UNIVERSITÀ DEGLI STUDI DELL'AQUILA

Software Quality Engineering

Homework 2: Software performance modeling and analysis

Author :
Anusha Annengala
Mhreteabe Dula
Juliette Waltregny

Professor : V. Cortellessa
Year : 2023-2024

Performance Modeling Report for Route Optimization System

March 8, 2024

Contents

1 Introduction

1.1 System Overview

Building upon the foundation of the existing Research Infrastructure for Big Data and Social Mining (SoBigData) platform, our system aims to further exploit social and geographical datasets through advanced algorithms. The primary focus is on providing optimized transportation routes, enhancing urban mobility solutions with greater efficiency and sustainability. By working together with SoBigData, our system will use its powerful data processing abilities to improve what we offer, adding our own algorithms for digging into data and analyzing it. This collaboration is expected to make a big difference in how we understand people's movement patterns and improve traffic control and urban planning.

1.2 Objectives

The objective of this performance modeling exercise is to critically evaluate and increase the processing efficiency of our system with SoBigData. This means identifying and mitigating potential bottlenecks, optimizing resource usage, and ensuring the system's scalability to accommodate growing data volumes. Enhancing throughput, minimizing latency, and improving the precision and response time of transportation route suggestions are our main goals. This exercise aims to not only boost the operational efficiency of the system but also to significantly contribute to sustainable urban development through data-driven insights.

2 Performance Requirements

The Route Optimization System, designed for an initial core team of data scientists, transportation experts, and managers, establishes the following core performance metrics to ensure efficient operation and high-quality service:

1. **Response Time:**
    - *optimizing routes:* The system shall deliver optimized routing suggestions within 5 seconds under an arrival rate of 5 jobs/seconds.

- *analyzing data:* Data analysis operations, crucial for understanding traffic patterns and planning, are to be completed within 5 seconds under an arrival of 5 jobs/seconds.

2. **Throughput:**

   - The system shall support 3 Analyse data jobs per second, under an arrival rate of 5 jobs/seconds.

   - The system shall support 2 Route optimizations jobs per second, under an arrival rate of 5 jobs/seconds.

3. **Resource Utilization:**

   - **Database:** It shall utilize up to 60% of its allocated resources.
   - **Display:** It shall maintain a low resource utilization of up to 40%, given its less computationally intensive tasks.
   - **Analytics Engine:** It shall operate at up to 80% of capacity for regular data analysis tasks.
   - **Optimization Engine:** It shall consume up to 70% of resources for route optimization processes.

3 Sequence Diagrams

We decided to modify only the sequence diagram of Optimize Route which is explained below. The others remain unchanged as we think they have been correctly illustrated in the first homework.

3.1 Optimize Route

The use case Optimize Route is derived from the "Configure route optimization engine and optimise routes" use case. The reason we decided to split the two tasks that was because we believed they should be independent of each other, as the use case "Configure route optimization engine and optimize routes" was performing two independent tasks which are: optimizing routes and configuring the engine. For our analysis we have kept the Optimize route.

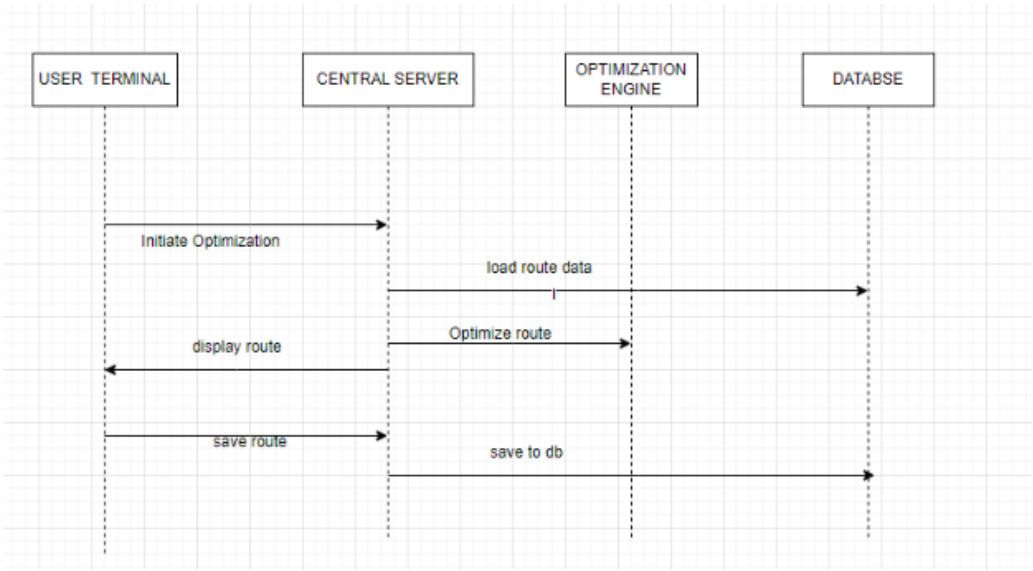

Figure 1: Optimize Route

4 Execution Graphs

Execution Graphs provide a detailed visual representation of the execution paths within a software system, highlighting the sequence of operations, decision points, and potential parallel executions. These graphs are important in identifying performance bottlenecks, understanding resource dependencies, and optimizing the execution flow for better performance. In this section, we will present four execution graphs derived from our sequence diagrams.

