Winter Simulation Conference 2025 Tech Document

WSC 2025 Simulation Challenge Tech Document

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

University of Exeter

SUMMARY

The Simulation Challenge at the Winter Simulation Conference (WSC), originally launched in 2022 as the "Case Study Competition," continues to showcase the crucial role of simulation in bridging academia and industry. This initiative fosters interdisciplinary collaboration, sparks innovation, and addresses real-world challenges through advanced simulation technologies.

In 2025, aligning with the WSC theme "*Look to the Future! Simulation 2050 and Beyond*" the Challenge will focus on transforming maritime systems, particularly in port operations and cargo logistics. As global trade increasingly depends on complex maritime logistics, participants are encouraged to unleash their creativity in designing, optimizing, and evaluating dynamic port systems to meet the evolving demands of the industry. This competition offers a unique opportunity to develop forward-thinking solutions that could shape the future of port operations.

This year's Challenge continues the theme of maritime logistics, with an emphasis on cargo port operations, from vessel berthing to container handling and Automated Guided Vehicle (AGV) management. The simulation model has been further upgraded, and participants are encouraged to explore advanced methods beyond rule-based policies. The competition now promotes the integration of optimization algorithms and reinforcement learning (RL) techniques fostering innovation and development of effective and intelligent solutions. This marks an exciting evolution, as AI technologies and their applications become increasingly central to the future of port logistics.

Participants are invited to optimize cargo port operations with the aim of enhancing efficiency and fostering creative solutions. A foundational simulation model has been built using discrete-event modeling methods and implemented in Python with the o2despy framework, an object-oriented discrete event simulation. Participants are required to choose either of them to build design and implement their own strategy algorithm, competing against other teams. Participants are expected to embed their custom algorithms into specific components of the model, applying their skills in model training and large-scale search, to improve the port operation performance. The teams that achieve the best overall performance will have the best chance of winning the competition.

This document is intended to provide participants with detailed descriptions of the simulation model structure along with step-by-stepguidance onfile downloads and submission procedures. Through this Challenge, we aim to inspire innovative solutions that can amke a lasting impact on the future of the maritime industry.

CONTENTS

Overview

1.1 Problem Description

The 2025 Simulation Challenge continues its focus on maritime logistics, addressing a complex and dynamic problem in cargo port operations. This year's challenge centers s on optimizing the interactions and operations within a cargo port, spanning processes from ship berthing to container handling and the management of Automated Guided Vehicle (AGV). Participants are invited to propose innovative and effective solutions that enhance port operational efficiency through simulation and optimization.

Traditional port operations often suffer from inefficiencies due to static scheduling, manual processes, and limited real-time data. These challenges can result in vessel berthing delays, container misallocation, and undrutilization of AGVs and cranes. This year, the primary objective of the Challenge is to optimize the average service time for vessels, from their arrival att the port to their departure by developing smarter and more adaptive operational strategies

To address the challenges of traditional port operations, the competition introduces a simplified yet comprehensive port simulation model. The model focuses on three critical decision variables: scheduling vessel berths, allocating containers to yard blocks, and assigning AGVs. These decisions typically are based on fixed schedules and limited real-time data, leading to inefficiencies in throughput and resource allocation. Participants are encouraged to rethink these operations using innovative data-driven approaches. The upgraded the model supportsexploration of advanced techniques beyond ruled-based policies, including optimization algorithms and reinforcement learning (RL). This opens the door to leverage cutting-edge technologies such as simulation, deep learning, and other AI-driven methods to significantly improve port operational efficiency and perfomance.

This competition is not only about devising winning strategies but also about fostering learning and discovery in the field of port logistics. We hope participants will gain a deep understanding of the complexities involved inport operations while exploring creative and efficient solutions to these real-world challenges. The 2025 WSC Simulation Challenge offers unique opportunity to apply theoretical knowledge in a practical setting, fostering personal growth and potentially driving transformative change in the industry.

We look forward to seeing the unique perspectives and innovative ideas each team will bring.

Join us in shaping the future of port operations, apply your skills, embrace the challenge, and enjoy the process of pioneering the future of port logistics.

Good luck and welcome to the exciting journey of the 2025 WCS Simulation Challenge!

1.2 Competition Rules

Participants are required to refine the decision-making modules within the provided simulation model to optimize cargo port operations, including but not limited to the following areas:

- Design and implement a control or decision-making system that minimizing total time (waiting + service).
- Participants are encouraged to:
  - Modify or enhance the provided demonstration modules (default, decision_maker_heuristic, decision_maker_learning) as desired.
  - Propose entirely new approaches or algorithms that operate independently of the provided demonstration modules.
  - Alter any component of the project (init, learning, etc.), **as long as the simulation logic and integrity remain intact.**
  - Insert additional interfaces or hooks into the simulation model, **as long as they do not disrupt existing logic.**
- Participants must ensure their submitted project:
  - Runs seamlessly without errors.
  - Clearly documents the implemented approach and modifications made to the original codebase.

- Submit **the complete project and experiment results,** clearly labeled and structured for ease of evaluation, ensuring the project can be executed without additional dependencies.

1.3 General Evaluation Guide and Support Overview

During the evaluation stage, the pre-specified undisclosed scenarios and random seeds will serve as the input of the programs. The organizers will then execute each participant's simulation program and record the output of performance index values. These outputs will serve as the basis for scoring and final rankings. Evaluation will focus exclusively on each solution's ability to minimize the total time (waiting + service) while maintaining the integrity of the simulation logic.

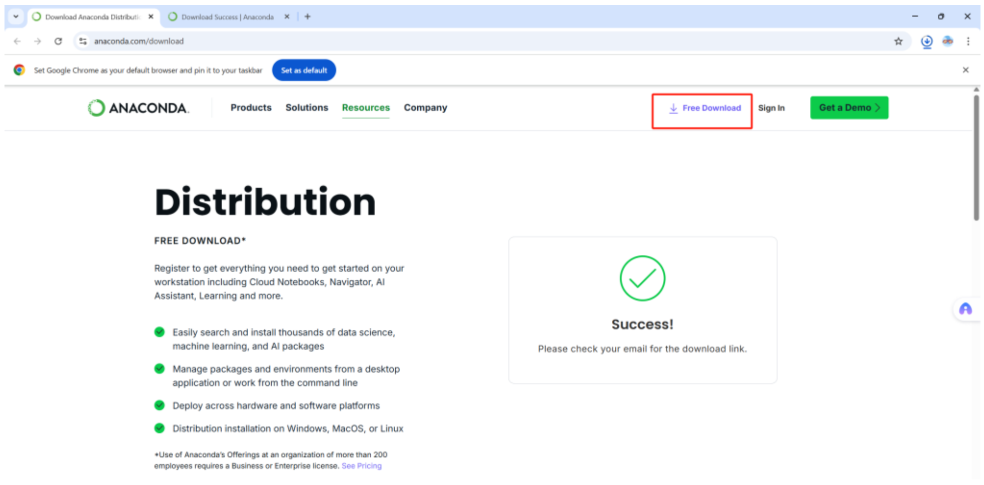
To support participants in developing their solutions, the organizers will provide two demo templates, each demonstrating a different approach: 1) Operational policymaking 2) Reinforcement learning

Comprehensive guidance on software installation and environment setup is provided in Chapter Two. Chapter Three outlines the model structure and the dataset in detail. For a discussion of decision makers analysis and optimization strategies, refer to Chapter Four. Submission procedures and evaluation criteria are thoroughly addressed in Chapter Five.

User Guidance

2.1 Download and install Anaconda

1) Go to website https://www.anaconda.com/download and click **Free Download** button to download Anaconda.


Download Anaconda Distributi
Download Success | Anaconda
anaconda.com/download
Set Google Chrome as your default browser and pin it to your taskbar
Set as default
ANACONDA
Products
Solutions
Resources
Company
Free Download
Sign In
Get a Demo

Distribution

FREE DOWNLOAD*

Register to get everything you need to get started on your workstation including Cloud Notebooks, Navigator, AI Assistant, Learning and more.

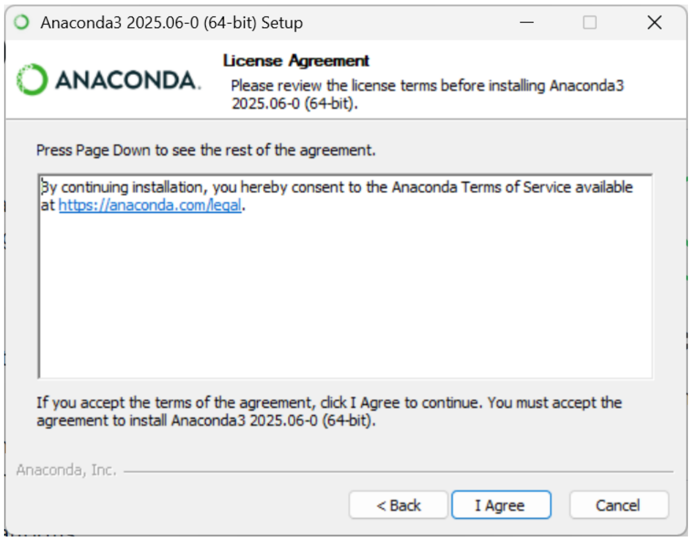
- Easily search and install thousands of data science, machine learning, and AI packages
- Manage packages and environments from a desktop application or work from the command line
- Deploy across hardware and software platforms
- Distribution installation on Windows, MacOS, or Linux

*Use of Anaconda's Offerings at an organization of more than 200 employees requires a Business or Enterprise license. See Pricing

Success!

