INL/RPT-24-80449

Light Water Reactor Sustainability Program

A Full-scale Demonstration of Pressurized Water Reactor Core Design Optimization using Multi-Cycle Optimization Methodology

September 2024

U.S. Department of Energy

Office of Nuclear Energy

INL/RPT-24-80449

A Full-scale Demonstration of Pressurized Water Reactor Core Design Optimization using Multi-Cycle Optimization Methodology

Junyung Kim[1]
Mohammad Abdo[1]
Congjian Wang[1]
Geon Kim[1]
Svetlana Lawrence[1]
Juan Cristhian Luque Gutierrez[2]
Nicholas Rollins[2]
Jason Hou[2]

[1] Idaho National Laboratory
[2] North Carolina State University

September 2024

Prepared for the
U.S. Department of Energy
Office of Nuclear Energy
Under DOE Idaho Operations Office
Contract DE-AC07-05ID14517

Page intentionally left blank

EXECUTIVE SUMMARY

The U.S. nuclear utilities encounter a difficulty in upholding essential safety standards while also securing economic viability for continued operation. Safety stands as a pivotal factor across all facets of operations within light water reactor nuclear power plants. Achieving economic feasibility alongside safety can be facilitated through the utilization of a risk-informed framework, exemplified by the ongoing development within the Risk-Informed Systems Analysis (RISA) Pathway under the auspices of the U.S. Department of Energy's LWRS Program. This initiative advocates for a diverse array of research and development endeavors aimed at optimizing both safety and economic efficacy within nuclear power plants, particularly pertinent as many plants contemplate subsequent license renewals.

The RISA Pathway has two main goals: deploy methodologies and technologies that better represent safety margins, cost, and safety factors and develop advanced applications that enable cost-effective plant operation. This report assesses the potential for resolving multi-cycle plant reload challenges through real-world scenarios utilizing the Plant ReLoad Optimization (PRLO) framework. This framework offers reactor core design developers analytic tools of reactor safety and fuel performance with the assistance of artificial intelligence (AI) to enhance core design solutions. Multi-objective genetic algorithm alongside acceleration techniques is considered as an enabling technology for improving fuel efficiency while upholding safety thresholds. The demonstration of multi-cycle core design optimization is performed, then results are compared to benchmarks. This report investigates the practical application of the PRLO platform in addressing real-world core design challenges and contrasting outcomes with those derived from heuristic or conventional algorithms.

The cost of fuel is a critical consideration for nuclear power plants. Various factors can be manipulated to reduce the cost, including batch size, enrichment levels, and the use of burnable poison. Alongside these economic factors, meeting cycle energy requirements and operation and safety constraints, such as the hot zero power moderator temperature coefficient (MTC) and hot channel factor ($F_{\Delta H}$), is imperative. The large number of design variable combinations at both the lattice and core levels, along with multiple design objectives and constraints, make single-cycle optimization highly challenging. Furthermore, the current cycle's core loading influences subsequent cycles, establishing interdependencies between reload cycles in both boiling water reactors (BWRs) and pressurized water reactors (PWRs). From an optimization standpoint, sequential cycle-by-cycle optimization with no coupling between cycles may not yield optimal results from multi-year operation standpoint. Therefore, multi-cycle core design throughout the planning horizon becomes crucial when the goal is to minimize multi-cycle fuel costs or, at the very least, to link sequential cycles through carefully formulated objectives in the plant reload process.

The plant reload optimization using AI offers several benefits discussed below:

Efficiency - AI can analyze large volumes of data to optimize plant reload processes quickly and efficiently. It can identify patterns, correlations, and trends that may not be apparent to humans, leading to more efficient process of core design.

Cost Reduction - By optimizing core designs, AI can reduce costs associated with volume of fresh fuel and spent fuel needing processing, down-power, and inventory management.

Improved Safety - Optimized core design can enhance safety by reducing the risk of accidents and incidents. AI can identify potential hazards of the suggested core designs and recommend preventive measures to mitigate risks.

Enhanced Performance - AI-driven optimization can improve the overall performance of the plant by maximizing safety margins potentially reducing the need to down-power during plant operation which can lead to increased productivity, higher throughput, and better overall performance metrics for the plant.

Adaptability – AI algorithms possess the adaptability to learn and adjust to various plant designs and operating conditions. This ability ensures that the system remains efficient and effective, regardless of alterations in the environment or operational parameters.

In this fiscal year, the PRLO framework, in which the Risk Analysis Virtual Environment (RAVEN) plays a pivotal role, has been upgraded to include optimization capabilities for *n*-th cycle and consecutive cycles for a PWR. As part of this enhancement, the RAVEN-SIMULATE3 interface was expanded to handle fuel label maps in addition to just fuel types, enabling the optimization process to account for fuel assemblies from previous cycles. To showcase this new capability, single-cycle optimization cases with two distinct objectives were analyzed. Furthermore, a case involving the optimization of two consecutive cycles, starting from the 10th cycle with a specific fuel Inventory Management strategy, was also examined. The reactor model used for these optimizations was a generic AP1000 model based on open-source data. In the two single-cycle optimization cases, the goal was to exceed the performance of an equilibrium cycle reference design using the same inventory. For the two-consecutive-cycle case, the aim was to optimize each cycle individually to achieve superior fuel performance compared to the reference design across both cycles. The results demonstrated that the PRLO framework effectively guided the design process, yielding solutions that outperformed the reference model in all scenarios tested.

Page intentionally left blank

CONTENTS

FIGURES

TABLES

Page intentionally left blank

ACRONYMS

AI	artificial intelligence
API	application programming interface
AS	active subspaces
BOC	beginning of cycle
BWR	boiling water reactor
DBA	design basis accident
DHM	decreasing high mutation
EFPD	effective full power day
EOC	end of cycle
FDM	finite difference method
FY	fiscal year
FA	fuel assembly
GA	genetic algorithm
ILC	increasing low crossover
INL	Idaho National Laboratory
LP	loading pattern
LWRS	Light Water Reactor Sustainability
ML	machine learning
MOEA	multi-objective evolutionary algorithm
MOOP	multi-objective optimization problem
MTC	moderator temperature coefficient
NEI	Nuclear Energy Institute
NSGA-II	Non-Dominated Sorting Genetic Algorithm II
PMX	partially mapped crossover
PRLO	plant reload optimization
PWR	pressurized-water reactor
RAVEN	Risk Analysis and Virtual Environment
RISA	Risk-Informed Systems Analysis
SC	shuffling scheme
SPSA	simultaneous perturbation stochastic approximation
ZDT	Zitzler, Deb and Thiele

Page intentionally left blank

1. INTRODUCTION

The U.S. Department of Energy (DOE) Light Water Reactor Sustainability (LWRS) Program's Risk-Informed Systems Analysis (RISA) Pathway Plant ReLoad Optimization (PRLO) project is a pivotal initiative designed to address one of the top-priority needs of the nuclear power industry –improved efficiency, safety, and economic viability of nuclear energy. The initiative is a part of a broader strategy to enhance the economic sustainability and competitiveness of light water reactors (LWRs), which play a critical role in the U.S. energy landscape. According to the Nuclear Energy Institute's (NEI) Nuclear Costs in Context report [1], fuel costs constitute approximately 18% of the total generation costs as shown in Figure 1. In 2021, total electricity generation from nuclear power was reported to be 778 TWh. [2] The total fuel cost for pressurized water reactors (PWRs) is estimated at $3,041.78 million ($48.28 million per reactor unit), while the total fuel cost for boiling water reactors (BWRs) is estimated at $1,576.95 million ($50.87 million per reactor unit). This substantial expense underscores the importance of optimizing fuel use to maintain economic viability of the U.S. nuclear fleet. APPENDIX A shows the estimated fuel cost of the U.S. nuclear operating plants.

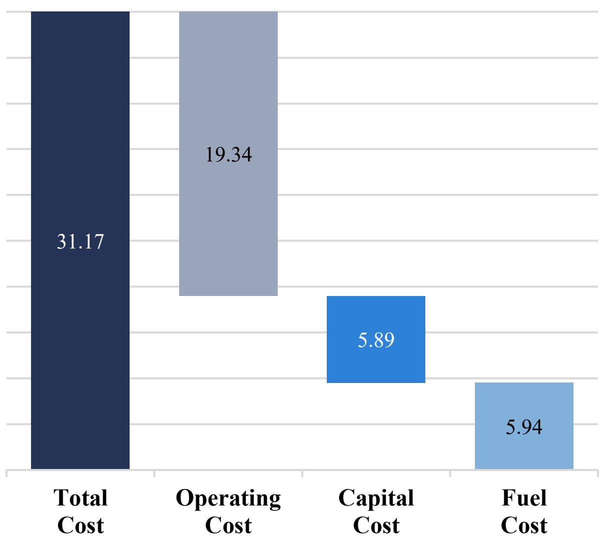


Figure 1. U.S. Nuclear Plant (PWR) Costs in 2021. Unit: $/MWh. [1]

The effective utilization of nuclear fuel involves efficiently using nuclear materials within reactors, ensuring safety, and maximizing energy output while minimizing costs. The efficient utilization of nuclear fuel implies reducing the frequency of refueling outages, which are both time-consuming and expensive, and lowering the cost of new fuel batches. By extending the operational cycle of reactors through improved fuel use, plants can increase their uptime and reduce the costs associated with shutdowns and restarts. Enhancing reactor core design can enable a smaller fresh fuel batch to generate the same amount of electricity. This enhancement not only reduces new fuel costs but also significantly decreases expenses in the back-end fuel cycle by lowering the volume of spent fuel that needs processing. Figure 2 illustrates the factors influencing nuclear fuel utilization.

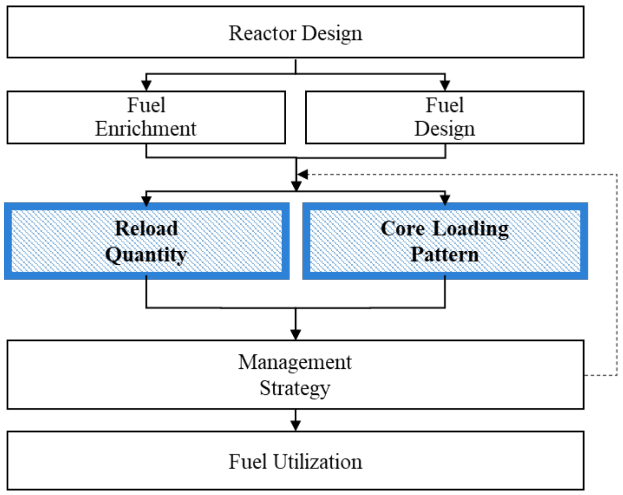


Figure 2. Factors influencing fuel utilization. [3] The highlighted section is the focus of the Plant ReLoad Optimization project.

Designing a nuclear reactor core is an exceptionally complex process due to several factors. First, the intricate integration of physics and engineering principles presents a significant challenge, particularly in accurately modeling neutron transport and diffusion, which is essential for predicting neutron behavior within the core. This requires a deep understanding of nuclear reactions and precise calculations to ensure efficient reactor operation. Additionally, thermal-hydraulic considerations must be meticulously balanced to effectively remove heat from the core while maintaining safe operating temperatures, preventing overheating, and ensuring structural integrity. Safety and regulatory requirements add further complexity, as nuclear reactors must adhere to stringent safety standards, necessitating rigorous safety protocols and comprehensive accident analyses for worst-case scenarios. Extensive testing and validation are often required, increasing the design's complexity and time. Material performance is another critical aspect, with fuel and cladding materials needing to withstand extreme conditions, such as high radiation and corrosive environments, over extended periods. Ensuring materials long-term durability and resistance to corrosion and wear is crucial for safe and efficient reactor operation, requiring advanced material science and engineering expertise. Economic considerations also play a significant role, as designers must balance initial fuel costs with long-term operational efficiency (e.g., single-cycle and multi-cycle optimization). Lastly, the design space for reactor cores is vast, with over 10^{30} possible combinations for a 17×17 PWR core design. In addition, traditional methods of deciding core loading pattern (LP) and reload quantity are labor-intensive and time-consuming.

The PRLO project aims to develop an integrated, comprehensive framework offering an all-in-one solution for reload evaluations with a special focus on optimization of core design. [4] The project is leveraging artificial intelligence (AI) – machine learning (ML) techniques to find optimal solutions, a nearly impossible task for humans due to a large design space. The aim of this study is to enhance nuclear reactor efficiency by improving the process of reloading fresh fuel and optimizing fuel shuffling scheme. This optimization could lead potential fuel cost savings for a single PWR unit in the order of 5 – 10% ($2M to $5M per year on average) where these savings apply solely to new fuel costs and do not include additional savings from reduced spent fuel processing costs.

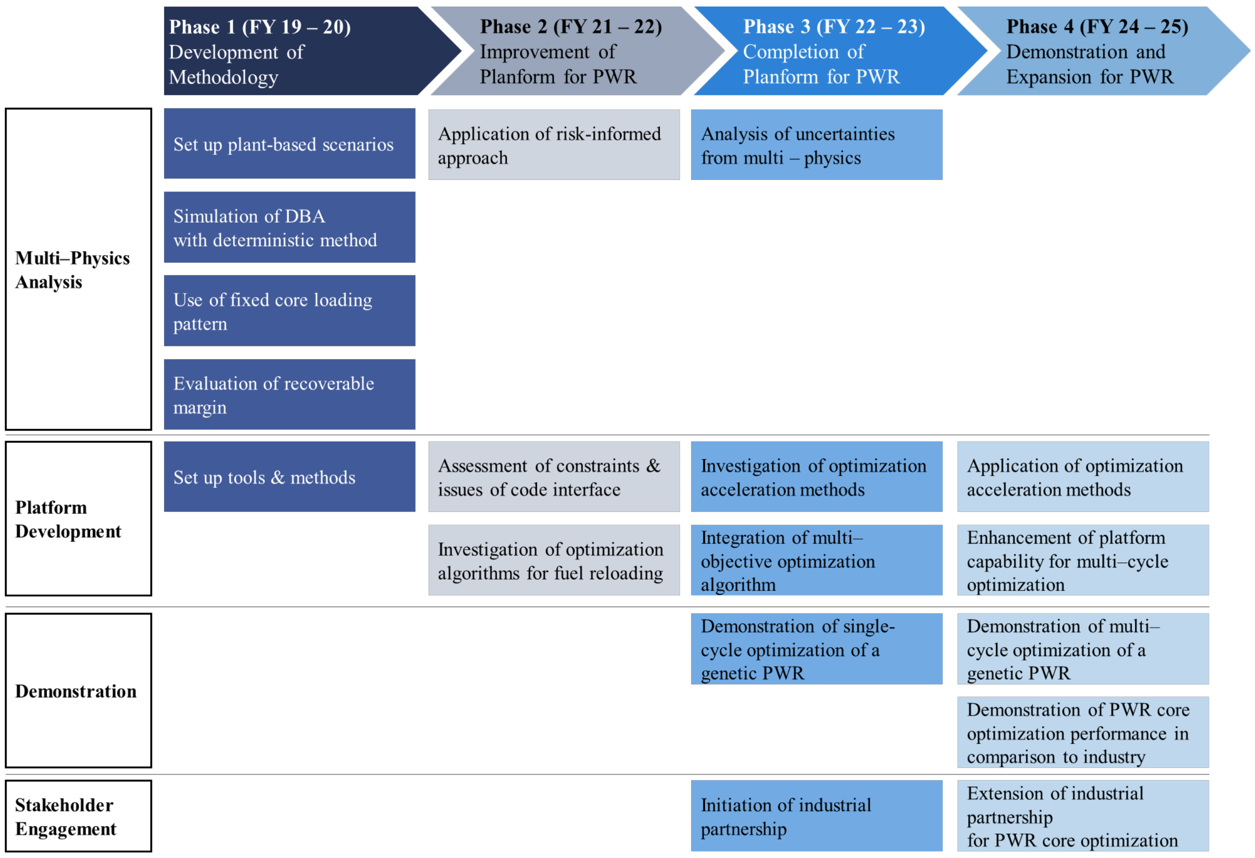

Phase 1 (FY 19 – 20) Development of Methodology
Phase 2 (FY 21 – 22) Improvement of Planform for PWR
Phase 3 (FY 22 – 23) Completion of Planform for PWR
Phase 4 (FY 24 – 25) Demonstration and Expansion for PWR
Multi–Physics Analysis
Set up plant-based scenarios
Application of risk-informed approach
Analysis of uncertainties from multi – physics
Simulation of DBA with deterministic method
Use of fixed core loading pattern
Evaluation of recoverable margin
Platform Development
Set up tools & methods
Assessment of constraints & issues of code interface
Investigation of optimization acceleration methods
Application of optimization acceleration methods
Investigation of optimization algorithms for fuel reloading
Integration of multi–objective optimization algorithm
Enhancement of platform capability for multi–cycle optimization
Demonstration
Demonstration of single-cycle optimization of a genetic PWR
Demonstration of multi–cycle optimization of a genetic PWR
Demonstration of PWR core optimization performance in comparison to industry
Stakeholder Engagement
Initiation of industrial partnership
Extension of industrial partnership for PWR core optimization

Figure 3. Technology roadmap of the PRLO project in Risk-Informed Systems Analysis Pathway of Light Water Reactor Sustainability Program.

Figure 3 illustrates the PRLO project's technology roadmap and research strategy spanning multiple fiscal years (FYs), comprising distinct phases of research and development. In Phase 1 (FY19 – 20), available tools and methods were investigated and tested for fuel reloading optimization. Plant-based design basis accident (DBA) scenarios were simulated using traditional deterministic methods with the RELAP5-3D thermal-hydraulic analysis code [5] developed at Idaho National Laboratory (INL). Simulations used fixed core loading and evaluated recoverable margins. In Phase 2 (FY21 – 22), the development of an optimization framework using the genetic algorithm (GA) and application programming interfaces (APIs) of nuclear system code in Risk Analysis and Virtual Environment (RAVEN) was initiated. During this development, RAVEN's capability to perform neutronics and thermohydraulic analyses was enhanced. As an initial test, ten limiting DBA scenarios for a generic PWR and a single objective optimization framework were developed. Additionally, constraints in computational tools, particularly in reactor core design and fuel performance system codes, were identified. [6] These constraints were reviewed through benchmark studies, and the applicability of the risk-informed approach for plant reload optimization was assessed. In Phase 3 (FY22 – 23), the project enhanced the framework with additional capabilities to support regulatory-required fuel safety analyses. The development and deployment of the multi – objective optimization process using the Non-Dominated Sorting Genetic Algorithm II (NSGA-II) in RAVEN were completed. The newly developed NSGA-II optimization framework was demonstrated with a constrained multi – objective optimization of a PWR core LP. In addition, the project has initiated engagement with Constellation Energy© for applications of fuel reload optimization framework deployments to industry. In Phase 4, the project focuses on demonstration of multi – cycle fuel reloading optimization showing effectiveness of the platform comparing to status-quo industrial approach for fuel reloading optimization. For FY24, two consecutive –

cycle fuel management problems have been solved leveraging the optimization framework and the outcomes of the optimization were benchmarked against genetic PWR fuel reloading data. For FY25, the project is targeting to evolve the platform to the equilibrium-cycle fuel management problem.

This report consists of four sections: Section 2 outlines the key features of RAVEN, which serves as the foundation of the PRLO framework, and provides the theoretical background of the optimization algorithm and multi-cycle optimization scheme. Section 3 presents the demonstrative study: single and multi-cycle optimization of a genetic AP1000 reactor core. Section 4 presents conclusions of the report and outlines future work.

2. DEVELOPMENT OF MULTI-CYCLE PLANT RELOAD OPTIMIZATION FRAMEWORK

2.1 Optimization Framework in RAVEN

The main driving force behind the PRLO framework is RAVEN developed at INL designed to facilitate advanced risk analysis, uncertainty quantification, and optimization tasks in various engineering domains, particularly focusing on nuclear energy systems. [7] The tool integrates state-of-the-art methodologies to provide a comprehensive framework for modeling, simulation, and decision support, making it an invaluable asset for enhancing the safety, efficiency, and reliability of nuclear reactors.

One of the core strengths of RAVEN is its powerful optimization capabilities. It supports a wide range of optimization algorithms, including gradient-based methods, evolutionary algorithms, and hybrid approaches, enabling users to tackle complex, multi-objective optimization problems. RAVEN's optimization framework is highly flexible and can be tailored to specific user needs. It supports not only the definition of custom objective functions and constraints, making it adaptable to various types of optimization problems, but also the tools necessary to find solutions that meet multiple criteria simultaneously. This is particularly useful in the nuclear industry, where decisions often need to balance competing objectives such as cost, safety, performance, and regulatory compliance. By incorporating these advanced optimization techniques, RAVEN allows engineers to identify optimal configurations and operational strategies that enhance reactor performance while minimizing risks and uncertainties.

Another significant feature of RAVEN is its robust API, which facilitates seamless integration with multiple nuclear system codes. This capability is critical for conducting comprehensive simulations and analyses that require coupling of different modeling tools. The API allows RAVEN to interface with established codes such as RELAP5-3D, SIMULATE3 [8] , PARCS [9], and BISON [10], among others, enabling the exchange of data and execution of coupled simulations. This interoperability ensures that users can leverage the strengths of various specialized tools within a unified framework, enhancing the accuracy and reliability of their analyses.

The API-driven integration also supports the execution of complex workflows involving multiple stages of simulation and analysis. For instance, in a typical nuclear safety assessment, one might need to perform thermal-hydraulic simulations, structural integrity analyses, and probabilistic risk assessments. RAVEN's API allows these tasks to be orchestrated in a coordinated manner, ensuring that the results from one stage can seamlessly inform the next. This capability not only streamlines the analysis process but also enhances the fidelity of the overall assessment by ensuring consistency across different modeling domains.

2.2 RAVEN – SIMULATE3 Interface Development

SIMULATE3, developed by Studsvik©, is a neutronics code designed to simulate nuclear fuel depletion within reactor cores. It plays a vital role in the detailed analysis of core behavior by solving neutron diffusion equations across various energy groups. The code is specifically tailored to model the time-dependent changes in nuclear fuel during fission, accurately capturing the evolution of isotopic compositions within fuel assemblies (FAs) over time. Renowned for its precision in predicting core reactivity, power distribution, and burnup, SIMULATE3 is a tool for optimizing reactor performance and ensuring safety. Additionally, the code integrates complex fuel management strategies, providing precise calculations that support informed decisions on fuel LPs and cycle lengths.