4.1 Analyze Data

This graph illustrates the execution path for Analyze Data. After the system has collected the filtered data, they want them to be analyzed to extract sentiment or transportation behaviours. We can distinguish three different types of data: raw, social and mobility. The raw data is every data not related to the behaviours of people and their sentiment like their age, nationality etc. Social and mobility data are respectively data on sentiment and mobility behaviours. To extract valuable analyzed data, the system retrieves the previously filtered data from the database, then analyzes it. After that, in

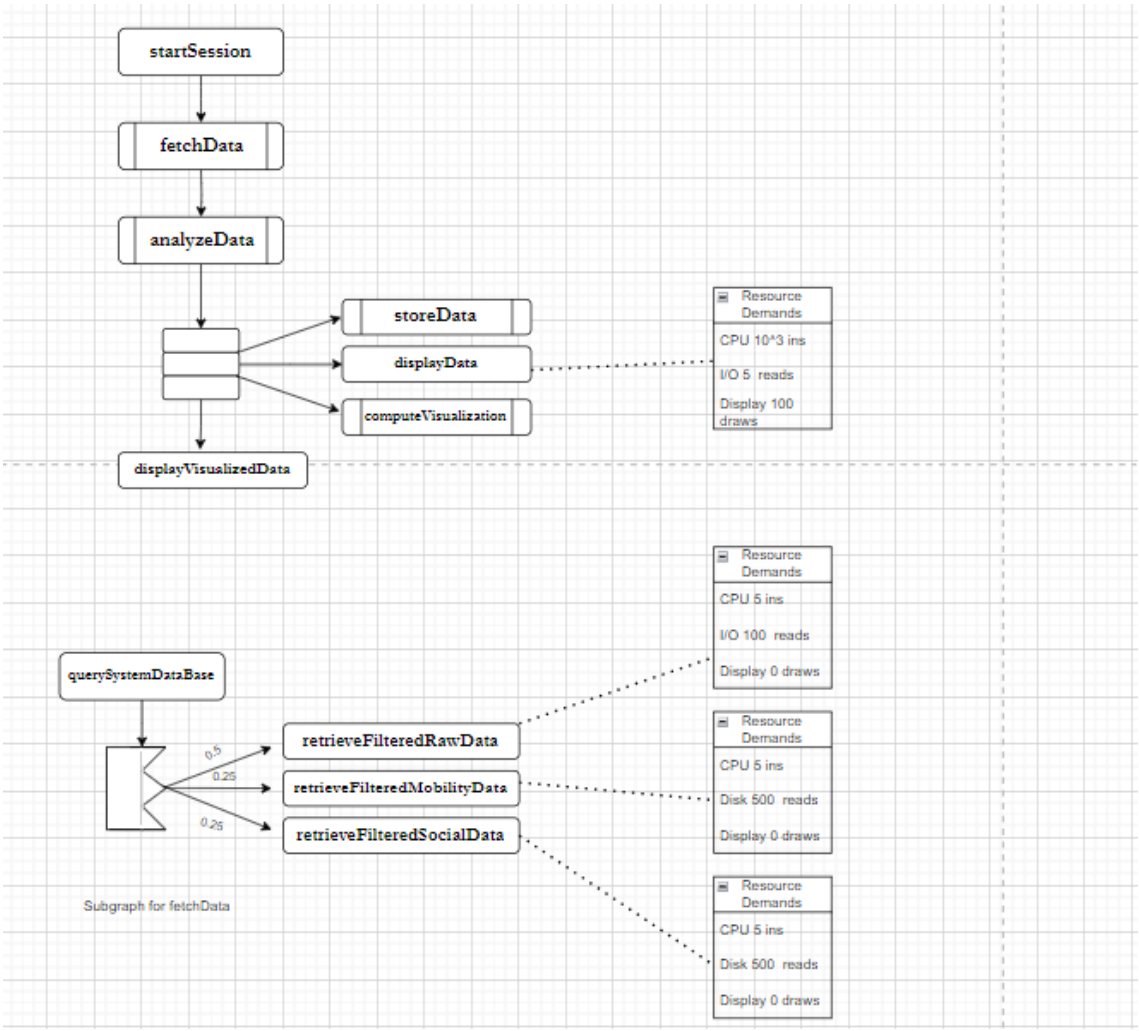

Figure 2: Main execution graph and subgraph for fetchData

parallel, the data is stored, is displayed to the UI and is being computed to output the visualization. These visualizations can be graphs, maps or other representation requested. The central server therefore routes the different tasks to be done to their respective engine. After that, the visualized data is displayed to the UI.

