SYSC 4005 L2

Discrete Simulation/Modelling

Winter 2021

Milestone #3

Group Members: Julian Mendoza (101067270)

Hamna Nimra Manzoor (101057437)

Colin Wallace (100872525)

Due: April 1st, 2021

**TABLE OF CONTENTS**

**DELIVERABLE 3……………………………………………………………………………………………………………….…3-10**

[**Model Verification and Validation** 3](#_Toc68180522)

[**Verification method 1: Flow Diagrams:** 3](#_Toc68180523)

[**1.** **General simulation flow chart** 3](#_Toc68180524)

[**2.** **Event Handling** 4](#_Toc68180525)

[**3.** **Event change logic** 5](#_Toc68180526)

[**4.** **Priority Logic** 6](#_Toc68180527)

[**Verification Method 2: Examining Model output for Reasonableness.** 7](#_Toc68180528)

[**Current Contents** 7](#_Toc68180529)

[**Total Counts** 7](#_Toc68180530)

[**Verification Method 3: Tracing** 7](#_Toc68180531)

[**Validation Methods** 7](#_Toc68180532)

[**Sensitivity Analysis** 7](#_Toc68180533)

[**Input-Output Transformation** 8](#_Toc68180534)

[**Other Validation Alternatives** 8](#_Toc68180535)

[**Production Runs and Analysis** 8](#_Toc68180536)

[**Determining Number of Replications** 8](#_Toc68180537)

[**Simulation Results** 9](#_Toc68180538)

[**Initialization Phase** 9](#_Toc68180539)

**LIST OF FIGURES**

[Figure 1: Flowchart for a general simulation 3](#_Toc68177367)

[Figure 2: Flowchart handling new event occurrences 4](https://cmailcarletonca-my.sharepoint.com/personal/julianmendoza_cmail_carleton_ca/Documents/SYSC%204005%20-%20Project%20Report.docx#_Toc68177368)

[Figure 3: Combine images for a flowchart for an event change 5](https://cmailcarletonca-my.sharepoint.com/personal/julianmendoza_cmail_carleton_ca/Documents/SYSC%204005%20-%20Project%20Report.docx#_Toc68177369)

[Figure 4: Flowchart to determine priority of workstations 6](#_Toc68177370)

**LIST OF TABLES**

[Table 1: Results of simulation of 16 replications 9](#_Toc68177755)

[Table 2: Statistical Values 10](#_Toc68177755)

[Table 3: Sample average throughputs 10](#_Toc68177755)

# **Model Verification and Validation**

## **Verification method 1: Flow Diagrams:**

After each team member inspects the model closely, flow diagrams are made to make sure that the implemented logic makes sense. Flow diagrams give another perspective to verify that our model is going to do what it supposed to do.

### **General simulation flow chart**

The general simulation flow chart makes sure that we begin the simulation initializing the correct components in order, the process of proceeding to simulate the simulation is correct and that we can verify that our simulation will terminate under the correct circumstances. It is important to note that we can proceed with this logic due to the implemented FEL. Since our FEL uses a Queue in Java that is genetically typed to a SimEvent which implements comparable, the Queue acts as a priority queue which will retrieve the event with the lowest service time (time at which the event occurs).

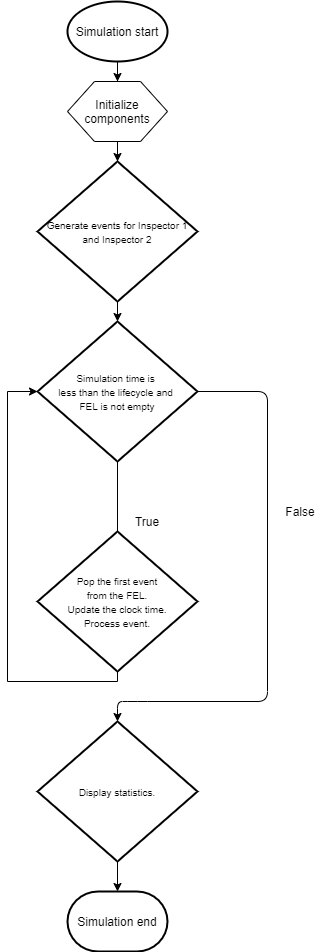


Figure 1: Flowchart for a general simulation

### **Event Handling**

Depending on which event the FEL pops from the queue, the simulation will behave differently. For inspector events, it is important that we create new event of the same type that is handled if the Inspector can find a route to a workstation, so that our simulation keeps running. Note that upon an event change, we call a helper function that looks at the states of different variables in the simulation and changes their state accordingly.

