SEIFERT THESIS

BY

LUKE SEIFERT

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Nuclear, Plasma, Radiological Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2022

Urbana, Illinois

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Energy

In modern society, the usefulness and convenience supplied by advancing technologies has allowed quality of life to increase drastically over time. Within these advancing technologies, ranging from air conditioning to food production, there is a common thread of energy requirement. To cool the air, energy is required in order to compress, pressurize, and cool the refrigerant. To produce food, energy is required to pump water, create fertilizer, and operate heavy machinery. Thus, the importance of energy is significant in maintaining and developing quality of life for society.

However, energy is not free, and it takes effort to harvest and convert energy into usable forms. For wind power, this means building large turbines which can be turned by the wind. For solar power, this means building solar panels, which convert photons from the sun into electrical energy. However, for most large-scale electrical generation needs, it is more efficient for the conversion aspect to use boiling water to turn a turbine, which then generates electricity, as can be seen in Figure 1.1. Regardless of how it is generated, electricity can then be transported or stored to be used by people for any variety of purpose.

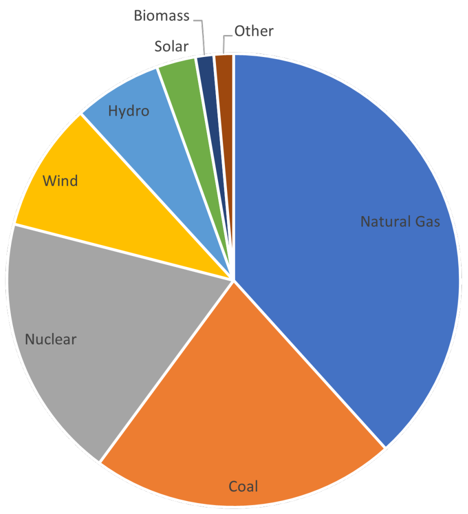

Figure 1.1: Energy generation in the United States in 2021 from the US Energy Information Administration CITE https://www.eia.gov/electricity/monthly/.

Figure 1.1 shows the portion of energy generation. From this figure, it can be seen that the methods which dominate energy contribution are natural gas, coal, and nuclear. Of the various methods to generate electricity, desirable characteristics would be safety, efficiency, consistency, availability, and economics. Nuclear energy is strong in these characteristics, though there is an economic problem due to high up-front costs, which are on the order of $10 billion for a 1GW nuclear power plant [14].

1.2 Nuclear Energy

Part of the reason nuclear energy has high up-front costs is because the nuclear industry is highly safety conscious and has many regulations in place in order to minimize risks. Although this mindset increases costs, the effort has paid dividends, which is apparent when viewing the deaths associated with different energy generation methods, shown in Table 1.1 [10].

Table 1.1: Mortality rates for each energy source in deaths per billion kWh produced (recreated from [10]).

Energy Source	Mortality Rate (deaths per billion kWh)
Coal (Global Average)	100
Biofuel/Biomass	24
Coal (U.S.)	15
Oil	36
Natural Gas	4
Hydro (Global Average)	1.4
Solar (Rooftop)	0.44
Wind	0.15
Nuclear (Global Average)	0.04

Even though the up-front costs are high, nuclear power plants are licensed to operate for 40 years and can get extensions to extend that time by 10-20 years [9]. This means that nuclear plants are a very long term investment. An investment at this time scale is difficult to make for a private industry though, as there is less risk when investing in cheaper plants, such as natural gas.

In order to address the economic issue of nuclear power plants, there are many different approaches which are implemented. One of these is to build smaller, cheaper nuclear plants which are then more affordable and are have less up-front cost. Another approach is from the regulation perspective, and involves either taxing negative externalities more harshly or subsidizing carbon-free energy sources. There is also the approach of developing new generation IV reactor designs [24].

1.3 Molten Salt Reactors

One of the new generation IV reactor designs is the molten salt reactor, or MSR [24]. The MSR potentially offers a lower cost of electricity when compared to coal and pressurized water reactors, which are a large portion of the nuclear fleet [29]. The liquid fueled MSR design is significantly different from the current light water reactor, or LWR, designs which dominate the nuclear fleet currently in use. Light water reactors include boiling water reactors (BWRs) and pressurized water reactors (PWRs), both of which are well understood reactor designs currently implemented. These LWRs use a solid fuel containing uranium-235 with a water coolant, which also serves as a moderator to slow down the neutrons to thermal energies. This fuel generates heat as the fissile uranium-235 fissions. After some lengthy amount of depletion, it can be reshuffled or removed from the reactor to be placed into a cooling pool.

Molten salt reactors, instead of a solid fuel, can use a liquid fuel composed of a molten salt mixed with some fissile isotope, such as uranium-235. Using a liquid fuel allows for online reprocessing, which is a continuous process which can involve adding fresh fuel to the system as well as chemically removing undesirable products which are generated through fission. The system also allows for batchwise reprocessing, which is a process which occurs in discrete steps, such as salt disposal. Typically, most reprocessing involved in an MSR is continuously performed while the reactor is online.

The usefulness of reprocessing, particularly online reprocessing, is that the reactor does not need excess reactivity in order to continue operating for a long period of time. In a typical LWR, burnable poisons and control rods are used to lower reactivity initially so the reactor can burn longer. In a liquid fueled MSR, the parasitic absorbers, such as xenon-135, can be removed from the reactor. Additionally, fresh fuel salt can be added to the reactor in order to maintain stable operation.

This fresh fuel feed can include fissile isotopes, such as uranium-233 or uranium-235, fertile isotopes, such as uranium-238 or thorium-232, or some combination of both. Adding a fissile isotope to the reactor will allow for the reactor to continue fission when the isotope is fissioned by a neutron. For the fertile isotopes, the neutron must first be absorbed and the product decays, leading to a fissile isotope. This process can be seen in Equation (1.1), which is the breeding process used in the Molten Salt Breeder Reactor.

$$^{232}_{90}\mathrm{Th} + ^{1}_{0}\mathrm{n} \longrightarrow ^{233}_{90}\mathrm{Th} \longrightarrow ^{233}_{91}\mathrm{Pa} \longrightarrow ^{233}_{92}\mathrm{U} \tag{1.1}$$

The half life of thorium-233 via beta decay is on the order of tens of minutes, while the half life of protactinium-233 via beta decay is roughly 27 days. Without reprocessing, the protactinium-233 remaining in the core for such a period of time would be more likely to interact with neutrons, impacting the breeding efficiency. The Molten Salt Breeder Reactor continuously chemically extracts the protactinium from the fuel salt, allowing it to decay in a separate tank. While it decays, another continuous chemical process strips uranium which is formed and brings the uranium back into the reactor, continuing the process.

1.4 Reprocessing Methods

For molten salt reactor reprocessing, the two different physical approaches are continuous and batchwise. Additionally, these reprocessing schemes can be online, while the reactor is operating, or offline, when the reactor is shutdown. Of particular importance for MSR behaviour is online reprocessing, which changes the fuel composition while the reactor is operating.

In order to model online reprocessing, there are two different computational methods which can be implemented. These methods are named continuous and batchwise reprocessing methods, the same naming convention as the physical processes which they represent. However, it is fairly common for MSR analysis works which implement a phyiscally continuous reprocessing scheme to simulate this scheme using batchwise mathematical models.

Thus, this work seeks to investigate the continuous and batchwise computational reprocessing methods. This investigation involves effects of depletion step size, validity of batchwise methods approximating continuous methods, and different sub-methods within both batchwise and continuous methods which can be implemented.

Chapter 2

Literature Review

2.1 Molten Salt Reactors

Modeling and simulation of liquid fueled molten salt reactors (MSRs) differs from solid fueled reactors in two main areas. The first is in the fresh fuel feed and fission product removal streams used during operation in MSRs, called online reprocessing. Modeling an MSR without online reprocessing will cause very different results during depletion calculations due to the lack of fresh fuel and buildup of fission products. In order to simulate this reprocessing functionality, software can either use batchwise reprocessing or continuous reprocessing.

The second difference is in the movement of the fuel salt, which causes delayed neutron precursors (DNPs) to move in the core. Because the DNPs have different decay times before the delayed neutrons are born, the effect of the fuel movement on each group varies. However, the overall result is more delayed neutrons in less important regions, such as external piping, which means there is a reduced effective delayed neutron fraction in the core.

2.1.1 Online Reprocessing

The online reprocessing functionality of MSRs is simulated in different ways depending on the particular software used, and depending on the methods employed by that software. The two main ways to approach the online reprocessing problem is by either using batch-wise reprocessing or continuous reprocessing. For these approaches, the physical process may represent either approach, such as continuously removing protactinium from the core, but adding in the uranium it decays into only after some time has elapsed. Typically, however, a purely batchwise modeling approach will not fit the reprocessing design used by a reactor since online reprocessing is carried out continuously. An MSR using offline reprocessing, however, can be accurately represented using purely batchwise reprocessing.

Codes such as SaltProc and older versions of ChemTriton implement batchwise reprocessing by running the simulation, stopping, and then adding and removing materials [39, 8]. The process is then iterated. This method is useful because it is fairly straightforward to implement, easy to customize, and makes mass balancing of the core straightforward. Some issues with the batchwise approach are that using large time steps will make the results more inaccurate, it has increased computational cost with smaller time steps, it has to rerun each time step for pre-reprocessing and post-reprocessing, and it is an approximation of the actual physical process of reprocessing. It is an approximation because, in online reprocessing of a reactor, the fission products are continuously removed while any feed flows are continuously fed into the reactor.

Codes such as Serpent2 and newer versions of ChemTriton implement continuous reprocessing by adding terms to the Bateman equation similar to an extra decay term, as can be seen in Equation (2.5) [4, 23]. The benefits of using continuous reprocessing are that the model is more physically accurate, it doesn't impact computational cost as much as a batchwise process, and it allows for larger time steps to be used. However, the maximum time step size cannot become too large, or the simulation will become less accurate. This is because the neutron spectrum and cross section data does not update to the new material compositions until a new depletion step begins.

Another aspect of online reprocessing that must be considered is mass balancing. For batchwise reprocessing, mass balancing can be solved in several different ways. One method is to have the net mass of the feed rate over some depletion step be equivalent to the mass removed over that same depletion step, which is the current approach used by SaltProc [39]. Alternatively, the volume of the system can be adjusted so that constant density is maintained in order to keep cross section data consistent [34]. Another approach is to move excess fuel salt into a bleed-off tank [34]. For continuous reprocessing, the depletion time steps can be much larger. This means that during the interval between steps, the mass will not be balanced. However, the same batchwise methods can be used with continuous reprocessing at new depletion steps. Implementing these approaches increases computational cost when performed in Serpent2. This is because using a single input script with depletion steps will cause each depeletion step to be run once, but altering the input script by changing volumes or removing mass to a bleed-off tank will cause the time step to be run twice.

2.1.1.1 Batchwise Reprocessing

Batchwise reprocessing starts with some removal rate where $X\%$ of some element is removed from the fuel salt in the system over some time period T. Over a depletion step τ, $\gamma\%$ of the element's mass is removed, where γ corresponds to X according to Equation (2.1). The scaling term γ for the removal efficiency is a linear approximation based on the time step adjustment in Equation (2.1). If the time step used causes the removal fraction to be greater than 1, the amount removed remains at 100% so as to not have negative mass.

An alternative approach is to establish a depletion time step τ such that the efficiency based time value T is a multiple of τ and only have the $X\%$ batchwise removal occur at the steps when τ is a multiple of T, thus removing the need for scaling.

$$\gamma = X\frac{\tau}{T} \tag{2.1}$$

For the Molten Salt Breeder Reactor (MSBR), SaltProc performs batchwise reprocessing every 3 days using a linear approximation of the cycle times given by Robertson et al in the ORNL report [35, 39]. The xenon and krypton removals in SaltProc use a gas separation system equation, while the protactinium uses a modeled liquid-liquid reductive extraction process. These efficiencies differ from the cycle time values, causing slight differences in those terms. The MSBR also has two varying feed inputs of ^{232}Th and ^{233}U.