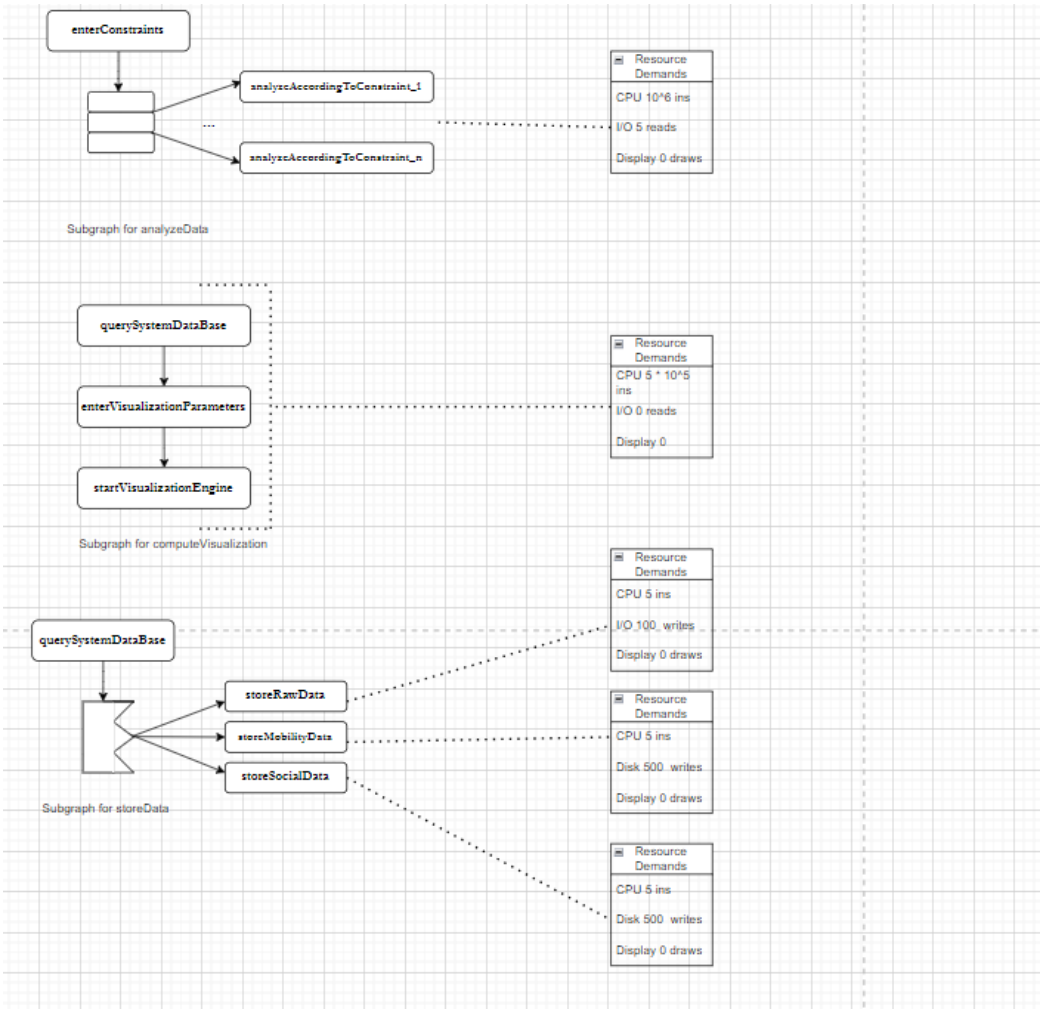

Figure 3: Subgraphs of analyzeData, computeVisualization and store data

4.2 Optimize Route

This graph illustrates the execution path for Optimizing a given route, Which is one of the main tasks in our system. As we can see from the diagram it consists of 4 sub tasks: Load route, optimize route, display route and save to Db.

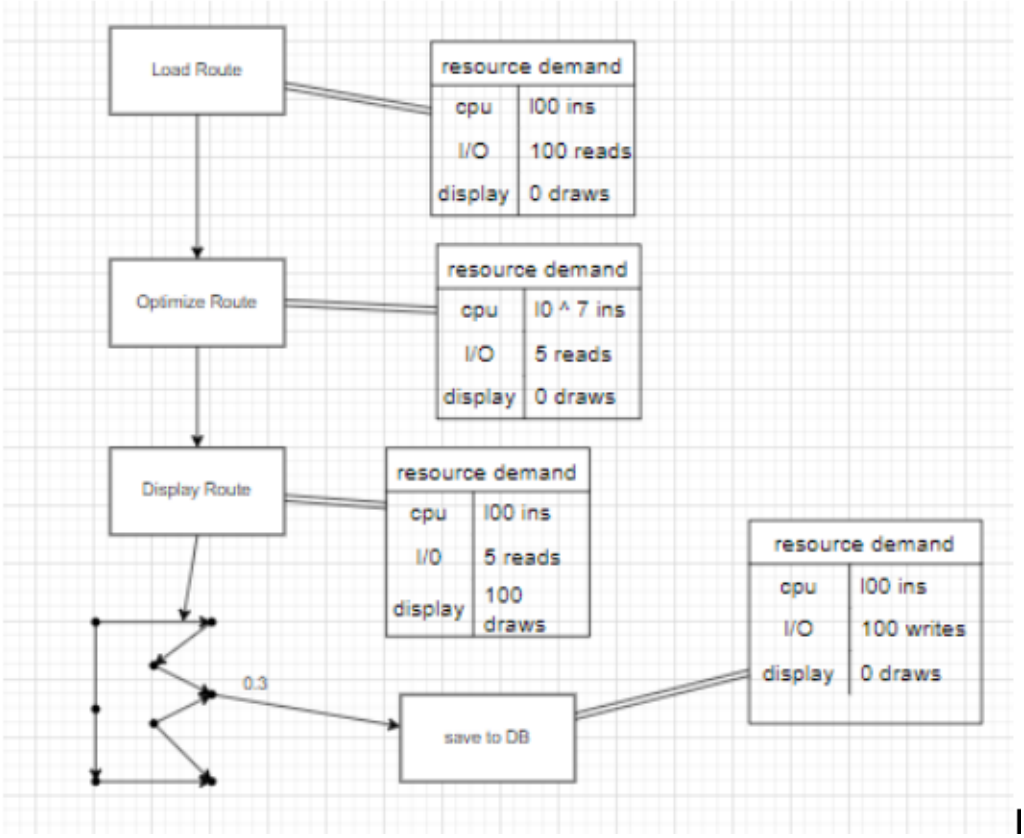

Figure 4: execution graph for Optimize Route

4.3 Take Part in Training

This graph focuses on the training module of the system, identifying critical paths and potential bottlenecks in the system's processing. When the user accesses a training, the system retrieves the corresponding data, either a new training data or the corresponding session data. The session data refers to the previous session of the user related to the particular training module. The system then displays the data and saves the corresponding session data.

