Demand Driven Deployment Capabilities in Cyclus, a Fuel Cycle Simulator

Gwendolyn J. Chee[a], Roberto E. Fairhurt Agosta[a], Jin Whan Bae[b], Robert R. Flanagan[c], Anthony Scopatz[c], Kathryn D. Huff[a,*]

[a] *Dept. of Nuclear, Plasma, and Radiological Engineering, University of Illinois at Urbana-Champaign, Urbana, IL 61801*
[b] *Oak Ridge National Laboratory, Oak Ridge, TN, United States*
[c] *Nuclear Engineering Program, University of South Carolina*

Abstract

The present United States nuclear fuel cycle faces challenges that hinder the expansion of nuclear energy technology. The U.S. Department of Energy identified four nuclear fuel cycle options we could transition to, which would make nuclear energy technology more desirable. To successfully analyze the transition from our current fuel cycle to these promising fuel cycles, we need a nuclear fuel cycle simulator that can predictively and automatically deploy fuel cycle facilities to meet user-defined power demand. In this work, we developed demand-driven deployment capabilities in Cyclus, a nuclear fuel cycle simulator. User-controlled capabilities such as supply buffers, facility preferences, prediction algorithms, and installed capacity deployment were introduced to give users tools to minimize power undersupply in a transition scenario simulation. We demonstrate `d3ploy`'s capability to automatically deploy fuel cycle facilities to set up transition scenarios for promising nuclear fuel cycle options.

Keywords: nuclear fuel cycle, python, time series forecasting

---

*Corresponding Author
Email address: `kdhuff@illinois.edu` (Kathryn D. Huff)

1. Introduction

Nuclear Fuel Cycle (NFC) simulators are system analysis tools used to evaluate quantitative measures of dynamic NFC performance in both high and low resolution. Plutonium concentration in a single used fuel bundle and total electricity produced are examples of high and low resolution elements, respectively. The primary purpose of NFC simulators is to understand the dependence between various input parameters and components in the NFC and the impact their variations have on the system's performance. The results of NFC simulators are used to guide research efforts, advise future design choices, and provide decision-makers with a transparent tool for evaluating Fuel Cycle Options (FCO) to inform big-picture policy decisions [1].

Many fuel cycle simulators, automatically deploy reactor facilities to meet a user-defined power demand. However, the user must define a deployment scheme of supporting facilities to avoid gaps in the supply chain resulting in idle reactor capacity. To avoid this issue, some users choose to set infinite capacity for supporting facilities but this is an inaccurate representation of reality resulting in misrepresented results. It is straightforward to manually determine a deployment scheme for a once-through fuel cycle, however, it is not straightforward for complex closed fuel cycle scenarios. To ease setting up realistic NFC simulations, a NFC simulator should bring demand-responsive deployment decisions into the simulation logic dynamics [2]. Thus, a next-generation NFC simulator should predictively and automatically deploy fuel cycle facilities to meet a user-defined power demand.

In Cyclus, an agent-based nuclear fuel cycle simulation framework [3], each entity (i.e. `Region`, `Institution`, or `Facility`) in the fuel cycle is an agent. `Region` agents represent geographical or political areas that `Institution` and `Facility` agents reside. `Institution` agents control the deployment and decommission of `Facility` agents and represent legal operating organizations such ass utilities, governments, etc. [3]. `Facility` agents represent nuclear fuel cycle facilities such as mines, conversion facilities, reactors, reprocessing facilities, etc. Cycamore [4] provides basic Cyclus' `Region`, `Institution`, and `Facility` archetypes.

1.1. Context of Work

The impact of climate change on natural and human systems is increasingly apparent. The production and use of energy contribute to two-thirds of the total Green House Gas (GHG) emissions [5]. Furthermore, as the human population increases and previously under-developed nations urbanize rapidly, global energy demand is forecasted to increase. The types of power generation technologies used will heavily impact the effects of growing energy demand on climate change. Large scale deployment of nuclear power plants has significant potential to reduce GHG production due to their low carbon emissions [5].

However, the nuclear power industry is facing four major challenges of large scale nuclear power deployment: cost, safety, proliferation, and waste [6]. Nuclear power has high overall lifetime costs and increases risks of nuclear proliferation. There is also an unresolved long-term nuclear waste management strategy and perceived adverse safety, environmental, and health effects [6]. The nuclear power industry must overcome these four challenges to ensure continued global use and expansion of nuclear energy technology.

The four challenges described above are associated with the present once-through fuel cycle in the United States (US), in which fabricated nuclear fuel is used once and placed into storage to await disposal. An evaluation and screening study of a comprehensive set of nuclear fuel cycle options [7] was conducted to assess for performance improvements compared with the existing once-through fuel cycle (EG01) in the US across a wide range of criteria. Fuel cycles that involved continuous recycling of co-extracted U/Pu or U/TRU in fast spectrum critical reactors consistently scored high on overall performance. Table 1 describes these fuel cycles: EG23, EG24, EG29, and EG30.

The evaluation and screening study assumed the nuclear energy systems were at equilibrium to understand the end-state benefits of each Evaluation Group (EG) [8]. In this work, our goal is to model the transition from the

Fuel Cycle	Open or Closed	Fuel Type	Reactor Type
EG01 (current)	Open	Enriched-U	Thermal Critical
EG23	Closed	Recycled U/Pu + Natural-U	Fast Critical
EG24	Closed	Recycled U/TRU + Natural-U	Fast Critical
EG29	Closed	Recycled U/Pu + Natural-U	Fast Critical & Thermal Critical
EG30	Closed	Recycled U/TRU + Natural-U	Fast Critical & Thermal Critical

Table 1: Descriptions of the current and other high performing nuclear fuel cycle evaluation groups described in the evaluation and screening study [7].

initial EG01 state to these promising future end-states. To successfully analyze time-dependent transition scenarios, the NFC simulator tool must automate the transition scenario simulation setup. Therefore, the Demand-Driven Cycamore Archetypes project (NEUP-FY16-10512) was initiated to develop demand-driven deployment capabilities in Cyclus. This capability, `d3ploy`, is a Cyclus `Institution` agent that deploys facilities to meet user-defined power demand.

1.2. Novelty

We utilized time series forecasting methods to effectively predict future commodity supply and demand in `d3ploy`. Solar and wind power generation commonly use these methods to make future predictions based on past time series data [9, 10, 11, 12]. This is a novel approach that has never been applied to NFC simulators.

1.3. Objectives

The main objectives of this paper are: (1) to describe the demand-driven deployment capabilities in Cyclus, (2) to describe the prediction methods available in `d3ploy`, and (3) to demonstrate the use of `d3ploy` in setting up

EG01-23, EG01-24, EG01-29, and EG01-30 transition scenarios with various power demand curves.

2. Methodology

In Cyclus, developers have the option to design agents using C++ or Python. The `d3ploy Institution` agent was implemented in Python to enable the use of well-developed time series forecasting Python packages.

In a Cyclus NFC simulation, at every time step, `d3ploy` predicts the supply and demand of each commodity for the next time step. When there exists undersupply for any commodity, `d3ploy` deploys facilities to meet its predicted demand. Figure 1 shows the logic flow of `d3ploy` at every time step.

