2024

WSC 2024
Simulation Challenge
Tech Document

NUS National University of Singapore
Centre of Excellence in Modelling and Simulation for Next Generation Ports
Industrial Systems Engineering and Management
College of Design and Engineering

University of Exeter Engineering

SEALOR TECH

SUMMARY

The Simulation Challenge at the **Winter Simulation Conference (WSC)**, evolving from its initial launch in 2022 as the "Case Study Competition", continues to underscore the critical role of simulation in bridging the gap between academia and industry. This initiative thrives on promoting interdisciplinary collaboration, sparking innovation, and addressing real-world challenges with advanced simulation solutions.

This year, aligning with the WSC 2024 theme “Simulation for the Imagination Age”, the Challenge invites creators to unleash the potential of simulation technology in transforming maritime systems. As we navigate a world increasingly reliant on complex maritime logistics for global trade, we are shifting our focus towards empowering participants to reimagine the future of port operations, yard management, and cargo logistics. The challenge encourages participants to harness their creativity to design, optimize, and evaluate dynamic systems using simulation, crafting solutions that meet the ever-evolving demands of the maritime industry with ingenuity and foresight.

This year's Challenge pivots towards maritime logistics, focusing on a cargo port operation issue involving complex interactions and activities within a cargo port, from ship berthing to container handling and automated guided vehicle (AGV) management. Participants are invited to optimize cargo port operations to improve efficiency and foster creative solutions. We have built a basic simulation system model with discrete-event modeling methods. The model is implemented in C# programming language using O2DES.NET which is a set of frameworks for object-oriented discrete event simulation. In this competition, participants are required to build, design, and implement their own strategy algorithm. Participants are expected to embed their own algorithms in certain parts of the model, using their skills in model training and large-scale search, with the objective of improving port operation performance. The team that generate the best overall performance stand the best chance to win.

The purpose of this document is to provide the participants with detailed descriptions of the model structure and to guide them through the process of downloading the files and the required procedures for submitting their solutions.

TABLE OF CONTENTS

SUMMARY ..... **i**

TABLE OF CONTENTS .....

Chapter 1. Overview **1**

1.1 Problem Description 1

1.2 Competition Rules 2

1.3 General Conditions of the Challenge 3

Chapter 2. Technichal Instractions **4**

2.1 Installing Visual Studio 4

2.2 Source Code (in C#) 4

Chapter 3. Discrete-Event Simulation Model **6**

3.1 Entities 6

3.1.1 Entity Relationships 7

3.1.2 Entity Attributes and Definitions 7

3.2 Event Flow Diagram 11

3.2.1 Berth and Vessel 13

3.2.2 Vessel and QC Line 13

3.2.3 QC Line and Container 13

3.2.4 Vessel and QC 14

3.2.5 Container and QC 14

3.2.6 Container and AGV 15

3.2.7 Container and YC 16

3.2.8 AGV and YC 17

Chapter 4. Assessment Precedure **18**

4.1 File Submission Format 18

4.2 File Submission Method 18

4.3 Assessment Criteria 19

Appendices **20**

Appendix A: Information about distances between control points 20

Appendix B: Explanation of transhipment.csv 22

Chapter 1.

Overview

1.1 Problem Description

Welcome to the **WCS Simulation Challenge 2024** where this year's challenge is dedicated to simulating port operations. As global trade escalates, the efficiency of port operations becomes critical for economic continuity and competitiveness. Participants are invited to delve into the complex management of a container port terminal, emphasizing the optimization of interactions between vessels, containers, and port machinery including quay cranes (QCs), yard cranes (YCs), and automated guided vehicles (AGVs).

Traditional port operations often struggle with inefficiencies stemming from manual processes and static scheduling. These conventional methods can lead to significant delays in vessel berthing, container misallocation, and suboptimal usage of AGVs and cranes. In this competition, for each vessel arriving at the port, a waiting time exceeding **two hours** before berthing is considered a **failure**. Our goal is to reduce such failures and the final assessment of solutions submitted for this competition will be based on this performance metric.

In response to these challenges, our competition introduces a simplified but comprehensive port model. This model is centered around three critical decision variables: scheduling vessel berths, allocating containers to specific yard blocks, and assigning AGVs. These variables traditionally rely on fixed schedules and limited real-time data, leading to inefficiencies in throughput and resource allocation. This year, we encourage participants to reimagine these operations using advanced, innovative methods. By employing cutting-edge technologies and methodologies such as simulation, deep learning, and other data-driven approaches, contestants have the opportunity to significantly enhance the port's operational efficiency.

This competition is not only about devising winning strategies but also about fostering learning and discovery in port logistics. We aim for the participants to gain deep insights about the operational complexities of ports and to explore creative, efficient solutions to these real-world challenges. It is an opportunity to apply the theoretical knowledge practically, which may lead to not just personal growth but

potentially transformative changes within the industry. We look forward to seeing the unique perspectives and innovative solutions that each participant will bring to the WCS Simulation Challenge 2024. So, engage in this exploration of modern port operations, apply your skills, and enjoy the process of pioneering the future of port logistics!

Good luck and have fun in the WCS Simulation Challenge 2024!

1.2 Competition Rules

As shown in Figure 1, the given model contains data and source code according to the following three building blocks of information flow:

(A) Input: Example simulation scenarios with specific parameters settings.

(B) Interface: The three decision-modules. Here is where users can modify the code for their own decision algorithms for scheduling vessel berths, allocating containers to specific yard blocks, and the assignment of AGVs. The respective names for the methods are "CustomeizedAllocatedBerth", "CustomeizedDetermineYardBlock", and "CustomeizedAllocatedAGVs". The default algorithms are be provided too.

(C) Output: Performance indicators to measure the efficiency and quality of the port operations.