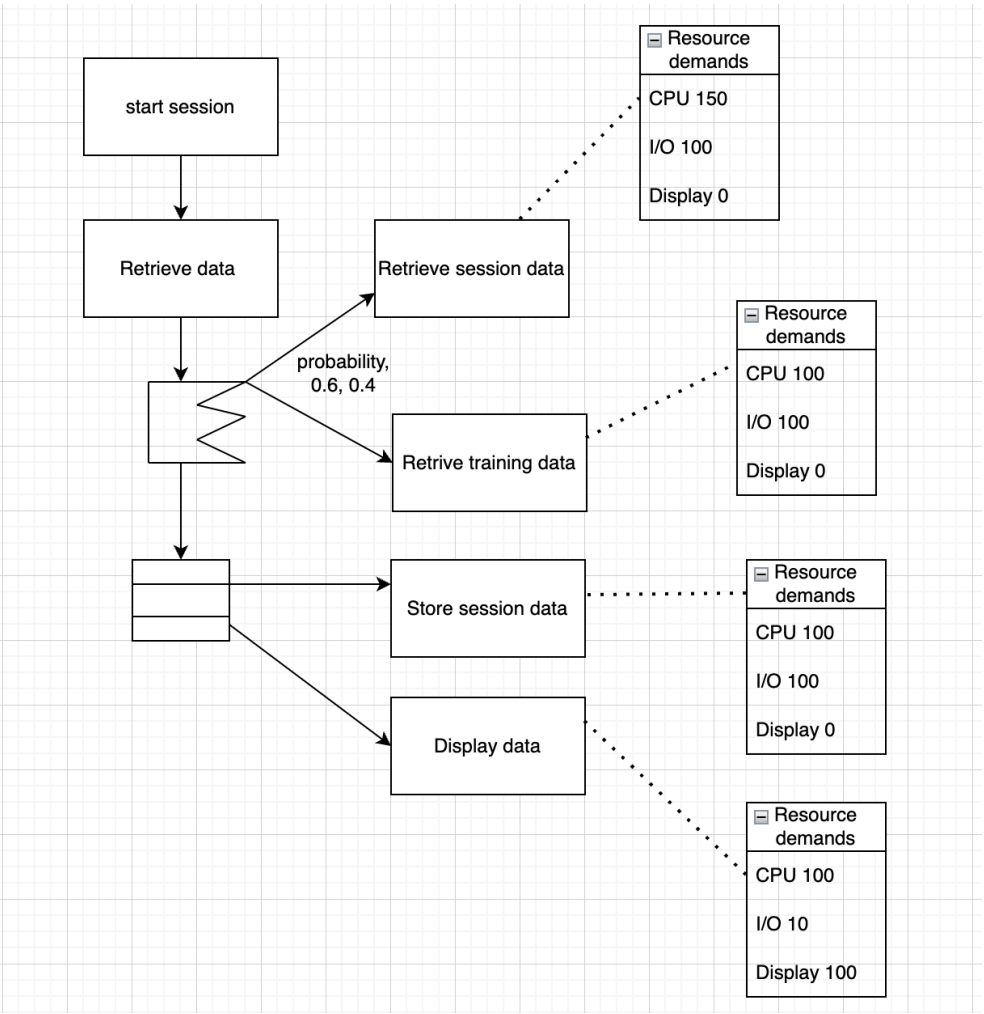

Figure 5: Take Part In Training

4.4 Use Validation Framework

This graph focuses on the validation framework of the system, identifying critical paths and potential bottlenecks in the system's processing. When the user tries to validate against the ethical or legal framework, the system first retrieves the corresponding validation (legal or ethical), computes the visualization and then displays it to the user, the display could either just be the response of the validation or the visualization engine.

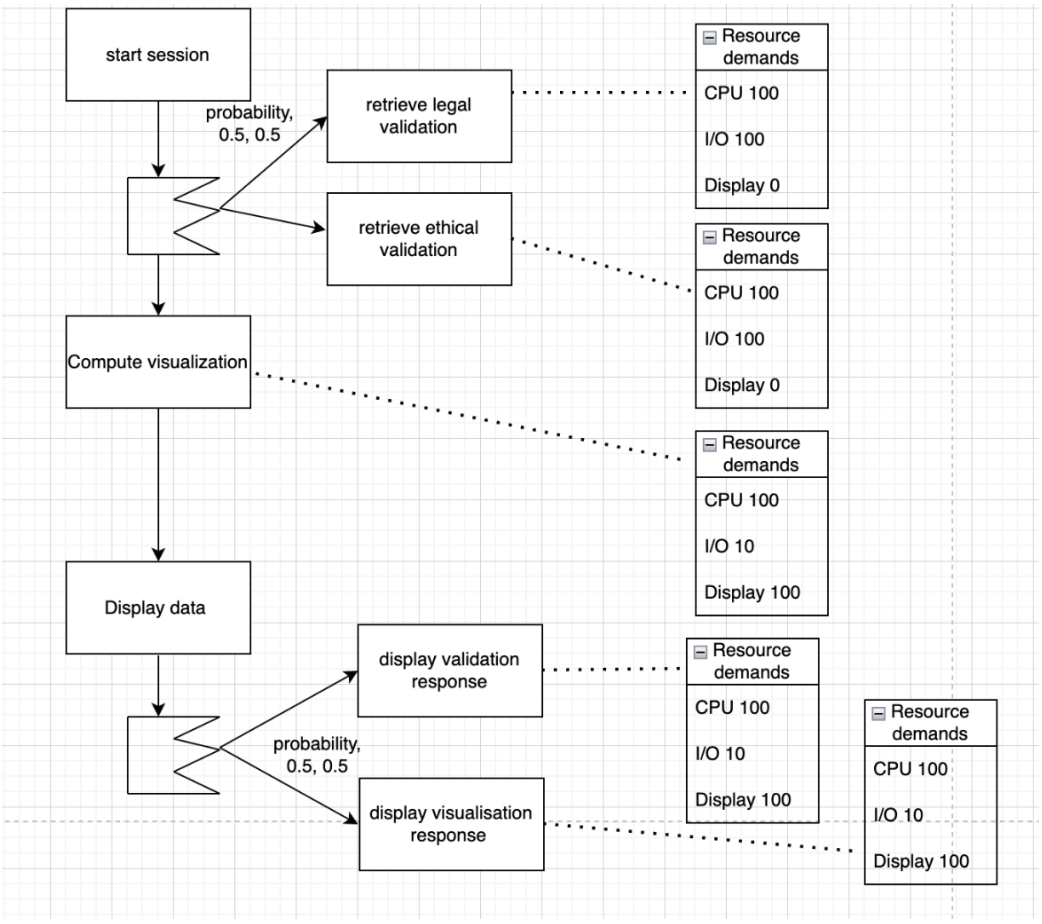

Figure 6: Use Validation Framework

5 Worst-Case Analysis

This section presents a detailed analysis of the worst-case performance scenarios for our system, based on the execution graphs previously discussed. The analysis begins with an examination of the resource overheads for key processes in the system.

