Numerical Experiments for Verifying Demand Driven Deployment Algorithms

Deterministic-Optimizing Algorithm

Jin Whan Bae, Gwendolyn Chee, Kathryn Huff

May 7, 2018

1 Introduction

For many fuel cycle simulations, it is currently up to the user to define a deploy scheme, or facility parameters, to make sure that there's no gap in the supply chain. Or, the same goal is achieved by setting the supply `facility`'s capacity to infinity, which does not reflect real-world conditions.

The Demand-Driven Cycamore Archetype project (NEUP-FY16-10512) aims to develop demand-driven deployment capabilities in Cyclus. To understand the terms used in this report, please read [1].

The developed algorithm, in the form of Cyclus `Institution` agent, deploys `Facilities` to meet the front-end and back-end demands of the fuel cycle.

This report describes numerical experiments for the deterministic -optimizing algorithm.

These prediction models are being developed by the University of South Carolina. The numerical experiments will be designed for both the once through nuclear fuel cycle and advanced fuel cycles.

2 Method

This report lists necessary capabilities of the new Cyclus `institute` for demand-driven deployment of fuel cycle facilities. Then the report lists tests to check correct implementation of the capabilities, with a sample fuel cycle with well-defined facility parameters.

3 Configuration

The user defines a demand equation of a commodity (e.g. power, spent fuel, plutonium etc.) and a supply chain that results in the creation of the demanded commodity. For example, a user would define a power demand equation, and list the facilities that lead up to power production, as such:

```xml
<!-- Definition of demand commodity and demand equation -->
<demand_commodity> POWER </demand_commodity>
<piecewise_function>
        <piece>
                <start>0</start>
                <function>
                        <type>linear</type>
                        <params> 1 2 </params>
                </function>
        </piece>
        <piece>
                <start>5</start>
                <function>
                        <type>linear</type>
                        <params> -2 7 </params>
                </function>
        </piece>
</piecewise_function>

<!-- Definition of supply chain leading to power production -->
<supply_chain>
        <val> source </val>
        <val> enrichment </val>
        <val> reactor </val>
</supply_chain>

<!-- Added definition of supply and demand commodities -->
```

Note that the user input is expansive and detailed due to the independent, agent-based nature of CYCLUS. The user-defined parameters will allow the institution create a supply chain, demand timeseries for each commodity, and deployment timeseries, which will be explained in the next section.

4 Algorithm Flow

The algorithm is an `Institution`, which is a CYCLUS agent type that deploys and decommissions facilities.

Upon entering, the CYCLUS `Institution` accesses the parameters for each facility in the chain to extract the supply and demand parameters (capacity, throughput). The algorithm then creates a matrix of commodity demand quantities for every facility of the supply chain. It does so by first creating a vector of the final demand (e.g. power) and back-calculating the demand for other commodities in order to meet power production demand. The matrix has a size of `[Timestep X Length of Chain]`. The demand quantity matrix is then calculated into a deployment matrix that lists the number of facilities to be deployed at a given timestep.

4.1 Specifics

4.2 Time Step Execution in Cyclus

As a reference, the time step execution for CYCLUS is illustrated in figure 1.

At `Tock`, the algorithm calculates the demand and supply for the next timestep, and calculates the difference. If the difference is bigger than the capacity of one facility, it schedules to deploy a new facility, or a decommissioning of an existing facility. The decommissioning takes place in the `Decommission` phase of CYCLUS, and the construction takes place in the `Build` phase of CYCLUS. Thus, all the adjustment of facilities occur prior to the next dynamic resource exchange (DRE).