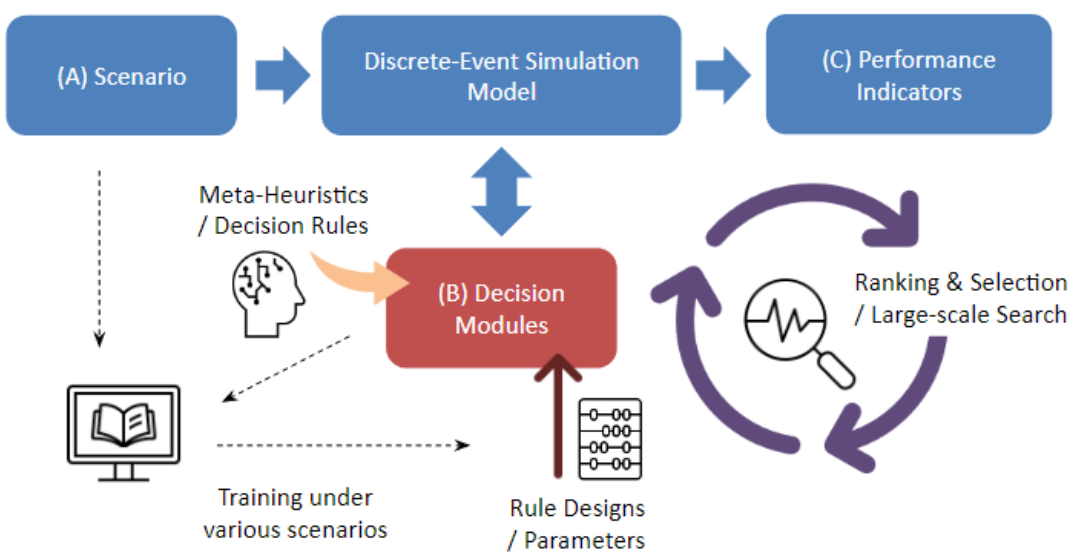


Figure 1 – Information flow diagram.

In this contest, you are expected to rewrite and replace the existing decision modules (in Block B), to maximize the archived performance of the system (in Block C) under different scenarios (from Block A).

You can generate the logic rules in Part (B) in various ways, including but not limited to:

a. Writing rule-based scripts or heuristic algorithms embedded in decision events,

b. Embedding this simulation model into some external optimization search algorithms,

c. Developing a machine learning model to identify and conclude the best rule parameters and embedding them in the decision events.

You **only** need to provide **Part (B)** of the program code (with any required data), and there is no need to submit your optimization and training programs.

Please Note: Other source codes (except those for Part B) will not be evaluated or run by the assessors.

1.3 General Conditions of the Challenge

Your data and program code will be embedded in the discrete-event simulation model (that we have provided in advance). It will overwrite the corresponding original code, then it will be compiled and generates an executable simulation program. Your program will run under a variety of scenarios and random seeds. The winner will be the one whose model generates the top average performance index for each case.

For further instructions regarding file downloads and programming please check Chapter 2. Simulation model structures and the available data is explained in Chapter 3. Last but not least, the submission guidelines and detailed assessment criteria for this competition can be found in Chapter 4.

Chapter 2.

Technichal Instractions

2.1 Installing Visual Studio

1) Go to Microsoft website https://visualstudio.microsoft.com/ Then, choose the version that suits your computer and click the **Download** button to download This IDE.

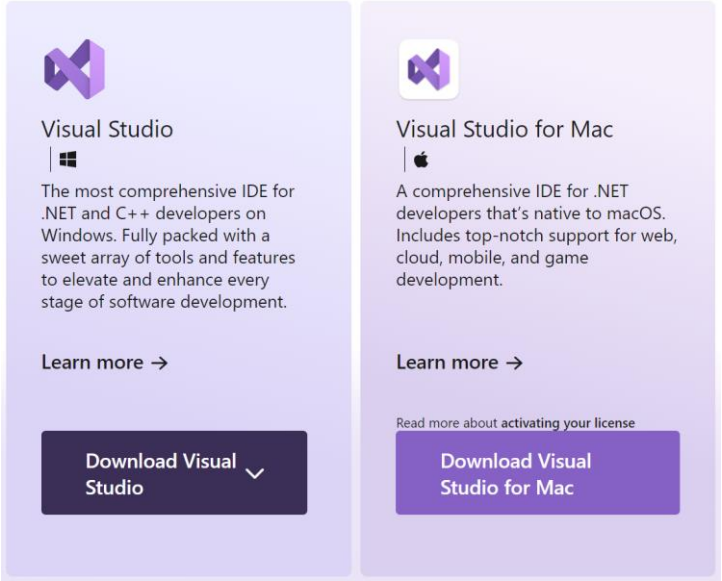


Figure 2 - Microsoft Visual Studio download page.

2) For detailed instructions on the installation steps, you may refer to the following links:

Windows OS: https://learn.microsoft.com/en-us/visualstudio/install/install-visual-studio?view=vs-2022

Mac OS: https://learn.microsoft.com/en-us/visualstudio/mac/installation?view=vsmac-2022

3) You should select the workload to run the provided simulation model using **C# language**.

2.2 Source Code (in C#)

1) After registration, you will receive the zip package of our source code by email. The file will look like "**WSC Simulation Challenge 2024.zip**".

2) Unzip the package.

3) Figure 3 shows the structure of the provided Source code for the challenge:

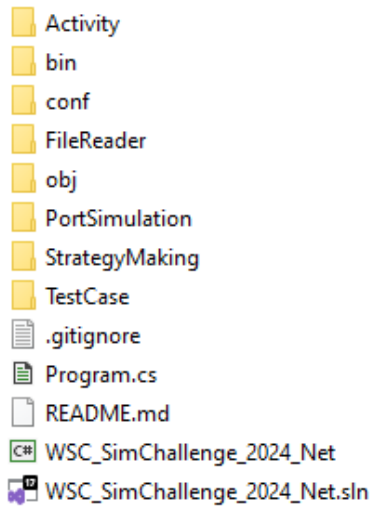


Figure 3- A glance on the content of the provided source code.

Here, the **conf** folder includes the **scenario files**, and **StrategyMaking** folder includes the **Default** and **DecisionMaker** files.

4) You may open the project by double click on **" WSC_SimChallenge_2024_Net.sln "**. For compiling and running the code, you may simply click on the green play button on the top part of the window (shown with a red arrow in Figure 4).