`d3ploy` aims to minimize the undersupply of power (Equation 1).

$$obj = min \sum_{t=1}^{t_{end}} |D_{t,power} - S_{t,power}| \tag{1}$$

The sub-objectives are (1) to minimize the number of time steps of undersupply or under-capacity of any commodity:

$$obj = min \sum_{i=c_1}^{c_M} \sum_{t=1}^{t_N} |D_{t,i} - S_{t,i}|, \tag{2}$$

(2) to minimize excessive oversupply of all commodities:

$$obj = min \sum_{i=c_1}^{c_M} \sum_{t=1}^{t_N} |S_{t,i} - D_{t,i}|. \tag{3}$$

where:

$$\begin{aligned} D &= \text{Demand} \\ S &= \text{Supply} \\ c &= \text{Commodity type} \\ M &= \text{Number of commmodities} \\ N &= \text{Number of time steps} \end{aligned}$$

Minimizing excessive oversupply reflects reality in which utilities avoid undersupply of power on the grid by ensuring power plants are never short of fuel while

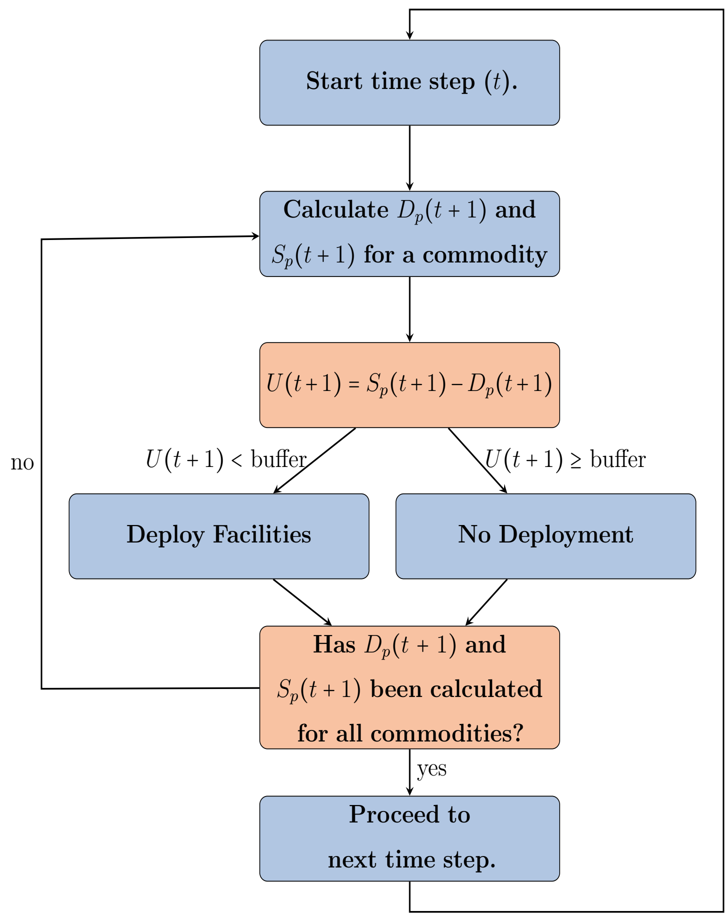


Figure 1: `d3ploy` logic flow at every time step in CYCLUS [13].

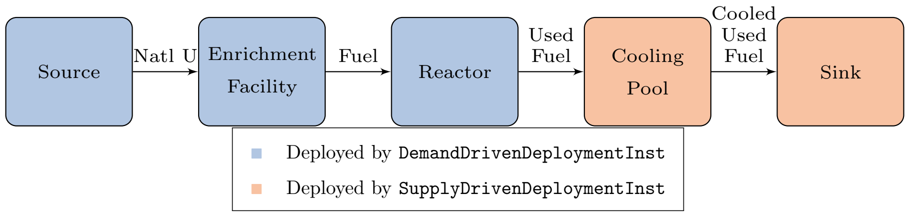


Figure 2: Simple once-through fuel cycle depicting which facilities are deployed by `DemandDrivenDeploymentInst` and `SupplyDrivenDeploymentInst`.

avoiding expensive oversupply. NFC simulators often face power undersupplies due to lack of viable fuel, despite having sufficient installed reactor capacity. Using `d3ploy` to automate the deployment of supporting facilities prevents this.

2.1. Structure

In `d3ploy`, two distinct institutions control front-end and back-end fuel cycle facilities: `DemandDrivenDeploymentInst` and `SupplyDrivenDeploymentInst`, respectively. The reason for this distinction is that front-end facilities meet the demand for commodities they produce, whereas back-end facilities meet supply for the commodities they demand. For example, when a reactor facility demands fuel, `DemandDrivenDeploymentInst` deploys fuel fabrication facilities to create fuel supply. For back-end facilities, the reactor generates spent fuel, and `SupplyDrivenDeploymentInst` deploys waste storage facilities to create capacity to store the spent fuel. Figure 2 depicts a simple once-through fuel cycle and the `Institution` type governing each facility's deployment.

2.1.1. Deployment Driving Method

The user may deploy facilities based on the difference between predicted demand and predicted supply, *or* predicted demand and installed capacity. Using installed capacity instead of predicted supply has two advantages. First, to prevent over-deployment of facilities with an intermittent supply such as reactors that require refueling. If predicted supply is selected instead of installed capacity, `d3ploy` will deploy surplus reactors during refueling downtimes to meet the temporary power undersupply. Second, to prevent infinite deployment of a facility that demands a commodity no longer available in the simulation. For example, a reprocessing plant that fabricates Sodium-Cooled Fast Reactor (SFR) fuel might demand Pu after depletion of the existing Pu inventory and decommissioning of the LWR reactors that produce it, resulting in infinite deployment of reprocessing facilities in a futile attempt to produce SFR fuel.

2.2. Input Variables

Table 2 lists and gives examples of the input variables `d3ploy` accepts. The user must do the following: define the facilities in the simulation, their respective capacities, the demand driving commodity, its demand equation, the deployment driving method, and prediction method. The user also has the option to define supply/capacity buffers for individual commodities, facility preferences, and facility fleet shares. The subsequent sections describes the buffers, facility preferences, and prediction methods.

2.2.1. Supply/Capacity Buffer

In `DemandDrivenDeploymentInst`, the user has the option to specify a supply buffer for each commodity; `d3ploy` accounts for the buffer when calculating predicted demand and deploys facilities accordingly. In `SupplyDrivenDeployment Inst`, the user has the option to specify a capacity buffer for specific commodities; d3ploy accounts for the buffer when calculating predicted supply and deploys facilities accordingly. The buffer is defined as a percentage (equation 4) or absolute value (equation 5).

$$S_{pwb} = S_p(1 + d) \tag{4}$$

$$S_{pwb} = S_p + a \tag{5}$$

	Input Parameter	Examples
Required	Demand driving commodity	Power
	Demand equation	P(t) = 10000, sin(t), 10000*t
	Facilities it controls	Fuel Fab, LWR reactor, Sink, etc.
	Capacities of the facilities	3000 kg, 1000 MW, 50000 kg
	Prediction method	Power: fast fourier transform Fuel: moving average Spent fuel: moving average
	Deployment driven by	Installed Capacity
Optional	Supply/Capacity Buffer type	Absolute
	Supply/Capacity Buffer size	Power: 3000 MW Fuel: 0 kg Spent fuel: 0 kg
	Facility preferences	LWR reactor = 100-t SFR reactor = t-100
	Fleet share percentage	MOX LWR = 85% SFR = 15%

Table 2: `d3ploy`'s required and optional input parameters with examples.

where:

$$S_{pwb} = \text{predicted supply/capacity with buffer}$$
$$S_p = \text{predicted supply/capacity without buffer}$$
$$d = \text{percentage value in decimal form}$$
$$a = \text{absolute value of the buffer}$$

For example, the user sets the power commodity's absolute supply buffer to be 2000 MW and predicted demand is 10000 MW, `d3ploy` deploys reactor facilities to meet the predicted demand and supply buffer, resulting in a power supply of:

$$\begin{aligned} S_{pwb} &= S_p + a \\ S_{pwb} &= 10000\text{MW} + 2000\text{MW} \\ &= 12000\text{MW} \end{aligned}$$

Using a combination of the buffer capability and installed capacity deployment driving method in a transition scenario simulation effectively minimizes undersupply of a commodity while avoiding excessive oversupply. This is demonstrated in section 3.1.

2.3. Facility Preference and Fleet Share

The user has the option to give preferences to facilities' that supply the same commodity. These preferences are in the form of a time-dependent equation so that the preferences can be dynamic with time. `d3ploy` uses these equations to determine which facility to deploy during a commodity shortage. In table 2, the Light Water Reactor (LWR) reactor has a preference of $100 - t$, and the Sodium-Cooled Fast Reactor (SFR) reactor has a preference of $t - 100$. At time step 1, LWR preference is 99, while SFR preference is -99; therefore a LWR is deployed if there is a commodity shortage. At time step 105, LWR preference is -5, while SFR preference is 5; therefore a SFR is deployed if there is a commodity shortage.

The user also has the option to specify percentage-share for facilities that provide the same commodity. In table 2, the mixed oxide (MOX) LWR has a share of 85%, while the SFR has a share of 15%. This constrains SFR deployment to 85% of total power demand and MOX LWR deployment to 15% of total power demand.