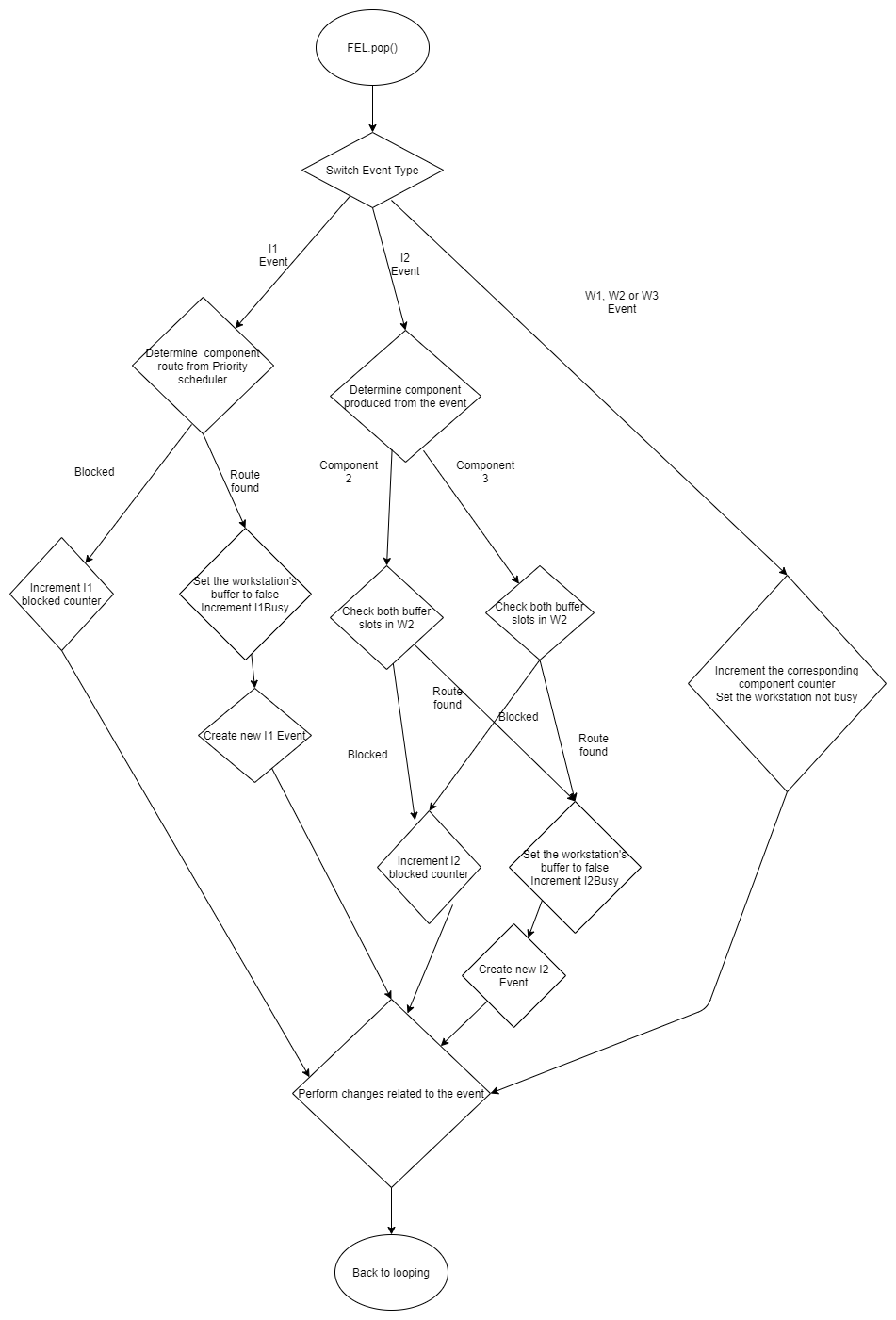


Figure 2: Flowchart handling new event occurrences

### **Event change logic**

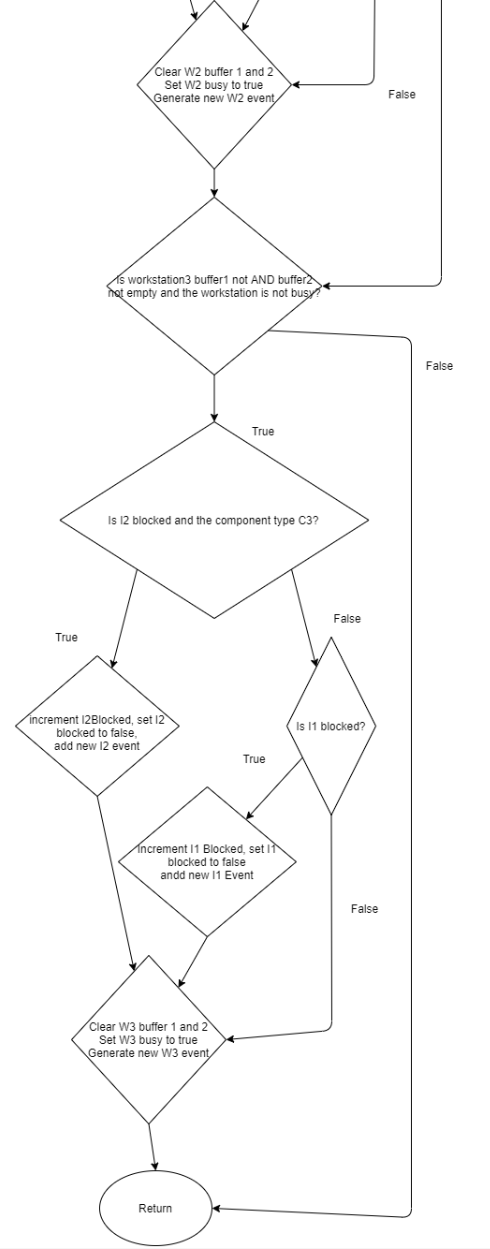
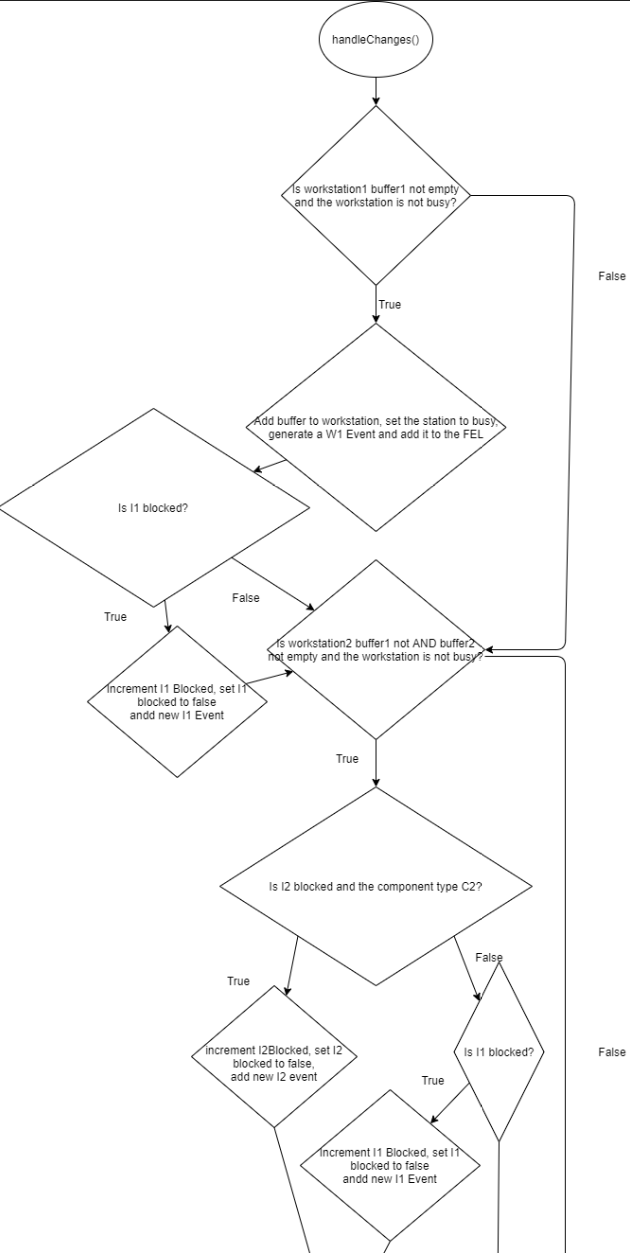
The method handleChanges will check all the state variables and determine if the right conditions are met to perform an appropriate event change. This method has the most responsibility as it handles the buffers in the workstation.