5.1 Resource Overheads

Below is a table detailing the resource requirements for each critical process within our system. These table provides an estimated number of computational units in our system along with their dependency.

Devices	**CPU**	**Disk**	**Display**
Service Unit	1000 instructions	100 I/O	100 units
CPU	1	0	0
I/O	0.1	1	0
Screen	0.05	0	1
Service Time	0.333×10^{-6} sec	0.001 sec	0.001 sec

Table 1: Resource overhead

5.2 Formula Used to Calculate Worst-Case Response Time

$$
\begin{aligned}
\textit{Total processing time} &= \text{CPU time} + \text{Disk time} + \text{Display time} \\
\text{CPU time} &= \left(\frac{\text{Number of instructions}}{1000}\right) \times \text{CPU service time} \\
&\quad + \text{Disk overhead} \times \text{Disk service time} \\
&\quad + \text{Display overhead} \times \text{Display service time} \\
\text{Disk time} &= \left(\frac{\text{Disk operations}}{100}\right) \times \text{Service time per operation} \\
&\quad + \text{CPU overhead} \times \text{CPU service time} \\
&\quad + \text{Display overhead} \times \text{Display service time} \\
\text{Display time} &= \left(\frac{\text{Number of draw operations}}{100}\right) \times \text{Service time per draw} \\
&\quad + \text{Number of draw operations} \times \text{CPU overhead} \times \text{CPU service time} \\
&\quad + \text{Number of draw operations} \times \text{Disk overhead} \times \text{Disk service time}
\end{aligned}
$$

5.3 Worst-Case Analysis for Analyze Data

Load Data:

$$\text{CPU Time} = \frac{5}{1000} \times 0.333 \times 10^{-6}\,\text{sec}$$
$$\text{Disk Time} = 500 \times 0.1 \times 0.333 \times 10^{-6} + \frac{500}{100} \times 0.001\,\text{sec}$$
$$\text{Total Time} = \text{CPU Time} + \text{Disk Time} = 0.005016651665\,\text{sec}$$

Analyze Data:

$$\text{CPU Time} = \frac{10^6}{1000} \times 0.333 \times 10^{-6}\,\text{sec}$$
$$\text{Disk Time} = 5 \times 0.1 \times 0.333 \times 10^{-6} + \frac{5}{100} \times 0.001\,\text{sec}$$
$$\text{Total Time} = \text{CPU Time} + \text{Disk Time} = 0.0003831665\,\text{sec}$$

Compute Visualization:

$$\text{CPU Time} = \frac{5 * 10^5}{1000} \times 0.333 \times 10^{-6}\,\text{sec}$$
$$\text{Total Time} = \text{CPU Time} = 0.0001665\,\text{sec}$$

Store Analyzed Data:

$$\text{CPU Time} = \frac{5}{1000} \times 0.333 \times 10^{-6}\,\text{sec}$$
$$\text{Disk Time} = 500 \times 0.1 \times 0.333 \times 10^{-6} + \frac{500}{100} \times 0.001 + \text{ sec}$$
$$\text{Total Time} = \text{CPU Time} + \text{Disk Time} = 0.005016651665\,\text{sec}$$

Display Analyzed Data:

$$\text{CPU Time} = \frac{10^5}{1000} \times 0.333 \times 10^{-6}\,\text{sec}$$
$$\text{Disk Time} = 5 \times 0.1 \times 0.333 \times 10^{-6} + \frac{5}{100} \times 0.001\,\text{sec}$$
$$\text{Display Time} = 100 \times 0.05 \times 0.333 \times 10^{-6} + 100 \times 0.001\,\text{sec}$$
$$\text{Total Time} = \text{CPU Time} + \text{Disk Time} = 0.1000851315\,\text{sec}$$

Display Visualized Data:

$$\text{CPU Time} = \frac{10^3}{1000} \times 0.333 \times 10^{-6}\,\text{sec}$$
$$\text{Disk Time} = 5 \times 0.1 \times 0.333 \times 10^{-6} + 5 \times 0.001 + \text{ sec}$$
$$\text{Display Time} = 500 \times 0.05 \times 0.333 \times 10^{-6} + \frac{500}{100} \times 0.001\,\text{sec}$$
$$\text{Total Time} = \text{CPU Time} + \text{Disk Time} + \text{Display Time} = 0.5000588245\,\text{sec}$$

Overall System Time :

$$\begin{aligned}\text{Overall System Time} &= \text{Total Time for Load Data} + \text{Total Time for Analyze Data} \\ &\quad + \text{Total Time for Compute Visualization} + \text{Total Time for Store Analyzed} \\ &= 0.60507 \text{ sec}\end{aligned}$$

To calculate the time for the worst execution path,we sum the time taken for the individual processes, which comes about 0.60507 sec.

Concerning the throughput, we have an arrival rate for the Analyze data job of 5 jobs/seconds. The service time is calculated above, so 0.60507. With the formula, we get:

$$\textit{Throughput} = \left(\frac{\text{Arrival Rate}}{\text{Service time}}\right) = \left(\frac{5}{0.60507}\right) = 8.26 \textit{jobs/seconds}. \quad (1)$$

The requirement for the throughput of Analyze Data job is met.