The year the transition begins is selected by customizing facility preferences to begin preference for advanced reactors at a certain year, and the sharing capability determines the percentage share of each type of reactor to transition to. Therefore, when `d3ploy` predicts an undersupply of a commodity it deploys facilities in order of preference, starting at the highest and moving down if the facility percentage share is already met. If a facility type does not have any preferences, `d3ploy` deploys available facilities to minimize the number of deployed facilities and oversupply of the commodity.

2.4. Prediction Methods

`d3ploy` records supply and demand values at each time step for all commodities to provide time-series data for `d3ploy`'s time series forecasting methods to predict future supply and demand for each commodity. The time series forecasting methods investigated include non-optimizing, deterministic-optimizing, and stochastic-optimizing methods. Non-optimizing methods are techniques that harness simple moving average and autoregression concepts which use historical data to infer future supply and demand values. Deterministic-optimizing and stochastic-optimizing methods are techniques that use an assortment of more sophisticated time series forecasting concepts to predict future supply and demand values. Deterministic-optimizing methods give deterministic solutions, while stochastic-optimizing methods give stochastic solutions.

Depending on the scenario in question, each forecasting method offers distinct benefits and disadvantages. The various methods are compared for each type of simulation to determine the most effective prediction method for a given scenario. The following sections describe the prediction methods.

2.4.1. Non-Optimizing Methods

Non-optimizing methods include: Moving Average (`MA`), Autoregressive Moving Average (`ARMA`), and Autoregressive Heteroskedasticity (`ARCH`). The `MA` method calculates the average of a user-defined number of previous entries in a commodity's time series and returns it as the predicted value (equation 6).

$$Predicted\ Value = \frac{V_1 + V_2 + ... + V_n}{n} \tag{6}$$

where:

$$\begin{aligned} V &= \text{Time series value} \\ n &= \text{length of timeseries} \end{aligned} \tag{7}$$

The `ARMA` method combines moving average and autoregressive models (equation 8). The first term is a constant, second term is white noise, the third term is the autoregressive model, and the fourth term is the moving average model.

The `ARMA` method is more accurate than the `MA` method because of the inclusion of the autoregressive term.

$$X_t = c + \epsilon_t + \sum_{i=1}^{p} \varphi_i X_{t-i} + \sum_{i=1}^{q} \theta_i \epsilon_{t-i} \tag{8}$$

where:

$$\begin{aligned} c &= \text{constant} \\ \varphi &= \text{parameters} \\ \epsilon_t &= \text{white noise} \\ p &= \text{equation order} \end{aligned} \tag{9}$$

The `ARCH` method modifies the original moving average term (described in equation 8). This modification makes the `ARCH` method better than the `ARMA` method for volatile time-series data [14]. The StatsModels [15] Python package is used to implement `ARMA` and `ARCH` methods in `d3ploy`.

2.4.2. Deterministic-Optimizing Methods

Deterministic methods include Fast Fourier Transform (`FFT`), Polynomial Fit (`POLY`), Exponential Smoothing (`EXP-SMOOTHING`), and Triple Exponential Smoothing (`HOLT-WINTERS`). The `FFT` method computes the discrete Fourier transform of the time series to predict future demand and supply values (equation 10). This method is implemented in `d3ploy` using the SciPy [16] Python package.

$$X_k = \sum_{n=0}^{N-1} x_n e^{-i2\pi kn/N} \tag{10}$$

where:

$$\begin{aligned} k &= 0, ..., N-1 \\ N &= \text{No. of data points} \end{aligned} \tag{11}$$

The POLY method models the time series data with a user-defined nth degree polynomial to determine future demand and supply values. This method was implemented in d3ploy using the NumPy [17] Python package. The EXP-SMOOTHING and HOLT-WINTERS methods use a weighted average of time-series data with exponentially decaying weights for older time series values [18] to create a model to determine future demand and supply values. The EXP-SMOOTHING method excels in modeling univariate time series data without trend or seasonality, whereas the HOLT-WINTERS method applies exponential smoothing three times, resulting in higher accuracy when modeling seasonal time series data. The StatsModels [15] Python package was used to implement both of these methods in d3ploy.

2.5. Stochastic-Optimizing Methods

There is one stochastic-optimizing method: step-wise seasonal method (SW-SEASONAL). The method was implemented in d3ploy by the auto Auto-Regressive Integrated Moving Averages (ARIMA) method in the pmdarima [19] Python package. The ARIMA model is a generalization of the Autoregressive Moving Average (ARMA) model to make the model fit the time series data better.

3. Results

To demonstrate d3ploy's capability conduct transition scenario analysis effectively and meet the objectives described in section 1.3, this section (1) demonstrates d3ploy's capability in simple transition scenarios, (2) compares the use of different prediction methods in EG01-EG23, EG01-EG24, EG01-EG29, and EG01-EG30 transition scenarios, and (3) demonstrates successful d3ploy setup of EG01-EG23, EG01-EG24, EG01-EG29, and EG01-EG30 transition scenarios. The input files and scripts to reproduce the results and plots in this paper are found in [20] and [21].

	Input Parameters	Simple Transition Scenario
Required	Demand driving commodity	Power
	Demand equation [MW]	$t < 40 = 1000, t \geq 40 = 1000 + 250t$
	Available facilities	`Source`, `Reactor`, `Sink`
	Prediction method	`FFT`
	Deployment driving method	Installed Capacity
Optional	Buffer type	Absolute
	Buffer size	Power: 2000MW, Fuel: 1000kg

Table 3: `d3ploy`'s input parameters for the simple transition scenario with linearly increasing power demand.

3.1. Demonstration of `d3ploy`'s capabilities

We conducted a simple transition scenario simulation with linearly increasing power demand to demonstrate `d3ploy`'s capabilities and inform input parameter choices when setting up complex many-facility transition scenarios. This simulation only includes three facility types: `source`, `reactor`, and `sink`. The simulation begins with ten `reactor` facilities (`reactor1` to `reactor10`). These reactors have staggered cycle lengths and lifetimes to prevent simultaneous refueling and setup gradual decommissioning. `d3ploy` deploys `new reactor` facilities to fill power supply gap created when the initial `reactor` facilities decommision. Table 3 shows the `d3ploy` input parameters for this simulation.

Figures 3, 4a, and 4b demonstrate `d3ploy`'s capability to deploy reactor and supporting facilities to minimize undersupply when meeting linearly increasing power demand and subsequent secondary commodities demand. In Figure 3 there exists no time steps in which the supply of power falls under demand, meeting the main objective of `d3ploy`. By using a combination of the `FFT` method for predicting demand and a power supply buffer of 2000MW (the capacity of 2 reactors), we minimized the number of undersupplied time steps for every commodity.

In figure 4a, a large-throughput source facility is initially deployed to meet

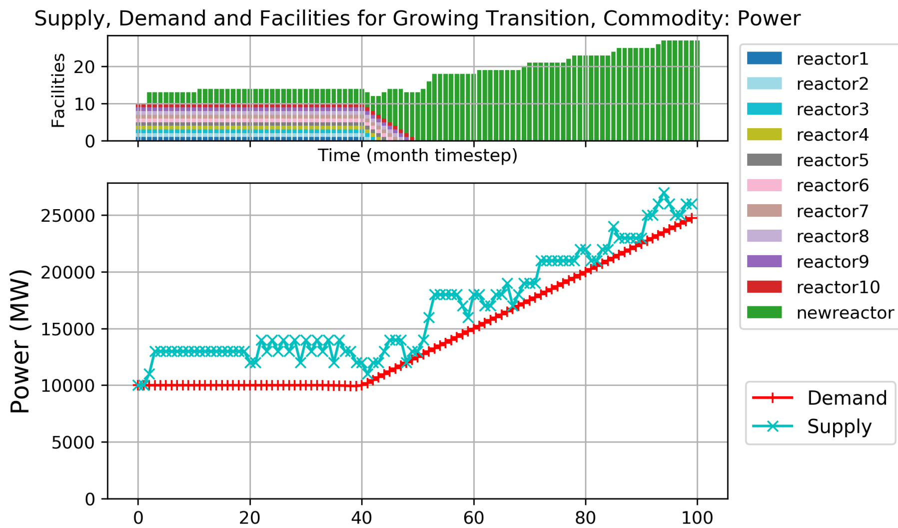
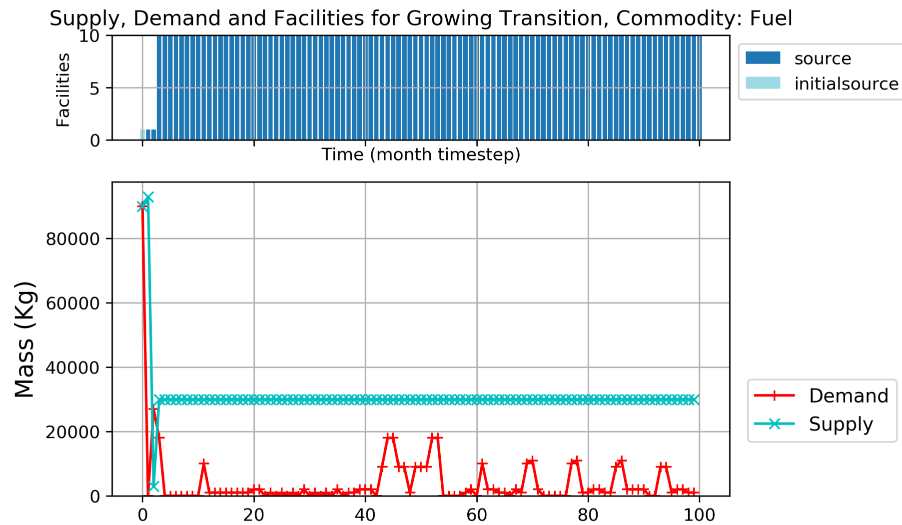


