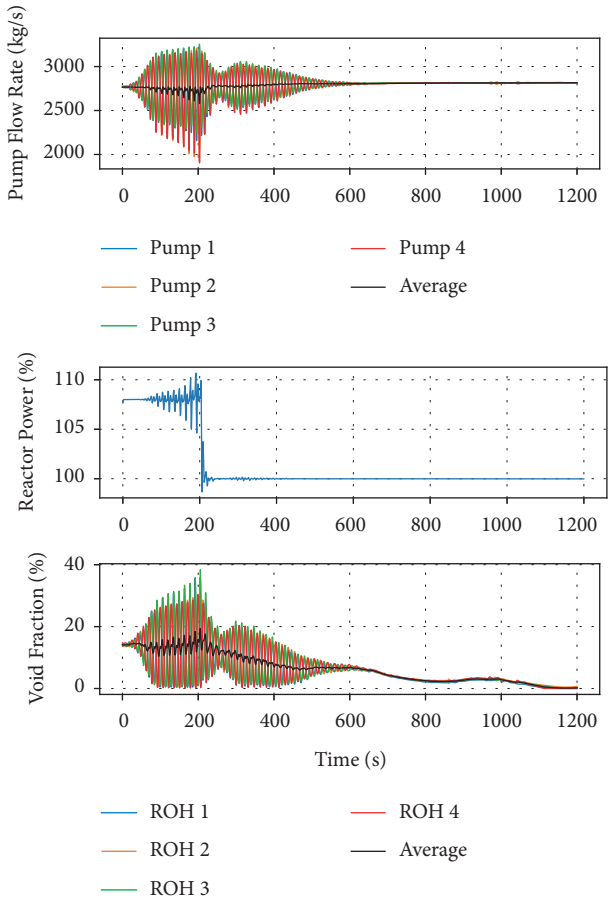

Figure 24: Flow oscillations at 108% full power, with power reduced to 100% after 200 seconds. The even pair of pumps and headers oscillate out-of-phase with the odd pair.

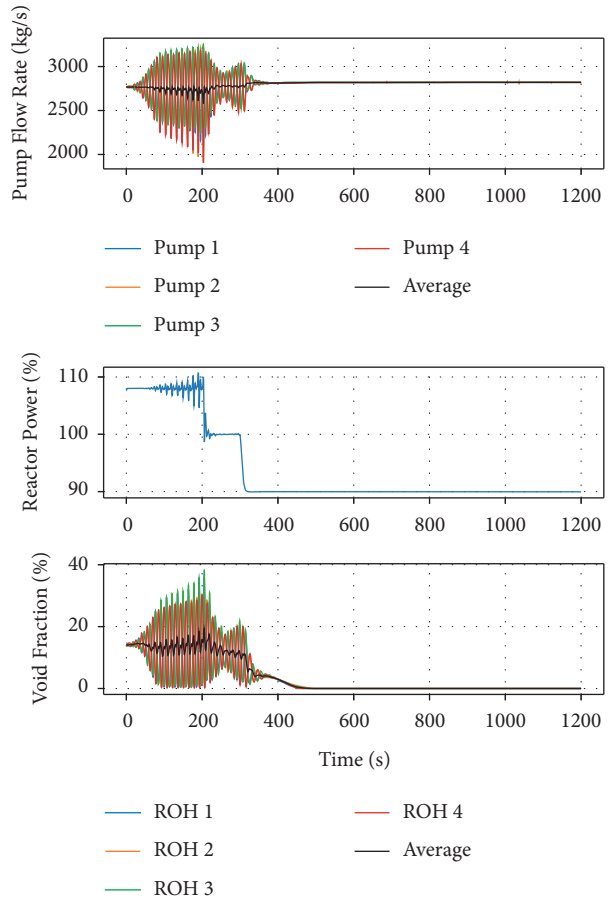

Figure 25: Flow oscillations at 108% full power, with power reduced to 100% after 200 seconds and to 90% after 300 seconds. The even pair of pumps and headers oscillate out-of-phase with the odd pair.

900 MW CANDU model, flow oscillations could occur once the system state was changed in a way that induced an increase in void fraction in the outlet headers. In this case, this was done either by reducing the pump speed or by reducing the system pressure. The initial pump speed and system pressure are 75% nominal speed and 10 MPa, respectively.

With no header interconnect, when the pump speed is reduced, flow oscillations can be induced as in the experiments. One significant difference between the simulations and experiments was that a gradual pump speed reduction would not induce growing oscillations, while an abrupt change would. It suggests that in the simulation, the initial perturbation is important, where, for a given pump speed, small oscillations die out while larger oscillations grow. It is not observed in the experimental results that induce large oscillations when the pump speed is reduced in several smaller increments [21]. However, there are no experimental results available for a true gradual reduction of the pump speed. Thus, while a gradual pump speed reduction does not reproduce the experimental results, an abrupt reduction in the pump speed results in growing oscillations up to a large amplitude, similar to the experimental data, as shown in Figure 35. The oscillation period is roughly 15 seconds, comparable to the results for the 900 MW CANDU model, however, it is significantly different from the 19-second value from the RD-14M experiments [21].

Figure 36 shows a test, where, with no header interconnect, the system pressure was reduced in increments, temporarily isolating the surge tank after each increment. This test does not directly match any of the experimental tests but evaluates the same results as one of the tests in a more quantitative manner. The test shows that the system is unstable between 9.3 and 9.6 MPa while the surge tank is isolated. In the experiments, this range was instead between 9.5 and 9.8 MPa [21]. While differing quantitatively, there is a qualitative agreement, showing that there is a pressure range, and hence a corresponding outlet header void fraction range makes the system unstable. The system is stable outside of this pressure and void fraction range in either direction. This observation also supports the theory discussed in [19].

When the "geometric similarity" header interconnect is used, the simulations and experiments are comparable, showing that the interconnect is effective when the system pressure is reduced but not when the pump speed is reduced. The results of the system pressure reduction are shown in Figure 37. A temporary oscillation occurs when the surge

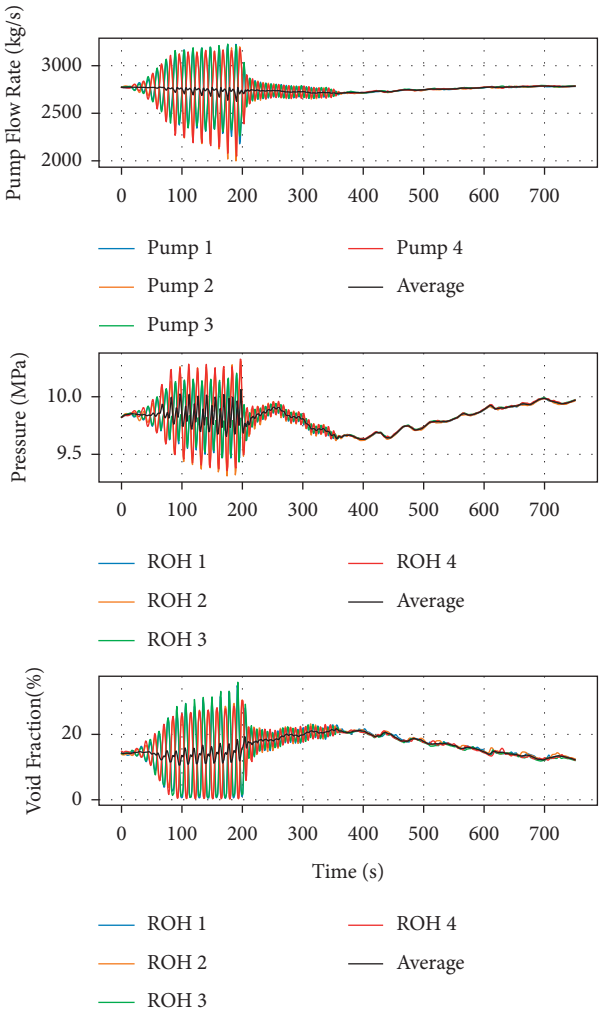

FIGURE 26: Flow oscillations at 108% full power, with balance headers unblocked after 200 seconds. The even pair of pumps and headers oscillate out-of-phase with the odd pair.

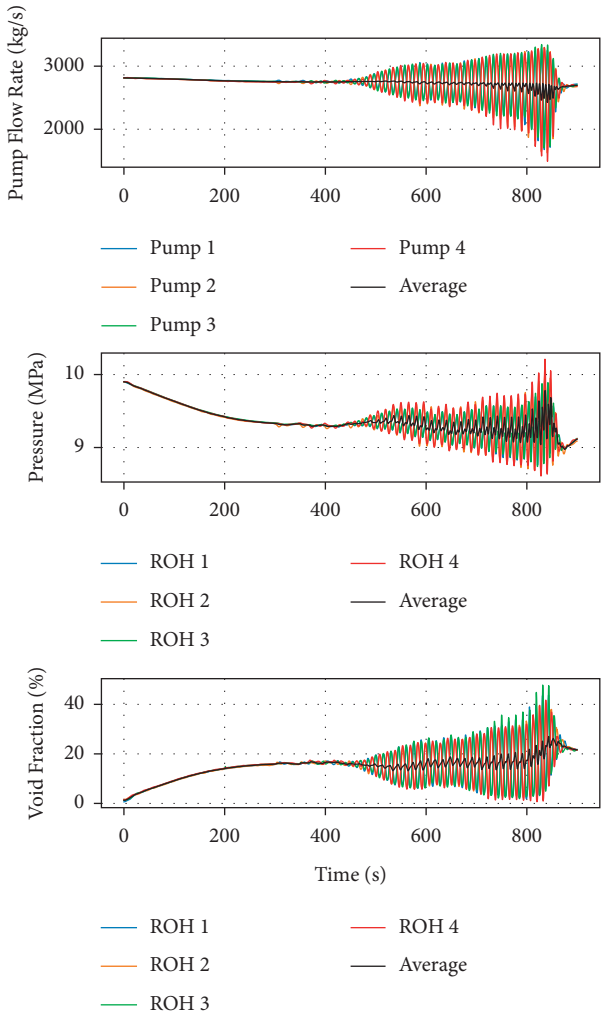

FIGURE 27: Flow rate oscillations at reduced pressure with blocked header interconnects. The even pair of pumps and headers oscillate out-of-phase with the odd pair.

tank is isolated but dies out after several oscillation periods. However, in the simulations, a pump speed reduction to 68%, as performed in the experiments, did not result in growing oscillations. Instead, a simulation was performed, which reduced the pump speed first to 60%, then to 55%, then increasing it to 64% [20], with Figure 38 showing the oscillations subsequent to this final increase. At this pump speed, the oscillation amplitude gradually decreases until the surge tank is isolated at $t = 1320$ s, which causes the oscillations to grow again until the surge tank is reconnected at $t = 1440$ s. The pump speed is increased to 66% at $t = 1600$ s, which causes the oscillations to decay more rapidly.

When the "dynamic similarity" header interconnect is used, the simulations and experiments show that the system is stable to the figure-of-eight oscillations for both a system pressure reduction and a pump speed reduction. Figure 39 shows that this interconnect is more strongly stabilizing for system pressure reduction than the "geometric similarity" interconnect (Figure 37). For the pump speed reduction, in addition to the experimental procedure of reducing the pump speed gradually to 66% nominal speed, a more extreme simulation was performed, with a step reduction to 55% nominal speed. The initial oscillations resulting from this perturbation decay over several oscillation periods are as shown in Figure 40.

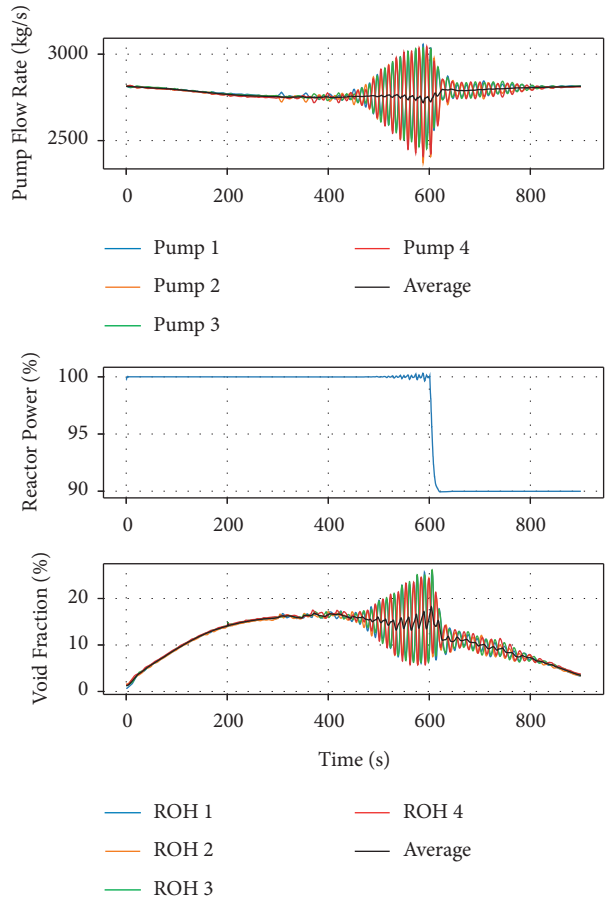

FIGURE 28: Flow oscillations at reduced pressure, with power reduced to 90% after 600 seconds. The even pair of pumps and headers oscillate out-of-phase with the odd pair.

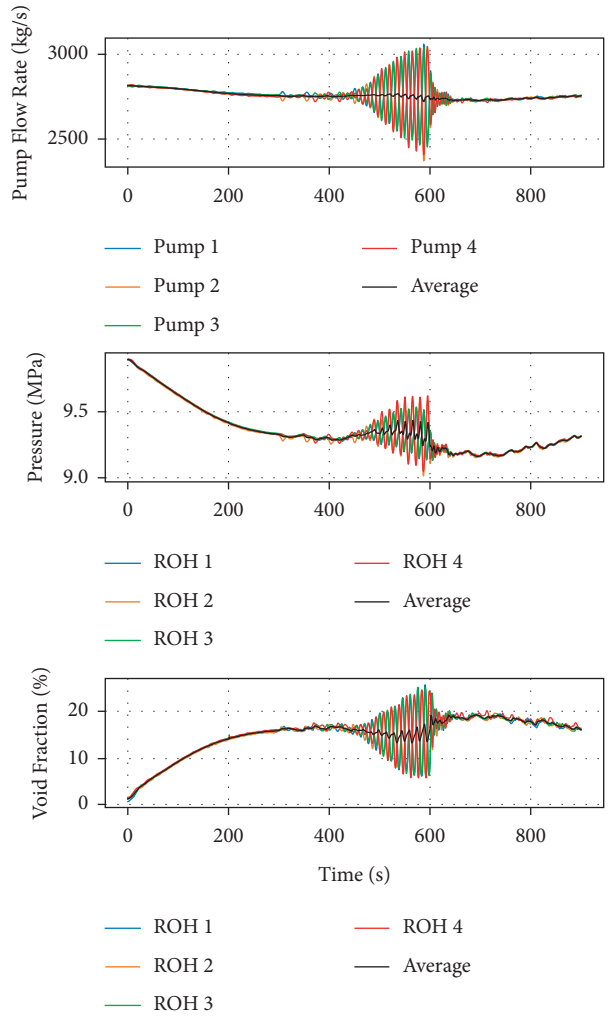

Figure 29: Flow oscillations at reduced pressure, with balance headers unblocked after 600 seconds. The even pair of pumps and headers oscillate out-of-phase with the odd pair.

Table 3: Event timings compared to station data and previous studies.

Event	Station data (s)	[18] (s)	This work (s)
Loss of class IV power	0	0	0
Turbine trip	0	0	0
Emergency stop valve (ESV) close	No data	0.28	0.28
Stepback on turbine trip	0.5	0.5	0.5
CSDV open	0.8	0.8	0.8
ASDV open	1.0	13.6	13.5
SDS1 low flow trip	2.9	3.3	3.4[1]
Liquid relief valve (LRV) open	3.5	4.2	Not actuated
SDS2 low flow trip	11.6	N/A	N/A

[1]This is the timing of when the trip condition is met—shutdown rod actuation is delayed by an additional 0.3 seconds.

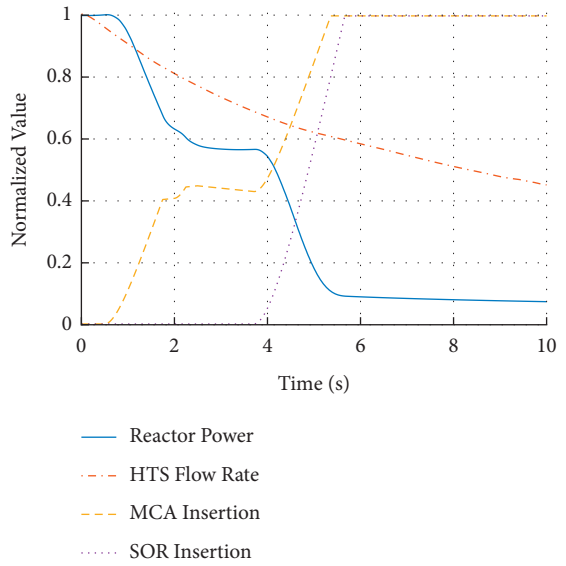

Figure 30: Normalized power, heat transport system flow rate, and rod insertion for loss of flow event. Rod insertion is given for the mechanical control absorbers (MCAs) and shutoff rods (SORs).

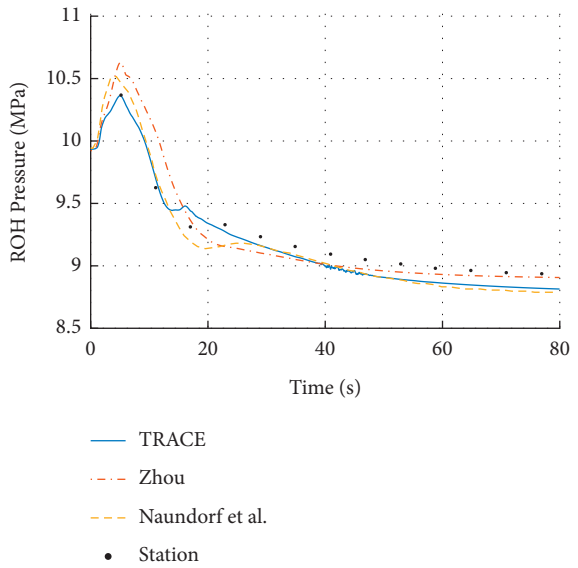

Figure 31: Reactor outlet header pressures for the loss of flow event.

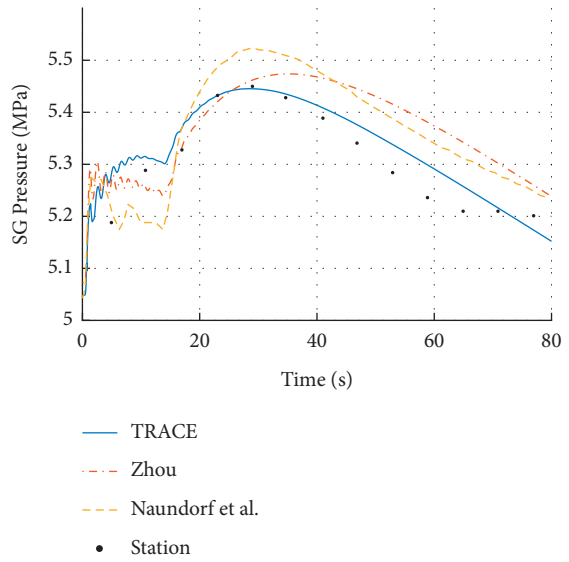

Figure 32: Steam generator pressures for the loss of flow event.

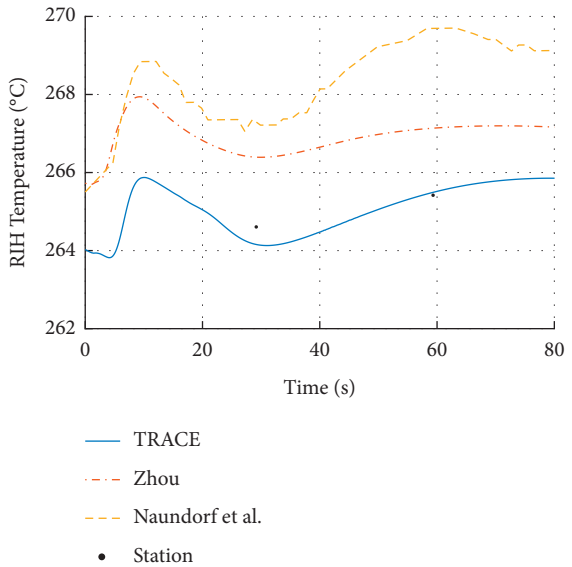

Figure 33: Reactor inlet header temperatures for the loss of flow event.

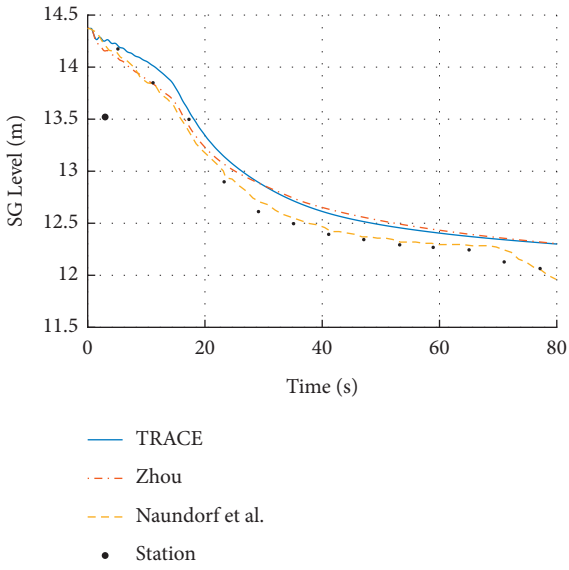

Figure 34: Steam generator levels for the loss of flow event.

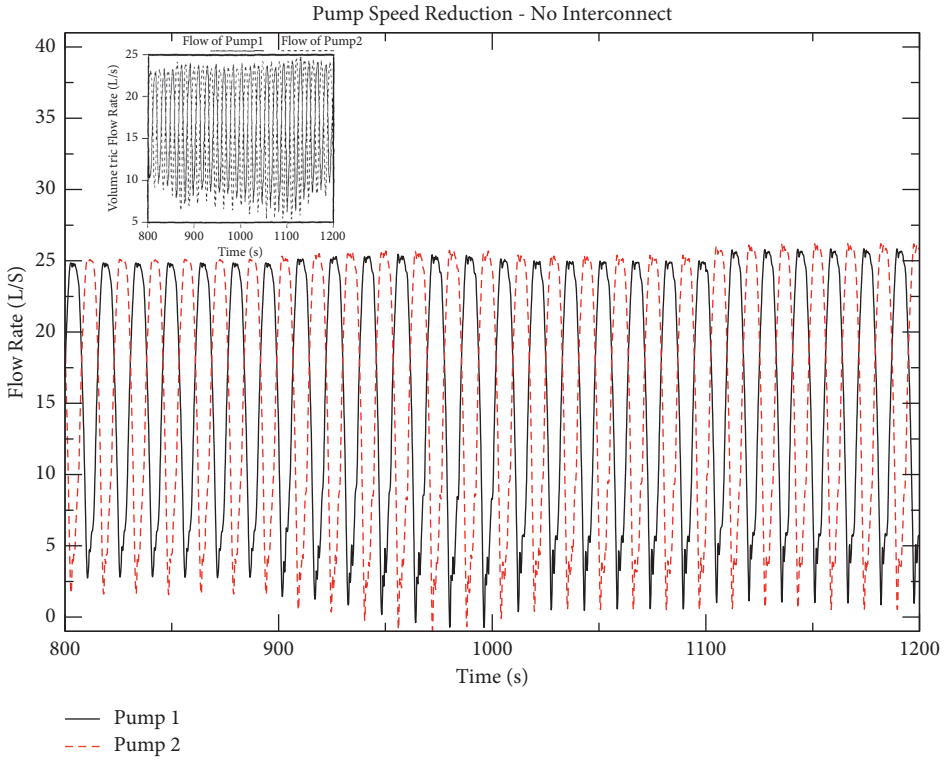

Figure 35: Flow oscillations with reduced pump speed. Experiment inset [21].