Examples of the batchwise method used in SaltProc can be seen in Equations (2.2) and (2.3). Both examples show what the resulting mass removal per step, or γ, term should be for different cycle time values. The first shows a cycle time of 20s, which is shorter than the step time of 3 days. This results in a removal rate over 100%, which is set to 100%. The second shows a cycle time of 60 days, longer than the step size. This results in a 5% removal per step. An alternative approach to the second example, which is implemented in the earlier versions of SaltProc, is to perform no removal until 60 days have elapsed and then remove 100% of material.

$$X = 100\%, T = 20s, \tau = 3d \rightarrow \gamma > 100\% \rightarrow 100\% \tag{2.2}$$

$$X = 100\%, T = 60d, \tau = 3d \rightarrow \gamma = 5\% \tag{2.3}$$

2.1.1.2 Continuous Reprocessing

The continuous approach manipulates the Bateman equation by adding a reprocessing term, which contains a reprocessing constant and the concentration of the isotope. This is one of the methods used in Serpent2 continuous reprocessing, an extension developed by Aufiero et al [4]. Equation (2.4) shows the generic form of the Bateman equation, while Equation (2.5) shows the terms added to Equation (2.4), which are the reprocessing constants, λ_r. These terms represent removal of the current isotope as well as feed of the current isotope from other sources.

$$\begin{aligned}\frac{dN_j}{dt}_{base} = &\sum_{i\neq j}\left[\left(\gamma_{i\rightarrow j}\sigma_{f,i}\Phi + \lambda_{i\rightarrow j} + \sigma_{i\rightarrow j}\Phi\right)N_i\right] \\ &- \left(\lambda_j + \sigma_j\Phi\right)N_j\end{aligned} \tag{2.4}$$

$$\frac{dN_j}{dt}_{net} = \frac{dN_j}{dt}_{base} - \lambda_{r,j}N_j + \sum_{mat}\lambda_{r,i\rightarrow j}N_i \tag{2.5}$$

The symbols given in the equations are defined as follows from Equation [25]:

N_j is the atomic density of isotope j.

$\gamma_{i\rightarrow j}$ is the fractional fission product yield of j in the fission of isotope i.

$\sigma_{f,i}$ is the microscopic fission cross section of isotope i.

Φ is the spectrum-averaged scalar flux in the fuel region.

$\lambda_{i\rightarrow j}$ is the decay constant of decay $i \rightarrow j$.

$\sigma_{i\rightarrow j}$ is the microscopic transmution cross section of reaction $i \rightarrow j$.

N_i is the atomic density of isotope i.

λ_j is the decay constant of isotope j.

$\lambda_{r,j}$ is the reprocessing constant for removal of isotope j.

σ_j is the microscopic total transmutation cross section of isotope j.

$\lambda_{r,i\rightarrow j}$ is the reprocessing constant for feed of material $i \rightarrow j$.

Equation (2.5) shows that the reprocessing removal has the same mathematical operation as the decay rate. Unlike decay, the isotopes removed by reprocessing instead are transfered to a different material, which operates as a feed for that material. This can be seen in the summation term, which sums over the different materials and moves them from material i to material j.

Serpent2 has three different settings for continuous reprocessing which alter the effect on the Bateman equations. The zeroth setting does not conserve mass, so it will not be discussed. The first setting, which adds a decay-like term to the Bateman equation, is shown in Equations (2.4) and (2.5). This approach of adding a decay-like term to the Bateman equation has been done with other codes [23, 36]. For this setting, three different approaches are implemented in order to compare their results.

The first approach is labeled the Cycle Time Decay (CTD) approach. It takes half the value of the cycle times and treats them as half life values. A similar method of treating removal periods for reprocessing can be seen in Brovchenko et al [11]. Equation (2.6) shows how this approach works with a cycle time of 20 seconds. The operation cuts the cycle time in half and then calculates the decay constant, or reprocessing constant, using the half life from the cycle time.

$$20s \rightarrow \tau = 10s \rightarrow \lambda_r = \frac{ln(2)}{\tau} \approx 6.93E\text{-}2 \tag{2.6}$$

The next approach is labeled the SaltProc Cycle Rate (SPCR) approach. It takes the same efficiency rates used by SaltProc, which uses 3 day depletion steps for the MSBR, and directly converts those rates to continuous reprocessing rates. Taking the inverse of the cycle time is a common way to generate the removal rate [39, 31]. Equation (2.7) shows how this approach works with a cycle time of 20s. This operation uses the SaltProc value, which in this case becomes 100% removal over 3 days, converts it to be per second, and then converts that to the rate form usable in Serpent2.

$$20s \rightarrow \frac{1}{3d} \rightarrow X = \frac{4E\text{-}6}{1s} \rightarrow \lambda_r = ln\left(\frac{1}{1-X}\right) \approx 4E\text{-}6 \tag{2.7}$$

This natural logarithm in this approach comes from the differential equation solution based on the removal rate provided. The method used to derive this equation can be seen in Equations (3.16), (3.17), and (3.20). These equations assume that the X term, or the fractional removal rate, is in units of per second.

$$\frac{dN}{dt} = -\lambda_r N \tag{2.8}$$

$$N(t) = (1-X)N_{cur} = N_{cur}e^{-\lambda_r} \tag{2.9}$$

$$-ln(1-X) = \lambda_r = ln\left(\frac{1}{1-X}\right) \tag{2.10}$$

The final approach is labeled the Cycle Rate (CR) approach. It takes the cycle times and uses a linear approximation to generate the efficiency rate directly. Equation (2.11) shows how this approach works using a cycle time of 20 seconds. This operation takes the cycle time, determines the efficiency rate for 100% removal over 20 seconds, and then uses that value to calculate the Serpent2 reprocessing constant.

$$20s \rightarrow X = \frac{1}{20s} \rightarrow \lambda_r = ln\left(\frac{1}{1-X}\right) \approx 5.12E\text{-}2 \tag{2.11}$$

The second continuous reprocessing setting used in Serpent2 removes a constant value from each isotopic Bateman equation for an element. This approach can be seen in Equation 2.12, where the C term represents the constant value being removed by continuous reprocessing.

$$\frac{dN_j}{dt} = \sum_{i\neq j}\left[\left(\gamma_{i\rightarrow j}\sigma_{f,i}\Phi + \lambda_{i\rightarrow j} + \sigma_{i\rightarrow j}\Phi\right)N_i\right] - \left(\lambda_j + \sigma_j\Phi\right)N_j - C \tag{2.12}$$

This approach has a flaw which can be directly demonstrated through an example if the reprocessing constants are defined for an element and not for each isotope. Imagine an element exists with two isotopes A and B. There is a removal rate of 10% per second defined for the element. A batchwise process could remove 10% of each isotopes relative abundance after a second by using a one second time step, which is physical in terms of fractional removal. This can be seen directly in Table 2.1, where 10% of both isotopes and the net count are removed.

Table 2.1: Serpent2 Second Setting Approaches

Labels	Initial	Batch	Net	A	B	Null
A	1000	900	949.5	900	999	990
B	10	9	-40.5	-90	9	0
Net	1010	909	909	810	1008	990

For the continuous reprocessing, the first attempt might be to try and remove 10% of the net by splitting it amongst the isotopes evenly. The issue with this approach is immediately noticeable, which is that the concentration of isotope B becomes negative, as can be seen in the Net column. A logical next approach would be to try and make one of the isotopes exactly correct while bringing the other along with it, which can be seen in the A and B columns. The results of this approach are seen in the table, and it can be seen that only by looking at the isotope with the smallest concentration and using that to determine the amount to remove can the result be ensured to be non-negative. One final approach to consider is to set the smallest concentration to 0, and bring the other isotopes along. This approach can be seen in the $Null$ column, and it also does not work very well.

2.1.2 DNP Movement

The movement of DNPs in liquid fueled molten salt reactors impacts the operation of the reactor directly since the decay of the precursors occurs at a different location compared to a static reactor [44, 3]. There are several effects which this movement can have on the reactor. Since the delayed neutron precursors move, there is a probability of decay in non-core regions of the reactor, meaning there are fewer delayed neutrons in the core. This results in a reduced effective delayed neutron fraction, which reduces controllability of the reactor. Additionally, since delayed neutrons have a softer spectrum than prompt neutrons, there is a difference in spectrum in a reactor which has no flow against a reactor which has a circulating fuel. In a solid fueled reactor, the flux and the concentration of delayed neutron precursors have the same profile. However, a moving fluid fuel causes the precursors to have a shifted profile based on the fluid flow rate, mixing, and decay rate of the precursors. This was shown very clearly by Jun Shi and Massimiliano Fratoni in their work, which can be seen in Figure 2.1.

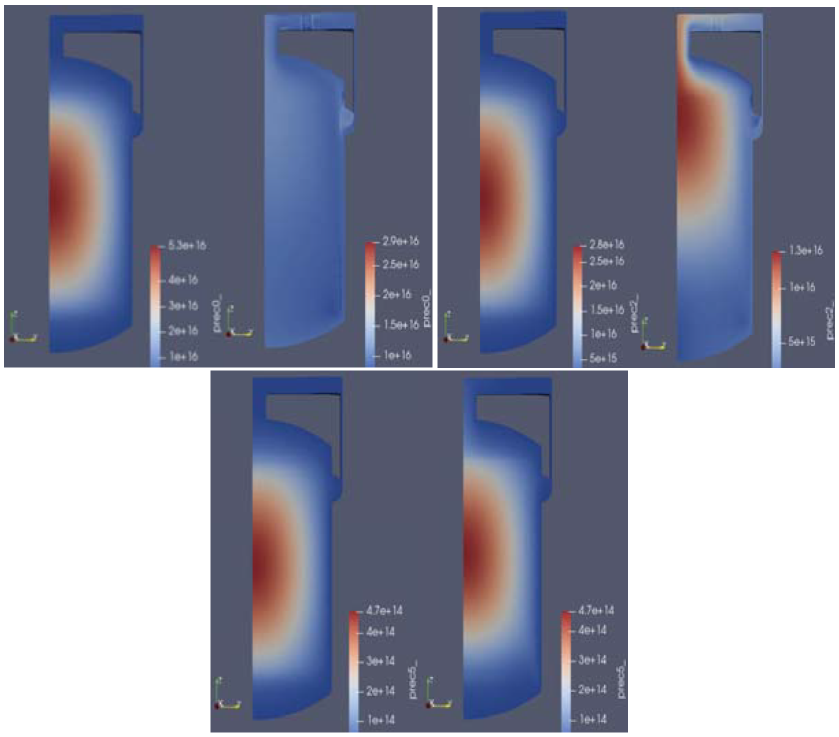

Figure 2.1: Plots of DNP concentrations in the molten salt reactor experiment [41]. The left side of each image shows concentration with no flow, while the right shows the concentration with a 1200 gallon per minute flow rate. Top Left) Longest lived DNP group. Top Right) Third longest lived DNP group. Bottom) Shortest lived DNP group.

From these images, it can be seen that for the long shortest lived DNPs, the movement of the fuel does not have any significant impact upon the DNP distribution when compared to static fuel. This is reasonable since the precursors do not live for very long in this group, so they do not have much chance to travel. For the intermediate DNP group, there is a clear shift in the direction of the fuel flow, which causes more precursors to produce delayed neutrons in the less important piping and upper plenum regions. The longest lived DNP group seems to be spread almost equally throughout the entire reactor, which means that those delayed neutrons are contributing significantly less to the overall neutron economy. This reduces the overall effective delayed neutron fraction since the delayed neutrons are in less important regions. This directly affects the controllability of the reactor through the reactor period, which is heavily dependent upon the delayed neutrons.