The RAVEN – SIMULATE3 interface aims to communicate RAVEN and SIMULATE3 during the optimization process. This interface supports generating SIMULATE3 input files and executes them, to later collect information needed for evaluating individual designs and inform the optimization process in GA. On top of the RAVEN – SIMULATE3 interface already developed in [11], new features allowing users having *n*-th cycle optimization (not necessarily the first cycle optimization) capabilities are added.

Figure 4 illustrates workflows for RAVEN – SIMULATE3 interface and data stream. The API requires three interface python script files: `SimulateInterface.py`, `SpecificParser.py`, and `SimulateData.py`, and three RAVEN and CMS (CasMo/Simulate) input files. The ‘`SimulateInterface.py`’ file acts as a bridge between other classes and methods in RAVEN and the input/output files of SIMULATE3. It runs two interface scripts: one to modify the input files and another to extract information from the output files. Additionally, it organizes the folder structure for the perturbed runs. The ‘`SpecificParser.py`’ file creates SIMULATE3 input files based on samples provided by RAVEN. For fresh core optimization, it generates input files for the LP, while for *n*-th cycle optimization, which is further explained in Section 2.3.1, it produces input files with a map of FA labels. The ‘`SimulateData.py`’ file extracts and stores information from SIMULATE3 output files after each run executed through RAVEN. As of the date of this publication, the variables that can be read and stored are time-dependent multiplication factor (k-eff), time-dependent heat flux hot channel factor (F_Q), time-dependent nuclear enthalpy rise hot channel factor ($F_{\Delta H}$), time-dependent critical boron concentration, cycle length determined by the critical boron concentration being 10 ppm, time-dependent relative pin power distribution, average burnup for each FA type at the end of cycle (EOC), maximum neutron leakage, core average burnup at the EOC, and front end fuel cost. The ‘`Sim3-param.xml`’ file is where the user can specify input information needed to generate SIMULATE3 input files, including reactor inlet temperature, pressure, power percentage, coolant mass flow, core width, number of FAs, and FA types. Table 1 shows node names used in `Sim3-param.xml` and their descriptions. The ‘`Sim3-perturb.xml`’ file defines the number of FAs which must match the length of chromosome in GA. The ‘`input.inp`’ file is in need as a placeholder for perturbed SIMULATE3 input deck. Sample scripts of XML files are listed in – SAMPLE RAVEN INPUT FILES FOR RAVEN – SIMULATE3 INTERFACE.

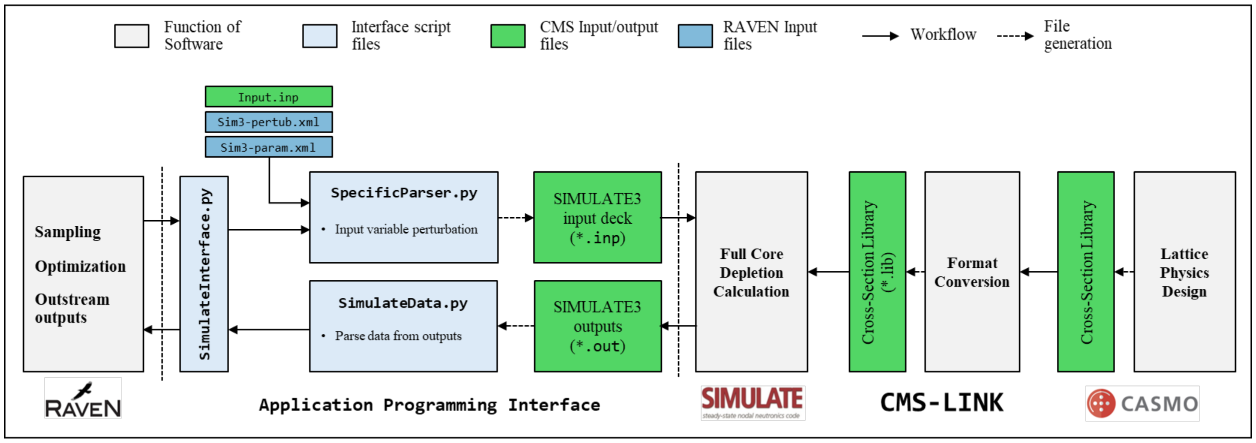


Figure 4. RAVEN – SIMULATE3 interface workflow.

Table 1. Node name and description in `Sim3-param.xml.`

Node Name	**Information Type**		**Description**
pins	Reactor Core Physics		Number of fuel pins across in a fuel assembly (FA)
core_width			Number of fuel assemblies (FAs) across in a reactor
load_point			Burnup step to read from restart file
depletion			Maximum burnup limit (unit: GWd/MT)
axial_nodes			The number of axial nodes for fuel assemble
active_height			Reactor active height (unit: cm)
batch			The cycle number that runs
pressure			Reactor pressure (unit: psia)
Boron			(Initial) estimated boron concentration (unit: pcm)
Power			Reactor power (unit: MWth)
Flow			Percentage of coolant massflow (unit: %)
inlet_temperature			Reactor inlet coolant temperature (unit: K)
map_size			Size of map to be printed in SIMULATE3 input file
Symmetry			Type of symmetry
restart_file			Name of restart file
cs_lib			Cross-section library name (including file extension)
number_assemblies			Number of FAs in the reactor core
working-dir			Working directory where out files will be saved.
FA-list	Fuel Assembly Label Map	name	Name for FA used in RAVEN
		FAid	ID for FA used in RAVEN
		type1 ~ type 4	Location in the reactor core map

2.3 Multi–Objective Optimization – Non-Dominated Sorting Genetic Algorithm II

The fuel reload optimization in nuclear reactors is inherently a multi-objective optimization problem due to the necessity to balance multiple, often conflicting objectives to achieve optimal performance and safety. Some primary objectives typically include minimizing the volume of new fuel, maximizing the utilization of existing fuel, and ensuring the reactor operates within safety limits. This must be balanced with the objective of maximizing the burnup of existing fuel to extract as much energy as possible from the current fuel inventory, thereby extending fuel life and reducing overall fuel consumption. Additionally, safety and regulatory constraints impose critical objectives that cannot be compromised. These include maintaining reactor power distribution within acceptable limits, and ensuring thermal margins to prevent overheating. Each of these safety constraints interacts with the economic goals, creating a complex trade-off landscape. For example, while increasing fuel burnup may reduce costs, it may lead to unsafe power peaking factors or compromise the structural integrity of FAs.

When a problem involves multiple objectives, it results in a set of optimal solutions known as Pareto-optimal solutions instead of a single optimal solution. In the absence of additional information, the solutions on the Pareto curve (or Pareto front) are assumed to be the optimal solutions, thus Pareto-optimal solutions. Traditional optimization methods, including multi-criteria decision-making techniques, recommend transforming the multi-objective optimization problem (MOOP) into a single-objective optimization problem by emphasizing one Pareto-optimal solution during single simulation. However, for a problem with multiple solutions, this approach needs to be applied multiple times, with each simulation expected to yield a different solution.

A MOOP includes a set of n decision variables, k objective functions, and a set of (m inequality and p equality) constraints. The optimization goal is:

$$\text{Min/Max } \boldsymbol{y}(\boldsymbol{x}) = \left(f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_k(\boldsymbol{x})\right), k \geq 2 \tag{1}$$

$$\text{Subject to } g_i(\boldsymbol{x}) \leq 0, i = 1, 2, \ldots, m \tag{2}$$

$$h_j(\boldsymbol{x}) = 0, i = 1, 2, \ldots, p \tag{3}$$

where $\boldsymbol{x} = (x_1, \ldots, x_n)$ is an n-dimensional decision vector in $\boldsymbol{x} \in \mathbb{R}^n$ ($\mathbb{R}$ is the set of real numbers), $\boldsymbol{y}$ is a k-dimensional objective vector in $\mathbb{R}^k$, f defines the mapping function, g_i is the i^{th} inequality constraint, and h_j is the j^{th} equality constraint.

If the following conditions are satisfied, $\boldsymbol{x}_1$ can be considered as superior to $\boldsymbol{x}_2$, where $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are the two feasible solution vectors of the multi minimization problem.

$$f_j(\boldsymbol{x_1}) \leq f_j(\boldsymbol{x}_2) \text{ for all } j = \{1,2, \ldots, k\}, \text{ and } f_j(\boldsymbol{x}_1) < f_j(\boldsymbol{x}_2) \text{ for at least one } j = \{1,2, \ldots, k\}, \tag{4}$$

where k is the number of objective functions and $f_j(\boldsymbol{x})$ is j^{th} value of an objective function for decision vector $\boldsymbol{x}$.

Here, the vector value $\boldsymbol{x}$ is the Pareto-optimal solution when it is not dominated by any other feasible solutions. The collection of all Pareto-optimal solutions is a Pareto set, and the objective vectors that correspond to the Pareto set are called a Pareto front, as illustrated in Figure 5.

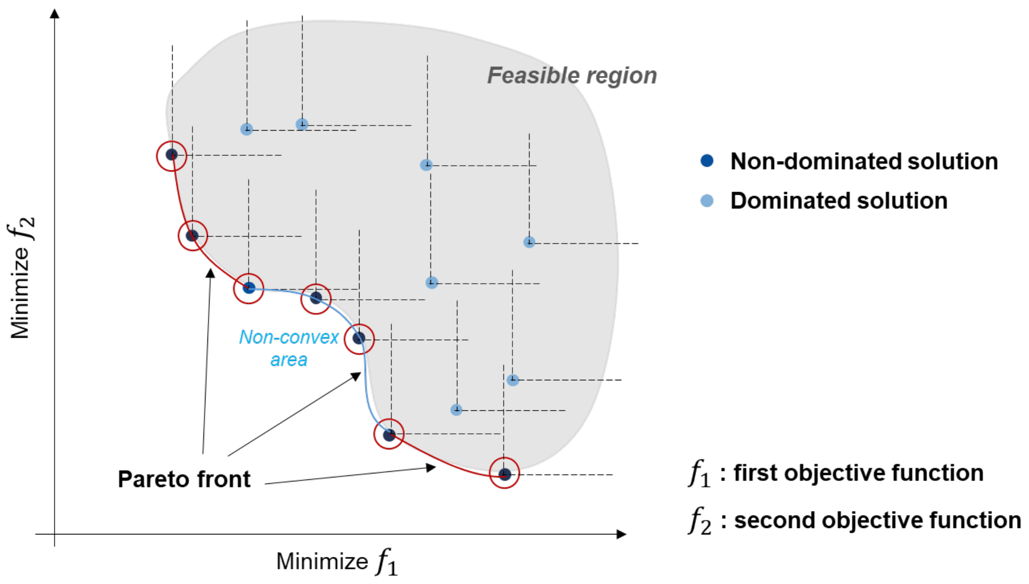


Figure 5. Pareto dominance in a multi objective optimization problem. [12]

Several multi-objective evolutionary algorithms (MOEAs) have been proposed with different purposes and applicability. APPENDIX B shows a summary of the different MOEAs. For the plant fuel reload optimization, the NSGA-II was selected for various reasons. Firstly, after testing it on multiple testing problems, NSGA-II showed an advantage in finding a wide range of solutions and converging characteristics compared to the other contemporary MOEAs [13]. NSGA-II, initially proposed by Deb et al. in 2000 [13], is a powerful GA-based method for solving MOOPs and problems with continuous and discrete variables. Furthermore, NSGA-II has shown its efficiency in managing many engineering optimization problems [14].

The NSGA-II optimization inherits definitions used in the GA method. For instance, the initial set of solutions—called a population—is composed of a chromosome, which is a vector of variables (called genes in NSGA-II). Figure 6 shows a schematic diagram of the population and its element.

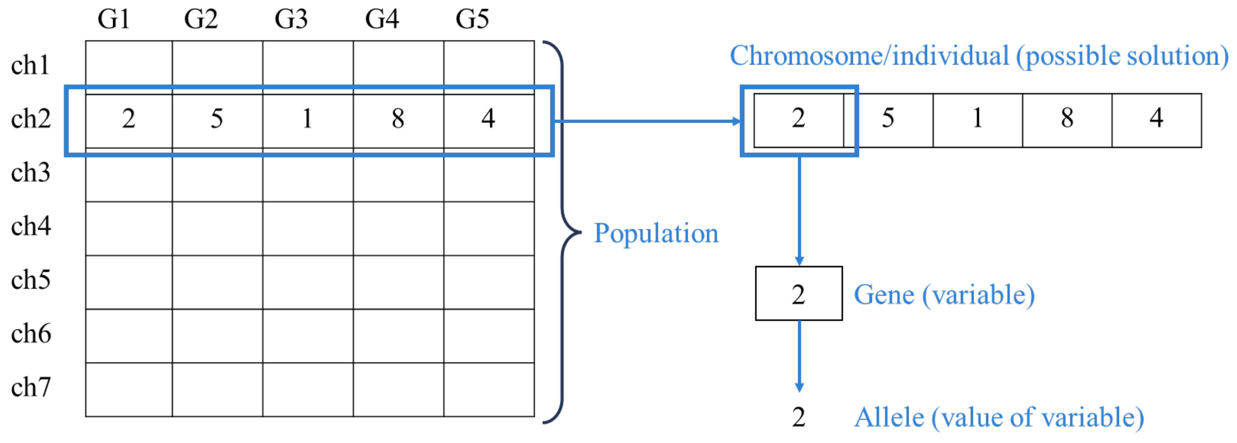


Figure 6. The set of potential solutions (population) and their elements. [4]

2.3.1 Dominance Depth Method

The dominance depth method sorts non-dominated solutions using the Pareto dominance concept. The non-dominated sorting procedure commences by allocating the initial population's non-dominated members to the first front (or so-called “rank” in NSGA-II). These members are then categorized into the first front and are removed from the initial population. The remaining population members undergo the dominance depth method. The non-dominated members of the residual population are then designated the second rank and added to the second front. This process is reiterated until all population members are grouped into different fronts based on their respective ranks. Figure 7 shows an example of the dominance depth method. The solutions are scattered and non-dominated in the left figure and sorted with four different Pareto fronts in the right figure.

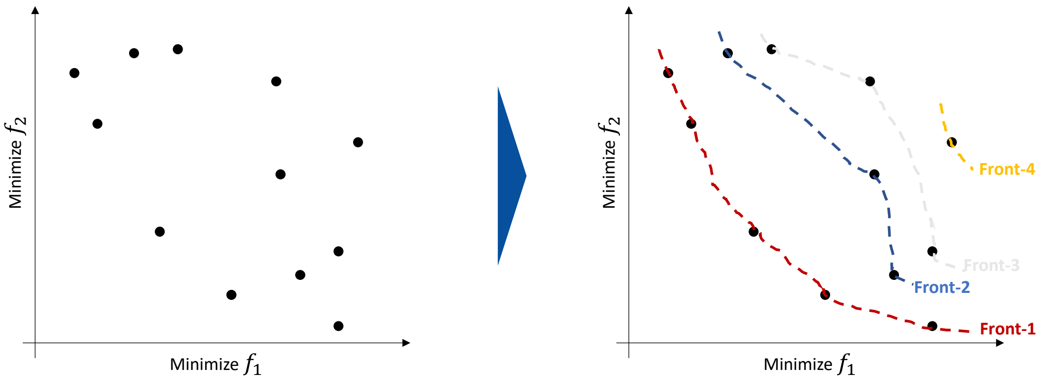


Figure 7. The dominance depth method. [12]

2.3.2 Elitism

Elitism, also known as the elite preserving strategy, is an essential concept that NSGA-II emphasizes. It conserves a population's elite solutions by directly transferring them to the succeeding generation. Put differently, the non-dominated solutions discovered in each generation proceed to the next generations until some solutions dominate them.

2.3.3 Crowding Distance

To assess the density of solutions surrounding a specific solution, the crowding distance is computed. It represents the average distance between two solutions on each side of the solution along each objective. When comparing two solutions that have different crowding distances, the one with the greater crowding distance is believed to exist in a less congested area. The ith solution's crowding distance is the average side length of the cuboid, as depicted in Figure 8. If f_j^i is the jth value of an objective function for the ith solution and f_j^{max} and f_j^{min} are the maximum and minimum values, respectively, of jth objective function among all the solutions, the crowding distance of ith solution is defined as the distance of the two nearest solutions on either side, as given in Equation (5).

$$cd(i) = \sum_{j=1}^{k} \frac{f_j^{i+1} - f_j^{i-1}}{f_j^{max} - f_j^{min}} \tag{5}$$

where k is the number of objective functions.

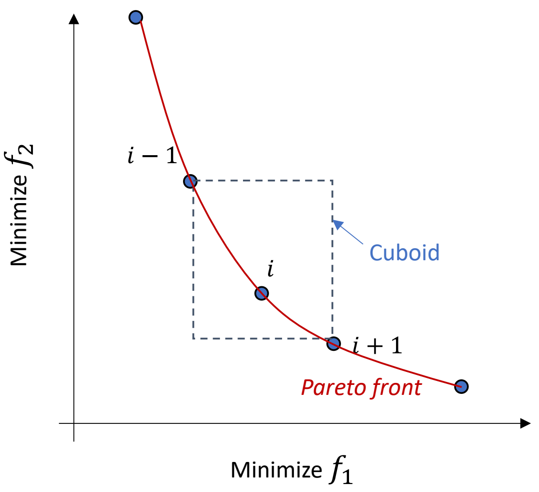


Figure 8. Cuboid with neighboring solutions for calculating crowding distance. [12]

2.3.4 Survivor Selection

The population for the next generation was selected using a tournament selection operator, which uses the rank of chromosomes and their crowding distances for selecting ones out of chromosomes for the next generation. The survivor selection process is:

[1] Select chromosomes that do not violate any constraints

[2] If both the chromosomes have different ranks, the one with the better rank is selected for the next generation

[3] If both the chromosomes are of the same ranks, the one with the higher crowding distance is selected for the next generation.

2.3.5 Optimization Procedures

The NSGA-II procedure begins with generating an initial population P(t=0) of size N, where t represents the number of iterations. Then a new population Q(t=0) (offspring) is created after performing crossover and mutation operations on the population P(t=0). After that, the population P(t=0) and Q(t=0) are combined to form a new population R(t=0) (which is the size of 2 × N), and the non-dominated sorting procedure is performed on R(t=0). Then the population members of R(t=0) are ranked into different fronts according to their non-domination levels.

The next process is to select N members from R(t=0) to create the next population P(t=1). If the size of the first front is greater than or equal to N, only N members are selected from the least crowded region of the first front to form P(t=1). On the contrary, if the size of the first front is less than N, the chromosomes of first front are directly transferred to the next generation, and the remaining members are taken from the least crowded region of the second front and added to P(t=1). If the size of P(t=1) is still less than N, the same procedure is followed for the next consecutive fronts until the size of P(t=1) becomes equal to N. The populations of P(t=2), P(t=3), …, are constructed following same procedure until the stopping criteria are satisfied. The NSGA-II procedure is shown in Figure 9.

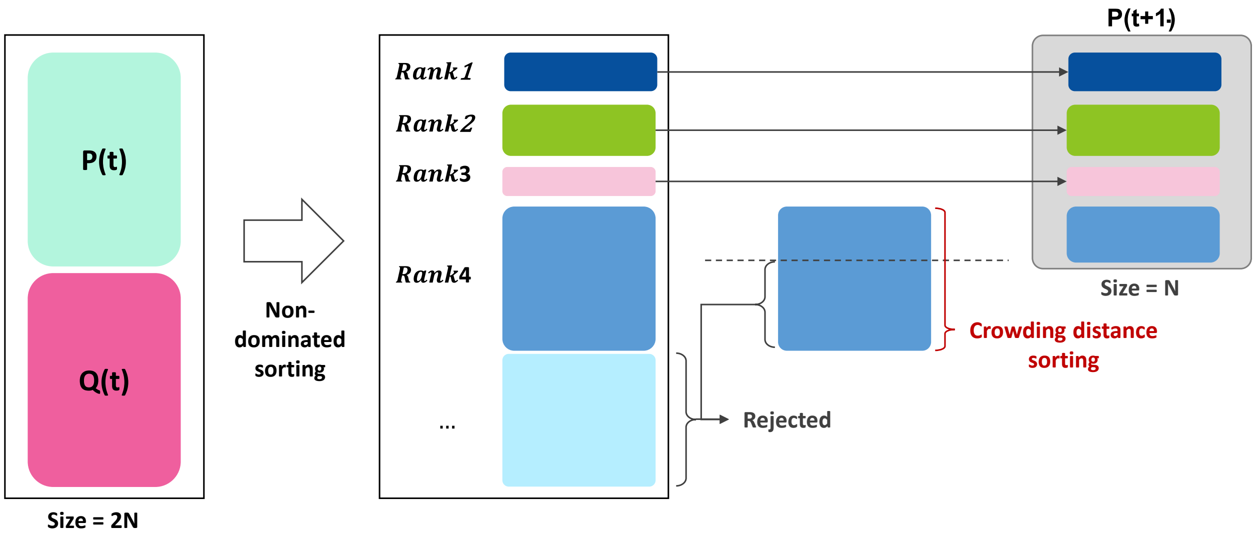


Figure 9. Procedure of NSGA-II. [12]

2.4 Multi–Cycle Fuel Reloading Optimization

The previous fuel-management optimization capabilities within the PRLO framework were centered on single cycle – single, or multi-objective optimization of fuel LPs for PWRs. [15] While this approach provided valuable insights, it primarily focused on the rearrangement of a fresh fuel inventory. In practice, fuel reload design is inherently a multi-cycle optimization challenge, as a common strategy in fuel management is to recycle FAs from previous cycles. This multi-cycle approach opens up a vast design space, making the single – optimization process across multiple cycles more complex due to the extensive search space that needs to be explored.