5.4 Worst-Case Analysis for Optimize Route

Load Route:

$$\begin{aligned}\text{CPU Time} &= \frac{100}{1000} \times 0.333 \times 10^{-6}\,\text{sec}\\ \text{Disk Time} &= 100 \times (0.001 + 10^{-7} \times 0.333)\,\text{sec}\\ \text{Total Time} &= \text{CPU Time} + \text{Disk Time}\\ &= 0.1000033633 \text{ sec}\end{aligned}$$

Optimize Route:

$$\begin{aligned}\text{CPU Time} &= \frac{10^7}{1000} \times 0.333 \times 10^{-6}\,\text{sec}\\ \text{Disk Time} &= 5 \times (0.001 + 10^{-6} \times 0.333)\,\text{sec}\\ \text{Total Time} &= \text{CPU Time} + \text{Disk Time}\\ &= 0.0038301665 \text{ sec}\end{aligned}$$

Display Route:

$$\begin{aligned}\text{CPU Time} &= \frac{100}{1000} \times 0.333 \times 10^{-6}\,\text{sec}\\ \text{Disk Time} &= 5 \times (0.001 + 10^{-6} \times 0.333)\,\text{sec}\\ \text{Display Time} &= 100 \times (0.001 + 0.05 \times 10^{-6} \times 0.333)\,\text{sec}\\ \text{Total Time} &= \text{CPU Time} + \text{Disk Time} + \text{Display Time}\\ &= 0.1000333 \text{ sec}\end{aligned}$$

Save to DB:

$$\begin{aligned}\text{CPU Time} &= \frac{100}{1000} \times 0.333 \times 10^{-6}\,\text{sec}\\ \text{Disk Time} &= 100 \times (0.001 + 10^{-6} \times 0.333)\,\text{sec}\\ \text{Total Time} &= \text{CPU Time} + \text{Disk Time}\\ &= 0.100000666 \text{ sec}\end{aligned}$$

Overall System Time :

$$\begin{aligned}\text{Overall System Time} &= \text{Total Time for Load Route } + \text{Total Time for Optimize Route}\\ &\quad + \text{Total Time for Display Route } + \text{Total Time for Save to DB}\\ &= 0.30 \text{ sec}\end{aligned}$$

To calculate the time for the worst execution path,we sum the time taken for the individual processes, which comes about 0.3 sec.

Concerning the throughput, we have an arrival rate for the Analyze data job of 5 jobs/seconds. The service time is calculated above, so 0.3. With the formula, we get:

$$Throughput = \left(\frac{\text{Arrival Rate}}{\text{Service time}}\right) = \left(\frac{5}{0.3}\right) = 16.666 jobs/seconds. \quad (2)$$

The requirement for the throughput of Optimize Route job is met.

5.5 Worst-Case Analysis for Take Part In Training

Retrieve data:

$$\begin{aligned}\text{CPU Time} &= \frac{150}{1000} \times 0.333 \times 10^{-6}\,\text{sec}\\ \text{Disk Time} &= 100 \times 0.1 \times 0.333 \times 10^{-6} + 0.001 \times 100\,\text{sec}\\ \text{Total Time} &= 0.10000337995\,\text{sec}\end{aligned}$$

Store session data:

$$\begin{aligned}\text{CPU Time} &= \frac{100}{1000} \times 0.333 \times 10^{-6}\,\text{sec}\\ \text{Disk Time} &= 100 \times 0.1 \times 0.333 \times 10^{-6} + 0.001 \times 100\,\text{sec}\\ \text{Total Time} &= 0.1000033633\,\text{sec}\end{aligned}$$

Display data:

$$\begin{aligned}\text{CPU Time} &= \frac{100}{1000} \times 0.333 \times 10^{-6}\,\text{sec}\\ \text{Disk Time} &= 10 \times 0.1 \times 0.333 \times 10^{-6} + 0.001 \times 10\,\text{sec}\\ \text{Display Time} &= 100 \times 0.05 \times 0.333 \times 10^{-6} + 0.001 \times 100\,\text{sec}\\ \text{Total Time} &= 0.1100020313\,\text{sec}\end{aligned}$$

Overall System Time :

$$\begin{aligned}\text{Overall System Time} = \text{Total Time for Retrieve Data } &+ \text{Total Time for Store session Data} \\ &+ \text{Total Time for Display Data}\end{aligned}$$

When we sum the results we get approximately **0.3 sec** of response time

5.6 Worst-Case Analysis for Use Validation Framework

Load validation:

$$\begin{aligned}\text{CPU Time} &= \frac{100}{1000} \times 0.333 \times 10^{-6}\,\text{sec} \\ \text{Disk Time} &= 100 \times 0.1 \times 0.333 \times 10^{-6} + 0.001 \times 100\,\text{sec} \\ \text{Total Time} &= 0.1000033633\,\text{sec}\end{aligned}$$

Compute visualisation:

$$\begin{aligned}\text{CPU Time} &= \frac{100}{1000} \times 0.333 \times 10^{-6}\,\text{sec} \\ \text{Disk Time} &= 10 \times 0.1 \times 0.333 \times 10^{-6} + 0.001 \times 10\,\text{sec} \\ \text{Display Time} &= 100 \times 0.05 \times 0.333 \times 10^{-6} + 0.001 \times 100\,\text{sec} \\ \text{Total Time} &= 0.1100020313\,\text{sec}\end{aligned}$$

Display data:

$$\begin{aligned}\text{CPU Time} &= \frac{100}{1000} \times 0.333 \times 10^{-6}\,\text{sec} \\ \text{Disk Time} &= 10 \times 0.1 \times 0.333 \times 10^{-6} + 0.001 \times 10\,\text{sec} \\ \text{Display Time} &= 100 \times 0.05 \times 0.333 \times 10^{-6} + 0.001 \times 100\,\text{sec} \\ \text{Total Time} &= 0.1100020313\,\text{sec}\end{aligned}$$

Overall System Time :

$$\text{Overall System Time} = \text{Total Time for Load Validation} + \text{Compute Visualization} + \text{Total Time for Display Data}$$

When we sum the results we get approximately **0.3 sec** of response time

6 Queueing Network Model

6.1 Model Description

In our Queueing Network Model, each service station represents a key component in our system's architecture, utilizing container technology to achieve improved scalability, isolation, and efficiency..