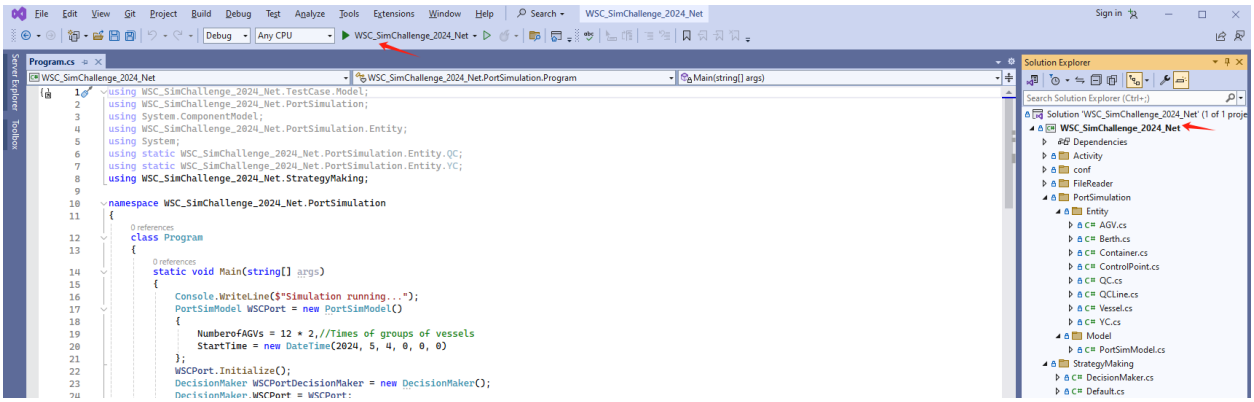


Figure 4- the provided C# code in Visual Studio.

Chapter 3. Discrete-Event Simulation Model

3.1 Entities

There are seven entities involved in this model, Quay Crane (QC), Berth, Container, Automated Guided Vehicles (AGV), Vessel, Yard Block and Yard Crane. Each is represented by a class and has its own set of attributes. The Entity Relationship Diagram (ERD) is shown in Figure 5 to give an overview of the relationships among the entities and the attributes associated with them.

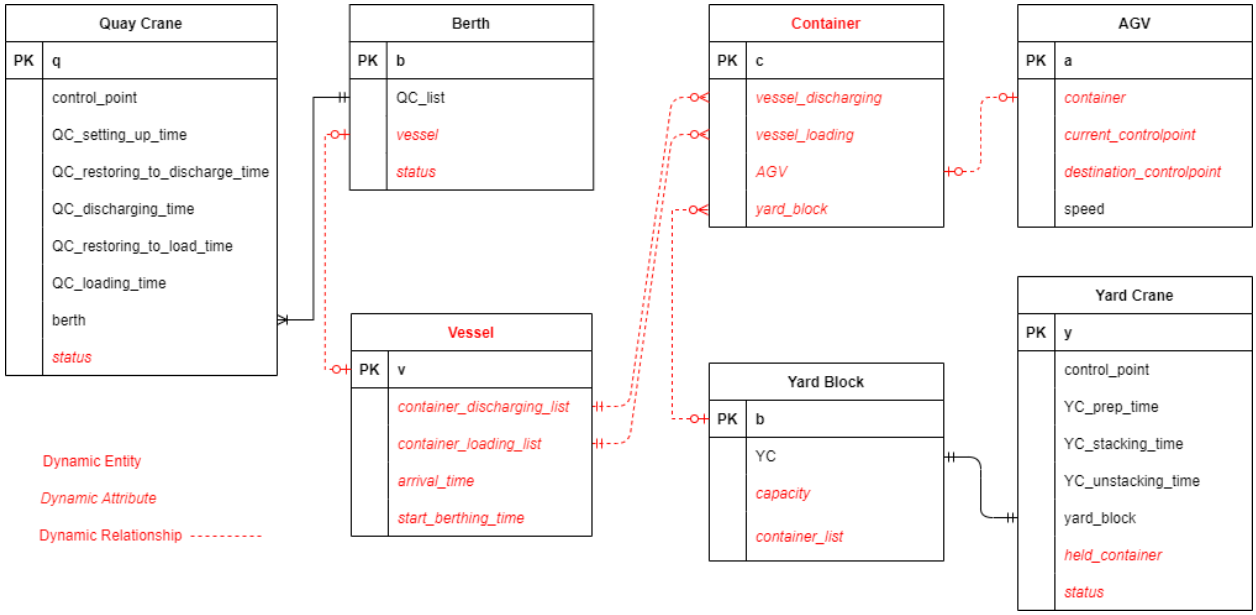


Figure 5- Entity Relationship Diagram (ERD).

Notes about Figure 5**:**

1) **PK:** primary key, which uniquely identifies the entity object.
2) **Static attribute:** entity attributes that do not change over time. These attributes are owned by the class itself (shown in black color in the ERD).
3) **Dynamic attribute (italic):** entity attributes that change over time (in red color).

3.1.1 Entity Relationships

The following relationships are in place between the mentioned entities of the system:

- Each Berth has one or many Quay Cranes.
- Each QC corresponds to exactly one Berth.
- Each Berth has either zero or one Vessel.
- Each Vessel corresponds to either zero or one Berth.
- Each Vessel has zero or many Containers.
- Each Container corresponds to exactly one Vessel.
- Each Container corresponds to either zero or one AGV.
- Each AGV can have either zero or one Container.
- Each Container corresponds to either zero or one Yard Block.
- Each Yard Block can have zero or many Containers.
- Each Yard Block has one and only one Yard Crane (i.e., each YC corresponds to one and only one Yard Block).

3.1.2 Entity Attributes and Definitions

The details about the entity attributes and their definitions are explained below. In the following tables, dynamic attributes are differentiated using italic font. Table 1 lists the attributes of Quay Crane that are all static except for its string attribute “status”.

Table 1 - Attributes of Quay Crane

Attribute Name	Type	Description
index	string	The primary key of Quay Crane.
control_point	ControlPoint	Control point of the Quay Crane.
QC_setting_up_time	double	Preparation time for Quay Crane before operation.
QC_restoring_to_discharge_time	double	Time taken for Quay Crane to restore for discharging a container.
QC_discharging_time	double	Time taken for Quay Crane to discharge a container.
QC_restoring_to_load_time	double	Time taken for Quay Crane to restore for loading a container.
QC_loading_time	double	Time taken for Quay Crane to load a container.
berth	Berth	Associated berth for the Quay Crane.
status	string	Current status of the Quay Crane, such as idle and occupied.

Table 2 summarizes the attributes of Berth.