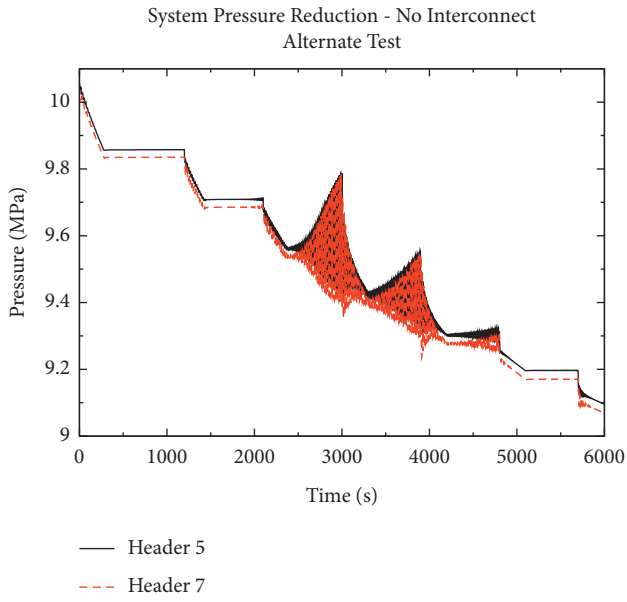

Figure 36: Flow oscillations resulting from system pressure reduction.

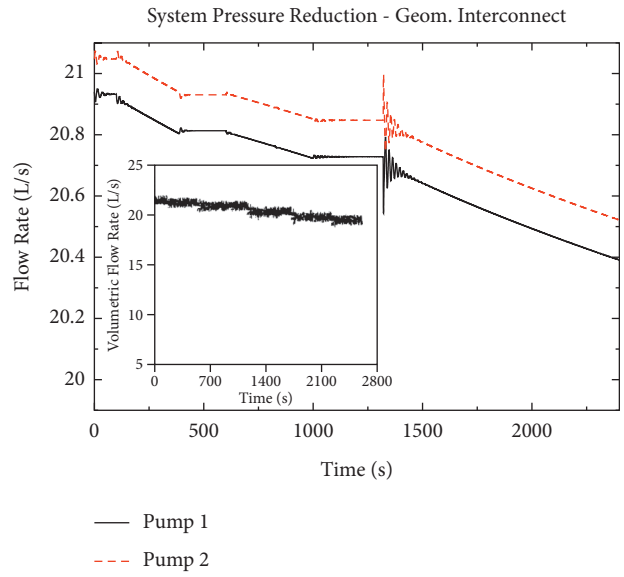

Figure 37: Flow oscillations resulting from system pressure reduction with geometric similarity interconnect.

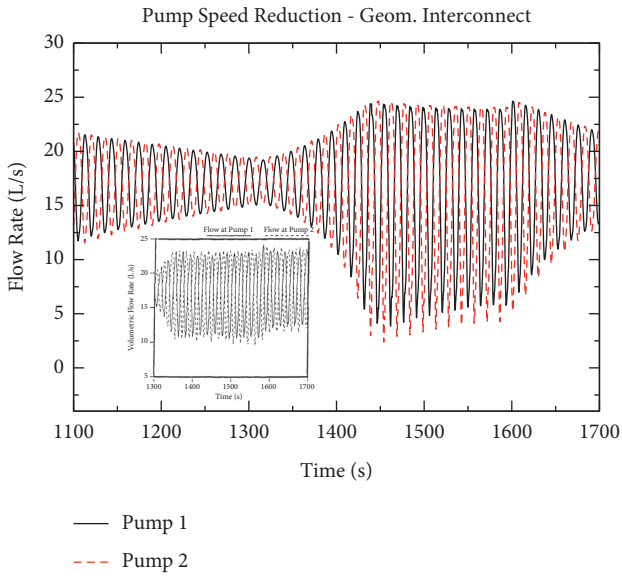

Figure 38: Flow oscillations with reduced pump speed with geometric similarity interconnect. Experiment inset [21].

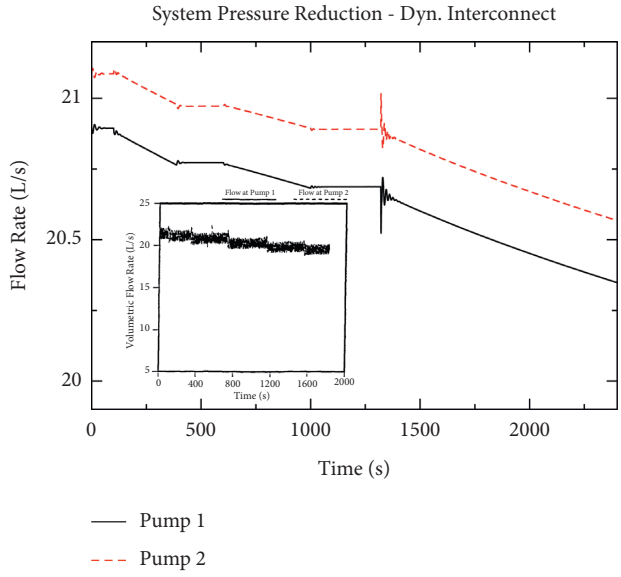

Figure 39: Flow oscillations resulting from system pressure reduction with dynamic similarity interconnect.

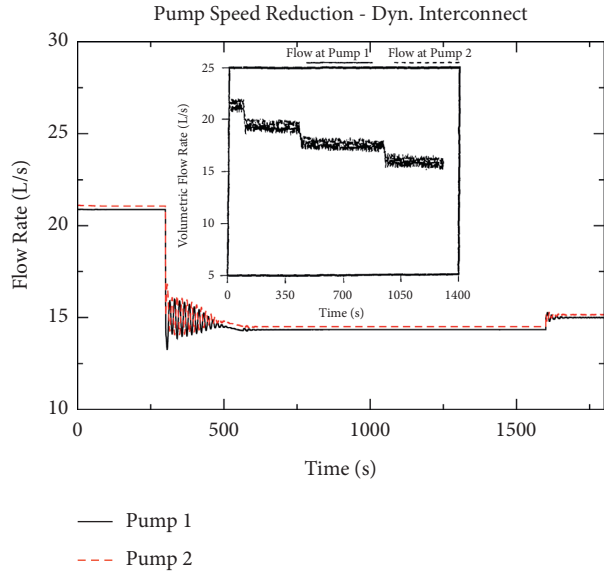

Figure 40: Flow oscillations with reduced pump speed with dynamic similarity interconnect. Experiment inset [21].

Overall, these results, combined with the results on the 900 MW CANDU model, suggest that TRACE_Mac1.0 code can simulate the CANDU figure-of-eight flow oscillation phenomenon (while the work was carried out using TRACE_Mac1.0, the ability for TRACE as distributed to model this phenomenon should be identical.). However, quantitative differences between the RD-14M simulations and experiments suggest that the current RD-14M TRACE model contains inaccuracies that affect the simulation of flow oscillations. The investigation of these inaccuracies is beyond the scope of this work.

4. Conclusions

This work demonstrates that the built-in coupling between PARCS_Mac1.0 and TRACE_Mac1.0, when combined with coupling additional models using ECI, can be used to model a CANDU unit with minimal changes and additions to PARCS and TRACE themselves. This work was able to quantitatively reproduce the results obtained for the same CANDU-6 model running with external coupling while improving the computational efficiency of the model on the whole, particularly for transient analysis. In addition, successful coupling with a reactor regulating system model using ECI was demonstrated as it was able to qualitatively reproduce the behavior of CANDU RRS and show similar results to prior analyses with RELAP5 and other codes.

Certain source code modifications were required to achieve the desired coupling, most notably, additional signal variable functionality for coupled PARCS_Mac1.0-TRACE_Mac1.0 simulations, direct ECI coupling to signal variables, enabling noncoupled POWER components to function in a coupled simulation, and enabling the PARCS-TRACE coupling of control device positions in TRACE_Mac1.0.

As additional programs were developed modularly, the programs can be adapted to different applications. In particular, the ECI Python package is completely independent of the RRS module and its components, and therefore, they may be developed separately, and several other auxiliary programs were developed utilizing the ECI Python package. Both programs may be adapted to future applications, including the simulations of reactor operation, core follow analysis, other safety analysis cases, and uncertainty analysis.

Data Availability

The TRACE and PARCS model data used to support the findings of this study have not been made available due to the confidentiality of the models.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was supported by the University Network of Excellence in Nuclear Engineering (UNENE) and the Natural Sciences and Engineering Research Council of Canada (NSERC). The authors would like to thank Feng Zhou, Kai Groves, and Michael Tucker for their prior work in developing the TRACE and PARCS CANDU models which were used in this research. They would also like to thank Anatol Mysen and David Hummel for their prior work in developing the TRACE RD-14M model used in this research.

References

[1] T. Downar, A. Ward, and Y. Xu, *PARCS v3.3.1 Release Volume I*, Input Manual, Washington, DC, USA, 2018.

[2] United States Nuclear Regulatory Commission, *Trace V. 5.1262 User's Manual Volume 1*, Input Specification, Washington, DC, USA, 2019.

[3] K. Groves, *A TRACE/PARCS Coupling, Uncertainty Propagation and Sensitivity Analysis Methodology for the IAEA ICSP on Numerical Benchmarks for Multi-Physics Simulation of Pressurized Heavy Water Reactor Transients*, McMaster University, Hamilton, Canada, 2020.

[4] R. Miró, P. Ana, T. Barrachina, J. C. Martínez-Murillo, C. Pereira, and G. Verdú, *Implementation of the Control Rod Movement Option by Means of Control Variables in RELAP5/PARCS v2.7 Coupled Code*, United States Nuclear Regulatory Commission, Washington, USA, 2014.

[5] R. Pericas, K. Ivanov, F. Reventós, and L. Batet, "Code improvement and model validation for Ascó-II Nuclear Power Plant model using a coupled 3D neutron kinetics/thermal-hydraulic code," *Annals of Nuclear Energy*, vol. 87, pp. 366–374, 2016.

[6] R. Pericas, "Contribution to the validation of best estimate plus uncertainties coupled codes for the analysis of NK-TH nuclear transients," Doctoral thesis, Universitat Politècnica de Catalunya, Barcelona, Spain, 2015.

[7] B. Rouben, "RFSP-IST: the industry standard tool computer program for CANDU reactor core design and analysis," in *Proceedings of the 13th Pacific Basin Nuclear Conference*, Shenzhen, China, October 2002.

[8] B. N. Hanna, "CATHENA: a thermalhydraulic code for CANDU analysis," *Nuclear Engineering and Design*, vol. 180, no. 2, pp. 113–131, 1998.

[9] W. S. Liu, A. Tahir, E. Zaltsgendler, W. Kelly, and R. K. Leung, *Development Status of TUF Code*, Ontario Hydro Nuclear, Toronto, Canada, 1995.

[10] A. Oussoren and A. Delja, *International Agreement Report Assessment of Channel Coolant Voiding in RD-14M Test Facility Using TRACE*, NUREG, Ottawa, Canada, 2014.

[11] D. Naundorf, J. Yin, and A. Petruzzi, *RELAP5 Simulation of Darlington Nuclear Generating Station Loss of Flow Event*, NUREG, Washington, USA, 2011.

[12] United States Nuclear Regulatory Commission, *The Exterior Communications Interface*, United States Nuclear Regulatory Commission, USA, 2004.

[13] United States Nuclear Regulatory Commission, *MAPTAB Input Manual and User Guide*, United States Nuclear Regulatory Commission, USA, 2017.

[14] E. G. Price, C. R. Boss, W. Z. Novak, and R. W. L. Fong, "Absorber materials in CANDU PHWR's," Technical Report, Atomic Energy of Canada Ltd., Mississauga, Canada, 1995.

[15] P. Peterson, "F2PY: a tool for connecting Fortran and Python programs," *International Journal of Computational Science and Engineering*, vol. 4, no. 4, pp. 296–305, 2009.

[16] J. Koclas, *Reactor Control and Simulation*, Montréal: École Polytechnique de Montréal, Montreal, Canada, 1996.

[17] M. Krause, A. Trottier, and International Atomic Energy Agency, *Numerical Benchmarks for Multiphysics Simulation of Pressurized Heavy Water Reactor Transients*, International Atomic Energy Agency, Vienna, Austria, 2018.

[18] F. Zhou and D. R. Novog, "RELAP5 simulation of CANDU Station Blackout accidents with/without water make-up to the steam generators," *Nuclear Engineering and Design*, vol. 318, pp. 35–53, 2017.

[19] W. J. Garland and S. H. Pang, "CANDU-600 heat transport system flow stability," *Nuclear Technology*, vol. 75, no. 3, pp. 239–260, 1986.

[20] S. Younan and D. Novog, *Simulation of RD-14M Flow Stability in TRACE*, McMaster University, Hamilton, Canada, 2021.

[21] P. J. Ingham, A. J. Melnyk, and T. V. Murray, "Two-phase flow stability experiments in RD-14M," in *Proceedings of the Canadian Nuclear Society—12th Annual Conference*, pp. 33–39, Saskatoon, Canada, June 1991.

6 Paper 3

6.1 Publication Details

S. Younan, D. Novog, "Modelling and Validation of CANDU Shim Operation Using Coupled TRACE/PARCS with Regulating System Response," *submitted to Science and Technology of Nuclear Installations.*

The additional code modifications to TRACE, PARCS, and the RRS model from Chapter 5, were performed by the first author (Simon Younan). The TRACE and PARCS CANDU models used in this work are the same 900 MW CANDU models used in Chapter 5, originally developed by Michael Tucker for prior studies and adapted for the study for Chapter 5. These models have been further adapted by the first author for more consistency with the CANDU 900 reactor being simulated in this work and to perform long-duration transients. The first author developed the driver scripts responsible for executing the shim operation transients and collecting the results. The research supervisor (Dr. David Novog) was responsible for research conceptualization, project administration, and obtaining code licenses. Both authors worked together to select sensitivity cases to analyze. The paper was written by the primary author, while Dr. David Novog provided support in reviewing and editing the paper.

6.2 Preface

In addition to performing safety analyses on short to medium term events, it is often necessary to analyze longer-term operation of the CANDU reactor for safety or economic purposes. As the CANDU reactor operates, the fuel in the core is depleted, channels are refuelled, and the reactor regulating system (RRS) responds to reactivity changes, primarily by adjusting the liquid zone compartment water levels to both maintain a power setpoint as well as to maintain a balanced flux distribution in the core. It is possible to simulate and analyze both normal operation as well as off-normal events.

One possible event is a shim operation event. It is sometimes necessary to perform maintenance on the fuelling machines while the reactor is operational. In this case, the reactor needs to operate for an extended duration with no refuelling. The core reactivity goes down as the fuel depletes, and this is compensated by draining the liquid zone compartments. When additional reactivity is needed, adjuster rods, normally in-core for flux flattening, are withdrawn. The reactor power is correspondingly decreased to

account for the loss of flux flattening. This operation is continued, with one bank of adjuster rods being withdrawn at a time, until it is possible to refuel again.

In this work, the shim event that is simulated is based on a shim event at a CANDU 900 reactor, where refuelling was available only intermittently. The reactor was fuelled ahead, and moderator poison used to compensate for the excess reactivity. As the operation proceeded, the moderator poison was removed, and the first three banks of adjuster rods were then withdrawn in sequence.

The goal of the simulation was to model the different phenomena that affect shim operation, reproduce the transient as closely as possible, as well as evaluate the sensitivities that affect shim operation. The key results to reproduce included the timeline of the shim operation as well as the spatial distribution of reactivity, indicated by the liquid zone compartment levels required to spatially control the reactor. In addition to running a base case as a best estimate to reproduce the transient, a number of sensitivity cases were executed to evaluate different uncertainty components and the significance of different modelling parameters, including adjuster rod depletion, reactivity device worths, nuclear data uncertainty, and burnup distribution of the fuel. Nuclear data uncertainty effects were determined by running Sampler on the SCALE/TRITON lattice model to generate a set of perturbed homogenized cross-section tables for PARCS, then running the coupled simulation for each perturbation. A comparison was also made between the initial state modelled in this work to the corresponding initial state modelled by the station operators using SORO.

In addition, the simulation was extended to evaluate the effect of a power reduction to 59%FP at that point in the shim operation. Such a power reduction initiates a transient predominantly driven by xenon and RRS. The outcome of such a transient for both the reference case simulation as well as the sensitivity cases was evaluated.

In contrast to the transients performed for Chapter 5, which were primarily driven by thermalhydraulic phenomena, the shim operation is primarily reactor physics driven, thus the reactor physics model used for Chapter 5 needed to be modified for the specific CANDU 900 being simulated. The model was updated to include a 13 bundle nodalization, appropriate extrapolation radii for the radial reflector, along with reactivity device positioning and specifications consistent with the station. The RRS model was also modified to incorporate a model of the fully instrumented channels used for thermal power measurements and for calibration of fission power measurements derived from in-core flux detectors.

6.3 Summary of Contributions

This work is built upon the prior work for Chapter 5, whose contributors are listed in Section 5.3. I performed additional minor source code changes for this work, primarily in PARCS to correct inconsistencies in the use of history files as well as to add a transient depletion capability to update the fuel burnups after each time step of a transient calculation. I also made further updates to the PyECI package as well as to the RRS model, with the most significant change being the incorporation of the fully instrumented channels (FINCHes) to the power calibration routine.

The 900 MW CANDU model, which was originally developed by Michael Tucker for his thesis work and adapted by myself for the work in Chapter 5, has been further adapted by myself for this work. The model was adapted for a 13 bundle nodalization, as well as ensuring that the calandria and reactivity devices were modelled consistently with the specifications provided for the CANDU 900 station being simulated.

Collaboration with the utility was necessary to obtain the necessary data from the shim operation as well as to update the model for consistency with the station. Communication was primarily through Dr. David Novog, though I determined the data that was needed for the work. Multiple meetings were held to provide progress updates and determine what additional data would be useful to continue progress.

Michael Tucker was responsible for running the lattice physics models, both the infinite lattice model using SCALE/TRITON and the NEWT transport solver for reference, history, and branch calculations, including perturbed nuclear data calculations using Sampler, as well as the super-cell model using Serpent to generate reactivity device incremental cross-sections. The resulting PMAXS files generated by him were provided to me for use in my work. Tucker also performed simulated core follows based on his research work, providing burnup snapshots that I could use for this work, for both the reference nuclear data and perturbed nuclear data. This arrangement was performed as the prior work in developing a simulated core follow methodology was already performed by Tucker, thus avoiding the repetition of work, and the use of consistent PMAXS files between the simulated core follow and subsequent simulated shim operation was desired.

I collaborated with Dr. Novog to determine the sensitivity cases that would be modelled. I was responsible for setting up and executing the driver scripts for the shim operation simulations. This included implementing the methodology to execute steady-state, depletion, and transient simulations as needed by the shim operation, set up sensitivity cases, as well as collect and present the results.

The writing of the journal paper, including the presentation of the results, was my own work, with editorial support from Dr. Novog. Further modifications are expected during the peer review process.

Science and Technology of Nuclear Installations

Modelling and Validation of CANDU Shim Operation Using Coupled TRACE/PARCS with Regulating System Response

Simon Younan,[1] David R. Novog,[1]

[1] Department of Engineering Physics, McMaster University, Hamilton L8S 4L8, Canada.

Correspondence should be addressed to Simon Younan; younans@mcmaster.ca

Abstract

In CANDU reactors, shim operation is used when the online refuelling capability becomes temporarily unavailable. Adjuster rods, normally in-core to provide flux flattening, are withdrawn in sequence to provide additional reactivity as the fuel is depleted. In a CANDU 900 reactor, up to three of the eight adjuster banks may be withdrawn, with the power derated accordingly.

In this study, the shim operation was modelled using a combination of TRACE_Mac1.1, PARCS_Mac1.1, and scripts modelling the reactor regulating system, all running as a single coupled simulation. A driver script simulated the operation as a sequence of steady-state, depletion, and transient models. The results were compared to operational data from a Nuclear Power Plant, evaluating the key figures of merit. The simulation was extended beyond the operational data by reducing the power to 59%FP and withdrawing the remaining adjusters, to observe the behaviour of the simulated reactor for a deeper reactivity driven transient. Sensitivity cases, including adjuster rod depletion and nuclear data uncertainty, were also evaluated.

This study was able to successfully reproduce the general results of the shim operation. Some discrepancies were observed between the simulation and dataset for the duration of shim, particularly for the one-bank-out phase of shim. Several potential causes for the early phase behaviour were identified.

When the simulation was extended, the model predicted that a power reduction below 60%FP would lead to xenon poison-out when the adjusters were depleted, with the timing sensitive to the adjuster depletion. Nodalization of the PARCS model also had a significant impact, due to the effect on adjuster nodalization and its area-of-effect with respect to the actual adjuster locations. Nuclear data uncertainty had a lesser but still noticeable effect. Other parameters, such as the distribution of fuel burnups in the core, only had a small effect on the shim operation.

Introduction

The CANDU reactor has unique characteristics in its operation, primarily due to the use of natural uranium fuel and heavy water moderator. Most notably, CANDU reactors are fuelled on-line, with the capability to refuel individual fuel channels multiple times per day while at power. This "continuous refuelling" allows for excess reactivity to be kept at a minimum, improving the overall neutron economy to achieve a higher exit burnup out of the natural uranium fuel. CANDU reactors are also loosely spatially coupled, thus requiring spatial reactivity control to maintain a nominal flux distribution with no tilts. When modelling long-term reactor operation, it is important to model the core evolution, including depletion, saturating fission products, and the actions of the reactor regulating system (RRS) [1].

During normal reactor operation, the power of the reactor is controlled primarily using 14 liquid zone compartments. These may be filled and drained with light water, which acts as an absorber in a heavy water moderated reactor. The RRS liquid zone control algorithm acts to control both the bulk reactor power as well as its spatial distribution and can counteract the reactivity imbalances that result from refuelling along with other minor reactivity perturbations. Adjuster rods, normally inserted, are used to flatten the flux profile in the core and may be withdrawn to provide additional reactivity when necessary. Mechanical control absorbers, normally withdrawn, are used to provide negative reactivity, and may also be partially or fully dropped to the core to rapidly reduce power without fully shutting down the reactor. Moderator poison may also be used to introduce negative reactivity.

On-line refuelling is achieved using two fuelling machines, one on either end of the reactor. When refuelling a channel, one fuelling machine will push new fuel bundles into a channel, while the other machine will receive the used bundles. One possible operational occurrence is a fuelling machine outage, where a fuelling machine is unavailable due to maintenance and thus the reactor cannot be refuelled. In this scenario, the reactor may be operated in a "shim operation" mode, using the adjuster rods for reactivity shim. During the initial stage of shim operation, the reactivity loss due to depletion is compensated by draining the liquid zone compartments. When additional reactivity is required beyond what the liquid zone control can provide, banks of adjuster rods are sequentially withdrawn. This is continued until the reactor is either shut down or refuelling resumes.

A number of previous studies have been performed on adjuster rods, shim operation, and/or other adjuster-driven transients under various conditions. Some examples include:

- Evaluating reactivity devices using DUPIC fuel compared to natural uranium fuel [2]. The study modelled CANDU-6 reactivity devices and shim operation using coupled RFSP and NUCIRC, finding that the maximum shim duration was significantly shorter for DUPIC fuel.
- Evaluating reactivity devices using DUPIC fuel for xenon override in a CANDU-6 [3], using RFSP, finding comparable xenon override capability to the natural uranium cycle.
- Optimization of adjuster rods in thorium-based fuel cycles to achieve comparable xenon override capability and shim duration to natural uranium fuel cycles [4], using DRAGON and DONJON.

Unlike these studies, which used CANDU-specific tools and evaluated theoretical shim operation performance for advanced fuel cycles when compared to natural uranium, this study focused on implementing the methodology using the PARCS and TRACE codes, which are typically used for analysis of light water reactors rather than CANDU reactors, as a basis, and modifying them to support the necessary coupling capabilities. This study involved simulating the shim operation of a 900 MW class CANDU reactor using a standard natural uranium fuel cycle and comparing with operational data, to demonstrate that these tools could be adapted for modelling long CANDU transients and considered for analysis of either actual or theoretical reactor transients. In addition, the use of coupled transient simulations to supplement quasi-steady state calculations is a novel contribution, as it allows for the reactor dynamics to be properly captured at key points in an otherwise quasi-static reactor evolution.

The simulation and analysis of reactor operation may be performed using one of two methodologies based upon the evolution of the reactor state. If the reactor fuel composition and transients evolve very slowly, then the state of the reactor at any given time can be modelled as a quasi-steady state. Under such a simulation, the burnup and saturating fission product state of the core, which evolves slowly, is held constant, while more rapidly evolving variables such as the neutron flux and thermalhydraulic conditions are solved for their equilibrium values. The flux is then held constant and used to advance the burnup and saturating fission product state by a given period of time. If the core state evolves rapidly, the reactor is simulated as a transient, with the neutron flux and thermalhydraulic conditions evolving with time and effects of fuel composition changes are secondary.