Please check your email for the download link.

2) Follow the instructions to begin the installation process.



Anaconda3 2025.06-0 (64-bit) Setup

License Agreement

Please review the license terms before installing Anaconda3 2025.06-0 (64-bit).

Press Page Down to see the rest of the agreement.

By continuing installation, you hereby consent to the Anaconda Terms of Service available at https://anaconda.com/legal.

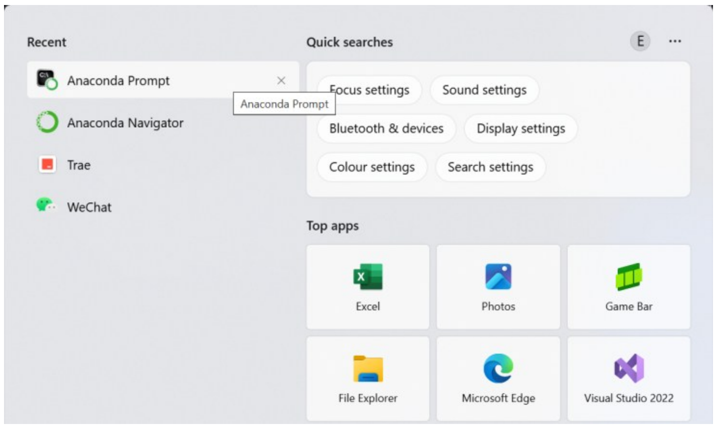
If you accept the terms of the agreement, click I Agree to continue. You must accept the agreement to install Anaconda3 2025.06-0 (64-bit).

Anaconda, Inc.

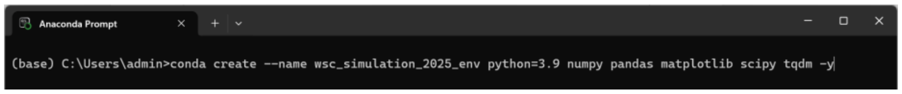
Wait for a while to complete the installation process and after successful setup, close the dialogue box

2.2 Build a virtual environment

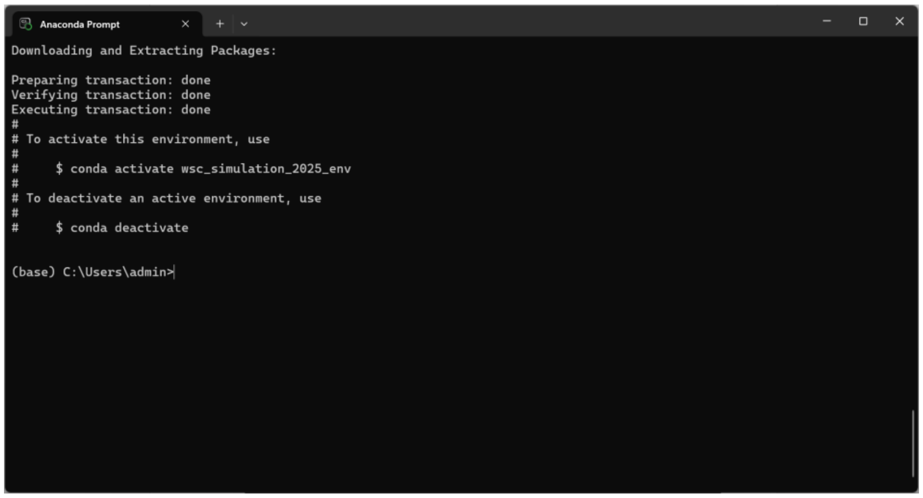
Click Anaconda Prompt from your system's Start Menu,



Then build a virtual environment “wsc_simulation_2025_env” and simultaneously install the necessary packages, such as numpy pandas matplotlib scipy, and tqdm.

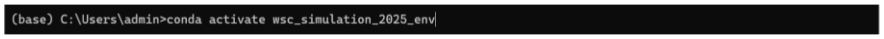


```
(base) C:\Users\admin>conda create --name wsc_simulation_2025_env python=3.9 numpy pandas matplotlib scipy tqdm -y
```

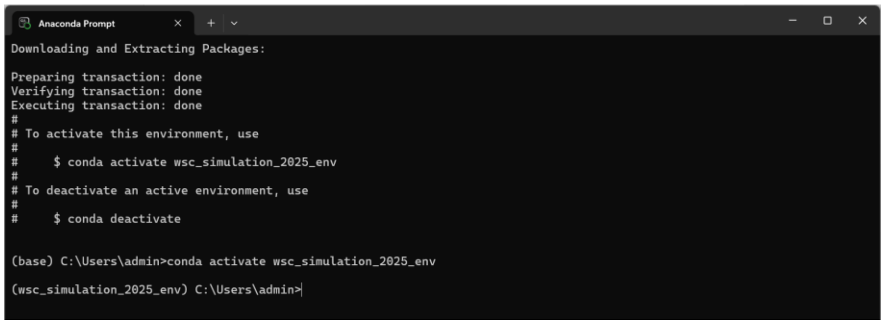


```
Downloading and Extracting Packages:

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
# To activate this environment, use
#
#     $ conda activate wsc_simulation_2025_env
#
# To deactivate an active environment, use
#
#     $ conda deactivate


(base) C:\Users\admin>
```

This process could last several minutes. After that, activate our virtual environment:



```
(base) C:\Users\admin>conda activate wsc_simulation_2025_env
```



```
Downloading and Extracting Packages:

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
# To activate this environment, use
#
#     $ conda activate wsc_simulation_2025_env
#
# To deactivate an active environment, use
#
#     $ conda deactivate


(base) C:\Users\admin>conda activate wsc_simulation_2025_env

(wsc_simulation_2025_env) C:\Users\admin>
```

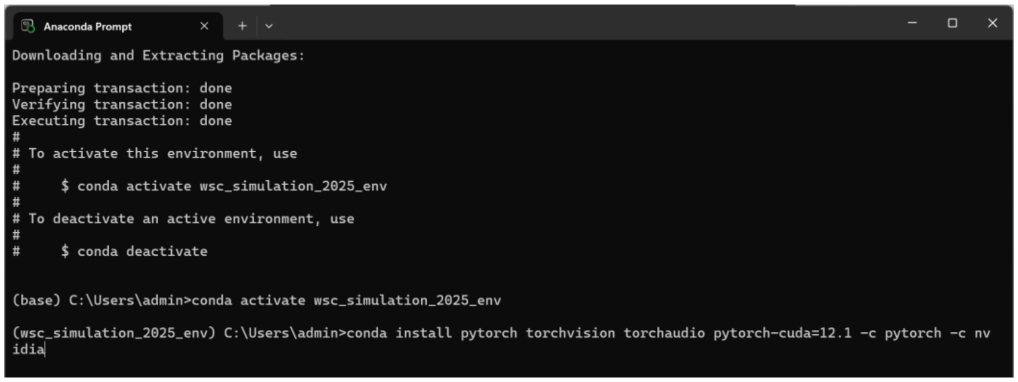
2.3 Install pytorch

This process depends on your hardware configuration:

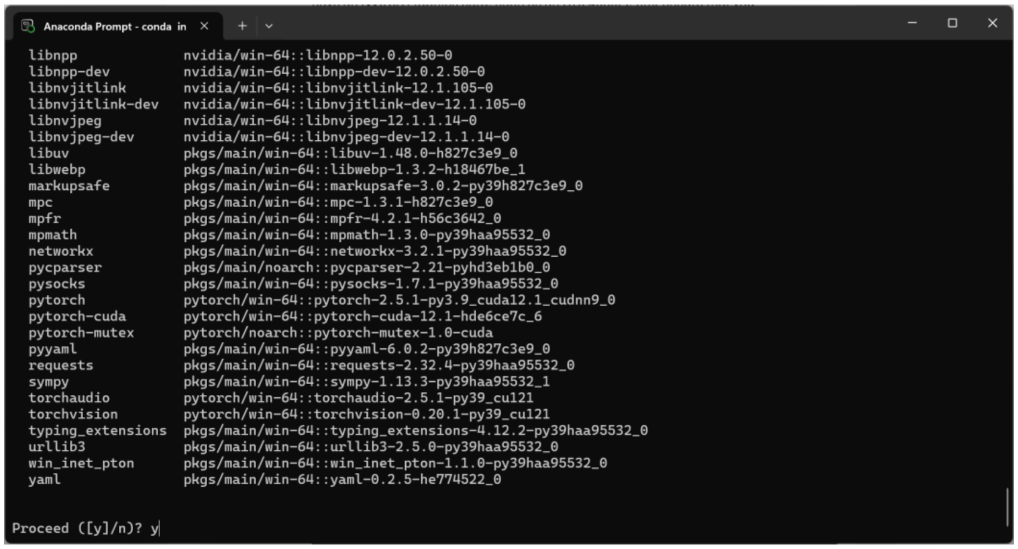
- **For CPU-only computation:** If your computer does not have a dedicated NVIDIA GPU, run the following command:

conda install pytorch torchvision torchaudio cpuonly -c pytorch

- **For NVIDIA GPU computation (Recommended for RTX 40 Series):** If you have an NVIDIA graphics card, such as an RTX 4080M, firstly ensure that you have the appropriate drivers installed. Then, to install PyTorch with GPU support, run the command below. For an RTX 4080M, using the version compiled with CUDA 12.1 is recommended: conda install pytorch torchvision torchaudio pytorch-cuda=12.1 -c pytorch -c nvidia