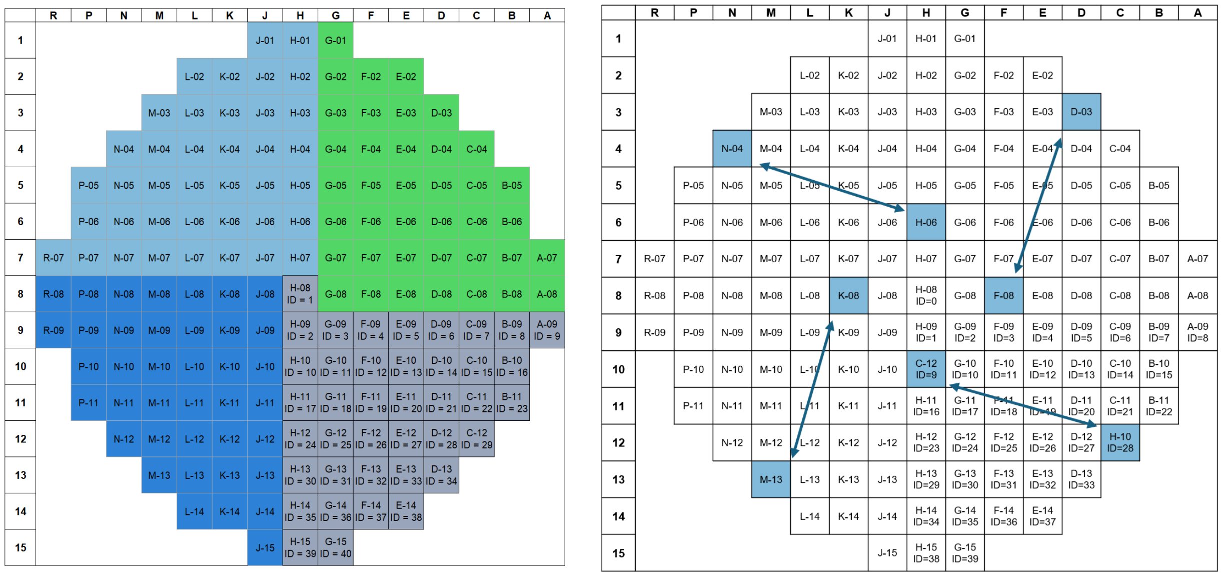
In this study, an initial effort was made to extend the PRLO framework to incorporate multi-cycle optimization capabilities by implementing n-th cycle optimization and demonstrating individual cycle optimization using a given Inventory Management strategy. In this approach, the Inventory Management is treated separately from the optimization process. The RAVEN optimizer perturbs only the defined fuel inventory to identify an optimal fuel reload, while the optimization of Inventory Management itself is planned for future work.

2.4.1 The *n*-th cycle optimization

The approach for n-th cycle optimization involves using the RAVEN optimizer to shuffle the defined FAs into different positions within the reactor core. In this method, the chromosome representation of the fuel loading includes fuel labels that specify the position of each FA. The size of this chromosome depends on the reactor map size and the symmetry applied. Unlike using only fuel types, incorporating labels allows for consideration of FAs from previous cycles. Figure 10 illustrates a label map for a generic AP1000 reactor with 157 FAs, where colors represent quadrants (except for the H-08 position) and IDs in Figure 10 (b) indicate the position of an FA within the quarter-core.

The permuted FAs from the given inventory in the core are represented as a chromosome, with each gene corresponding to an FA label and its specific position within the reactor. These FAs can be either fresh or reused, as defined by the inventory management. The process allows only permutations, meaning no fuel is added or removed during the shuffling of FAs. Given that each FA may have a unique burnup history, a chromosome representing an unrestricted map has a length equal to the number of FAs in the core. This results in a large design space; for a 157-assembly reactor, the design space is 157!.

Since real-world core design always follow a certain level of symmetry (i.e., ¼ or 1/8 core symmetry), maintaining symmetry in the optimization process is a natural choice which can greatly reduce the size of the design space. Some of the symmetries that can be applied include quarter mirror, rotational, and octant symmetries. For example, a quarter rotational symmetrical perturbation divides the reactor into four quadrants with a fixed central position, allowing permutations in one quadrant to be repeated in the others. This approach reduces the chromosome size to 39 genes, making the optimization process more manageable.

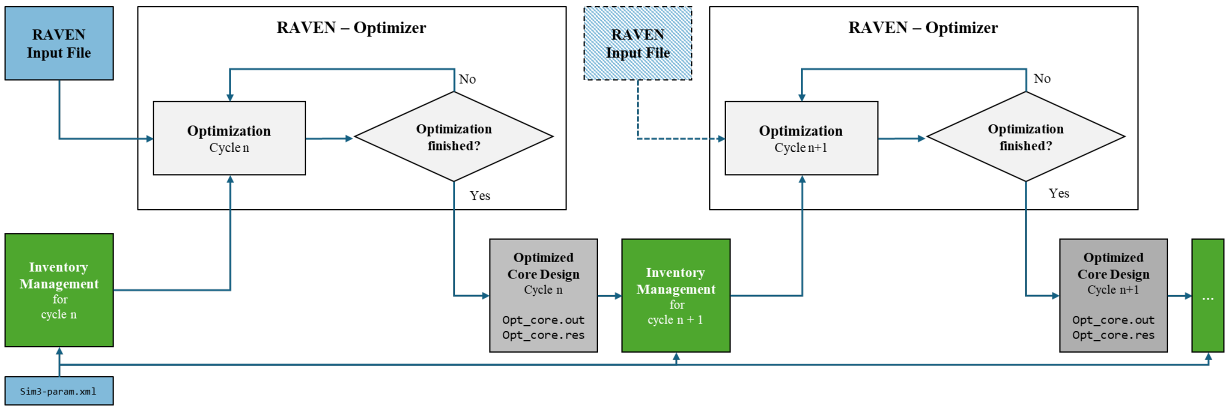


(a) (b)

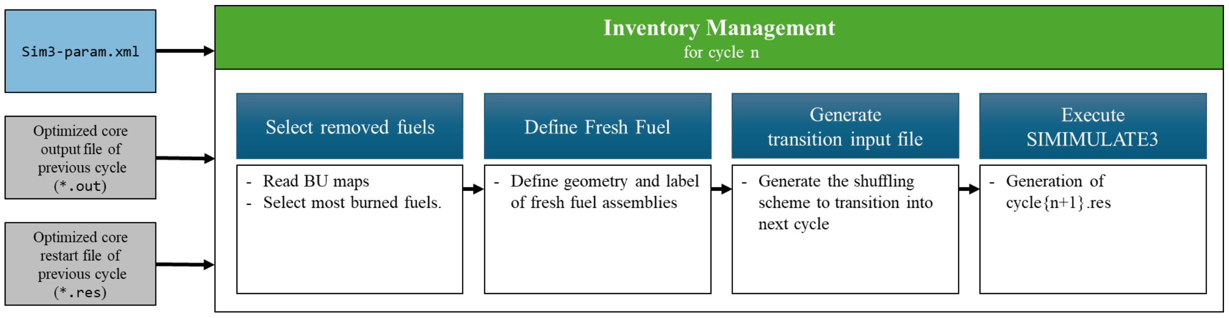
Figure 10. The approach for *n*-th cycle optimization (a) Label map for a generic AP1000 reactor with 157 fuel assemblies, using quarter symmetrical perturbations. (b) Quarter symmetrical perturbation. For instance, C-12 and H-12 were swapped. This resulted in simultaneous permutation of their symmetrical counterparts in the other quadrants.

2.4.2 Multi – cycle optimization with fixed Inventory Management strategy

On top of single cycle optimization capability, additional capability – multi-cycle optimization with externally given Inventory Management – was developed in the PRLO framework. Figure 11 shows multi–cycle optimization process in the PRLO framework. Figure 11(a) shows the workflow of *n*-th and *n+1*-th cycle optimization. The *n*-th cycle means a cycle which does not necessarily the initial fuel cycle, and *n+1*-th cycle is the one next to *n*-th cycle. The optimization in RAVEN–Optimizer needs two inputs: RAVEN input file (see sample RAVEN input script in APPENDIX C.3. Sample RAVEN Input Script of Multi-Cycle Optimization with Given Inventory Management) and a series of steps called "Inventory Management." Figure 11(b) illustrates steps for the Inventory Management for cycle *n*. To complete the Inventory Management steps, reactor specifications such as core map size, reactor inlet temperature, and pressure must be provided. Additionally, to maintain the fuel depletion histories, the results from the optimized core should be used to define the inventory for the subsequent cycle. Given these inputs, Python scripts read the burnup maps at the EOC from the optimized previous cycle output and selects the most burned FAs. Users can specify the quantity of FAs to be discarded. The geometry, labels, and enrichment of the fresh FAs are then defined, and these new fresh FAs are placed in the positions of the discarded ones. Next, the SIMULATE3 input file for cycle n+1 is generated, which includes the fuel inventory information that will be perturbed in the RAVEN–Optimizer. Finally, SIMULATE3 is executed using the input file generated in the previous step, creating a restart file for cycle n+1. This restart file contains inventory information, including the history from the previous cycle, enabling the user to proceed with the optimization of the next cycle. It is important to note that different inventory definition strategies can be applied depending on specific needs for each cycle. The developed approach involves single-cycle optimization applied successively across different cycles, which can be challenging without an appropriate constraint violation penalty weight in the fitness function.



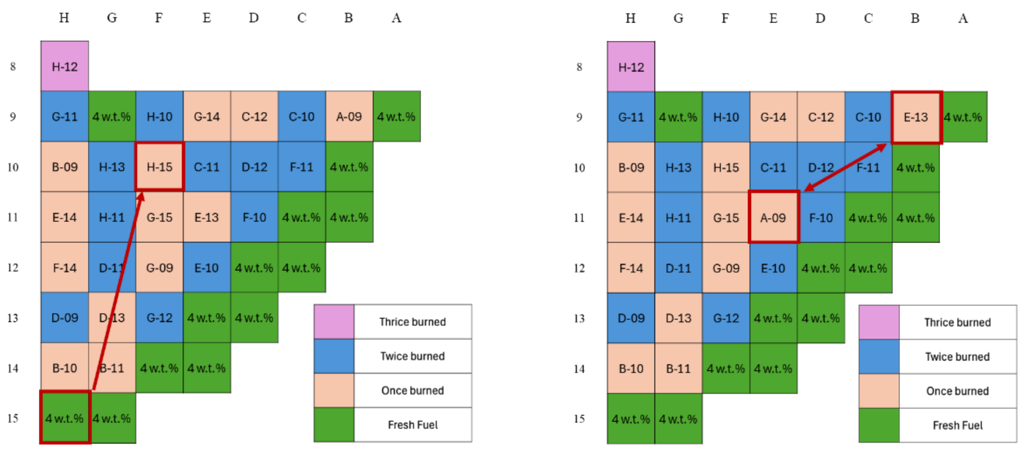
(a)


Sim3-param.xml
Optimized core output file of previous cycle (*.out)
Optimized core restart file of previous cycle (*.res)
Inventory Management
for cycle n
Select removed fuels
Define Fresh Fuel
Generate transition input file
Execute SIMIMULATE3
- Read BU maps
- Select most burned fuels.
- Define geometry and label of fresh fuel assemblies
- Generate the shuffling scheme to transition into next cycle
- Generation of cycle{n+1}.res

(b)

Figure 11. Multi–cycle optimization process in the PRLO framework. (a) Overall workflow of *n*-th and *n+1*-th cycle optimization. The dotted block of RAVEN Input File is optional in case users want to change settings for the optimization after *n*-th cycle. (b) Exemplary strategy for the Inventory Management.

The information needed to transition from one cycle to the next is typically encoded in SC, which gives instructions on the batch definitions, sizes, and the positions of each FA. Figure 12 shows two steps of perturbations of SC. Figure 12(a) illustrates an steps showing what FAs are present and where they came from. Note that Fresh FA in H-15 position, after burning one cycle, is reused in F-10 position. FAs with labels which do not appear in the map imply that they are discarded. Figure 12(b) shows a step of permuting FAs. Given a fixed inventory information at a given cycle, the FAs can be shuffled in the core.



(a) (b)

Figure 12. Two-step approach for perturbations in the shuffling scheme: (a) The Inventory Management – defining fuel inventory (i.e., number of the fuel assembles per batch, amount of fresh fuel used, and reused inventories), and (b) Rearranging inventories through permutation.

Currently, the PRLO framework addresses two distinct steps, as shown in Figure 12: fuel management is handled through external Python scripts, while the shuffling of FAs is managed by the RAVEN Optimizer. The future plan involves incorporating flexible inventory definition and the optimization of fuel inventory directly within RAVEN's optimization framework.

2.4.3 Evolution Operators Development in RAVEN

Fixed-inventory optimization with a SC is a permutation-based combinatorial problem, where FAs are rearranged to occupy different positions. Since each gene in the chromosome represents a distinct FA with its own burnup history, it is essential to use GA operators that will preserve the genes in a chromosome.

The crossover operator in a GA combines information from two parents to generate offspring. In this study, a two-point crossover method is used, where two points in the chromosome are selected, and the genetic material between these points is swapped between the two parents. This ensures that each offspring contains information from both parents. However, this method does not preserve the uniqueness of genes. For example, Offspring 1 in Figure 13 ends up with duplicate Cs and Hs, but lacks D and G. Applying this crossover technique to a permutation-based problem can lead to significant issues, as repeated genes in a chromosome would mean placing the same FA in multiple positions at a given cycle, while neglecting the FA represented by the missing gene.

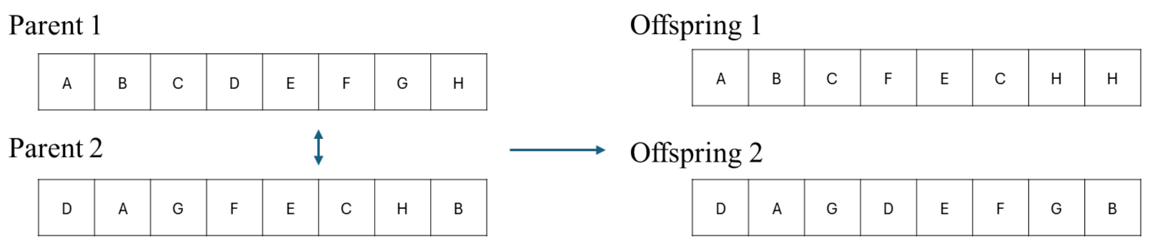


Figure 13. Two-point crossover operation in GA.

To preserve the genes in a chromosome, a two point partially mapped crossover (PMX) operator was added to the available crossover operators in the PRLO framework. The two-point PMX works by creating an intermediate step where the swapped sections of the chromosomes are mapped to each other. The swapped regions are preserved, and the outside genes are mapped back according to the previously done mapping.

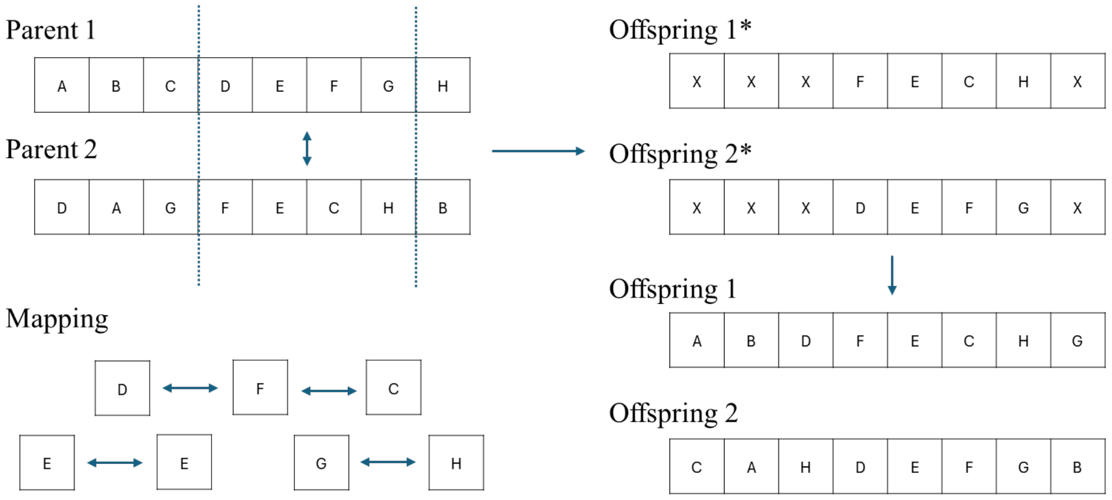


Figure 14. Steps of a performing two point partially mapped crossover.

Mutation operation in GA involves randomly altering an offspring to replicate the concept of imperfect copies in nature. This process enables exploration of design space regions that are not represented by the traits of previous populations. In this study, the swap mutation method is chosen because it maintains the uniqueness of the genes. Figure 15 shows the schematic of the swap mutation in GA.

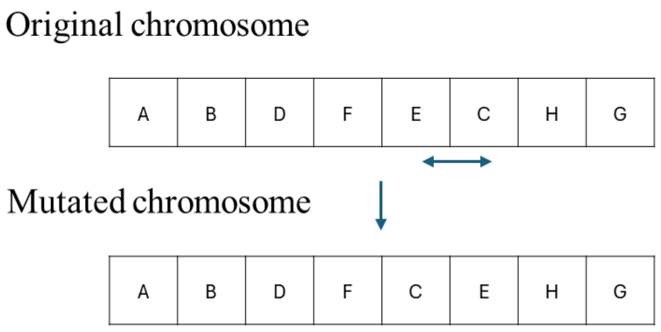


Figure 15. Swap Mutation in GA

2.5 Optimization Acceleration Methods

2.5.1 Adaptive Mutation / Crossover Probabilities

2.5.1.1 Introduction

In GA, mutation and crossover probability play crucial roles in exploring and exploiting the search space. Static mutation probability refers to a fixed rate at which random alterations are introduced into the chromosomes of a population. This probability determines how often parts of a solution are randomly changed, ensuring genetic diversity and preventing premature convergence to suboptimal solutions. For instance, a mutation rate of 0.01 means that 1% of the genes in the chromosome will undergo random changes during each iteration. Static crossover probability, on the other hand, is the fixed rate at which pairs of chromosomes (i.e., parents) exchange segments of their genetic material to produce offspring. This process combines the traits of two parents to create potentially superior solutions. A crossover probability of 0.7 means that 70% of the selected pairs will undergo crossover, blending their genetic information to explore new areas of the solution space.

Static mutation and crossover probabilities, though simple to implement, have significant drawbacks compared to adaptive mutation and crossover probabilities. The main issue with static probabilities is their lack of adaptability; they remain constant throughout the evolutionary process and cannot adjust to the changing needs of the population. This inflexibility can lead to inefficiencies as the algorithm progresses. For instance, a higher mutation rate is beneficial in the early stages for exploring the search space, while a lower rate is preferable later for fine-tuning solutions. Static rates cannot accommodate these shifting requirements. Additionally, static probabilities increase the risk of premature convergence, as a low mutation rate may result in insufficient genetic diversity, causing the population to settle on suboptimal solutions too early. Conversely, a high crossover rate might disrupt high-quality solutions rather than enhancing them. This inability to balance exploration and exploitation effectively hampers optimization.

Adaptive mutation and crossover probabilities, on the other hand, adjust dynamically based on the algorithm’s performance and the state of the population, thereby maintaining diversity and enhancing the quality of solutions. These adaptive probabilities improve convergence speed and the overall efficiency of the optimization process by better balancing exploration and exploitation. In summary, while static probabilities offer simplicity, adaptive probabilities provide the necessary flexibility for more effective and efficient optimization, especially in complex and dynamic search environments. Several adaptive mutation and crossover algorithms are reviewed, and the DHM (Decreasing High Mutation) / ILC (Increasing Low Crossover) approach [16] was applied in this work.

2.5.1.2 DHM / ILC

The DHM/ILC approach controls mutation and crossover probabilities solely based on the number of iterations. DHM/ILC means decreasing mutation probability from 1 to 0 and increasing crossover probability from 0 to 1. The reasoning behind this approach is that fixed high mutation rate can’t converge into global optima. Initial state of GA uses high mutation rate in pursuance of more opportunity that improve the diversity of GA. And as iteration goes, mutation probability gradually gets lower in order to remain stable while crossover probability gets higher in order to shuffle genes of survived chromosomes. Equation (6) ~ (7) show the definition of the DHM/ILC. Figure 16 shows mutation and crossover probability changes over iterations.

$$\text{Mutation Prob.} = 1 - \frac{\text{Current \# of Iteration}}{\text{Total \# of Iterarion}} \tag{6}$$

$$\text{Crossover Prob.} = \frac{\text{Current \# of Iteration}}{\text{Total \# of Iterarion}} \tag{7}$$

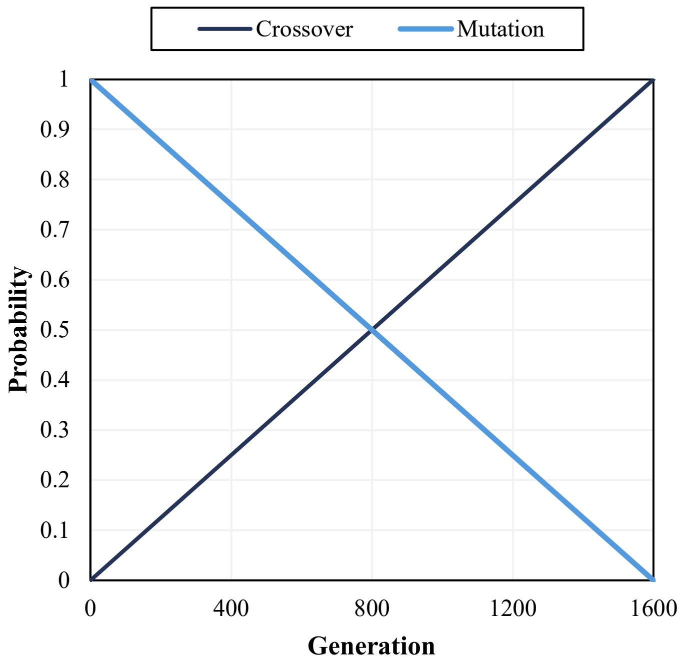


Figure 16. The DHM (Decreasing High Mutation) / ILC (Increasing Low Crossover) approach.