Materials and Methods

Background

The CANDU reactor is a pressurized heavy water reactor (PHWR) design, differing significantly from the more popular light water reactor (LWR) designs. Some key traits of the CANDU design that differ from the PWR design include [5]:

- Heavy water used as the coolant and moderator.
- Natural (unenriched) uranium as the fuel.
- Horizontal fuel channel arrangement, with fuel channels serving as the pressure boundary. Coolant and moderator are physically separated, with the fuel channels surrounded by the moderator in an unpressurized calandria.
- Fuel channels have a significant degree of separation, to allow for moderation by heavy water, access by the fuelling machines, and for the insertion of vertically oriented reactivity devices.
- On-line refuelling, through the use of two fuelling machines which can latch onto either end of a fuel channel to insert and discharge fuel bundles while the reactor is operating at power.
- Reactivity control primarily using 14 liquid zone compartments, whose light water levels can be varied independently to control reactor power and its spatial distribution.

On-line refuelling is important due to the use of natural uranium fuel. This fuel has far less excess reactivity when compared to enriched uranium fuel, leading to lower exit burnups and core residence times for CANDU fuel when compared to LWR fuel. A batch refuelling scheme for a CANDU reactor would require a refuelling outage every few months [5], which would greatly diminish the station's capacity factor. As well, the continuous refuelling allows for the excess reactivity on a core-wide basis to be kept at a consistent minimal value, improving the overall neutron economy and thus the exit burnup.

On-line refuelling is achieved using a pair of fuelling machines, with one on each end of the reactor. Each fuelling machine is capable of latching onto one end of a fuel channel, removing the channel closure and shield plug, then either inserting new fuel bundles or receiving used fuel bundles, before replacing the shield plug and channel closure [5]. Typically, not all fuel bundles in a channel are replaced, but rather only 4 or 8 bundles (out of 12 or 13 bundles) are replaced, with the remaining bundles closer to the inlet end of the channel remaining in the core but being shifted towards the outlet end of the channel. The horizontal orientation of the fuel channels allows for bidirectional refuelling, where adjacent channels are refuelled in opposite directions, along with coolant flow being in opposite directions, to achieve symmetry in the neutron flux distribution.

This introduces additional complexities in simulating reactor operation which are not present in the simulation of LWR operation. For an LWR, the fuel physical location remains the same over the course of a fuel cycle, with the fuel being depleted over the course of the cycle and the core kept critical through the use of moderator poison, reactivity devices, and/or negative power feedbacks, depending on the type of reactor. In a CANDU reactor, the fuel distribution changes on a daily basis as individual channels are refuelled. Thus, high fidelity simulations require a full accounting for both effects of burn-up and fuel relocation due to fuelling prior to the event of interest, as well as the transient of interest. Different methods for performing the core follow are as follows:

- Using actual station data, the operating history of the reactor is simulated, from the start of operations or from a given snapshot in time to a later snapshot in time using the fuelling and operating histories of that core. This may be performed using any diffusion code, with the industry typically using RFSP, but studies have also been done using DONJON [6], to model the core state leading up to a transient.
- Starting with an initial core of fresh fuel, reactor operation is simulated, including the simulation of fuel channel selection for refuelling, until the core reaches an equilibrium. This typically takes 400 to 500 full-power days [5]. Such fuelling simulations have been previously performed using PARCS [7].

When simulating reactor operation, it is thus often important to have frequent snapshots which capture the salient core features at each discreate point in time. For a given snapshot, the slow-changing quantities, such as burnup, may be held constant, and the equilibrium state of the reactor determined. Then, a depletion step may be performed, where the slow-changing quantities are updated under the assumption that the equilibrium state of the reactor changes little between successive snapshots. The burnups in one or more channels would then be updated to simulate refuelling. Depending on the model and particularly on the time between snapshots, the

saturating fission product densities may be treated as a slow-changing quantity to calculate transiently, or as a fast-changing quantity to calculate as an equilibrium. For Xe-135, the density reaches an equilibrium on the order a day for a reactor at full power, based on an approximately 6.7-hour half-life of I-135 [5]. Thus, for daily or less frequent snapshots, an equilibrium xenon model is reasonable except for recently refuelled channels.

In addition to determining the flux distribution in a core snapshot, it is also important to model the reactivity devices, since reactivity devices have an important effect of the flux distribution. The following are used to control reactivity in the CANDU design:

1. 14 liquid zone compartments (LZCs) distributed throughout the core. These are filled with light water, which acts as a neutron poison due to its high absorption cross-section compared to heavy water. The unfilled portion of each compartment is filled with helium gas. The liquid levels are varied automatically by the reactor regulating system (RRS) running on a digital control computer (DCC). Both bulk control and spatial control are used. Bulk control adjusts the liquid levels equally in all 14 compartments to add positive or negative reactivity and control the reactor power. Spatial control adjusts the liquid levels independently to balance the spatial distribution of the flux. [1]
2. Adjuster absorbers, which are normally inserted in the inner part of the core to provide flux flattening. Flattening of the flux distribution is desirable as it allows for the total reactor power to be increased without exceeding power limits for individual fuel channels or fuel bundles. Adjuster absorbers are grouped into eight banks. The RRS will withdraw one or more banks of adjusters if additional positive reactivity is required, based on either a low average liquid zone level (indicating that the LZCs cannot provide enough reactivity) or a large negative power error (indicating that positive reactivity is needed more rapidly) [1]. Withdrawing of some of the adjusters requires the reactor power to be derated due to a loss of the flux flattening effect.
3. Mechanical control absorbers (MCAs), which are normally positioned out of the core. The RRS will insert one or more banks of MCAs if additional negative reactivity is required, based on either a high average liquid zone level (indicating that the LZCs cannot provide enough reactivity) or a large positive power error (indicating that negative reactivity is needed more rapidly). [1]
4. Moderator poison, either boron or gadolinium. Boron is used on initial start-up to depress the excess core reactivity from a full load of fresh fuel during the first weeks of operation. Gadolinium is used to compensate for transient excess positive reactivity beyond the worth of the LZCs and MCAs, particularly to manage the xenon transient resulting from an outage and subsequent restart. [5]

For managing the normal operation of the core, the LZCs are most important, with the other reactivity device states typically not being actuated/changed. The LZCs are used to manage small reactivity perturbations, particularly those that occur from depletion and refuelling. When a channel is refuelled, not only does the new fuel introduce positive reactivity, but it also locally increases the flux near the refuelled channel. For a large reactor such as the CANDU-9 reactors, the core radius is far greater than the neutron migration length, and the reactor is susceptible to certain flux harmonics, particularly the first azimuthal (side-to-side or top-bottom) and first axial (end-to-end) modes [8]. Therefore, spatial control, via the LZCs, is required to control these flux

modes. Conceptually, the reactor is divided into 14 spatial control zones, with one LZC corresponding to each zone. Spatial control is used to maintain a design ratio of measured powers between the 14 zones, suppressing any tilts resulting from refuelling, xenon perturbations, or the movements of other reactivity devices. When calculating a snapshot, the 14 LZC water levels must be determined, both to achieve criticality as well as to eliminate any flux tilts.

It is also of interest to simulate operating conditions other than normal operation at full power. One such example is for the simulation of shim operation. CANDU on-line refuelling relies on a pair of fuelling machines. If the fuelling machines are not available, then the reactor will need to be operated without refuelling. In shim operation, the negative reactivity due to depletion is counteracted by the LZCs and the adjusters. Reactor power must be derated prior to withdrawing adjuster banks to counteract the resulting increase in channel and bundle power peaking factors. These power reductions also lead to xenon perturbations that must be managed by the LZC system. Once the fuelling machines are available, the reactor can gradually be refuelled until it reaches its normal level of reactivity. If the fuelling machines are unavailable for an extended period of time, the reactor may need to be shut down. In the case of planned maintenance, extra refuelling can be performed prior to shim operation, using LZCs or moderator poison to compensate for positive reactivity. This will extend the possible duration of shim operation.

Figure 1 provides the liquid zone designation for the CANDU 900 reactor used in this work.

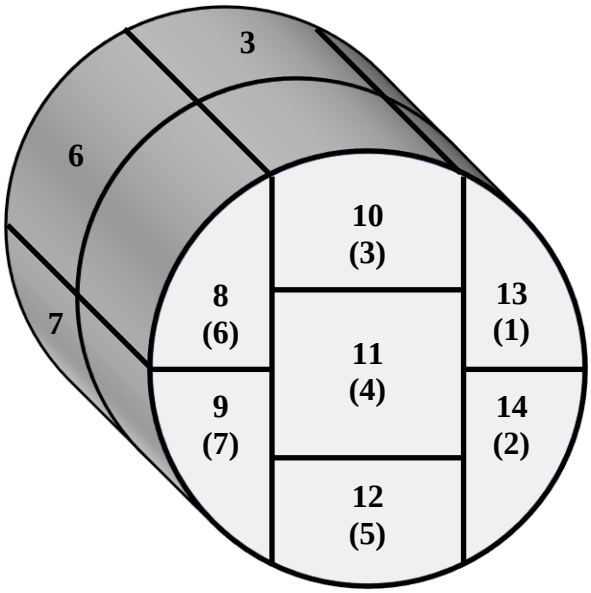

Figure 1: Liquid zone number designation for CANDU 900

Methodology for Modelling and Simulation

This work uses the TRACE [9] and PARCS [10] computer codes, licensed through the NRC's Code Application and Maintenance Program (CAMP), to simulate system thermal-hydraulics and core physics, respectively. More specifically, this work uses modified versions of the codes, hereafter named TRACE_Mac1.1 and PARCS_Mac1.1. It also uses the Exterior

Communications Interface (ECI) [11] to couple an RRS model to PARCS through TRACE. The RRS model was developed in the Python programming language and uses a Python ECI library built around the original Fortran-based ECI library. This coupled model has been previously evaluated against shorter transient events such as figure-of-eight flow oscillations (using a modified system model without header interconnects) and a loss-of-flow transient initiated by a loss of Class IV power [12]. This section covers the additional work performed to model long-duration transients such as adjuster shim operation.

For modelling of CANDU adjuster shim operation, there are two simulation regimes to consider. During the majority of the operation, the core state evolves slowly, and the only significant effects to consider are the effects of depletion and saturating fission product evolution. The second regime corresponds to a time-dependent situation where device configuration (e.g., adjuster positions and liquid zone levels) change over a relatively short period of time due to a change in RRS control actions.

During the slowly evolving quasi-steady state periods the reactor can be modelled as a steady-state simulation. During this simulation, the depletion, saturating fission product densities, and reactor power are held at constant values, while the spatial flux distribution, thermalhydraulic state, and control system state are computed. To model the core evolution during these steady-state steps, a PARCS depletion calculation is performed with PARCS_Mac1.1, with the reactivity device and thermalhydraulic feedbacks held constant. The depletion calculation is used to calculate an updated depletion and saturating fission product density state for the next steady-state step. This methodology is very similar to a core follow, as it generates periodic core snapshots modelling the evolution of the core over a period of time. Core follow analysis has been previously performed using both PARCS [7] as well as other core physics tools, particularly RFSP.

For the steady-state model, certain simulation parameters have been adapted to support eigenvalue calculations, accelerate model convergence, and improve performance:

- The power error calculation module (CEP) may override the power error with either a k_{eff} error or a zone level error for steady state analysis, to converge either the system k_{eff} or average zone level to a target value. The zone levels may also be frozen to a set of known values.
- The PARCS eigenvalue calculation is performed only once every 20 TRACE time steps. This greatly accelerates the simulation as the PARCS eigenvalue calculation is significantly more computationally intensive than the TRACE time step. Sensitivity studies demonstrate that the transients predicted here are not sensitive to the exchange rate in this regard, as long as the three models (neutronics, thermalhydraulics, RRS) are given sufficient time to converge and that the neutronics-RRS feedback loop is stable.
- The RRS time step size is decoupled from the TRACE simulation and set to a fixed constant. This allows for the RRS to converge more quickly when the TRACE time step size is small. The maximum step size is limited by stability due to the feedback between the RRS and PARCS models and is currently set to 4 seconds per PARCS step (0.2 seconds per TRACE time step).

- Convergence of thermal calibration of the ion chambers and in-core flux detectors (ICFDs) is greatly accelerated. Calibration is applied every two seconds instead of every 16 seconds, and the time constants of the filters in the thermal calibration routine are decreased to 1.875 seconds (instead of 30 seconds or 180 seconds).
- Calculation of the delayed components in the ICFD response is performed using the flux from the steady-state calculation of the previous depletion step and the flux from the current calculation, under the assumption that the flux changed linearly over the course of the previous depletion step. The ICFD detector response in this work is based on the response characteristics calculated in reference [13], with the four normalized cases averaged. The resulting prompt fraction was 1.048 (4.8% overprompt), and the response also includes four different delayed components with time constants ranging from minutes to days.

This approach is very similar to RFSP's asymptotic zone control functionality [2], except that the liquid zone control module functions identically in the steady state and transient modes, only manipulating its inputs (bulk power error and time step size) to implement the steady-state mode. This guarantees that the asymptotic zone levels are consistent with transient modelling, as long as the power error converges to zero.

The second regime corresponds to time-dependent actuation of the device positions caused by RRS or operator actions. As the reactor is depleted, the reactor regulation system is expected to decrease the average liquid zone level in order to maintain criticality. Once the liquid zone level reaches a certain threshold, a bank of adjuster rods is to be withdrawn. The dynamic nature of the adjuster rod withdrawal process is simulated using a coupled transient model with transient depletion and saturating fission product evolution. In advance of withdrawing the adjuster bank, the reactor power will be derated by the operators to an appropriate level for withdrawing the adjuster bank consistent with station operating rules. This is done at a higher liquid zone level threshold, to allow the xenon-135 transient to trigger the adjuster withdrawal and provide enough reactivity margin so that only one adjuster bank is withdrawn. The full transient procedure is thus as follows:

1. As the zone level nears a threshold where action to remove adjuster may be required, the simulations will decrease reactor power to the setpoint for the next adjuster bank to be withdrawn to mimic operator actions.
2. The simulations will then continue in time to allow average LZC level to decrease to the adjuster withdraw threshold as the xenon transient progresses.
3. The simulations will then withdraw an adjuster bank at the time and rate dictated by the design parameters. Average LZC level will increase to maintain criticality.
4. Average LZC level will continue to decrease to a minimum value corresponding to peak xenon, then subsequently increase as xenon density returns to a new equilibrium value.
5. As xenon approaches equilibrium, average LZC level will begin to decrease again once depletion becomes the dominant effect.

For this work, the coupled transient is simulated from the start of the power derate until shortly after each adjuster bank is withdrawn. The most rapid changes in flux distribution and reactivity device positions occur during adjuster bank movement, and it is desired to model the behaviour

of the flux distribution as the bank moves out of the core. A significant transient flux tilt may degrade safety margins, and if sufficient flux tilt is detected, an RRS setback may be triggered, further reducing power below the intended setpoint. In this model, the setback threshold for flux tilt is set to 20%, based on zone power measurements excluding the centre zones 4/11. Once a setback is initiated, it continues until the flux tilt drops below 18%. Once a bank is fully out of the core, the spatial flux distribution is relatively stable, and the depletion and saturating fission product evolution may be modelled using a steady-state depletion model once again.

Therefore, the overall procedure for the shim simulation is as follows:

1. Perform steady-state simulation and depletion steps while monitoring average zone level (AZL).
2. As AZL approaches threshold to derate power, determine the depletion step size to reach the AZL threshold. Iterate until the calculated AZL is at the threshold.
3. Perform coupled transient simulation of the power derate and subsequent adjuster bank pull.
4. Perform steady-state simulation and depletion steps until the end of the xenon transient.
5. Repeat steps 1-3 for the next adjuster bank.

For step 1, a 6-hour depletion interval has been selected. Once step 3 has been completed, step 4 is performed until 12 hours pass from the end of the previous adjuster bank withdrawal before returning to step 1. The step size starts at 15 minutes and is doubled until it reaches the original 6-hour interval again. The 12-hour delay prevents the next adjuster bank withdrawal being triggered by the AZL threshold due to the xenon transient, ensuring that each adjuster bank movement is triggered due to depletion and not xenon evolution.

Figure 2 outlines a high-level methodology that can be used to implement this simulation. The simulation alternates between performing a coupled steady-state calculation and a standalone depletion calculation, until the average zone level approaches the threshold to perform the adjuster bank withdrawal process. At this stage, a coupled transient run is performed to simulate the power manoeuvre and subsequent adjuster bank withdrawal.

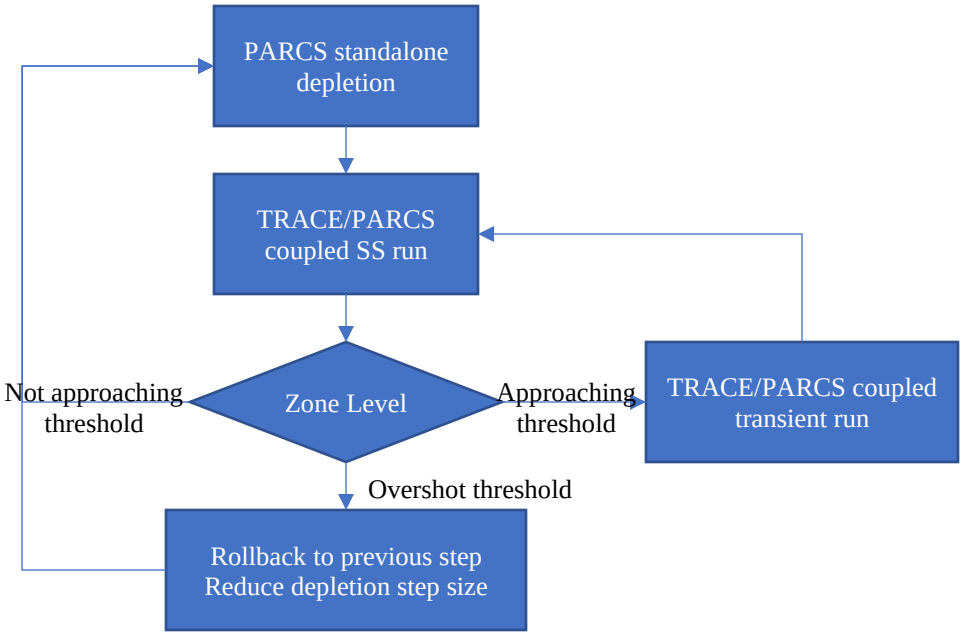

Figure 2: Summary of hybrid methodology for performing a shim transient

A more detailed version of the methodology is provided in Figure 3. While similar to the high-level methodology presented by Figure 2, it includes more detailed logic for determining which action to perform after a steady-state simulation. First, it checks that there were no abnormalities in the simulation, such as a setback, stepback, or reactor trip. If such an abnormality is detected, then the simulation will either revert to before the previous depletion step and repeat it with a smaller depletion interval or will switch to coupled transient simulation. This allows for the simulation to pinpoint the occurrence of such an event and simulate it as a transient simulation. Secondly, the PARCS steady-state simulation is limited to a fixed reactor power, but the indicated power in RRS does not exactly match the power in the PARCS simulation. Therefore, at the end of a steady-state step, the indicated power in the simulated RRS is compared to its setpoint, and if their difference exceeds a certain tolerance, the PARCS power is adjusted, and the steady-state step is repeated.

The simulation must then determine whether it has reached a part of the simulation that should be performed as a fully coupled transient. This occurs when either of the following conditions is met:

1. The average zone level is at or below the threshold to perform the next manoeuvre, which consists of a power derate (if needed) followed by adjuster pull.
2. The average zone level is above the threshold, but the estimated time until the average zone level reaches the threshold is less than the minimum depletion step size (864 seconds, or 0.01 days). This estimate is calculated by extrapolating the average zone level from the previous two steady-state calculations. If the estimate is negative, due to the average zone level increasing over the previous depletion step, then it is ignored.

If neither condition is met, then the next step performed is a depletion step. A nominal depletion step size of 6 hours was selected in this work, but the size of a given depletion step is determined as follows:

1. If the estimated time from step 2 of the previous list is positive, and no greater than the nominal depletion step size plus 15 minutes, then the depletion step size is set to this estimated value. Otherwise, the nominal step size is selected.
2. In addition, for a scheduled action that is not to be simulated using a coupled transient (e.g., refuelling), the depletion intervals are sequentially shortened as the action is approached, up until the action time is reached.
3. The actual depletion step size is the lesser of this selected step size, or twice the step size of the previous depletion step.

This creates the effect of performing fixed depletion intervals until the next transient portion of the simulation approaches, then performing a depletion interval which is estimated to bring the simulation to the starting time of the transient. After a transient is completed, the subsequent depletion step size is set to 1 hour, then allowed to double with each depletion step until it reaches the nominal step size. For actions not requiring transient simulation and that are based on a schedule, finer steps are taken around the time of the action, both before and after. For this transient, several refuelling events are scheduled. To simulate a refuelling, a script is used to read the binary depletion file produced after a depletion step and modify the burnups in the selected channel (pushing 4 or 8 new bundles into the channel and pushing the existing bundles towards the end of the channel, discharging channel out the other end) before rewriting the file.

Since exact controller parameters for the 900 MW CANDU design are not available in literature, realistic approximate or representative values were selected. The following values were used:

- K_P = 31.0 (LZC bulk controller gain)
- K_T = 5.0 (LZC spatial controller gain for flux)
- K_H = 0.6 (LZC spatial controller gain for level, when flux control is phased out)
- K_L = 0 (LZC spatial controller gain for level, independent of flux control being phased in)
- Adjuster movement speed = 10.5 cm/s
- LZC drain rate = 5.0 cm/s
- LZC max net fill rate = 5.0 cm/s
- LZC valve lift/bias for constant level = 60%
- Flux tilt setback threshold = 20%FP (difference from highest to lowest zone power, excludes centre zones 4 and 11)

The LZC drain/fill rate was selected based on taking approximately one minute to fill/drain a "long" zone compartment between 0% and 100% [1], the longest of which in this model is 285.8 cm, resulting in a rate of 4.76 cm/s, which was rounded to 5 cm/s.

The most sensitive parameters are the gains and detailed sensitivity results are performed to determine the effect of the gains on the transient. Outcomes related to reactor setback are also highly sensitive to the flux tilt setback threshold.

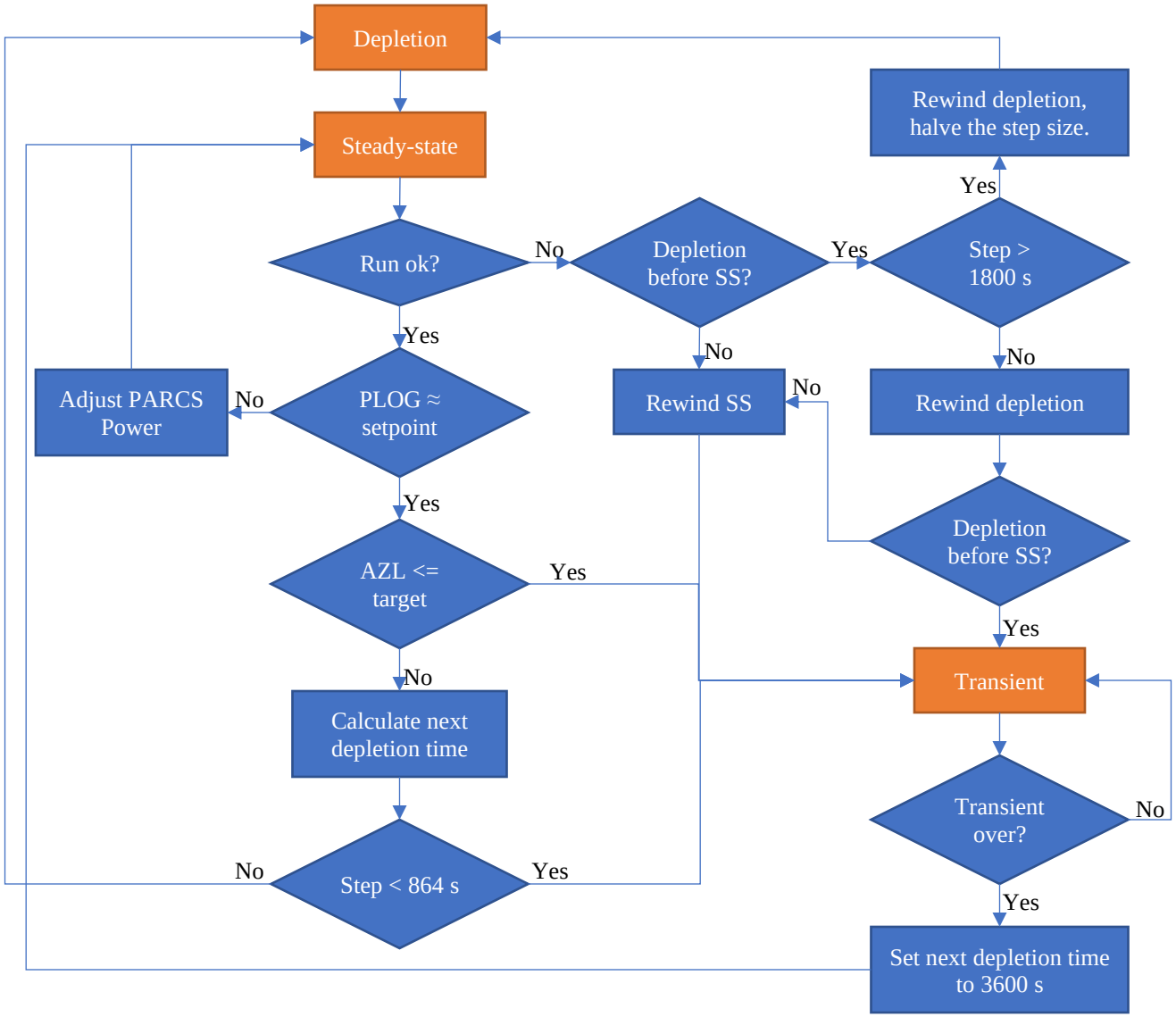

Figure 3: Hybrid methodology for performing a shim transient

Station Description and Modelling

The station design being simulated is a 900 MW class CANDU reactor, based on those used in previous coupled simulations of TRACE_Mac1.0 and PARCS_Mac1.0 [12]. When comparing preliminary results to station data, it was found necessary to update the core physics model for better consistency with the specific CANDU 900 under analysis. This included increasing the number of fuel bundles per channel from 12 to 13 both in the thermalhydraulic and reactor physics models, as well as applying appropriate extrapolation distances as these are known to have a significant impact on full-core modelling outcomes [14] and was found to have a significant impact in this work.