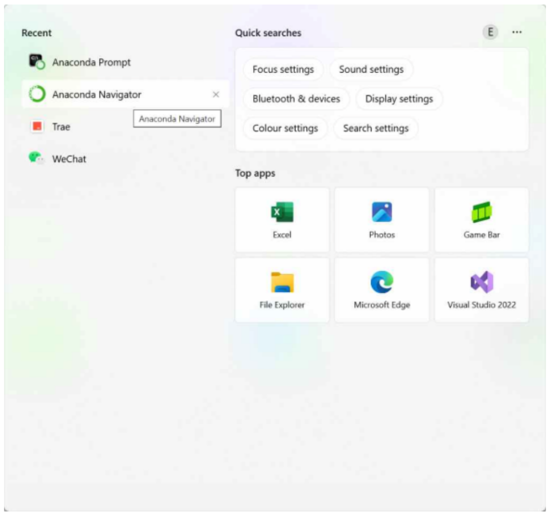
```
Downloading and Extracting Packages:

Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
# To activate this environment, use
#
#     $ conda activate wsc_simulation_2025_env
#
# To deactivate an active environment, use
#
#     $ conda deactivate


(base) C:\Users\admin>conda activate wsc_simulation_2025_env

(wsc_simulation_2025_env) C:\Users\admin>conda install pytorch torchvision torchaudio pytorch-cuda=12.1 -c pytorch -c nv
idia
```



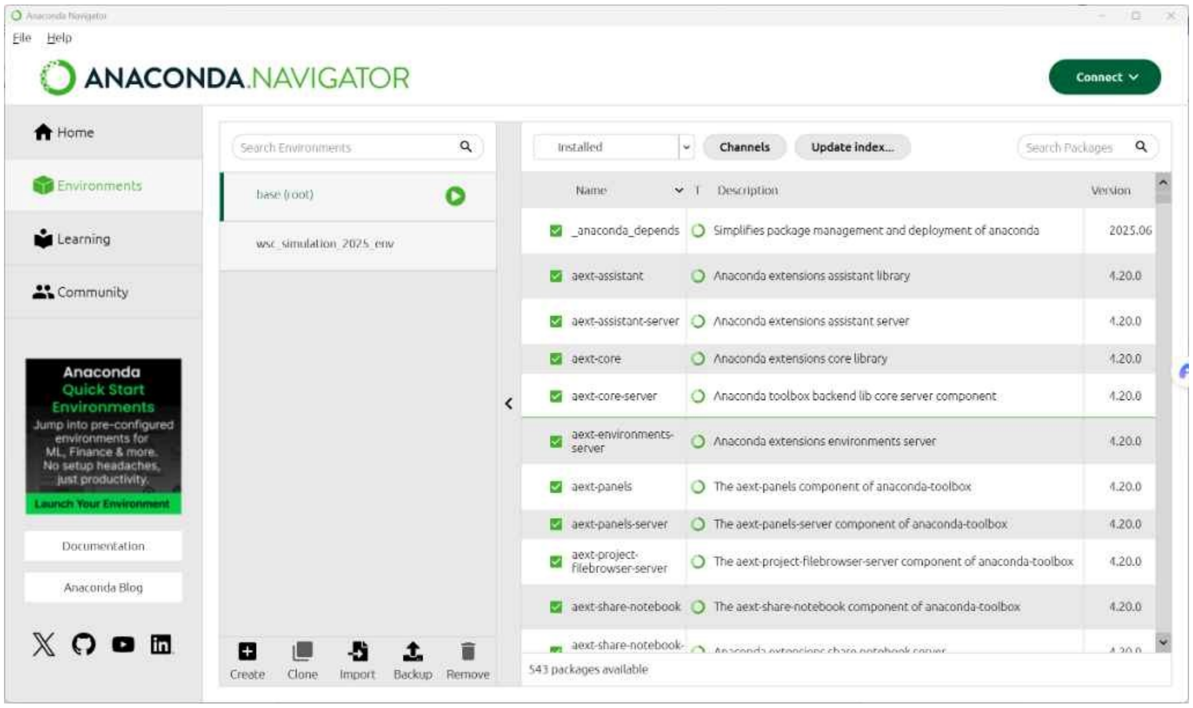
```
  libnpp           nvidia/win-64::libnpp-12.0.2.50-0
  libnpp-dev       nvidia/win-64::libnpp-dev-12.0.2.50-0
  libnvjitlink     nvidia/win-64::libnvjitlink-12.1.105-0
  libnvjitlink-dev nvidia/win-64::libnvjitlink-dev-12.1.105-0
  libnvjpeg        nvidia/win-64::libnvjpeg-12.1.1.14-0
  libnvjpeg-dev    nvidia/win-64::libnvjpeg-dev-12.1.1.14-0
  libuv            pkgs/main/win-64::libuv-1.48.0-h827c3e9_0
  libwebp          pkgs/main/win-64::libwebp-1.3.2-h18467be_1
  markupsafe       pkgs/main/win-64::markupsafe-3.0.2-py39h827c3e9_0
  mpc              pkgs/main/win-64::mpc-1.3.1-h827c3e9_0
  mpfr             pkgs/main/win-64::mpfr-4.2.1-h56c3642_0
  mpmath           pkgs/main/win-64::mpmath-1.3.0-py39haa95532_0
  networkx         pkgs/main/win-64::networkx-3.2.1-py39haa95532_0
  pycparser        pkgs/main/noarch::pycparser-2.21-pyhd3eb1b0_0
  pysocks          pkgs/main/win-64::pysocks-1.7.1-py39haa95532_0
  pytorch          pytorch/win-64::pytorch-2.5.1-py3.9_cuda12.1_cudnn9_0
  pytorch-cuda     pytorch/win-64::pytorch-cuda-12.1-hde6ce7c_6
  pytorch-mutex    pytorch/noarch::pytorch-mutex-1.0-cuda
  pyyaml           pkgs/main/win-64::pyyaml-6.0.2-py39h827c3e9_0
  requests         pkgs/main/win-64::requests-2.32.4-py39haa95532_0
  sympy            pkgs/main/win-64::sympy-1.13.3-py39haa95532_1
  torchaudio       pytorch/win-64::torchaudio-2.5.1-py39_cu121
  torchvision      pytorch/win-64::torchvision-0.20.1-py39_cu121
  typing_extensions pkgs/main/win-64::typing_extensions-4.12.2-py39haa95532_0
  urllib3          pkgs/main/win-64::urllib3-2.5.0-py39haa95532_0
  win_inet_pton    pkgs/main/win-64::win_inet_pton-1.1.0-py39haa95532_0
  yaml             pkgs/main/win-64::yaml-0.2.5-he774522_0


Proceed ([y]/n)? y
```

2.4 Launching Spyder in Your Anaconda Environment

Open the Anaconda Navigator application from your system's Start Menu.



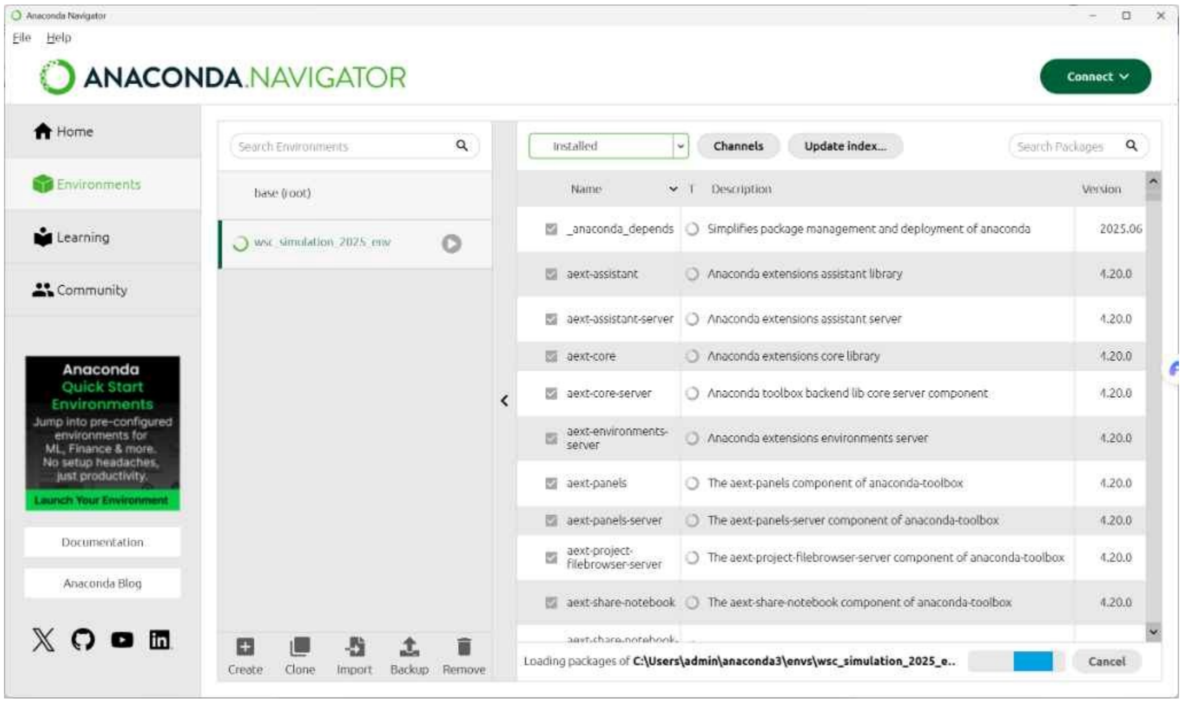
On the "Environments" tab, check your environment and all its packages (like PyTorch and Pandas).