2.5.1.3 Demonstration with ZDT1 Problem

In the realm of multi-objective optimization, the ZDT1 problem [17] is a widely recognized as a benchmark problem. It serves as a standard test case for evaluating the performance of multi-objective optimization algorithms. The ZDT1 problem is specifically designed to assess an algorithm's ability to handle problems with convex Pareto-optimal fronts and to maintain diversity among solutions. The ZDT1 problem is defined in a continuous search space and involves two conflicting objectives, minimizing two objective functions (i.e., $f_1(x)$ and $f_2(x)$). The mathematical formulation of the ZDT1 problem is as follows:

$$f_1(x_1) = x_1 \tag{8}$$

$$f_2(x_1, \dots, x_n) = g \cdot h \tag{9}$$

$$g(x_2, \dots, x_n) = 1 + \frac{9}{n-1} \sum_{i=2}^{n} x_i \tag{10}$$

$$h(f_1, g) = 1 - \sqrt{\frac{f_1}{g}} \tag{11}$$

where

$$0 \leq x_i \leq 1 \; (i = 1, \dots, n)$$

Two-point crossover and random mutation method are used to solve the ZDT1 problem using GA. Figure 17 illustrates the optimization results with static and adaptive mutation & crossover probabilities when the population size equals 10.

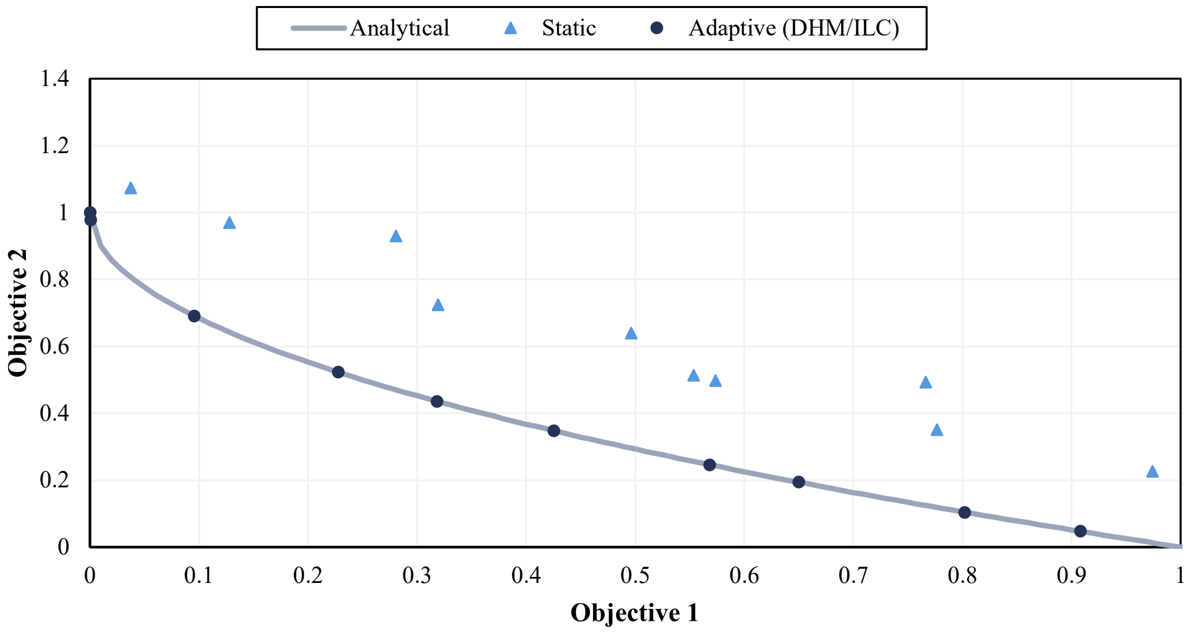


Figure 17. Optimization result of ZDT1 with Population size of 10 and 30 iterations: static mutation/crossover vs. adaptive – DHM (Decreasing High Mutation) / ILC (Increasing Low Crossover – approach.

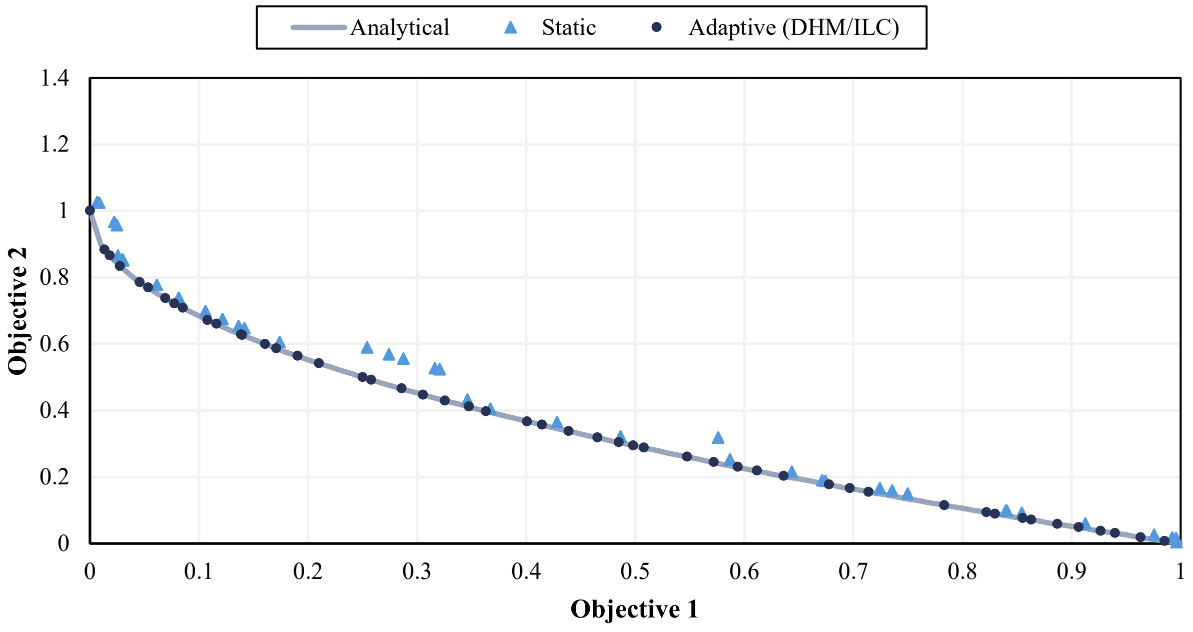


Figure 18. Optimization result of ZDT1 with Population size of 50 and 30 iterations: static mutation/crossover rates vs. adaptive – DHM (Decreasing High Mutation) / ILC (Increasing Low Crossover – approach.

Figure 17 and Figure 18 demonstrate that dynamically adjusting crossover and mutation rates significantly enhances the performance of GA on the ZDT1 problem. The GA using DHM/ILC closely approximates the analytical solution, indicating excellent convergence and diversity maintenance. In contrast, the GA with static mutation/crossover rates failed to converge to the analytical solutions within the given iterations. Table 2 compares the computational time for finding optimal solutions to the ZDT1 problem with a population size of 200, using static versus adaptive mutation/crossover rates. The results show that the adaptive method outperformed the static method by a factor of 60 in terms of computation time, while also achieving better accuracy.

Table 2. Comparison of computational time for finding optimal solutions of ZDT1 problem with 200 population size – static vs. adaptive mutation/crossover rates (2.3GHz 8-core Intel Core i9).

Method	**Number of Iteration**	**Elapsed time (sec)**	**Mean Squared Error (%)**
Static	300	846 sec	0.001%
Adaptive (DHM/ILC)	15	13.89 sec	0.0001%

2.5.2 Active Subspaces for an Efficient GA in high-dimensional Problems

2.5.2.1 Searching for Active Subspaces

The Active Subspaces (AS) method [18] is a dimensionality reduction technique used in parameter space studies. The core concept of the AS method is to identify the crucial directions in the input parameter space that significantly impact the output of a given function. By projecting the high-dimensional input space onto a lower-dimensional subspace, the AS method simplifies the optimization process, enhancing computational efficiency while maintaining substantial accuracy.

The AS method aims to reduce the input dimension $\vec{\mu} = (\mu_1, \mu_2, \ldots, \mu_\mathrm{K})$ of a scalar function $f(\vec{\mu}): \Omega \subset \mathbb{R}^\mathrm{K} \to \mathbb{R}$ by defining a linear transformation $\vec{\mu}_M = \mathbf{A}\vec{\mu}$. This approach necessitates evaluating the gradients of f since A depends on the second moment matrix C of the target function's gradient. The matrix C is defined as follows:

$$\mathbf{C} = \mathbb{E}[\nabla_{\vec{\mu}} f \cdot \nabla_{\vec{\mu}} f^\mathrm{T}] \tag{11}$$

where with the symbol $\mathbb{E}[\cdot]$ we denote the expected value, and $\nabla_{\vec{\mu}} f \equiv \nabla f(\vec{\mu}) \in \mathbb{R}^\mathrm{K}$. The matrix C can be decomposed by the Eigenvalue decomposition as

$$\mathbf{C} = \mathbf{W}\Lambda\mathbf{W}^\mathrm{T} \tag{12}$$

where W represents the orthogonal matrix containing the eigenvectors, and Λ denotes the eigenvalue matrix arranged in descending order. These two matrices can be decomposed as follows:

$$\Lambda = \begin{bmatrix} \Lambda_1 & \square \\ \square & \Lambda_2 \end{bmatrix}, \qquad \mathbf{W} = [\mathbf{W}_1, \mathbf{W}_2], \qquad \mathbf{W}_1 \in \mathbb{R}^{\mathrm{K}\times\mathrm{M}}, \qquad \mathbf{W}_2 \in \mathbb{R}^{(\mathrm{K}-\mathrm{M})\times\mathrm{M}} \tag{13}$$

where M represents the dimension of the active subspace. The active subspace of dimension M is defined as the principal eigenspace corresponding to the eigenvalues before the major spectral gap. We refer to the active variable(s) as $\vec{\mu}_M = \mathbf{W}_1^\mathrm{T}\vec{\mu} \in \mathbb{R}^\mathrm{M}$, and the inactive variable(s) as $\vec{\eta} = \mathbf{W}_2^\mathrm{T}\vec{\mu} \in \mathbb{R}^{\mathrm{K}-\mathrm{M}}$. $\vec{\mu}$ can be expressed using the identified eigenvectors and approximated as shown in Equation 14 and 15.

$$\vec{\mu} = \mathbf{W}_1\mathbf{W}_1^\mathrm{T}\vec{\mu} + \mathbf{W}_2\mathbf{W}_2^\mathrm{T}\vec{\mu} \tag{14}$$

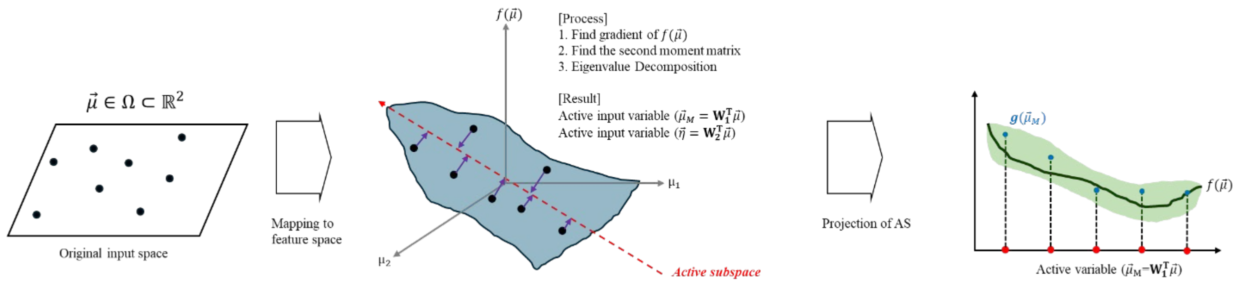
$$\vec{\mu} \cong \mathbf{W}_1\vec{\mu}_M \tag{15}$$

To combat the curse of dimensionality in GA, a study on a supervised learning approach utilizing active subspaces in high-dimensional optimization problems has been conducted. [19] By leveraging the active subspaces property of the objective function, one can select individuals in the reduced parameter space, mutate and mate them, and then map them back to the full parameter space. This process transforms the original optimization problem from Equation 16 to Equation 17:

$$\min_{\mu \in \Omega \subset \mathbb{R}^\mathrm{K}} f(\vec{\mu}) \tag{16}$$

$$\min_{\substack{\vec{\mu}_\mathrm{M} \in \mathrm{P} \subset \mathbb{R}^\mathrm{M} \\ \vec{\mu} \in \Omega}} g(\vec{\mu}_\mathrm{M} = \mathbf{W}_1^\mathrm{T}\vec{\mu}) \tag{17}$$

where Ƥ is a feasible region in the reduced dimensional active subspace, $\mathbb{R}^{M}$. Figure 19 illustrates the progress of the AS method.



$\vec{\mu} \in \Omega \subset \mathbb{R}^2$
Original input space
Mapping to feature space
$f(\vec{\mu})$
[Process]
1. Find gradient of $f(\vec{\mu})$
2. Find the second moment matrix
3. Eigenvalue Decomposition
[Result]
Active input variable ($\vec{\mu}_M = \mathbf{W}_1^T\vec{\mu}$)
Active input variable ($\vec{\eta} = \mathbf{W}_2^T\vec{\mu}$)
μ_1
μ_2
Active subspace
Projection of AS
$g(\vec{\mu}_M)$
$f(\vec{\mu})$
Active variable ($\vec{\mu}_M$=$\mathbf{W}_1^T\vec{\mu}$)

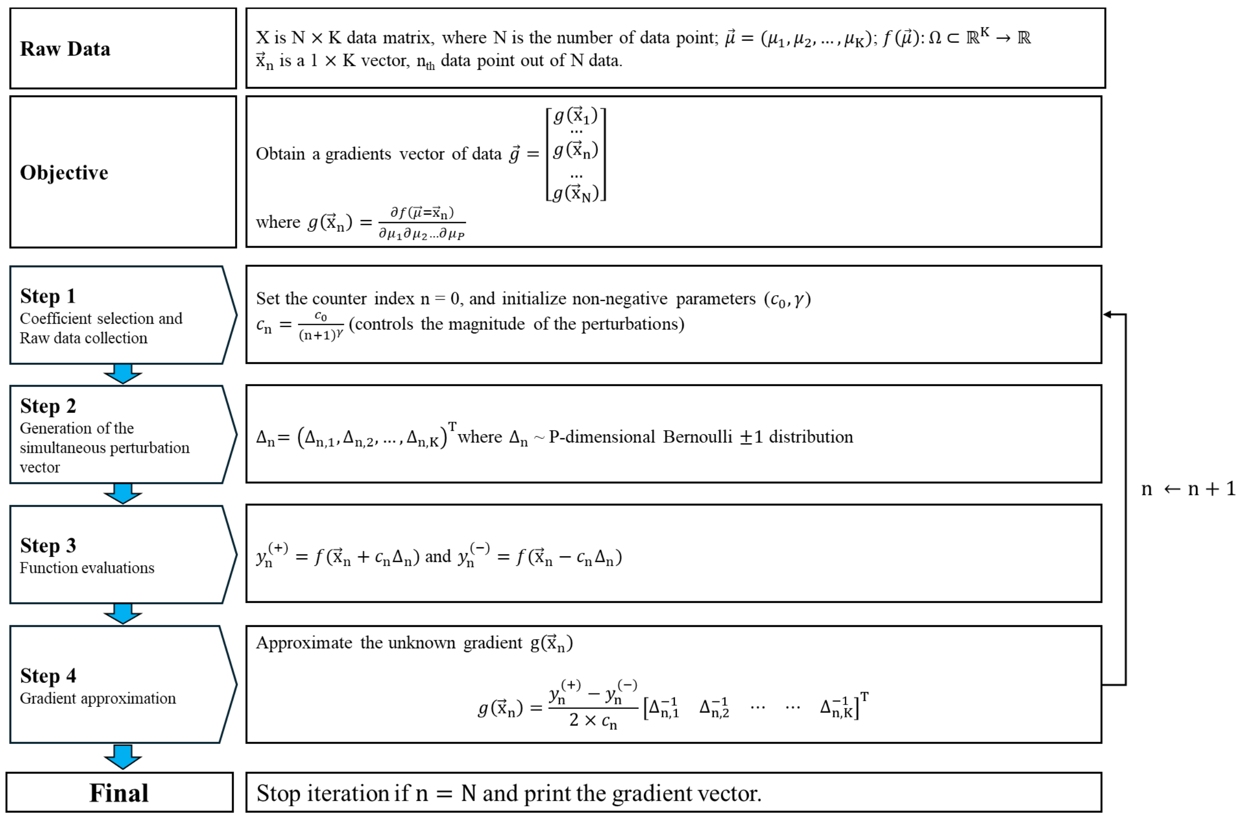
Figure 19. Illustration of the application of active subspace method.

Computing the gradient $\nabla_{\vec{\mu}} f$ can be challenging, particularly for functions that discontinuous, or computationally expensive to evaluate, or when the governing equation is unknown. To address this issue, Simultaneous Perturbation Stochastic Approximation (SPSA) method [20] can be incorporated into the AS method. SPSA method is a powerful gradient approximation technique that estimates the gradient by perturbing all input parameters simultaneously with random perturbations.

2.5.2.2 Simultaneous Perturbation Stochastic Approximation (SPSA)

The SPSA method is a stochastic approach for minimizing differentiable multivariate functions by approximating the function's gradient. It is especially useful for functions where gradient evaluation is either impossible or too resource-intensive. To achieve this, the target function is evaluated only twice using perturbed parameter vectors that are independent of the number of variables: each component of the original parameter vector is simultaneously shifted by a randomly generated value. This differs from the finite difference method (FDM), where only one component of the parameter vector is shifted per evaluation, resulting in computational costs that scale linearly with the number of parameters. [21]

Typically, the SPSA method is used to approximate an input vector such that its gradient equals zero. When a target function is known, one can use either FDM or SPSA to obtain the gradient. However, in many cases, the function of interest is unknown or difficult to determine. To address this, a modified SPSA method has been developed in this study, as illustrated in Figure 20.



Raw Data
X is N × K data matrix, where N is the number of data point; $\vec{\mu} = (\mu_1, \mu_2, \ldots, \mu_K)$; $f(\vec{\mu}): \Omega \subset \mathbb{R}^K \to \mathbb{R}$
$\vec{x}_n$ is a 1 × K vector, n_{th} data point out of N data.
Objective
Obtain a gradients vector of data $\vec{g} = \begin{bmatrix} g(\vec{x}_1) \\ \ldots \\ g(\vec{x}_n) \\ \ldots \\ g(\vec{x}_N) \end{bmatrix}$
where $g(\vec{x}_n) = \frac{\partial f(\vec{\mu}=\vec{x}_n)}{\partial \mu_1 \partial \mu_2 \ldots \partial \mu_P}$
Step 1
Coefficient selection and Raw data collection
Set the counter index n = 0, and initialize non-negative parameters (c_0, γ)
$c_n = \frac{c_0}{(n+1)^\gamma}$ (controls the magnitude of the perturbations)
Step 2
Generation of the simultaneous perturbation vector
$\Delta_n = (\Delta_{n,1}, \Delta_{n,2}, \ldots, \Delta_{n,K})^T$ where Δ_n ~ P-dimensional Bernoulli ±1 distribution
Step 3
Function evaluations
$y_n^{(+)} = f(\vec{x}_n + c_n\Delta_n)$ and $y_n^{(-)} = f(\vec{x}_n - c_n\Delta_n)$
Step 4
Gradient approximation
Approximate the unknown gradient $g(\vec{x}_n)$
$g(\vec{x}_n) = \frac{y_n^{(+)} - y_n^{(-)}}{2 \times c_n}[\Delta_{n,1}^{-1} \quad \Delta_{n,2}^{-1} \quad \cdots \quad \cdots \quad \Delta_{n,K}^{-1}]^T$
$n \leftarrow n + 1$
Final
Stop iteration if n = N and print the gradient vector.

Figure 20. Flowchart of modified SPSA.

Figure 20 shows steps how the modified SPSA method works. This approach starts with obtaining raw data matrix X. X is N × K matrix where N is the number of data point and K is the input space dimension. n is a counter index of raw data set. Once c_0 and γ which are user defined parameters controlling the magnitude of the perturbations are set, one should obtain a simultaneous perturbation vector (Δ_n). The element of Δ_n can be obtained using a distribution that generates only -1 and 1. Once Δ_n is obtained, we need to calculate the values of $y_n^{(+)}$ and $y_n^{(-)}$ to determine approximated gradient.

$$y_n^{(+)} = f(\vec{x}_n + c_n\Delta_n) \tag{18}$$

$$y_n^{(-)} = f(\vec{x}_n - c_n\Delta_n) \tag{19}$$

Equation (20) shows how the gradients are approximated.

$$g(\vec{x}_n) = \frac{y_n^{(+)} - y_n^{(-)}}{2 \times c_n}[\Delta_{n,1}^{-1} \quad \Delta_{n,2}^{-1} \quad \cdots \quad \cdots \quad \Delta_{n,K}^{-1}]^T \tag{20}$$

2.5.2.3 Comparative Study of Gradient Approximation

This section aims to evaluate the effectiveness of the SPSA method in estimating the gradient of a high-dimensional complex function by comparing its results with those obtained analytically or by FDM method. The function of interest and its derivative function are shown in Equation (21) and (22):

$$f(\vec{\mu}) = \sum_{i=1}^{K} \mu_i^2 + \sum_{i=1}^{K} \sin \mu_i^2 \tag{21}$$

$$\frac{\partial f}{\partial \mu_i} = 2\mu_i + 2\mu_i \cos(\mu_i^2) \tag{22}$$

where $\vec{\mu} = (\mu_1, \mu_2, \ldots, \mu_K)$ $f(\vec{\mu}): \Omega \subset \mathbb{R}^K \to \mathbb{R}$.