In addition, adjuster rod aging was accounted for by prorating adjuster incremental cross-sections, based on prior industry investigations, to account for their decrease in worth as a function of irradiation. An adjuster rod age of 182,500 effective full-power hours (EFPH) was selected for this work. The effect of aging was to reduce the effective thermal absorption cross-section of the adjuster by approximately 10% to 20%, depending on the adjuster type. Other cross-sections were perturbed by varying amounts.

The RRS model was also updated, particularly to implement an approximate model for fully instrumented channels (FINCHes). The FINCH powers are approximated as the PARCS computed channel powers with a 30 second decay filter. The FINCH powers are compared against a nominal power distribution and are used to calibrate the zone power readings from the ICFDs as part of the RRS control features modelled in these simulations.

In the TRACE model, the 480 channels are reduced down to 40 representative channels, based on zone designation, then subdividing each zone into two regions, being upper and lower half for the centre zone, and inner and outer regions based on average channel power for other zones. All channels in a given region that belong to the same flow loop and pass are grouped into a single channel in the TRACE model. This results in a total of 40 channels in the TRACE model to represent the flow through all 480 channels. The coolant temperature and density, as well as fuel temperature, are shared for all channels in a given grouping.

The 900 MW CANDU design uses flow-power matching to achieve a similar exit enthalpy in channels with varying average channel power, trimming the flow in lower-power channels [5]. To approximate this flow-power matching, a minor loss representing a flow orifice was added to the lower-power channel groups where the inlet feeder connects to the inlet end fitting such that the outlet enthalpy of all channels was approximately equivalent.

For most CANDU reactors, adjuster bank withdrawal typically requires power reductions to ensure thermal margins are maintained during these periods where radial core power peaking is increased. To simulate this, the simulations perform the following power changes with each adjuster bank:

- One bank out: 93.5%FP
- Two or three banks out: 90.0%FP
- More than three banks out: 59.0%FP

Event Timeline

The transient being simulated is a period of shim operation where only intermittent refuelling was available. A summary of events is as follows:

- Day 1: Initial core state. Reactor power setpoint is at 100%FP. Reactivity banking was performed in anticipation of shim operation. A small amount of moderator poison (modelled as equivalent boron) is used to compensate for the excess reactivity.
- Day 5-6: Seven channels are refuelled (32 bundles total).
- Day 13: Reactor power setpoint is decreased from 100%FP to 93.5%FP in accordance with shim operation procedures. Adjuster Bank A is subsequently withdrawn.
- Day 18: Reactor power setpoint is decreased from 93.5%FP to 90.0%FP in accordance with shim operation procedures. Adjuster Bank B is subsequently withdrawn.
- Day 20: There is a brief period where the fuelling machine is available. Four channels are refuelled (16 bundles) to extend shim operation.
- Day 25: Adjuster Bank C is withdrawn.
- Day 29: End state for shim operation.

Figure 4 presents the timeline for reactor power and liquid zone level changes. The depletion trend, three adjuster bank manoeuvres, and the refuelling run are visible in the liquid zone level trend.

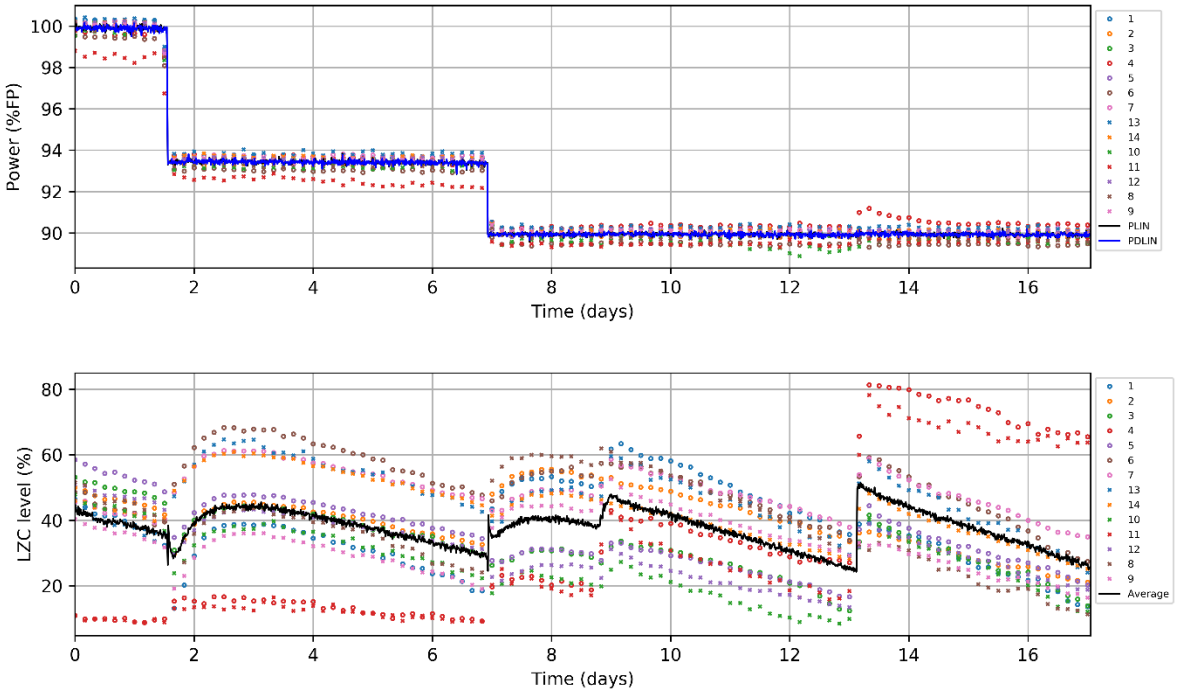

Figure 4: Reactor power and liquid zone level trends for CANDU 900 shim operation. Day 0 in this figure corresponds to the start of day 12 in the event timeline.

To simulate this sequence of events, specific conditions must be identified for triggering the initiation of each event. The simulated sequence is as follows:

- The simulation starts at the initial known burnup distribution, with 0.35 ppm of boron used to represent the equivalent moderator poison that was deemed to be present. The moderator poison is gradually reduced to zero, following the known trend of equivalent moderator poison from station snapshots and linearly interpolating between them.
- The seven channel refuellings on days 5 and 6 are performed consistent with station actions.
- At an average LZC level of 35%, the reactor power setpoint is set to 93.5%FP. The subsequent xenon transient causes the average LZC to decrease rapidly. At an average LZC level of 25%, adjuster Bank A is withdrawn. The xenon transient is allowed to continue until reaching a new equilibrium.
- At an average LZC level of 30%, the reactor power setpoint is set to 90.0%FP. This causes a similar transient as before, and at an average LZC level of 25%, adjuster Bank B is withdrawn.
- While "shim" operation typically does not have fresh fuel inserted, in this operating transient there were 4 fuelling operations. For the simulation, the first channel refuel is performed 1.9 days after the withdrawal of adjuster Bank B is completed. This timing and the interval between refuelling each of the four channels is taken from the operational history of the station.
- At an average LZC level of 25%, adjuster Bank C is withdrawn, increasing the zone level.
- The average liquid zone level continues to decrease from depletion until it reaches a final zone level of approximately 25%, at which point the comparison with the station data ends.
- The reactor power setpoint is set to 59.0%FP, as outlined in the previous subsection, to extend the simulation and evaluate the effect of such a power reduction. This causes a xenon transient that causes the average LZC level to rapidly decrease. The subsequent adjuster bank pulls to compensate are performed each time the average liquid zone level gets too low.
- With all adjusters out, if the average zone level drops to 10%, the reactor is shut down, as it would go subcritical anyway. The simulation is ended when the actual reactor power has dropped below 2.0%FP or when the simulation time has exceeded 6 hours from the power reduction to 59%FP.

Figures of Merit

The following key figures were identified for evaluating the performance of the simulations compared to the station data:

- Event timings (adjuster bank A, B, C withdrawals and subsequently reaching 25% average zone level).
- Trends in average liquid zone levels.
- Trends in individual liquid zone levels.
- Channel power distribution.

In addition, it is possible to calculate reactivity device worths in the simulated cases for comparative study.

For individual liquid zone level trends and channel power distribution, the average zone level of the two central zones (4 and 11) was selected as an indicator of the radial reactivity balance. More specifically, it was checked 5 minutes prior to the end of the station data. This point was selected as the average zone level would be consistent (around 25%) and level for the two central zones was generally within the range of 30% to 70% for both the station data and the simulations.

The following phenomena have a significant effect on these figures of merit as they affect the core reactivity:

- Depletion – the rate of reactivity change due to depletion will govern the overall length of the transient.
- Power coefficient of reactivity – a positive coefficient will result in a reactivity decrease when the power is derated for an adjuster withdrawal, shortening the time to the next subsequent adjuster movement. A negative power coefficient has the opposite effect.
- Void coefficient of reactivity – the CANDU reactor has a positive void coefficient of reactivity. Its effect on the shim transient is similar to the power coefficient, except only at high power, where voiding would be expected near the channel outlets for high-power channels.
- Absorption by reactivity devices – each reactivity device affects the core both through its absorption of neutrons as well as the resulting redistribution of flux in the core.
- Saturating fission products – when the power is decreased, there is an initial negative reactivity change as the concentration of Xenon-135 increases. After a couple of days, a new equilibrium is approached with a slight net positive change in reactivity. The xenon-135 negative reactivity transient is also the driving force of the extended portion of the simulation, when the power is reduced to 59%FP.

The duration of each phase of the shim operation (one bank out, two banks out, three banks out) is determined by the reactivity worth of the adjusters. For one and two banks out, there is also an effect from the power coefficient, void coefficient, and saturating fission products, due to the power reduction immediately prior to the removal of the first two adjuster banks. The rate of reactivity change due to depletion also influences the duration of the shim operation.

For the two bank out portion of the shim operation, four channels are refuelled, replacing 16 depleted bundles with 16 fresh bundles. The resulting reactivity increase delays the withdrawal of bank C and subsequent events of the shim operation.

The outcome of the 59%FP portion of the simulation is sensitive to the reactivity worth of the remaining five adjuster banks as well as the xenon-135 transient, along with some contribution from the power coefficient of reactivity. The power coefficient affects the initial part of the transient – the first seconds to minutes after the power manoeuvre as the TH conditions settle to a new equilibrium. Beyond that, the core reactivity is dictated predominantly by the xenon-135 transient, and adjuster rod banks are withdrawn to compensate for this transient.

The average liquid zone level may be used as an approximate analogue for the reactivity balance of the core. The liquid zones contribute roughly 6 mk of reactivity. Other changes to core conditions that affect reactivity will introduce a compensating change to liquid zone levels from the RRS. However, two considerations must be kept in mind when considering the liquid zone levels as a reactivity measure:

1. The reactivity to zone level relationship depends on which zone(s) are being filled and their level(s). For example, the top of zones 3 and 10 are near the edge of the core and extend into the reflector, thus the marginal reactivity worth of these zones is low when their fill level is high.
2. The model may overestimate or underestimate the reactivity worth of the liquid zones, thus underestimating or overestimating the change in zone level resulting from a given reactivity change.

Thus, the analysis of any discrepancies in the average zone level should consider the entire simulation to determine whether a discrepancy is due to the non-LZC reactivity change or due to the LZCs themselves. For example, if the LZC level change is consistently underestimated throughout the transient, it suggests that the worth of the LZCs themselves is being overestimated.

The individual liquid zone levels may be used to evaluate the spatial distribution of reactivity, due to the distribution of fuel burnups, adjuster rods, and xenon-135. The LZCs are used to spatially control the flux distribution to compensate for perturbations due to refuelling and xenon-135 and will also attempt to compensate for adjuster movements. Any differences in individual zone levels from the data suggest a reactivity discrepancy for part of the core.

Sensitivity and Uncertainty Analysis

For this study, the sensitivities of the shim operation to different uncertainties and parameters must be identified. These may be compared to any differences identified between the model and station data to determine any potential sources of discrepancies. The sensitivities may also be evaluated by comparing the different models to one another. Some of the key parameters being evaluated include:

Burnup distribution – To evaluate the effect of the distribution of fuel burnups, a simulated core follow calculation was performed on the reactor. Using a random snapshot as a starting point, thousands of days of core operation and refuelling are simulated prior to the shim transient. This generates a set of burnup distributions that can be used as initial conditions for simulating the shim operation. The results from different initial core snapshots may be used to identify whether the fuel burnup distribution can potentially affect the evolution of the transient. If different snapshots behave similarly during shim operation, it suggests that the actual station burnup distribution was unlikely to have a significant effect.

Nuclear Data – The neutronic calculation is highly reliant on the nuclear data used in the lattice physics code (SCALE/TRITON [15]) to calculate the homogenized cross-sections for the diffusion code (PARCS [10]). SCALE includes a stochastic uncertainty analysis tool, called

Sampler, which runs a SCALE sequence using one or more perturbed sets of nuclear data. These data sets are pre-perturbed using the covariance matrix that accompanies the nuclear data and are distributed with SCALE.

For this work, lattice calculations were performed using 60 perturbed nuclear datasets from the SCALE 252-group library based on ENDF/B-VII.1. The depletion and branch calculations are then performed using each dataset, for both the infinite lattice cell and the lattice edge cell, along with performing reflector calculations. The homogenized fuel and reflector cross-sections are then combined with the device incrementals from the supercell calculations, which are not perturbed by Sampler.

This creates a set of perturbed PARCS PMAXS cross-section datasets which may be used either to simulate the station transient from its actual burnup state or to perform simulated core follows to generate snapshots to simulate the station transient from different burnup states. A new core follow is performed for each dataset to construct core snapshots consistent with the perturbed fuel properties, as described in [7].

It should be noted that the Shift module of SCALE [16] was not released and hence the nuclear data uncertainty impact on the device incrementals could not be directly calculated. Instead, incremental cross sections were perturbed directly as described below.

Incrementals Modelling and Uncertainty – While it is not possible to perform a proper uncertainty analysis for the reactivity devices in the current methodology, it is possible to perform a sensitivity study. The device incrementals can be perturbed in order to increase or decrease their worth. There are several uncertainties in the modelling of the reactivity devices:

- Uncertainties in the true device geometry due to manufacturing tolerances (assumed to be small).
- Uncertainties in the lattice super-cell calculation, either from the microscopic cross-section data or from modelling approximations.
- Modelling uncertainties introduced by the application of the device incremental cross-sections to the PARCS core model, e.g., from the expansion of the device's effective volume to fit to the PARCS mesh.
- Adjuster rod irradiation.

Incremental Cross Section Placement in the Lattice – The lattice super-cell calculation uses a repeating lattice of 2 lattice pitches across by 1 lattice pitches vertically by 1 bundle length axially (2x1x1 region) and homogenizes a 1x1x1 region centered on the reactivity device. Incremental cross-sections are determined by comparing the macroscopic cross-sections with and without the device. It is assumed that this super-cell geometry is representative of all possible device locations in the core, and the same incremental cross-sections are applied for a given device for all PARCS nodes, even reflector nodes, except that fission incrementals are zeroed for the reflector nodes.

While the homogenization region is a 1x1x1 region centered on the reactivity device, since the device is vertically oriented, running perpendicular to and between the fuel channels, it is not

possible to apply the incremental cross-sections directly to a 1x1 fuel channel region. Instead, the incremental cross-sections are distributed to both adjacent fuel channels, with a weight of 0.5 assigned to each channel. Furthermore, as most reactivity devices are not perfectly centered on a single axial node, the incremental cross-sections must be further distributed between two axial planes, with the relative weights determined by the actual device location. The distribution of device incrementals over a larger volume has a tendency to weaken the spatial self-shielding effect, increasing their predicted reactivity worth.

The following three models are evaluated:

- 12 axial plane model, with 12 fuel bundles per channel, and extrapolation distance applied axially. The burnups from the station, which has 13 fuel bundles per channel, are averaged between adjacent bundles. In this model, the liquid zone compartments are centered on an axial plane, while the centre row of adjusters is distributed between the two centre axial planes.
- 13 axial plane model, with 13 fuel bundles per channel, with the lengths of the two end planes being equal to a ½ bundle plus an extrapolation length[1]. In this model, the centre row of adjusters is centered on an axial plane, while the liquid zone compartments are distributed between two axial planes.
- 26 axial plane model, with 13 fuel bundles per channel, with the length of the two end planes consisting solely of the extrapolation length used in the 13-plane model. Both the liquid zone compartments and the adjusters are distributed between two axial planes, but as the length of each axial plane is half of its value in the other cases, this is equivalent to one axial plane in the other cases.

The reference case uses the 13 axial plane model. It should be noted that, while the modelling of the centre row of adjusters is significantly affected by the axial nodalization, the effect on the other two rows is less as they do not fall neatly along a bundle or half-bundle position axially (they are offset 80 cm from the centre row on either side, or 1.62 bundle lengths). For these adjusters, the 12 axial plane model has these adjusters more heavily weighted towards one axial plane. Thus, the effect of the nodalization of these adjusters is expected to be similar to the liquid zone compartments, except with a smaller magnitude.

Adjuster Irradiation/Aging – As the adjuster rods are normally in-core, they are continuously exposed to the full in-core flux and are thus irradiated over their lifetime. The neutron-absorbing materials in the rods will gradually be transmuted to less-strongly absorbing isotopes, over time decreasing the macroscopic absorption cross-section of the rods. The effect can be significant over the lifetime of the adjuster rods. For the CANDU 900 data, the adjuster rods are significantly aged and therefore aging factors must be applied to achieve accurate results for k_{eff} and flux distribution. The incremental cross-sections for unirradiated adjusters, calculated using Serpent, are perturbed by an aging factor dependent on the adjuster rod age, adjuster rod type, and the cross-section type. These aging factors were derived from prior industry work using

[1] While the actual length of the core is exactly 12 bundle lengths, the flux does not go exactly to zero at the ends of the core, with the extrapolated length being approximately 12 cm longer than the actual length (see page 22 of chapter 2 of [5]). For the 13 plane model, this extrapolation is applied by lengthening the nodes at either end.

DRAGON. Perturbation factors are calculated separately for each reaction type and energy group.

For this work, an adjuster rod age of 182,500 effective full-power hours (EFPH) was selected. Adjuster rod ages of 195,000 EFPH and 154,008 EFPH were also selected as sensitivity cases, along with a "new adjusters" case.

Results

Reference Case

For the reference case, using a nodalization of 13 axial planes (one bundle per plane, with the end bundles shortened to a ½ bundle plus extrapolation distance), using the initial CANDU 900 depletion snapshot and zone levels, unperturbed nuclear data, unperturbed device incremental cross-sections, and aged adjuster rods with an irradiation of 182,500 EFPH, the coupled TRACE/PARCS model, running the modified codes TRACE_Mac1.1 and PARCS_Mac1.1, it reproduces the overall behaviour of the transient at the CANDU 900 station.

Figure 5 shows a comparison of PARCS_Mac1.1 channel powers against SORO channel powers obtained from independent simulations by station staff, given identical bundle depletions and equivalent boron moderator poison. The channel power discrepancies for the vast majority of fuel channels do not exceed 2%, except for edge channels (channels with fewer than four neighbours) and some channels in the vicinity of the liquid zone compartments. In the PARCS model, the edge channels use a different set of cross-sections calculated by homogenizing an edge channel lattice model.

Table 1 summarizes the initial zone levels calculated by the coupled RRS model compared to SORO predictions made by the operators. Most of the predicted zone levels in this simulation are very close to the SORO predictions, except for zones 3, 4, and 11. For zone 3, the discrepancy corresponds to a rather small reactivity difference as the zone is near the top of the core, and the change in fill from 70% to 84% corresponds to near the top of the core, where the flux is lower and thus the zone level change contributes less reactivity. The other two zones, 4 and 11, are in the centre of the core, and the lower zone level indicates either excess absorption near the centre of the core or excess reflection from the periphery of the core, or a combination of the two.

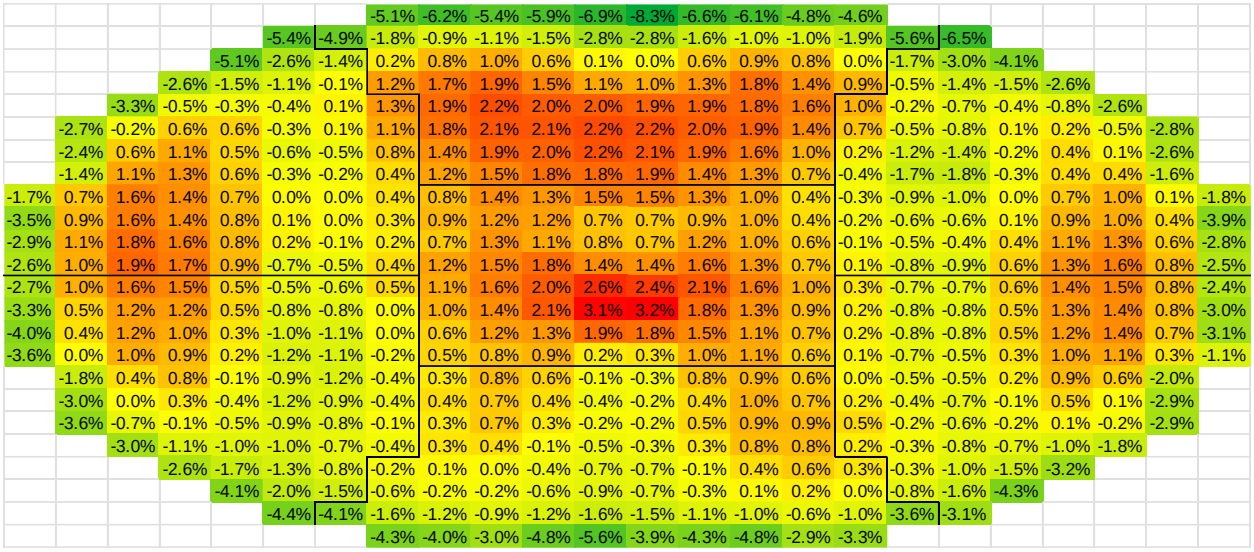

Figure 5: Channel power distribution for TRACE_Mac1.1/PARCS_Mac1.1 model versus SORO

Table 1: Calculated model initial zone levels versus SORO

Zone	SORO Data (%)	PARCS Simulation (%)	Error (%)
1	51.58%	57.91%	6.33%
2	50.22%	48.84%	-1.38%
3	70.10%	83.57%	13.47%
4	39.93%	28.08%	-11.85%
5	66.77%	67.64%	0.87%
6	51.06%	53.26%	2.20%
7	47.64%	47.95%	0.31%
8	52.87%	59.45%	6.58%
9	43.16%	43.91%	0.75%
10	53.51%	56.63%	3.12%
11	39.22%	15.91%	-23.31%
12	52.52%	51.90%	-0.62%
13	40.91%	50.97%	10.06%
14	41.68%	42.01%	0.33%

The effect of the thermalhydraulic channel grouping (40 channels) was evaluated by modelling 480 single channels using the conditions from the coupled model, then feeding the single channel model conditions back into PARCS. The effect was found to be insignificant, with k_{eff} changing by -0.03 mk and the vast majority of channel powers changing by less than 0.2%.