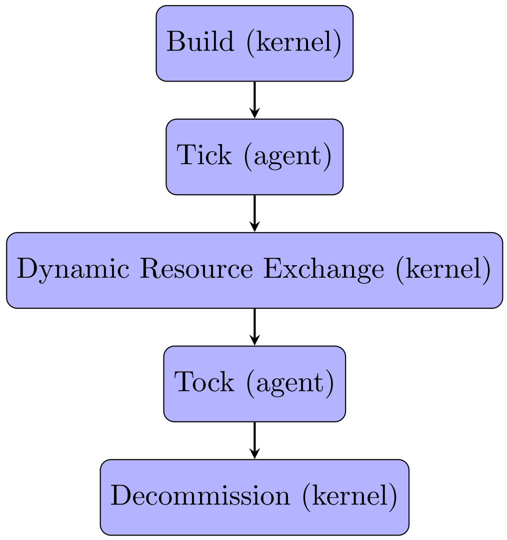

Figure 1: Each timestep in CYCLUS follows the five steps in order. Processes labeled kernel are executed by the CYCLUS framework, whereas processes labeled agent are executed by individual agents. What happens in the 'Tick' and 'Tock' is thus unique to each archetype.

5 Simulation parameter for Test Scenarios

Simple parameters are given to fuel cycle facilities for the numerical testing of the algorithm. Only `source` and `reactor` facilities are used in the test scenarios.

Table 1 provides basic parameters for each test scenario. Table 2 provides the parameters for the `source`, `reactors` and `sink` in the test scenarios.

Table 1: Basic Test Parameters

Test Scenario Parameters	**Value**	**Units**
Duration	15	timesteps
Timestep	1	month
Start Month	1	month
Start Year	2000	year

Table 2: Source, Reactor and Sink Parameters

Source Parameters	**Value**	**Description**
Throughput	1	Quantity of output commodity the facility can output in o
Output Commodity	fuel	The name of the output commodity
Lifetime	7	Lifetime of the facility
Wait time	3	Wait time for the facility to decommission when no longer
Reactor Parameters	**Value**	**Description**
Input Commodity	fuel	Name of input commodity
Output Commodity	spent_uox	Name of output commodity
Output Power Capacity	1	MWe of power generated by reactor per operational timest
Cycle Time	1	Duration of a full operational cycle of reactor
Refuel Time	0	Duration of a full refueling period
Lifetime	7	Lifetime of the facility
Wait time	3	Wait time for the facility to decommission when no longer
Assembly Size	1	Quantity of commodity per assembly
Num. assemblies per batch	1	Number of assemblies discharged and refueled per cycle
Num. assemblies per core	3	Number of assemblies that make up the entire reactor core

6 Numerical Tests for the Deterministic optimizing prediction method

The deterministic optimizing prediction method is tested by comparing its output for various scenarios against their analytical solutions . In this section, the tests that must be met is described based on the parameters defined in table 1 and 2 and analytical solution of a defined simple scenario. Unit test examples are included in Appendix B.

The tests are split into test A types and test B types. Test A refers to the test scenarios where there are only `source` facilities. Test B refers to the test scenarios where there are both `source` and `reactor` facilities. Each test type is sub-categorized into type 1,2 and 3. Test A1 and B1 refer to test scenarios where facilities are expected to be deployed. Tests A2 and B2 refers to the test scenarios where facilities are expected to be decommissioned. Tests A3 and B3 refers to test scenarios where facilities are expected to be deployed and decommissioned. In all scenarios the ultimate demand commodity is `Power`. Figure 2 and 3 show the commodity flow for test type A and B respectively.

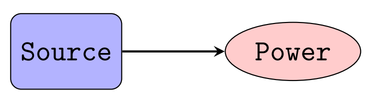

Figure 2: Demand flow of Test A type where only source facilities are present. Blue blocks refer to a facility type and red ellipses refer to a commodity type.

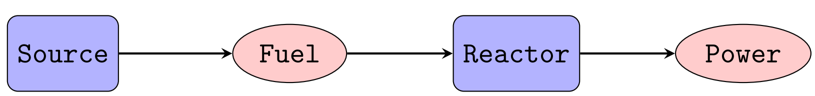

Figure 3: Demand flow of Test B type where source and reactor facilities are present. Blue blocks refer to a facility type and red ellipses refer to a commodity type.

The tests listed below are unique to the deterministic optimizing algorithm. The tests written in the non-optimizing algorithm report will also be used to test the deterministic optimizing algorithm but are not listed in this report.