6.1.1 Service Stations

Service Stations	Description	Number of servers
Central Server	Coordinates system-wide activities	1
Optimization Server	Handles route optimization	2
Display	Manages graphics rendering	1
Database	Repository for cached and operational data	2
Visualization Engine	Transforms data into visual formats	1
Analytics Engine	Performs data analysis	1
Legal and Ethical Validator	Ensures compliance	1

Table 2: Service stations

We have a Container based architecture which allows each of these components to be hosted in isolated environments, ensuring that system resources are optimally utilized and that services can be scaled independently based on demand.

6.1.2 Job Classes

By analyzing our execution graphs, we have delineated the job classes within our system into four primary meta classes, each reflecting critical aspects of our system's functionality:

- **Loading Data:** Includes jobs like load route, load training, load validation, and load analyzed data.
- **Saving Data:** Encompasses save route, save session data, store analyzed data.
- **Displaying Data:** Covers display route, display training, visualize validation, display data, display analyzed data, display visualize data.
- **Computational Jobs:** Comprises optimize route, analyze data, compute visualization.

6.2 Model Analysis and Bottleneck Identification

In our first iteration of implementing the queuing network, we allocated two servers for database, one server for display, one of central server and two servers for route optimization engine. We observed the simulation results as shown in the diagram which do not satisfy our requirements.

We will have to modify the number of servers to observe a better simulation result.

Performance Index	**Requirement**	**Simulated Value**	**Remark**
Response time for optimize route	5 sec	5×10^{-3} sec	OK
Response time for analyze data	5 sec	0.175 sec	OK
Utilization of Database	60%	100%	Critical
Utilization of Display	40%	100%	Critical
Utilization of analytics engine	80%	43%	OK
Utilization of optimization engine	70%	1%	OK
Throughput for optimization engine	2 jobs/sec	4 jobs/sec	OK
Throughput for analysis engine	3 jobs/sec	4 jobs/sec	OK

Table 3: Performance Indices, Requirements, and Simulated Values

Most of our requirements are met with the exception of Utilization of Database and display. We will try to solve these by increasing the number of servers from 2 to 4 and from 1 to 2 for database and display respectively

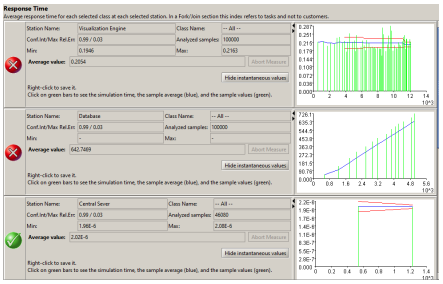

Figure 7: Response Time ,1st iteraiton

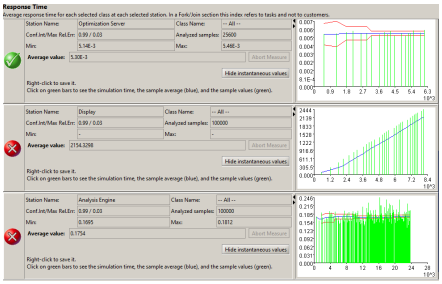

Figure 8: Response Time ,1st iteration

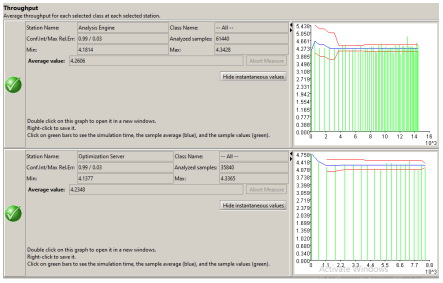

Figure 9: Throughput ,1st iteration

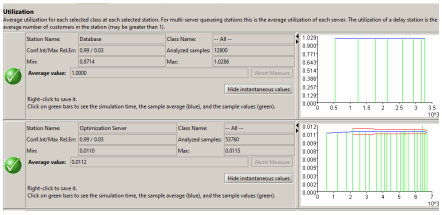

Figure 10: Utilization of database and Optimization server, 1st iteration

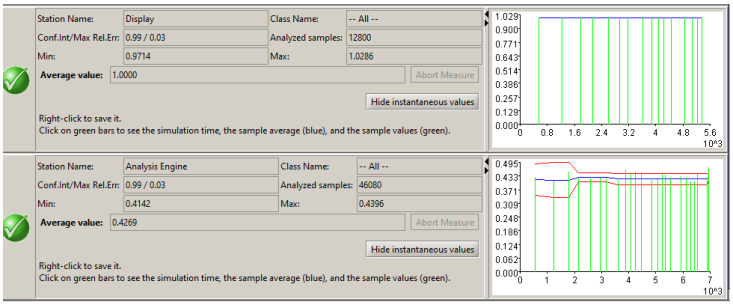

Figure 11: Utilization of Display and Analysis Engine, 1st iteration

6.3 Model Improvement and Analysis

In our second iteration, we see that the requirement is still not met for Display and Database engine (Figure 14).

We further increased the number of servers for display to 3 and database to 5 and the simulation results are shown in Figure 15. It is observed that the results now satisfy our requirements.