Figure 6 presents the liquid zone level trends of the simulated reference case against the station data, with a rolling window used to reduce the noise for individual zone levels. The two subplots are synchronized based on a known offset between the initial snapshot used to start the simulation and the station data. The trends in are overall similar between the station data and simulation, but some differences were noted:

- The average liquid zone levels for the central zones (zones 4 and 11) are up to 15% lower in the simulation compared to the station data, for similar points in the transient. This indicates that there is slightly less reactivity in the centre of the core than the periphery in the simulation, compared to the station data, that the liquid zone controllers are compensating for.
- The maximum liquid zone levels after each adjuster pull, as well as after the refuelling run, are slightly less in the simulation than the station data (by less than 5%).
- The withdrawal of adjuster bank A occurs approximately one day earlier in the simulation. The withdrawal of adjuster bank B occurs approximately 3 days earlier in the simulation, giving a discrepancy of two days for the interval between adjuster banks A and B. The remainder of the simulation shows little time discrepancy from the station data, besides the 3-day shift arising from the previous discrepancies.
- The zone level tilt between the two zones in each zone pair is considerably larger in the simulated core than in the real core, particularly for the 3/10 and 4/11 zone pairs.

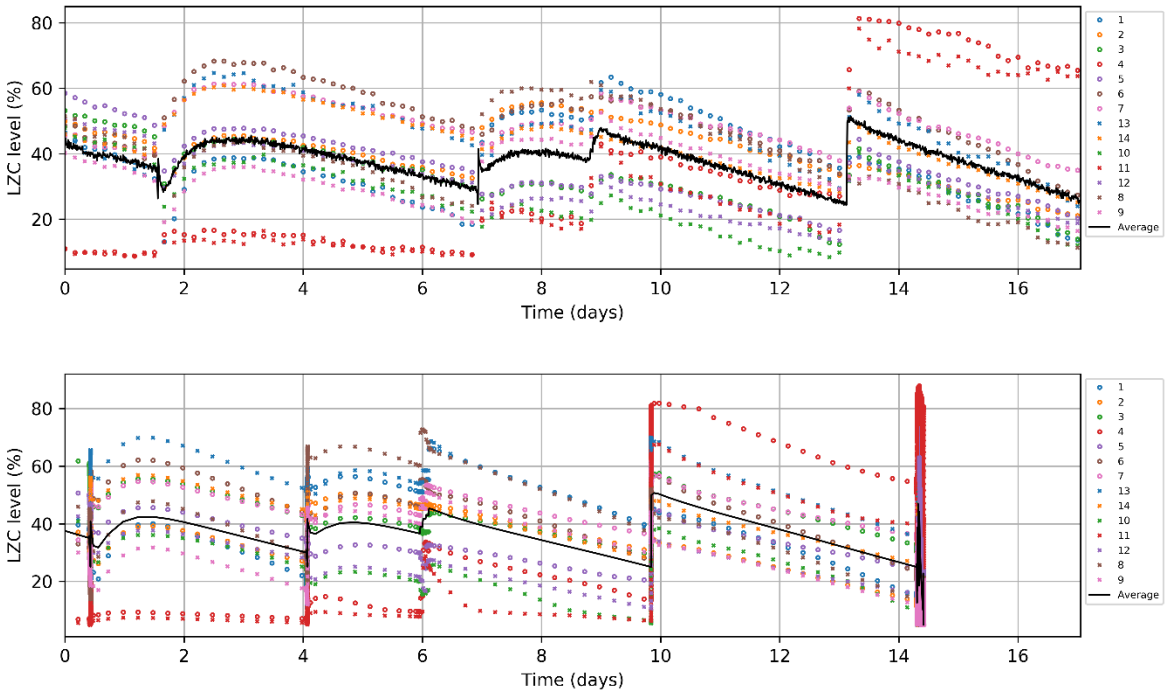

Figure 6: Evolution of simulated shim operation – station (top) versus reference case (bottom)

Figure 7 shows the continuation of the simulation, where the power is reduced to 59%FP, showing the simulated reactor and zone powers along with liquid zone compartment levels. After roughly 2.5 hours, the average zone level reaches 10% with all adjusters withdrawn, and the reactor is shut down. The zone levels behave as expected, with the zone levels in the central zones (4/11) increasing in particular to counteract the flux peaking from the removed adjusters. The zone powers also follow expected trends, where fully drained zones decrease in power while the remaining zones increase in power. As well, towards the end of the simulation, the total power decreases by approximately 2%FP. This is due to the RRS response from the liquid zone control algorithm, from the following two effects:

1. With twelve liquid zones fully drained, the bulk controller's effective gain is reduced, as a given negative valve lift will drain only two zones instead of all 14. Thus, a larger power error is required for a given reactivity insertion rate (dictated by the xenon transient being offset to maintain criticality).
2. The spatial controller is applying positive valve lift to the 12 drained zones (to attempt to bring their fill levels up to the average fill level) as well as applying positive valve lift to the two central zones (to attempt to bring the zone power down to average zone power). The bulk controller must counteract this positive lift by applying negative lift, which requires a negative power error.

This effect would not have been captured in a quasi-static calculation that assumes reactor power is held at the setpoint while controlling for k_{eff} and spatial flux, thus demonstrating the advantage of using a dynamic (transient) approach at key points in a long-term simulation.

The maximum power tilt immediately prior to shutdown, ignoring zones 4 and 11, is approximately 13%. The flux tilt increases over the course of the transient as the zone levels become fully drained or filled and cannot spatially control the flux, combined with the positive feedback effect from xenon.

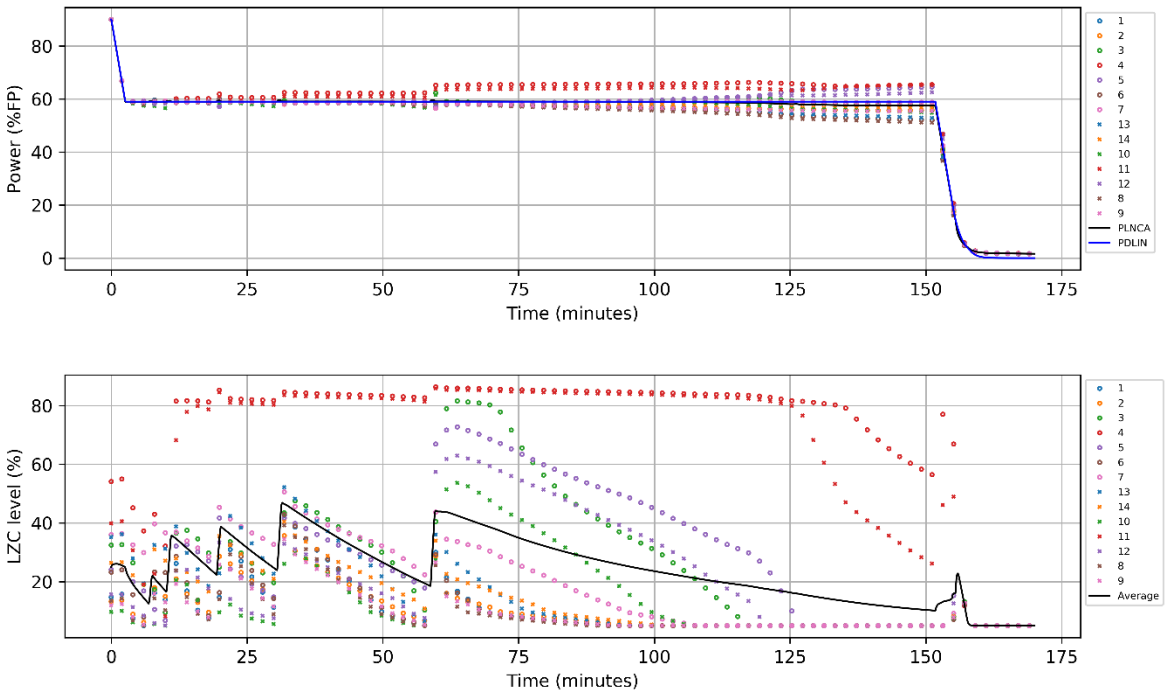

Figure 7: Evolution of final phase of simulated shim operation – reactor/zone powers (top) and liquid zone levels (bottom)

Effect of Adjuster Rod Irradiation

Table 2 gives the effect of applying the adjuster depletion on the reactivity worth of the adjusters. The reactivity worth of each bank is calculated from the initial fuel depletion snapshot by calculating the reactivity change between the bank-in and bank-out states with all liquid zones

filled to 50% and with all prior banks withdrawn. Overall, the reactivity worth of the adjusters is reduced by 10.92% when aged to 182,500 EFPH, with a greater effect on adjuster types 1, 2, and 4.

Table 2: Effect of adjuster depletion on reactivity device worths (in mk)

Device	0 EFPH	154,008 EFPH		182,500 EFPH		195,000 EFPH	
Liquid Zones	5.751	5.765	(+0.26%)	5.763	(+0.22%)	5.762	(+0.20%)
Adjusters (All)	12.240	11.116	(-9.19%)	10.904	(-10.92%)	10.814	(-11.66%)
Adjuster Bank A (Type 3)	1.116	1.058	(-5.20%)	1.043	(-6.48%)	1.038	(-6.94%)
Adjuster Bank B (Type 3)	1.195	1.116	(-6.59%)	1.100	(-7.95%)	1.093	(-8.54%)
Adjuster Bank C (Type 2)	1.708	1.529	(-10.47%)	1.495	(-12.47%)	1.481	(-13.29%)
Adjuster Bank D (Type 4)	0.927	0.838	(-9.51%)	0.824	(-11.02%)	0.815	(-11.99%)
Adjuster Bank E (Type 1)	1.505	1.351	(-10.25%)	1.322	(-12.17%)	1.311	(-12.90%)
Adjuster Bank F (Type 3)	1.404	1.330	(-5.27%)	1.318	(-6.13%)	1.309	(-6.76%)
Adjuster Bank G (Type 2)	2.052	1.847	(-9.98%)	1.808	(-11.90%)	1.794	(-12.57%)
Adjuster Bank H (Type 1)	2.335	2.047	(-12.33%)	1.994	(-14.58%)	1.973	(-15.50%)

Table 4 at the end of the section summarizes the differences in the timeline of the transient between the station data and the different simulations. Overall, the length of the operation with zero and two banks out is slightly underpredicted, the length of the operation with three banks out is overpredicted, and the length of the operation with one bank out is greatly underpredicted. The greater the adjuster age, the shorter each interval becomes, with the three-bank-out interval being most affected. With unirradiated adjuster rods, the entire operation is 1.22 days longer than for the reference case.

The adjuster depletion also influences the radial flux distribution in the core. Table 5 shows the magnitude of this effect, with the difference in zone levels in zones 4 and 11 between the simulations with irradiated versus unirradiated adjusters at 28.0% towards the end of the transient, with the unirradiated adjusters showing a greater underprediction against the station data when compared to the reference case.

Finally, Figure 8 shows the results of changing the effective adjuster rod age to the end of the simulated transient. Results are shown for fresh adjusters, as well as ages of 154,008 EFPH, 182,500 EFPH, and 195,000 EFPH. The irradiation of the adjuster rods has a significant effect on the outcome of the transient. With fresh adjusters, the reactor survives xenon poison-out with a minimum zone level of 35%, achieved with all adjusters out of the core. With an adjuster rod irradiation of 154,008 EFPH, the reactor initially survives xenon poison-out with a minimum zone level of 12%. However, the spatial xenon transient continues until reaching a 20% power

tilt, which triggers a setback in the simulation. This reduces the power setpoint to 55.8%FP, causing the xenon transient to poison out the reactor. With adjuster rod irradiations of 182,500 EFPH (reference case) and 195,000 EFPH, the reactor poisons out without a setback, with the poison-out occurring 10 minutes sooner for the 195,000 EFPH case.

The 154,008 EFPH case demonstrates that the reactor becomes unrecoverable once the average zone level is too low, as the lack of spatial control leads to a flux tilt setback. Thus, the cases that are shut down without a setback upon reaching 10% average zone level would be expected to inevitably shut down, either from fully draining the liquid zones and becoming subcritical, or from a flux tilt setback leading to the reactor becoming subcritical.

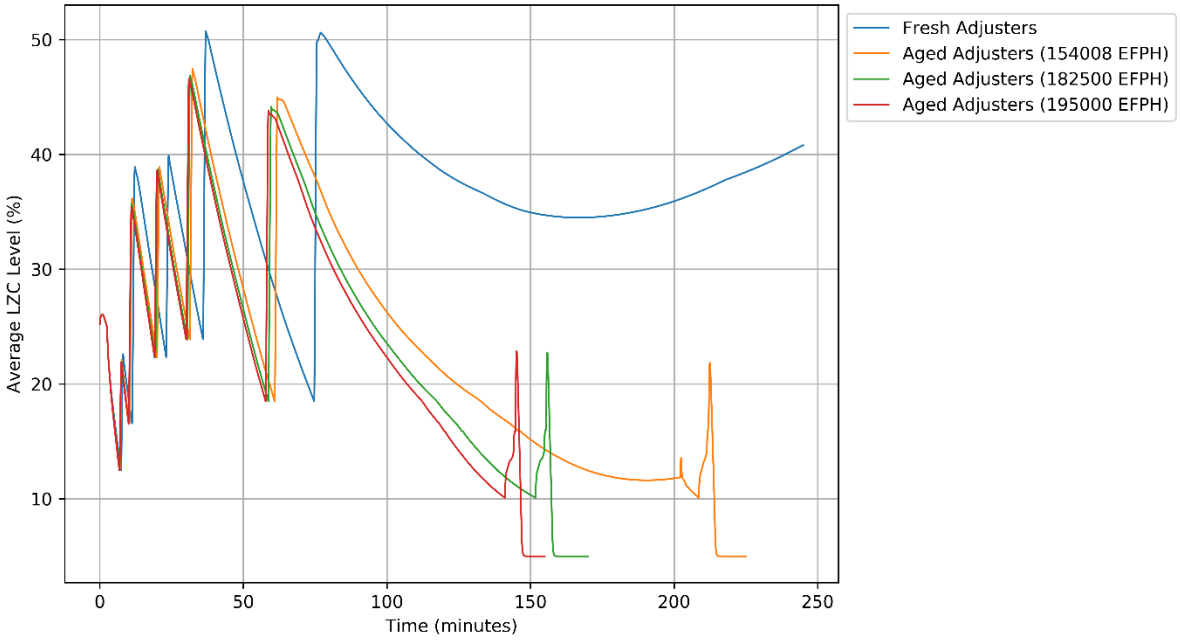

Figure 8: Effect of adjuster depletion – 59%FP transient

Effect of Liquid Zone Reactivity Worth

For this sensitivity case, a perturbation was applied to the liquid zone compartment incremental cross-sections, decreasing their values by 10%. This resulted in decreasing their reactivity worth by approximately 8.6%, from 5.76 mk to 5.26 mk. This change only has a small impact on the poison-out trajectory compared to the reference case, with the adjuster movement timings essentially unchanged, and the poison-out occurring roughly 10 minutes earlier. The rest of the shim transient is slightly shorter. This effect is in line with expectations, as the reactivity gained from draining the liquid zones from 50% (initial state) to 25% (threshold for adjuster movements) and then to 10% (threshold for poison-out) is slightly decreased. These results are quantified in Table 4 and Table 5.

Effect of Axial Nodalization

For this sensitivity case, the axial nodalization is altered. The effect on the final phase of the transient is shown in Figure 9. When a 26-plane nodalization is used instead of the 13-plane nodalization, the simulation predicts that the minimum zone level reaches 10%, then poisons out when the flux tilt triggers a setback. When a 12-plane nodalization is used, the simulation predicts that the reactor survives xenon poison-out with a minimum zone level of approximately 18%. There is also a small effect on the duration of the shim operation; the effect is shown in Table 4.

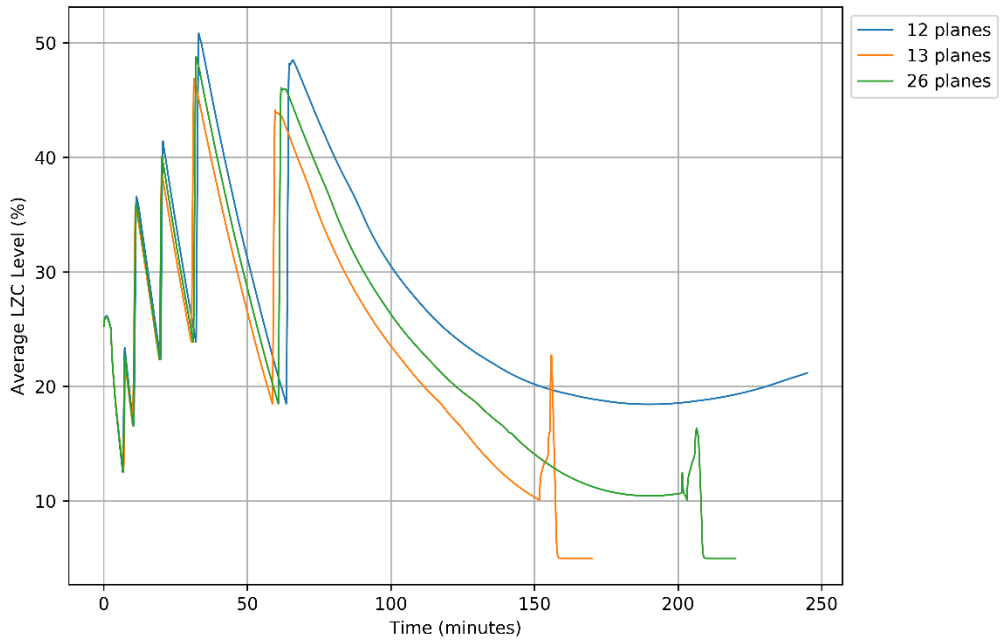

Figure 9: Effect of core nodalization – 59%FP transient

Table 3 shows how the nodalization affects the predicted reactivity worth of each set of reactivity devices. Note that the 13-plane model predicts a higher liquid zone controller worth than the 12-plane model. Conversely, the 12-plane model predicts higher adjuster worths than the 13-plane model, with the effect coming from banks D, F, G, and H, which are the centre row banks. This is consistent with expectations, where a model that distributes the device incrementals over a larger volume will predict a larger reactivity worth due to the weaker spatial self-shielding. This also explains the results of the previous figures. For Figure 9, the 12-plane model survives poison-out as the adjuster reactivity worths are significantly greater. The results of the 26-plane model fall intermediately between the 12-plane and 13-plane models.

It should also be noted that the effect of the nodalization on the liquid zone compartments is of a similar magnitude to the liquid zone sensitivity study. Therefore, the effect of the nodalization of the liquid zone compartments, independent of the adjusters, will be similar to the results presented for the effect of the liquid zone reactivity worth.

Table 3: Effect of core nodalization on reactivity device worths (in mk)

Device	13 planes	12 planes		26 planes	
Liquid Zones	5.763	5.347	(-7.22%)	5.571	(-3.33%)
Adjusters (All)	10.904	11.230	(+2.99%)	11.056	(+1.39%)
Adjuster Bank A (Type 3)	1.043	1.029	(-1.35%)	1.032	(-1.12%)
Adjuster Bank B (Type 3)	1.100	1.095	(-0.38%)	1.096	(-0.34%)
Adjuster Bank C (Type 2)	1.495	1.454	(-2.72%)	1.459	(-2.35%)
Adjuster Bank D (Type 4)	0.824	0.884	(+7.27%)	0.858	(+4.09%)
Adjuster Bank E (Type 1)	1.322	1.307	(-1.14%)	1.309	(-0.94%)
Adjuster Bank F (Type 3)	1.318	1.416	(+7.40%)	1.376	(+4.41%)
Adjuster Bank G (Type 2)	1.808	1.882	(+4.11%)	1.876	(+3.79%)
Adjuster Bank H (Type 1)	1.994	2.162	(+8.41%)	2.049	(+2.73%)

Effect of Nuclear Data Uncertainty

The comparison of the shim transient timings is included in Table 4. The station data fell within roughly two standard deviations of the average simulation results, except for the one-bank-out interval. However, the durations of the intervals are highly correlated (>96%), while the differences to station data for different intervals have opposite signs, thus nuclear data uncertainty alone cannot explain all of the differences between simulations and station data for shim durations.

Figure 10 plots the effect of nuclear data perturbation on the extended portion of the transient, with each sample plotted as a coloured line, with time zero set as the time at which the reactor power setpoint is set to 59.0%FP. Each of the 60 samples follows a similar trend, though the timing varies. The results of the samples are as follows:

- 44 samples (73.3%) reach the 10% average zone level threshold, triggering the simulation to shut down the reactor. These can be considered to poison out from bulk xenon only, even without the effect of spatial xenon.
- 15 samples (25.0%) do not reach the 10% average zone level threshold at first, but a reactor setback on flux tilt is subsequently triggered. These can be considered to not poison out directly from bulk xenon, but poison out due to spatial xenon increasing the flux tilt to an unacceptable level.
- 1 sample (3.3%) does not reach the 10% average zone level threshold and does not experience a setback.

Due to the negative valve lift from the bulk controller along with spatial control being phased out below 10% zone level, it is possible for the zone power in drained zones to exceed the zone power in spatially controlled zones while the zones remain drained. This was observed for zones 5 and 12. This continues until either a flux tilt setback occurs, or a crossover point is reached where the liquid zone compartment starts to refill, providing positive feedback as spatial control is phased in, at which point the liquid zone levels rapidly change to a new equilibrium.

The sensitivity to liquid zone controller gains was evaluated, with a focus on the marginal cases (i.e., those closest to the threshold of whether a setback occurs) and it was found that realistic perturbations to the controller gains did not significantly alter the outcome of the cases evaluated in this work.

This indicates that, when only nuclear data uncertainty is considered, the simulation predicts a poison-out with an approximately 73% confidence, or 0.62σ, if the setback is not considered. However, if setbacks within the simulation duration are also considered, then the confidence rises to 98%, or 2.1σ. In addition, the nuclear data uncertainty does not include uncertainty in the incremental cross-sections of the reactivity devices, as perturbed incremental cross-sections were not available in the current methodology. However, the sensitivity of the outcome to the adjuster and liquid zone incremental cross-sections was determined in the prior subsections.

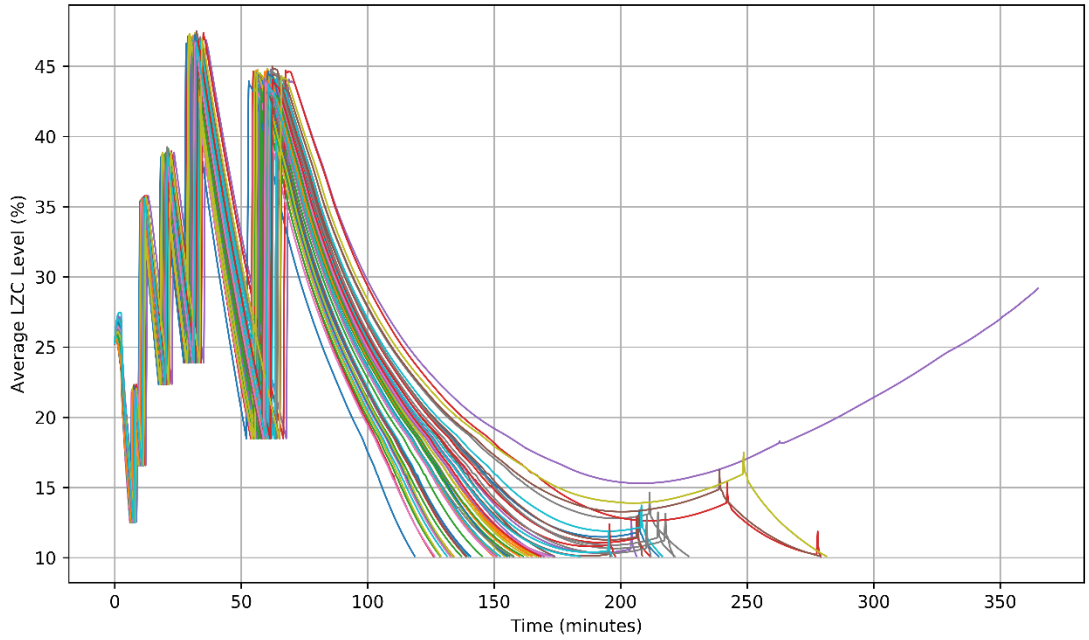

Figure 10: Effect of nuclear data perturbation – 59%FP transient

For comparison, 24 cases were run with unirradiated adjusters modelled, and showed that none of these cases poison out with unirradiated adjusters, and that the nuclear data uncertainty has far less of an impact than the adjuster rod irradiation effect.