The prediction algorithm for the deterministic optimizing method has three user-defined input parameters for the scenario. The aim of the various test scenarios are to check if the deterministic optimizing method archetype will deploy or decommission facilities correctly when there is a variation in the combination of these input parameters. The input parameters are listed as an example in section 3.

1. End demand commodity
2. Demand equation
3. Supply chain of facilities

For each test scenario, there is one table that states the test scenario's input parameters and another table that states the exact analytical solution.

Additionally, we created base tests that pass when the supply meets the demand within a given facility number tolerance. In other words, when the supply exceeds the demand by the specified tolerance quantity, the test still passes. For this numerical experiment, the tolerance is set to one facility.

6.1 Test A-dep-1

The goal of test A-dep-1 is to determine is a source facility is deployed when the lifetime of another is ending to meet the upcoming loss of power supply. Table 3 shows the input parameters of the `Institution` in the test scenario. Table 4 shows the expected analytical solution based on the test scenario. Table **??** shows the accepted range of total number of facilities deployed and decommissioned over the test scenario which will pass the base test, which factors in the facility over or under prediction tolerance of 1.

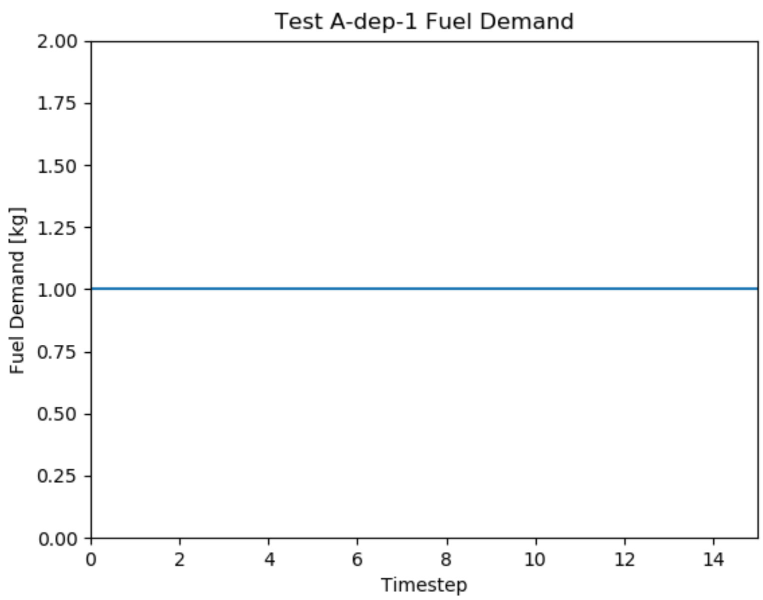

Figure 4: Fuel demand for test A-dep-1.

Table 3: Test A-dep-1 Scenario Input Parameters

Input Parameter	**Value**	**Units**
End demand commodity	fuel	kg
Demand Equation	D = 1	
Supply Chain	source $\rightarrow$ `End Demand Commodity`	

Table 4: Test A-dep-1 Analytical Solution (If the time step is skipped over, it is because there are no facilities deployed or decommissioned during that time step.)

Time Step	**No. of Source Facilities Deployed**	**No. of Source Exits**
1	1	0
7	0	1
8	1	0
14	0	1
15	1	0

Table 5: Test A-dep-1 Base Test Acceptance

Acceptable total No. of Source Facilities Deployed + tolerance
3 < x < 4

6.2 Test A-dep-2

The goal of test A-dep-2 is to determine whether a source facility is deployed to meet the upcoming loss of supply due to the end of a lifetime of another. The difference between this test and test A-dep-1 is that multiple facilities exit and are deployed at different times. Table **??** shows the input parameters of the `Institution`. Table 7 shows the expected analytical solution based on the test scenario. Table **??** shows the accepted range of total number of facilities deployed and decommissioned over the test scenario which will pass the base test, which factors in the facility over or under prediction tolerance of 1.