Table 2 - Attributes of Berth

Attribute Name	Type	Description
index	string	The primary key of berth.
QC_list	List<Quay Crane>	List of quay cranes assigned to the berth.
vessel	Vessel	Vessel currently at the berth.
status	string	Current status of the berth, such as idle and occupied.

Table 3 summarizes the attributes of Vessel.

Table 3 - Attributes of Vessel

Attribute Name	Type	Description
index	string	The primary key of Vessel
container_discharging_list	List<Container>	List of containers being discharged from the vessel.
container_loading_list	List<Container>	List of containers being loaded onto the vessel.
arrival_time	DateTime	Actual arrival time of the vessel.
start_berthing_time	DateTime	The time when the vessel starts berthing.

Table 4 summarizes the attributes of Container.

Table 4 - Attributes of Container

Attribute Name	Type	Description
index	string	The primary key of Container.
vessel_discharing	Vessel	Vessel from which the container is being discharged.
vessel_loading	Vessel	Vessel to which the container is being loaded.
AGV	AGV	AGV transporting the container.
yard_block	Yard Block	Yard block where the container is stored.

Table 5 summarizes the attributes of AGV.

Table 5 - Attributes of AGV

Attribute Name	Type	Description
index	string	The primary key of AGV.
container	Container	Container currently being transported by the AGV.
current_controlpoint	ControlPoint	Current control point of the AGV.
Destination_controlpoint	ControlPoint	Destination control point of the AGV.
speed	double	Speed of the AGV.

Table 6 summarizes the attributes of Yard Block.

Table 6 - Attributes of Yard Block

Attribute Name	Type	Description
index	string	The primary key of Yard Block.
YC	Yard Crane	Yard crane corresponding to the yard block.
capacity	Integer	Capacity of the yard block.
Container_list	List<Container>	List of containers in the yard block.

Table 7 summarizes the attributes of Yard Crane.

Table 7 - Attributes of Yard Crane

Attribute Name	Type	Description
index	string	The primary key of Yard Crane.
control_point	ControlPoint	Control point of the Yard Crane.
YC_pre_time	double	Preparation time for Yard Crane before operations.
YC_stacking_time	double	Time taken for Yard Crane to stack a container.
YC_unstacking_time	double	Time taken for Yard Crane to unstack a container.
yard_block	Yard Block	Yard block where the yard crane operates.
held_container	Container	Container being held by the yard crane
status	string	Current status of the yard crane, such as idle and occupied.

3.2 Event Flow Diagram

The event flow diagram (EFD) in Figure 6 provides a clear depiction of our activity flow including entities, resource allocation, signal sources, etc. To help better understanding of this EFD, we will break it down and explain it separately for the interactions of each pair of our entities, i.e.,:

- Berth and Vessel
- Vessel and Quay Crane line (QC line)
- QC line and Container
- Vessel and Quay Crane (QC)
- Container and QC
- Container and AGV
- Container and Yard Crane (YC)
- AGV and YC

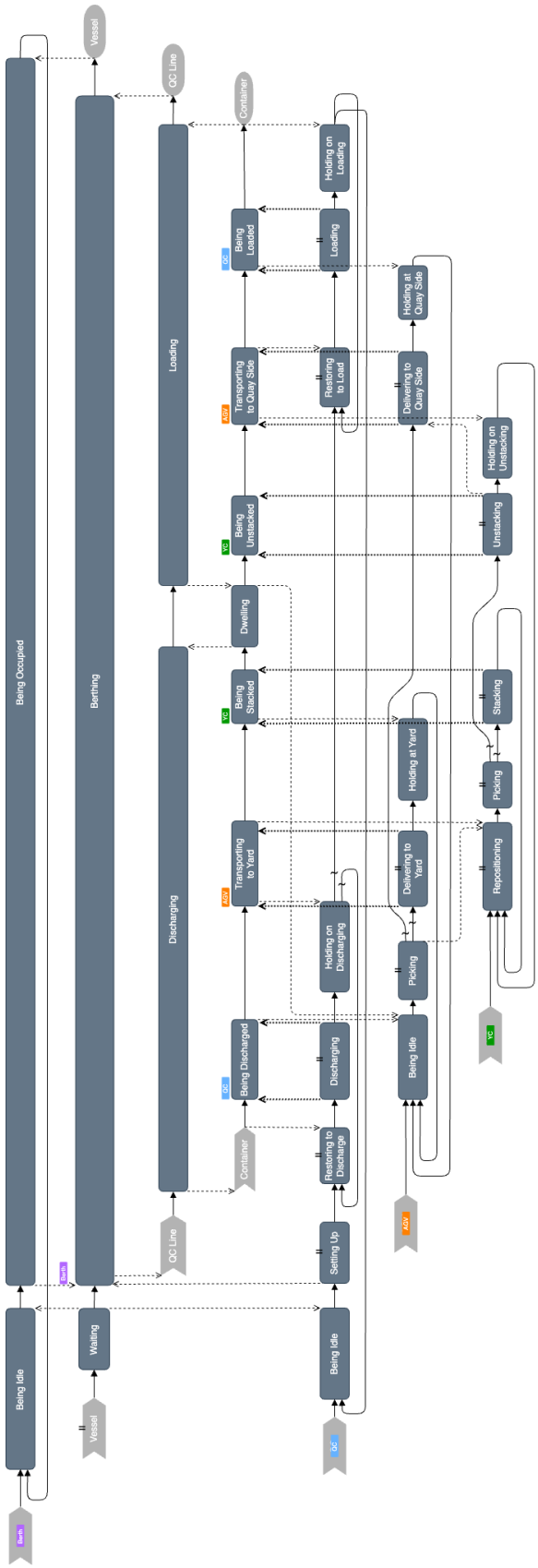


Figure 6 - Event Flow Diagram for the Simulation Model.

3.2.1 Berth and Vessel

Figure 7- Event Flow Diagram for Berth and Vessel.

When a vessel arrives, it first enters a “Waiting” status. It sends a signal to the idle berths, and once the berth is available, it sends a signal back to the vessel and confirms. At this point, if a QC is also allocated to the vessel (Vessel and QC will be introduced later), then the vessel transitions into the "Berthing" status, while the status of the berth changes from "Being Idle" to "Being Occupied". Upon completion of the berthing process, the vessel departs the system and notifies the associated berth of its departure, at which time the berth's status reverts to “Being Idle”.