Figure 3: Power demand and supply, and reactor facility deployment plot for a simple linearly increasing power demand transition scenario with three facility types: `source`, `reactor`, and `sink`. Power demand is a user-defined equation and power is supplied by the reactors. There are no time steps with undersupply of power.

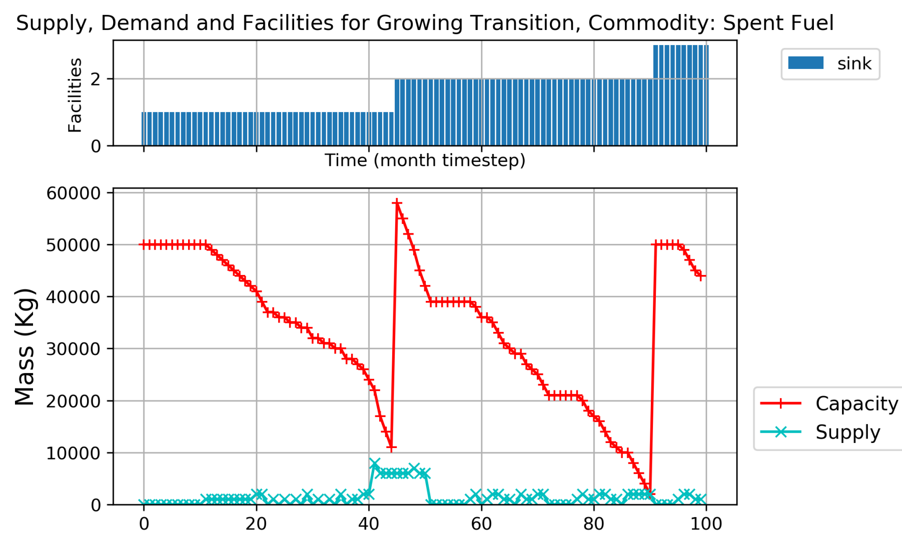
the large initial fuel demand for the commissioning of ten reactors. By having a large-throughput source facility exist for the first few time steps, `d3ploy` does not deploy supporting facilities that become redundant at later times in the simulation. This reflects reality in which reactor manufacturers accumulate an appropriate amount of fuel inventory before starting up reactors. There is one time step in which a power undersupply exists after the decommissioning of the large initial facility; this is unavoidable as the prediction methods in `d3ploy` are unable to foresee this sudden drop in demand.

3.2. Comparison of Prediction Methods

EG01-EG23, EG01-EG24, EG01-EG29, and EG01-EG30 transition scenarios are set up in Cyclus using `d3ploy`. EG01-23 and EG01-29 transition scenario simulations have a constant power demand, while EG01-24 and EG01-30 have



(a) Fuel demand and supply, and source facility deployment plot. Fuel is demanded by reactors and supplied by source facilities. There is only one time step with undersupply of fuel.



(b) Spent fuel demand and supply, and sink facility deployment plot. Spent Fuel is supplied by reactors and the capacity to store them is provided by sink facilities. There are no time steps with under-capacity of sink space.

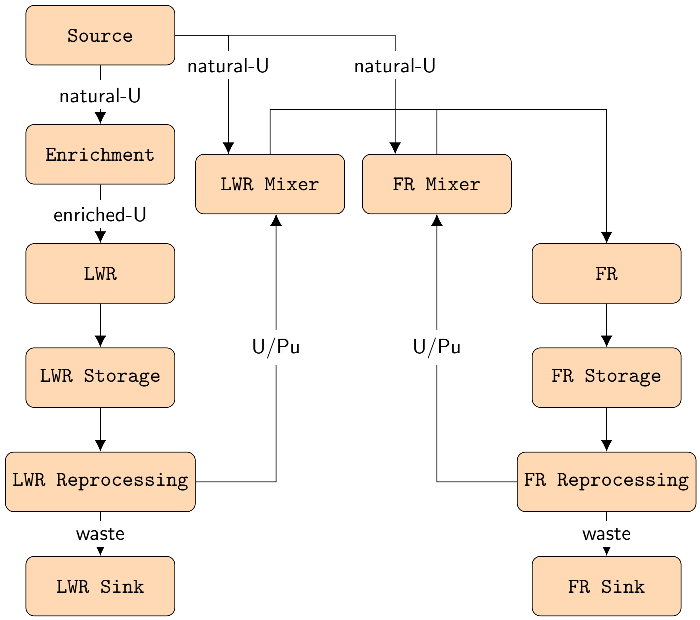
Figure 4: Simple linearly increasing power demand transition scenario with three facility types: `source`, `reactor`, and `sink`.

a linearly increasing power demand. We identified the most effective d3ploy prediction method for each scenario by comparing the results of using each prediction method in each scenario. Similar to the simple transition scenario, these transition scenario simulations begin with an initial fleet of LWRs that start progressively decommissioning at the 80-year mark, after which `d3ploy` deploys SFRs and MOX LWRs to meet the power demand. Figure 5 shows the setup of facilities and mass flows for EG01-23 and EG01-29 in Cyclus. In EG01-23 and EG01-29, recycled plutonium from LWR spent fuel produces SFR fuel. EG01-24 and EG01-30 are similar to EG01-23 and EG01-29, respectively, with the exception that all transuranic elements are recycled.

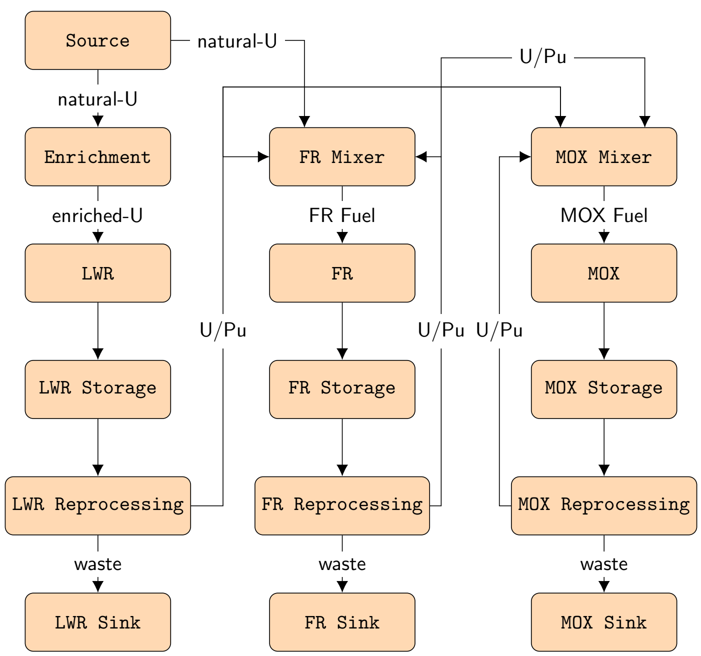
In Figure 6, each histogram represents the number of time steps with undersupply or under capacity for all commodities for each prediction method. Table 7 shows the number of time steps with power undersupply for constant power EG01-EG23 and EG01-29, linearly increasing power EG01-24 and EG01-30 transition scenarios. Figure 6 demonstrates that the `POLY` method perform the best for the EG01-23 transition scenario, with the smallest bars on the plot, indicating that they have the fewest number of time steps with undersupply and under capacity of commodities. We conducted a similar analysis for the constant power EG01-29 scenario, and as seen in Table 7, the `POLY` prediction method also performed best for minimizing undersupply of power.