base (root)
wsc_simulation_2025_env

Name	T	Description	Version
_anaconda_depends		Simplifies package management and deployment of anaconda	2025.06
aext-assistant		Anaconda extensions assistant library	4.20.0
aext-assistant-server		Anaconda extensions assistant server	4.20.0
aext-core		Anaconda extensions core library	4.20.0
aext-core-server		Anaconda toolbox backend lib core server component	4.20.0
aext-environments-server		Anaconda extensions environments server	4.20.0
aext-panels		The aext-panels component of anaconda-toolbox	4.20.0
aext-panels-server		The aext-panels-server component of anaconda-toolbox	4.20.0
aext-project-filebrowser-server		The aext-project-filebrowser-server component of anaconda-toolbox	4.20.0
aext-share-notebook		The aext-share-notebook component of anaconda-toolbox	4.20.0

543 packages available



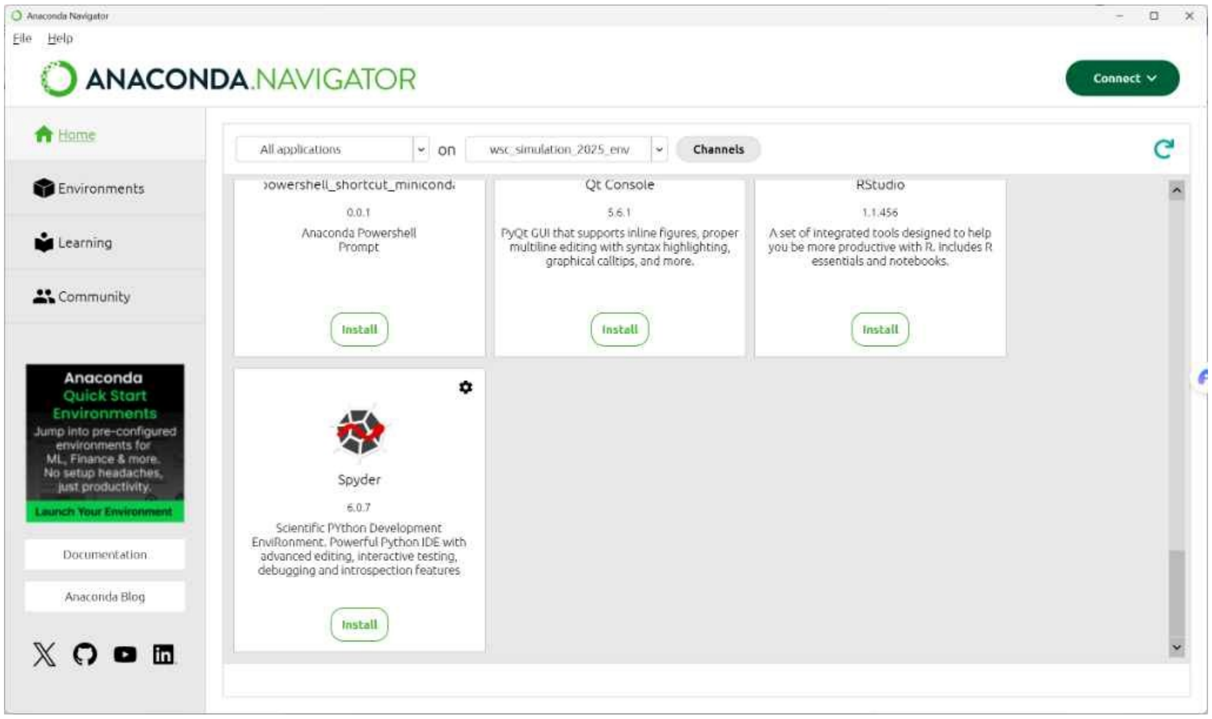
base (root)
wsc_simulation_2025_env

Name	T	Description	Version
_anaconda_depends		Simplifies package management and deployment of anaconda	2025.06
aext-assistant		Anaconda extensions assistant library	4.20.0
aext-assistant-server		Anaconda extensions assistant server	4.20.0
aext-core		Anaconda extensions core library	4.20.0
aext-core-server		Anaconda toolbox backend lib core server component	4.20.0
aext-environments-server		Anaconda extensions environments server	4.20.0
aext-panels		The aext-panels component of anaconda-toolbox	4.20.0
aext-panels-server		The aext-panels-server component of anaconda-toolbox	4.20.0
aext-project-filebrowser-server		The aext-project-filebrowser-server component of anaconda-toolbox	4.20.0
aext-share-notebook		The aext-share-notebook component of anaconda-toolbox	4.20.0

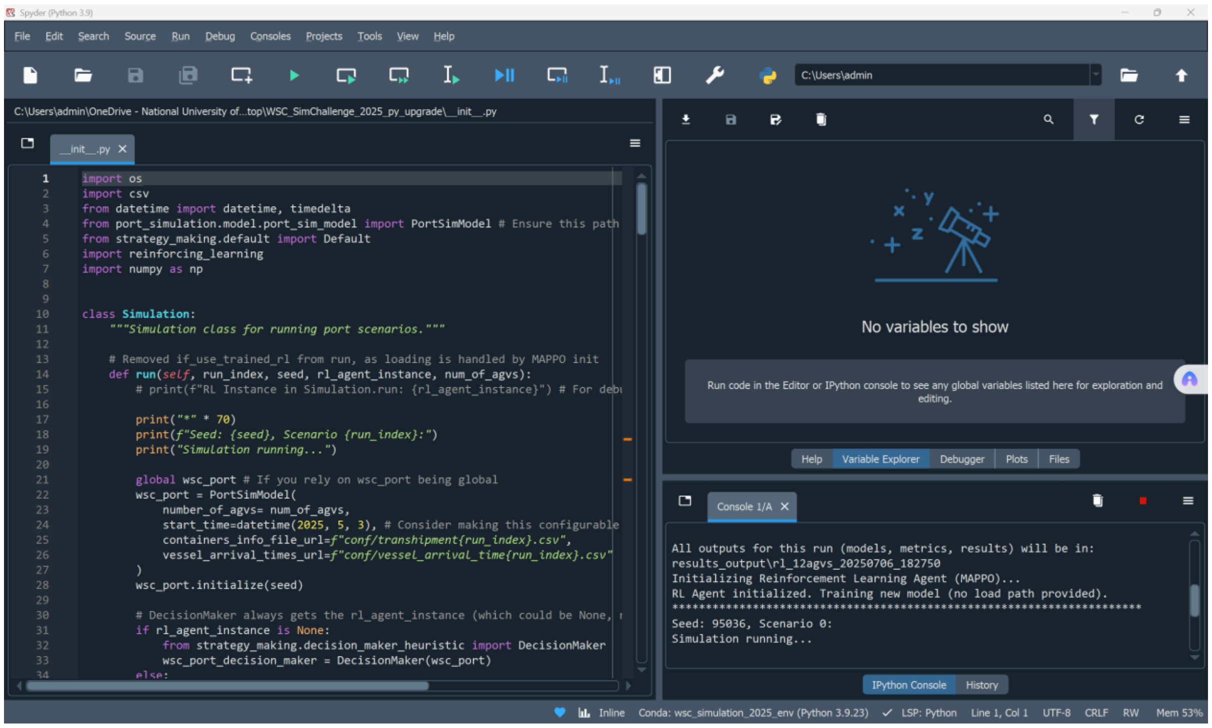
Loading packages of **C:\Users\admin\anaconda3\envs\wsc_simulation_2025_e...**

On the "Home" tab, use the applications dropdown menu to select “**wsc_simulation_2025_env**”. With your environment selected, find the Spyder application tile and click "Install". This installs

Spyder directly into your active environment. After the installation finishes, click the "Launch" button on the Spyder tile. Spyder will now open.