3.2.2 Vessel and QC Line

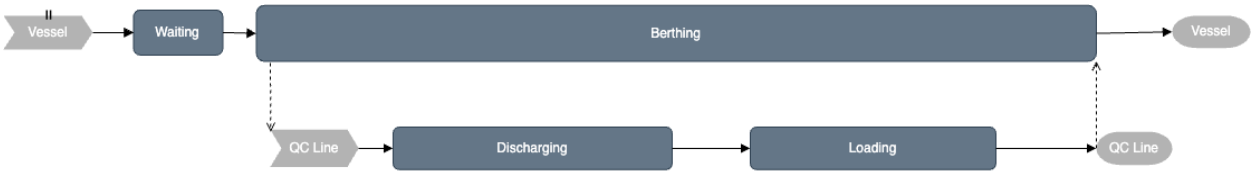


Figure 8- Event Flow Diagram for Vessel and QC Line.

As mentioned earlier, when a vessel arrives, it first enters a waiting state and then gets into berthing. During berthing, each quay crane creates a QC Line. Each QC line starts with “Discharging”. After all discharging operations are finished, that QC line transits to “Loading”, and this QC line ends once all loading tasks are done. When all QC lines end for each vessel, the vessel is informed that all required operations are finished and it can depart. Upon completion of these two phases, the QC line is eliminated.

3.2.3 QC Line and Container

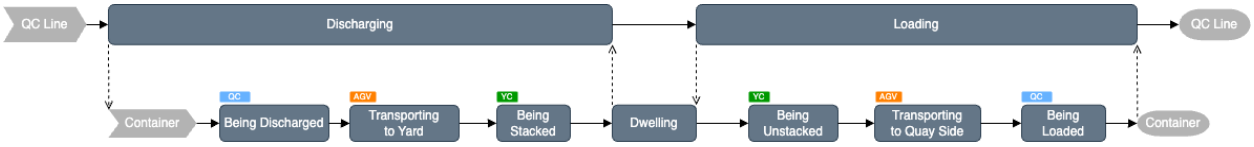


Figure 9- Event Flow Diagram for QC Line and Container.

The QC line processes each container individually. During the Discharging phase, each container undergoes "Being Discharged", "Transporting to Yard", and "Being Stacked" states until all containers of the vessel are moved to the "Dwelling" area. At this point, the Dwelling area signals back to the QC line, indicating that all discharging tasks are completed. The QC line then commences the Loading phase, where each container goes through "Being Unstacked", "Transporting to Quay Side", and "Being Loaded" states. Once all containers have been loaded, the QC line's Loading phase is finished, and the QC line ceases operation.

3.2.4 Vessel and QC

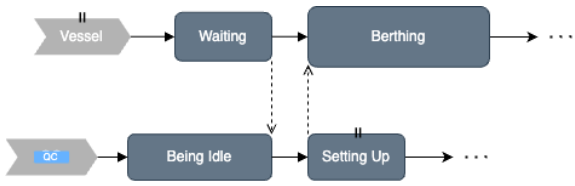


Figure 10- Event Flow Diagram for Vessel and QC.

When a vessel arrives, it first enters a waiting activity. It sends a signal to the idle QCs. Once the QCs confirm availability, a signal is sent back to the vessel. At this point, if there is a berth also allocated to the vessel, then the vessel transitions into the berthing status, and the QC is allocated to that vessel and starts to operate on it. Please note that in this model, each berth is equipped with three fixed QCs. Therefore, when a berth is allocated to a vessel, the corresponding QCs of that berth are also allocated to the vessel.

3.2.5 Container and QC

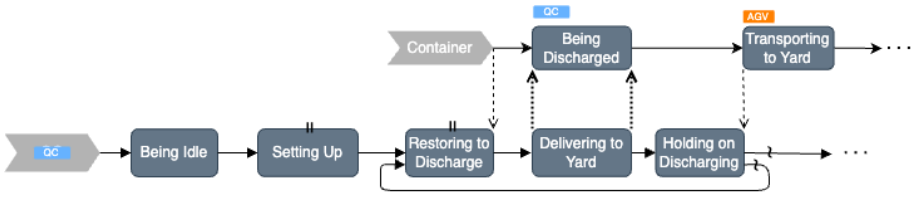


Figure 11- Event Flow Diagram for Container and QC - Part (a).

As seen in Figure 9, every task executed on the QC line for each container involves the joint operation of a QC, an AGV, and a YC. Initially the QC is idle, waiting for a new vessel arrival to serve. When the QC is called for discharging containers from the vessel, it is considered as "Being Occupied" and will move to its designated working place to start discharging containers from the vessel to the AGVs. When the AGV

arrives at the pickup point, the QC transfers the container to the AGV. The QC then continues to execute the task list on the QC line until all containers that need to be discharged are off the vessel.

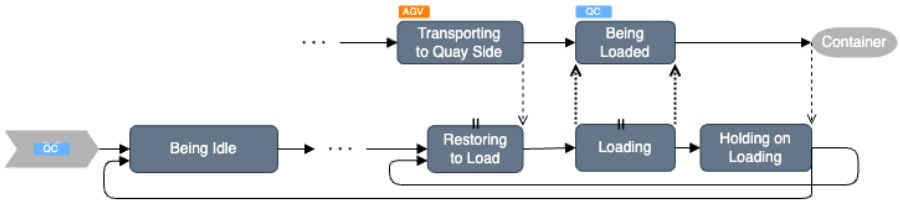


Figure 12- Event Flow Diagram for Container and QC - Part (b).

Similarly, when a QC completes the discharging process, there will be a call to QC for a loading work from the yard side to vessel. The QC will move to working position and load the containers from AGVs to the vessel. When all required containers are loaded to the vessel, the QC is released from the current working status and becomes "idle", waiting for a new call.

3.2.6 Container and AGV

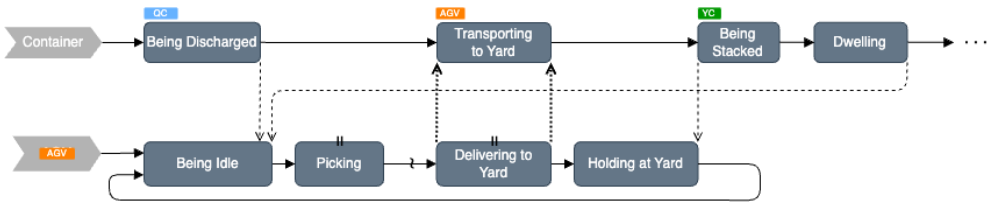


Figure 13- Event Flow Diagram for Container and AGV - Part (a).