In Figure 7, each histogram represents the number of time steps with undersupply or under capacity for all commodities for each prediction method. Figure 7 demonstrates that the `FFT` method perform the best for the EG01-24 transition scenario. We conducted a similar analysis for the linearly increasing power EG01-30 scenario, and as seen in Table 7, the `FFT` prediction method also performed best for minimizing undersupply of power.

From Figures 6, 7, and Table 7, we can see that the `POLY` method performs best for constant power transition scenarios, and the `FFT` method performs best for linearly increasing power transition scenarios. Undersupply and undercapacity of commodities occur during two main time periods: initial demand for the commodity and during the transition period. To further `d3ploy`'s primary



(a) EG01-EG23.



(b) EG01-EG29.

Figure 5: Facility and mass flow of the transition scenarios EG01-EG23 and EG01-EG29 in Cyclus.

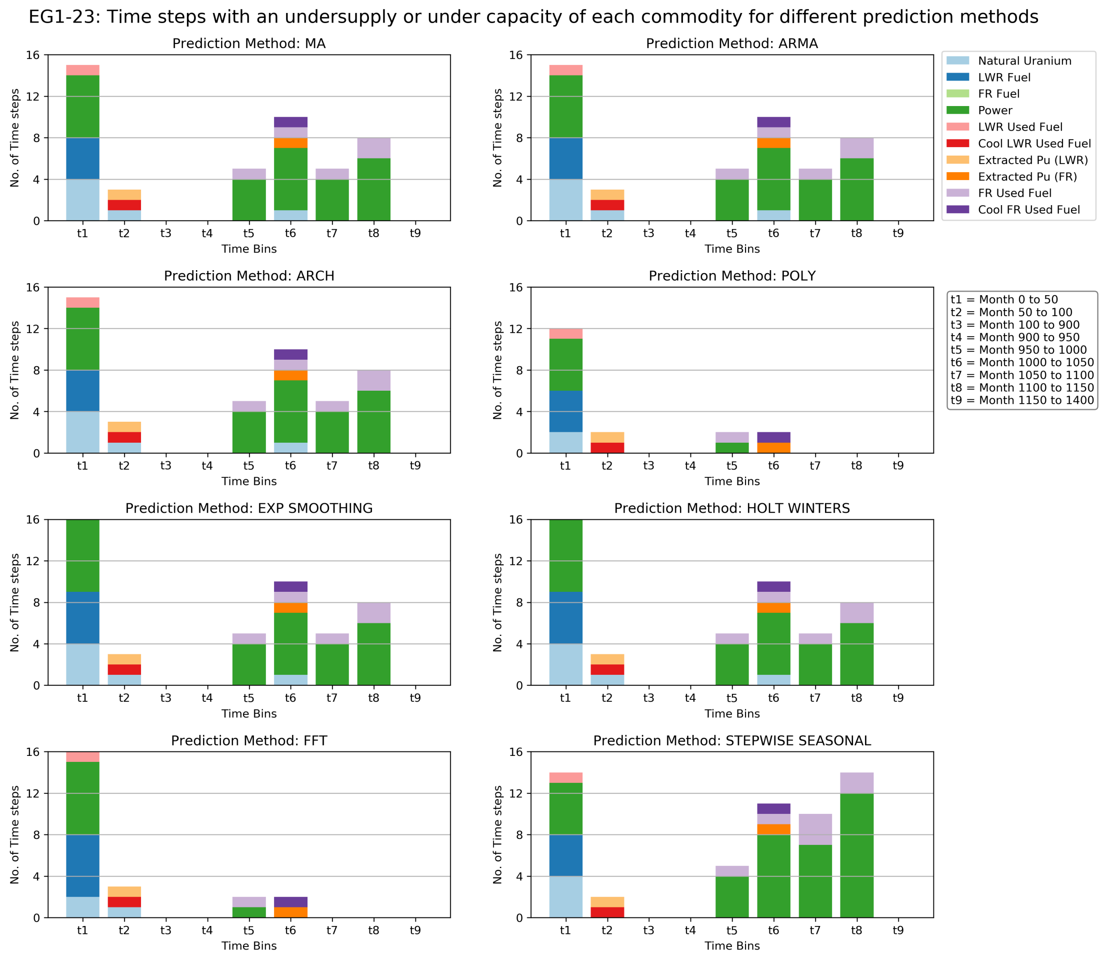


Figure 6: EG01-23 transition scenario with constant power demand. Each subplot shows the total number of time steps in which there exists undersupply and under capacity of commodities for each prediction method. The different colors represent different commodities, and each time bin refers to a specific time periods in the simulation. The `POLY` method performs the best, with the least number of time steps with undersupply and under capacity.

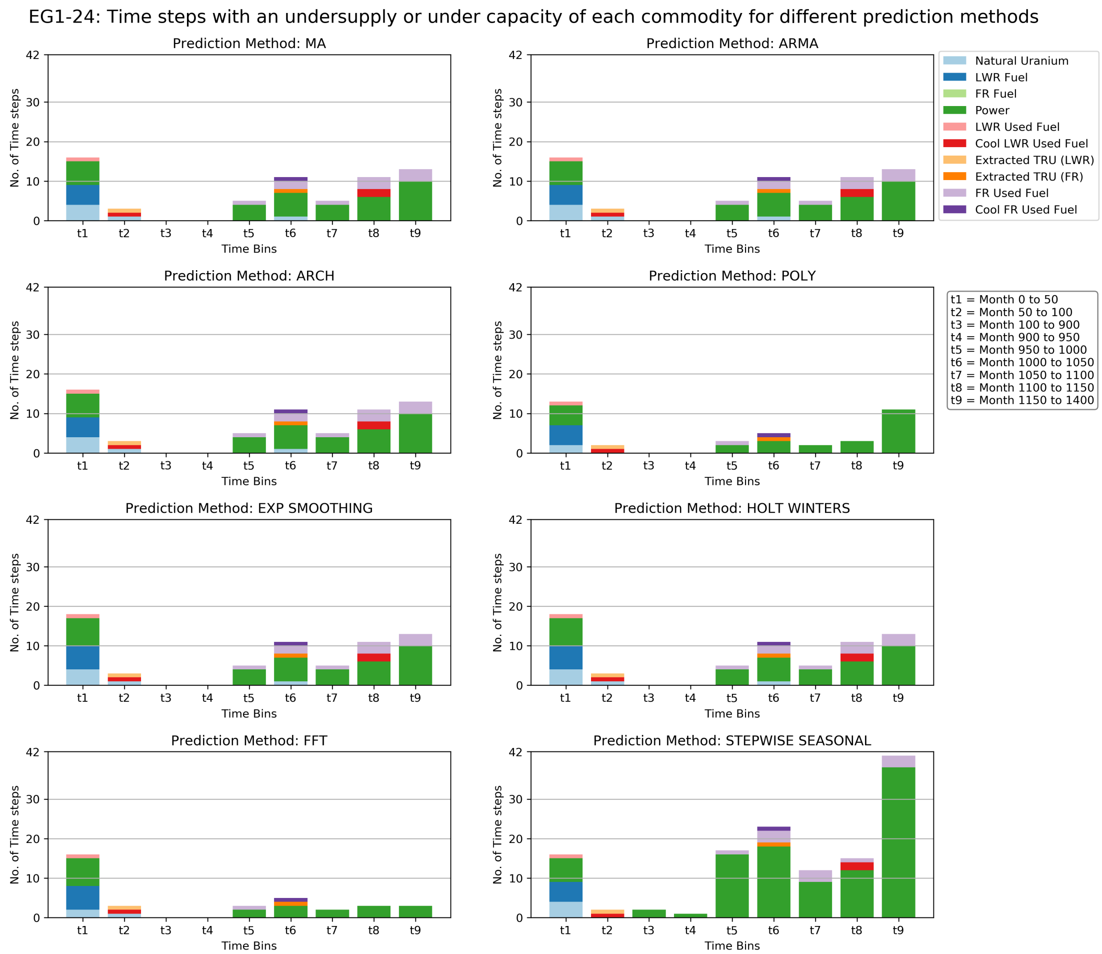


Figure 7: EG01-24 transition scenario with linearly increasing power demand. Each subplot shows the total number of time steps in which there exists undersupply and under capacity of commodities for each prediction method. The different colors represent different commodities, and each time bin refers to a specific time periods in the simulation. The `FFT` method performs the best, with the least number of time steps with undersupply and under capacity.