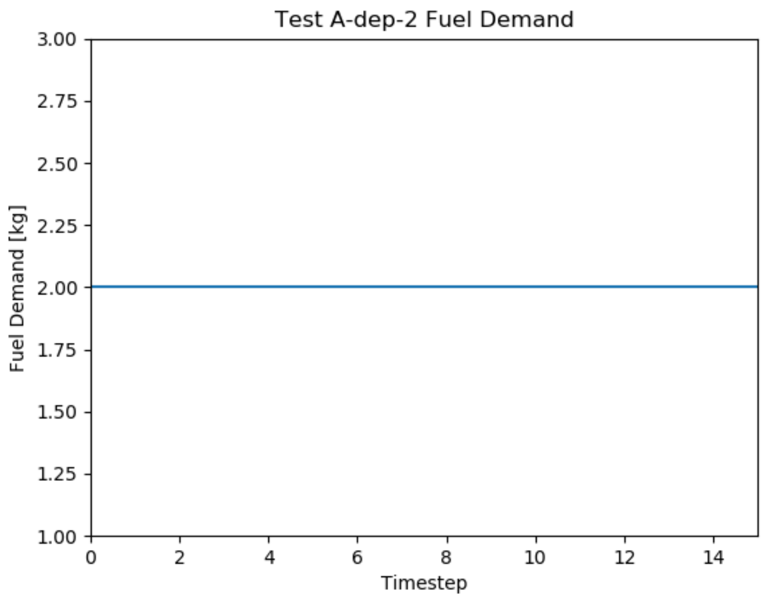

Figure 5: Fuel demand for test A-dep-2.

Table 6: Test A-dep-2 Scenario Input Parameters

Input Parameter	**Value**	**Units**
End demand commodity	fuel	kg
Demand Equation	D = 2	
Supply Chain	source $\rightarrow$ `End Demand Commodity`	

s

Table 7: Test A-dep-2 Analytical Solution (If the time step is skipped over, it is because there are no facilities deployed or decommissioned during that time step.)

Time Step	No. of Source Facilities Deployed	No. of Source Facilities Decommissioned
0	1	0
1	1	0
6	0	1
7	1	1
8	1	0
13	0	1
14	1	1
12	1	0

Table 8: Test A-dep-2 Base Test Acceptance

Acceptable total No. of Source Facilities Deployed + tolerance
6 < x < 7

6.3 Test A-mix-1

The goal of test A-mix-1 is to determine if a source facility is not decommissioned until after the wait time (defined in Table 2) even if there is over-supply of fuel commodity. Figure 6 shows the plot of the fuel demand for test A-mix-1. Table **??** shows the input parameters of the `Institution` in the test scenario. Table 10 shows the expected analytical solution based on the test scenario.

Table 9: Test A-mix-1 Scenario Input Parameters

Input Parameter	Value	Units
End demand commodity	fuel	kg
Demand Equation	$D = \begin{cases} 1, & 0 < t < 2 \\ 0, & 2 < t \end{cases}$	
Supply Chain	source $\rightarrow$ `End Demand Commodity`	

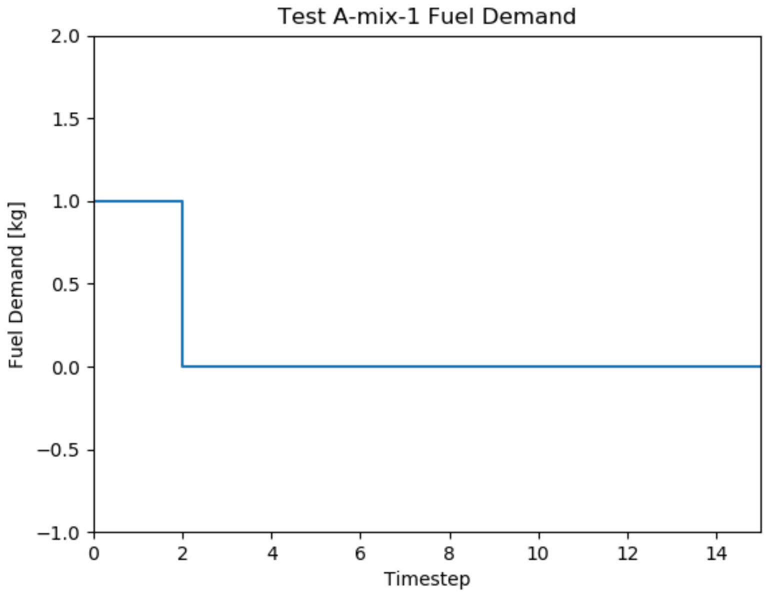

Figure 6: Fuel demand for test A-mix-1.