The QC discharges the containers from the vessel and transfers them onto an AGV. The AGVs then transport the containers from the quayside to the specific storage blocks on the yard side. Upon the AGV's arrival at the designated storage blocks, it waits for the corresponding YC to remove the container from the AGV to the yard block. Once the container is taken by the YC, the AGV remains stationary, becomes "idle", and waits for the next call, exemplifying its responsive operation.

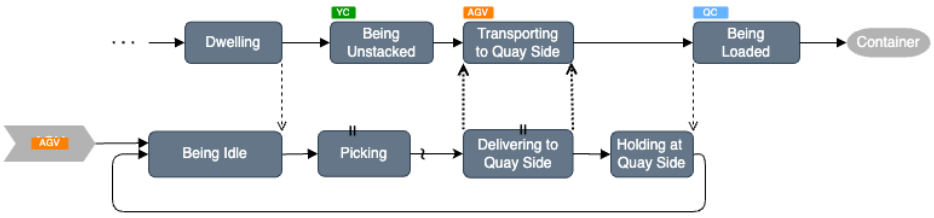


Figure 14- Event Flow Diagram for Container and AGV - Part (b).

For loading operation from yard side to vessel, AGV is called by the container that needs to be loaded. The empty AGV will move to appointed yard block. Once it arrives at the loading position, the calling container will be placed on the AGV by the YC of the yard, and then AGV carries the container to quay side. After arriving at the loading location, AGV queues for a QC to transfer the carried container from AGV to vessel. Once the container is obtained by the QC, the AGV is released and becomes "idle". It stays at the current control point and ready for a new call.

3.2.7 Container and YC

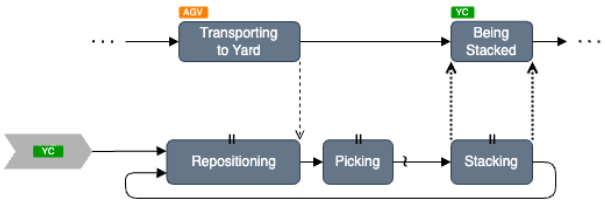


Figure 15- Event Flow Diagram for Container and YC- Part (a).

When YC is called into action as an AGV deliver containers to the yard block. YC is activated to transfer containers from AGV to designated slots in the storage block. After the container is placed in the specified block, the YC returns to the "Repositioning" and waits for the next call.

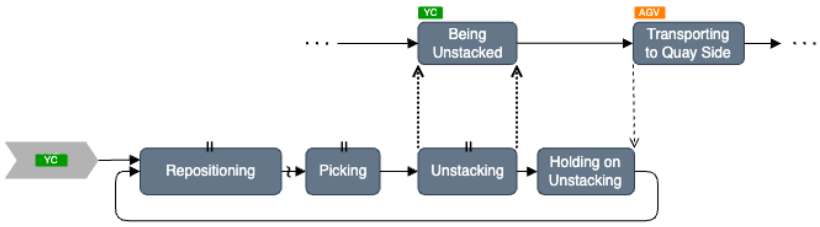


Figure 16- Event Flow Diagram for Container and YC- Part (b).

When loading a container onto a vessel, the process begins with the YC receiving a signal from the AGV. The YC then retrieves the container from the designated storage block. As the AGV arrives at the control point, the YC loads the container onto the AGV, which transports it to the quayside corresponding to the vessel's destination. Once the container is placed on the AGV, the loading task is considered complete from the YC's perspective, and the YC returns to "Repositioning", waiting for another call.

3.2.8 AGV and YC

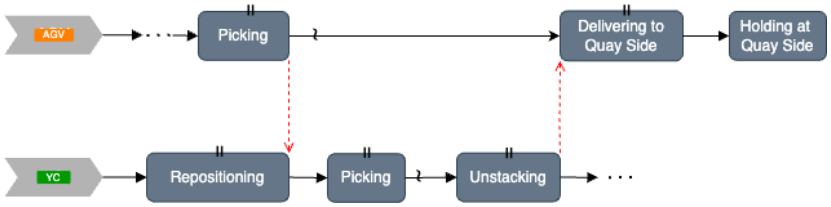


Figure 17- Event Flow Diagram for AGV and YC.

To prevent the occurrence of bugs, during the loading process, we require that the YC only begins to pick up a container when it is informed that an AGV is available to retrieve the container. Once the YC has completed unstacking the container onto the AGV, the AGV can then deliver the container to the Quay Side.

Chapter 4.

Assessment Precedure

4.1 File Submission Format

Please follow these two steps before submitting your files:

1) Zip package of the submitted folder

   Attention: Any modification should be put under the **StrategyMaking** folder, including newly created files for algorithms purposes and explanatory documents if necessary. Changes in other folders will not be considered.

2) Save your code and name it in the following format: **Team Name_Round Number** (e.g. **SealTeam_Round1.zip**)

4.2 File Submission Method

The solution of each team should be submitted via **Email** to the following address by the announced deadline of each round. In the **Title**, please include the **Team Name** and **Round Number**.

Submission Email Address: wsc2024SimChallenge@gmail.com

4.3 Assessment Criteria

1) Competitors' strategy will be adopted to simulate the given scenario as well as the hidden scenario for the last round.
2) "**Your final result**" from the output is the only criteria to evaluate system performance.
3) In the evaluation stage, multiple random seeds will be used to calculate the average performance for each round. And only codes that run successfully will get scores.
4) Weightage and score of each round:

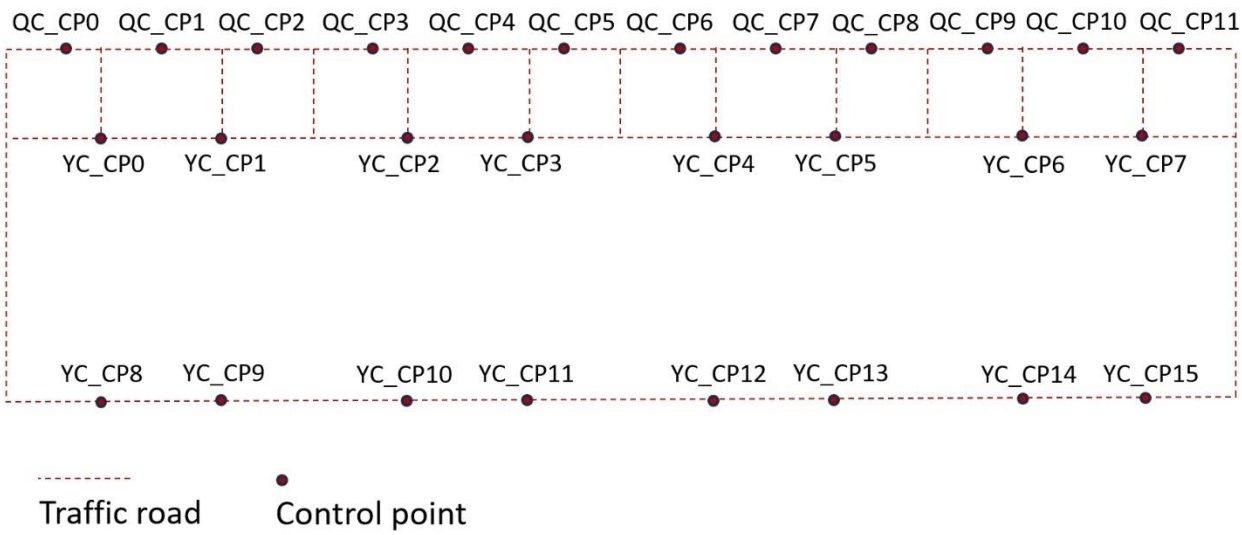
Weightage Table

Round Number	Round 1	Round 2	Round 3	Hidden Round
Weightage	5%	15%	30%	50%

The full score of each round is 100. The score for the top 10 teams will decrease by 5 from 100 according to team's rank (i.e. the number 1 ranking team will be scored as 100, the second ranking team 95, etc.). The rest of the teams after the top 10 teams will receive a score of 50.

Appendices

Appendix A: Information about distances between control points



The figure above shows the location of all control points in the layout. In the code, participants can get the distance between two control points by calling the `CalculateDistance(ControlPoint cp1, ControlPoint cp2)` function. Please note that regarding distance, we only consider the shortest routes in this competition.

```
AGV.cs
WSC_SimChallenge_2024_Net | WSC_SimChallenge_2024_Net.PortSimulation.Entity.AGV | Manhatt
5 references
public static double CalculateDistance(ControlPoint cp1, ControlPoint cp2)
{
    double distance = 0;
    if (Math.Abs(cp1.Ycoordinate - cp2.Ycoordinate) > 100)//travel between quayside and second row YC
    {
        double leftRouteDistance = ManhattanDistance(cp1, LeftNetworkCP) + ManhattanDistance(LeftNetworkCP, cp2);
        double rightRouteDistance = ManhattanDistance(cp1, RightNetworkCP) + ManhattanDistance(RightNetworkCP, cp2);

        distance = Math.Min(leftRouteDistance, rightRouteDistance);
    }
    else
        distance = ManhattanDistance(cp1, cp2);

    return distance;
}
```

The following figure shows the distance values between all pairs of control points.