No. of Time Steps with Power Undersupply for Each Transition Scenario				
Algorithm	**EG01-EG23**	**EG01-EG24**	**EG01-EG29**	**EG01-EG30**
`MA`	26	36	15	24
`ARMA`	26	36	15	24
`ARCH`	26	36	15	21
`POLY`	6	65	4	9
`EXP-SMOOTHING`	27	37	16	25
`HOLT-WINTERS`	27	37	16	25
`FFT`	8	20	5	9
`SW-SEASONAL`	36	107	14	51

Table 4: Total number of time steps with undersupply of power for the EG01-EG23, EG01-24, EG01-29, EG01-30 transition scenarios for different prediction methods.

objective of minimizing the power undersupply, sensitivity analysis of the power supply buffer is conducted with the best-performing prediction method for each transition scenario.

3.3. Sensitivity Analysis

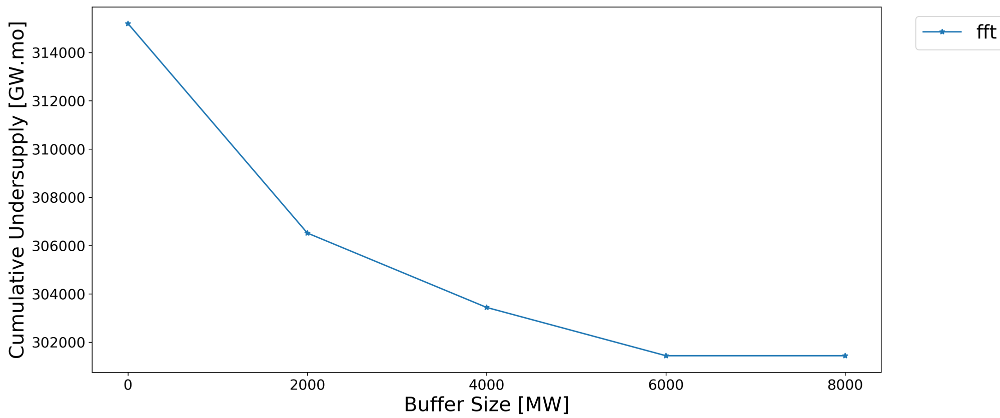
We conducted a sensitivity analysis of the power buffer size for the EG01-EG23, EG01-24, EG01-29, and EG01-30 transition scenarios. Varying the power buffer size does not impact the number of undersupply time steps for the EG01-EG23 and EG01-EG29 constant power demand transition scenarios with the `POLY` prediction method. In Table 7, there are 6 and 4 time steps in which there is power undersupply for the EG01-EG23 and EG01-29 transition scenarios, respectively. As seen in figure 6, these undersupply time steps occur at the beginning of the simulation and for one time step when the transition begins. We expected this since without time-series data at the beginning of the simulation, `d3ploy` takes a few time steps to collect time-series data about power demand to predict and start deploying reactor and supporting fuel cycle facilities. When the transition begins, power is undersupplied for one time step, following this, `d3ploy` accounts for the undersupply by deploying facilities to meet power

Buffer [MW]	Undersupply	EG01-24	EG01-30
0	Time steps [#]	20	9
	Energy $[GW \cdot mo]$	315791	152517
2000	Undersupplied [#]	9	6
	Energy $[GW \cdot mo]$	306520	147166
4000	Time steps [#]	8	6
	Energy $[GW \cdot mo]$	303438	143166
6000	Time steps [#]	7	5
	Cumulative $[GW]$	303438	139083
8000	Time steps [#]	7	5
	Energy $[GW \cdot mo]$	303438	135083

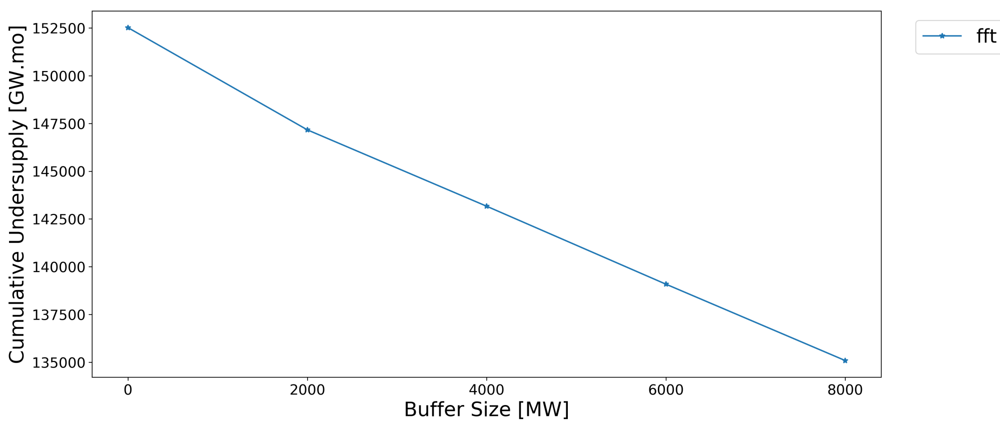
Table 5: The effect of sensitivity analysis of power buffer size on cumulative undersupply of power for EG01-EG24 and EG01-EG30 transition scenarios with linearly increasing power demand using the `FFT` prediction method.

demand. Therefore, we minimized the power undersupply for constant power EG01-EG23 and EG01-EG29 transition scenarios with a 0MW power supply buffer.

We varied the power buffer size for the EG01-24 and EG01-30 linearly increasing power demand transition scenarios. Figures 8a, 8b, and Table 5 show that increasing the buffer size decreases the number of power undersupply time steps. For EG01-24, the cumulative undersupply plateaus at 6000MW, and for EG01-30, the cumulative undersupply is smallest for a buffer size of 8000MW. These undersupply time steps occur at the beginning of the simulation and for one time step when the transition begins. We expected this since without time-series data at the beginning of the simulation, `d3ploy` takes a few time steps to collect time-series data about power demand to predict and start deploying reactor and supporting fuel cycle facilities. Therefore, a buffer of 6000MW and 8000MW minimizes the power undersupply for EG01-EG24 and EG01-EG30, respectively.



(a) EG01-24: Power buffer size vs. cumulative undersupply



(b) EG01-30: Power buffer size vs. cumulative undersupply

Figure 8: The effect of sensitivity analysis of power buffer size on cumulative undersupply of power for EG01-EG24 and EG01-EG30 transition scenarios with linearly increasing power demand using the `FFT` prediction method.

3.4. Best Performance Models

Table 6 shows the `d3ploy` input parameters for EG01-EG23, EG01-EG24, EG01-EG29, and EG01-EG30 transition scenarios that minimize the undersupply of power and undersupply and under-capacity of the other commodities in the simulation. The need for commodity supply buffers is a reflection of reality in which a supply buffer is usually maintained to ensure continuity in the event of an unexpected failure in the supply chain.