Figure 3: Combine images for a flowchart for an event change

### **Priority Logic**

The priority of the Workstation will determine where each component will route to. In the problem description, it states that the priority is given to the workstation with the least number of buffers, then it will give priority to W1, then W2 followed by W3. We can change this method to implement different scheduling systems.

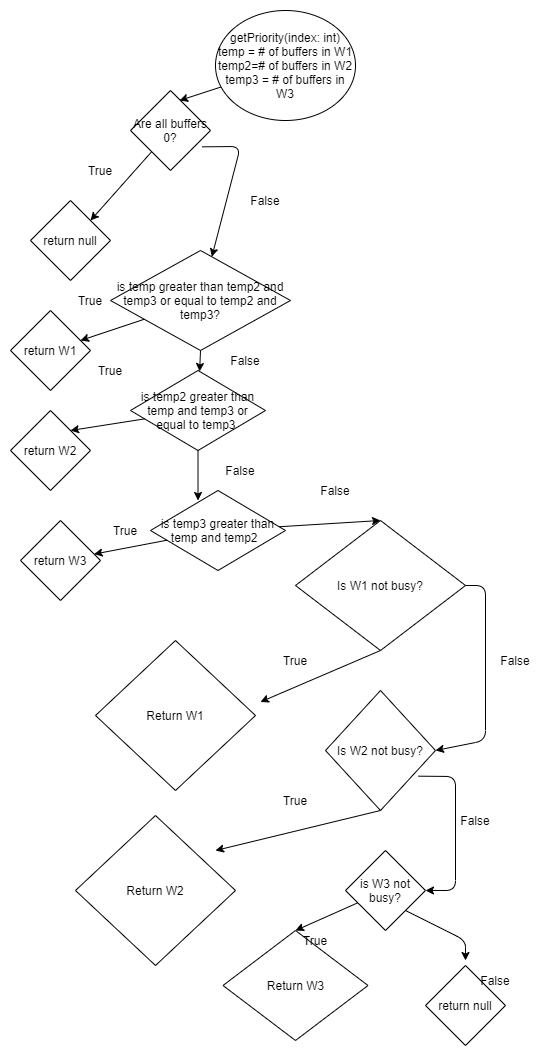


Figure 4: Flowchart to determine priority of workstations

## **Verification Method 2: Examining Model output for Reasonableness.**

### **Current Contents**

By looking at the number of products produced and the current state of the buffers in the workstations, we should be able to verify the correct behavior of the model. We know that our current model will produce far more product 1’s since our scheduler prioritizes workstation 1, workstation 1 only requires one component and the average time to create a product for this workstation is far less than the other two.

### **Total Counts**

In general, the number of events created should be more than the total number of products produced. Because we know that the current scheduler favors workstation one, the number of events will be close to the number of product ones produced once we reach steady state. The number of product twos and product threes should be roughly the same.

## **Verification Method 3: Tracing**

Upon running the model, whenever an event occurs, we call a helper function which prints the state variables of the simulation. We can confirm that our model is correct if the behavior of the trace is as expected.

## **Validation Methods**

We use different validation methods to make sure that our simulation is close to the real system as possible. This iterative process requires us to have a real system with its known behavior; however, we can address possible methods to reinsure that our model as close to the problem statement as possible.

## **Sensitivity Analysis**

Currently, the only values that the user can change to alter the data for the simulation is the lifecycle of the simulation. A sensitivity analysis based off the lifecycle would be completely useless because of the simulation’s initial conditions. If the lifecycle is too short, we would have less outputs and the statistics would be skewed. We know that the current scheduler produces more product ones than product two and product three. Increasing the lifecycle would not change the outcome of the products but dramatically change the statistics for the internal components.