2.2 Depletion

Depletion is the process of burning the fuel and simulating the changes to the materials this causes in the system. Depletion requires specific information to run properly, such as decay constants, fission yields, and fission and transmutation cross section data. The fission and transmutation cross section data can come from a transport calculation, while the other data can be read from a data library [25]. Additionally, running depletion involves solving the Bateman equations, a generic version of which can be seen in Equation 2.4.

There are several different approaches to solving these equations, such as the Transmutation Trajectory Analysis (TTA) method and the Chebyshev Rational Approximation Method (CRAM). These are the two different methods used in Serpent2, and are both built into Serpent2 [26]. Other codes may use external software to compute depletion, such as REM used by MCNP and ORIGEN-S used by KENO-VI [4]. Another code, aside from Serpent2, that contains built-in depletion solvers is ERANOS [4].

The purpose behind solving these equations and having depletion models is to generate an accurate representation of the composition of the target materials during reactor operation. This is important in determining how long the reactor can run, how the safety of the reactor develops over time, and how operation may have to change to adjust to the new reactor state. For molten salt reactors, depletion also allows for information regarding fresh fuel feed rate and fission product removal rates, as well as how variations of those parameters can affect reactor performance and behaviour.

2.3 The Molten Salt Breeder Reactor

The molten salt breeder reactor (MSBR) is a useful design to analyze for several reasons. Because it is a molten salt reactor, online reprocessing is used within the design. In addition, it contains an inner and outer zone, each of which has different neutronic behaviour [35]. More specifically, the inner zone has a softer spectrum, a higher fission rate, and a 13% fuel-to-graphite ratio [32]. The outer zone has a harder spectrum, a higher breeding rate, and a 37% fuel-to-graphite ratio. The inner and outer zones prove to be an issue when accurately modeling using a unit-cell or one-region approach, as those models are unable to capture the different characteristics of each region [39]. The differences in the regions can also be seen in Figure 2.2, which allows for the difference in the fuel-to-graphite ratio to be visually noticeable.

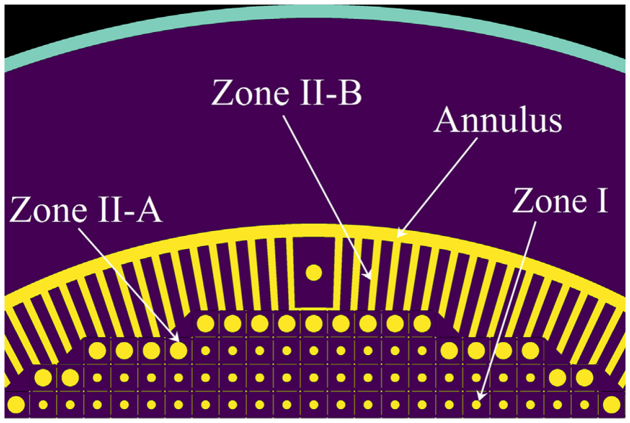

Figure 2.2: MSBR core axial slice showing the different regions from [39]. Zones II-A and II-B are where the spectrum is harder and there is increased breeding. Zone I is where there is more fission and a softer spectrum. The yellow is fuel salt, the purple is graphite, and the cyan is the reactor vessel.

Table 2.2 has general information on the MSBR, while Table 2.3 has specific information on the distribution of the fuel salt during operation. It can be seen in Table 2.2 that the salt inventory cycle time is 10 days, which means that for the entire salt inventory to be cycled takes 10 days. However, the loop cycle time is 11 seconds [35].

Table 2.2: MSBR General Data [35]

Component	Data
Thermal Capacity	2250 MW$_{th}$
Vessel Inner Diameter	6.77 m
Core Height	3.96 m
Vessel Pressure	0.52 MPa
Inner Salt Fraction	0.13
Outer Salt Fraction	0.37
Maximum Flow Velocity	2.6 $\frac{m}{s}$
Fuel Salt (Vessel)	30.4 m^3
Fuel Salt (Primary System)	48.7 m^3
Thorium Inventory	68,100 kg
Breeding Ratio	1.06
Processing Rate	1 gpm
Salt Inventory Cycle Time	10 days
Salt Components	^{7}LiF$-$BeF$_2$$-ThF_4$$-UF_4$
Salt Composition	71.7-16-12-0.3 mole %

Table 2.3 shows that the salt inventory in the core is approximately 19 m^3 due to each core zone. However, SaltProc uses the volume of the primary system for its calculation [39]. This is because the other parts of the MSBR outside of the core are not modeled, but the geometry implemented in Serpent2 uses the correct dimensions. This allows cross sections to remain as if the correct volume were implemented, but also allows the depletion results to be closer to the expected result if the fuel salt were fully utilized.

Table 2.3: MSBR Primary System Salt Inventory [35]

Region	Volume [m^3]
Fuel Salt (Primary System)	48.7
Reactor	-
Core Zone I	8.2
Core Zone II	10.8
Plenums, Inlets, Outlets	6.2
Annulus	3.8
Reflectors	1.4
Primary Heat Exchangers	-
Tubes	7.6
Inlets, Outlets	0.8
Pump Bowls	5.2
Piping (including drain line)	4.1
Off-gas bypass loop	0.3
Tank heels and miscellaneous	0.3

Table 2.4: MSBR Online Reprocessing Cycle Times [35]

Reprocessing Group	Element(s)	Cycle Time (Full Power)
Rare Earths	Y, La, Ce, Pr, Nd, Pm, Sm, Gd	50 days
Rare Earths	Eu	500 days
Noble Metals	Se, Nb, Mo, Tc, Ru, Rh, Pd, Ag, Sb, Te	20 seconds
Seminoble Metals	Zr, Cd, In, Sn	200 days
Gases	Kr, Xe	20 seconds
Volatile Fluorides	Br, I	60 days
Discard	Rb, Sr, Cs, Ba	3435 days
Salt Discard	Th, Li, Be, F	3435 days
Protactinium	Pa	3 days
Higher Nuclides	Np, Pu	16 years

Shown in Table 2.4 are the cycle times for different elements in the MSBR. According to Robertson et al, the protactinium and rare earth processing has the largest impact on neutronics and performance [35].

The fuel processing of the MSBR is defined to operate continuously with processing methods [35]. For protactinium removal, fluorination is first used to remove uranium before protactinium isolation. Next is countercurrent bismuth with lithium and thorium for stripping any remaining uranium and the protactinium. This is followed by hydrofluorination to separate the uranium and protactinium. This process can be seen in Figure 2.3. For a 10 day protactinium cycle time, a fuel salt flow rate of 0.88 gallons per minute (gpm) is used. The processed salt is processed for rare earth fission products before being fed back into the reactor.

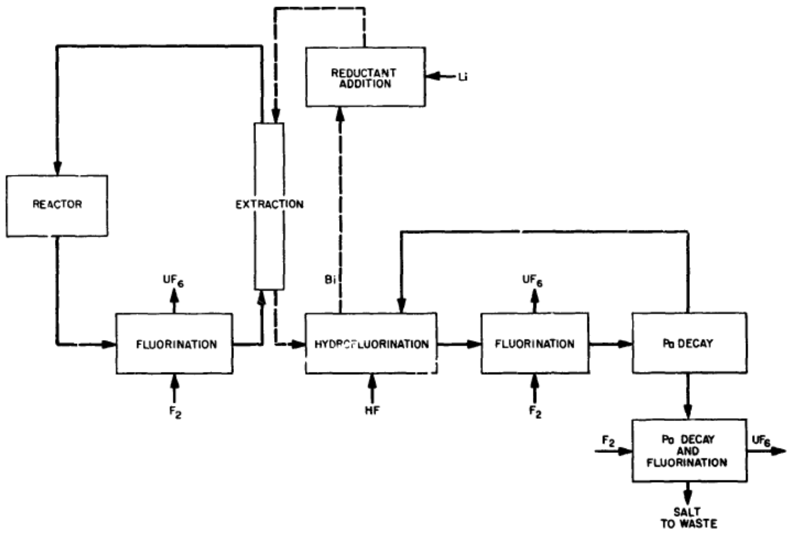

Figure 2.3: MSBR protactinium processing scheme from Robertson et al [35].

Rare earth fission product removal is performed using the metal-transfer process, which uses lithium chloride and bismuth containing a reductant. More specifically, bismuth containing a reductant of thorium and lithium is used to strip the rare earth fission products from the fuel salt. The rare earth fission products are then transported to the lithium chloride acceptor salt, though lithium bromide or a mix of both could be used. Figure 2.4 shows this process, and Figure 2.5 shows the protactinium and rare earth processing schemes together.

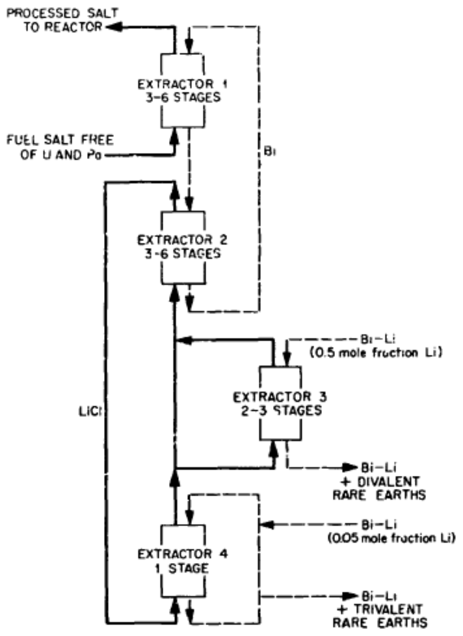
PROCESSED SALT TO REACTOR
EXTRACTOR 1 3-6 STAGES
FUEL SALT FREE OF U AND Pa
Bi
EXTRACTOR 2 3-6 STAGES
Bi-Li (0.5 mole fraction Li)
EXTRACTOR 3 2-3 STAGES
LiCl
Bi-Li + DIVALENT RARE EARTHS
Bi-Li (0.05 mole fraction Li)
EXTRACTOR 4 1 STAGE
Bi-Li + TRIVALENT RARE EARTHS

Figure 2.4: MSBR rare earth processing scheme from Robertson et al [35].

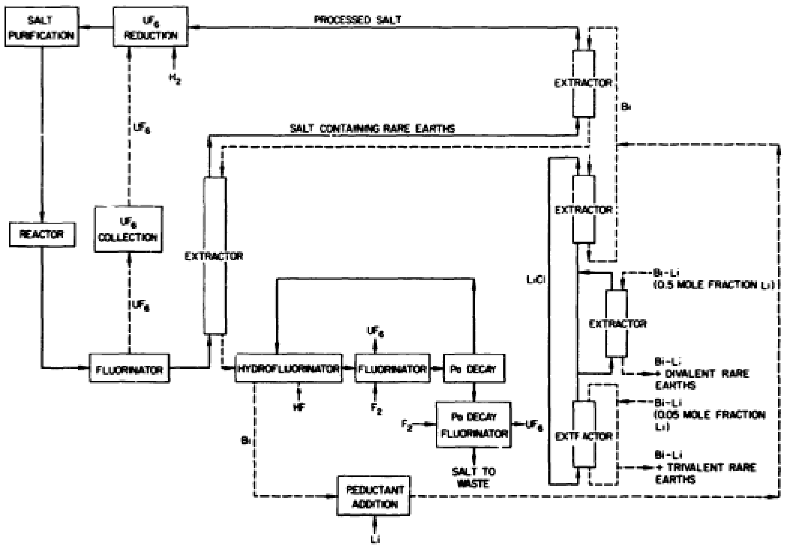

Figure 2.5: Combined MSBR rare earth and protactinium processing schemes from Robertson et al [35].