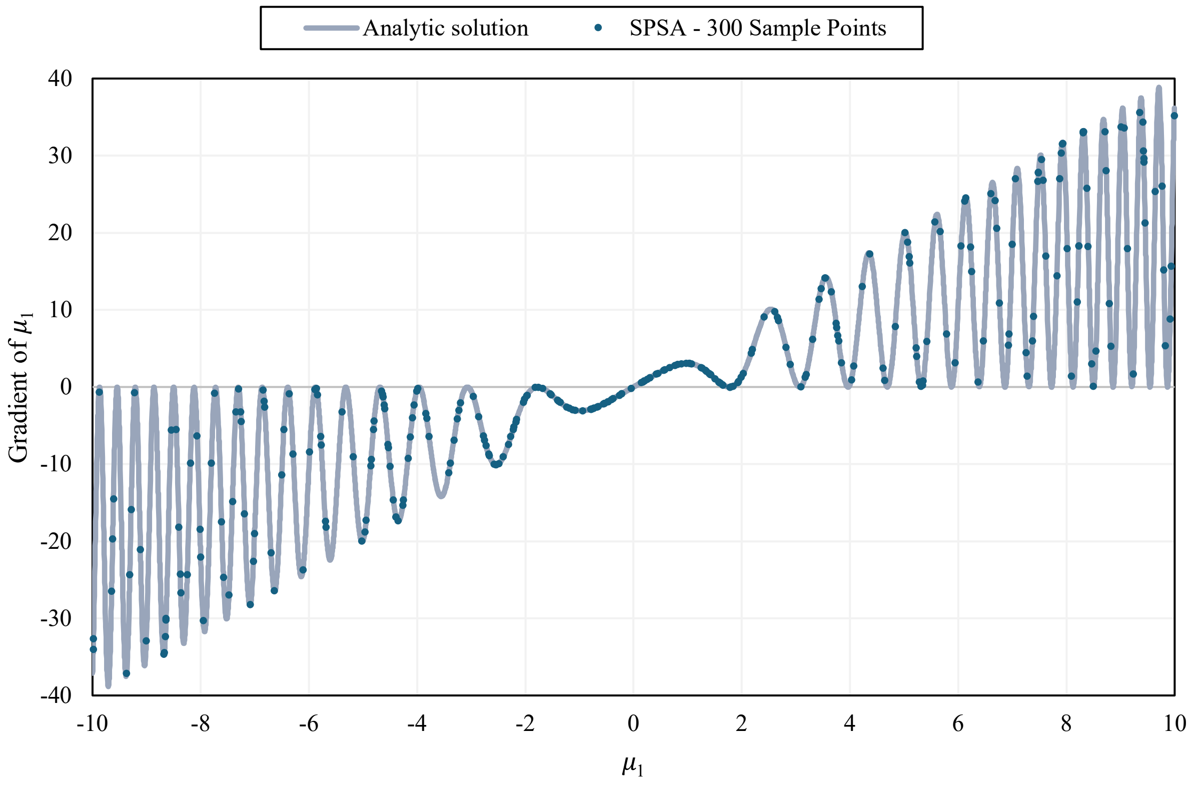


Figure 21. Performance analysis of SPSA method for gradient approximation: a comparison with analytical solutions. K=25 is used for Equation (21) and (22).

Figure 21 shows the result that compares gradients of μ_1 found by SPSA method and gradients found analytically. The SPSA method showed its effectiveness on approximating gradient of each data point.

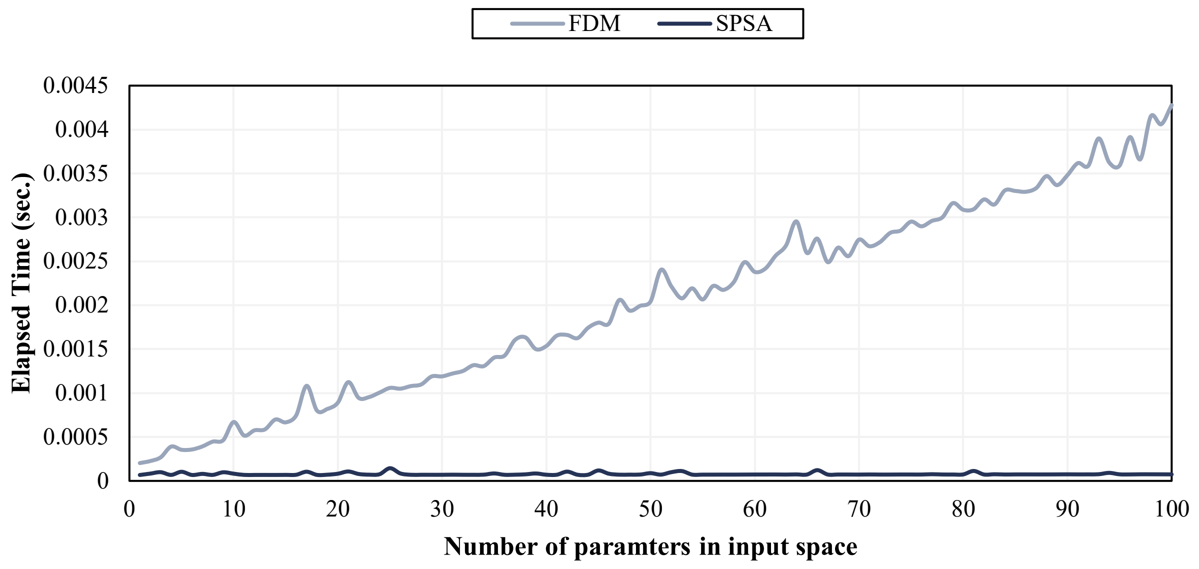


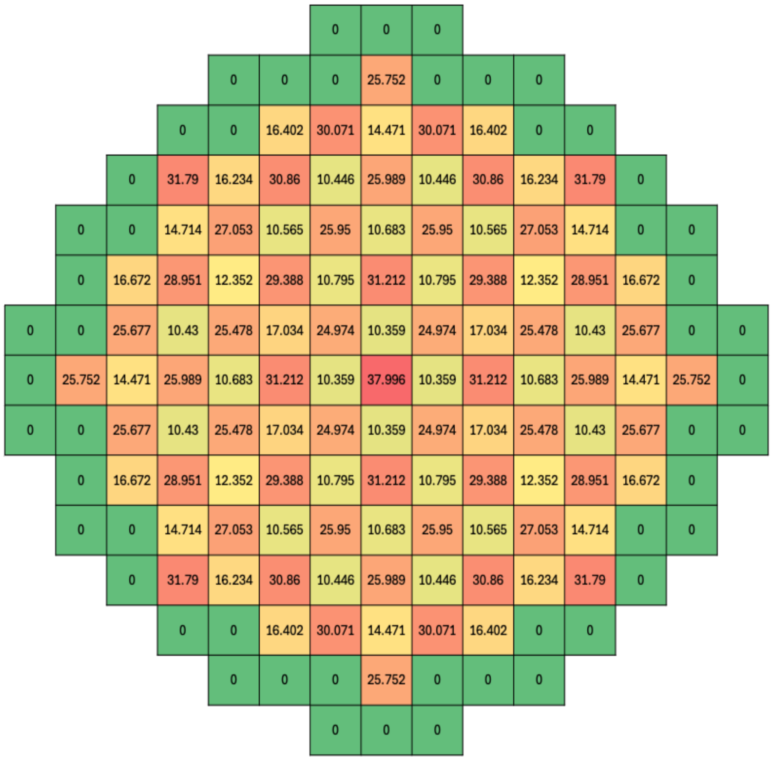
Figure 22. Computational time comparison between FDM and SPSA method for gradient calculation/prediction of 100 data points (2.3GHz 8-core Intel Core i9).

Figure 22 presents a comparison of the elapsed time between the SPSA method and the FDM method. The X-axis represents the dimension of the input space, while the Y-axis indicates the corresponding computational time required to find a gradient for each data point. Both methods exhibit a similar overall pattern, with the SPSA method showing nearly consistent computation times regardless of the increased input parameters. In contrast, the FDM method's computation time increases linearly.

3. DEMONSTRATION OF CORE DESIGN OPTIMIZATION WITHIN PRLO FRAMEWORK

3.1 Single – Cycle Optimization – Cycle 10

A generic AP1000 equilibrium core model, developed in SIMULATE3 based on references [22] [23] [24] [25], was utilized for the optimization demonstration. This model includes a total of 175 FAs. The SC employed to achieve the equilibrium cycle follows a traditional checkerboard pattern, and it reached equilibrium cycle conditions at cycle 9 with $\Delta BU < 0.5$ GWd/MT. Key performance parameters of the generic AP1000 equilibrium cycle can be found in Table 3 and the fuel inventory for the AP1000 model is defined in Table 4. In Table 4, the thrice burned fuel is always positioned in the center of the reactor core, and it consists of the FA with the lowest burnup in the twice burned batch from the previous cycle. All FAs are enriched at 4 wt.% for simplicity and have a 17×17 lattice. No burnable poison is present in the inventory.



						0	0	0						
				0	0	0	25.752	0	0	0				
			0	0	16.402	30.071	14.471	30.071	16.402	0	0			
		0	31.79	16.234	30.86	10.446	25.989	10.446	30.86	16.234	31.79	0		
	0	0	14.714	27.053	10.565	25.95	10.683	25.95	10.565	27.053	14.714	0	0	
	0	16.672	28.951	12.352	29.388	10.795	31.212	10.795	29.388	12.352	28.951	16.672	0	
0	0	25.677	10.43	25.478	17.034	24.974	10.359	24.974	17.034	25.478	10.43	25.677	0	0
0	25.752	14.471	25.989	10.683	31.212	10.359	37.996	10.359	31.212	10.683	25.989	14.471	25.752	0
0	0	25.677	10.43	25.478	17.034	24.974	10.359	24.974	17.034	25.478	10.43	25.677	0	0
	0	16.672	28.951	12.352	29.388	10.795	31.212	10.795	29.388	12.352	28.951	16.672	0	
	0	0	14.714	27.053	10.565	25.95	10.683	25.95	10.565	27.053	14.714	0	0	
		0	31.79	16.234	30.86	10.446	25.989	10.446	30.86	16.234	31.79	0		
			0	0	16.402	30.071	14.471	30.071	16.402	0	0			
				0	0	0	25.752	0	0	0				
						0	0	0						

Figure 23. Burnup [GWd/MT] map of generic AP1000 equilibrium cycle model at the beginning of cycle.

Table 3. Performance parameters of equilibrium reference AP1000 model.

Variable	Value
Cycle Length (EFPD)	353.2
Critical Boron Concentration (ppm)	1457.2
F_Q	1.901
$F_{\Delta H}$	1.552

Table 4. Fuel inventory of equilibrium cycle in reference AP1000 model.

Fuel Batch	Number of Fuel Assemblies
Fresh fuel	52
Once burned	52
Twice burned	52
Thrice burned	1

3.1.1 Problem Statement

For the single-cycle optimization demonstration, the objective was to identify a reactor core design that achieves improved parameters compared to the reference performance parameters of the equilibrium cycle, while utilizing the same inventory. Two cases are developed: Case A and Case B. The goal of the optimized core in Case A is to maximize the cycle length (i.e., Effective Full Power Day [EFPD]). The target constraints values were defined by using the values of the equilibrium model as follows $\mathrm{F_Q} <$ 1.901, $\mathrm{F_{\Delta H}} < 1.552$ and $\mathrm{Boron\ Concentrtation(pcm)} < 1{,}457.2$. The constraint violation weights were defined as shown in Equation 23.

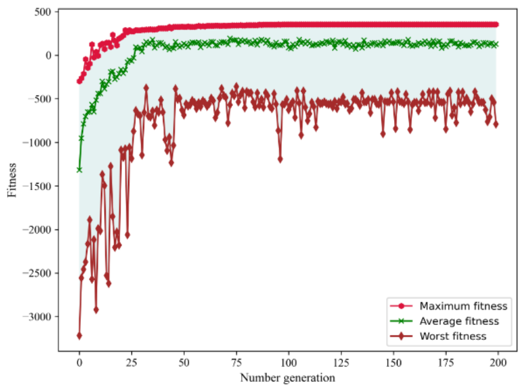
$$\begin{aligned}\mathrm{fitness} = 1.0\ \times \mathrm{EFPD} - 400 \times \max\left(0, \mathrm{F_Q} - 1.901\right) - 400 \times \max(0, \mathrm{F_{\Delta H}} - 1.552) \\ - 1{,}600 \times \max(0, \mathrm{Boron\ Concentration} - 1{,}457.2)\end{aligned} \tag{23}$$

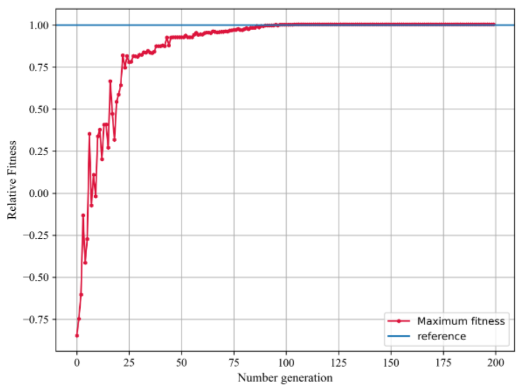
The goal of the optimized core in Case B is to maximize core average burnup. The target constraints values were defined by using the values of the equilibrium model as shown above. The constraint violation weights were defined as shown in Equation 24.

$$\begin{aligned}\mathrm{fitness} = 1.0\ \times \mathrm{core\ average\ burnup} - 40 \times \max\left(0, \mathrm{F_Q} - 1.901\right) \\ - 40 \times \max(0, \mathrm{F_{\Delta H}} - 1.552) \\ - 4{,}000 \times \max(0, \mathrm{Boron\ Concentration} - 1{,}457.2)\end{aligned} \tag{24}$$

3.1.2 Optimization Results and Analysis

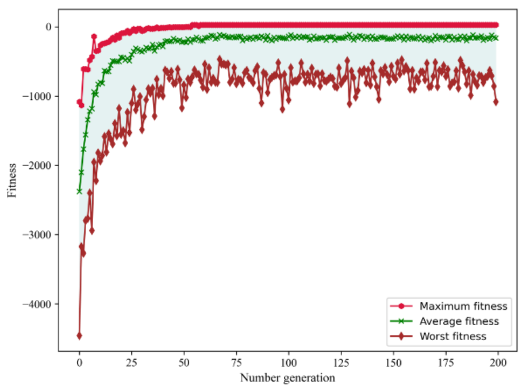
The GA optimization was conducted over 200 generations, with a population size of 100 and a quarter rotational symmetry. Figure 18 and Figure 19 illustrate the evolution of the fitness function during the optimization process and its relative performance. The relative fitness is defined such that the fitness value for the AP1000 reference core is set to 1. The results show that designs with performance comparable to the reference model were achieved after approximately 85 generations for Case A and around 60 generations for Case B. By the end of the optimization, designs with improved parameters were identified for both cases.

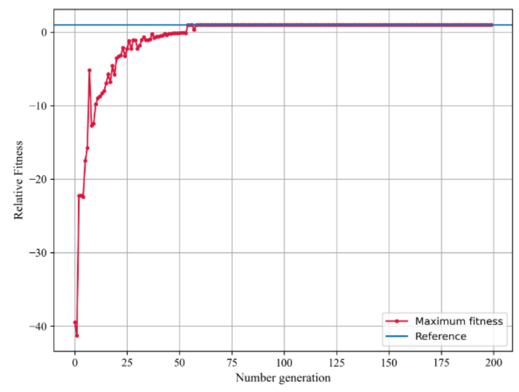




(a) (b)

Figure 18. Fitness behavior of Case A. (a) fitness function vs. generations (b) relative fitness.





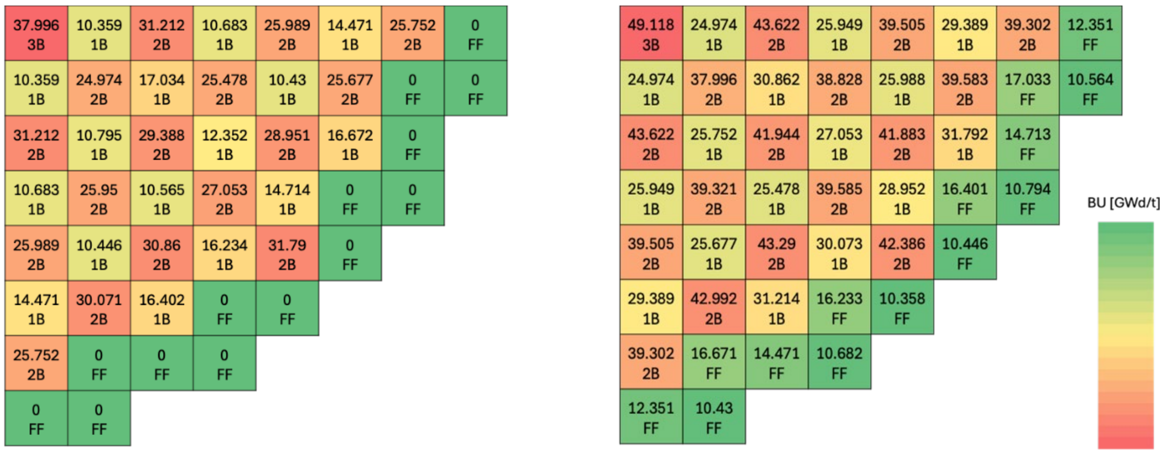
(a) (b)

Figure 19. Fitness behavior of Case B. (a) Fitness function vs. generations (b) Relative fitness.

Table 5. Summary of key variables of single cycle optimization: Reference, Case A and Case B.

Variable	**Reference**	**Case A**	**Case B**
Cycle Length (EFPD)	353.2	353.5	353.5
Boron Concentration (ppm)	1457.20	1456.94	1456.90
F_Q	1.901	1.781	1.775
$F_{\Delta H}$	1.552	1.463	1.458

Figure 24, Figure 25, and Figure 26 show quarter burnup map for three cases: reference, Case A and Case B respectively. It is worth noting that the optimization process led us to designs with a loading that resembles industry practice in both Case A and B, by starting from a randomly generated initial population using GA. The fresh FAs were placed mostly along the periphery of the reactor.



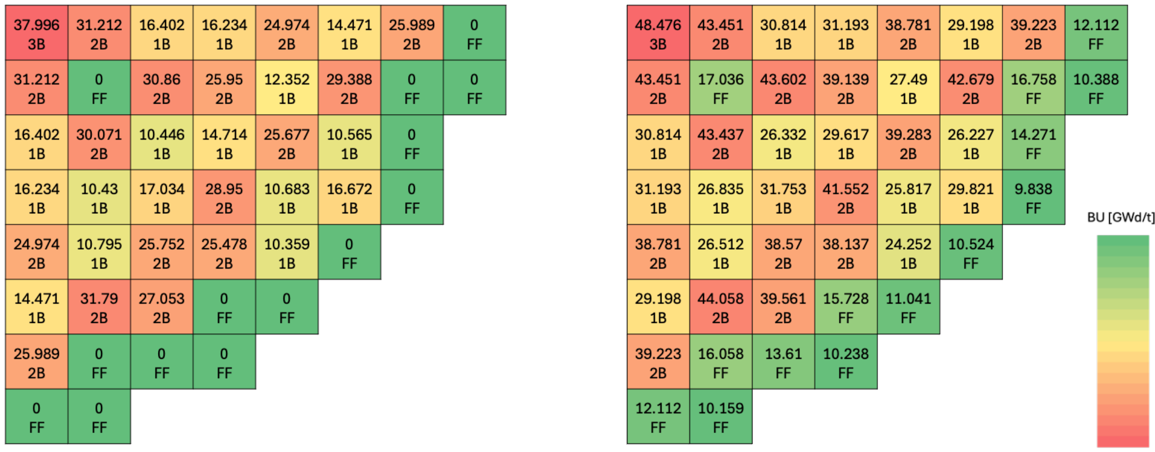
(a)

37.996 3B	10.359 1B	31.212 2B	10.683 1B	25.989 2B	14.471 1B	25.752 2B	0 FF
10.359 1B	24.974 2B	17.034 1B	25.478 2B	10.43 1B	25.677 2B	0 FF	0 FF
31.212 2B	10.795 1B	29.388 2B	12.352 1B	28.951 2B	16.672 1B	0 FF	
10.683 1B	25.95 2B	10.565 1B	27.053 2B	14.714 1B	0 FF	0 FF	
25.989 2B	10.446 1B	30.86 2B	16.234 1B	31.79 2B	0 FF		
14.471 1B	30.071 2B	16.402 1B	0 FF	0 FF			
25.752 2B	0 FF	0 FF	0 FF				
0 FF	0 FF						

(b)

49.118 3B	24.974 1B	43.622 2B	25.949 1B	39.505 2B	29.389 1B	39.302 2B	12.351 FF
24.974 1B	37.996 2B	30.862 1B	38.828 2B	25.988 1B	39.583 2B	17.033 FF	10.564 FF
43.622 2B	25.752 1B	41.944 2B	27.053 1B	41.883 2B	31.792 1B	14.713 FF	
25.949 1B	39.321 2B	25.478 1B	39.585 2B	28.952 1B	16.401 FF	10.794 FF	
39.505 2B	25.677 1B	43.29 2B	30.073 1B	42.386 2B	10.446 FF		
29.389 1B	42.992 2B	31.214 1B	16.233 FF	10.358 FF			
39.302 2B	16.671 FF	14.471 FF	10.682 FF				
12.351 FF	10.43 FF						

Figure 24. Quarter burnup map for equilibrium reference core. (a) The BOC, and (b) The EOC.