Effect of Burnup Distribution

Figure 11 plots the effect of simulating the transient with different initial fuel burnup distributions. Unlike the simulations using the original CANDU 900 snapshot, these cases do not include any refuelling runs. Therefore, a case was also run using the original CANDU 900 snapshot, but without any refuelling runs, to be the reference against which other burnup distributions are compared. All 14 snapshots lead to a poison-out, with the reference case falling roughly in the middle of the 14 snapshots. Some reactor setbacks were observed earlier in the

transient than in the above sections, as the generated burnup distributions were less optimal than the CANDU 900 snapshot.

When the transient was simulated with both nuclear data uncertainty and different initial fuel distributions, with each nuclear data sample having its own corresponding core snapshot, the distribution of shim operation trajectories was found to be similar to those observed for the CANDU 900 samples in Figure 10, except that earlier setbacks were observed as in Figure 11. All 20 of the perturbed cases led to a poison-out. These results are quantified in Table 4 and Table 5.

For one of the nuclear data samples with its corresponding snapshot, the simulated reactor tripped on regional overpower while withdrawing adjuster bank A. This was due to the snapshot having a single channel at an abnormally high power (>7.1 MW) which increased CPPF to an abnormal level (>1.15), reducing trip margin (note: the reactor would never be operated in such a state in reality). For this sample, the snapshot was replaced with a different snapshot from the same simulated core follow.

A minority of the perturbed cases also experienced setbacks during the adjuster movements. In this case, the simulation was set up to restore the reactor power as soon as the setback condition cleared.

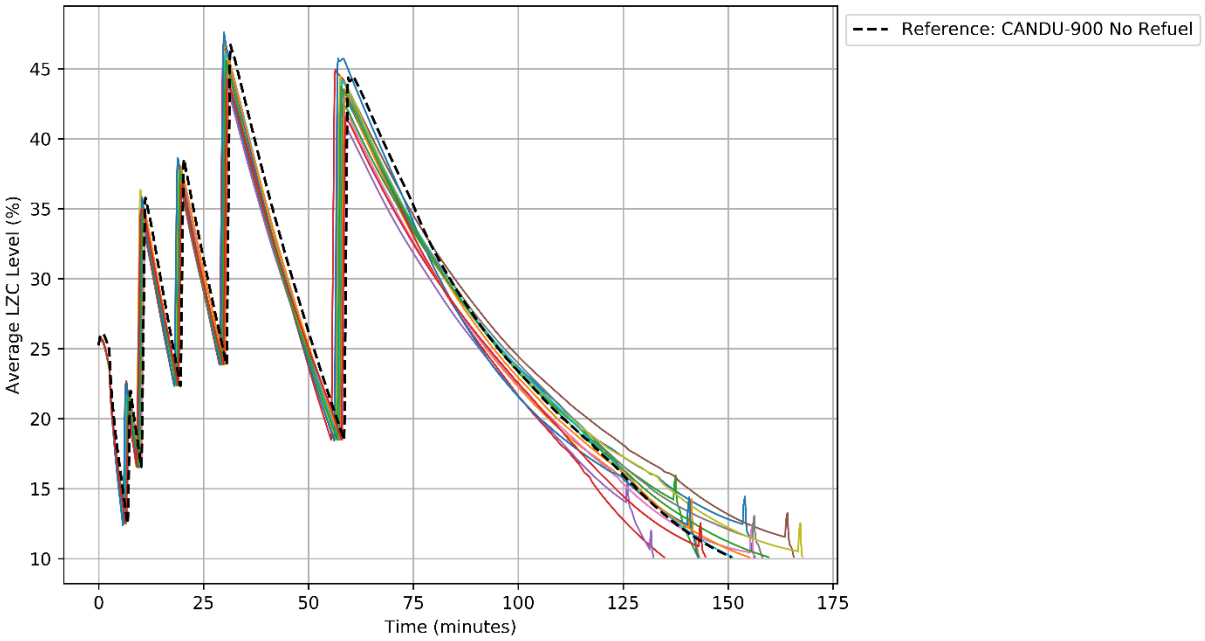

Figure 11: Effect of random burnup snapshots – 59%FP transient

Summary

Table 4 and Table 5 summarize and quantify the figures of merit for the reference case and all of the sensitivity cases. These will not be further discussed in this section, but instead support the discussion in the following section as well as the results presented earlier in this section.

Table 4: Comparison of station data and simulations on shim transient timings (in days)

Case	Zero banks out	One bank	Two banks	Three banks	Total
Station	**12.47**	**5.38**	**6.19**	**3.92**	**27.95**
0 EFPH	11.51	3.91	6.05	4.95	26.42
154008 EFPH	11.36	3.69	5.80	4.55	25.40
182500 EFPH (ref.)	**11.33**	**3.64**	**5.77**	**4.46**	**25.20**
195000 EFPH	11.32	3.62	5.76	4.42	25.12
90% LZC weight	11.14	3.56	5.73	4.48	24.91
12 plane nodalization	11.03	3.48	5.60	4.27	24.39
26 plane nodalization	11.25	3.54	5.68	4.32	24.79
Nuclear data unc.	11.4 ± 0.6	3.7 ± 0.2	5.8 ± 0.3	4.5 ± 0.3	25.3 ± 1.3
No refuel reference	**9.76**	**3.63**	**4.30**	**4.55**	**22.24**
Burnup distribution unc.	n/a	3.74 ± 0.09	4.47 ± 0.06	4.72 ± 0.14	n/a
Burnup + data unc.	n/a	3.98 ± 0.31	4.55 ± 0.28	4.79 ± 0.18	n/a

Table 5: Comparison of station data and simulations on zone level figures of merit

Case	Average zone level (4+11 only) (%) 5 minutes before end of station data	Average zone level (%) 105 minutes after 59%FP
Station	**65.6**	**n/a**
0 EFPH	19.0	41.4
154008 EFPH	42.3	24.7
182500 EFPH (ref.)	**47.0**	**21.9**
195000 EFPH	49.2	20.6
90% LZC weight	52.1	21.4
12 plane nodalization	42.1	28.8
26 plane nodalization	44.9	24.6
Nuclear data unc.	47.0 ± 15.2	22.3 ± 2.6
No refuel reference	**46.9**	**21.7**
Burnup distribution unc.	n/a	21.4 ± 1.0
Burnup + data unc.	n/a	21.8 ± 3.4

Discussion

This study demonstrates that, by coupling TRACE_Mac1.1 and PARCS_Mac1.1 along with an RRS model, shim operation can be simulated, and sensitive parameters can be evaluated. These sensitivities are most interesting to examine for the extended portion of the simulation, where the reactor power is reduced to 59%FP. The importance of these sensitivities on this phase of the

simulation, as identified from this study, are presented through listing the sensitivities from most to least significant below, quantified using the average zone level at 105 minutes:

1. Adjuster rod irradiation (-19.5%, reference versus unirradiated).
2. Axial nodalization (+6.9%, 12 planes versus 13 planes, due to effect on adjusters).
3. Nuclear data uncertainty (σ = 2.6%).
4. Burnup distribution (σ = 1.0%).
5. Liquid zone compartment reactivity worth (-0.6% for 10% incremental change).

The adjuster rod irradiation was the greatest factor in defining the end of the transient, especially when comparing irradiated and unirradiated adjusters. When the transient is simulated with unirradiated adjusters, they had sufficient reactivity depth to maintain criticality and prevent the core from being poisoned out by xenon, with a reasonable margin. When the transient is simulated with irradiated adjusters, the core is poisoned out. The reactivity worth of the five banks of adjusters was reduced by 0.96 mk, or 11.6%, for the irradiated case compared to the unirradiated case. Varying the irradiation around the selected reference value had a proportionally smaller effect on the transient.

The axial nodalization had a significant effect on the end of the transient, as the centre row of adjusters, which dominate the adjuster contribution to the end of the transient, were significantly affected by the doubling of their effective volume. The calculated worth of the five banks of adjusters was 0.38 mk, or 5.3%, higher for the 12-plane model compared to the 13-plane model – nearly half of the adjuster irradiation effect! Since the adjuster worth, particularly for the centre row, dominates the outcome of the end of the transient, the 13-plane model should be more accurate in this regard, as the adjuster effective volume is closer to the original homogenization volume. As well, it should be considered that the model used in this study also doubles the effective volume of the adjusters in the horizontal (x) direction, making this an area for future investigation.

The nuclear data uncertainty was also found to have a significant effect. When the average zone level was taken 105 minutes after the reactor power setpoint is set to 59%FP, Table 5 showed that the effect of 28,500 EFPH of adjuster depletion was 2.8% zone level, while one standard deviation of the nuclear data uncertainty was 2.6% zone level, thus one standard deviation of nuclear data uncertainty is equivalent to approximately:

$$28500 \text{ EFPH} + \frac{2.6\%}{2.8\%} = 26500 \text{ EFPH} \tag{0.1}$$

Thus, one standard deviation of nuclear data uncertainty, not including device incremental uncertainty, is roughly 15% of the total adjuster depletion effect. The influence of the fuel distribution in the core is equal to approximately 40% of the influence of the nuclear data uncertainty, based on a standard deviation of 1.0% zone level, compared to 2.6% zone level for the nuclear data uncertainty.

The other significant figure of merit was the timing of events from the initial snapshot up to the end of three-bank-out operation at 25% average zone level. The reference case under-predicted

the total length of the shim operation by 2.75 days (10%), primarily due to an under-prediction of the one-bank-out duration by 1.73 days (32%), as well as an under-prediction of the zero-banks-out duration by 1.14 days (9%). The importance of the sensitivities, from most important to least important, is as follows:

1. Nuclear data uncertainty (σ = 1.31 days).
2. Adjuster rod irradiation (-1.21 days, reference versus unirradiated).
3. Axial nodalization (-0.82 days, 12 planes versus 13 planes).
4. Burnup distribution (σ = approximately half of nuclear data uncertainty).
5. Liquid zone compartment reactivity worth (-0.29 days for 10% incremental change).

The discrepancy thus cannot be entirely explained from the uncertainties studied in this work, especially for the one-bank-out duration. The two-bank-out and three-bank-out durations are within two standard deviations (from nuclear data uncertainty) of the station data, but in opposite directions. The discrepancies must be explained by phenomena that were not evaluated in this work, with potential phenomena as follows:

- Nodalization of the core physics model, outside of the evaluated axial nodalization meshes.
- Differences in depletions of different adjusters of the same type (up to a 5% variation in flux, and thus depletion, was identified to exist, but not applied to the model). This effect is expected to be small (equivalent to a 5% variation in adjuster depletion, but for individual banks).
- Errors affecting the rate that fuel depletion changes the core reactivity, either from approximations in the lattice physics and core physics model, or parameters in the lattice physics model (e.g., fuel density).
- Errors affecting the power coefficient of reactivity, void coefficient of reactivity, or xenon reactivity, such as the modelling of fuel temperature or coolant voiding. A negative change to the power coefficient would add positive reactivity when the reactor power is decreased, lengthening the shim operation (results move closer to the station data). A negative change to the void coefficient would have a similar effect as the power coefficient, but only at high power.
- Changes to thermalhydraulic setpoints over the course of the transient that are neither captured in the model nor provided with the station data. No known changes to setpoints were identified for this transient.
- The presence of moderator poisons and changes in their concentration during the shim operation. While the equivalent moderator poison was estimated periodically over the course of the real-world transient using SORO, no actual measurements were taken. The simulation in this work used the equivalent moderator poison values, which went to zero prior to the first adjuster bank being withdrawn. If these estimates were inaccurate, particularly for the initial state as well as for the time at which adjuster bank A is withdrawn, then this would create a discrepancy in the shim duration. More specifically, if moderator poison was present when bank A was withdrawn, and subsequently removed, it would extend the duration of the one-bank-out shim operation.

At the time of this writing, the most likely explanations for the discrepancies in shim duration are a combination of nuclear data uncertainty and errors in the estimated moderator poison concentrations.

The discrepancy in radial power (as indicated by central zone compartment levels) was also measured. The importance of the sensitivities, from most important to least important, is as follows:

1. Adjuster rod irradiation (+28%).
2. Nuclear data uncertainty (σ = 15%).
3. Axial nodalization (-5.0%, 12 planes versus 13 planes).
4. Liquid zone compartment reactivity worth (+5.1% for 10% incremental).

The LZC reactivity worth is ranked lower as the effect is predominantly due to an increase in the amount of water required for the same amount of spatial control. Thus, zone levels are not a good indication of radial power balance when the LZC worth changes significantly. Since the LZC worth also changes for the axial nodalization sensitivity case, the LZC worth effect partially offsets the adjuster worth effect. Note that the difference between the reference simulation and station data was -18.6%. Therefore, while other errors may be present, the discrepancy to the station data may be explained by the nuclear data uncertainty.

Conclusions

This work was able to demonstrate that a shim operation could be simulated using a coupled model of TRACE_Mac1.1, PARCS_Mac1.1, and an ECI-coupled RRS model, and model depletion and adjuster movements comparably to real-world data. The coupled model is able to simulate the effect of depletion, xenon-135, thermalhydraulics, and reactor regulating system response on a CANDU reactor.

When the transient was extended to evaluate a possible continuation, with the power reduced to 59%FP, the reference case predicted that xenon poison-out would occur 2.5 hours after the power reduction. The model showed that adjuster depletion had a major impact on the outcome of the simulation, with no poison-out occurring with unaged adjusters. When including cases that poison-out due to flux tilt setback, 59/60 nuclear data uncertainty cases led to a poison-out. The effect that the burnup distribution had on the simulated transient is less significant than nuclear data uncertainty, with only roughly 40% as great of an effect on the final stage of the transient.

Overall, out of all parameters evaluated, the modelling of the adjusters had the greatest impact on the results and figures of merit for the transient. Not only did the adjuster depletion have a significant impact on the results, but the nodalization of the adjusters also played a significant role. This work demonstrated that nodalizing the core such that an appropriate equivalent volume is achieved for the adjusters is important, and that a larger equivalent volume increases the estimated adjuster worth. While the modelling of the LZCs is similarly affected, the LZCs had far less impact on the results of the transient.

The nuclear data uncertainty was shown to be able to significantly perturb the results, both in terms of the duration of the shim operation, the radial flux distribution, and the outcome at the end of the extended simulation. Therefore, the effect of nuclear data uncertainties cannot be ruled out as a contributing factor to the observations made in this work or for other similar studies.

Future Work

While this work was able to generally model the real-world shim operation and identify several important sensitivities in the modelling of a shim transient, some unaccounted-for discrepancies were still identified, and some sensitivities were identified as candidates for future analysis. The results of this work present several areas for future investigation.

This work identified nodalization of the core physics model as a significant contributing factor, due to the effect on the modelling of the adjuster rods. However, this work only tested a few nodalizations in the axial direction, and these nodalizations do not completely solve the issue of adjuster incremental cross-sections being diluted over a larger volume. In particular, in the horizontal (x-axis) direction, the incremental cross-sections are distributed between two adjacent channels.

Horizontal subdivision of the model was not pursued in this study due to the extra effort required in PARCS when compared to axial renodalization. Currently, there is no method in PARCS to specify independent meshes for fuel assemblies/channels and for the macroscopic cross-sections, the latter of which are needed for reactivity device incremental cross-sections. This contrasts with some CANDU-specific core physics codes, such as RFSP. Therefore, PARCS would treat these subdivided channels as independent fuel assemblies/channels, requiring extra effort to map powers and fluxes to TRACE and particularly to map powers and fluxes to the ECI through signal variables as is done in the current version of PARCS_Mac1.1 and TRACE_Mac1.1. The execution time of the simulation would also be greatly increased.

However, due to the identified importance of nodalization, it is recommended that future studies further quantify the effect of the nodalization on the adjuster rod reactivity worths. The current model was able to quantify the effect of changing adjuster worths on the simulation of a shim transient, thus a study that estimates the change in adjuster reactivity worth due to other modelling parameters (such as nodalization in the x-axis) may be used to estimate the effect these parameters would have on the shim transient. Future work may also investigate other methods for correcting for nodalization effects, such as extending the rod cusping correction methodology already present in PARCS, or applying similar methodologies to calculate correction factors. Monte Carlo models of a full core or mini-core may be created to calculate reactivity device worths that are unaffected by nodalization, to compare against PARCS models.

Secondly, there were some large discrepancies in the duration of different phases of the shim operation, particularly for the one-bank-out phase. One significant “unknown” that was identified was the state of moderator poison in the core. While an “equivalent moderator poison” was calculated at the time of the operation using SORO, no true moderator poison measurements were made. Any error in the moderator poison estimation, either at the initial state or upon withdrawing an adjuster bank, would impact the timing of each phase of the shim operation, as

the coupled simulation assumed these estimates to be accurate. Another possible area for future investigation is the thermalhydraulic feedbacks.

Several recommendations can thus be made for studies aiming to reproduce these results or further expand on this work. The first recommendation is to obtain data from other shim operations besides the one used in this work, ideally using a case where moderator poison can be definitively ruled out as a contributing factor. These may include shim operations from either 900 MW class or 600 MW class CANDU reactors. The second recommendation would be to obtain thermalhydraulic data, as the dataset provided for this work did not include any thermalhydraulic data against which to compare the simulated thermalhydraulic model. Another related recommendation is to obtain real quantities for RRS parameters, such as liquid zone control gains, rather than estimating these parameters, to more accurately model RRS feedbacks, eliminating a source of uncertainty.

As an example, consider the hypothesis that the large under-estimation of the duration of the one-bank-out phase of the operation was due to a discrepancy between the estimated moderator poison concentration (through equivalent boron used in SORO) and the actual moderator poison concentration. If similar discrepancies are not found when simulating different shim operations, it suggests that the hypothesis may still hold, and alternative hypotheses, such as a systematic modelling error, may be ruled out, or at least considered less likely. If the discrepancy does recur, then it suggests that either the hypothesis may be ruled out (and alternative hypotheses that may be common to both events considered), that a common discrepancy in estimating the moderator poison occurred in the events (the likelihood of which can be estimated by comparing the timelines of both events, particularly the use of moderator poison and the estimates of equivalent moderator poison), or that the discrepancy was co-incidental, with two different causes for the two events.

Another area to be investigated is the use of nuclear data and estimation of the uncertainty of nuclear data. Since nuclear data uncertainty was found to have a significant impact, the use of newer nuclear datasets, such as ENDF/B-VIII, may impact both the best estimate as well as uncertainty results. Additionally, the methodology in this study was limited as it was unable to apply the stochastic nuclear data perturbations (from Sampler) to the reactivity device incremental cross-section calculations (using Serpent, which is outside of the SCALE framework supported by Sampler). An improved uncertainty analysis would integrate the reactivity device calculations into the nuclear data uncertainty analysis, such as by migrating the supercell calculation to CSAS-Shift in the upcoming SCALE 6.3 release [16]. Finally, the methodology described in reference [7] could be applied to the uncertainty analysis, performing a core follow from a much earlier state to the point of the shim event using the reactor's operational history, to achieve core depletion profiles consistent with their respective nuclear data perturbations.

The methodology used in this work may also be applied to the analysis of design changes, such as the adoption of advanced fuel cycles, which had been previously studied for DUPIC fuel [2][3] as well as thorium-based fuel [4], with PARCS_Mac serving as a viable alternative to RFSP and DONJON while allowing coupling to TRACE_Mac for thermalhydraulic feedback as well as taking advantage of the uncertainty propagation capabilities present in SCALE using Sampler.

Data Availability

The TRACE and PARCS model data used to support the findings of this study have not been made available due to the confidentiality of the models. The detailed data used to support this work is also unavailable as it was provided confidentially, with approval to present the data included in this article.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Funding Statement

This work was supported by the University Network of Excellence in Nuclear Engineering (UNENE) and the Natural Sciences and Engineering Research Council of Canada (NSERC).

Acknowledgments

The authors would like to thank Michael Tucker for his prior work in developing the TRACE and PARCS CANDU models which were used in this research. In addition, the authors would like to thank him for his support in running the lattice physics calculations using SCALE/TRITON, Sampler, and Serpent 2, to generate the macroscopic cross-sections for PARCS, as well as in running simulated core follows to generate burnup snapshots, based on his previously developed methodology [7]. Both the macroscopic cross-sections and burnup snapshots were used in this work in addition to his own work.

References

[1] J. Koclas, *Reactor Control and Simulation*. Montréal: École Polytechnique de Montréal, 1996.

[2] C. J. Jeong and H. Choi, "Compatibility analysis on existing reactivity devices in CANDU 6 reactors for DUPIC fuel cycle," *Nucl. Sci. Eng.*, vol. 134, no. 3, pp. 265–280, 2000, doi: 10.13182/NSE00-A2115.

[3] C. J. Jeong and H. Choi, "Xenon transient analysis for direct use of spent pressurized water reactor fuel in CANDU reactors (DUPIC)," *Ann. Nucl. Energy*, vol. 27, no. 3, pp. 269–278, 2000, doi: 10.1016/S0306-4549(00)82026-5.

[4] E. St-Aubin and G. Marleau, "CANDU-6 reactivity devices optimization for advanced fuel cycles – Part I: Adjuster rods optimization," *Nucl. Eng. Des.*, vol. 337, pp. 471–481, 2018, doi: 10.1016/j.nucengdes.2015.10.027.

[5] UNENE, *The Essential CANDU - A Textbook on the CANDU Nuclear Power Plant Technology*. Hamilton: University Network of Excellence in Nuclear Engineering (UNENE), 2019. [Online]. Available: https://www.unene.ca/education/candu-textbook

[6] E. Varin, R. Roy, R. Baril, and G. Hotte, "CANDU-6 operation post-simulations using the reactor physics codes DRAGON/DONJON," *Ann. Nucl. Energy*, vol. 31, no. 18, pp. 2139–2155, 2004, doi: 10.1016/j.anucene.2004.06.004.

[7] M. D. Tucker and D. R. Novog, "The impact of fueling operations on full core uncertainty analysis in CANDU reactors," *J. Nucl. Eng. Radiat. Sci.*, vol. 6, no. 3, pp. 1–8, 2020, doi: 10.1115/1.4045485.

[8] F. N. McDonnell and R. E. Green, "Spatial Power Control in Large Candu Reactors: the Whys and

Wherefores.," 1978.

[9] United States Nuclear Regulatory Commission, "TRACE V5.1262 User's Manual Volume 1: Input Specification," Washington, 2019.

[10] T. Downar, A. Ward, Y. Xu, *et al.*, "PARCS v3.3.1 Release Volume I: Input Manual," Washington, 2018.

[11] United States Nuclear Regulatory Commission, "The Exterior Communications Interface," Washington.

[12] S. Younan and D. R. Novog, "Development and Testing of TRACE/PARCS ECI Capability for Modelling CANDU Reactors with Reactor Regulating System Response," *Sci. Technol. Nucl. Install.*, vol. 2022, p. 7500629, 2022, doi: 10.1155/2022/7500629.

[13] C. J. Allan, "A New Self-Powered Flux Detector," Chalk River, 1979.

[14] D. Kastanya, "Preliminary assessment of the impact of extrapolation distance on the results of Time-Average calculations in CANDU analyses," *Ann. Nucl. Energy*, vol. 166, p. 108815, 2022, doi: 10.1016/j.anucene.2021.108815.

[15] Oak Ridge National Laboratory, "SCALE Code System," Oak Ridge, 2017.

[16] K. B. Bekar, C. Celik, G. Davidson, *et al.*, "CSAS/Shift Integration in SCALE 6.3," *2019 SCALE Users' Group Workshop*, 2019. https://www.ornl.gov/file/csasshift-integration-within-scale-63/display

7 Conclusions

7.1 Summary of Findings

This thesis has investigated the use of novel analysis tools and methodologies for the analysis of CANDU reactors. This included the use of the SCALE/Polaris lattice physics code as well as the use of the PARCS core physics code and TRACE system thermalhydraulics code. These codes were modified and adapted to be applied to the needs of CANDU analysis. By leveraging multipurpose tools, CANDU applications can take advantage of the greater level of support offered for these tools, given their broader userbase compared to CANDU-specific tools.