To strip gaseous fission products, a bubble generator introduces sparging gas in 15 to 20 mil bubbles. 10 percent of the flow is redirected to the bypass loop which contains a gas separator. This gas separator strips the fission products with approximately 100 percent efficiency. The off-gas system also includes an approximately 2 hour holdup during which noble metals deposit on the fuelsalt drain tank surface. Figure 2.6 shows this process, beginning with the number per time, or flux, of the noble gasses in the salt migrating to the graphite voids and helium sparging bubbles. This is followed by the volume holdup into more processing to handle the noble metal fission product cleanup.

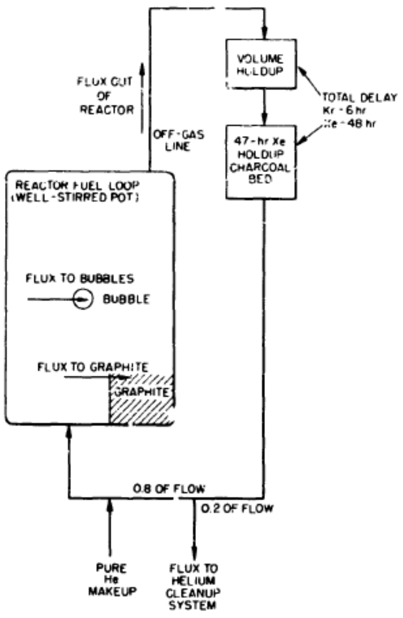
FLUX OUT OF REACTOR
OFF-GAS LINE
VOLUME HOLDUP
TOTAL DELAY Kr - 6 hr Xe - 48 hr
47-hr Xe HOLDUP CHARCOAL BED
REACTOR FUEL LOOP (WELL-STIRRED POT)
FLUX TO BUBBLES
BUBBLE
FLUX TO GRAPHITE
GRAPHITE
0.8 OF FLOW
0.2 OF FLOW
PURE He MAKEUP
FLUX TO HELIUM CLEANUP SYSTEM

Figure 2.6: MSBR noble metal flux and off-gas system from Robertson et al [35].

Periodic discard of salt at a rate of 0.1 cubic feet per day occurs due to buildup of non-volatile fluorides during fluorination accumulating in the decay tank, such as from zirconium and corrosion product nickel.

2.4 MSR Modeling Approaches

For modeling of MSRs, there are generally two different models which are used. The first is a model which takes a given fuel composition and analyzes the performance of the reactor given that fuel composition. This may include focus on aspects such as DNP movement [15, 41] or reactor dynamics [43, 12, 5, 13, 42].

The second type of model is one which depletes the fuel. Because the online reprocessing of an MSR has such large effects on its fuel composition, depletion models of MSRs typically incorporate online reprocessing. These models can also account for other factors, such as DNP movement, as shown by Zhou et al [46]. The movement of DNP can have a noticeable effect on reactor performance by affecting neutron energy spectra and distribution of delayed neutrons [7]. Most, however, focus primarily on depletion and do not consider the DNP movement during depletion, as can be seen in Table 2.5. In order to handle spatial dependence of fuel salt without modeling the piping of the core, models can use the scaled flux method [6].

For all of the continuous reprocessing models given in Table 2.5, all of them use the "fictitious decay constant" approach, which is shown in Equation (2.5). For the batchwise models, they typically use the approach shown in Equation (2.1) and remove a fraction at each depletion step rather than bulk removal at certain steps.

The "Reprocessing" column of Table 2.5 shows the mathematical approach taken to the online reprocessing of the work, and does not necessarily represent the reactor reprocessing approach. However, since online reprocessing is considered in the table, offline batchwise reprocessing is not included, such as in the work by Zou et al [48].

Table 2.5: Molten Salt Reactor Models with Online Reprocessing

Index	Reprocessing	Model	Reactor	DNP	Ref.
1	Batchwise	Full 3D	MSBR	No	[39]
2	Continuous	Full 3D	MSFR	No	[4]
3	Batchwise	Full 3D	MSBR	No	[32]
4	Batchwise	Unit Cell	MSBR	No	[8]
4*	Continuous	Unit Cell	N/A	No	[23]
5	Batchwise	Unit Cell	DMSR	No	[2]
6	Continuous	Full 3D	MSBR	Yes	[46]
7	Batchwise	Full 3D	MSBR	No	[31]
8	Batchwise	Full 3D	MOSART	No	[40]
9	Continuous	Unit Cell	ADNA	No	[36]
10	Mixed	2D Slice	MSFR	No	[16]
11	Continuous	Full 3D	MSFR	No	[47]
12	Batchwise	Full 3D	Toy	No	[34]
13	Mixed	2D Slice	N/A	Yes	[20]
14	Mixed	Full 3D	TMSR	No	[28]
15	Mixed	Unit Cell	Toy	Yes	[30]
16	Continuous	Full 3D	N/A	No	[45]

The first entry in Table 2.5 is SaltProc's MSBR example, which uses Serpent2, a full 3D core, 3 day time steps, and runs for 60 years. The batchwise reprocessing in SaltProc includes both the removal and feed processes.

The second entry is by Aufiero et al, and displays the Serpent2 newly developed continuous reprocessing functionality. A full 3D simplified core layout of the molten salt fast reactor (MSFR) is used, and continuous reprocessing is used to operate the removal and feed processes. It evaluates the model at various scales, from steady state uranium concentrations at over 60 years to radiotoxicity over 10^8 years. The step sizes vary as well, though not specifically mentioned, appearing very small at BOC and expanding to around 10 years at SS for one of the figures presented.

The third entry by Park et al is very similar to the MSBR example in SaltProc, with the primary difference being MCNP6 is used for transportation calculations instead of Serpent2. The depletion was performed using CINDER90 and custom Python scripts. Another difference is that SaltProc uses adjusted values for the removal efficiency rates of xenon, krypton, and protactinium, while the work by Park et al uses the cycle time values from Robertson et al [35]. This model uses 3 day time steps and models the fuel composition over 20 years.

The fourth entry is for ChemTriton, a tool designed for SCALE to perform online reprocessing of MSRs. It contains work on multiple reactors, such as the MSBR and the MSRE. The tool iteratively runs SCALE/TRITON over small time steps to simulate continuous reprocessing. For the MSBR case, 3 day time steps are used, similar to SaltProc and Park et al. The reason for not using shorter time steps is that using shorter time steps increases computational cost while also providing very little improvements to calculated eigenvalue and fuel cycle metrics, discussed by Betzler et al as well as by Powers et al [33].

ChemTriton has been updated after TRITON implemented new tools for continuous material feeds and removals. The work discusses in depth how the removal rates for isotopes can be calculated using a well mixed approximation and accounting for non-modeled piping by introducing a correcting decay factor. Although it does not account for DNP movement, it does account for neutron important regions without having to model the piping regions.

The fifth entry by Ahmad et al analyzes single fluid and two fluid denatured molten salt reactors (DMSRs). This work uses MCNP5 and ORIGEN2 as the codes to perform transport and depletion over 30 years. Although the steps at which fuel is adjusted is at 5 year intervals, the burnup intervals are actually set to 6 months. The reason such large steps were used is because the work analyzes proliferation and resource requirements for the reactors. This means that the neutronics and safety parameters were not as important, and so the model did not have to be as precise in that regard. Additionally, only gaseous fission products or fission products which are removed via sparging are included in the reprocessing scheme, no other chemical extraction takes place. This is because the authors wanted the system to be as simple as possible. ORIGEN provides continuous reprocessing capability, however for this work it would not have been necessary since the authors wanted to use a simpler approach for the model [17].

The sixth entry by Zhou et al is for FAMOS (Fuel cycle Analysis code for MOlten Salt reactors). This code uses OpenMC to generate homogenized cross section data followed by the DIF3D diffusion solver along with extensions to model DNP movement, thermal hydraulics, depletion, and continuous reprocessing. Their work also analyzes the molten salt fast reactor over 200 operational years. The depletion step size is not specifically given in the work, but from analysis of the plots, small time steps seem to have been used at the beginning of cycle and then larger time steps of 100 days are then used for the rest of the steps.

The seventh entry by Nuttin et al analyzes the MSBR using a slightly simplified geometry with MCNP. The reprocessing approach used here is also the same as the approach used in SaltProc, which is to take the inverse of the cycle time to determine the removal rate. Depletion is performed in this work by coupling MCNP to an evolution program and the NJOY code. However, this work uses 10 day steps instead of 3 day steps that SaltProc uses. It also runs for 100 years instead of 20.

The eight entry by Sheu et al analyzes the Molten Salt Actinide Recycler & Transmuter (MOSART) reactor and uses custom scripts to implement batchwise reprocessing. The codes used are SCALE6 for transport and TRITON for depletion. The step sizes used include 15, 30, and 60 days, in which the authors found that 30 days gave the best results in terms of accuracy and run time balancing for the 30 year simulation.

The ninth entry by Rodriguez-Vieitez analyzes the Accelerator-Driven Neutron Applications (ADNA) Tier-1 reactor using MCNP. Although the model used does implement continuous reprocessing in the Bateman equation, it is not used for the purpose of fuel cycle analysis and depletion. Rather, the work iteratively works towards the equilibrium fuel composition in order to analyze the effect of varying different reactor parameters, such as geometry.

The tenth entry by Fiorina et al is on the Molten Salt Fast Reactor (MSFR) and uses a modified version of the ERANOS-based EQL3D procedure. This model makes use of continuous reprocessing for non-soluble fission products, but uses batchwise reprocessing for soluble fission products. The batchwise reprocessing was incorporated for simplicity in the model. The time steps used are shorter at beginning of cycle at extend out to 30 year steps for a 200 year depletion. Additionally, the radiotoxicity analysis extends out to 1 million years.

The eleventh entry by Zhuang et al discusses an OpenMC extension called OpenMCB-MSR to implement continuous reprocessing in an analysis of the MSFR. The depletion step sizes used are not specified in the work, though with continuous reprocessing step sizes could have been set to above single digit days and still maintained high precision.

The twelfth entry by Ridley and Chvala use batchwise reprocessing on a toy MSR using Serpent2 and a python library. The feed rates were set to fluctuate in order to maintain criticality while also accounting for density imbalance in Serpent2 by increasing the volume of the material. The full core 3D model was then analyzed using 7 day depletion steps for 10 years.

The thirteenth entry is on the Reactor Optimum Design (ROD) code, which allows users to use continuous or batchwise reprocessing for the MSR model. ROD operates by taking a one or two dimensional input and determining the equilibrium fuel composition. This is a useful code for determining a prediction of a model, but is less precise than more modern codes which offer more detailed geometry and other features.

The fourteenth entry by Merle-Lucotte et al covers depletion of the Thorium Molten Salt Reactor (TMSR) using MCNP4 [19] for transport and REM for material evolution. Continuous reprocessing is used for gaseous fission product extraction, while the other fission products are processed with batchwise reprocessing.

The fifteenth entry by Nagy et al uses a toy model to go through different reprocessing schemes based on the MSBR and MOSART reprocessing designs [35, 22]. SCALE-5 [1] is used for neutron transport and ORIGEN-S is used for depletion. To account for DNPs, the authors make a few approximation which lead to reducing the calculated eigenvalues by 90 pcm. Depeletion steps are performed every 5 days, during which continuous reprocessing was implemented for either only gaseous fission product removal or more, depending on the reprocessing scheme chosen. Additionally, the batchwise reprocessing was performed by altering the binary output of ORIGEN, as is performed for the feed fuel and more, depending on the reprocessing scheme chosen.

The sixteenth entry by Xia et al demonstrates the MOlten salt reactor specific DEpletion Code (MODEC). This code uses continuous reprocessing, and has been used to analyze the Molten Chloride salt Fast Reactor (MCFR) by Liaoyuan et al [27]. The MCFR analysis uses SCALE6.1 for transport and MODEC for depletion.

Chapter 3

Methods

3.1 Batchwise Reprocessing

SaltProc is used to handle batchwise reprocessing in this work because it provides a batchwise reprocessing scheme which can be customized fairly easily. There are different versions of SaltProc which have slightly different methods in how the depletion data is stored as well as how reprocessing rates are calculated.