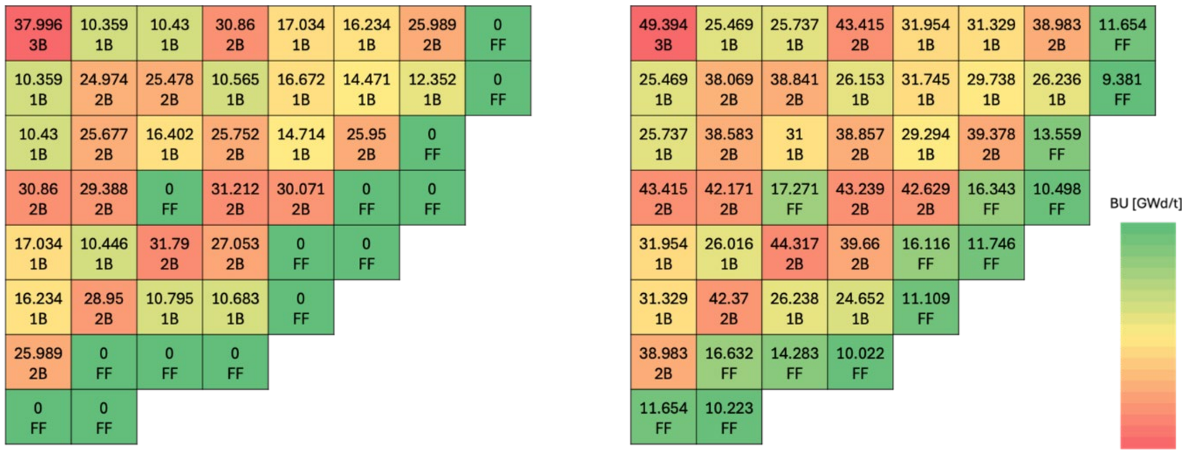
(a)

37.996 3B	31.212 2B	16.402 1B	16.234 1B	24.974 2B	14.471 1B	25.989 2B	0 FF
31.212 2B	0 FF	30.86 2B	25.95 2B	12.352 1B	29.388 2B	0 FF	0 FF
16.402 1B	30.071 2B	10.446 1B	14.714 1B	25.677 2B	10.565 1B	0 FF	
16.234 1B	10.43 1B	17.034 1B	28.95 2B	10.683 1B	16.672 1B	0 FF	
24.974 2B	10.795 1B	25.752 2B	25.478 2B	10.359 1B	0 FF		
14.471 1B	31.79 2B	27.053 2B	0 FF	0 FF			
25.989 2B	0 FF	0 FF	0 FF				
0 FF	0 FF						

(b)

48.476 3B	43.451 2B	30.814 1B	31.193 1B	38.781 2B	29.198 1B	39.223 2B	12.112 FF
43.451 2B	17.036 FF	43.602 2B	39.139 2B	27.49 1B	42.679 2B	16.758 FF	10.388 FF
30.814 1B	43.437 2B	26.332 1B	29.617 1B	39.283 2B	26.227 1B	14.271 FF	
31.193 1B	26.835 1B	31.753 1B	41.552 2B	25.817 1B	29.821 1B	9.838 FF	
38.781 2B	26.512 1B	38.57 2B	38.137 2B	24.252 1B	10.524 FF		
29.198 1B	44.058 2B	39.561 2B	15.728 FF	11.041 FF			
39.223 2B	16.058 FF	13.61 FF	10.238 FF				
12.112 FF	10.159 FF						

Figure 25. Quarter burnup map for optimized with Case A. (a) The BOC, and (b) The EOC.



(a)

37.996 3B	10.359 1B	10.43 1B	30.86 2B	17.034 1B	16.234 1B	25.989 2B	0 FF
10.359 1B	24.974 2B	25.478 2B	10.565 1B	16.672 1B	14.471 1B	12.352 1B	0 FF
10.43 1B	25.677 2B	16.402 1B	25.752 2B	14.714 1B	25.95 2B	0 FF	
30.86 2B	29.388 2B	0 FF	31.212 2B	30.071 2B	0 FF	0 FF	
17.034 1B	10.446 1B	31.79 2B	27.053 2B	0 FF	0 FF		
16.234 1B	28.95 2B	10.795 1B	10.683 1B	0 FF			
25.989 2B	0 FF	0 FF	0 FF				
0 FF	0 FF						

(b)

49.394 3B	25.469 1B	25.737 1B	43.415 2B	31.954 1B	31.329 1B	38.983 2B	11.654 FF
25.469 1B	38.069 2B	38.841 2B	26.153 1B	31.745 1B	29.738 1B	26.236 1B	9.381 FF
25.737 1B	38.583 2B	31 1B	38.857 2B	29.294 1B	39.378 2B	13.559 FF	
43.415 2B	42.171 2B	17.271 FF	43.239 2B	42.629 2B	16.343 FF	10.498 FF	
31.954 1B	26.016 1B	44.317 2B	39.66 2B	16.116 FF	11.746 FF		
31.329 1B	42.37 2B	26.238 1B	24.652 1B	11.109 FF			
38.983 2B	16.632 FF	14.283 FF	10.022 FF				
11.654 FF	10.223 FF						

Figure 26. Quarter burnup map for optimized Case B. (a) The BOC, and (b) The EOC.

Figure 27 and Figure 28 display the behavior of cycle length and boron concentration during the optimization process. In Case A, by around generation 100, core designs are identified that match the reference boron concentration, with cycle lengths beginning to exceed those of the reference. In Case B, the core average burnup consistently surpasses the reference value after approximately 80 generations, indicating that the RAVEN optimizer effectively identified core designs that maximize core average burnup.

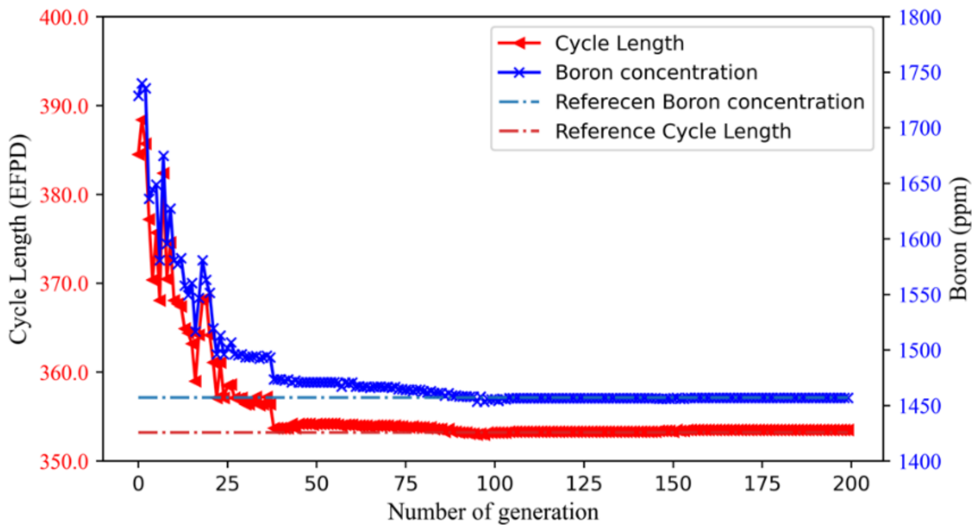


Figure 27. Cycle length and boron concentration behavior during optimization process in Case A.

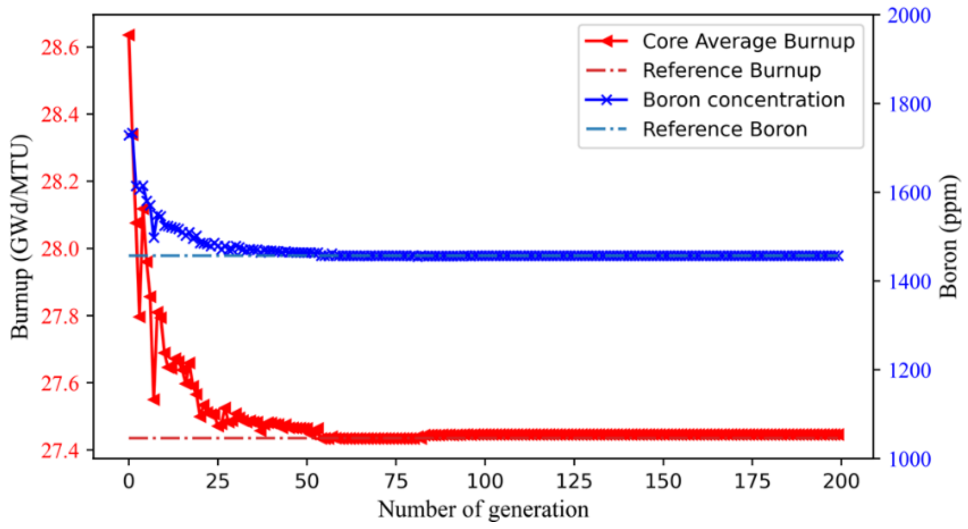
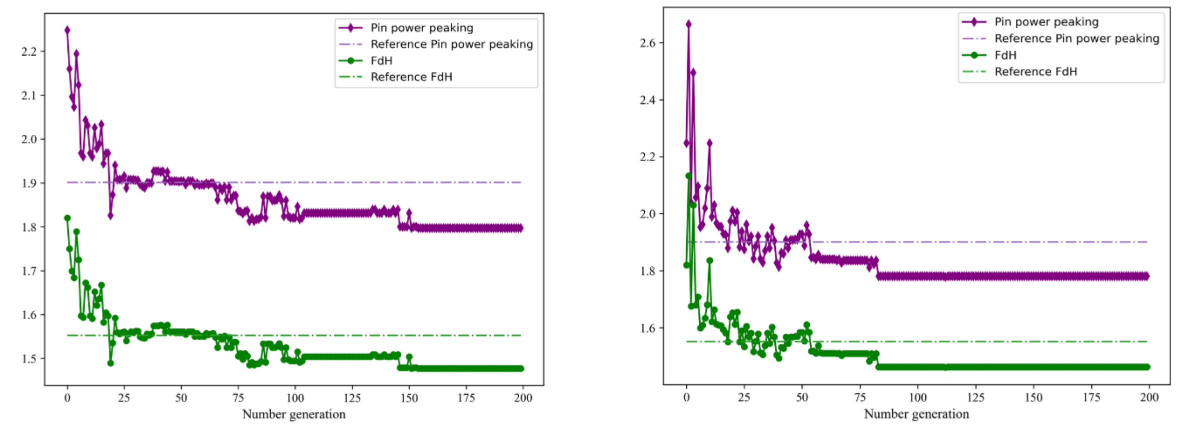


Figure 28. Core average burnup at the EOC and Boron concentration behavior during optimization process in Case B

As shown in Figure 29, the peaking factors improved significantly compared to the reference values through optimization in both Case A and Case B. Starting around generation 50, lower hot channel factors were consistently identified, indicating the effectiveness of the optimization process.



(a) (b)

Figure 29. Hot channel factors. (a) Case A and (b) Case B.

The results from Case A and Case B demonstrated that RAVEN's *n*-th cycle optimization capabilities can generate improved designs. Despite using different approaches in Case A and Case B, the optimization process ultimately led to similar designs in both cases.

3.2 Multi – Cycle Optimization – Cycle 10 & Cycle 11

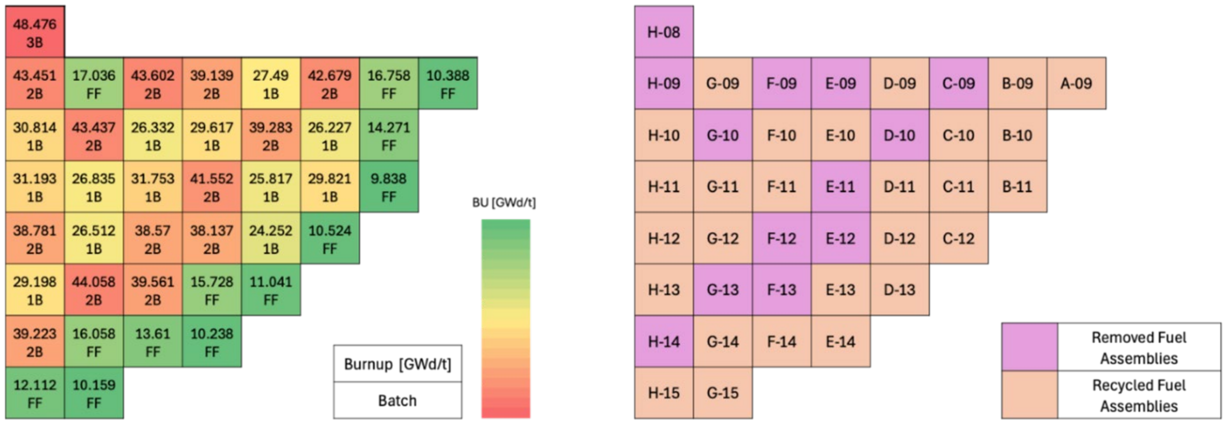
In this demonstration, two consecutive cycles, cycles 10 and 11, were independently optimized using the generic equilibrium AP1000 model as a reference. The optimized core results from Cycle 10 were used to define the inventory for Cycle 11, ensuring that the fuel loading from the equilibrium model was preserved. The objective was to provide a proof of concept for RAVEN's multi-cycle optimization capabilities.

3.2.1 Problem Statement

For the multi-cycle optimization demonstration, the objective was to find a reactor core design that will result in better parameters compared to the reference model by using the same inventory in both Cycle 10 and Cycle 11. The fuel inventory for the AP1000 reference equilibrium is shown in Table 4.

3.2.2 Inventory Management

To transition into subsequent cycles, the Inventory Management is conducted prior to each cycle optimization, as outlined in Figure 11. This process involves selecting FAs to be removed, reused, and introducing a fresh fuel batch. To maintain the same inventory as the reference model, the following approach was employed: FAs are first selected for removal based on their burnup levels—those with higher burnup are removed and do not participate in the next cycle. The number of removed assemblies matches the batch size (52 FAs) to ensure inventory consistency. Figure 30 shows the optimization results after Cycle 10.



(a) (b)

Figure 30. Optimization results after Cycle 10. (a) Optimized burnup map at the EOC. (b) Fuel assemblies to be removed and reused for Cycle 11.

3.2.3 Consecutive Cycles Optimization

The goal of the optimized core for consecutive cycle optimization is set up to maximize cycle length. The target constraints values were defined by using the values of the base equilibrium model as follows $F_Q < 1.901$, $F_{DH} < 1.552$ and $\text{Boron Concentraion} < 1{,}457.2$. A single objective optimization method was used. The constraint violation weights for calculating fitness value were defined as shown in Equation 25.

$$\text{fitness} = 1.0 \times \text{EFPD} - 400 \times \max\left(0, F_Q - 1.901\right) - 400 \times \max(0, F_{\Delta H} - 1.552) \\ - 3{,}200 \times \max(0, \text{Boron Concentraion} - 1{,}457.2) \tag{15}$$

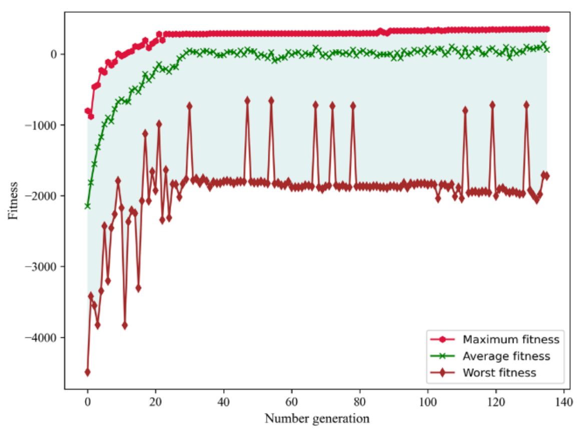
3.2.4 Optimization Results and Analysis

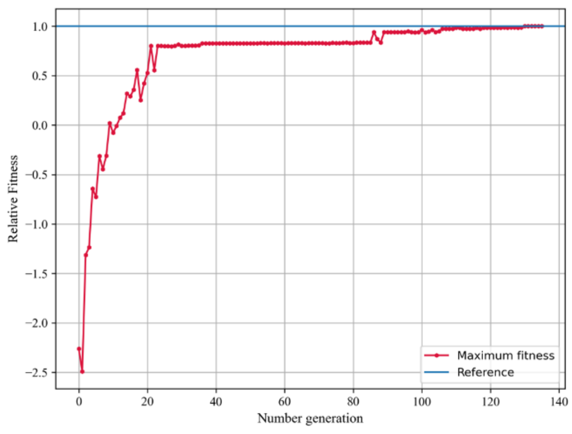
The results from Case A, optimized for Cycle 10, were used as the basis to define the fuel inventory for optimizing Cycle 11. Beginning with randomly generated designs, the RAVEN–Optimizer guided the search through the design space, leading to designs with improved parameters for both Cycle 10 and Cycle 11 compared to the reference equilibrium cycle model as shown in Table 6.

Table 6. Summary of key variables of consecutive cycles optimization: Reference, Cycle 10 and Cycle 11.

Variable	Reference	Cycle 10	Cycle 11
Cycle Length (EFPD)	353.2	353.5	353.3
Boron Concentration (ppm)	1457.2	1456.9	1456.8
F_Q	1.90	1.78	1.83
$F_{\Delta H}$	1.55	1.46	1.49

Figure 31 shows that, after 135 generations, designs with performance similar to the reference model were achieved for Cycle 11 using an inventory defined by the optimized Cycle 10 design in Case A.





(a) (b)

Figure 31. Fitness behavior for Cycle 11. (a) Fitness function vs. generations. (b) Relative fitness

Figure 32 illustrates that the design exploration during the optimization process meets the hot channel factor requirements as early as 20 generations. Additionally, Figure 33 shows that after 100 generations, the exploration identified solutions that satisfy the reference boron concentration requirement. Since meeting the boron concentration requirement was more challenging, it was assigned a higher weight in the fitness function for violation of boron concentration constraint compared to the hot channel factors. (400 vs. 3200 in Equation 25)

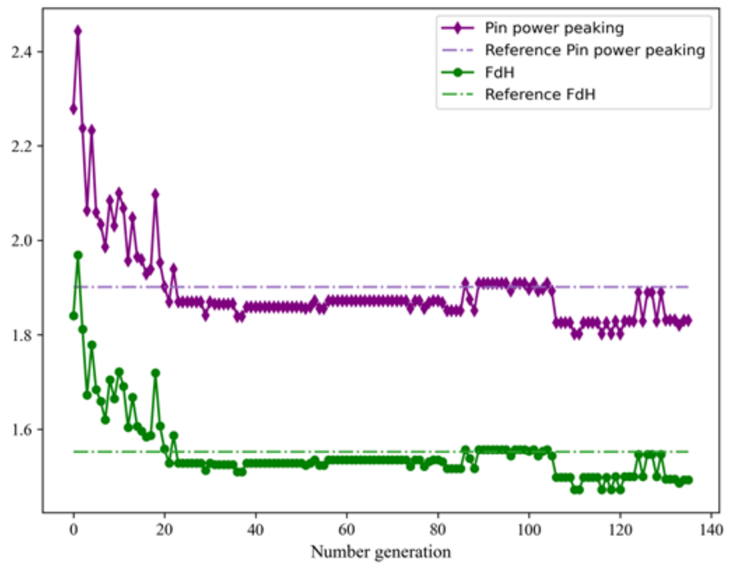


Figure 32. Hot channel factors for Cycle 11.

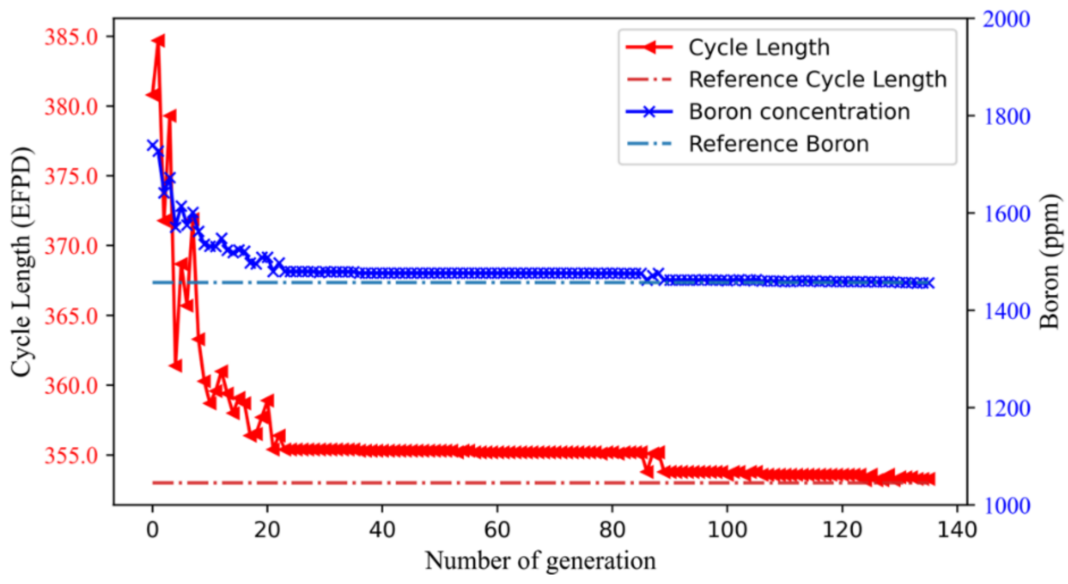
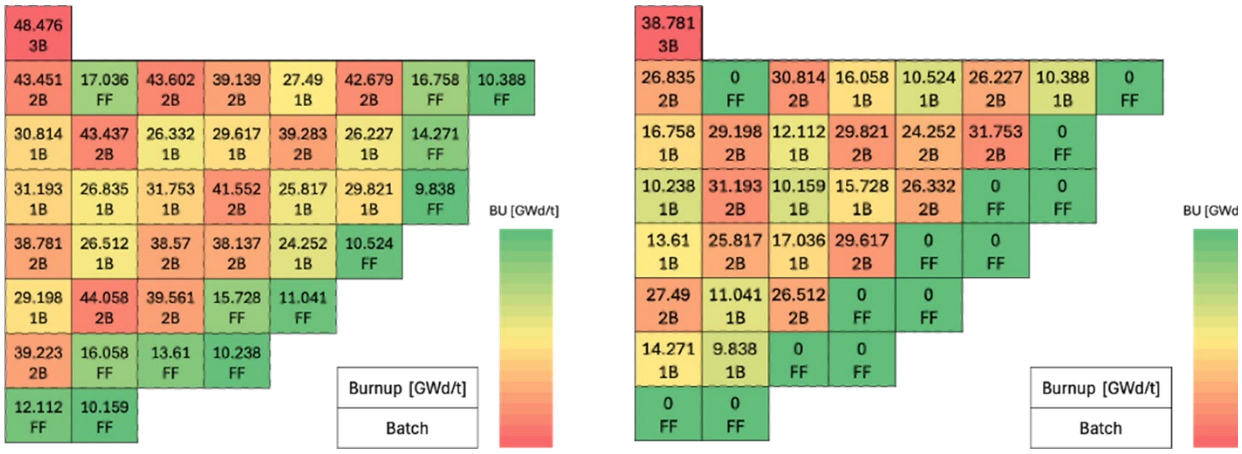


Figure 33. Cycle length and boron concentration behavior during optimization process of Cycle 11.