	QC_CP0	QC_CP1	QC_CP2	QC_CP3	QC_CP4	QC_CP5	QC_CP6	QC_CP7	QC_CP8	QC_CP9	QC_CP10	QC_CP11	YC_CP0	YC_CP1	YC_CP2	YC_CP3	YC_CP4	YC_CP5	YC_CP6	YC_CP7	YC_CP8	YC_CP9	YC_CP10	YC_CP11	YC_CP12	YC_CP13	YC_CP14	YC_CP15
QC_CP0	0	133.33	266.66	400	533.33	666.66	800	933.33	1066.66	1200	1333.33	1466.66	166.66	300	566.66	700	966.66	1100	1366.66	1500	700	833.34	1100	1233.34	1500	1633.34	1900	2033.34
QC_CP1	133.33	0	133.33	266.67	400	533.33	666.67	800	933.33	1066.67	1200	1333.33	166.67	166.67	433.33	566.67	833.33	966.67	1233.33	1366.67	833.33	966.67	1233.33	1366.67	1633.33	1766.67	2033.33	2033.33
QC_CP2	266.66	133.33	0	133.34	266.67	400	533.34	666.67	800	933.34	1066.67	1200	300	166.66	300	433.34	700	833.34	1100	1233.34	966.66	1100	1366.66	1500	1766.66	1900	2033.34	1900
QC_CP3	400	266.67	133.34	0	133.33	266.66	400	533.33	666.66	800	933.33	1066.66	433.34	300	166.66	300	566.66	700	966.66	1100	1100	1233.34	1500	1633.34	1900	2033.34	1900	1766.66
QC_CP4	533.33	400	266.67	133.33	0	133.33	266.67	400	533.33	666.67	800	933.33	566.67	433.33	166.67	166.67	433.33	566.67	833.33	966.67	1233.33	1366.67	1633.33	1766.67	2033.33	2033.33	1766.67	1633.33
QC_CP5	666.66	533.33	400	266.66	133.33	0	133.34	266.67	400	533.34	666.67	800	700	566.66	300	166.66	300	433.34	700	833.34	1366.66	1500	1766.66	1900	2033.34	1900	1633.34	1500
QC_CP6	800	666.67	533.34	400	266.67	133.34	0	133.33	266.66	400	533.33	666.66	833.34	700	433.34	300	166.66	300	566.66	700	1500	1633.34	1900	2033.34	1900	1766.66	1500	1366.66
QC_CP7	933.33	800	666.67	533.33	400	266.67	133.33	0	133.33	266.67	400	533.33	966.67	833.33	566.67	433.33	166.67	166.67	433.33	566.67	1633.33	1766.67	2033.33	2033.33	1766.67	1633.33	1366.67	1233.33
QC_CP8	1066.66	933.33	800	666.66	533.33	400	266.66	133.33	0	133.34	266.67	400	1100	966.66	700	566.66	300	166.66	300	433.34	1766.66	1900	2033.34	1900	1633.34	1500	1233.34	1100
QC_CP9	1200	1066.67	933.34	800	666.67	533.34	400	266.67	133.34	0	133.33	266.66	1233.34	1100	833.34	700	433.34	300	166.66	300	1900	2033.34	1900	1766.66	1500	1366.66	1100	966.66
QC_CP10	1333.33	1200	1066.67	933.33	800	666.67	533.33	400	266.67	133.33	0	133.33	1366.67	1233.33	966.67	833.33	566.67	433.33	166.67	166.67	2033.33	2033.33	1766.67	1633.33	1366.67	1233.33	966.67	833.33
QC_CP11	1466.66	1333.33	1200	1066.66	933.33	800	666.66	533.33	400	266.66	133.33	0	1500	1366.66	1100	966.66	700	566.66	300	166.66	2033.34	1900	1633.34	1500	1233.34	1100	833.34	700
YC_CP0	166.66	166.67	300	433.34	566.67	700	833.34	966.67	1100	1233.34	1366.67	1500	0	133.34	400	533.34	800	933.34	1200	1333.34	666.66	800	1066.66	1200	1466.66	1600	1866.66	2000
YC_CP1	300	166.67	166.66	300	433.33	566.66	700	833.33	966.66	1100	1233.33	1366.66	133.34	0	266.66	400	666.66	800	1066.66	1200	800	933.34	1200	1333.34	1600	1733.34	2000	1866.66
YC_CP2	566.66	433.33	300	166.66	166.67	300	433.34	566.67	700	833.34	966.67	1100	400	266.66	0	133.34	400	533.34	800	933.34	1066.66	1200	1466.66	1600	1866.66	2000	1733.34	1600
YC_CP3	700	566.67	433.34	300	166.67	166.66	300	433.33	566.66	700	833.33	966.66	533.34	400	133.34	0	266.66	400	666.66	800	1200	1333.34	1600	1733.34	2000	1866.66	1600	1466.66
YC_CP4	966.66	833.33	700	566.66	433.33	300	166.66	166.67	300	433.34	566.67	700	800	666.66	400	266.66	0	133.34	400	533.34	1466.66	1600	1866.66	2000	1733.34	1600	1333.34	1200
YC_CP5	1100	966.67	833.34	700	566.67	433.34	300	166.67	166.66	300	433.33	566.66	933.34	800	533.34	400	133.34	0	266.66	400	1600	1733.34	2000	1866.66	1600	1466.66	1200	1066.66
YC_CP6	1366.66	1233.33	1100	966.66	833.33	700	566.66	433.33	300	166.66	166.67	300	1200	1066.66	800	666.66	400	266.66	0	133.34	1866.66	2000	1733.34	1600	1333.34	1200	933.34	800
YC_CP7	1500	1366.67	1233.34	1100	966.67	833.34	700	566.67	433.34	300	166.67	166.66	1333.34	1200	933.34	800	533.34	400	133.34	0	2000	1866.66	1600	1466.66	1200	1066.66	800	666.66
YC_CP8	700	833.33	966.66	1100	1233.33	1366.66	1500	1633.33	1766.66	1900	2033.33	2033.34	666.66	800	1066.66	1200	1466.66	1600	1866.66	2000	0	133.34	400	533.34	800	933.34	1200	1333.34
YC_CP9	833.34	966.67	1100	1233.34	1366.67	1500	1633.34	1766.67	1900	2033.34	2033.33	1900	800	933.34	1200	1333.34	1600	1733.34	2000	1866.66	133.34	0	266.66	400	666.66	800	1066.66	1200
YC_CP10	1100	1233.33	1366.66	1500	1633.33	1766.66	1900	2033.33	2033.34	1900	1766.67	1633.34	1066.66	1200	1466.66	1600	1866.66	2000	1733.34	1600	400	266.66	0	133.34	400	533.34	800	933.34
YC_CP11	1233.34	1366.67	1500	1633.34	1766.67	1900	2033.34	2033.33	1900	1766.66	1633.33	1500	1200	1333.34	1600	1733.34	2000	1866.66	1600	1466.66	533.34	400	133.34	0	266.66	400	666.66	800
YC_CP12	1500	1633.33	1766.66	1900	2033.33	2033.34	1900	1766.67	1633.34	1500	1366.67	1233.34	1466.66	1600	1866.66	2000	1733.34	1600	1333.34	1200	800	666.66	400	266.66	0	133.34	400	533.34
YC_CP13	1633.34	1766.67	1900	2033.34	2033.33	1900	1766.66	1633.33	1500	1366.66	1233.33	1100	1600	1733.34	2000	1866.66	1600	1466.66	1200	1066.66	933.34	800	533.34	400	133.34	0	266.66	400
YC_CP14	1900	2033.33	2033.34	1900	1766.67	1633.34	1500	1366.67	1233.34	1100	966.67	833.34	1866.66	2000	1733.34	1600	1333.34	1200	933.34	800	1200	1066.66	800	666.66	400	266.66	0	133.34
YC_CP15	2033.34	2033.33	1900	1766.66	1633.33	1500	1366.66	1233.33	1100	966.66	833.33	700	2000	1866.66	1600	1466.66	1200	1066.66	800	666.66	1333.34	1200	933.34	800	533.34	400	133.34	0

Centre of Excellence in Modelling and Simulation for Next Generation Ports
Industrial Systems Engineering and Management
College of Design and Engineering
NUS National University of Singapore
University of Exeter Engineering

Appendix B: Explanation of transhipment.csv

	vessel 0	vessel 1	vessel 2	vessel 3
vessel 0	0	2	3	10
vessel 1	9	0	5	5
vessel 2	2	9	0	4
vessel 3	8	10	6	0

The table above assumes that four vessels will arrive each week. Each row represents the number of containers that a vessel is scheduled to discharge. For example, the vessel 0 highlighted in the red box will discharge a total of 2+3+10 containers. These containers will be loaded onto vessels 1, 2, and 3, with 2, 3, and 10 containers respectively.

	vessel 0	vessel 1	vessel 2	vessel 3
vessel 0	0	2	3	10
vessel 1	9	0	5	5
vessel 2	2	9	0	4
vessel 3	8	10	6	0

Similarly, each column represents the number of containers that a vessel is scheduled to load. For example, the vessel 0 enclosed in the blue box will load a total of 9+2+8 containers. These containers are sourced from vessels 1, 2, and 3, which respectively had discharged 9, 2, and 8 containers.