Table 10: Test A-mix-1 Analytical Solution (If the time step is skipped over, it is because there are no facilities deployed or decommissioned during that time step.)

Time Step	**No. of Source Facilities Deployed**	**No. of Source Facilities Decommissioned**
1	1	0
5	0	1

6.4 Test B-dep-1

The goal of test B-dep-1 is to determine if source and reactor facilities are deployed when the lifetime of another is ending. Figure 7 shows the plot of the power demand for test B-dep-1. Table **??** shows the input parameters of the `Institution` in the test scenario. Figure 7 shows the plot of the fuel demand for test B-dep-1. Table 12 shows the expected analytical solution based on the test scenario. Table **??** shows the accepted range of total number of facilities deployed over the test scenario which will pass the base test, which factors in the facility over prediction tolerance of 1.

Table 11: Test B-dep-1 Scenario Input Parameters

Input Parameter	**Value**	**Units**
End demand commodity	fuel	kg
Demand Equation	D = 1	
Supply Chain	source → `End Demand Commodity`	

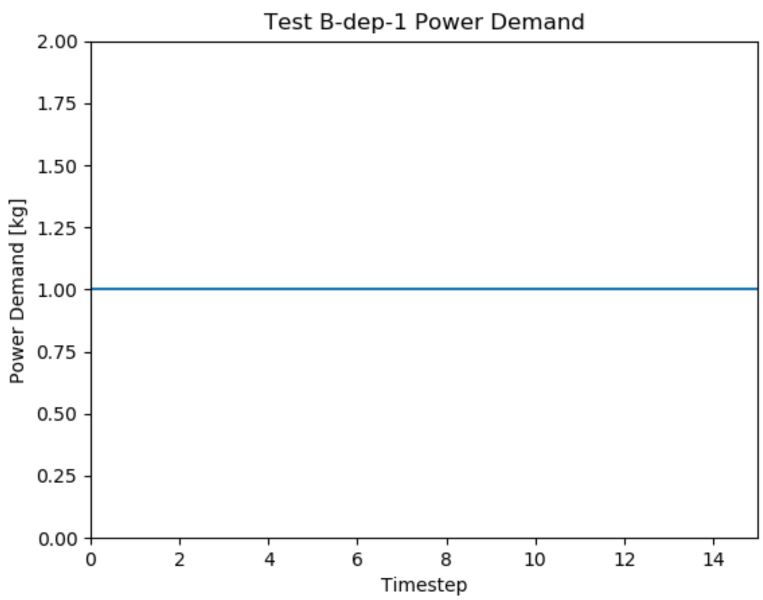

Figure 7: Fuel demand for test B-dep-1.

Table 12: Test B-dep-1 Analytical Solution

Time Step	**No. of Source Deployed**	**No. of Reactor Deployed**	**No. of Sources Exit**	**No. of Reactors Exit**
1	3	1	0	0
7	0	0	3	1
8	3	1	0	0
14	0	0	3	1
15	3	1	0	0

Table 13: Test B-dep-1Base Test Acceptance

Acceptable total No. of Source Facilities Deployed + tolerance	**Acceptable total No. of Reactor Facilities Deployed + tolerance**
$9 < x < 10$	$3 < x < 4$

7 Numerical Test Results

8 References

References

[1] K. D. Huff, M. J. Gidden, R. W. Carlsen, R. R. Flanagan, M. B. McGarry, A. C. Opotowsky, E. A. Schneider, A. M. Scopatz, and P. P. H. Wilson. Fundamental concepts in the Cyclus nuclear fuel cycle simulation framework. *Advances in Engineering Software*, 94:46–59, Apr. 2016.

Appendix A - parameter configuration

Appendix B - Sample Test Code

Appendix C - Numerical Experiment Solution for test scenarios