## **Input-Output Transformation**

To validate the input and outputs of the simulation, we can provide the equation that must be true for each replication of the simulation. After each replication, the following must be true:

1. Total number of events generated by I1 = Total # (product 1 produced+ product 2 produced + product 3 produced) + number of products in(W1Buffer+W2Buffer0+W3Buffer0) + 1 IFF Inspector1 is not blocked
2. Total number of events generated by I2 = Total # (product 2 produced + product 3 produced) + number of products in(W2Buffer1+W2Buffer1) + 1 IFF Inspector2 is not blocked

## **Other Validation Alternatives**

An alternative validation method would be to compare the model with the historical data. Unfortunately, this is not an option since we don’t have the input/output transformation of the real model. Our only option is to make sure that the model is consistent with different lifecycles, seeds and replications.

### **Production Runs and Analysis**

**Quantities and Interests**

The statistics that we are looking for in the simulation are:

* 1. Total number of product #1 per given timeframe
  2. Throughput of product #1 per given timeframe
  3. Total number of product #2 per given timeframe
  4. Throughput of product #2 per given timeframe
  5. Total number of product #3 per given timeframe
  6. Throughput of product #3 per given timeframe
  7. Utilization (percentage) of Inspector 1
  8. Percentage blocked of Inspector 1
  9. Utilization (percentage) of Inspector 2
  10. Percentage blocked of Inspector 2

## **Determining Number of Replications**

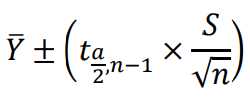
In the project requirements, it states that we must have 95% confidence intervals with a width that does not exceed 20% of the estimated values. To achieve this, we must determine an appropriate lifecycle. This lifecycle value must be greater than the initialization phase. A large value (10000) seconds was chosen for computing these statistics. To remove an amount of unbiases, each simulation will be run at minimum 5 times. This is to ensure that the randomness of the seed does not affect the half-width. Once the simulation runs at least 5 times, it will calculate the statistics and determine if the half-width\*2 is within the 20% threshold. If the simulation is appropriate, it will break from the loop and output the number of replications.

### **Simulation Results**

Table 1: Results of simulation of 17 replications

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Trial | Product1 Tally | Product1 Through-put | Product2 Tally | Product2 Through-put | Product 3 Tally | Product3 Through-put | Utilization I1 | %  Blocked  I1 | Utilization I2 | %  Blocked  I2 |
| 0 | 879 | 0.087879 | 64 | 0.006398 | 48 | 0.004799 | 99.96772 | 0.019946 | 19.06088 | 74.14625 |
| 1 | 826 | 0.082596 | 62 | 0.0062 | 40 | 0.004 | 99.92046 | 0.028824 | 19.16333 | 75.87816 |
| 2 | 876 | 0.087598 | 56 | 0.0056 | 33 | 0.0033 | 100 | 0 | 14.7209 | 79.93681 |
| 3 | 889 | 0.088837 | 55 | 0.005496 | 48 | 0.004797 | 99.90097 | 0.034302 | 20.45527 | 72.44747 |
| 4 | 817 | 0.0817 | 68 | 0.0068 | 52 | 0.0052 | 100 | 0 | 23.86502 | 71.01927 |
| 5 | 868 | 0.086772 | 41 | 0.004099 | 29 | 0.002899 | 99.90664 | 0 | 15.02661 | 78.13881 |
| 6 | 880 | 0.087996 | 53 | 0.0053 | 49 | 0.0049 | 100 | 0 | 18.39763 | 74.76312 |
| 7 | 788 | 0.078797 | 39 | 0.0039 | 22 | 0.0022 | 100 | 0 | 12.02538 | 84.81002 |
| 8 | 871 | 0.087045 | 59 | 0.005896 | 43 | 0.004297 | 99.97932 | 0.018712 | 17.99154 | 75.05427 |
| 9 | 889 | 0.088824 | 83 | 0.008293 | 49 | 0.004896 | 99.52653 | 0.414264 | 22.86446 | 70.48544 |
| 10 | 843 | 0.084291 | 58 | 0.005799 | 37 | 0.0037 | 99.90806 | 0 | 16.52061 | 77.94918 |
| 11 | 905 | 0.090457 | 70 | 0.006997 | 51 | 0.005098 | 99.9676 | 0.018626 | 24.11292 | 71.09808 |
| 12 | 869 | 0.086718 | 57 | 0.005688 | 48 | 0.00479 | 99.8591 | 0.023712 | 21.39355 | 72.71549 |
| 13 | 858 | 0.085735 | 72 | 0.007195 | 46 | 0.004597 | 100 | 0 | 23.41527 | 70.45346 |
| 14 | 866 | 0.086566 | 60 | 0.005998 | 49 | 0.004898 | 99.70796 | 0.161722 | 23.43493 | 71.67271 |
| 15 | 864 | 0.086376 | 49 | 0.004899 | 40 | 0.003999 | 99.95093 | 0 | 14.73324 | 80.38879 |
| 16 | 862 | 0.086168 | 65 | 0.006498 | 42 | 0.004198 | 100 | 0 | 23.15131 | 69.15671 |