Anaconda Navigator
File Help
ANACONDA.NAVIGATOR
Connect
Home
Environments
Learning
Community
Anaconda Quick Start Environments
Jump into pre-configured environments for ML, Finance & more. No setup headaches, just productivity.
Launch Your Environment
Documentation
Anaconda Blog
All applications
on
wsc_simulation_2025_env
Channels
powershell_shortcut_minicond
0.0.1
Anaconda Powershell Prompt
Install
Qt Console
5.6.1
PyQt GUI that supports inline figures, proper multiline editing with syntax highlighting, graphical calltips, and more.
Install
RStudio
1.1.456
A set of integrated tools designed to help you be more productive with R. Includes R essentials and notebooks.
Install
Spyder
6.0.7
Scientific PYthon Development EnviRonment. Powerful Python IDE with advanced editing, interactive testing, debugging and introspection features
Install

Find _init_.py, and click _init_.py directly or use Spyder to open and run it.

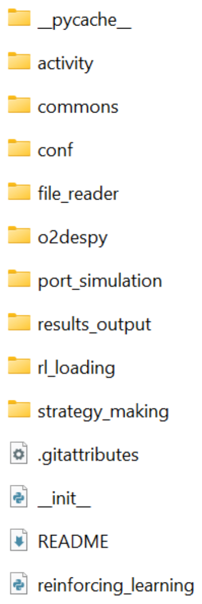

Spyder (Python 3.9)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\admin
C:\Users\admin\OneDrive - National University of...top\WSC_SimChallenge_2025_py_upgrade\__init__.py
__init__.py
No variables to show
Run code in the Editor or IPython console to see any global variables listed here for exploration and editing.
Help
Variable Explorer
Debugger
Plots
Files
Console 1/A
IPython Console
History
Inline
Conda: wsc_simulation_2025_env (Python 3.9.23)
LSP: Python
Line 1, Col 1
UTF-8
CRLF
RW
Mem 53%

```python
import os
import csv
from datetime import datetime, timedelta
from port_simulation.model.port_sim_model import PortSimModel # Ensure this path
from strategy_making.default import Default
import reinforcing_learning
import numpy as np


class Simulation:
    """Simulation class for running port scenarios."""

    # Removed if_use_trained_rl from run, as loading is handled by MAPPO init
    def run(self, run_index, seed, rl_agent_instance, num_of_agvs):
        # print(f"RL Instance in Simulation.run: {rl_agent_instance}") # For debu

        print("*" * 70)
        print(f"Seed: {seed}, Scenario {run_index}:")
        print("Simulation running...")

        global wsc_port # If you rely on wsc_port being global
        wsc_port = PortSimModel(
            number_of_agvs= num_of_agvs,
            start_time=datetime(2025, 5, 3), # Consider making this configurable
            containers_info_file_url=f"conf/transhipment{run_index}.csv",
            vessel_arrival_times_url=f"conf/vessel_arrival_time{run_index}.csv"
        )
        wsc_port.initialize(seed)

        # DecisionMaker always gets the rl_agent_instance (which could be None,
        if rl_agent_instance is None:
            from strategy_making.decision_maker_heuristic import DecisionMaker
            wsc_port_decision_maker = DecisionMaker(wsc_port)
        else:
```

```
All outputs for this run (models, metrics, results) will be in:
results_output\rl_12agvs_20250706_182750
Initializing Reinforcement Learning Agent (MAPPO)...
RL Agent initialized. Training new model (no load path provided).
**********************************************************************
Seed: 95036, Scenario 0:
Simulation running...
```

2.5 Source Code (Python)

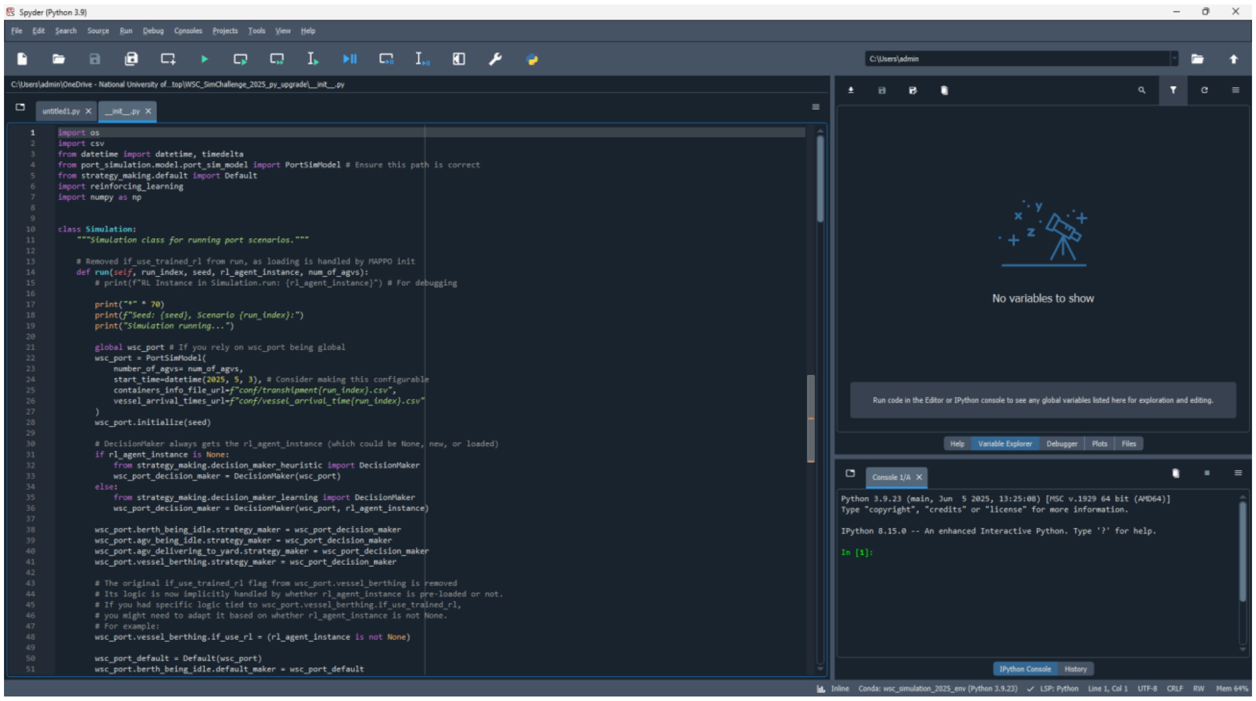
1) After registration and joining in a team, you will receive the zip package of source code by email: “**WSC Simulation Challenge 2025.zip**”.
2) Decompose zip package.
3) Source code structure of ‘**WSC Simulation Challenge 2025**’.



pycache
activity
commons
conf
file_reader
o2despy
port_simulation
results_output
rl_loading
strategy_making
.gitattributes
init
README
reinforcing_learning

Note： conf folder includes scenario files； strategy_making folder includes default, decision_maker_heuristic and decision_maker_learning files (py. Source File)

4) Click _init_ directly or use Spyder to open the project.


Spyder (Python 3.9)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\admin
C:\Users\admin\OneDrive - National University of...top\WSC_SimChallenge_2025_py_upgrade\__init__.py
untitled1.py
__init__.py
No variables to show
Help Variable Explorer Debugger Plots Files
Console 1/A
IPython Console History

```python
import os
import csv
from datetime import datetime, timedelta
from port_simulation.model.port_sim_model import PortSimModel # Ensure this path is correct
from strategy_making.default import Default
import reinforcing_learning
import numpy as np


class Simulation:
    """Simulation class for running port scenarios."""

    # Removed if_use_trained_rl from run, as loading is handled by MAPPO init
    def run(self, run_index, seed, rl_agent_instance, num_of_agvs):
        # print(f"RL Instance in Simulation.run: {rl_agent_instance}") # For debugging

        print("*" * 70)
        print(f"Seed: {seed}, Scenario {run_index}:")
        print("Simulation running...")

        global wsc_port # If you rely on wsc_port being global
        wsc_port = PortSimModel(
            number_of_agvs= num_of_agvs,
            start_time=datetime(2025, 5, 3), # Consider making this configurable
            containers_info_file_url=f"conf/transhipment{run_index}.csv",
            vessel_arrival_times_url=f"conf/vessel_arrival_time{run_index}.csv"
        )
        wsc_port.initialize(seed)

        # DecisionMaker always gets the rl_agent_instance (which could be None, new, or loaded)
        if rl_agent_instance is None:
            from strategy_making.decision_maker_heuristic import DecisionMaker
            wsc_port_decision_maker = DecisionMaker(wsc_port)
        else:
            from strategy_making.decision_maker_learning import DecisionMaker
            wsc_port_decision_maker = DecisionMaker(wsc_port, rl_agent_instance)

        wsc_port.berth_being_idle.strategy_maker = wsc_port_decision_maker
        wsc_port.agv_being_idle.strategy_maker = wsc_port_decision_maker
        wsc_port.agv_delivering_to_yard.strategy_maker = wsc_port_decision_maker
        wsc_port.vessel_berthing.strategy_maker = wsc_port_decision_maker

        # The original if_use_trained_rl flag from wsc_port.vessel_berthing is removed
        # Its logic is now implicitly handled by whether rl_agent_instance is pre-loaded or not.
        # If you had specific logic tied to wsc_port.vessel_berthing.if_use_trained_rl,
        # you might need to adapt it based on whether rl_agent_instance is not None.
        # For example:
        wsc_port.vessel_berthing.if_use_rl = (rl_agent_instance is not None)

        wsc_port_default = Default(wsc_port)
        wsc_port.berth_being_idle.default_maker = wsc_port_default
```

Run code in the Editor or IPython console to see any global variables listed here for exploration and editing.

```
Python 3.9.23 (main, Jun  5 2025, 13:25:08) [MSC v.1929 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 8.15.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]:
```

Inline   Conda: wsc_simulation_2025_env (Python 3.9.23)   ✓ LSP: Python   Line 1, Col 1   UTF-8   CRLF   RW   Mem 64%

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. Figure 2 presents an Entity Relationship Diagram (ERD) that provides an overview of the entities their attributes, and the relationships among them.