3.1.1 Bulk

SaltProc version 0.1 is the first full version of SaltProc which was publicly released, and provides all the functionality necessary for modeling the MSBR with batchwise reprocessing. This version uses 3 day steps and does not perform fractional reprocessing at each step, but instead performs reprocessing according to the cycle times, which is termed as the "Bulk" approach to batchwise reprocessing. This is the same approach used by Gehin and Powers for an analysis of the MSBR [18].

For example, a 30 day cycle time would be implemented as no reprocessing during the first 27 days, then full 100% removal of the target at the 30 day mark. This works well for reactor processes which are performed batchwise, such as adding uranium to the reactor which can be performed as a batch process. However, for approximating online reprocessing, this approach is not ideal, as it is less able to capture the frequency of the reprocessing.

For the refueling of the reactor, the thorium feed rate was set to maintain the thorium mass in the reactor. The uranium feed rate was set to be equivalent to the protactinium removal rate. This carries an implicit assumption that there is already a backlog of decayed protactinium which has transmuted into uranium which is ready to be pumped back into the core. Both the thorium feed and protactinium to uranium refueling processes occur every 3 days. These refueling rates can be seen in Figures 3.1 and 3.2, which are recreated from the data from Rykhlevskii [37].

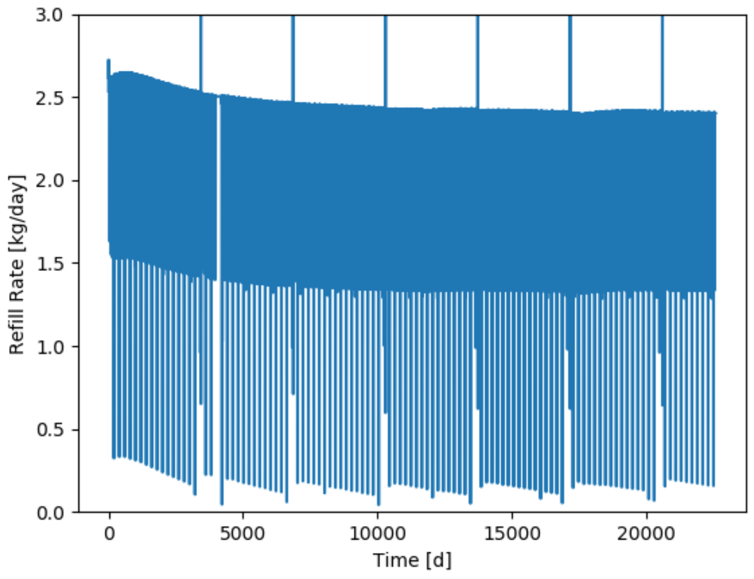

Figure 3.1: Thorium feed rate in the MSBR as a function of time while using bulk batchwise reprocessing [37].

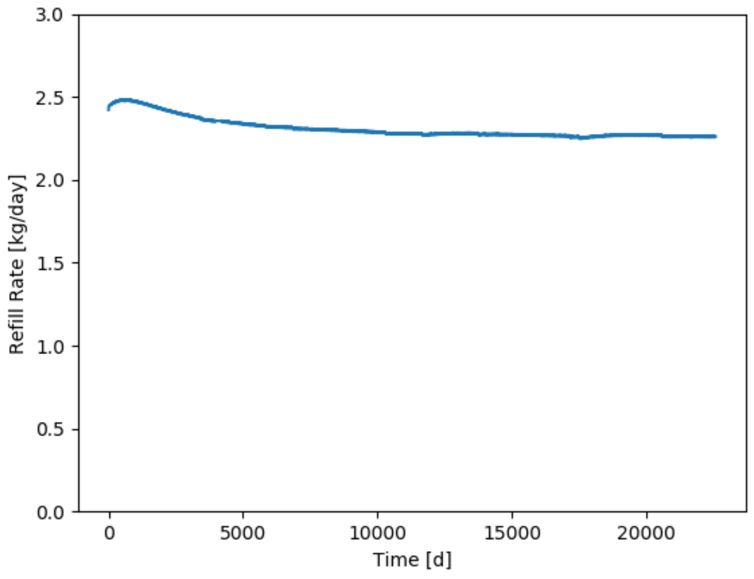

Figure 3.2: Uranium feed rate in the MSBR as a function of time while using bulk batchwise reprocessing [37].

The large jumps in the thorium feed rate are due to the bulk removal of absorbers, leading to spikes in the effective multiplication factor [37]. The uranium feed rate is smooth because it is based purely on the outflow of protactinium and does not fluctuate as rapidly as the thorium, which is the only external feed in the MSBR system. The average thorium-232 feed rate is 2.38 kg/day, while the average uranium-233 feed rate is 2.31 kg/day.

These average values are used as the baseline for feed rates for the other batchwise method and for the continuous methods.

Another difference is that this version makes use of the h5py Python package to handle the hdf5 data files which contain depletion data. Within the hdf5 file, the atom density of each isotope before and after batchwise reprocessing is recorded as well as different reactor parameters such as the neutron multiplication factor.

3.1.2 Steady

SaltProc version 0.3 includes an example MSBR which users can run as soon as SaltProc is installed, though some of the run parameters such as neutrons per generation and run time have to be altered to generate a more valid model. This model also uses 3 day steps, but instead of performing the batch removal only at the cycle time value, this model removes a fraction every 3 days, which is called the "Steady" approach for batchwise reprocessing.

For example, a 30 day cycle time would result in 10% removal every 3 days. This more accurately models online reprocessing, which is primarily what the MSBR employs in its reprocessing scheme aside from the salt discard.

The reactor refueling is performed slightly differently from v0.1 as well. This version assumes a constant ratio between the uranium refueling and the thorium refueling. The net refueling for each 3 day depletion step is set to maintain the mass of the system. Thus, the average values of 2.38 kg/day and 2.31 kg/day for thorium-232 and uranium-233, respectively, are used to determine the ratio of uranium to thorium which is implemented.

Additionally, this version of SaltProc uses the PyTables Python package for handling the hdf5 data files. The data files are structured similarly to v0.1 of SaltProc, but instead of atom density they use mass, which makes analysis of the files slightly more user friendly.

3.1.3 Batch Approaches Summary

Table 3.1: Batchwise Reprocessing Methods

Approach	Δt $[s]$	T_{cyc} $[s]$	Fractional Removal Rate [s^{-1}]	Step Removal
Bulk	1	20	-	0/1*
Bulk	10	20	-	0/1*
Bulk	40	20	-	1
Steady	1	20	0.05	0.05
Steady	10	20	0.05	0.5
Steady	40	20	0.025	1

* Bulk removal is 0 until the depletion step is equal to the cycle time, at which point it removes 100%.

Table 3.1 reflects the information shown in Figure 3.3. The table shows how the bulk method will only remove 100% at the cycle time, and otherwise removes nothing. Additionally, the bulk method will remove 100% as soon as possible if the depletion step size, Δt, is larger than the cycle time, T_{cyc}.

The table also describes how the steady method fractional removal is constant up until the depletion step size becomes larger than the cycle time, at which point the removal at each step is 100%, causing the steady method to then be equivalent to the bulk method. However, for shorter times, the steady method allows for a consistent fractional removal rate by adjusting the removal at each step.

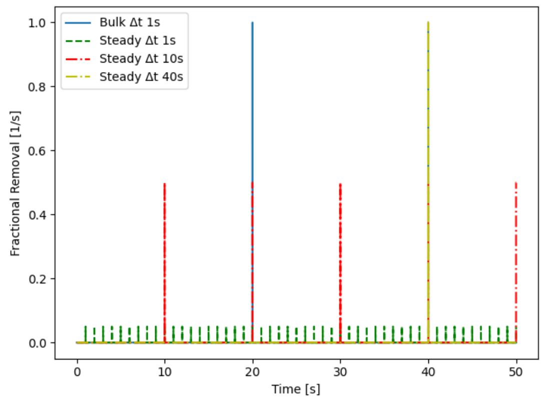

Figure 3.3: Plot showing how bulk and steady batchwise reprocessing removal works as a function of time for an example with a cycle time of 20 seconds.

Figure 3.3 shows an example fractional removal scheme for a fission product with a cycle time of 20 seconds. The Bulk method simply extracts 100% of the product every 20 seconds, no matter how small the depletion step size is. The steady method removes some fraction every depletion step, which allows a semi-continuous process as the depletion step size becomes smaller and smaller. This can be seen in how a 10 second depletion step results in 50% removal every 10 seconds, whereas a 1 second step allows for 5% removal per depletion step.

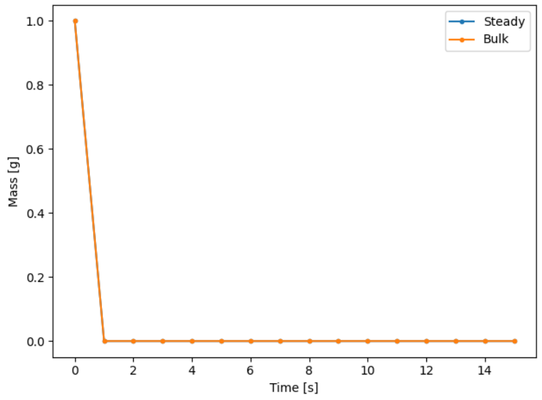

Figure 3.4: Plot showing how bulk and steady batchwise reprocessing are identical while the cycle time is shorter than or equal to the depletion step size.

A useful piece of information to note is that if the cycle time is shorter than or equal to the depletion step size, then the steady and bulk batchwise methods will be identical to each other, as demonstrated in Figure 3.4 which shows a simple model case with 1 second depletion steps and 1 second cycle times for an imaginary isotope. For the MSBR simulation, this means that the noble metals, gasses, and protactinium reprocessing will remain the same for either batchwise method employed. However, the longer cycle time groups such as the rare earths and volatile fluorides have cycle times which are 10-20 times longer than the depletion step size. Figure 3.5 shows a simulation of this for a situation where an isotope is generated at a rate of 1 gram per second, and is simulated with some combination of cycle time and depletion step size where the depletion step size is smaller than the cycle time.

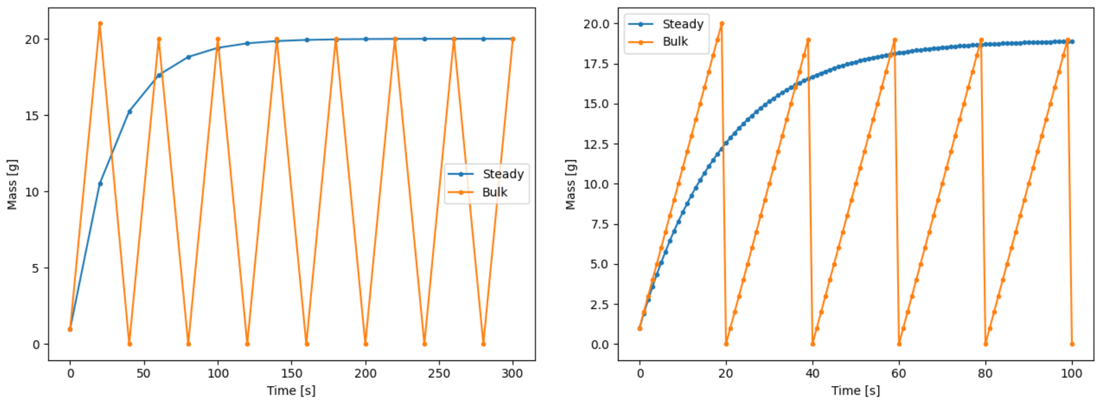

Figure 3.5: Plot showing how bulk and steady batchwise reprocessing are different while the cycle time is longer than the depletion step size for a 40 second cycle time and 20 second depletion step (top) and a 20 second cycle time with a 1 second depletion step (bottom).