In the case of Polaris, for which the work is described in Chapter 4, significant modifications were made to the components of the code related to input processing and geometry building, to permit the specification of a CANDU cluster-type geometry as an alternative to the existing PWR and BWR geometry capabilities. No fundamental limitations were found in Polaris for modelling arbitrary geometry; the existing geometry restrictions were found to be solely a product of how the geometry input is structured. Polaris uses a streamlined geometry input where, rather than constructing the geometry from specifying arbitrary surfaces and cells as needed, a particular type of geometry and its parameters are specified. For example, a PWR lattice would specify the lattice dimensions, lattice pitch, pin geometries and arrangement. The code itself is then responsible for converting this input into the cells and surfaces that make up the geometry. Thus, support for CANDU cluster-type geometry was provided by adding the capability to specify the parameters pertaining to a CANDU geometry, such as pin ring pitches, pin count per ring, along with pressure tube and calandria tube radii, and then construct the geometry.

In evaluating the Polaris code for CANDU lattice physics calculations, a particular emphasis was placed on the embedded self-shielding method (ESSM) [3] as this was a novel addition with little prior knowledge for CANDU applications. Other components of Polaris, such as its Method of Characteristics (MoC) transport solver, were considered to be well-understood, as the Method of Characteristics is commonly used in other transport codes, such as DRAGON [4]. Overall, the lattice properties calculated using Polaris were comparable to those from other SCALE sequences, including TRITON/NEWT using CENTRM self-shielding as well as Monte Carlo calculations using both continuous-energy and multi-group KENO. While the ESSM did not offer the same level of rigour as CENTRM for self-shielding, it offered far quicker performance without a significant discrepancy in the evaluated lattice properties. When the self-shielded cross-sections calculated by ESSM were evaluated alongside those from CENTRM against those from tallying a continuous-energy Monte Carlo KENO, greater discrepancies were found for ESSM,

especially in the thermal energy range. However, it was also found that these discrepancies could be greatly reduced, and made comparable to CENTRM, by recalculating the self-shielding factors used by ESSM, using a model more representative of a CANDU lattice, rather than using the distributed self-shielding factors based on LWR lattices. Such a self-shielding factor recalculation performs a series of CENTRM calculations to effectively "calibrate" the relationship between the self-shielding factors and the lattice geometry and composition.

In the case of PARCS and TRACE, for which the work is described in Chapters 5 and 6, previous studies have benchmarked the use of these codes for CANDU modelling [25]. However, this work took advantage of the built-in coupling capabilities between PARCS and TRACE, avoiding the extra overhead that would be needed to use an external coupling script. The native coupling between PARCS and TRACE is capable of transferring node powers and thermalhydraulic properties between the two codes on every time step, with an additional file provided to PARCS to specify the mapping between PARCS nodes and TRACE components. However, CANDU analysis places additional demands on the code coupling capability, as reactivity devices are controlled in real time by a digital computer, using the readings from flux detectors and instrumented fuel channels. Therefore, it was necessary to update PARCS and TRACE to fully support coupling of reactivity device positions as well as to get flux and power readings from the PARCS model. Reactivity device coupling was already partially implemented prior to this work, though this thesis made the feature fully functional and corrected a number of identified issues with PARCS reactivity device movement during transients.

Coupling of additional models to TRACE is provided through the Exterior Communications Interface (ECI). While the libraries provided to create ECI-enabled tools were written in Fortran, this thesis developed an interface for Python programs to use the ECI to couple with TRACE, providing a much more convenient alternative to Fortran for coupling scripts or models of low computational intensity. This was used to develop and couple a model of the CANDU reactor regulating system (RRS) to the TRACE model, and thus indirectly to the PARCS model.

This thesis demonstrated that the combination of the TRACE/PARCS coupling interface and the ECI, with the necessary code modifications, named TRACE_Mac1.1 and PARCS_Mac1.1, respectively, can be used to model a variety of CANDU analysis cases. These cases range from simple benchmarks and tests, to short-duration transients modelling safety-relevant events against station data, such as a loss of forced flow, to long-duration operational simulation against station data, such as a period of shim operation. The resulting simulations produced results comparable to either expected results or actual station data.

In Chapter 5, it was shown that internally coupled models could produce similar results to externally coupled models for both a variety of accident conditions in a CANDU-6 code coupling benchmark, as well as for non-normal conditions in a 900 MW CANDU reactor model, such as figure-of-eight flow oscillations and loss of flow. In the case of figure-of-eight oscillations, when header interconnects were blocked and the reactor operated in an abnormal condition to increase outlet voiding, flow oscillation would occur that matched the expected properties of such oscillations. When a loss of forced flow due to loss of Class IV power was modelled, the predictions from the TRACE model running TRACE_Mac1.0 were comparable to the station data as well as prior models in RELAP5. In this work, as the transients were driven primarily by thermalhydraulics, the accuracy of the thermalhydraulic model was deemed to be more important than the accuracy of the core physics model – only the bulk behaviour of the physics model was important. Indeed, in Chapter 6, the physics model, based on a generic 900 MW class CANDU reactor, was found to be inadequate, despite being adequate for the work in Chapter 5, and needed to be updated with station-specific parameters.

In Chapter 6, it was shown that using the codes TRACE_Mac1.1 and PARCS_Mac1.1, along with a model of the RRS, that a series of coupled simulations, executed by a driver script, could simulate shim operation at a CANDU 900 station and produce similar results to the real station. As this simulation was more physics-driven than the simulations of Chapter 5, it was important to modify the generic CANDU physics model to use parameters specific to the CANDU 900 station being modelled. Specifically, it was important to correctly predict spatial flux distribution, spatial flux control, and reactivity feedback effects from reactivity devices. In addition, the sensitivity analysis found that it was crucial that adjuster rod depletion was accounted for. While the reference simulation predicted similar results to the station data, it underpredicted the total duration of the operation (i.e., the simulation ended on an earlier date than the actual shim operation). In the sensitivity cases, the adjuster rod depletion, adjuster rod reactivity worth, and nuclear data uncertainty were found to significantly affect the transient, particularly when it was extended by simulating a power reduction to 59%FP. However, nuclear data uncertainty could not explain the entire discrepancy in shim duration to the station data. Liquid zone reactivity worth and the burnup distribution of the fuel were found to only have a small effect on the shim operation.

7.2 Discussion of Future Work

7.2.1 Use of Polaris and other SCALE codes for CANDU Analysis

Chapter 4 concerned the development of the Polaris lattice physics code to add CANDU modelling capabilities. However, for the work in Chapters 5 and 6, it was chosen to use existing tools, primarily TRITON/NEWT, for lattice physics modelling. The primary

reason for this choice was that the goal of the work was to evaluate the implementation of the coupled TRACE/PARCS and RRS methodology, and for Chapter 6 to also evaluate the shim transient methodology. Thus, it was decided to stick to lattice physics methodologies for which significant experience already exists, as using a novel lattice physics tool would complicate the analysis, as a comparative analysis between multiple lattice physics tools would be required to evaluate the accuracy of the Polaris results.

However, it is still desirable to apply Polaris to core physics calculations with thermalhydraulic feedback, as the verification and validation efforts performed thus far in Chapter 4 were limited to evaluating lattice properties (k_∞, coolant void coefficient, fuel temperature coefficient, compositions of depleted fuel) along with evaluating the self-shielding calculation (comparing the self-shielded cross-sections calculated using different methods and also sensitivities to these cross-sections). Further evaluation would involve applying the homogenized cross-sections and other homogenized fuel properties to a core physics code, such as PARCS, and perform a variety of calculations using core physics with thermalhydraulic feedback, such as TRACE/PARCS, using homogenized cross-sections provided from different tools, including Polaris, TRITON/NEWT, KENO, Serpent, MCNP, and DRAGON. Ideal candidates for simulation include benchmark models which are well-characterized and potentially have known results. Other well-studied transients such as LOCA, loss of flow, loss of reactor regulation, or various operational occurrences that perturb neutronics and/or thermalhydraulics may also be considered. It is also recommended to perform uncertainty analysis using Sampler and simulate the system models with perturbed homogenized cross-sections. While Polaris and NEWT were well-correlated for predicting k_∞ and fuel temperature coefficient when perturbing cross-sections with Sampler, they were poorly correlated for predicting coolant void reactivity [34]. This indicates that Polaris can predict different sensitivities to the nuclear data when compared to other lattice physics solutions, which will propagate to uncertainty analysis in the coupled core physics/system thermalhydraulics model. Further evaluation is necessary to determine if these differences are acceptable, and under which applications, as well as to determine if methodology changes or code upgrades may be necessary. One potential methodology change would be to run IRFFACTOR on each perturbed library, thus “calibrating” the self-shielding factors independently for each sample. Potential code upgrades would include implementing alternative self-shielding methodologies, such as a subgroup method. In addition, multi-group nuclear data libraries with a larger number of energy groups could be developed to lessen the impact of the self-shielding calculation, at the expense of performance.

In Chapter 6, one limitation of the methodology used for uncertainty propagation is that the nuclear data uncertainties, applied via random perturbed cross-section libraries

using Sampler, were not applied to the reactivity device incrementals, due to not having a suitable 3D multicell code in SCALE. Version 6.3 of SCALE will include a new tool, Shift, which is a Monte Carlo code that can be used for the multicell calculation and that can be used with Sampler. Future studies should include the device incrementals in the analysis of the nuclear data uncertainties once SCALE 6.3 is officially released.

7.2.2 TRACE, PARCS, and ECI Coupling Capabilities

This work identified a number of potential future improvements that could be made to the TRACE and PARCS codes to facilitate the types of analysis performed in this thesis or other similar analyses on CANDU reactors. One such upgrade would be to make PARCS arrays more easily accessible to TRACE and the ECI for coupling purposes. Currently, the built-in coupling between PARCS and TRACE does not expose any information to the ECI, and only a limited amount of information can be exposed indirectly, through signal variables that link to PARCS. Even with this, TRACE_Mac1.1 needed to implement additional signal variables to expose the PARCS nodal powers and fluxes to the ECI.

The proper implementation of flux detectors within PARCS and TRACE would greatly facilitate this type of work. Currently, TRACE possesses a signal variable for PARCS detector signals, but it was found to be non-functional. TRACE also possessed a non-functional signal variable for PARCS nodal powers, which was made functional in TRACE_Mac1.0 by adding temporary code to the PARCS/TRACE coupling. In addition, the placement of PARCS detectors at arbitrary using the DET_LOC card was found to be faulty, mapping the detectors to an incorrect set of nodes, inconsistent with the requested detector location. The correction of these issues would allow for detector coupling to be handled directly by PARCS and TRACE, with ECI-coupled scripts only needing to retrieve the detector readings and pass them to modules that use them.

TRACE was also found to have difficulty with handling a very large number of signal variables and control blocks (>10000), when used by other control blocks (e.g., for summation blocks summing multiple signal variables). This is due to TRACE looking up the requested variable on each time step in the list of signal variables, rather than remembering its location, and also performing a linear search to locate the variable. This was overcome by implementing the summation in an ECI-coupled script, as ECI *does* save the memory locations of its variables, avoiding repeated searching. It is recommended, however, that future releases of TRACE improve the efficiency of handling large numbers of signal variables and control blocks.

PARCS was also found to have a number of limitations and inefficiencies during this work. Nodal methods were found to perform poorly for CANDU models over the course of this work, with the analytical nodal method (ANM) failing to converge while the nodal

expansion method (NEM) produced results inconsistent with expectations, overpredicting the flux near the periphery of the core. Only the most basic kernel, the finite difference method (FDM), produced acceptable results. Further testing is required, as these nodal methods may be more reliant on lattice properties not typically used for CANDU analysis, such as assembly discontinuity factors (ADFs).

In addition, it was impractical to define the reactivity devices in the most desirable manner due to the limitations in defining the PARCS mesh. In CANDU-specific core physics codes, such as RFSP, the mesh for which the flux is solved, and for which homogenized few-group cross-sections are calculated, can be defined as an arbitrary subdivision of the fuel channel/bundle mesh. This is usually done to apply reactivity device incremental cross-sections over a consistent volume (usually one channel pitch by one bundle length). This capability currently does not exist in PARCS (the subdivisions will be treated as separate channels/assemblies), thus for this work the device incremental cross-sections were instead spread over a larger volume as necessary, with appropriate weighting factors for each node occupied by the device. This is not ideal as the nodalization was found to influence the simulated reactivity worth of these reactivity devices.

Two potential solutions exist that may be pursued. The first is to add support to PARCS for subdividing assembly meshes for the purpose of applying device incremental cross-sections on a sub-assembly level, taking a similar approach to RFSP. The second solution is to consider the rod cusping correction feature in PARCS [35], which typically handles partially rodded nodes in the context of a rod being partially inserted into a node in its path of travel. This cusping correction is currently designed only for LWR geometries (rods inserted along the Z direction, parallel to assemblies). A three-dimensional cusping correction may be able to mitigate the effect of the nodalization. However, it may come with a significant performance penalty as many nodes would be “partially rodded”, as opposed to only a few in an LWR model. It may also be possible to pre-calculate a correction to the node weights of the reactivity devices using a fine-mesh model, similar to the methodology used in the cusping correction.

Further evaluation of the coupled RRS model should also be carried out. The shim transient was found to be relatively insensitive to specific RRS parameters, such as controller gains or maximum reactivity change rates, as the transient was driven primarily by depletion and xenon transients which are slow enough that RRS keeps the reactor close to equilibrium. The primary RRS sensitivity for shim operation is the total reactivity worth of each reactivity device. However, other transients may be more sensitive to the RRS response, and it is desirable to model an accurate response. Standalone testing (no coupling) of RRS modelling should be performed using station measurements (e.g.,

detector readings) as input, to validate that the modelled RRS matches the station response given the same input.

7.3 Conclusion

This thesis has presented how analysis tools can be adapted to the needs of modelling CANDU reactors. The modifications made to these tools for the work of this thesis, or similar modifications, will potentially be implemented into future distributions of the tools themselves, to enable the analyses performed in this work for a wider userbase. In addition, the PyECI library may be used to develop Python programs or scripts that can be coupled to a TRACE model. The results of this work may be applied to a variety of analyses for CANDU operation and safety. It also showed that the SCALE, PARCS, and TRACE codes may be combined for best-estimate-plus-uncertainty (BEPU) analysis, with nuclear data uncertainties being propagated through the lattice physics calculation using Sampler, to a coupled PARCS/TRACE reactor model with RRS response, to quantify the uncertainty for an anticipated operational occurrence (AOO).

8 References

[1] D. Knott and A. Yamamoto, "Lattice Physics Computations," in *Handbook of Nuclear Engineering*, Boston: Springer, 2010, pp. 913–1239. doi: 10.1007/978-0-387-98149-9_9.

[2] Oak Ridge National Laboratory, "SCALE Code System," Oak Ridge, 2017.

[3] M. A. Jessee, W. A. Wieselquist, T. M. Evans, *et al.*, "Polaris : a New Two-Dimensional Lattice Physics Analysis Capability for the Scale Code System," *Physor*, p. 14, 2014.

[4] G. Marleau, A. Hebert, R. Roy, and A. Hébert, "A User Guide for DRAGON Version 4," Montréal, 2016.

[5] A. Ward, Y. Xu, and T. Downar, "GenPMAXS – v6.2," 2016.

[6] R. E. MacFarlane, "Neutron Slowing Down and Thermalization," in *Handbook of Nuclear Engineering*, Boston: Springer, 2010, pp. 189–277. doi: 10.1007/978-0-387-98149-9_9.

[7] M. a Jessee, W. a Wieselquist, M. L. Williams, and K. S. Kim, "Lattice Physics Benchmarking," in *Transactions of the American Nuclear Society*, 2013, vol. 109, no. 2012, pp. 1413–1415. doi: 10.1007/s10967-012-2210-3.2.

[8] M. A. Jessee, W. A. Wieselquist, C. A. Gentry, and U. Mertyurek, "BWR geometry enhancements for the Polaris lattice physics code," in *Transactions of the American Nuclear Society*, 2017, vol. 117, no. October, pp. 1301–1305.

[9] A. Labarile, R. Miró, T. Barrachina, and G. Verdú, "TRITON vs POLARIS . Comparison Between Two Modules for LWRs Modelling in SCALE 6 . 2," in *24th International Conference Nuclear Energy for New Europe*, 2015, pp. 1–10.

[10] UNENE, *The Essential CANDU - A Textbook on the CANDU Nuclear Power Plant Technology*. Hamilton: University Network of Excellence in Nuclear Engineering (UNENE), 2019. [Online]. Available: https://www.unene.ca/education/candu-textbook

[11] R. Pristavu and A. Rizoiu, "Reactor Physics Assessment of Modified 37-Element CANDU Fuel Bundles," in *Proceedings of NUCLEAR 2016 the 9th annual international conference on sustainable development through nuclear research and education Part 1/3*, 2016, pp. 111–120.

[12] H. Choi and C. J. Park, "Benchmark calculation of WIMS/RFSP against physics measurement data of Wolsong nuclear power plants," *Nucl. Sci. Eng.*, vol. 159, no. 2, pp. 153–168, 2008, doi: 10.13182/NSE159-153.

[13] B. Rouben, “RFSP-IST, The Industry Standard Tool Computer Program for CANDU Reactor Core Design and Analysis,” Mississauga. [Online]. Available: https://www.nrc.gov/docs/ML0236/ML023600318.pdf

[14] M. D. Dehart, “Lattice Physics Capabilities of the SCALE Code System Using TRITON,” 2006.

[15] F. Mascari, G. Vella, V. Casamassima, and F. Parozzi, “Analyses of Trace-Parcs Coupling Capability,” in *Nuclear Energy for New Europe 2011*, 2011, pp. 816.1-816.9.

[16] H. Choi, G. Roh, and D. Park, “Benchmarking MCNP and WIMS/RFSP against measurement data - II: Wolsong nuclear power plant 2,” *Nucl. Sci. Eng.*, vol. 150, no. 1, pp. 37–55, 2005, doi: 10.13182/NSE05-A2500.

[17] Y. Liu, W. Martin, M. Williams, and K. S. Kim, “A full-core resonance self-shielding method using a continuous-energy quasi-one-dimensional slowing-down solution that accounts for temperature-dependent fuel subregions and resonance interference,” *Nucl. Sci. Eng.*, vol. 180, no. 3, pp. 247–272, 2015, doi: 10.13182/NSE14-65.

[18] R. Berry, L. Zou, H. Zhao, J. Peterson, and H. Zhang, “RELAP-7 : Demonstrating Flow Simulation in a Single- Pipe , Two-Phase Reactor Core and Steam Separator / Dryer,” 2013.

[19] United States Nuclear Regulatory Commission, “TRACE V5.1262 Theory Manual,” Washington, 2019.

[20] D. J. Richards, J. A. Lorenc, and W. I. Midvidy, “Verification of SOPHT for Parallel Channel Flow Stability,” in *Numerical Methods in Nuclear Engineering*, 1983, pp. 201–225.

[21] W. Lui, W. Yousef, J. Pascoe, A. Tomasone, M. Williams, and J. C. Luxat, “TUF: A two-fluid code thermal-hydraulic analysis,” 1989.

[22] B. N. Hanna, “CATHENA: A thermalhydraulic code for CANDU analysis,” *Nucl. Eng. Des.*, vol. 180, no. 2, pp. 113–131, 1998, doi: 10.1016/S0029-5493(97)00294-X.

[23] C. D. Fletcher and R. R. Schultz, “RELAP5/MOD3 Code Manual,” Idaho Falls, 1995.

[24] F. Zhou and D. Novog, “RELAP5 simulation of CANDU Station Blackout accidents with/without water make-up to the steam generators,” *Nucl. Eng. Des.*, vol. 318, pp. 35–53, 2017.

[25] K. Groves, “A TRACE/PARCS Coupling, Uncertainty Propagation and Sensitivity Analysis Methodology for the IAEA ICSP on Numerical Benchmarks for Multi-Physics Simulation of Pressurized Heavy Water Reactor Transients,” McMaster

University, 2020. [Online]. Available: http://hdl.handle.net/11375/25927

[26] W. Won, Y. Jiang, M. Kwee, and J. Xue, “Bruce NGS A Loss of Flow Analysis for Effectiveness of Level 2 Defence-in-Depth Provisions,” 2014.

[27] United States Nuclear Regulatory Commission, “TRACE V5.1262 User’s Manual Volume 1: Input Specification,” Washington, 2019.

[28] United States Nuclear Regulatory Commission, “MAPTAB Input Manual and User Guide,” Washington, 2017.

[29] United States Nuclear Regulatory Commission, “The Exterior Communications Interface,” Washington.

[30] C. R. Harris, K. J. Millman, S. J. van der Walt, *et al.*, “Array programming with NumPy,” *Nature*, vol. 585, no. 7825, pp. 357–362, Sep. 2020, doi: 10.1038/s41586-020-2649-2.

[31] P. Peterson, “F2PY: a tool for connecting Fortran and Python programs,” *Int. J. Comput. Sci. Eng.*, vol. 4, no. 4, pp. 296–305, 2009.

[32] J. Koclas, *Reactor Control and Simulation*. Montréal: École Polytechnique de Montréal, 1996.

[33] D. Kastanya, S. Boyle, J. Hopwood, and J. H. Park, “The impact of power coefficient of reactivity on CANDU 6 reactors,” *Nucl. Eng. Technol.*, vol. 45, no. 5, pp. 573–580, 2013, doi: 10.5516/NET.03.2013.709.

[34] S. Younan and D. Novog, “Validation of the Polaris CANDU Extension for Lattice Physics,” in *PHYSOR 2020: Transition to a Scalable Nuclear Future*, 2020, p. 8.

[35] T. Downar, A. Ward, Y. Xu, *et al.*, “PARCS v3.3.1 Release Volume I: Input Manual,” Washington, 2018.

9 Appendix

9.1 Program Description – PyECI Python ECI Interface Package

The PyECI Python ECI interface package was developed to provide an interface for Python programs to interact with the Exterior Communications Interface (ECI). It uses the F2PY tool [31] to generate an interface between the Fortran 90 ECI library and Python, and creates an object-oriented API to access this interface.

9.1.1 ECI Library Summary

The ECI library is distributed with TRACE as source code that may be incorporated into a program to add ECI coupling support. The library is written primarily in Fortran 90 and is designed to be used by Fortran programs, though the module responsible for the low-level inter-process communications is written in C. A description of the relevant portions of the library follows, to provide a background for describing the features of PyECI.

ECI variables are based on scheduled requests. When a request is scheduled, ECI associates a local variable in the program (via pointer) to a variable that exists in another task in the coupled model (via variable name, component number, and location indices), along with the synchronization point at which the data transfer should occur, and the direction of the data transfer. Once all requests have been scheduled, a resolution takes place in all tasks. First, the owning task of each requested component is located. Then, for each requested variable, the link is completed by locating the variable in the owning task. The result is that each task constructs transfer tables, such that, for each request, at the given synchronization point, the value of the given variable in one task is copied to the variable in the other task. The location in memory of the variable on either end is stored as a pointer.

9.1.2 ECI Library Modifications

A number of modifications were made to the version of the Fortran 90 ECI library included with PyECI, to accommodate a number of PyECI features as well as to make general improvements to the ECI library:

- The task list file accepts spaces when parsing task arguments, allowing for multiple arguments to be passed to a task.
- Argument parsing was adapted to read arguments in from Python rather than reading them directly. PyECI is responsible for separating out ECI arguments from non-ECI arguments. ECI arguments are passed to the ECI library to be processed, while non-ECI arguments are placed in a list for the application to parse.
- The socket queue and task message buffer sizes are increased by a factor of 10.

9.1.3 Fortran-Python Interface

The ECI Library uses Fortran 90 pointers to store the memory locations of variables participating in the data transfer. Therefore, it is necessary to allocate memory for variable storage. Thus, an additional Fortran module was created, PyInterfaceM.f90, which contains two variables:

- `values`, an allocatable array of real-valued variables.
- `nvars`, the number of variables allocated.

While these variables are defined in a Fortran module, the allocation and use of these variables is handled in Python.

PyInterfaceM.f90 also contains several helper subroutines to be called in Python, which provide a bridge between the Python portion of PyECI and the ECI library. These subroutines are as follows:

- `SetMissing`, which is a wrapper around the `IsMissing` subroutine from the ECI library which is used to schedule requests. It takes the variable name, component number, location indices, the direction of transfer, the synchronization point, and the index in the `values` array at which to store the variable locally. The subroutine then translates the index into a Fortran pointer and calls the `IsMissing` subroutine.
- `EciInit`, which calls two ECI library subroutines: `ProcessArgs` and `SetTasks`. It receives both a list of arguments and the number of arguments, and passes them to `ProcessArgs`, which is modified to receive its arguments from `EciInit` rather than fetching the command line arguments directly. This is needed as the command line arguments may include both ECI arguments as well as program-specific arguments.
- `EciLoad`, which calls two ECI library subroutines: `ResolveComponents` and `SetExTransfers`, to complete the scheduling of data transfers. It receives a single optional logical argument, which, if set to True, it will also call `WriteTaskInfo`.
- `EciTransfer`, which receives the name of a synchronization point and calls the ECI library subroutine `ExtTransfer` with the same synchronization point. It then checks if the `errExit` flag, and, if it is set, terminates the program.