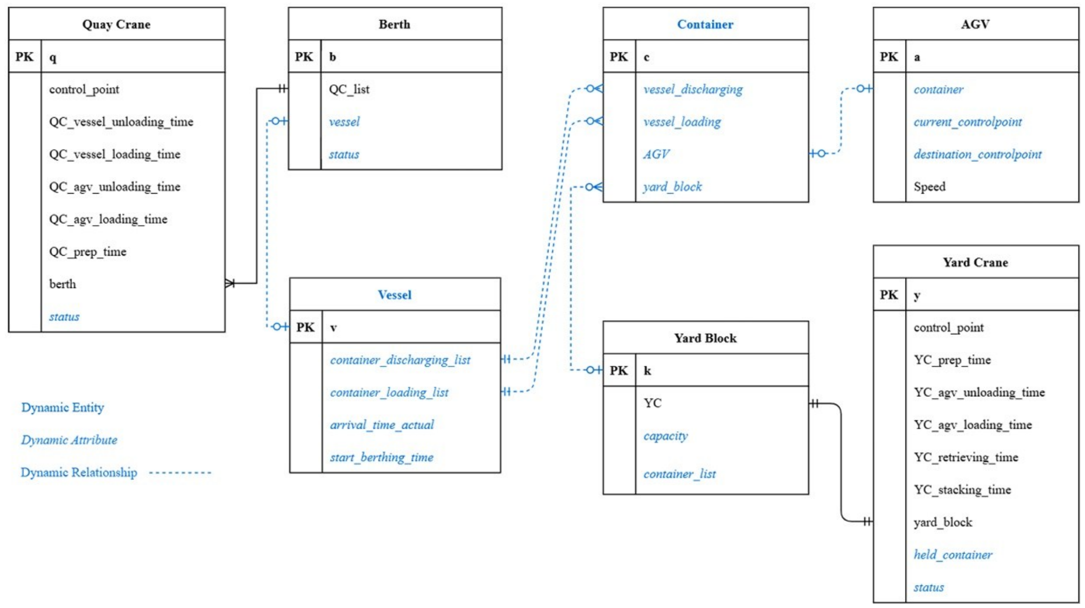


Figure 2 - Entity Relationship Diagram (ERD)

Note:

1) PK: short term for primary key, which uniquely identifies the entity object.
2) Static attribute: entity attributes that do not change overtime. Static attributes are owned by the class itself (in black color).
3) Dynamic attribute (italic): entity attributes that change overtime (in blue color).

3.1.1 Entity Relationship

1. A Berth has one to many Quay Cranes.

   A QC corresponds to one and only one Berth.

2. A Berth has zero to one Vessel.

   A Vessel corresponds to zero to one Berth.

3. A Vessel has zero to many Containers.

   A Container corresponds to one and only one Vessel.

4. A Container corresponds to zero to one AGV.

   An AGV can have zero to one Container.

5. A Container corresponds to zero to one Yard Block.

   A Yard Block can have zero to many Containers.

6. A Yard Block can have one and only one YC.

   A YC corresponds to one and only one Yard Block.

3.1.2 Entity Attributes and Definitions

The attributes of each entity and their definitions are explained in detail below.

Table 1 outlines the attributes of Quay Crane which are all static attributes except 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

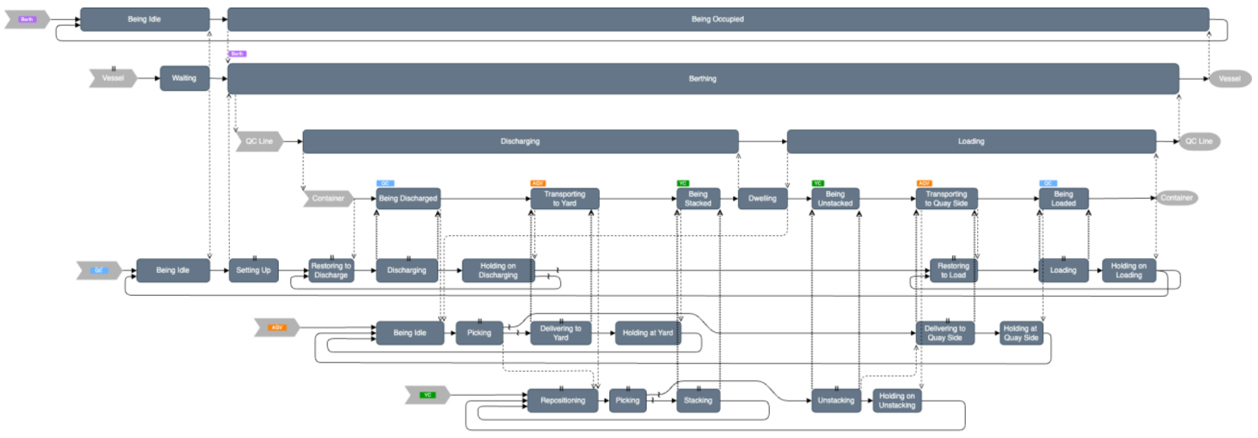


Figure 3 - Event Flow Diagram for the Model

The Event Flow Diagram (EFD), see Figure 3, provides a clear representation of the activity flow, including entities, resource allocation, signal sources, and more. To facilitate better understanding of the EFD, we will break it down and provide detailed explanation of the relationships between the following components:

- 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

3.2.1 Berth and Vessel

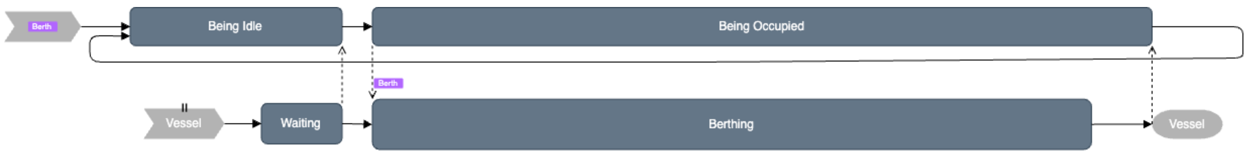


Figure 4 - Event Flow Diagram for Berth and Vessel

When a vessel arrives, it first enters a waiting status, see Figure 4 for more detailed flows. It sends a signal to an idle berth, and once the berth confirms availability, it also sends a signal back to the vessel. At this point, the vessel transitions into the berthing status, while the berth's status changes from idle to being occupied. Upon completion of the berthing process, the vessel departs and notifies the berth of its departure, at which time the berth's status reverts from being occupied back to being idle.

3.2.2 Vessel and QC Line

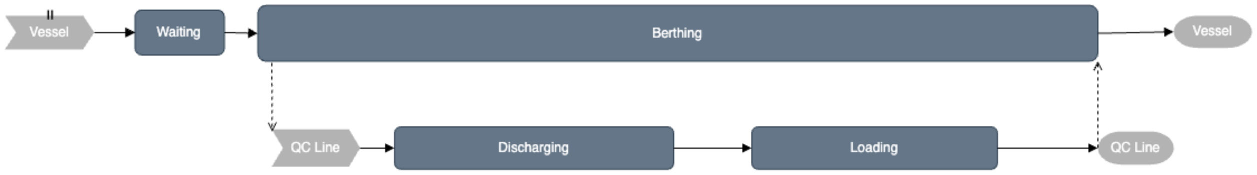


Figure 5 - Event Flow Diagram for Vessel and QC Line

When a vessel arrives, it first enters a waiting activity, before proceeding to the berthing stage. During berthing stage as shown in Figure 5, each quay crane creates a QC Line. A QC line starting with Discharging. After all discharging operations are finished, that QC line transits to Loading. Once all Loading tasks are done, which represents this QC line is ended. When all QC lines are finished, the vessel is notified that all operations have been completed and it can depart.