One interesting aspect to note from this figure is that the steady method approaches a steady state value which is at the peak of the bulk method. This result can be confirmed by checking that the solutions for both the steady and bulk methods are valid. To check the solution, the 40 second cycle time and 20 second depletion step will be used.

For the bulk method, after 100% removal after the first cycle time, or 40 seconds, the mass then oscillates between 20 and 0 grams. This makes sense since 20 grams are generated over 20 seconds while every 40 seconds there is 100% removal.

For the steady method, the initial mass of 1 gram gains 20 grams over 20 seconds, yielding 21 grams. However, the steady removal then requires removal of 50%, previously discussed and shown in Figure 3.3. This 50% removal then drops the mass to 10.5 grams. 20 more grams are added to this value and 50% is removed again, continuing iteratively. This can then be re-written as an infinite sum in order to determine the steady state solution. The first three iterations are shown in Equations (3.1) and (3.2), where the value of 20 is the mass added during each depletion step and the value of 0.5 is the fractional removal performed each depletion step.

$$m_{ss} = (((N_0 + 20)(0.5) + 20)(0.5) + 20)(0.5) \tag{3.1}$$

$$m_{ss} = 0.5^3 N_0 + 0.5^3(20) + 0.5^2(20) + 0.5(20) \tag{3.2}$$

Continuing this form infinitely yields Equation (3.3), where the infinite sum yields 1, resulting in a net value of 20, the result of which is unaffected by the initial mass of the isotope and instead depends on the generation and reprocessing rates.

$$m_{ss} = 20 \sum_{n=1}^{\infty} \frac{1}{2}^{n} \tag{3.3}$$

Overall, this shows that the steady and bulk batchwise methods have some similarities, but for elements with longer cycle times, the bulk method will experience oscillations whereas the steady method will level off smoothly.

3.2 Continuous Reprocessing

The continuous reprocessing functionality of Serpent2 was undocumented for several years aside from on the Serpent2 forums, but more recently work has gone into providing documentation. However, there has not been comparisons of Serpent2 continuous reprocessing with Serpent2 batchwise reprocessing with no differences between the models aside from the mathematical reprocessing approach used.

There are three different options which can be used for the Serpent2 built-in continuous reprocessing. The different options are defined in Serpent2 as 0, 1, and 2; which are henceforth referenced as Constant, Decay, and Step reprocessing. These names are selected because they closely describe the behaviour of each of the methods.

3.2.1 Constant

The Constant reprocessing method is the only method of the three options which does not conserve mass. This option instead generates mass the same way as the Decay reprocessing method for the initial step, then continues using that mass rate from that point onwards.

This method is useful for handling a constant fresh fuel feed rate, or any other situation where a constant addition of mass is desired. Because this method does not remove mass, it is not useful for any sort of constant removal.

3.2.2 Decay

The Decay reprocessing method is the most useful of the three options, and that is due to its flexibility in its usefulness. This method adds a decay term to whatever material it is attached to, and that decay term becomes a feed term to whatever material is set to receive the flow. For fission product removal, this is likely some material which is not modeled in the geometry of the problem but only exists to receive the fission product waste. However, this term can instead be leveraged to turn it into a feed rate.

This can be performed by attaching the decay term to a material which contains a volume of the desired feed, and then having that material feed into the core. The feed then "decays" from the feed tank into the core, essentially operating as a feed. Using this same method but altering the reprocessing constant or volume of the feed tank allows for an essentially constant feed rate, where the removed mass is negligible compared to the net mass.

Additionally, for this method, there are multiple approaches which can be used to calculate the reprocessing constants that should be implemented. These approaches are named "Cycle Time Decay", "Cycle Rate", and "SaltProc Cycle Rate" accordingly. The Cycle Time Decay approach treats the cycle time as if it is twice a half-life, and the cycle time is an exponential process. The Cycle Rate approach treats the cycle time as the time where 100% of removal occurs, and then linearly extrapolates by assuming an equal percentage removal occurs up to that point.

The reason these approaches are implemented is because molten salt reactor reprocessing schemes provide reprocessing data in terms of cycle times. The cycle time is the amount of time it takes for a given element to be removed from the system. This cannot be directly solved using the Bateman equation, as shown in Equations (3.4) through (3.6) and (3.7) through (3.10).

$$\frac{dN}{dt} = -\lambda_r N \tag{3.4}$$

$$N(\tau) = 0 = N_0 e^{-\lambda_r \tau} \tag{3.5}$$

$$-\,ln(0) = \lambda_r \tau \tag{3.6}$$

Equations (3.4) through (3.6) show how the solution results in a reprocessing constant of infinity when only continuous removal is considered.

$$\frac{dN}{dt} = C - \lambda_r N \tag{3.7}$$

$$N(\tau) = 0 = N_0 e^{-\lambda_r \tau} + \frac{C}{\lambda_r}\left(1 - e^{-\lambda_r \tau}\right) \tag{3.8}$$

$$0 = N_0 e^{-\lambda_r \tau} + \frac{C}{\lambda_r} - \frac{C}{\lambda_r} e^{-\lambda_r \tau} \tag{3.9}$$

$$0 = \lambda_r N_0 e^{-\lambda_r \tau} + C - C e^{-\lambda_r \tau} \tag{3.10}$$

Equations (3.7) through (3.10) show that the only real solution to a steady accumulation and continuous removal is with a reprocessing constant of 0.

3.2.2.1 Cycle Time Decay

The Cycle Time Decay approach is a simple method which makes use of the mathematical form of the Decay reprocessing method. Because it adds a "decay-like" term to the Bateman equation, this approach takes the cycle time for the target, cuts it in half, and then treats that value as the reprocessing half-life for that target. This process is shown in Equations (3.11) and (3.12).

$$\tau_{1/2} = \frac{T_{cyc}}{2} \tag{3.11}$$

$$\lambda_r = \frac{ln(2)}{\tau_{1/2}} \tag{3.12}$$

An example of implementing this approach for a 30 second cycle time would then have a 15 second reprocessing half-life. This is then converted to a reprocessing constant in the same way that a decay half-life is converted to a decay constant, which can be seen in Equation (3.13), resulting in a value of 0.0462 s^{-1}.

$$\lambda_r = \frac{ln(2)}{15} = 0.0462 s^{-1} \tag{3.13}$$

3.2.2.2 SaltProc Cycle Time Decay

This approach is the same as Cycle Time Decay, but alters for any target which has a cycle time less than the batchwise reprocessing step incorporated by SaltProc. For example, a 3 day cycle time for some target would be treated the same as the standard Cycle Time Decay. However, a cycle time shorter than the 3 day batchwise reprocessing step used by SaltProc for the MSBR has its half-life extended to the SaltProc minimum value of 1.5 days. For example, a 30 second cycle time would instead be treated as a 3 day cycle time, which would result in a half-life of 1.5 days. Plugging in, this would result in a reprocessing constant of 0.462 $days^{-1}$, or 5.348E-6 s^{-1}. This is the reprocessing constant for any cycle time which is less than or equal to three days.

3.2.2.3 Cycle Rate

The Cycle Rate approach uses a linear approximation such that the inverse of the cycle time is the rate at which material is removed. This is represented by, for example, 10% removal per second would neglect the efficiency decrease over time and give 100% removal after 10 seconds. This is calculated by investigating a unit time progression, i.e. 1 second or 1 day. Over this time period, the removal of atoms should be the fractional rate value, which means the final atom count at a time of 1 should be 1 - X, where X is the fractional removal rate. This can be seen in Equations (3.14) and (3.20).

$$X = \frac{1}{T_{cyc}} \tag{3.14}$$

$$\frac{dN}{dt} = -\lambda_r N \tag{3.15}$$

$$N(t) = N_0 e^{-\lambda_r t} \tag{3.16}$$

$$N(t=1) = (1-X)N_0 \tag{3.17}$$

$$(1-X)N_0 = N_0 e^{-\lambda_r (1)} \tag{3.18}$$

$$-ln(1-X) = \lambda_r \tag{3.19}$$

$$\lambda_r = ln\left(\frac{1}{1-X}\right) \tag{3.20}$$

However, the mathematical model actually results in exponential decay in the removal effectiveness due to a decreasing amount of the target to remove.

An example cycle time of 30 seconds would be modeled by taking the inverse, which gives 0.033 s^{-1}. This value is then converted to a reprocessing constant by plugging it into the solved differential equation form shown in Equations (3.21) and (3.22), giving a value of 0.0339 s^{-1}.

$$X = \frac{1}{30} = 3.33E\text{-}2s^{-1} \tag{3.21}$$

$$\lambda_r = ln\left(\frac{1}{1-3.33E\text{-}2}\right) = 0.0339s^{-1} \tag{3.22}$$

3.2.2.4 SaltProc Cycle Rate

The SaltProc Cycle Rate approach is the same as the Cycle Rate approach, but takes into account the limiting nature of the 3 day batchwise reprocessing step used by SaltProc. For example, a 6 day cycle time target would be modeled the same using the SaltProc Cycle Rate approach as the standard Cycle Rate approach. However, anything shorter than 3 days would be modeled differently, since that is the batchwise reprocessing step inocorporated by SaltProc for modeling the MSBR. For example, a 30 second cycle time would be analyzed instead as a 3 day cycle time, since SaltProc can only remove 100% of material after a minimum of 3 days. This means the inverse would be 0.333 $days^{-1}$, or 3.858E-4 s^{-1}. Converting to a reprocessing constant gives a value of 3.858E-6 s^{-1}. This is the reprocessing constant for any cycle time which is less than or equal to three days.

3.2.2.5 Direct Linear Approach

Another approach is to directly apply the Cycle Rate removal rate, which is the inverse of the cycle time, as the reprocessing constant [21]. This is referred to as the Direct Linear approach. This approach is similar to the Cycle Rate approach, though the derivation for it is slightly different. This can be seen in Equations (3.23) through (3.34), where Equations (3.32) and (3.33) uses L'Hôpital's rule.

$$X = \frac{1}{T_{cyc}} \tag{3.23}$$

$$\frac{dN}{dt} = -\lambda_r N \tag{3.24}$$

$$N(t) = N_0 e^{-\lambda_r t} \tag{3.25}$$

$$N_{cur} = N_{prev} e^{-\lambda_r \Delta t} \tag{3.26}$$

$$N_{cur} = (1 - \Delta t X) N_{prev} \tag{3.27}$$

$$(1 - \Delta t X) N_{prev} = N_{prev} e^{-\lambda_r \Delta t} \tag{3.28}$$

$$-ln((1 - \Delta t X)) = \lambda_r \Delta t \tag{3.29}$$

$$\lambda_r = \frac{-ln((1 - \Delta t X))}{\Delta t} \tag{3.30}$$

$$\lambda_r = lim_{\Delta t \to 0} \frac{-ln((1-\Delta t X))}{\Delta t} \tag{3.31}$$

$$\lambda_r = \frac{0}{0} \tag{3.32}$$

$$\lambda_r = lim_{\Delta t \to 0} \frac{X}{1 - X\Delta t} \tag{3.33}$$

$$\lambda_r = X = \frac{1}{T_{cyc}} \tag{3.34}$$

These equations follow a similar path to the Cycle Rate approach, but instead of using the linear approximation value after a unit time step, this approach generates the reprocessing constant while implementing the linear approximation as the time step goes to zero.

The differences between Cycle Rate and Direct Linear approaches can be seen in Table 3.2, where for longer cycle times, the difference is negligible, but for shorter cycle times, the difference becomes larger. However, since extremely short cycle times are not realistically practical, the two approaches are approximately equivalent. Overall, the Direct Linear approach is more numerically stable, however, since it does not have asymptotic until reaching a cycle time of 0 seconds, whereas the Cycle Rate method has asymptotic behaviour at 1 second cycle time. This can be seen in Figure 3.6. The asymptotic behaviour for a 0 second cycle time is not an issue because there are no negative cycle times, meaning the value is only approached from the positive side and behaves as physically expected.