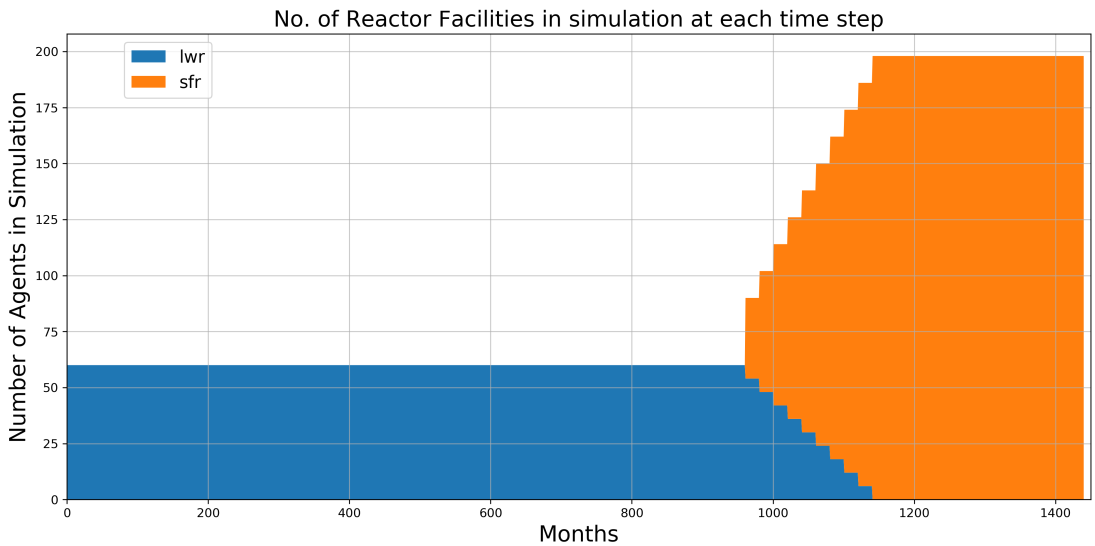
Figures 9 and 10 show time-dependent deployment of reactor and supporting facilities for the EG01-23 constant power demand and EG01-30 linearly increasing power demand transition scenarios, respectively. `d3ploy` automatically deploys reactor and supporting facilities to set up a supply chain to meet power demand during a transition from LWRs to SFRs for EG01-23, and from LWRs to MOX LWRs and SFRs for EG01-30. EG01-24 and EG01-29 facility deployment plots are similar to EG01-23 and EG01-30, respectively, therefore they are not shown.

Input Parameter	Simulation Description			
	EG01-23	**EG01-24**	**EG01-29**	**EG01-30**
Demand Driving Commodity	Power	Power	Power	Power
Demand Equation [MW]	60000	60000 +250t/12	60000	60000 +250t/12
Prediction Method	`POLY`	`FFT`	`POLY`	`FFT`
Deployment Driving Method	Installed Capacity	Installed Capacity	Installed Capacity	Installed Capacity
Fleet Share Percentage	MOX: 85% SFR: 15%	MOX: 85% SFR: 15%	MOX: 85% SFR: 15%	MOX :85% SFR: 15%
Buffer type	Absolute			
Power Buffer Size [MW]	0	6000	0	8000

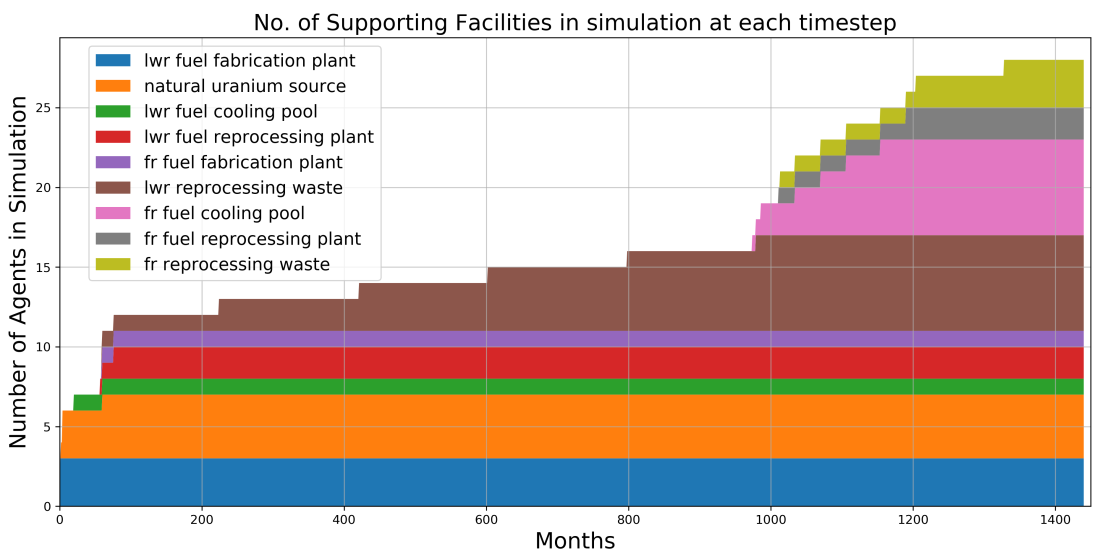
Table 6: `d3ploy`'s input parameters for EG01-EG23, EG01-EG24, EG01-EG29, and EG01-EG30 transition scenarios that minimizes undersupply of power and minimizes the undersupply and under-capacity of the other facilities.

	No. of Time Steps with Undersupply			
Transition Scenario	EG01-EG23	EG01-EG24	EG01-EG29	EG01-EG30
Commodities				
Natural Uranium	2	3	1	1
LWR Fuel	4	6	1	2
SFR Fuel	0	0	2	2
MOX LWR Fuel	-	-	2	2
Power	6	7	4	5
LWR Spent Fuel	1	1	1	1
SFR Spent Fuel	1	1	1	1
MOX LWR Spent Fuel	-	-	1	1

Table 7: Undersupply/capacity of key commodities for the best performing EG01-EG23,24,29,30 transition scenarios.

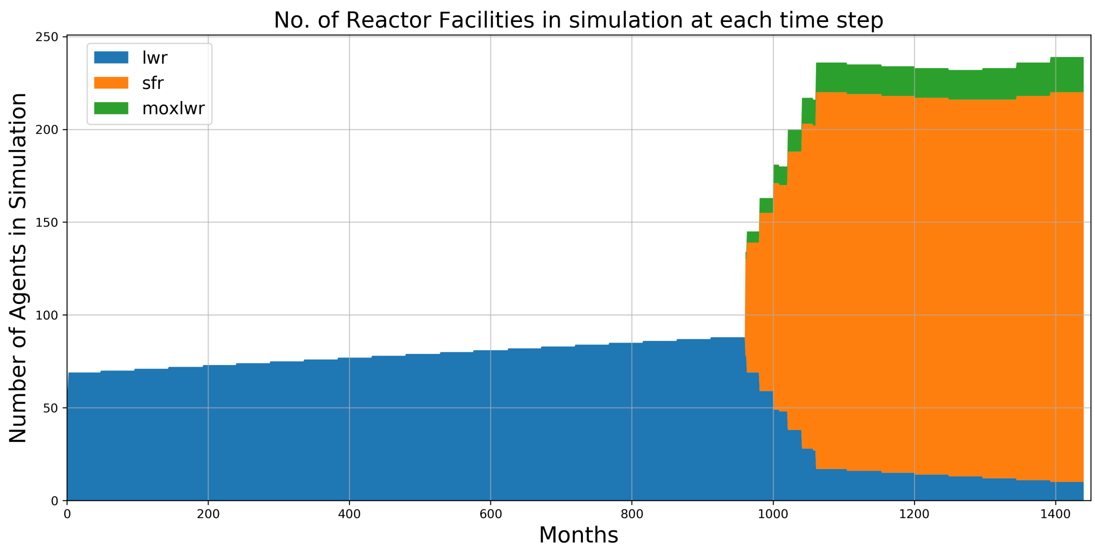


(a) EG01-23: Reactor Deployment

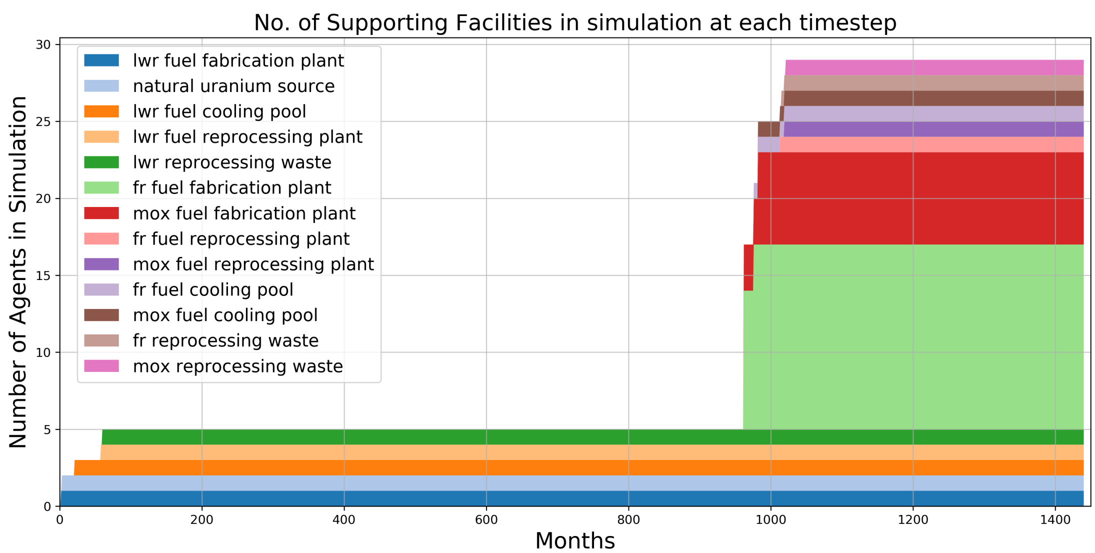


(b) EG01-23: Supporting Facility Deployment

Figure 9: Time dependent deployment of reactor and supporting facilities in the EG01-23 constant power demand transition scenario. `d3ploy` automatically deploys reactor and supporting facilities to setup a supply chain to meet constant power demand of 60000 MW during a transition from LWRs to SFRs.