From Table 1, we can see that 17 number of replications was required for lifecycle = 10000.

We use the following equation to calculate the confidence interval estimation:

From the t-table:

Table 2: Statistical values of the 17 replications lifecycle = 10000

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Statistic | Product1 Tally | Product1 Through-put | Product2 Tally | Product2 Through-put | Product 3 Tally | Product3 Through-put | Utilization I1 | %  Blocked  I1 | Utilization I2 | %  Blocked  I2 |
| Mean | 861.7647 | 0.086138 | 59.47059 | 0.005944 | 42.70588 | 0.004269 | 99.91737 | 0.042359 | 19.43134 | 74.71259 |
| Variance | 842.1912 | 8.33E-06 | 119.0147 | 1.19E-06 | 70.47059 | 7.03E-07 | 0.015717 | 0.010683 | 14.64573 | 18.58907 |
| Half width | 14.85126 | 0.001477 | 5.58288 | 0.000558 | 4.295977 | 0.000429 | 0.064157 | 0.052894 | 1.958454 | 2.206413 |
| 20% Error | 172.3529 | 0.017228 | 11.89412 | 0.001189 | 8.541176 | 0.000854 | 19.98347 | 0.008472 | 3.886269 | 14.94252 |
| Width | 29.70252 | 0.002954 | 11.16576 | 0.001115 | 8.591954 | 0.000858 | 0.128314 | 0.105788 | 3.916909 | 4.412826 |
| Upper Range | 846.9134 | 0.084661 | 53.88771 | 0.005387 | 38.40991 | 0.003839 | 99.85321 | 0 | 17.47289 | 72.50618 |
| Lower Range | 876.616 | 0.087615 | 65.05347 | 0.006502 | 47.00186 | 0.004698 | 99.98153 | 0.095253 | 21.3898 | 76.919 |
| 95 % Confidence Interval | 846.9134 –  876.616 | 0.084661 – 0.087615 | 53.88771 –  65.05347 | 0.005387 –  0.006502 | 38.40991 –  47.00186 | 0.003839 –  0.004698 | 99.85321 –  99.98153 | 0 –  0.095253 | 17.47289 –  21.3898 | 72.50618 –  76.919 |

### **Initialization Phase**

We know that is currently a bias with the current scheduling system which affects the statistics for a certain period. This period is called the initialization phase.

To determine the initialization phase of the system, we will determine the average for the throughput of products for a large lifecycle (10000 seconds) for 17 replications where the accepted error is 20% of the estimated value. We will start at a low lifecycle (200 seconds) for 20 replications. If the throughput of the products is within the bounds of the estimated mean, we will note the time of the lifecycle and consider this the end of the initialization period. If this requirement is not met, we will increment the lifecycle by 100 seconds and repeat the process.

Targets: P1: 0.084661 - 0.087615, P2: 0.005387 - 0.006502, P3: 0.003839 - 0.004698

Table 3: Sample average throughput of 20 replications at a given lifecycle.

|  |  |  |  |
| --- | --- | --- | --- |
| Lifecycle | P1 Throughput average | P2 Throughput average | P3 Throughput average |
| 200 | 0.08039 | 0.007381 | 0.004874 |
| 300 | 0.08241 | 0.006963 | 0.004037 |
| 400 | 0.082808 | 0.006344 | 0.004148 |
| 500 | 0.084176 | 0.006796 | 0.004342 |
| 600 | 0.086428 | 0.006535 | 0.004304 |
| 700 | 0.087782 | 0.006734 | 0.004249 |
| 800 | 0.088173 | 0.006873 | 0.004587 |

From the above table, we can conclude that the initialization phase is the period from 0 to about 600 seconds. This value can vary based off the seed number and number of replications. Increasing the number of replications would make the data more accurate.