BU [GWd/t]
Burnup [GWd/t]
Batch
BU [GWd

48.476 3B							
43.451 2B	17.036 FF	43.602 2B	39.139 2B	27.49 1B	42.679 2B	16.758 FF	10.388 FF
30.814 1B	43.437 2B	26.332 1B	29.617 1B	39.283 2B	26.227 1B	14.271 FF	
31.193 1B	26.835 1B	31.753 1B	41.552 2B	25.817 1B	29.821 1B	9.838 FF	
38.781 2B	26.512 1B	38.57 2B	38.137 2B	24.252 1B	10.524 FF		
29.198 1B	44.058 2B	39.561 2B	15.728 FF	11.041 FF			
39.223 2B	16.058 FF	13.61 FF	10.238 FF				
12.112 FF	10.159 FF						

38.781 3B							
26.835 2B	0 FF	30.814 2B	16.058 1B	10.524 1B	26.227 2B	10.388 1B	0 FF
16.758 1B	29.198 2B	12.112 1B	29.821 2B	24.252 2B	31.753 2B	0 FF	
10.238 1B	31.193 2B	10.159 1B	15.728 1B	26.332 2B	0 FF	0 FF	
13.61 1B	25.817 2B	17.036 1B	29.617 2B	0 FF	0 FF		
27.49 2B	11.041 1B	26.512 2B	0 FF	0 FF			
14.271 1B	9.838 1B	0 FF	0 FF				
0 FF	0 FF						

(a) (b)

Figure 34. Quarter burnup map. (a) At the EOC for optimized Cycle 10 core. (b) At the BOC for Cycle 11.

The SC illustrating the movement of FAs from optimized Cycle 10 to optimized Cycle 11 is depicted in Figure 35. Each alphanumeric label (e.g., H-12) indicates the origin of the FA from the previous cycle. The core's label map is provided in Figure 10.

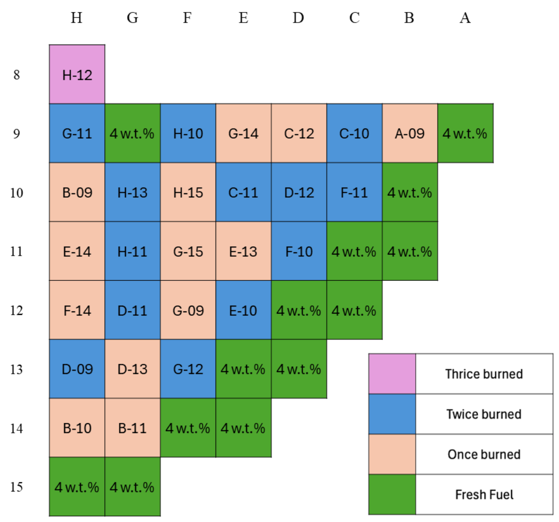


Figure 35. Shuffling scheme of fuel assemblies from optimized Cycle 10, to optimized Cycle 11.

Table 7. Average burnup distribution per batches for reference equilibrium model, optimized Cycle 10, and optimized Cycle 11.

Reactor	BU Step	Fresh	Once Burned	Twice Burned	Thrice burned	Core Average
Reference Model	BOC	0	13.17	27.93	37.996	13.85
	EOC	13.17	27.93	40.79	49.12	27.43
Cycle 10	BOC	0	13.17	27.93	37.996	13.85
	EOC	12.90	28.14	40.88	48.48	27.44
Cycle 11	BOC	0	12.90	28.14	38.781	13.84
	EOC	13.96	27.83	40.98	49.784	27.44

Table 7 shows some differences in the batchwise burnup distributions between the optimized cycles and the reference model. For Cycle 10, the batchwise burnup distribution at the beginning of the cycle (BOC) matches that of the reference model, but by the end of the cycle (EOC), the distribution differs due to the different placement of FAs. At the EOC of optimized Cycle 10, higher burnup was achieved in the twice-burned batch, which is then removed and not used in Cycle 11. Additionally, the fresh batch burnup at the EOC of optimized Cycle 10 is lower than the one in the reference model.

4. SUMMARY AND FUTURE WORKS

In FY24, the research was centered on advancing the PRLO framework that can handle both single and multi-cycle optimization, particularly for PWRs. This study delves into the intricate process of balancing economic considerations—such as batch size, enrichment levels, and fuel utilization—with critical reactor safety constraints. These factors are vital for ensuring that energy requirements and safety standards are consistently met across multiple fuel cycles. The challenge lies in the fact that the use of FAs in one cycle impacts subsequent cycles, creating interdependencies that add complexity to the optimization process.

As part of this research, a new functionality in the PRLO framework has been added by implementing *n*-th cycle optimization and demonstrating the feasibility of optimizing individual cycles within given Inventory Management strategy. In this enhanced approach, the management of the fuel inventory is handled separately from the optimization process. The RAVEN optimizer plays perturbing the predefined fuel inventory to identify the most optimal fuel reload configuration. The separation of Inventory Management from the optimization process allows for a more focused and effective exploration of optimal reload strategies, while the integration of the automated inventory management into the optimization process is planned for future development.

After several demonstrative cases with single and multi-cycle optimization, it has been shown that the PRLO framework has successfully come up with reactor core design optimizing fuel performance across multiple cycles, outperforming the reference design in both single-cycle and two-consecutive-cycle optimization scenarios. Even if this was a slight outperformance, this is promising as better results will be achieved once a more realistic setup is used and enough iterations are performed. These results highlight the potential of the PRLO platform to provide superior core designs compared to traditional heuristic or conventional methods. The enhanced performance achieved through this framework not only optimizes the economic aspects of fuel management but also reinforces safety standards within nuclear power plants.

Looking ahead, future work will focus on automating the inventory definition step as a part of the optimization process. This will involve incorporating the Inventory Management into the core optimization routine, leading to a more comprehensive and cohesive approach to fuel management across multiple cycles. Additionally, the research will expand to include equilibrium cycle optimization and multi-physics analysis to validate the reactor core designs suggested by the PRLO framework. These steps are expected to further refine the optimization process and ensure that the proposed core designs meet the stringent safety and performance requirements necessary for the continued operation of nuclear power plants.

ACKNOWLEDGEMENTS

This manuscript has been authored by Battelle Energy Alliance, LLC under Contract No. DE-AC07-05ID14517 with the U.S. Department of Energy. The U.S. Government retains and the publisher, by accepting the article for publication, acknowledges that the U.S. Government retains a nonexclusive, paid-up, irrevocable, world-wide license to publish or reproduce the published form of this manuscript, or allow others to do so, for U.S. Government purposes.

REFERENCE

[1] Nuclear Energy Institute, "Nuclear Costs in Context," Nuclear Energy Institute, Washington D.C. USA., 2023.

[2] Nuclear Energy Institute, "U.S. Nuclear Operating Plant Basic Information," Nuclear Energy Institute, July 2023. [Online]. Available: https://www.nei.org/resources/statistics/us-nuclear-operating-plant-basic-information. [Accessed 26 July 2024].

[3] International Atomic Energy Agency (IAEA), "Reload Design and Core Management in Operating Nuclear Power Plants. IAEA-TECDOC-1898," International Atomic Energy Agency (IAEA), Vienna, Austria, 2020.

[4] Y.-J. Choi, M. Abdo, G. Palamone, S. Heagy, C. Frepoli, K. Ogujiuba, N. Rollins, G. Deplipei and J. Hou, "Development and Demonstration of a Risk-Informed Approach to the Regulatory Required Fuel Reload Safety Analysis (NL/RPT-22-68628)," Idaho National Laboratory, Idaho Falls, 2022.

[5] RELAP5-3D CODE DEVELOPMENT TEAM, "RELAP5-3DCode Manual Volume I: Code Structure, System Models andSolution Methods, INL-EXT-98-00834, Rev. 4.1," Idaho National Laboratory, Idaho Falls, ID, USA, 2013.

[6] Y.-J. Choi, M. Abdo, D. Mandelli, A. Epiney, J. Valeri, C. Gosdin, C. Frepoli and A. Alfonsi, "Demonstration of the Plant Fuel Reload Process Optimization for an Operating PWR," Idaho National Laboratory, Idaho Falls, ID, USA, 2021.

[7] C. Rabiti, A. Alfonsi, J. Cogliati, D. Mandelli, C. Wang, P. Talbot, M. G. Abdo, D. J. McDowell, R. Yoshiura, D. P. Maljovec, J. Chen, J. Zhou, J. Kim, R. Kinoshita and S. Sen, "RAVEN User Manual INL/EXT-15-34123 Rev.10," Idaho National Laboratory, Idaho Falls, ID, USA, 2023.

[8] Studsvik Scandpower, "SIMULATE-3K Models and Methodology, SSP98/13 Rev. 3," Nyköping, Sweden, 2006.

[9] T. Downar, Y. Xu, V. Seker and D. Carlson, "PARCS: U.S. NRC Core Neutronics Simulator User Manuel," Purdue University, West Lafayette, IN, USA, 2006.

[10] R. L. Williamson, J. D. Hales, S. R. Novascone, G. Pastore, K. A. Gamble, B. Spencer, W. Jiang, S. A. Pitts, A. Casagranda, D. Schwen, A. X. Zabriskie, A. Toptan, R. Gardner, C. Matthews, W. Liu and H. Chen, "BISON: A Flexible Code for Advanced Simulation of the Performance of Multiple Nuclear Fuel Forms," *Nuclear Technology,* vol. 207, no. 7, pp. 954-980, 2020.

[11] Y.-J. Choi, M. Abdo, C. Wang, J. Valeri, C. Frepoli, K. Nguyen and J. Hou, "Development of Plant Reload Optimization Platform Capabilities for Core Design and Fuel Performance Analysis (INL/RPT-22-70382)," Idaho National Laboratory, Idaho Falls, ID, USA, 2022.

[12] J. Kim, M. Abdo, C. Wang and Y.-J. Choi, "Development of Genetic Algorithm Based Multi-Objective Plant Reload Optimization Platform. INL/RPT-23-71667," Idaho National Laboratory, Idaho Falls, ID, USA, 2023.

[13] K. Deb, A. Pratap, S. Agarwal and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation,* vol. 6, no. 20, p. 182–197, 2022.

[14] P. Wang, K. Ye, X. Hao and J. Wang, "Combining multi-objective genetic algorithm and neural network dynamically for the complex optimization problems in physics," *Scientific Reports,* vol. 13, no. 1, p. 880, 2023.

[15] J. Kim, M. Abdo, Y.-J. Choi, J. C. L. Gutierrez, J. Hou, C. Gosdin and J. Valeri, "Pressurized-Water Reactor Core Design Demonstration with Genetic Algorithm Based MultiObjective Plant Fuel Reload Optimization Platform (INL/RPT-23-74498)," Idaho National Laboratory, Idaho Falls, ID, USA, 2023.

[16] A. Hassanat, K. Almohammadi, E. Alkafaween, E. Abunawas, A. Hammouri and V. S. Prasath, "Choosing mutation and crossover ratios for genetic algorithms—a review with a new dynamic approach," *Information,* pp. 10(12), 390, 2019.

[17] E. Zitzler, D. Kalyanmoy and T. Lothar, "Comparison of Multiobjective Evolutionary Algorithms: Empirical Results," *Evolutionary computation,* pp. 173-195, 2000.

[18] P. G. Constantine, Active subspaces: Emerging ideas for dimension reduction in parameter studies, vol. 2, Philadelphia, PA: Society for Industrial and Applied Mathematics, 2015.

[19] N. Demo, M. Tezzele and G. Rozza, "A supervised learning approach involving active subspaces for an efficient genetic algorithm in high-dimensional optimization problems," *SIAM Journal on Scientific Computing,* vol. 43, no. 3, pp. B831-B853, 2021.

[20] J. C. Spall, "Multivariate stochastic approximation using a simultaneous perturbation gradient approximation," *IEEE transactions on automatic control,* pp. 37(3), 332-341, 1992.

[21] C. T. Kelley, "Iterative Methods for Optimization," *Society for Industrial and Applied Mathematics,* p. N/A, 1999.

[22] M. A. Elsawi and A. S. B. Hraiz, "Benchmarking of the WIMS9/PARCS/TRACE code system for neutronic calculations of the Westinghouse AP1000™ reactor," *Nuclear Engineering and Design,* vol. 293, pp. 249-257, 2015.

[23] R. J. Fetterman, "Advanced First Core Design for the Westinghouse AP1000," in *17th International Conference on Nuclear Engineering*, Brussels, Belgium, 2009.

[24] G. L. d. Stefani, J. M. L. Moreira, J. R. Maiorino and P. C. R. Rossi, "Detailed neutronic calculations of the AP1000 reactor core with the Serpent code," *Progress in Nuclear Energy,* vol. 116, pp. 95-107, 2019.

[25] Westinghouse Electric Company LLC. , "Westinghouse AP1000 Design Control Document Rev. 19 (ML11171A500)," U.S. NRC., Washington D.C., USA, 2011.

[26] A. Konak, D. W. Coit and A. E. Smith, "Multi-objective optimization using genetic algorithms: A tutorial," *Reliability Engineering & System Safety,* vol. 91, no. 9, p. 992–1007, 2006.

[27] J. D. Schaffer, "Multiple objective optimization with vector evaluated genetic algorithms," in *International Conference on Genetic Algorithm and Their Applications*, Pittsburgh, PA, USA, 1985.

[28] C. M. Fonseca and P. J. Fleming, "Multiobjective genetic algorithms," in *IEEE Colloquium on Genetic Algorithms for Control Systems Engineering, Digest No. 1993/130*, London, UK., 1993.

[29] P. Hajela and C. Y. Lin, "Genetic search strategies in multicriterion optimal design," *Structural optimization,* vol. 4, pp. 99-107, 1992.

[30] J. Horn, N. Nafpliotis and D. E. Goldberg, "A niched Pareto genetic algorithm for multiobjective optimization," in *IEEE Conference on Evolutionary Computation. IEEE World Congress on Computational Intelligence*, Orlando, FL., 1994.

[31] T. Murata and H. Ishibuchi, "MOGA: multi-objective genetic algorithms," in *IEEE international conference on evolutionary computation*, Perth, Australia, 1995.

[32] D. W. Corne, J. Knowles and M. Oates, "The Pareto envelope-based selection algorithm for multiobjective optimization," in *6th International Conference on Parallel Problem Solving*, Paris, France, 2000.

[33] J. Knowles and D. Corne, "The Pareto archived evolution strategy: a new baseline algorithm for Pareto multiobjective optimisation," in *Congress on Evolutionary Computation*, Washington D.C., USA, 1999 .

[34] N. Srinivas and K. Deb, "Multiobjective optimization using nondominated sorting in genetic algorithms," *Evolutionary Computation,* vol. 2, no. 3, pp. 221-248, 1984.

[35] K. Deb, S. Agrawal, A. Pratap and T. Meyarivan, "A fast elitist nondominated sorting genetic algorithm for multi-objective optimization: NSGA-II," in *6th International Conference on Parallel Problem Solving from Nature*, Paris, France, 2000.

[36] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach," *IEEE transactions on Evolutionary Computation,* vol. 3, no. 4, pp. 257-271, 1999.

[37] E. Zitzler, M. Laumanns and L. Thiele, "SPEA2: improving the strength Pareto evolutionary algorithm. TIK report 103," Swiss Federal Institute Techonology, , Zurich, Switzerland, 2001.

[38] H. Lu and G. Yen, "Rank-density-based multiobjective genetic algorithm and benchmark test function study," *IEEE Trans Evol. Comput.,* vol. 7, no. 4, p. 325–343, 2003.

[39] G. Yen and H. Lu, "Dynamic multiobjective evolutionary algorithm: adaptive cell-based rank and density estimation," *IEEE Trans Evol. Comput.,* vol. 7, no. 3, p. 253–274, 2003.

Page intentionally left blank

APPENDIX A – ESTIMATED FUEL COST OF THE U.S. NUCLEAR OPERATING PLANTS

Table 8. Estimated Fuel Cost of the U.S. Nuclear Operating Plants in 2021.

REACTOR NAME	STATE	REACTOR TYPE	CAPACITY FACTOR (%)	GENERATION (MWH)*	Total Fuel Cost (M$)**
Arkansas Nuclear One 1	AR	PWR	90	6,570,588	$39.42
Arkansas Nuclear One 2	AR	PWR	81	6,985,096	$41.91
Beaver Valley 1	PA	PWR	92.6	7,356,891	$44.14
Beaver Valley 2	PA	PWR	89	7,024,505	$42.14
Braidwood 1	IL	PWR	95.4	9,887,584	$59.32
Braidwood 2	IL	PWR	94.8	9,586,763	$57.51
Browns Ferry 1	AL	BWR	98.9	10,881,278	$63.29
Browns Ferry 2	AL	BWR	83.5	9,210,093	$53.57
Browns Ferry 3	AL	BWR	99.3	10,962,181	$63.76
Brunswick 1	NC	BWR	97.4	8,006,921	$46.57
Brunswick 2	NC	BWR	91.4	7,461,939	$43.40
Byron 1	IL	PWR	95.9	9,776,596	$58.65
Byron 2	IL	PWR	102.4	10,193,056	$61.15
Callaway	MO	PWR	39.3	4,292,433	$25.75
Calvert Cliffs 1	MD	PWR	103.4	7,895,813	$47.37
Calvert Cliffs 2	MD	PWR	94.9	7,097,820	$42.58
Catawba 1	SC	PWR	94.2	9,571,297	$57.42
Catawba 2	SC	PWR	89.5	9,014,422	$54.08
Clinton	IL	BWR	89.5	8,348,706	$48.56
Columbia Generating Station 2	WA	BWR	84.4	8,511,288	$49.50
Comanche Peak 1	TX	PWR	99.7	10,528,185	$63.16
Comanche Peak 2	TX	PWR	84.3	8,828,068	$52.96
Cooper	NE	BWR	102	6,880,622	$40.02
Davis-Besse	OH	PWR	99.3	7,779,141	$46.67
Diablo Canyon 1	CA	PWR	100.3	9,854,372	$59.12
Diablo Canyon 2	CA	PWR	67.6	6,622,994	$39.73
Donald C. Cook 1	MI	PWR	103.1	9,110,067	$54.65
Donald C. Cook 2	MI	PWR	86.4	8,843,917	$53.06
Dresden 2	IL	BWR	93.9	7,422,615	$43.17
Dresden 3	IL	BWR	96.1	7,534,065	$43.82
Edwin I. Hatch 1	GA	BWR	93.9	7,202,991	$41.89
Edwin I. Hatch 2	GA	BWR	90	6,962,077	$40.49
Fermi 2	MI	BWR	93.7	9,369,536	$54.49
Ginna	NY	PWR	92.8	4,727,764	$28.36
Grand Gulf 1	MS	BWR	95.9	11,772,058	$68.47
H.B. Robinson 2	SC	PWR	96.7	6,426,473	$38.55
Hope Creek 1	NJ	BWR	88.4	9,080,057	$52.81
Indian Point 31	NY	PWR	95	2,821,401	$16.93
James A. Fitzpatrick	NY	BWR	99.6	7,397,717	$43.03
Joseph M. Farley 1	AL	PWR	92.9	7,114,415	$42.68
Joseph M. Farley 2	AL	PWR	100.2	7,868,522	$47.21

LaSalle 1	IL	BWR	101.9	10,092,537	$58.70
LaSalle 2	IL	BWR	84.7	8,412,208	$48.93
Limerick 1	PA	BWR	102.5	10,050,781	$58.46
Limerick 2	PA	BWR	94.2	9,258,265	$53.85
McGuire 1	NC	PWR	102.1	10,361,236	$62.16
McGuire 2	NC	PWR	91.7	9,300,878	$55.80
Millstone 2	CT	PWR	92.6	6,919,561	$41.51
Millstone 3	CT	PWR	96.3	10,296,948	$61.77
Monticello	MN	BWR	92.9	5,022,858	$29.21
Nine Mile Point 1	NY	BWR	91.3	5,037,579	$29.30
Nine Mile Point 2	NY	BWR	111.4	11,155,916	$64.88
North Anna 1	VA	PWR	83.3	6,919,326	$41.51
North Anna 2	VA	PWR	102.2	8,452,320	$50.71
Oconee 1	SC	PWR	102.2	7,579,868	$45.47
Oconee 2	SC	PWR	94	6,981,796	$41.89
Oconee 3	SC	PWR	101.6	7,644,799	$45.86
Palisades2	MI	PWR	100.6	7,014,799	$42.08
Palo Verde 1	AZ	PWR	100.3	11,515,959	$69.09
Palo Verde 2	AZ	PWR	88	10,123,959	$60.74
Palo Verde 3	AZ	PWR	86.9	9,989,944	$59.93
Peach Bottom 2	PA	BWR	103.3	11,439,087	$66.53
Peach Bottom 3	PA	BWR	96.2	10,829,157	$62.98
Perry 1	OH	BWR	89.3	9,703,868	$56.44
Point Beach 1	WI	PWR	98.4	5,137,279	$30.82
Point Beach 2	WI	PWR	91.8	4,832,911	$28.99
Prairie Island 1	MN	PWR	104.9	4,785,979	$28.71
Prairie Island 2	MN	PWR	94.9	4,313,934	$25.88
Quad Cities 1	IL	BWR	96	7,635,531	$44.41
Quad Cities 2	IL	BWR	101.6	8,104,797	$47.14
River Bend 1	LA	BWR	87.8	7,441,875	$43.28
Salem 1	NJ	PWR	101.7	10,205,299	$61.23
Salem 2	NJ	PWR	88.8	8,856,743	$53.13
Seabrook 1	NH	PWR	90.2	9,856,117	$59.13
Sequoyah 1	TN	PWR	87.8	8,865,277	$53.19
Sequoyah 2	TN	PWR	89.1	8,789,535	$52.73
Shearon Harris 1	NC	PWR	94.6	7,986,733	$47.92
South Texas Project 1	TX	PWR	91	10,363,168	$62.17
South Texas Project 2	TX	PWR	93.6	10,491,836	$62.94
St. Lucie 1	FL	PWR	81.2	6,978,217	$41.86
St. Lucie 2	FL	PWR	83	7,175,466	$43.05
Surry 1	VA	PWR	90.5	6,640,059	$39.84
Surry 2	VA	PWR	89.4	6,559,811	$39.35
Susquehanna 1	PA	BWR	97.6	10,664,952	$62.03
Susquehanna 2	PA	BWR	84.9	9,278,594	$53.97
Turkey Point 3	FL	PWR	86.1	6,314,980	$37.89
Turkey Point 4	FL	PWR	100.6	7,589,624	$45.53
V.C. Summer	SC	PWR	77	6,552,773	$39.31
Vogtle 1	GA	PWR	93.7	9,443,128	$56.65

Vogtle 2	GA	PWR	102.5	10,343,780	$62.06
Waterford 3	LA	PWR	96.1	9,806,799	$58.83
Watts Bar 1	TN	PWR	91.2	8,974,715	$53.84
Watts Bar 2	TN	PWR	88.5	8,700,880	$52.20
Wolf Creek 1	KS	PWR	79.9	8,574,732	$51.44

Note:

* Source: U.S. Energy Information Administration
** These values are estimated based on the assumption that the fuel unit cost is $5.999/MWh in 2022 dollars for PWR and $5.816/MWh in 2022 dollars for BWR.