3.2.3 QC Line and Container

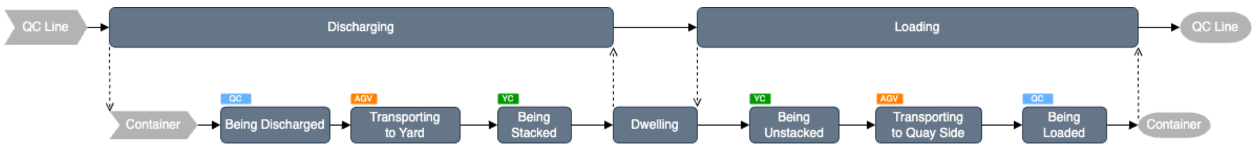


Figure 6 - Event Flow Diagram for QC Line and Container

A QC line initiates with Discharging, then it is followed by Loading (see Figure 6). Upon completion of these phases, the QC line disappears. The QC line processes each container individually. During the Discharging phase, each container undergoes the following steps: Being Discharged, Transporting to Yard, and Being Stacked, until all containers are moved to Dwelling area. At this point, Dwelling 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. 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

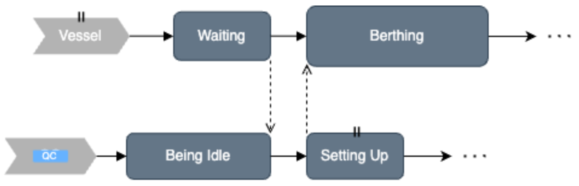


Figure 7 - Event Flow Diagram for Vessel and QC

When a vessel arrives, it firstly enters a waiting activity (refer to Figure 7 for more details). It sends a signal to idle QCs. Once the QCs confirm availability, it also sends a signal back to the vessel. At this point, the vessel transitions into the berthing status, and the QC is allocated to that vessel and starts to operate on it.

3.2.5 Container and QC

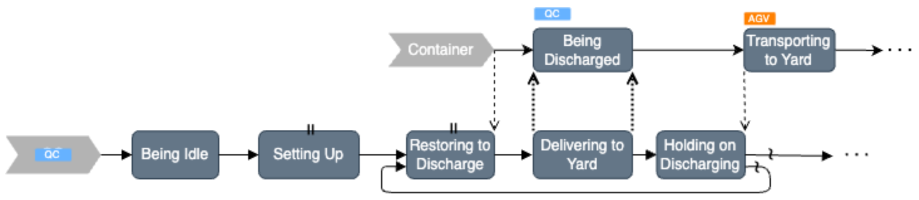


Figure 8(a) - Event Flow Diagram for Container and QC

Every task executed on the QC line for each container involves the joint operation of the QC, AGV, and YC (see Figure 8(a)). Initially QC is idle waiting for a new vessel arrival to be served. When QC is called for discharging containers from vessel, it considered as being occupied and will move to working place to start discharging containers from vessel to 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.

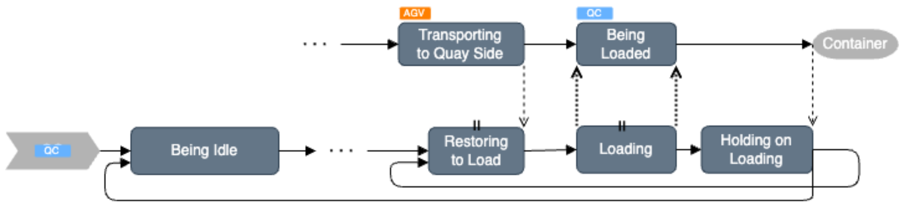


Figure 8(b) - Event Flow Diagram for Container and QC

When QC completes discharging process, similarly there would be a call to QC for loading work from yard side to vessel (refer to Figure 8(b)). QC will move to working position load the containers from AGVs to vessel. When all containers requiring to be transferred are loaded on vessel, the QC is released from its current working status and transition to an idle state, ready for its next call or assignment.

3.2.6 Container and AGV

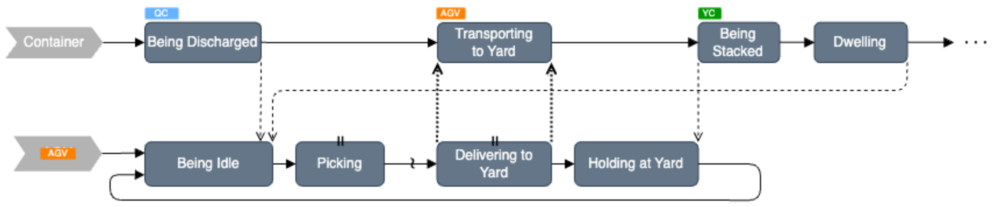


Figure 9(a) - Event Flow Diagram for Container and AGV

Figure 9(a) presents the Event Flow Diagram (EFD) for the container and AGV, illustrating how the QC discharges containers from the vessel and transfers them onto the AGV. The AGVs then transfer the containers from the quay side to the specific storage blocks on the yard side. When AGV arrives the designated storage blocks, it will wait YC to move the container to the yard block. Once container is stacked by YC, the AGV leaves storage block and becomes idle for next call.

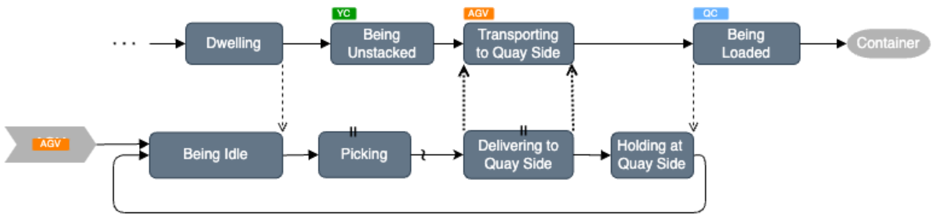


Figure 9(b) - Event Flow Diagram for Container and AGV

For loading operation from yard side to vessel as presented in Figure 9(b), AGV is called by the container that needs to be loaded. The empty AGV will move to appointed yard block. Once arrived at the loading position, the container would be placed on AGV by YC and then AGV will carry container to quay side. After arriving at the loading location, AGV will queue for QC to transfer container from AGV to vessel. Once the container is obtained by QC, the AGV is released and becomes idle for next call.

3.2.7 Container and YC

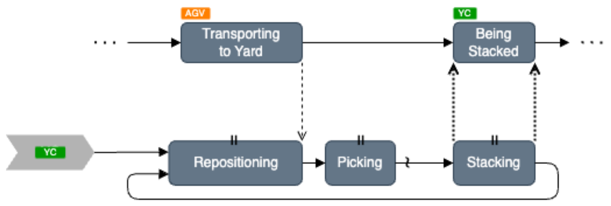


Figure 10(a) – Event Flow Diagram for Container and YC

YC stays idle until called into action when AGVs deliver containers to the yard block (see Figure 10(a)). During the discharging process, YCs are activated to transfer containers from AGVs to designated slots in the storage blocks. Once all containers are transferred to stack, the YC returns to idle.

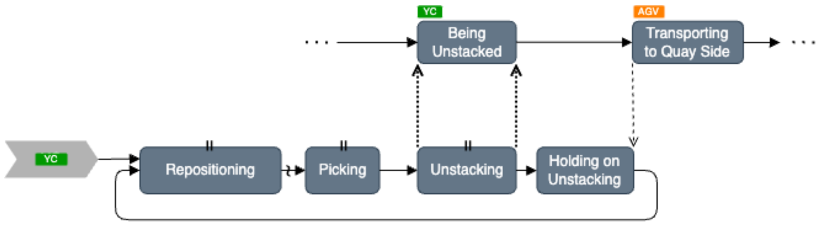


Figure 10(b) - Event Flow Diagram for Container and YC

For loading containers onto a vessel, YC receives the signal from the dwelling containers, then YC unstacks these containers from the specific storage blocks. When an AGV arrives at the pick-up point, YC places a container onto the AGV for delivery to the vessel. After completing all loading containers, YCs return to idle status, ready for the next call as seen in Figure 10(b)

3.2.8 AGV and YC

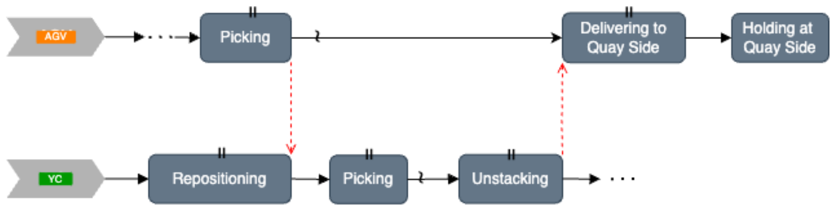


Figure 11 - Event Flow Diagram for AGV and YC

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, see Figure 11 for more details

Decision Makers Analysis and Optimization Guide

In the WSC 2025 port simulation system, decision makers are core components that affect port operation efficiency. The system provides two decision-making methods:

i. **Heuristic Decision Maker** (decision_maker_heuristic.py): Makes decisions based on predefined rules and logic;

ii. **Reinforcement Learning Decision Maker** (decision_maker_learning.py): Makes decisions based on reinforcement learning algorithms.

Both decision makers implement the same core methods but use different decision-making logic. Participants can choose one method or combine both to optimize port operations.