(a) EG01-30: Reactor Deployment



(b) EG01-30: Supporting Facility Deployment

Figure 10: Time dependent deployment of reactor and supporting facilities in the EG01-30 linearly increasing power demand transition scenario. `d3ploy` automatically deploys reactor and supporting facilities to setup a supply chain to meet linearly increasing power demand of $60000 + 250t/12$ MW during a transition from LWRs to MOX LWRs and SFRs.

4. Conclusion

In this paper, we demonstrate that with careful selection of `d3ploy` parameters, we can effectively automate the setup of constant and linearly increasing power demand transition scenarios for EG01-23, EG01-24, EG01-29, and EG01-30 with minimal power undersupply. Using `d3ploy` to set up transition scenarios is more efficient than the previous efforts that required a user to manually calculate and use trial and error to set up the deployment scheme for the supporting fuel cycle facilities. Transition scenario simulations set up this way are sensitive to changes in the input parameters resulting in an arduous setup process since a slight change in one input parameter would result in the need to recalculate the deployment scheme to ensure no undersupply of power. Therefore, by automating this process, when the user varies input parameters in the simulation, `d3ploy` automatically adjusts the deployment scheme to meet the new constraints.

5. Future Work

We simulate transition scenarios to predict the future; however, when implemented in the real world, the transition scenario tend to deviate from the optimal scenario. Therefore, NFC simulators must be used to conduct sensitivity analysis studies to understand the subtleties of a transition scenario better to reliably inform policy decisions. Previously it was difficult to conduct sensitivity analysis with Cyclus as users have to manually calculate the deployment scheme for a single change in an input parameter. By using the `d3ploy` capability, sensitivity analysis studies are more efficiently conducted to determine how variation in different input parameters impact the progress and final state of a transition scenario.

6. Acknowledgments

Department of Energy (DOE) Office of Nuclear Energy funds this research through the Nuclear Energy University Program (Project 16-10512, DE-NE0008567)

'Demand-Driven Cycamore Archetypes'. The authors want to thank members of the Advanced Reactors and Fuel Cycles (ARFC) group at the University of Illinois at Urbana-Champaign. Special thanks to Kip Kleimenhagen for his excellent proofreading help. We also thank our colleagues from the CYCLUS community for collaborative CYCLUS development.

References

[1] A. M. Yacout, J. J. Jacobson, G. E. Matthern, S. J. Piet, A. Moisseytsev, Modeling the Nuclear Fuel Cycle, in: The 23rd International Conference of the System Dynamics Society," Boston, Citeseer, 2005.
URL `http://www.inl.gov/technicalpublications/Documents/3169906.pdf`

[2] K. D. Huff, J. W. Bae, K. A. Mummah, R. R. Flanagan, A. M. Scopatz, Current Status of Predictive Transition Capability in Fuel Cycle Simulation, in: Proceedings of Global 2017, American Nuclear Society, Seoul, South Korea, 2017, p. 11.

[3] K. D. Huff, M. J. Gidden, R. W. Carlsen, R. R. Flanagan, M. B. McGarry, A. C. Opotowsky, E. A. Schneider, A. M. Scopatz, P. P. H. Wilson, Fundamental concepts in the Cyclus nuclear fuel cycle simulation framework, Advances in Engineering Software 94 (2016) 46–59, arXiv: 1509.03604. `doi:10.1016/j.advengsoft.2016.01.014`.
URL `http://www.sciencedirect.com/science/article/pii/S0965997816300229`

[4] R. W. Carlsen, M. Gidden, K. Huff, A. C. Opotowsky, O. Rakhimov, A. M. Scopatz, P. Wilson, Cycamore v1.0.0, Figshare-Http://figshare.com/articles/Cycamore_v1_0_0/1041829. `doi:http://figshare.com/articles/Cycamore_v1_0_0/1041829`.
URL `http://figshare.com/articles/Cycamore_v1_0_0/1041829`

[5] Climate Change and Nuclear Power 2018, Non-serial Publications, INTERNATIONAL ATOMIC ENERGY AGENCY, Vienna, 2018.
URL https://www.iaea.org/publications/13395/climate-change-and-nuclear-power-2018

[6] Massachusetts Institute of Technology, The Future of nuclear power: an interdisciplinary MIT Study., MIT, Boston MA, 2003, oCLC: 53208528.

[7] R. Wigeland, T. Taiwo, H. Ludewig, M. Todosow, W. Halsey, J. Gehin, R. Jubin, J. Buelt, S. Stockinger, K. Jenni, B. Oakley, Nuclear Fuel Cycle Evaluation and Screening - Final Report, US Department of Energy (2014) 51.
URL https://fuelcycleevaluation.inl.gov/Shared%20Documents/ES%20Main%20Report.pdf

[8] B. Feng, B. Dixon, E. Sunny, A. Cuadra, J. Jacobson, N. R. Brown, J. Powers, A. Worrall, S. Passerini, R. Gregg, Standardized verification of fuel cycle modeling, Annals of Nuclear Energy 94 (2016) 300–312. doi:10.1016/j.anucene.2016.03.002.
URL http://www.sciencedirect.com/science/article/pii/S0306454916301098

[9] G. Reikard, Predicting solar radiation at high resolutions: A comparison of time series forecasts, Solar Energy 83 (3) (2009) 342–349.

[10] M. Diagne, M. David, P. Lauret, J. Boland, N. Schmutz, Review of solar irradiance forecasting methods and a proposition for small-scale insular grids, Renewable and Sustainable Energy Reviews 27 (2013) 65–76.

[11] S. S. Soman, H. Zareipour, O. Malik, P. Mandal, A review of wind power and wind speed forecasting methods with different time horizons, in: North American Power Symposium 2010, IEEE, 2010, pp. 1–8.

[12] J. W. Taylor, P. E. McSharry, R. Buizza, Wind power density forecasting

using ensemble predictions and time series models, IEEE Transactions on Energy Conversion 24 (3) (2009) 775–782.

[13] G. Chee, J. W. Bae, K. D. Huff, R. R. Flanagan, R. Fairhurst, Demonstration of Demand-Driven Deployment Capabilities in Cyclus, in: Proceedings of Global/Top Fuel 2019, American Nuclear Society, Seattle, WA, United States, 2019.

[14] R. R. Flanagan, J. W. Bae, K. D. Huff, G. J. Chee, R. Fairhurst, Methods for Automated Fuel Cycle Facility Deployment, in: Proceedings of Global/Top Fuel 2019, American Nuclear Society, Seattle, WA, United States, 2019.

[15] GitHub Community, StatsModels: Statistics in Python Package (2019). URL `https://www.statsmodels.org/stable/faq.html`

[16] E. Jones, T. Oliphant, P. Peterson, SciPy: Open source scientific tools for Python, 2001, 2016.

[17] N. Developers, NumPy, NumPy Numpy. Scipy Developers.

[18] R. J. Hyndman, G. Athanasopoulos, Forecasting: principles and practice, OTexts, 2018.

[19] pmdarima: ARIMA estimators for Python (2019). URL `https://www.alkaline-ml.com/pmdarima/`

[20] arfc/d3ploy: A collection of Cyclus manager archetypes for demand driven deployment, 10.5281/zenodo.3464123 (Sep. 2019). URL `https://github.com/arfc/d3ploy`

[21] G. Chee, G. T. Park, K. Huff, arfc/transition-scenarios : Validation of Spent Nuclear Fuel Output by Cyclus, a Fuel Cycle Simulator Code (Aug. 2018). `doi:10.5281/zenodo.1401495`.