Table 3.2: Decay Reprocessing Approaches

Cycle Time	Removal Rate $[s^{-1}]$	CR λ_r $[s^{-1}]$	DL λ_r $[s^{-1}]$	$\Delta\lambda_r$
3 d	3.86E-6	3.86E-6	3.86E-6	7.44E-12
20 s	0.05	5.13E-2	5.00E-2	1.29E-3
5 s	0.2	2.23E-1	2.00E-1	2.31E-2
2 s	0.5	6.93E-1	5.00E-1	1.93E-1
1 s	1	-	1	-

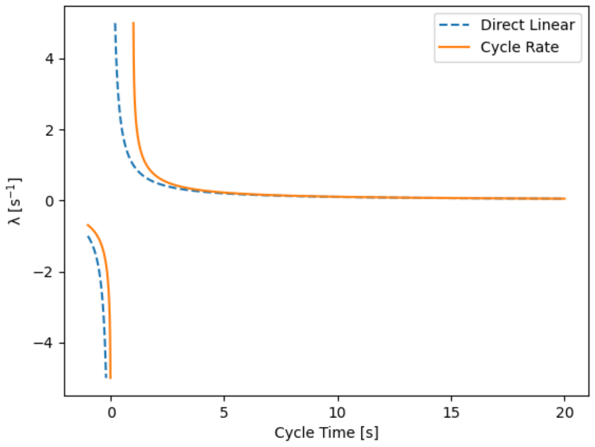

Figure 3.6: A comparison of the Direct Linear and Cycle Rate reprocessing constants for different cycle times.

Because the shortest cycle time for the MSBR is 20 seconds, this shows that the Cycle Rate and Directly Linear approaches are roughly equivalent for this reprocessing scheme.

3.2.2.6 Decay Approaches Summary

Figure 3.7 shows how the reprocessing constants for the different approaches vary as a function of cycle time.

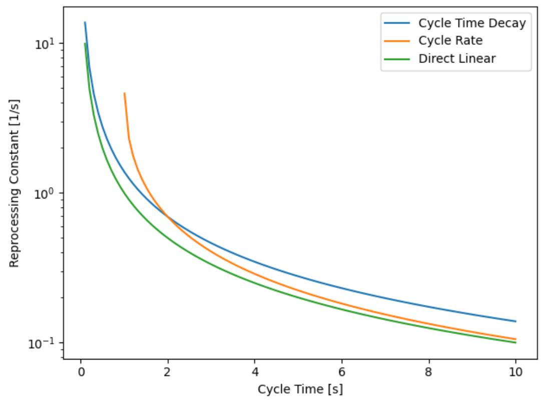

Figure 3.7: Plot of how reprocessing constants for different approaches vary with cycle time.

Figure 3.7 shows the three different continuous decay based methods for reprocessing as a function of various cycle times. From the figure, it can be seen that the Cycle Rate method does not have results for cycle times less than or equal to 1 second, which is a significant weakness of the method since physical processes could exist with a cycle time in that range. The other two methods provide results down to a 0 second cycle time, which allows full coverage of possible cycle time values.

The Direct Linear method behaves similarly to the Cycle Time Decay method during small cycle times and the Cycle Rate method for longer cycle times. This is due to the general form of the equations for each, where at small cycle times the inverse cycle time relationship with the Cycle Time Decay method dominates, while at larger cycle times the Cycle Rate log of the inverse term matches more closely.

3.2.3 Step

The Step reprocessing method implemented in Serpent2 is mathematically very similar to the Decay model, but instead of updating continuously in time, it instead is updated during new depletion steps. In this manner, it is a sort of mix between the Constant method and the Decay method. This is because over a single depletion step, it is a constant added or subtracted from the Bateman equation, while over many depletion steps, it follows the same exponential decay form of the Decay method.

This particular method is not useful for extracting fission products, and is not needed for constant feed rates since that can be modeled using the Decay method. This method could be useful for inducing a step drop in feed rate, though this would require running only a single depletion step until the drop occurred. Additionally, this drop could be simulated by running the Decay method and reducing the reprocessing constant. This would allow for flexibility in the distribution of depletion steps as well without having to worry about changing the behaviour of the reprocessing functionality.

Another potential issue with the Step method is that the depletion step can last long enough that the constant mass removal causes the mass to go negative. However, Serpent2 will cease running if this occurs. The Decay method does not mathematically allow for negative mass, the Constant method does not conserve mass, and the Step method stops running once there is negative mass.

The main potential use of the Step reprocessing may be for movement of material at a set rate. This could be implemented by writing a script to check the current number of atoms of the target and adjusting the reprocessing rate accordingly. However, for the MSBR, this form of reprocessing is not necessary, and is thus not implemented.

3.3 Mass Balancing

One of the useful features of batchwise reprocessing is that the net mass of the core can be balanced by adjusting the feed rates to provide the same amount into the core that is removed through reprocessing. Alternative methods also exist, such as removing excess mass or increasing the volume [34]. SaltProc version 0.1 does not account for mass balancing but maintains a constant thorium mass, whereas SaltProc version 0.3 has the feed rates set to maintain mass. Mass balancing is particularly important in Serpent2 due to the way masses in Serpent2 are handled.

In Serpent2, an increase in mass does not affect volume, but instead increases the density of that isotope in the material accordingly. This affects macroscopic cross section calculations, and can lead to variation in results if not accounted for, since in reality volumetric expansion could be assumed. Though there are several methods to handle mass balancing with a batch method, it not currently continuously possible in Serpent2.

In order to balance the mass in Serpent2 continuously, one approach could be to iteratively perform depletion calculations while updating feed rates until a balanced mass is found while also minimizing some other parameter, such as net mass of material added. However, the current reprocessing options available in Serpent2 only allows for pseudo-constant and decaying feed rates over the depletion step. With those two feed types available, it is not possible to have a constant mass balance.

It is possible to have the masses at the end of each depletion step remain constant, but this does handle mass fluctuations during depletion. It does, however, solve the issue of the density variation causing a difference in cross sections.

Overall, if the net mass difference is sufficiently small, it does not have to be considered since the results would not be significantly impacted. To check this for the MSBR, a simple back-of-the-envelope calculation can be used. To determine the maximum possible increase in density, the mass loss due to fission and reprocessing is neglected, and only mass addition from the thorium feed rate is considered. The uranium feed is not added because it is equivalent to the protactinium removal, so it would have a negligible impact on net mass.

$$\Delta m = \dot{m}_{feed}\Delta t \approx (2.5)(6000) = 15,000 kg \tag{3.35}$$

It can be seen in Equation (3.35) that the net mass gain for an average feed rate of 2.5 kilograms per day over 6000 days is 15,000 kg [38, 8]. This does seem to be a very large value, but the importance of the mass in the depletion calculation is primarily in how it affects cross sections, which means that the impact on the overall thorium density is important. The thorium density is 1.46 grams per cubic centimeter, and the net volume is approximately 48.71 cubic meters. The density with the added mass is calculated using Equation (3.36).

$$\rho_f = \frac{m_0 + \Delta m}{V} = \frac{(4.871E7)(1.45919E\text{-}3) + (15,000)}{4.871E7} \tag{3.36}$$

The final density comes out to be 1.767 grams per cubic centimeter, which is a percent difference of 21%. Although this is an upper bound on the mass difference, a 21% difference in the expected cross section would result in significant error in the results. Therefore, since it is possible for the mass balancing to have an impact, it should be investigated to ensure the mass balancing is not impacting the results in any unexpected manner.

3.4 Effects of Delayed Neutrons on Depletion

Delayed neutrons have a softer energy spectrum and drift along with the movement of the fuel salt in a fluid fueled molten salt reactor. This means that the delayed neutron precursors, of which some fraction leave the core and some move to less neutron important regions, have the potential to alter the depletion results of the MSBR model.

However, it has been shown by Zhou et al that for the MSBR, the delayed neutron precursor drift has a negligible impact on depletion results [46]. Although Zhou et al has shown this, it is still worth considering the maximum possible effect delayed neutron precursors could have on depletion results. In order to determine this, the Serpent2 functionality of disabling delayed neutrons is employed [26]. Using this method, two different models are generated of the MSBR. The first is one in which the delayed neutron precursors are evenly distributed within the core, which is the model implemented by SaltProc and this work [39]. The second model is one in which delayed neutrons no longer exist. This is beyond the greatest effects the delayed neutron precursor drift could have on depletion, so the absolute maximum effect of delayed neutron precursors on depletion results can be determined.

Because the delayed neutron precursors are fission products or come from the decay of fission products, depletion must be performed. For reprocessing, the average feed rate from SaltProc is used with continuous Direct Linear reprocessing, while Direct Linear reprocessing is used for fission product removal.

The impact on k_{eff} was investigated, and it was determined that the delayed neutrons from fission products add roughly 20 pcm to k_{eff} after 6,000 days of operation, which is roughly steady state operation. The masses of various isotopes, such as uranium-235 and thorium-232, were also compared, though no statistically significant difference was found. Therefore, the net impact of delayed neutrons from fission product precursors is negligible on depletion results, which agrees with the results from Zhou et al.

3.5 Serpent MSBR Model

The results generated in this work come from the MSBR, particularly the model developed by Rykhlevskii [37]. The geometry of the model has remained unchanged, while the reprocessing has been overhauled using the continuous reprocessing functionality in Serpent2 which was previously undocumented. Previously, SaltProc's batchwise reprocessing was used, but the continuous reprocessing in Serpent2 is incorporated in this work.

3.5.1 Reprocessing Structure

There are two different reprocessing schemes developed in this work. Both schemes follow the MSBR reprocessing scheme, and vary how refueling functions. The first method matches the method of Rykhlevskii, where the uranium-233 feed is treated as equivalent to the protactinium-233 removal rate. In this case, the average uranium-233 feed rate from Rykhlevskii is used over the entire simulation time [37]. A simplified overview of this scheme can be seen in Figure 3.9.

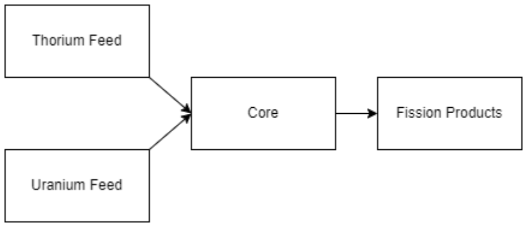

Figure 3.8: Simplified reprocessing scheme based on mimicking SaltProc.

The second reprocessing scheme is physically based, and instead creates a protactinium decay tank. From this tank, any generated uranium is continuously removed and sent back to the core, which is the intended design scheme for the MSBR [35]. A simplified version of the phyiscally realistic uranium feed can be seen in Figure 3.9.

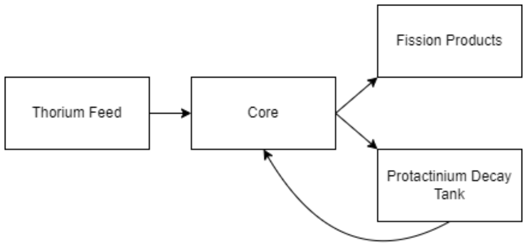

Figure 3.9: Simplified reprocessing scheme based on the phyiscal MSBR processes.

Overall, it can be anticipated that the two methods will be approximately equal at steady-state, since at that point the assumption by Rykhlevskii that the uranium input is the same as the protactinium output should be valid. However, there is expected to be a fairly large difference for the first month or two of operation, as the half life of protactinium-233 is on the order of one month. This means that the uranium feed will take some time before it is providing a steady source of fissile material to the core.

References

[1] (2005). SCALE: A Modular Code System for Performing Standardized Computer Analyses for Licensing Evaluations. vols IIII. ORNL/TM-2005/39, version 5.