9.1.4 Python Package

PyECI also provides an object-oriented interface to access the underlying Fortran interface. This allows for a Python program to establish its ECI-linked variables as Python objects. PyECI includes a container class which manages the variables, along with a Variable class for accessing the individual variables.

The package's __init__.py includes the following classes:

- `VariableData`, which is a container of ECI-linked variables. It initially starts empty, but the `addVariable` and `addTransfer` methods are used to populate the container. The `setTransfers` method is used to allocate the `values` array from the interface and allocate each variable to this array. The `getVariable` method is used for variable retrieval.
- `Variable`, which contains the information for a single ECI-linked variable, including the ECI variable name, component number, location indices, a list of transfers, and the corresponding index in the `values` array. The `addTransfer` method is used to schedule transfers, while setTransfers receives an index or an index iterable and allocates the variable on the `values` array using `SetMissing`. The index is then stored for future reference. The `Variable` object has `get` and `set` methods to retrieve and change the stored value.
- `ArrayVariable`, which inherits from `Variable` and represents a NumPy array of ECI-linked variables. Each variable in the array can have the ECI variables and transfers specified independently, though the typical usage is to associate it with an array of the same ECI variable among either different components or different location indices.
- `FastArrayVariable` and `MaskArrayVariable`, as optimized versions of `ArrayVariable`.
- `Transfer`, which is a simple structure that stores the name of a synchronization point and the direction of transfer.

The package's __init__.py includes the following functions:

- `ReadArgs`, which reads the arguments passed to the Python program, and separates out the ECI arguments from program-specific arguments. ECI arguments are returned so they can be given to `EciInit`, while non-ECI arguments are placed in a list named `args`, which may be parsed by the program.
- `Prefix`, which returns the prefix, if any, passed to the program.
- `EciInit`, which calls both `ReadArgs` and the Fortran `EciInit`.
- `EciLoad`, which calls the Fortran `EciLoad`.

The package also includes a module named state.py, which includes several classes to facilitate developing a model with time step backup and checkpointing capabilities. It includes the following classes:

- `StateEncoder`, which inherits from `json.JSONEncoder` and is used to write state data to a JSON file, providing checkpointing capability.

- `State`, which encapsulates a time-dependent system, storing the state as a dictionary, with one old-time copy and one new-time copy. It also includes the ability to save and load data to/from a JSON file, as well as being able to store a hierarchy of "child" `State` objects to facilitate the dump/restart feature.
- `StateContainer`, which encapsulates a `State` object and can be inherited from by the program. These derived classes may then possess additional variables alongside the time-dependent state.
- `MasterStateContainer`, which is a `StateContainer` object with the additional capability of acting as the top level of the `State` hierarchy, storing other `StateContainer` objects, and storing the file names for dump and restart capabilities.

The package also includes a module named cblock.py, which includes a class named `TracCBlockOutput`. This class allows for a variable in a `State` object to be linked to a `Variable` object. It includes a single method, `write`, which automatically copies the value of the `State` variable to the `Variable` object. The purpose of this class is for the automatic recording of data from a `State` object in the Python model to a TRACE control block, allowing for visualization in SNAP and AptPlot.

9.1.4.1 Class: VariableData

The `VariableData` class is a container object responsible for managing ECI-linked variables. To use PyECI, a program may construct an instance of the `VariableData` class, then use its methods to set up ECI variables and send that information to the ECI library. The `VariableData` class contains the following variables:

- `variables_list`: A Python list of `Variable` objects. This is initialized as an empty list.
- `variables`: A Python dictionary mapping variable names to `Variable` objects. This is initialized as an empty dictionary.
- `allocated`: A Boolean which indicates whether the `VariableData` object has allocated its variables onto the ECI library. This is initially False.

The `VariableData` class contains the following methods:

- `addVariable(self, name, varName, compNum, i1, i2, i3)`: Constructs a `Variable` object using the attributes `varName`, `compNum`, `i1`, `i2`, and `i3`. The variable is appended to `variables_list` and added to `variables` using `name` as the key.
- `addArrayVariable(self, name, shape, varName, compNum=0, i1=0, i2=0, i3=0)`: Constructs an `ArrayVariable` object using the attributes

shape, varName, compNum, i1, i2, and i3. The variable is appended to variables_list and added to variables using name as the key.

- addTransfer(self, name, syncPoint, direction, idx): Adds a transfer to the variable named by name. This looks up the Variable object from the variables dictionary and calls its addTransfer method.
- setTransfers(self): This performs the actual variable allocation in the ECI library. It allocates the values array in PyInterfaceM.f90 to the correct size to accommodate its variables. It then loops over variables_list and allocates each variable by calling its setTransfers method. Once this method is called, allocated is set to True.
- getVariable(self, i): Returns a Variable object. If I is an integer, it is used as the index for variables_list. If i is a string, it is used as the key for variables.

9.1.4.2 Class: Variable

The Variable class establishes a reference to one element of the variable array of PyInterface, as well as scheduling the ECI data transfers to and from that location. The Variable class contains the following variables:

- varName: The name of the variable in the remote program (e.g., TRACE).
- compNum: The component number of the component whose variable is to be accessed.
- i1, i2, i3: Location indices to locate an element of an array, when the named variable is an array variable. This can be used to locate a specific cell or edge for a component.
- transfers: A Python list of ECI transfers to schedule for this variable. Initially empty.
- idx: The index of the values array where the value of this variable is stored.
- allocated: A Boolean which indicates whether the Variable object has been allocated in ECI. Initially False.

The Variable class contains the following methods:

- addTransfer(self, syncPoint, direction): Adds a transfer to schedule by creating a Transfer object with the corresponding synchronization point and direction. This is appended to transfers.
- setTransfers(self, idx): Allocates this variable to the index given by idx, and schedules all ECI transfers using SetMissing. The index is saved to self.idx. Returns the value of idx plus one.

- `get(self)`: Returns the value stored in the ECI variable array corresponding to this variable.
- `set(self,a)`: Sets the value stored in the ECI variable array corresponding to this variable to `a`.
- `view(self)`: Returns a NumPy view of the location in the ECI variable array corresponding to this variable. A view refers directly to the ECI-coupled memory which is updated at each synchronization point.

In addition, the `Variable` object supports many operators. Most comparisons and operations will automatically call the `get` method, whereas in-place operators such as += will call the `set` method.

9.1.4.3 Class: ArrayVariable

The `ArrayVariable` class establishes an array of references to elements of the variable array of PyInterface, as well as scheduling the ECI data transfers to and from that location. Otherwise, it functions similarly to `Variable`. Internally, this is performed as a set of individual real-valued ECI variables, rather than as a single ECI variable representing the entire array. While this approach is less efficient, it is more flexible.

The correspondence between the `ArrayVariable` elements and ECI variables are stored as arrays of the same shape. Each array element can thus be configured independently, and can thus refer to completely unrelated ECI variables, different components, and/or different location indices, and can have transfers configured independently. However, typical use cases are expected to involve an array of related variables, such as an individual property (e.g., pressure) across multiple components or locations within a single component, or an array of related control blocks.

The `ArrayVariable` is initialized by providing a shape as well as arrays for each of the arguments used to configure the variables. These arrays are automatically broadcast to the full shape of the `ArrayVariable`, providing ease of use for cases where one or more properties are similar between all elements in the array (e.g., the same variable name).

The `ArrayVariable` class contains the following variables:

- `shape`: The shape of the array.
- `varName`: The name of the variable in the remote program (e.g., TRACE).
- `compNum`: The component number of the component whose variable is to be accessed.

- `i1`, `i2`, `i3`: Location indices to locate an element of an array, when the named variable is an array variable. This can be used to locate a specific cell or edge for a component.
- `transfers`: A Python list of ECI transfers to schedule for this variable. Initially empty.
- `idx`: The index of the `values` array where the value of this variable is stored.
- `allocated`: A Boolean which indicates whether the `Variable` object has been allocated in ECI. Initially False.

The `ArrayVariable` class contains the following methods:

- `addTransfer(self, syncPoint, direction, idx)`: Adds a transfer to schedule by creating a `Transfer` object with the corresponding synchronization point and direction. This is appended to `transfers`. The `idx` argument specifies the indices in the array for which to apply the transfer. If not provided, the transfer is applied to the entire array.
- `setTransfers(self, idxiter)`: Allocates this variable to the indices given by `idxiter`, and schedules all ECI transfers using `SetMissing`. The `idxiter` variable is incremented as each variable in the array is allocated. These indices are stored in the `idx` array. Returns the final value of `idxiter`, corresponding to the next index after the end of this array.
- `get(self)`: Returns an array of the values stored in the ECI variable array corresponding to this variable.
- `set(self,a)`: Sets the values stored in the ECI variable array corresponding to this variable to the values of `a`.
- `view(self)`: Returns a NumPy view of the location in the ECI variable array corresponding to this variable, reshaped to match the shape of the `ArrayVariable`. A view refers directly to the ECI-coupled memory which is updated at each synchronization point.

9.1.4.4 Class: FastArrayVariable

The `FastArrayVariable` class is an optimized version of the `ArrayVariable` class. It has several restrictions, including the requirement that every variable in the array has the same variable name and transfers (direction and synchronization point), with only the component number and location indices allowed to differ. Once allocated, it should be accessed by retrieving and manipulating a view of its data.

The `FastArrayVariable` class contains the following variables:

- `shape`: The shape of the array.

- `varName`: The name of the variable in the remote program (e.g., TRACE).
- `compNum`: The component number of the component whose variable is to be accessed.
- `i1`, `i2`, `i3`: Location indices to locate an element of an array, when the named variable is an array variable. This can be used to locate a specific cell or edge for a component.
- `transfers`: A Python list of ECI transfers to schedule for this variable. Initially empty.
- `idx`: The index of the `values` array where the value of this variable is stored.
- `allocated`: A Boolean which indicates whether the `Variable` object has been allocated in ECI. Initially False.

The `FastArrayVariable` class contains the following methods:

- `addTransfer(self, syncPoint, direction)`: Adds a transfer to schedule by creating a `Transfer` object with the corresponding synchronization point and direction. This is appended to `transfers`.
- `setTransfers(self, idx)`: Allocates this variable to the index given by `idx`, and schedules all ECI transfers using `SetMissingArray`. The index is saved to `self.idx`. Returns the value of `idx` plus the length of the array.
- `get(self)`: Returns an array of the values stored in the ECI variable array corresponding to this variable.
- `set(self,a)`: Sets the values stored in the ECI variable array corresponding to this variable to the values of `a`.
- `view(self)`: Returns a NumPy view of the location in the ECI variable array corresponding to this variable, reshaped to match the shape of the `FastArrayVariable`. A view refers directly to the ECI-coupled memory which is updated at each synchronization point. It is recommended to use `view` and manipulate the returned view object rather than `get` or `set` for performance.

9.1.4.5 Class: MaskArrayVariable

The `MaskArrayVariable` class is derived from `FastArrayVariable`, and can be used when variable data is shaped into an array but not every value in the array is required. For example, the simulation of neutron flux detectors may involve reading from a three-dimensional flux array, but data is only needed at the locations where flux detectors exist. This class adds an additional `mask` array to specify which parts of the array need ECI coupling. Zeros correspond to entries in the array that will not be ECI-coupled. When `setTransfers` is called, the Fortran subroutine `SetMissingMaskArray` is used for the allocation. This reduces the overhead of ECI data transfer by eliminating transfers

of data that are not used by the program. Otherwise, this class functions identically to `FastArrayVariable`.

The `MaskArrayVariable` class adds the following variable:

- `mask`: A mask array, initially set to all zeros, and in the same shape as `shape`. Array entries that set to 1 will indicate which parts of the array should be ECI-coupled.

9.1.4.6 Class: Transfer

The `Transfer` class is a simple structure containing the following elements:

- `direction`: The direction of transfer, either "get" or "put".
- `syncPoint`: The name of the synchronization point at which the data transfer shall occur.

9.1.4.7 Class: State (state.py)

The `State` class is used to facilitate the setup of models which support both time step backup/redo capability as well as checkpoint/restart capability. Time step backup/redo is supported by storing two copies of each variable, with one for the old time and one for the new time. At the beginning of each time step, the new-time variables are copied to the old-time variables. Computations to the model are then performed on the new-time variables. If a time step backup is requested, the new-time variables are restored from the old-time variables, reverting the state of the object to what it was at the beginning of the time step. Checkpoint/restart capability is implemented by implementing the capability to serialize the full state into a JSON encoding, as well as to decode the JSON encoding to restore its state.

The `State` class provides a hierarchical structure, allowing for a state to contain one or more "child" states. This hierarchy serves two purposes. Its main purpose is to allow for large models consisting of multiple `State` objects to be serialized into a single JSON object. It also simplifies the time step backup/redo capability by allowing the method to be automatically performed on all child states.

The `State` class consists of the following variables:

- `new`: A Python dictionary representing the current, or new-time, state of the model represented by this object.
- `old`: A Python dictionary representing the old-time state of the model represented by this object, i.e., its state at the beginning of the current time step.
- `time`: The problem time for this object corresponding to the new-time state.
- `oldtime`: The problem time for this object corresponding to the old-time state.
- `childrenL`: A Python list of unnamed child `State` objects.

- `children`: A Python dictionary of named child `State` objects.

The `State` class consists of the following methods:

- `addChild(self, name, childState)`: Adds a named child state object.
- `addChildL(self, childState)`: Adds an unnamed child state object.
- `newStep(self, recursive)`: Copies the new-time state to the old-time state, and if the `recursive` argument is True, performs the same on all descendants.
- `backupStep(self, recursive)`: Copies the old-time state to the new-time state, and if the `recursive` argument is True, performs the same on all descendants.
- `delt(self)`: Returns the difference between `time` and `oldtime`.
- `interpolate(self, name, time)`: For the state variable named by `name`, performs a linear interpolation (or extrapolation) between the new-time and old-time states, to the time given by `time`.
- `resetTime(self, t)`: For this state and descendants, shifts both `time` and `oldtime` by the same amount such that `time` becomes equal to `t`.
- `pack(self)`: Constructs a dictionary of this object's variables for serialization.
- `serialize(self)`: Constructs a string with a JSON representation of this object.
- `unserialize(self, s)`: Given a JSON string, decodes it and unpacks its values into this `State` object's variables.
- `dump(self, fname)`: Writes a JSON representation of this object and all children to a file.
- `restart(self, fname)`: Reads the JSON representation of this object and all its children and updates their respective values.
- `unpack(self, d)`: From a dictionary representing the decoded JSON string, the values of this state's variables along with all child states are updated to match the values being loaded. It is required that the structures and hierarchy of the states have already been initialized and match the structure being loaded, as this function will not create missing child states.
- `unpackState(self, d, s)`: Copies the values of elements of dictionary `d` to dictionary `s`.

9.1.4.8 Class: StateContainer (state.py)

The `StateContainer` class was created to represent a model or model component which includes both a `State` object as well as additional variables or parameters separate from the functionality of the `State` object. The `StateContainer` class is a base class from which a program should inherit from, as opposed to being used on its own.

The `StateContainer` class consists of the following variables:

- `state`: The contained `State` object.

The `StateContainer` class consists of the following methods:

- `newStep(self, recursive)`: Calls `state.newStep`.
- `backupStep(self, recursive)`: Calls `state.backupStep`.
- `get(self, name)`: Retrieves the new-time value of a named state variable.
- `set(self, name, value)`: Modifies the new-time value of a named state variable.
- `advance(self, newtime)`: Advances the state to the given new time. Derived classes should extend this method to perform model calculations.
- `dump(self, fname)`: Calls `state.dump`.
- `restart(self, fname)`: Calls `state.restart`.
- `serialize(self)`: Calls `state.serialize`.
- `unserialize(self, s)`: Calls `state.unserialize`.
- `pack(self)`: Calls `state.pack`.
- `unpack(self, d)`: Calls `state.unpack`.
- `resetTime(self, t)`: Calls `state.resetTime`.

9.1.4.9 Class: MasterStateContainer (state.py)

The `MasterStateContainer` class is a `StateContainer` which is designed to hold the top-level state in a hierarchy. It can also have associated dump and restart file names for checkpoint capability. The purpose of this class is to act as a single object to manage the checkpoint/restart capability of a model.

The `MasterStateContainer` class consists of the variables of `StateContainer` along with the following variables:

- `dump_file`: The name of the file to which to write JSON checkpoints.
- `restart_file`: The name of the file from which to read JSON checkpoints.
- `children`: A Python dictionary of child `StateContainer` objects.

The `MasterStateContainer` class consists of the methods of `StateContainer` along with the following methods:

- `add(self, name, child)`: Adds the `StateContainer` given by `child` with the name `name`. Also adds the child container's `State` as a child of the master container's `State`.
- `remove(self, name)`: Removes the child StateContainer with the name `name`. Also removes the child container's `State` from being a child of the master container's `State`.

- `dump(self)`: Calls `state.dump` with the file name `dump_file`.
- `restart(self)`: Calls `state.restart` with the file name `restart_file`.

9.1.4.10 Class: TracCBlockOutput (cblock.py)

The `TracCBlockOutput` class is used to establish a link between a variable contained within a `State` object and an ECI variable referenced by a `Variable` object. Its intended purpose was to allow a variable within the model to become the output of a TRACE control block, hence the name `TracCBlockOutput`. This can be used as part of driving a control system, or simply for visualization in SNAP or AptPlot. A single method, `write`, is used to perform the data transfer. There is no automatic synchronization of this data transfer with ECI; it is up to the program using this class to call `write` at an appropriate point in the time step loop.

The `TracCBlockOutput` class consists of the following variables:

- `cblock`: The destination `Variable` object (or other compatible object with a `set` method or index assignment).
- `obj`: The `StateContainer` which contains the source variable.
- `var`: The name of the source variable in the `StateContainer`'s `State`.
- `idx`: The index of the source variable when the source variable is an array. Default is `None`.
- `cidx`: The index of the destination variable when the destination variable is an array. Default is `None`.
- `f`: A function object which, if present, is applied to transform the value of the source variable before setting the destination variable. Default is `None`.

The `TracCBlockOutput` class consists of the following methods:

- `write(self)`: Performs the data transfer from the source variable to the destination variable.

9.1.5 ECI Programs and the TimeEvolve Template

PyECI includes a template for the time step loop in the form of a `TimeEvolve` function. This function is equivalent to the Fortran `TimeEvolve` template included in the original ECI library, converted to Python. This template consists of the synchronization points along with the necessary program flow to match with TRACE, including time step backup and iteration capability. All calculations to be performed by the user program are to be added to the `TimeEvolve` function.

A user's program should thus perform the following actions in order:

- Call the `EciInit` function.

- Construct a `VariableData` object, create all of the necessary variables for the simulation, then call its `setTransfers` method.
- Call the `EciLoad` function.
- Call the `TimeEvolve` function.

Other program-specific functions may be performed at any time where appropriate. For example, parsing of program arguments can be performed at any time after `EciInit`. The user should use `pyeci.args` instead of the usual `sys.argv` for parsing command-line arguments. The list `pyeci.args` contains the same arguments as `sys.argv`, except with the ECI-related arguments filtered out. Thus, `pyeci.args` should match the list of arguments provided in the `taskList` file. The user may use `sys.argv` if they wish to access ECI-related arguments specifically. Loading data from a checkpoint file or program-specific input files, and initializing program-specific models, would also be done prior to `TimeEvolve`.

9.2 Lattice Physics Branches used in PARCS Models

Table 9.1: Lattice Physics Branches used in Chapter 5 Loss of Flow Model

Branch Type	Rod	Coolant Density (g/cm^3)	Moderator Poison (ppm)	Fuel Temperature (K)	Coolant Temperature (K)
Reference	0	0.81212	0.00	941.29	560.66
Adjuster 1	**1**	0.81212	0.00	941.29	560.66
Adjuster Guide Tube	**2**	0.81212	0.00	941.29	560.66
LZC Type 1 Full	**3**	0.81212	0.00	941.29	560.66
LZC Type 1 Empty	**4**	0.81212	0.00	941.29	560.66
LZC Type 2 Full	**5**	0.81212	0.00	941.29	560.66
LZC Type 2 Empty	**6**	0.81212	0.00	941.29	560.66
LZC Type 3 Full	**7**	0.81212	0.00	941.29	560.66
LZC Type 3 Empty	**8**	0.81212	0.00	941.29	560.66
MCA	**9**	0.81212	0.00	941.29	560.66
MCA Guide Tube	**10**	0.81212	0.00	941.29	560.66
SOR	**11**	0.81212	0.00	941.29	560.66
SOR Guide Tube	**12**	0.81212	0.00	941.29	560.66
Coolant Density	0	**0.00000**	0.00	941.29	560.66
	0	**0.10000**	0.00	941.29	560.66
	0	**0.21212**	0.00	941.29	560.66
	0	**0.41212**	0.00	941.29	560.66
	0	**0.61212**	0.00	941.29	560.66
	0	**1.01212**	0.00	941.29	560.66
Fuel Temperature	0	0.81212	0.00	**300.00**	560.66
	0	0.81212	0.00	**541.29**	560.66
	0	0.81212	0.00	**641.29**	560.66
	0	0.81212	0.00	**741.29**	560.66
	0	0.81212	0.00	**841.29**	560.66
	0	0.81212	0.00	**891.29**	560.66
	0	0.81212	0.00	**991.29**	560.66
	0	0.81212	0.00	**1041.29**	560.66
	0	0.81212	0.00	**1200.00**	560.66
	0	0.81212	0.00	**1500.00**	560.66
	0	0.81212	0.00	**1800.00**	560.66
	0	0.81212	0.00	**2400.00**	560.66
Coolant Temperature	0	0.81212	0.00	941.29	**350.00**
	0	0.81212	0.00	941.29	**450.00**
	0	0.81212	0.00	941.29	**510.66**
	0	0.81212	0.00	941.29	**610.66**

Table 9.2: Lattice Physics Branches used in Chapter 6

Branch Type	Rod	Coolant Density (g/cm^3)	Moderator Poison (ppm)	Fuel Temperature (K)	Coolant Temperature (K)
Reference	0	0.81212	0.00	941.29	560.66
Dummy	**1**	0.81212	0.00	941.29	560.66
Adjuster Type 1	**2**	0.81212	0.00	941.29	560.66
Adjuster Type 2	**3**	0.81212	0.00	941.29	560.66
Adjuster Type 3	**4**	0.81212	0.00	941.29	560.66
Adjuster Type 4	**5**	0.81212	0.00	941.29	560.66
Adjuster Guide Tube	**6**	0.81212	0.00	941.29	560.66
LZC Type 1 Full	**7**	0.81212	0.00	941.29	560.66
LZC Type 1 Empty	**8**	0.81212	0.00	941.29	560.66
LZC Type 2 Full	**9**	0.81212	0.00	941.29	560.66
LZC Type 2 Empty	**10**	0.81212	0.00	941.29	560.66
LZC Type 3 Full	**11**	0.81212	0.00	941.29	560.66
LZC Type 3 Empty	**12**	0.81212	0.00	941.29	560.66
MCA	**13**	0.81212	0.00	941.29	560.66
MCA Guide Tube	**14**	0.81212	0.00	941.29	560.66
SOR	**15**	0.81212	0.00	941.29	560.66
SOR Guide Tube	**16**	0.81212	0.00	941.29	560.66
Coolant Density	0	**0.61212**	0.00	941.29	560.66
	0	**1.01212**	0.00	941.29	560.66
Moderator Poison	0	0.81212	**0.35**	941.29	560.66
Fuel Temperature	0	0.81212	0.00	**600.00**	560.66
	0	0.81212	0.00	**841.29**	560.66
	0	0.81212	0.00	**1041.29**	560.66
Coolant Temperature	0	0.81212	0.00	941.29	**510.66**
	0	0.81212	0.00	941.29	**610.66**

Notes:

- LZC Type 1 has one feeder tube and two scavenger tubes, and is used for the centre middle compartments and upper side compartments.
- LZC Type 2 has one scavenger tube, and is used for the lower compartments.
- LZC Type 3 has two feeder tubes and three scavenger tubes, and is used for the upper centre compartments.
- Dummy branch used in Table 9.2 as PARCS was found to miscalculate overlapping rod type 1 with other rod types when the total weight exceeded 1; this issue was identified during the work described in Chapter 6.