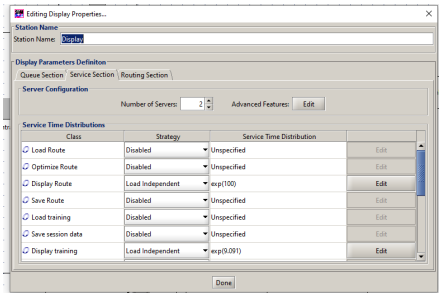

Figure 12: display server,2nd iteration

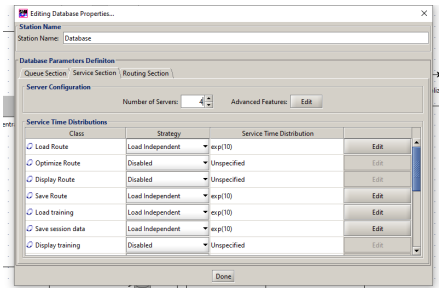

Figure 13: database server, 2nd iteration

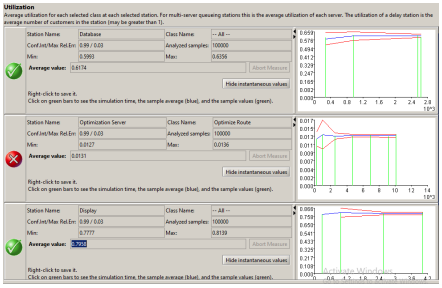

Figure 14: Utilization , 2nd iteration

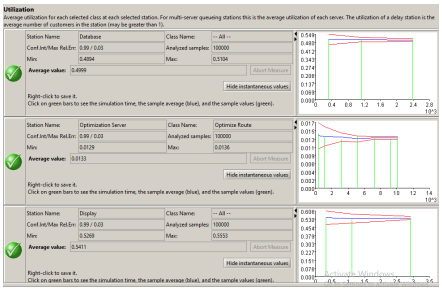

Figure 15: Utilization , final iteration

7 What If Analysis

7.1 Arrival Rate of Route Optimization Jobs

In this section, we will investigate how the arrival rate of route optimization jobs affects our system's capabilities. To do this, we will conduct a what-if analysis for the job load route, which is the causal parent of the route optimization job and is formed in a class switch. We will analyze the arrival rate of load route jobs from 5 jobs/sec to 12 jobs/sec, with an increment of 2, and observe how that affects our system.

1. Response Time for Optimize Route and Analyze Data

As more jobs start to arrive, the response time of the analytics engine has gone up, while the response time of the optimization engine has gone down. The latter result seems unexpected. One reason for this aberrant behavior could be that since more routes are being loaded and other servers are becoming congested, fewer jobs are coming to the optimization engine, which

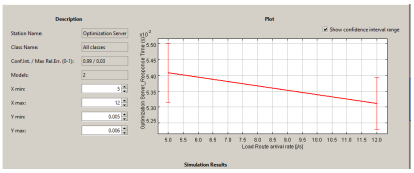

Figure 16: response time of analytics engine

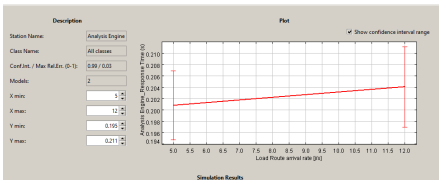

Figure 17: response time of route optimization engine

in turn would decrease its response time.

2. Throughput of the Analytics and Optimization Engine

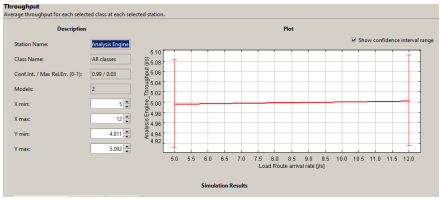

Figure 18: throughput of analytics engine

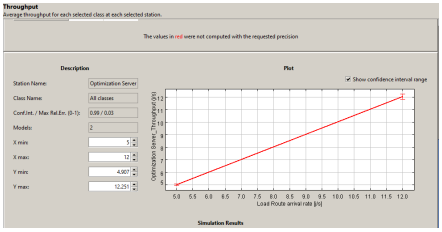

Figure 19: throughput of route optimization server

Our analytics engine doesn't seem to be affected by the increase in the arrival rate of the load route job, but the optimization engine is able to serve more jobs per unit of time because more jobs are coming.

3. Utilization of Database, Display, Optimization Server, Analytics Engine

Around 9 jobs/sec, the database can't meet the requirements we set, which is a utilization of 60%. The utilization of the optimization server has increased to 3%, but it doesn't affect the requirements. The utilization of the display has increased but is still below the threshold we set. When it comes to the analytics engine, it doesn't seem to be much affected by the increase in the number of load route jobs.

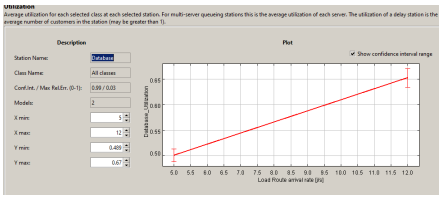

Figure 20: utilization of database

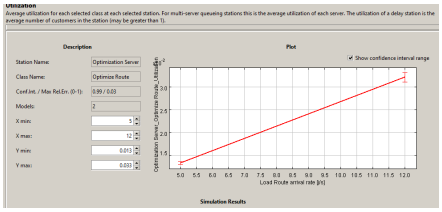

Figure 21: Utilization of route optimization engine

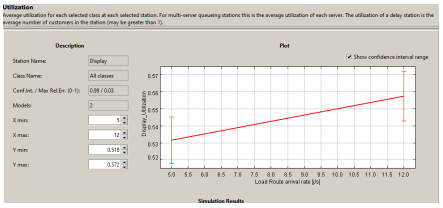

Figure 22: utilization of display

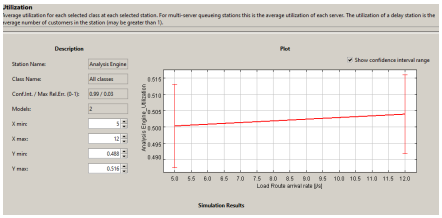

Figure 23: Utilization of analytics engine

8 Conclusion

Through our analysis using both stand-alone and dynamic performance models, we have determined that the system's performance generally meets our requirements. However, we have identified discrepancies in server utilization, with some servers being over-utilized and others under-utilized. This suggests that while the system performs adequately, there's room for improvement. Refactoring the system's design, considering these utilization patterns, could enhance efficiency and performance, aligning resource distribution more closely with demand.

9 References

1. G. Casale, G. Serazzi, *Java Modelling Tools User Manual*, November 3, 2023, for JMT version 1.2.5 and later.

2. Smith, Williams, *Performance Solutions: A Practical Guide to Creating Responsive, Scalable Software*.