APPENDIX B – COMPARISON SUMMARY AMONG MULTI-EVOLUTIONALRY ALGORITHMS

Table 9. Comparison summary among multi evolutionary algorithms. [26]

Algorithm	Fitness Assignment	Diversity Mechanism	Elitism	External Population	Advantages	Disadvantages
VEGA [27]	Each subpopulation is evaluated with respect to a different objective	No	No	No	First MOGA straightforward implementation	Tend to converge to the extreme of each objective
MOGA [28]	Pareto ranking	Fitness sharing by niching	No	No	Simple extension of single-objective GA	Usually, slow convergence
WBGA [29]	Weighted average of normalized objectives	Niching predefined wights	No	No	Simple extension of single-objective GA	Difficulties in non-convex objective function space
NPGA [30]	No fitness assignment Tournament selection	Niche count as tiebreaker in tournament selection	No	No	Very simple tournament selection	Problems related to niche size parameters
RWGA [31]	Weighted average of normalized objectives	Randomly assigned weights	Yes	Yes	Efficient and easy implement	Difficulties in non-convex objective function space
PESA [32]	No fitness assignment	Cell-based density	Pure elitist	Yes	Easy to implement and computationally efficient	Performance depends on cell sizes
PAES [33]	Pareto dominance is used to replace a parent if offspring dominates	Cell-based density as tiebreaker between offspring and parent	Yes	Yes	Random mutation hill-climbing strategy that is easy to implement and computationally efficient	Prior information needed about objective space, not a population-based approach, and performance depends on cell sizes
NSGA [34]	Ranking based on non-domination sorting	Fitness sharing by niching	No	No	Fast convergence	Problems related to niche size parameter
NSGA-II [35]	Ranking based on non-domination sorting	Crowding distance	Yes	No	Single parameter, well tested, and efficient	Crowding distance works in objective space only

SPEA [36]	Ranking based on the external archive of non-dominated solutions	Clustering to truncate external population	Yes	Yes	Well tested, with no parameter for clustering	Complex clustering algorithm
SPEA-2 [37]	Strength of dominators	Density based on the kth nearest neighbor	Yes	Yes	Improved SPEA and made sure extreme points are preserved	Computationally expensive fitness and density calculation
RDGA [38]	The problem reduced to bi-objective problem with solution rank and density as objectives	Forbidden region cell-based density	Yes	Yes	Dynamic cell update that was robust with respect to the number of objectives	More difficult to implement than others
DMOEA [39]	Cell-based ranking	Adaptive cell-based density	Yes (implicitly)	Yes	Includes efficient techniques to update cell densities and adaptive approaches to set GA parameters	More difficult to implement than others

APPENDIX C – SAMPLE RAVEN INPUT FILES FOR RAVEN – SIMULATE3 INTERFACE

APPENDIX C.1. Sim3-param.xml

```
<Sim3-input-gen>
   <pins> 17 </pins>
   <core_width> 15 </core_width>
   <load_point> 0.000 </load_point>
   <depletion> 20 </depletion>
   <axial_nodes> 25 </axial_nodes>
   <active_height>426.72</active_height>
   <batch> 10 </batch>
   <pressure> 2250.0 </pressure>
   <boron> 900.0 </boron>
   <power> 100.0 </power>
   <flow> 100.0 </flow>
   <inlet_temperature> 550.0 </inlet_temperature>
   <map_size> full </map_size>
   <symmetry> quarter_rotational </symmetry>
   <restart_file> cycle10.res </restart_file>
   <cs_lib> cms.ap1000-eq-all.lib </cs_lib>
   <number_assemblies> 157 </number_assemblies>
   <working-dir> SampleSpecificSim3 </working-dir>
   <FA-list>
    <FA name='FA1'    FAid ='0'    type1 ='H-08' type2 ='H-08' type3 ='H-08' type4 ='H-08'/>
    <FA name='FA2'    FAid ='1'    type1 ='H-09' type2 ='J-08' type3 ='H-07' type4 ='G-08'/>
    <FA name='FA3'    FAid ='2'    type1 ='G-09' type2 ='J-09' type3 ='J-07' type4 ='G-07'/>
    <FA name='FA4'    FAid ='3'    type1 ='F-09' type2 ='J-10' type3 ='K-07' type4 ='G-06'/>
    <FA name='FA5'    FAid ='4'    type1 ='E-09' type2 ='J-11' type3 ='L-07' type4 ='G-05'/>
    <FA name='FA6'    FAid ='5'    type1 ='D-09' type2 ='J-12' type3 ='M-07' type4 ='G-04'/>
    <FA name='FA7'    FAid ='6'    type1 ='C-09' type2 ='J-13' type3 ='N-07' type4 ='G-03'/>
    <FA name='FA8'    FAid ='7'    type1 ='B-09' type2 ='J-14' type3 ='P-07' type4 ='G-02'/>
    <FA name='FA9'    FAid ='8'    type1 ='A-09' type2 ='J-15' type3 ='R-07' type4 ='G-01'/>
    <FA name='FA10'   FAid ='9'    type1 ='H-10' type2 ='K-08' type3 ='H-06' type4 ='F-08'/>
    <FA name='FA11'   FAid ='10'   type1 ='G-10' type2 ='K-09' type3 ='J-06' type4 ='F-07'/>
    <FA name='FA12'   FAid ='11'   type1 ='F-10' type2 ='K-10' type3 ='K-06' type4 ='F-06'/>
    <FA name='FA13'   FAid ='12'   type1 ='E-10' type2 ='K-11' type3 ='L-06' type4 ='F-05'/>
    <FA name='FA14'   FAid ='13'   type1 ='D-10' type2 ='K-12' type3 ='M-06' type4 ='F-04'/>
    <FA name='FA15'   FAid ='14'   type1 ='C-10' type2 ='K-13' type3 ='N-06' type4 ='F-03'/>
    <FA name='FA16'   FAid ='15'   type1 ='B-10' type2 ='K-14' type3 ='P-06' type4 ='F-02'/>
    <FA name='FA17'   FAid ='16'   type1 ='H-11' type2 ='L-08' type3 ='H-05' type4 ='E-08'/>
    <FA name='FA18'   FAid ='17'   type1 ='G-11' type2 ='L-09' type3 ='J-05' type4 ='E-07'/>
    <FA name='FA19'   FAid ='18'   type1 ='F-11' type2 ='L-10' type3 ='K-05' type4 ='E-06'/>
    <FA name='FA20'   FAid ='19'   type1 ='E-11' type2 ='L-11' type3 ='L-05' type4 ='E-05'/>
    <FA name='FA21'   FAid ='20'   type1 ='D-11' type2 ='L-12' type3 ='M-05' type4 ='E-04'/>
    <FA name='FA22'   FAid ='21'   type1 ='C-11' type2 ='L-13' type3 ='N-05' type4 ='E-03'/>
    <FA name='FA23'   FAid ='22'   type1 ='B-11' type2 ='L-14' type3 ='P-05' type4 ='E-02'/>
    <FA name='FA24'   FAid ='23'   type1 ='H-12' type2 ='M-08' type3 ='H-04' type4 ='D-08'/>
    <FA name='FA25'   FAid ='24'   type1 ='G-12' type2 ='M-09' type3 ='J-04' type4 ='D-07'/>
    <FA name='FA26'   FAid ='25'   type1 ='F-12' type2 ='M-10' type3 ='K-04' type4 ='D-06'/>
    <FA name='FA27'   FAid ='26'   type1 ='E-12' type2 ='M-11' type3 ='L-04' type4 ='D-05'/>
    <FA name='FA28'   FAid ='27'   type1 ='D-12' type2 ='M-12' type3 ='M-04' type4 ='D-04'/>
    <FA name='FA29'   FAid ='28'   type1 ='C-12' type2 ='M-13' type3 ='N-04' type4 ='D-03'/>
    <FA name='FA30'   FAid ='29'   type1 ='H-13' type2 ='N-08' type3 ='H-03' type4 ='C-08'/>
    <FA name='FA31'   FAid ='30'   type1 ='G-13' type2 ='N-09' type3 ='J-03' type4 ='C-07'/>
    <FA name='FA32'   FAid ='31'   type1 ='F-13' type2 ='N-10' type3 ='K-03' type4 ='C-06'/>
    <FA name='FA33'   FAid ='32'   type1 ='E-13' type2 ='N-11' type3 ='L-03' type4 ='C-05'/>
    <FA name='FA34'   FAid ='33'   type1 ='D-13' type2 ='N-12' type3 ='M-03' type4 ='C-04'/>
    <FA name='FA35'   FAid ='34'   type1 ='H-14' type2 ='P-08' type3 ='H-02' type4 ='B-08'/>
    <FA name='FA36'   FAid ='35'   type1 ='G-14' type2 ='P-09' type3 ='J-02' type4 ='B-07'/>
    <FA name='FA37'   FAid ='36'   type1 ='F-14' type2 ='P-10' type3 ='K-02' type4 ='B-06'/>
    <FA name='FA38'   FAid ='37'   type1 ='E-14' type2 ='P-11' type3 ='L-02' type4 ='B-05'/>
    <FA name='FA39'   FAid ='38'   type1 ='H-15' type2 ='R-08' type3 ='H-01' type4 ='A-08'/>
    <FA name='FA40'   FAid ='39'   type1 ='G-15' type2 ='R-09' type3 ='J-01' type4 ='A-07'/>
   </FA-list>
</Sim3-input-gen>
```

APPENDIX C.2. Sim3-perturb.xml

```
<core-map>
  <loc id = '1' FAid ='0' />
  <loc id = '2' FAid ='0' />
  <loc id = '3' FAid ='0' />
  <loc id = '4' FAid ='0' />
  <loc id = '5' FAid ='0' />
  <loc id = '6' FAid ='0' />
  <loc id = '7' FAid ='0' />
  <loc id = '8' FAid ='0' />
  <loc id = '9' FAid ='0' />
  <loc id = '10' FAid ='0' />
  <loc id = '11' FAid ='0' />
  <loc id = '12' FAid ='0' />
  <loc id = '13' FAid ='0' />
  <loc id = '14' FAid ='0' />
  <loc id = '15' FAid ='0' />
  <loc id = '16' FAid ='0' />
  <loc id = '17' FAid ='0' />
  <loc id = '18' FAid ='0' />
  <loc id = '19' FAid ='0' />
  <loc id = '20' FAid ='0' />
  <loc id = '21' FAid ='0' />
  <loc id = '22' FAid ='0' />
  <loc id = '23' FAid ='0' />
  <loc id = '24' FAid ='0' />
  <loc id = '25' FAid ='0' />
  <loc id = '26' FAid ='0' />
  <loc id = '27' FAid ='0' />
  <loc id = '28' FAid ='0' />
  <loc id = '29' FAid ='0' />
  <loc id = '30' FAid ='0' />
  <loc id = '31' FAid ='0' />
  <loc id = '32' FAid ='0' />
  <loc id = '33' FAid ='0' />
  <loc id = '34' FAid ='0' />
  <loc id = '35' FAid ='0' />
  <loc id = '36' FAid ='0' />
  <loc id = '37' FAid ='0' />
  <loc id = '38' FAid ='0' />
  <loc id = '39' FAid ='0' />
  <loc id = '40' FAid ='0' />
</core-map>
```

APPENDIX C.3. Sample RAVEN Input Script of Multi-Cycle Optimization with Given Inventory Management

```xml
<?xml version="1.0" ?>
<Simulation verbosity="debug">
  <TestInfo>
    <name>test_loaddingpattern_NSGAII-SIM3</name>
    <author>luquj</author>
    <created>2024-07-25</created>
    <classesTested>Models.Code.CodeInterfaceBase.Simulate</classesTested>
    <description>
       Test to generate SIM3 input files for n-th cycle optimization. Chromosome contains
entire fuel map and does not respect any symmetry.
    </description>
  </TestInfo>

  <RunInfo>
    <WorkingDir>Simulate_RAVEN-NSGAII</WorkingDir>
    <Sequence>sampleGA-Simulate, print</Sequence>
    <batchSize>20</batchSize>
  </RunInfo>

  <Steps>
      <MultiRun name="sampleGA-Simulate" re-seeding="2286" clearRunDir="False">
      <Input class="Files" type="simulatedata">simulatedata_input</Input>
      <Input class="Files" type="perturb">simulateperturb_input</Input>
      <Input class="Files" type="input">input</Input>
      <Model class="Models" type="Code">MySimulate</Model>
      <Optimizer class="Optimizers"  type="GeneticAlgorithm" >GAopt</Optimizer>
      <SolutionExport class="DataObjects" type="PointSet">opt_export</SolutionExport>
      <Output class="DataObjects" type="PointSet">optOut</Output>
      <Output class="OutStreams" type="Print">opt_export</Output>
    </MultiRun>
    <IOStep name="print">
      <Input class="DataObjects" type="PointSet">opt_export</Input>
      <Input class="DataObjects" type="PointSet">optOut</Input>
      <Output class="OutStreams" type="Print">opt_export</Output>
      <Output class="OutStreams" type="Print">optOut</Output>
    </IOStep>
  </Steps>

  <Files>
    <Input name="simulatedata_input" type="simulatedata">sim3-param3.xml</Input>
    <Input name="simulateperturb_input" type="perturb">sim3-perturb3.xml</Input>
    <Input name="input" type="input">input.inp</Input>
  </Files>

  <Models>
    <Code name="MySimulate" subType="Simulate">
      <executable>simulate3 -k</executable>
      <sequence>simulate</sequence>
    </Code>
  </Models>

  <Functions>
    <External file="./constraints.py" name="impConstr1">
      <variables>pin_peaking,MaxFDH,max_boron</variables>
    </External>
    <External file="./constraints.py" name="impConstr2">
      <variables>pin_peaking,MaxFDH,max_boron</variables>
    </External>
    <External file="./constraints.py" name="impConstr3">
      <variables>pin_peaking,MaxFDH,max_boron</variables>
    </External>
  </Functions>

  <Distributions>
    <UniformDiscrete name='FA_dist'>
      <lowerBound>1</lowerBound>
      <upperBound>39</upperBound>
      <strategy>withoutReplacement</strategy>
    </UniformDiscrete>
  </Distributions>

  <Optimizers>
    <GeneticAlgorithm name="GAopt">
      <samplerInit>
       <limit>150</limit>
        <initialSeed>2286</initialSeed>
        <writeSteps>every</writeSteps>
        <type>max</type>
      </samplerInit>
      <GAparams>
        <populationSize>100</populationSize>
        <parentSelection>tournamentSelection</parentSelection>
        <reproduction>
          <crossover type="partiallyMappedCrossover">
            <crossoverProb>0.75</crossoverProb>
          </crossover>
          <mutation type="bitFlipMutator">
            <mutationProb>0.9</mutationProb>
          </mutation>
        </reproduction>
        <fitness type="feasibleFirst"></fitness>
        <survivorSelection>fitnessBased</survivorSelection>
      </GAparams>
      <convergence>
        <AHDp>0.0</AHDp>
      </convergence>
      <constant name="loc1">0</constant>
      <variable name="loc2">
        <distribution>FA_dist</distribution>
      </variable>
      <variable name="loc3">
        <distribution>FA_dist</distribution>
      </variable>
      <variable name="loc4">
        <distribution>FA_dist</distribution>
      </variable>
      <variable name="loc5">
        <distribution>FA_dist</distribution>
      </variable>
      <variable name="loc6">
        <distribution>FA_dist</distribution>
      </variable>
      <variable name="loc7">
        <distribution>FA_dist</distribution>
      </variable>
      <variable name="loc8">
        <distribution>FA_dist</distribution>
      </variable>
      <variable name="loc9">
        <distribution>FA_dist</distribution>
      </variable>
      <variable name="loc10">
        <distribution>FA_dist</distribution>
      </variable>
      <variable name="loc11">
        <distribution>FA_dist</distribution>
      </variable>
      <variable name="loc12">
        <distribution>FA_dist</distribution>
      </variable>
      <variable name="loc13">
        <distribution>FA_dist</distribution>
      </variable>
      <variable name="loc14">
        <distribution>FA_dist</distribution>
      </variable>
      <variable name="loc15">
        <distribution>FA_dist</distribution>
      </variable>
      <variable name="loc16">
    <distribution>FA_dist</distribution>
  </variable>
  <variable name="loc17">
    <distribution>FA_dist</distribution>
  </variable>
  <variable name="loc18">
    <distribution>FA_dist</distribution>
  </variable>
  <variable name="loc19">
    <distribution>FA_dist</distribution>
  </variable>
  <variable name="loc20">
    <distribution>FA_dist</distribution>
  </variable>
  <variable name="loc21">
    <distribution>FA_dist</distribution>
  </variable>
  <variable name="loc22">
    <distribution>FA_dist</distribution>
  </variable>
  <variable name="loc23">
    <distribution>FA_dist</distribution>
  </variable>
  <variable name="loc24">
    <distribution>FA_dist</distribution>
  </variable>
  <variable name="loc25">
    <distribution>FA_dist</distribution>
  </variable>
  <variable name="loc26">
    <distribution>FA_dist</distribution>
  </variable>
  <variable name="loc27">
    <distribution>FA_dist</distribution>
  </variable>
  <variable name="loc28">
    <distribution>FA_dist</distribution>
  </variable>
  <variable name="loc29">
    <distribution>FA_dist</distribution>
  </variable>
  <variable name="loc30">
    <distribution>FA_dist</distribution>
  </variable>
  <variable name="loc31">
    <distribution>FA_dist</distribution>
  </variable>
  <variable name="loc32">
    <distribution>FA_dist</distribution>
  </variable>
  <variable name="loc33">
    <distribution>FA_dist</distribution>
  </variable>
  <variable name="loc34">
    <distribution>FA_dist</distribution>
  </variable>
  <variable name="loc35">
    <distribution>FA_dist</distribution>
  </variable>
  <variable name="loc36">
    <distribution>FA_dist</distribution>
  </variable>
  <variable name="loc37">
    <distribution>FA_dist</distribution>
  </variable>
  <variable name="loc38">
    <distribution>FA_dist</distribution>
  </variable>
  <variable name="loc39">
    <distribution>FA_dist</distribution>
  </variable>
  <variable name="loc40">
```

```
      <distribution>FA_dist</distribution>
    </variable>
    <objective>MaxEFPD</objective>
    <TargetEvaluation class="DataObjects" type="PointSet">optOut</TargetEvaluation>
  <Sampler class="Samplers" type="MonteCarlo">MC_samp</Sampler>
  <ImplicitConstraint class='Functions' type='External'>impConstr1</ImplicitConstraint>
  <ImplicitConstraint class='Functions' type='External'>impConstr2</ImplicitConstraint>
  <ImplicitConstraint class='Functions' type='External'>impConstr3</ImplicitConstraint>
  </GeneticAlgorithm>
</Optimizers>

<Samplers>
  <MonteCarlo name="MC_samp">
    <samplerInit>
      <limit>100</limit>
      <initialSeed>20021984</initialSeed>
    </samplerInit>
    <constant name="loc1">0</constant>
    <variable name="loc2">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc3">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc4">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc5">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc6">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc7">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc8">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc9">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc10">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc11">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc12">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc13">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc14">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc15">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc16">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc17">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc18">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc19">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc20">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc21">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc22">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc23">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc24">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc25">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc26">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc27">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc28">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc29">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc30">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc31">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc32">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc33">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc34">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc35">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc36">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc37">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc38">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc39">
      <distribution>FA_dist</distribution>
    </variable>
    <variable name="loc40">
      <distribution>FA_dist</distribution>
    </variable>
  </MonteCarlo>
</Samplers>

<DataObjects>
  <PointSet name="optOut">
    <Input>
    loc1, loc2, loc3, loc4, loc5, loc6, loc7, loc8, loc9, loc10,
    loc11, loc12, loc13, loc14, loc15, loc16, loc17, loc18, loc19, loc20,
    loc21, loc22, loc23, loc24, loc25, loc26, loc27, loc28, loc29, loc30,
      loc31, loc32, loc33, loc34, loc35, loc36, loc37, loc38, loc39, loc40
      </Input>
      <Output> MaxEFPD, MaxFDH, pin_peaking, max_boron, batchId</Output>
    </PointSet>
    <PointSet name="opt_export">
    <Input>trajID</Input>
    <Output>
        loc1, loc2, loc3, loc4, loc5, loc6, loc7, loc8, loc9, loc10,
        loc11, loc12, loc13, loc14, loc15, loc16, loc17, loc18, loc19, loc20,
        loc21, loc22, loc23, loc24, loc25, loc26, loc27, loc28, loc29, loc30,
        loc31, loc32, loc33, loc34, loc35, loc36, loc37, loc38, loc39, loc40,
        MaxEFPD, pin_peaking, max_boron, MaxFDH, fitness,
        iteration, age, batchId, rank, CD, accepted
    </Output>
    </PointSet>
  </DataObjects>

  <OutStreams>
    <Print name="optOut">
      <type>csv</type>
      <source>optOut</source>
      <what>input, output</what>
    </Print>
    <Print name="opt_export">
      <type>csv</type>
      <source>opt_export</source>
      <clusterLabel>trajID</clusterLabel>
    </Print>
  </OutStreams>
</Simulation>
```