[2] Ahmad, A., McClamrock, E. B., and Glaser, A. (2015). Neutronics calculations for denatured molten salt reactors: Assessing resource requirements and proliferation-risk attributes. *Annals of Nuclear Energy*, 75:261–267.

[3] Aufiero, M., Brovchenko, M., Cammi, A., Clifford, I., Geoffroy, O., Heuer, D., Laureau, A., Losa, M., Luzzi, L., Merle-Lucotte, E., Ricotti, M. E., and Rouch, H. (2014a). Calculating the effective delayed neutron fraction in the Molten Salt Fast Reactor: Analytical, deterministic and Monte Carlo approaches. *Annals of Nuclear Energy*, 65:78–90.

[4] Aufiero, M., Cammi, A., Fiorina, C., Leppänen, J., Luzzi, L., and Ricotti, M. (2013). An extended version of the SERPENT-2 code to investigate fuel burn-up and core material evolution of the Molten Salt Fast Reactor. *Journal of Nuclear Materials*, 441(1-3):473–486.

[5] Aufiero, M., Cammi, A., Geoffroy, O., Losa, M., Luzzi, L., Ricotti, M. E., and Rouch, H. (2014b). Development of an OpenFOAM model for the Molten Salt Fast Reactor transient analysis. *Chemical Engineering Science*, 111:390–401.

[6] Betzler, B. (2021). Liquid-fueled Molten Salt Reactor Depletion Modeling. Technical report, Oak Ridge National Lab. (ORNL), Oak Ridge, TN (United States).

[7] Betzler, B., Powers, J., and Brown, N. (2017a). Implementation of Molten Salt Reactor Tools in SCALE. *undefined*.

[8] Betzler, B. R., Powers, J. J., and Worrall, A. (2017b). Molten salt reactor neutronics and fuel cycle modeling and simulation with SCALE. *Annals of Nuclear Energy*, 101:489–503.

[9] Bredimas, A. and Nuttall, W. J. (2008). An international comparison of regulatory organizations and licensing procedures for new nuclear power plants. *Energy Policy*, 36(4):1344–1354.

[10] Brook, B. W., Alonso, A., Meneley, D. A., Misak, J., Blees, T., and van Erp, J. B. (2014). Why nuclear energy is sustainable and has to be part of the energy mix. *Sustainable Materials and Technologies*, 1-2:8–16.

[11] Brovchenko, M., Kloosterman, J.-L., Luzzi, L., Merle, E., Heuer, D., Laureau, A., Feynberg, O., Ignatiev, V., Aufiero, M., Cammi, A., Fiorina, C., Alcaro, F., Dulla, S., Ravetto, P., Frima, L., Lathouwers, D., and Merk, B. (2019). Neutronic benchmark of the molten salt fast reactor in the frame of the EVOL and MARS collaborative projects. *EPJ Nuclear Sciences & Technologies*, 5:2.

[12] Cervi, E., Lorenzi, S., Cammi, A., and Luzzi, L. (2019). Development of a multiphysics model for the study of fuel compressibility effects in the Molten Salt Fast Reactor. *Chemical Engineering Science*, 193:379–393.

[13] Cui, Y., Chen, J., Wu, J., Zou, C., Cui, L., He, F., and Cai, X. (2022). Development and verification of a three-dimensional spatial dynamics code for molten salt reactors. *Annals of Nuclear Energy*, 171:109040.

[14] Du, Y. and Parsons, J. E. (2009). Update on the Cost of Nuclear Power.

[15] Fei, T., Feng, B., and Heidet, F. (2020). Molten salt reactor core simulation with PROTEUS. *Annals of Nuclear Energy*, 140:107099.

[16] Fiorina, C., Cammi, A., Krepel, J., Mikityuk, K., and Ricotti, M. E. (2012). Preliminary Analysis of the MSFR Fuel Cycle Using Modified-EQL3D Procedure. In *Volume 4: Codes, Standards, Licensing, and Regulatory Issues; Fuel Cycle, Radioactive Waste Management and Decommissioning; Computational Fluid Dynamics (CFD) and Coupled Codes; Instrumentation and Co*, page 293, Anaheim, California, USA. ASME.

[17] Gauld, I. C., Radulescu, G., Ilas, G., Murphy, B. D., Williams, M. L., and Wiarda, D. (2011). Isotopic Depletion and Decay Methods and Analysis Capabilities in SCALE. *Nuclear Technology*, 174(2):169–195.

[18] Gehin, J. C. and Powers, J. J. (2016). Liquid Fuel Molten Salt Reactors for Thorium Utilization. *Nuclear Technology*, 194(2):152–161.

[19] Goluoglu, S. and McClure, J. (1998). SOFTWARE QUALIFICATION REPORT for MCNP Version 4B2.

[20] H. F. Bauman, G. W. Cunnningham III, J. L. Lucius, H. T. Kerr, and C. W. Craven, Jr. (1971). ROD: a nuclear and fuel-cycle analysis code for circulating-fuel reactors.

[21] Hombourger, B., Křepel, J., and Pautz, A. (2020). The EQL0D fuel cycle procedure and its application to the transition to equilibrium of selected molten salt reactor designs. *Annals of Nuclear Energy*, 144:107504.

[22] Ignatiev, V., Feynberg, O., Gnidoi, I., Merzlyakov, A., Smirnov, V., Surenkov, A., Tretiakov, I., Zakirov, R., Afonichkin, V., Bovet, A., Subbotin, V., Panov, A., Toropov, A., and Zherebtsov, A. (2007). Progress in Development of Li,Be,Na/F Molten Salt Actinide Recycler & Transmuter Concept. *Proceedings of ICAPP*, page 10.

[23] Jr. Vicente Valdez, P., Betzler, B., Wieselquist, W., and Fratoni, M. (2020). Modeling Molten Salt Reactor Fission Product Removal with SCALE. Technical Report ORNL/TM–2019/1418, 1608211, ORNl.

[24] Kelly, J. E. (2014). Generation IV International Forum: A decade of progress through international cooperation. *Progress in Nuclear Energy*, 77:240–246.

[25] Leppänen, J. (2007). Development of a New Monte Carlo Reactor Physics Code. *VTT Publications*, 640:241.

[26] Leppänen, J., Pusa, M., Viitanen, T., Valtavirta, V., and Kaltiaisenaho, T. (2015). The Serpent Monte Carlo code: Status, development and applications in 2013. *Annals of Nuclear Energy*, 82:142–150.

[27] Liaoyuan, H., Shaopeng, X., Jingen, C., Guimin, L., Jianhui, W., and Yang, Z. (2021). Th-U Breeding Performances in an Optimized Molten Chloride Salt Fast Reactor. *Nuclear Science and Engineering*, 195(2):185–202. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/00295639.2020.1798728.

[28] Merle-Lucotte, E., Heuer, D., Allibert, M., Ghetta, V., Brun, C. L., Brissot, R., Liatard, E., and Mathieu, L. (2007). The thorium molten salt reactor: Launching the thorium cycle while closing the current fuel cycle. page 7, Brussels, Belgium.

[29] Moir, R. W. (2002). Cost of Electricity from Molten Salt Reactors. *Nuclear Technology*, 138(1):93–95. Publisher: Taylor & Francis _eprint: https://doi.org/10.13182/NT02-A3281.

[30] Nagy, K., Kloosterman, J. L., and Lathouwers, D. (2008). Parametric studies on the fuel salt composition in thermal molten salt breeder reactors. *International Conference on the Physics of Reactors*, page 8.

[31] Nuttin, A., Heuer, D., Billebaud, A., Brissot, R., Brun, C. L., Liatard, E., Meplan, O., Merle-Lucotte, E., Nifenecker, H., and Perdu, F. (2005). POTENTIAL OF THORIUM MOLTEN SALT REACTORS : DETAILED CALCULATIONS AND CONCEPT EVOLUTION WITH A VIEW TO LARGE SCALE ENERGY PRODUCTION. *Progress in Nuclear Energy*, 46(1):23.

[32] Park, J., Jeong, Y., Lee, H. C., and Lee, D. (2015). Whole core analysis of molten salt breeder reactor with online fuel reprocessing: Whole core analysis of MSBR with online fuel reprocessing. *International Journal of Energy Research*, pages n/a–n/a.

[33] Powers, J. J., Harrison, T. J., and Gehin, J. C. (2013). A new approach for modeling and analysis of molten salt reactors using SCALE. Technical report, American Nuclear Society - ANS; La Grange Park (United States).

[34] Ridley, G. and Chvala, O. (2017). A method for predicting fuel maintenance in once-through MSRs. *Annals of Nuclear Energy*, 110:265–281.

[35] Robertson, R. (1971). CONCEPTUAL DESIGN STUDY OF A SINGLE-FLUID MOLTEN-SALT BREEDER REACTOR. Technical Report ORNL–4541, 4030941, ORNL.

[36] Rodriguez-Vieitez, E., Lowenthal, M., Greenspan, E., and Ahn, J. (2002). TRANSMUTATION CAPABILITY OF ONCE-THROUGH CRITICAL OR SUB-CRITICAL MOLTEN-SALT REACTORS. In *National Research Council*.

[37] Rykhlevskii, A. (2018). Advanced online fuel reprocessing simulation for Thorium-fueled Molten Salt Breeder Reactor. Master's thesis, University of Illinois at Urbana-Champaign, Urbana, IL.

[38] Rykhlevskii, A. (2020). *Fuel Processing Simulation Tool for Liquid-fueled Nuclear Reactors*. Doctoral Dissertation, University of Illinois at Urbana-Champaign, Urbana, IL.

[39] Rykhlevskii, A., Bae, J. W., and Huff, K. D. (2019). Modeling and simulation of online reprocessing in the thorium-fueled molten salt breeder reactor. *Annals of Nuclear Energy*, 128:366–379.

[40] Sheu, R., Chang, C., Chao, C., and Liu, Y.-W. (2013). Depletion analysis on long-term operation of the conceptual Molten Salt Actinide Recycler & Transmuter (MOSART) by using a special sequence based on SCALE6/TRITON. *Annals of Nuclear Energy*, 53:1–8.

[41] Shi, J. and Fratoni, M. (2021). Gen-foam Model and Benchmark of Delayed Neutron Precursor Drift in the Molten Salt Reactor Experiment. *EPJ Web of Conferences*, 247:06040. JC0007.

[42] Singh, V., Lish, M. R., Chvála, O., and Upadhyaya, B. R. (2017). Dynamics and control of molten-salt breeder reactor. *Nuclear Engineering and Technology*, 49(5):887–895.

[43] Singh, V., Wheeler, A. M., Upadhyaya, B. R., Chvála, O., and Greenwood, M. S. (2020). Plant-level dynamic modeling of a commercial-scale molten salt reactor system. *Nuclear Engineering and Design*, 360:110457.

[44] Wooten, D. and Powers, J. J. (2018). A Review of Molten Salt Reactor Kinetics Models. *Nuclear Science and Engineering*, 191(3):203–230.

[45] Xia, S., Chen, J., Guo, W., Cui, D., Han, J., Wu, J., and Cai, X. (2019). Development of a Molten Salt Reactor specific depletion code MODEC. *Annals of Nuclear Energy*, 124:88–97.

[46] Zhou, S., Yang, W. S., Park, T., and Wu, H. (2018). Fuel cycle analysis of molten salt reactors based on coupled neutronics and thermal-hydraulics calculations. *Annals of Nuclear Energy*, 114:369–383.

[47] Zhuang, K., Li, T., Zhang, Q., He, Q., and Zhang, T. (2020). Extended development of a Monte Carlo code OpenMC for fuel cycle simulation of molten salt reactor. *Progress in Nuclear Energy*, 118:103115.

[48] Zou, C., Yu, C., Wu, J., Cai, X., and Chen, J. (2020). Transition to thorium fuel cycle in a small modular molten salt reactor based on a batch reprocessing mode. *Annals of Nuclear Energy*, 138:107163.