4.1 Heuristic Decision Maker Analysis

4.1.1 Code Structure

The DecisionMaker class in decision_maker_heuristic.py implements the following core methods:

- customeized_allocated_berth: Allocates berths to waiting vessels
- customeized_allocated_agvs: Allocates AGVs to containers
- customeized_determine_yard_block: Determines yard blocks for AGVs

4.1.2 Default Setting

Berth Allocation:

Simple First-In-First-Out (FIFO) strategy, selecting the first idle berth and the first waiting vessel.

AGV Allocation:

Select the idle AGV closest to the container.

Yard Block Allocation:

Select the yard block with available capacity closest to the AGV.

4.1.3 Optimization Directions

Berth Allocation

- Consider vessel priority (based on container count or waiting time)
- Consider berth characteristics (location) and vessel requirements compatibility
- Implement more complex scheduling algorithms, such as Shortest Processing Time (SPT) or Earliest Due Date (EDD)

AGV Allocation

- Consider avoiding AGV concentration in the same area
- Implement task batching to optimize overall AGV path planning
- Consider container priority

Yard Block Allocation

- Consider current utilization and expected future load of yard blocks
- Implement classification storage strategies based on container type and destination
- Consider expected container dwell time

4.2 Reinforcement Learning Decision Maker Analysis

4.2.1 Code Structure

The DecisionMaker class in decision_maker_learning.py implements the same core methods as the heuristic decision maker but uses reinforcement learning models for decision making. Additionally, it implements the get_reward_and_update method to calculate rewards and update the reinforcement learning model.

4.2.2 Default Setting

The state representation for each decision point is as follows:

- **Berth Allocation**: action_state_berth is a binary list of length 4 (number of berths) indicating whether each berth is idle
- **AGV Allocation**: action_state is a binary list of length equal to the number of AGVs indicating whether each AGV is idle
- **Yard Block Allocation**: action_state is a binary list of length 16 indicating whether each yard block has available capacity
- **Vessel Selection**: action_state_vessel is a binary list of length 3 indicating vessels in decision pool

Action Selection

For each decision point, the reinforcement learning model selects actions based on the current state.

Reward Mechanism

The system uses a delayed reward mechanism: When a vessel completes its service and leaves the port, the reward (negative of service time) is calculated and passed to all agents involved in serving that vessel, then the policy is then updated accordingly.

4.2.3 Optimization Directions

Enhanced State Representation

The current state representation is relatively simple, containing only binary availability information. The state representation can be enhanced by adding more environmental information:

i. distance information for AGV allocation
ii. vessel size and container count
iii. current port congestion
iv. expected number of arriving vessels
v. yard block utilization

Reward Function Optimization

The current reward function only considers service time. A more complex reward function can be designed. For example, reward function considering both waiting time and service time. Other possible reward factors: Berth utilization, AGV travel distance, Yard operation efficiency.

Hyperparameter Tuning

In reinforcing_learning.py, the MAPPO algorithm has several tunable hyperparameters. Tunable parameters include:

i. lr: Learning rate, controls the step size of model updates
ii. gamma: Discount factor, controls the importance of future rewards
iii. clip_param: Clipping parameter of the PPO algorithm, controls the magnitude of policy updates

4.3 Conclusion Reinforcement Learning Decision Maker Analysis

By optimizing decision makers, port operation efficiency can be significantly improved, reducing vessel waiting and service times. Participants can choose from the following methods:

1) Optimize heuristic decision logic: Implement more complex rules and algorithms
2) Enhance reinforcement learning state representation: Add more environmental information
3) Optimize reinforcement learning reward function: Design more complex reward mechanisms
4) Adjust MAPPO hyperparameters: Optimize the learning process
5) Combine heuristic and reinforcement learning methods: Create hybrid decision makers

Regardless of the method chosen, the key is to understand the unique characteristics and challenges of port operations, and design decision strategies that can adapt to these characteristics.

Evaluation

5.1 File Submission Format

1) Zip package of **whole program**

   Note: For the 2025 Challenge, participants have significant freedom to innovate. You are welcome to modify or replace the default, heuristic, and learning decision-maker modules, or even build entirely new algorithms from the ground up. Your modifications can extend across the entire project pipeline, from initialization scripts to training loops, and you may also introduce custom hooks or interfaces.

   However, there is one critical constraint: the core simulation logic and its input-output rules must remain intact. Your submission must pass all existing validation checks (e.g., Lines 61-122 in _init_.py). We also strongly recommend that all teams implement similar validation functions for their custom features to ensure the simulation runs successfully.

   For your final submission, please provide a Zip package of the entire program along with a brief note that outlines the changes you made and their rationale.

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

5.2 File Submission Method

Email address for submission: wsc2025SimChallenge@gmail.com

5.3 Evaluation Criteria

The criteria used for evaluation in this simulation challenge are outlined as follows:

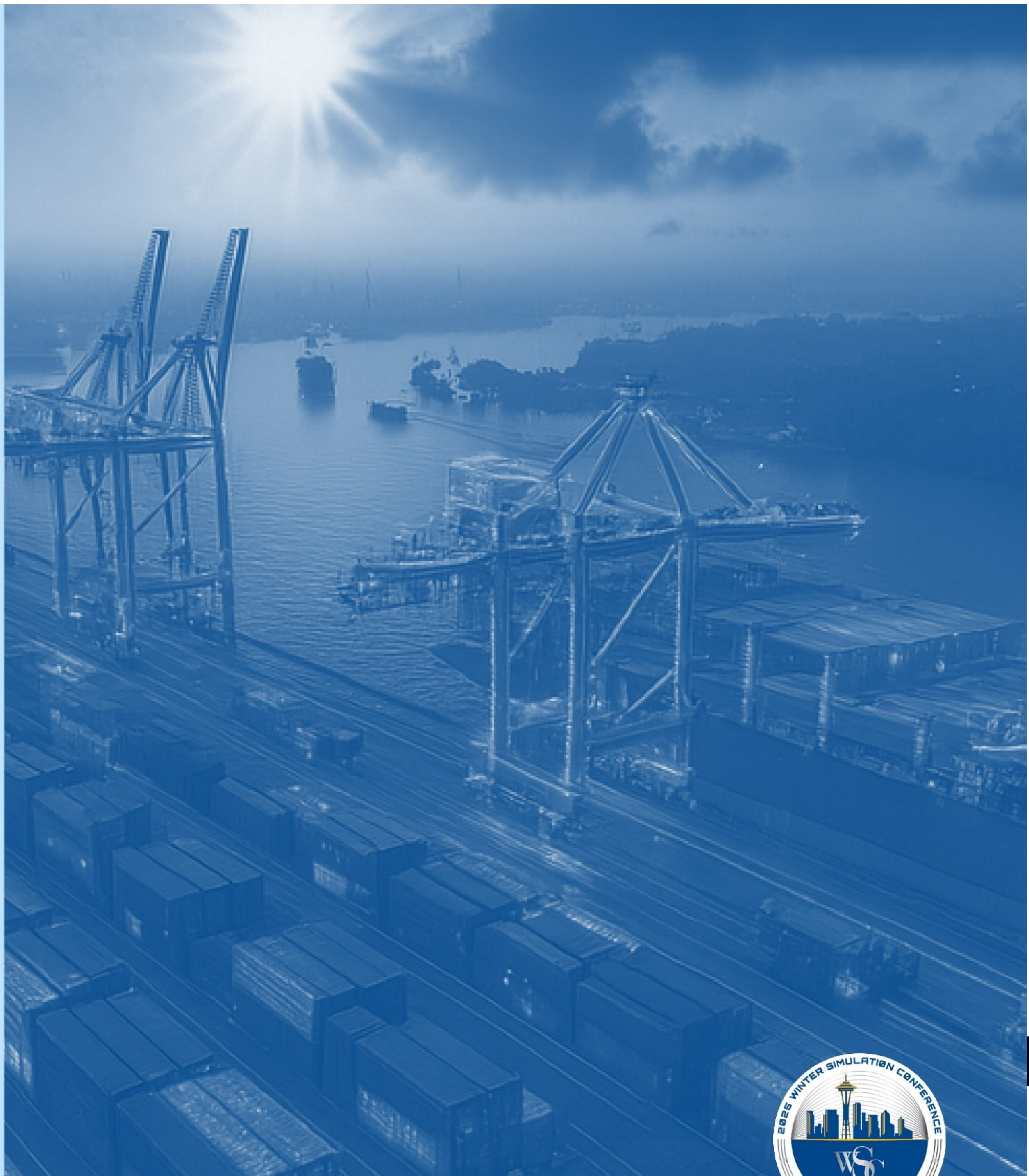
1) The strategy proposed by each competitor will be implemented to simulate both the given scenario and the hidden scenario in the last round.
2) “**Your final result**” from the output is the only criteria to evaluate system performance.
3) In the evaluation stage, multiple random seeds will be used to calculate the average performance for each round. Only codes that run successfully will be eligible for scoring.
4) Weightage and score of each round will be computed as follows:

Weightage Table

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

The full of Each round has a maximum score of 100 points. For the top 10 teams, scores will decrease in increments of 5 based on their ranking (i.e. the first-place team will receive 100 points, the second-place team 95 points, etc.). All teams ranked below the top 10 will receive a fixed score of 50.

nference 2025 Tech Document



2025 WINTER SIMULATION CONFERENCE